Time Series Representation Learning via Cross-Domain Predictive and Contextual Contrasting: Application to Fault Detection

Ibrahim Yousef[1*], Sirish L. Shah[2], and R. Bhushan Gopaluni[1]

[1]Department of Chemical and Biological Engineering, The University of British Columbia, Vancouver, Canada

[2]Department of Chemical and Materials Engineering, University of Alberta, Edmonton, Canada

[*]Corresponding author: iy641@mail.ubc.ca

Abstract

Data-driven methods for fault detection increasingly rely on large historical datasets, yet annotations are costly and time-consuming. As a result, learning approaches that minimize the need for extensive labeling, such as self-supervised learning (SSL), are becoming more popular. Contrastive learning, a subset of SSL, has shown promise in fields like computer vision and natural language processing (NLP), yet its application in fault detection is not fully explored. In this paper, we introduce Cross-Domain Predictive and Contextual Contrasting (CDPCC), a novel contrastive learning framework that integrates temporal and spectral information to capture informative time-frequency features from time series data. CDPCC consists of two key components: cross-domain predictive contrasting, which predicts future embeddings across time and frequency domains, and cross-domain contextual contrasting, which aligns time- and frequency-based representations in a shared latent space. We evaluate CDPCC on fault detection tasks using both simulated and industrial datasets. Our results show that a linear classifier trained on features learned by CDPCC performs comparably to fully supervised models. Moreover, CDPCC proves highly effective in scenarios with limited labeled data, achieving superior performance with only 50% of the labeled data compared to fully supervised training on the entire dataset. The source code is publicly available at `https://github.com/iy641/CDPCC.git`.

Keywords— Time series analysis, Neural networks, Contrastive learning, Self-supervised learning, Fault detection

1 Introduction

The advancement of modern industrial equipment has made industrial processes more complex and integrated, increasing the risk of process faults [1]. Therefore, fault detection technologies have become crucial to ensure safe and efficient operations in modern industrial systems [2]. Fault detection is concerned with determining whether a process is operating normally or abnormally (i.e., experiencing a fault) [3, 4]. Over the past decades, numerous fault detection ideas have been proposed. Early research relied on precise physical models and extensive knowledge bases of inference rules, both of which are challenging to acquire in real industrial settings [5, 6]. However, with the availability of *big data* and cost-effective parallel computing, data-driven fault detection has gained significant attention in academia and industry. Unlike traditional methods, data-driven fault detection methods require minimal or no prior knowledge, offering the potential for high detection accuracy at a relatively low cost [7].

Most modern data-driven fault detection methods are developed in a supervised learning manner that requires all process data for each time interval to be labeled with the corresponding process state. In supervised learning settings, models automatically extract discriminative features and patterns that maximize the separation between different process conditions, such as normal and faulty states [8, 9]. However, the performance of supervised learning models scales upwards with the amount of labeled data, making the availability of labeled data a major bottleneck [10]. To this end, manual labeling is not only time-consuming and costly but also prone to errors and ambiguities in industrial settings, which can lead to misclassifications and reduced model performance. Despite the abundance of available data, the lack of labeled annotations has prompted researchers to seek alternative approaches. Self-supervised learning (SSL), which has recently seen significant success in fields like computer vision and natural language processing (NLP), offers a promising solution [11]. SSL enables the extraction of robust feature representations from unlabeled data by leveraging the inherent properties of the data itself [12]. However, the application of SSL in the context of fault detection remains underexplored.

The idea behind SSL is straightforward: design a task where the model can generate its own supervisory signals without manual annotation and then train the model to solve that task [13]. This process allows the pre-trained model to learn general and transferable features from the input data, which can be applied to various downstream tasks, such as fault detection. Compared to supervised learning, where models are trained on fully labeled data, SSL models can achieve comparable performance with significantly less labeled data [14]. Existing work in SSL can be broadly categorized into two groups: pretext task-based methods and contrast-based methods (i.e., contrastive learning). Pretext task-based methods involve designing auxiliary tasks that leverage the intrinsic structure within the data, enabling the model to generate pseudo-labels and learn meaningful representations. As the model learns to predict these labels, it must recognize and exploit this underlying structure to solve the task successfully. Examples include predicting the degree of rotation of an image [15], solving jigsaw puzzles [16], or colorizing grayscale images [17]. However, the choice of pretext task can limit the generalizability of the learned features. For instance, a model trained to predict image rotations may focus primarily on geometric transformations, potentially overlooking other valuable features such as color and texture [18].

Contrastive learning methods can be viewed as methods that learn through comparison. In contrastive learning, feature representations are learned by comparing different views of the same input against those of other inputs [19]. The underlying intuition is that embeddings of similar inputs (positive pairs) should cluster closely in the representation space, while embeddings of dissimilar inputs (negative pairs) should be far apart [19, 20]. Although contrastive learning has been successfully applied to time series data, existing methods still face notable limitations. First, most existing methods are inspired by experiences in computer vision and NLP domains, which often rely on strong inductive biases, such as transformation- and cropping-invariance [21]. These assumptions do not always hold true for time series data. For example, while cropping an image may retain the underlying object, cropping a time series can change its semantics and distribution. Secondly, existing methods primarily focus on learning instance-level representations that describe the whole input time series [21]. These instance-level representations may not be suitable for tasks that require low-level representations, such as fault detection. Lastly, recent approaches in contrastive learning for time series typically sample contrastive pairs along the temporal axis only, ignoring the spectral axis [22]. As a result, they fail to exploit the time-frequency cross-correlations that are intrinsic to time series data.

To address the limitations of existing methods, we introduce a contrastive learning framework for time series named Cross-Domain Predictive and Contextual Contrasting (CDPCC). CDPCC promotes contrastive representation learning by simultaneously exploring temporal and spectral relationships within time series data. CDPCC consists of two key modules. The first module, cross-domain predictive contrasting, trains the model to predict future embeddings across both domains: it uses a temporal context to predict future spectral embeddings and a spectral context to predict future temporal embeddings. This bidirectional learning approach allows the model to capture useful cross-domain features. The second module, cross-domain contextual contrasting, ensures that the representations derived from the temporal and spectral domains of the same time series sample are closely aligned in the shared latent space while maximizing their separation from representations of other time series samples. This allows the model to learn more discriminative representations on top of the robust features captured by the first module.

The main contributions of this work are summarized as follows:

- We propose CDPCC, a novel contrastive learning framework for time series representation learning. CDPCC samples contrastive pairs along both the temporal and spectral domains, exploiting the time-frequency cross-correlations inherent in time series data.
- We introduce a novel cross-domain predictive contrasting module that learns robust representations by designing a challenging cross-domain prediction task. Additionally, we incorporate a cross-domain contextual contrasting module to further learn discriminative features.
- We conduct comprehensive experiments of our proposed CDPCC framework using simulated and industrial datasets. The experimental results demonstrate that the learned representations are effective for fault detection tasks across supervised learning, semi-supervised learning, and transfer learning settings.

2 Background & Related Work

Data-driven fault detection. Traditionally, data-driven fault detection methods have relied on feature extraction and dimensionality reduction. Feature extraction identifies important local or global patterns, such as statistical measures (e.g., mean, variance, range), which are then used as inputs to a shallow classifier for fault detection [23, 24]. Dimensionality reduction tools project high-dimensional noisy data onto a lower-dimensional space where key information is concentrated [25]. Commonly used techniques for fault detection include principal component analysis (PCA) [26, 27], partial least squares (PLS) [28, 29], and canonical correlation analysis (CCA) [30, 31].

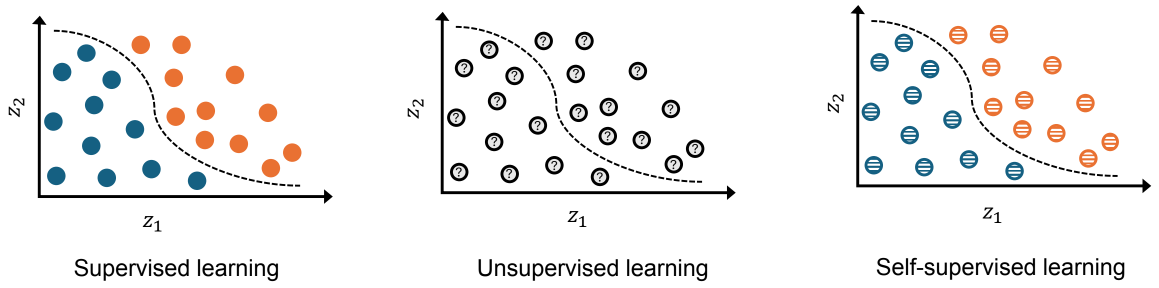

Figure 1: A high-level comparison of three representation learning paradigms. In supervised learning (left), models use class labels (blue and orange circles) to separate samples from different classes. In unsupervised learning (middle), models identify patterns within unlabeled data points (grey circles). Self-supervised learning (right) generates pseudo-labels from the data (striped circles) and trains models to distinguish between them. This figure is inspired by a similar illustration in [37].

Recently, deep learning models have gained significant research interest for industrial fault detection due to their ability to learn meaningful representations directly from data, bypassing the tedious feature extraction process [32, 33]. However, supervised learning, which dominates modern deep learning-based fault detection research, requires large amounts of labeled data [34]. Acquiring such labeled data is challenging in industrial settings due to the harsh operating conditions that make real-time fault recording difficult. In addition, manual labeling is costly, time-consuming, and requires extensive process knowledge. Hence, unsupervised learning has been explored as an alternative [35, 36]. However, unsupervised learning relies on assumptions about data structure rather than fault-specific patterns, making it less effective for accurately identifying and classifying faults. To improve learning from abundant unlabeled data, SSL has emerged as a promising paradigm for fault detection.

Self-supervised learning (SSL). SSL is a newly popular learning paradigm that involves predictive tasks where the supervisory signal comes directly from the data without the need for explicit labels. While SSL has only recently become a popular research focus, its origins can be traced back to ideas that were initially categorized under unsupervised learning [38]. Interestingly, early research efforts that are now considered foundational to SSL— such as DeepCluster [39], Instance Discrimination [40], and context prediction [41]— were originally introduced as unsupervised learning methods. The rebranding of these methods was driven by the growing recognition that labeling these methods as purely unsupervised was somewhat misleading [42]. Unlike traditional unsupervised learning, where the goal is often to discover hidden structures in the data without any supervision, SSL relies on predictive tasks that provide an internal supervisory signal [43]. This signal, while not derived from human annotations, is nonetheless a form of supervision because it guides the training process based on the inherent structure of the data. As a result, SSL has emerged as a distinct learning paradigm, distancing itself from other learning paradigms. Figure 1 illustrates the distinction between SSL and other representation learning paradigms.

Contrastive learning for time series. Contrastive learning, a popular type of SSL, aims to learn useful representations by contrasting positive pairs against negative ones. This approach involves sampling pairs from the data to learn a representation space where positive pairs are pulled together while negative pairs are pushed apart [19, 20]. Following the recent success of contrastive learning in computer vision [44–47] and NLP [48, 49], there has been growing research interest in applying these ideas to time series data.

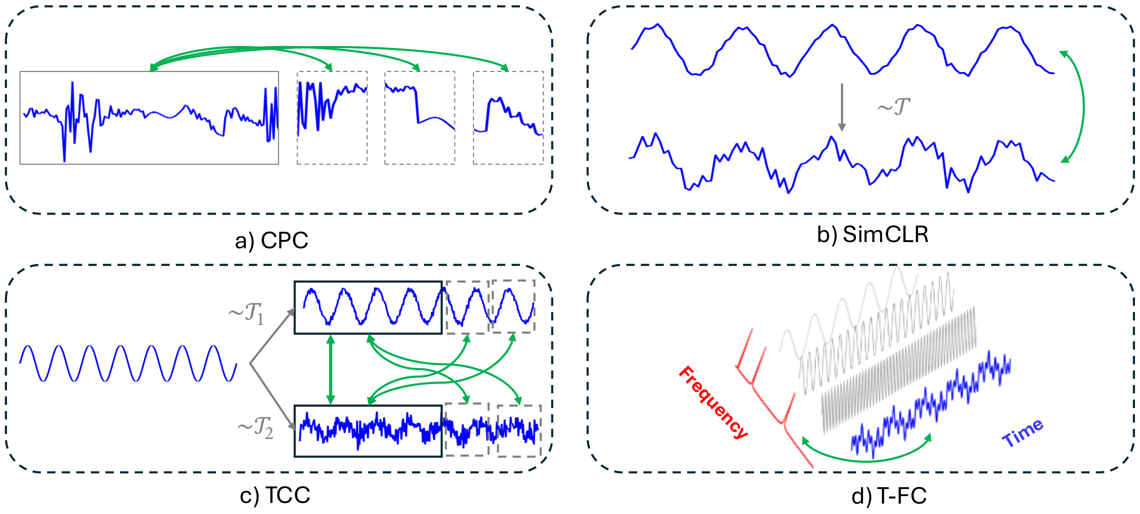

Figure 2: Positive pair selection strategies in state-of-the-art contrastive learning methods for time series. (a) CPC: Positive pairs are created by contrasting past and future segments from the same time series. (b) SimCLR: The original time series and its augmented view form a positive pair. (c) TCC: Positive pairs are formed by contrasting the context of one augmented view with the future time steps of another, as well as between the contexts of the two augmented views. (d) T-FC: Positive pairs consist of the time-domain signal and its corresponding spectral representation.

However, time series data present unique challenges, such as temporal dependencies, irregular sampling, and varying semantic meanings, making it difficult to directly apply methods developed for other domains. As a result, several approaches have emerged as baselines for contrastive learning for time series.

Contrastive Predictive Coding (CPC) is designed to capture temporal features by predicting future data points in the latent space using autoregressive models [18]. It constructs a contrastive loss to maximize mutual information between data from the same time series, preserving temporal dependencies in the learned latent representations. SimCLR uses data augmentations to generate positive pairs from the same data points and applies contrastive loss to maximize the similarity between them [50]. The goal is to learn representations that are invariant to augmentations, as they typically do not change the underlying semantic meaning. Although SimCLR was initially designed for images, [51] adapted it for time series (EEG signals) by developing time series-specific augmentations. Next, Temporal Contextual Contrasting (TCC) combines two tasks: a cross-view prediction task (temporal contrasting) and a contrastive task (contextual contrasting) [52]. The cross-view prediction task helps to learn robust temporal features, while the contrastive task focuses on learning more discriminative features. Time-Frequency Consistency (T-FC) leverages the rich spectral information in time series data by ensuring that time-based and frequency-based representations from the same sample are closer to each other in the latent space than representations from different samples [53]. Overall, the main difference between these methods lies in their strategies for selecting contrastive pairs. Each method employs a different sampling policy to construct positive pairs. Figure 2 summarizes the various positive pair selection strategies adopted by the aforementioned methods.

Rationale for CDPCC. Employing predictive tasks that span different domains (i.e., time and frequency domains) forces the model to learn representations that are consistent and informative in both domains. The time-domain signal and its frequency transform (e.g. Fourier spectrum) reflect the same underlying data from complementary perspectives: the time domain represents temporal order and local variations, while the frequency domain emphasizes periodicities and frequency content. By predicting across these domains, we encourage the model to encode information that allows one representation to be mapped to or predict the other. This cross-domain prediction acts as a form of multi-view self-supervision, akin to learning from two different *modalities* of the same data. As a result, the learned representations become invariant to domain-specific noises and capture variations that are consistent in both time and frequency domains.

Traditional time series contrastive learning methods often rely on intra-domain (i.e., same domain) comparisons. For instance, contrasting different augmented time series samples with each other in the time domain alone. While this can allow the model to learn temporal features, the model might overfit to patterns visible only in that domain. On the other hand, cross-domain (time $\leftrightarrow$ frequency) prediction is fundamentally more challenging and constraining because the model must learn features that map information between how a pattern looks in time and how it looks in frequency. This can allow the model to reveal latent structures that might be hidden when viewed from a single domain. For instance, a transient spike in the time domain corresponds to a broad range of frequencies in the spectrum.

3 Problem Formulation

Given a labeled time series dataset $\mathcal{D} = \{X, Y\} = \{(x_1, y_1), (x_2, y_2), \ldots, (x_N, y_N)\}$ consisting of N samples, where each sample x_i has p channels and L time steps (i.e., $x_i \in \mathbb{R}^{L\times p}$) and is associated with a class label $y_i \in \{0, \ldots, C-1\}$, with C denoting the total number of classes. The goal is to learn a non-linear encoder g_E that maps each x_i to a representation h_i that captures its underlying structure. During the pre-training phase, the encoder is trained in a self-supervised manner by optimizing a contrastive loss function using only the input data X, disregarding the labels Y. After pre-training, g_E and its optimized parameters Θ are employed for downstream tasks. While the proposed method is applicable to all time series data, in this work, we specifically focus on fault detection tasks, where the classification task involves distinguishing between different operating conditions. Specifically, X represents process measurements over time, and Y indicates the process state (e.g., normal or faulty).

The learned representations are evaluated using three common protocols, as illustrated in Figure 3. In linear evaluation, g_E parameters Θ remain unchanged, and a new linear layer is added on top of g_E. Only this new layer is trained using the labeled data $\mathcal{D}$ for the downstream task. In fine-tuning, a new linear layer is also added, but the entire model, including both g_E and the new layer, is retrained with the labeled data $\mathcal{D}$. Transfer learning involves training the pre-trained encoder and a new linear classifier on a different dataset than the one used for self-supervised pre-training.

In this work, we employ two evaluation protocols: linear evaluation on the same dataset (Step 2-I in Figure 3), which assesses the quality of the learned features, and transfer learning (i.e., fine-tuning on a different dataset, Step 2-III in Figure 3), which evaluates the generalizability and transferability of the representations learned during self-supervised pre-training.

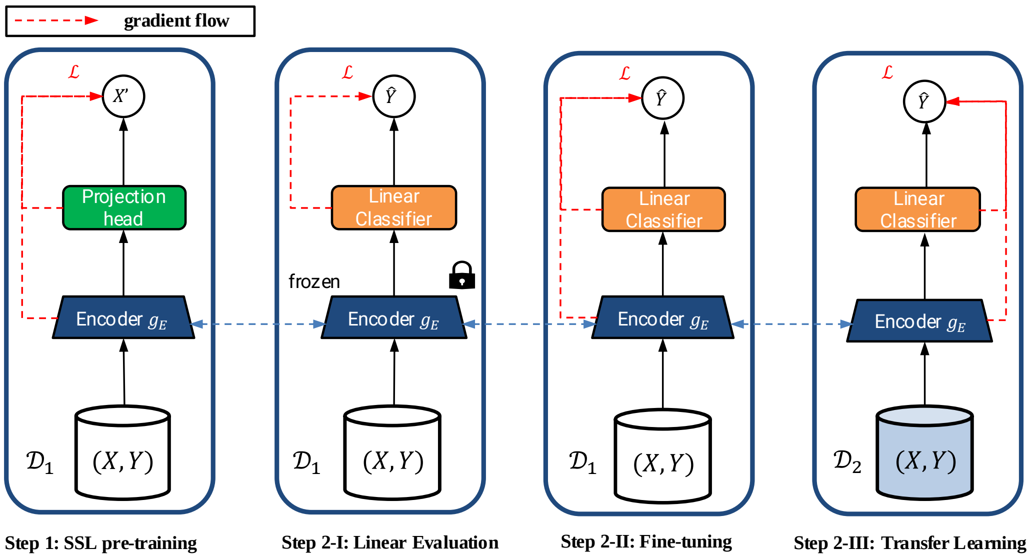

Figure 3: A conceptual illustration of SSL training and evaluation protocols. Step 1: SSL Pre-training: The encoder is trained in a self-supervised manner using a contrastive loss, exploiting the inherent structure of X without relying on class labels Y. A projection head is only used during pre-training but is discarded in evaluation. The goal of SSL is to learn an encoder that can extract meaningful and generalizable features from raw input data. After pre-training, the learned encoder is evaluated on a downstream classification task using labeled data $\mathcal{D}$. Three common evaluation protocols are used: I) Linear evaluation: The pre-trained encoder is frozen, and a linear classifier is added. Only the classifier is trained on the labeled dataset to map the embeddings to class predictions, and the encoder parameters remain unchanged. II) Fine-tuning: A linear classifier is added on top of the pre-trained encoder. Both the classifier and the encoder are updated during supervised training. III) Transfer learning: The pre-trained encoder and the added linear classifier are trained on a different dataset than the one used for self-supervised pre-training.

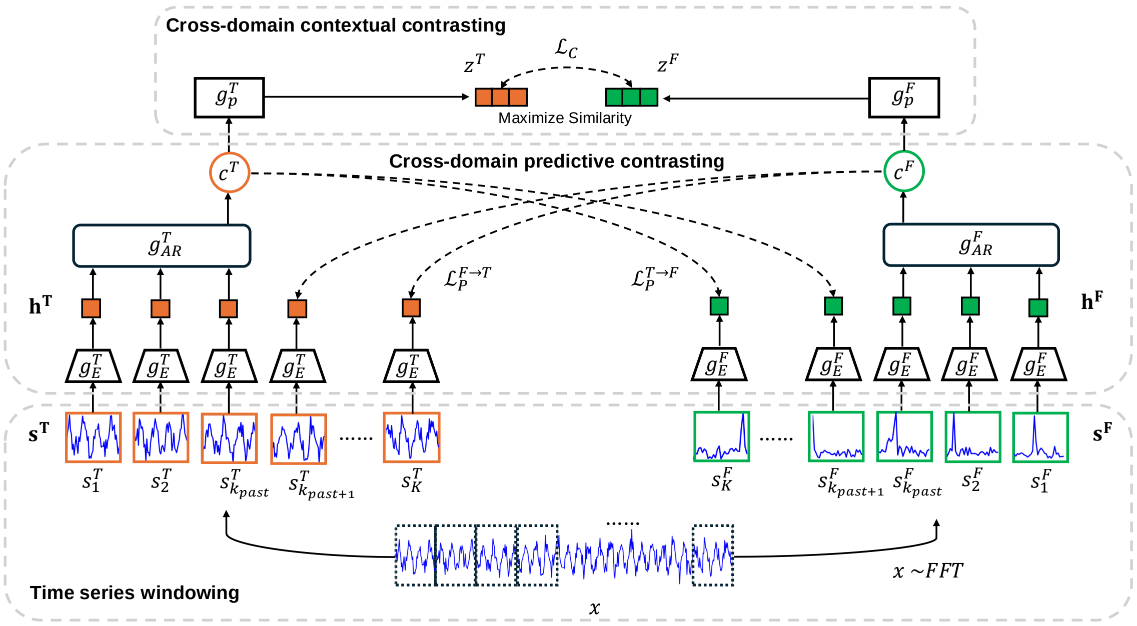

Figure 4: Overall architecture of the proposed CDPCC framework. Our CDPCC model has six components: time- and frequency-domain encoders (g_E^T and g_E^F), autoregressive models (g_{AR}^T and g_{AR}^F), and non-linear projection heads (g_P^T and g_P^F). First, the input time series x is split into K non-overlapping frames $\mathbf{s^T}$, each transformed into its spectral view $\mathbf{s^F}$ via FFT. In the cross-domain predictive contrasting module, time-based and frequency-based representations are produced ($\mathbf{h^T} = g_E^T(\mathbf{s^T})$ and $\mathbf{h^F} = g_E^F(\mathbf{s^F})$). The autoregressive models summarize the dynamics of the first k_{past} frames (here, $k_{\text{past}} = 3$) to generate context vectors, which are then used to predict future embeddings in the other domain. The cross-domain contextual contrasting module further aligns these context vectors to learn discriminative feature representations.

4 Methodology

In this section, we describe the proposed CDPCC framework in detail. The overall architecture of the CDPCC framework is shown in Figure 4. First, we slice the input data into non-overlapping frames of equal size. Each frame is then subjected to a fast Fourier transform (FFT) to extract its frequency components. Next, we introduce a cross-domain predictive contrasting module designed to capture the cross-domain dynamics of the data using an autoregressive model. In this module, the model performs a cross-domain prediction task, where it predicts the future embeddings of one domain using the context (past) of the other domain. We further maximize the agreement between the contexts of the same sample in the cross-domain contextual contrasting module. This contrastive task makes the final representation more discriminative. The following subsections provide a detailed description of each component.

4.1 Time series windowing

The CDPCC framework focuses on learning segment-level representations, where each frame or segment of the time series is encoded separately. In contrast to instance-level representations, which describe the entire time series, segment-level representations capture fine-grained details that might be overlooked in an instance-level approach. This is particularly important for tasks like fault detection, where faults may occur within short time intervals or with small magnitudes. To facilitate segment-level representation learning, we apply time series windowing to the input data. Specifically, a time series x_i is split into K non-overlapping frames, each of size l, resulting in $\mathbf{s_i^T} = \{s_{i,1}^T, s_{i,2}^T, \ldots, s_{i,K}^T\}$, as defined below:

$$s_{i,k}^T = x_i[(k-1)\cdot l : k \cdot l], \quad 1 \leq k \leq K \tag{1}$$

The key objective of contrastive learning methods is to maximize the similarity between different views of the same sample while minimizing its similarity to other samples. Typically, data augmentation techniques are employed to generate these different views. Hence, it is important to employ proper data augmentation techniques that preserve the underlying semantic meaning of the data. However, selecting appropriate augmentations can be challenging and is often highly specific to the task and data at hand. Previous work has shown that the composition of multiple augmentation methods can yield better downstream performance than relying on a single technique [50, 54]. Despite this, time series data presents unique challenges due to its inherent temporal dependencies. For example, common augmentations, such as cropping or shuffling, can disrupt the underlying temporal structures.

In our CDPCC framework, instead of relying on augmentations to create different views, we exploit the inherent relationship between the time and frequency domains, which are distinct yet complementary views of the same data. The transformation between these domains, grounded in signal processing theory, ensures an invariance that remains valid across different time series datasets. Specifically, for each time-domain frame $s_{i,k}^T$, we first apply Hamming windowing to prevent spectral leakage. Then, we perform FFT on the Hamming-windowed frames to obtain the corresponding spectral view $s_{i,k}^F$, as defined below, i.e.:

$$s_{i,k}^F = \mathrm{FFT}(s_{i,k}^T \odot w_h) \tag{2}$$

where w_h is the Hamming window function, which has the form:

$$w_h[n] = 0.54 - 0.46\cos\left(\frac{2\pi n}{l-1}\right), \quad 0 \leq n < l \tag{3}$$

Since FFT produces complex-valued outputs, we use the magnitude spectrum of the first $l/2$ frequency components. This ensures that we capture only the meaningful frequency components while respecting the symmetric property of the FFT output. This transformation results in the spectral representation $\mathbf{s_i^F} = \{s_{i,1}^F, s_{i,2}^F, \ldots, s_{i,K}^F\}$ which serves as a natural alternative view to the time-domain representation $\mathbf{s_i^T}$ without the need for task-specific augmentations. Note that we use the superscript ^{T} to denote the time domain, while the superscript ^{F} denotes the frequency domain.

4.2 Cross-domain predictive contrasting

The cross-domain predictive contrasting module deploys a contrastive loss to capture cross-domain correlations and dynamic features between the time and frequency domains. After constructing the contrastive pairs $\mathbf{s_i^T}$ and $\mathbf{s_i^F}$, we pass each of them to a domain-specific encoder to extract temporal and spectral features. Specifically, the time-domain encoder g_E^T encodes each frame $s_{i,k}^T$ into a higher-dimensional embedding $h_{i,k}^T$ (i.e., $h_{i,k}^T = g_E^T(s_{i,k}^T)$), while the frequency-domain encoder g_E^F transforms each frequency frame into its corresponding embedding $h_{i,k}^F$ (i.e., $h_{i,k}^F = g_E^F(s_{i,k}^F)$). The resulting embeddings for the time domain are denoted as $\mathbf{h_i^T} = \{h_{i,1}^T, h_{i,2}^T, \ldots, h_{i,K}^T\}$, where K is the total number of frames and $h_{i,k}^T \in \mathbb{R}^d$, with d representing the embedding dimension. Similarly, the frequency-domain embeddings are represented as $\mathbf{h_i^F}$, where the superscript ^{F} replaces ^{T} to indicate the frequency domain.

Given the latent representations $\mathbf{h_i^T}$ and $\mathbf{h_i^F}$, we use an autoregressive model g_{AR} to summarize the dynamics of the first k_{past} frames in each domain. The autoregressive model g_{AR} takes in the embeddings up to the $k_{\text{past}}^{\text{th}}$ frame and produces a context vector $c_i = g_{AR}(h_{i,k \leq k_{\text{past}}})$, where $c_i \in \mathbb{R}^m$. The context vector obtained from the time domain c_i^T is then used to predict the future spectral embeddings ($h_{i,k}^F$ for $k > k_{\text{past}}$), and vice versa.

To predict future embeddings, we employ a simple log-bilinear model as proposed in [18]. The log-bilinear model serves as the score function that computes a similarity score between the predicted future embeddings and the actual embeddings. Specifically, the score function evaluates how well the context vector from one domain predicts the future embeddings in the opposite domain. We use one score function for each domain. Formally, the score functions are defined as:

$$f_{i,k}^{T\to F}(c_i^T, h_{i,k}^F) = \exp\left((h_{i,k}^F)^\top \mathbf{W}_k^{T\to F} c_i^T\right), \quad k_{\text{past}} < k \leq K \tag{4}$$

$$f_{i,k}^{F\to T}(c_i^F, h_{i,k}^T) = \exp\left((h_{i,k}^T)^\top \mathbf{W}_k^{F\to T} c_i^F\right), \quad k_{\text{past}} < k \leq K \tag{5}$$

where $\mathbf{W}_k^{T\to F}$ and $\mathbf{W}_k^{F\to T}$ are learnable transformation matrices, each in $\mathbb{R}^{d\times m}$, and are unique to each step k between k_{past} and K.

The key objective of the cross-domain prediction task is to maximize the similarity between the predicted cross-domain embeddings and the true ones of the same sample while minimizing the similarity with the future embeddings of other samples in the batch $\mathcal{D}_B$. Consequently, we compute the prediction losses $\mathcal{L}_P^{T\to F}$ and $\mathcal{L}_P^{F\to T}$ as follows:

$$\mathcal{L}_P^{T\to F} = -\frac{1}{N_B(K-k_{\text{past}})} \sum_{i=1}^{N_B} \sum_{k=k_{\text{past}}+1}^{K} \log\left(\frac{\exp\left((h_{i,k}^F)^\top \mathbf{W}_k^{T\to F} c_i^T\right)}{\sum_{n\in\mathcal{D}_B} \exp\left((h_n^F)^\top \mathbf{W}_k^{T\to F} c_i^T\right)}\right) \tag{6}$$

$$\mathcal{L}_P^{F\to T} = -\frac{1}{N_B(K-k_{\text{past}})} \sum_{i=1}^{N_B} \sum_{k=k_{\text{past}}+1}^{K} \log\left(\frac{\exp\left((h_{i,k}^T)^\top \mathbf{W}_k^{F\to T} c_i^F\right)}{\sum_{n\in\mathcal{D}_B} \exp\left((h_n^T)^\top \mathbf{W}_k^{F\to T} c_i^F\right)}\right) \tag{7}$$

where, N_B denotes the batch size.

4.3 Cross-domain contextual contrasting

The cross-domain contextual module is designed to learn discriminative features by maximizing the alignment of time- and frequency-domain contexts of the same sample in the latent time-frequency space. This module uses a non-linear projection head, denoted as g_p, which maps the context vectors c_i^T and c_i^F from the time and frequency domains into a lower-dimensional space where the contrastive loss is computed. Specifically, the time-domain context vector c_i^T is projected by g_p^T into the lower-dimensional representation z_i^T, while the frequency-domain context vector c_i^F is projected by g_p^F into z_i^F. Prior research suggests that adding projection heads in contrastive learning can reduce computational complexity and improve the generalization of learned features [50, 55].

Given a batch of N_B samples, we obtain two contexts for each sample, one from each domain, resulting in $2N_B$ contexts. The context representation pair (z_i^T, z_i^F) constitutes a positive pair, while the remaining $(2N_B–2)$ context representations from other samples in the batch serve as negative pairs. The goal is to maximize the similarity between the positive pair (the context representations of the same sample) while minimizing the similarity between the negative pairs. The cross-domain contextual contrastive loss, $\mathcal{L}_C$, is defined as follows:

$$\mathcal{L}_C = -\frac{1}{N_B}\sum_{i=1}^{N_B} \log\left(\frac{\exp\left(\text{sim}(z_i^T, z_i^F)/\tau\right)}{\sum_{j=1}^{2N_B} \mathbb{1}_{[i\neq j]} \exp\left(\text{sim}(z_i^T, z_j^F)/\tau\right)}\right) \tag{8}$$

where $\text{sim}(a, b) = a^\top b/(\|a\|\|b\|)$ denotes the dot product between two normalized vectors a and b (i.e., cosine similarity), $\mathbb{1}_{[i\neq j]}$ is an indicator function that equals to 1 when $i \neq j$, and τ is a temperature parameter.

The total CDPCC loss, $\mathcal{L}_T$, combines the two predictive contrasting losses, $\mathcal{L}_P^{T\to F}$ and $\mathcal{L}_P^{F\to T}$, with the contextual contrasting loss, $\mathcal{L}_C$, and is defined as follows:

$$\mathcal{L}_T = \lambda_1\left(\mathcal{L}_P^{T\to F} + \mathcal{L}_P^{F\to T}\right) + \lambda_2\mathcal{L}_C \tag{9}$$

here, λ_1 and λ_2 are fixed scalar hyperparameters that control the relative contributions of each loss term.

5 Experiments

We evaluate the proposed CDPCC framework against seven baseline methods across three diverse datasets. The performance is assessed in the context of fault detection under both supervised learning and transfer learning scenarios. Detailed descriptions of the experimental setups and results are provided in the following subsections.

5.1 Experimental set-up

This subsection outlines the experimental framework used to evaluate the proposed CDPCC framework. This includes a detailed description of the datasets, the baseline methods used for benchmarking, and the training and testing protocols followed.

5.1.1 Datasets

We evaluate our proposed model using three publicly available benchmarks. First, we use the simulated Continuous Stirred Tank Heater (CSTH) dataset to assess the performance in a controlled environment. Next, we test the model on the industrial Arc Loss dataset, which offers a large-scale setting to evaluate the robustness of the CDPCC framework. Additionally, we use the Fault Diagnosis (FD) dataset to examine the transferability of the learned features in a transfer learning setup. Specifically, we use the FD-A dataset (collected under operating condition A) for pre-training and the FD-B dataset (collected under operating condition B) for fine-tuning. Table 1 summarizes key statistics for each dataset.

Continuous stirred tank heater (CSTH) dataset. The CSTH system simulates the dynamic behavior of a nonlinear process where hot water and cold water are mixed and heated by steam in a tank [56]. The system operates under a closed-loop control mechanism to regulate the tank temperature, level, and CW flow. Measurements include temperature, CW flow, and tank level, recorded under normal or faulty

Table 1: Datasets statistics.

Dataset	L	p	N_{train}	N_{val}	N_{test}	C
CSTH	200	3	6300	900	1800	2
Arc Loss	1101	96	2258	323	645	2
FD-A	5120	1	8184	2728	-	3
FD-B	5120	1	128	64	13450	3

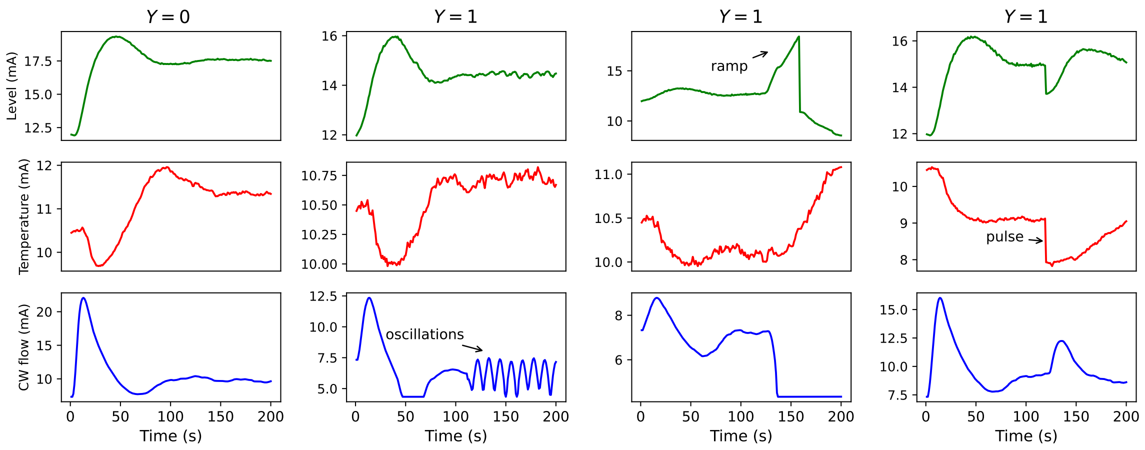

Figure 5: Visualization of normal and faulty operating conditions in the CSTH dataset. The first column, ($Y = 0$), represents normal operation, while the other three columns, ($Y = 1$), correspond to faulty operating conditions. Three different types of faults are considered: i) oscillations in CW flow measurements, ii) a ramp change in tank level measurements, and iii) an abrupt pulse disturbance in temperature measurements.

operating conditions. Faults include abrupt pulse changes in level transmitter signals, random parameter changes in the temperature controller, and sinusoidal noise in the CW flow controller output. The goal is to classify each input signal as either corresponding to normal ($Y = 0$) or faulty ($Y = 1$) conditions. Figure 5 provides a visual comparison between process measurements recorded during a normal operating condition ($Y = 0$) and three faulty operating conditions ($Y = 1$).

Arc Loss dataset. The Arc Loss benchmark dataset originates from an industrial pyrometallurgical smelting process, where high-grade oxidized ore deposits are converted into refined base metals [57]. This process involves multiple stages, including grinding, drying, dehydrating, and smelting in a direct current electric arc furnace (DC EAF). The DC EAF employs plasma arcs to produce the necessary heat but is prone to arc loss faults, which are disruptions that lead to temperature fluctuations and decreased smelting efficiency. The dataset captures operational data to accurately classify whether the data correspond to normal conditions ($Y = 0$) or faulty conditions due to an arc loss event ($Y = 1$). The dataset was preprocessed following the workflow outlined in [58]. Figure 6 illustrates the severity of the arc loss event on the process by providing a side-by-side comparison of process measurements recorded during normal ($Y = 0$) and faulty ($Y = 1$) operating conditions. Compared to stable operating conditions, the faulty regime exhibits increased variability, abrupt changes, and fluctuations in multiple process variables.

Fault diagnosis (FD) dataset. The FD dataset was collected from an electromechanical drive system

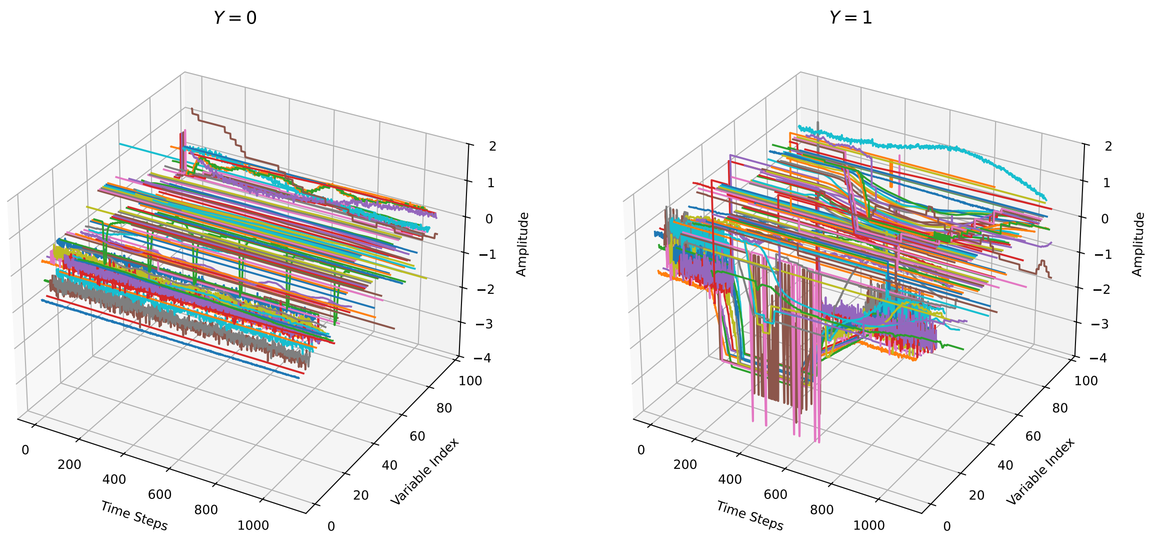

Figure 6: Visualization of normal and arc loss operating conditions in the Arc Loss dataset. The left-hand side figure ($Y = 0$) represents normal operation, where process variables remain relatively stable. The right-hand side figure ($Y = 1$) corresponds to faulty operation due to an arc loss event.

designed to monitor the condition of rolling bearings and detect failures [59]. The dataset includes three classes: undamaged ($Y = 0$), inner damaged ($Y = 1$), and outer damaged ($Y = 2$) bearings. Figure 7 provides a visual comparison of time-domain vibration signals recorded under the three conditions. Data were collected under four different conditions (A, B, C, and D), each representing a distinct domain. We use the FD-A and FD-B subsets to evaluate the transferability of the learned features. Specifically, we pre-train our models on the FD-A dataset and then fine-tune and test them using the FD-B dataset. To formulate a realistic transfer learning scenario, the fine-tuning set (FD-B) is limited to a small subset of 128 samples only.

5.1.2 Baselines

We evaluate our proposed model against seven baselines. This includes four state-of-the-art time series contrastive learning methods: **SimCLR**, **CPC**, **TCC**, and **T-FC**. We also include a non-deep learning model, **DTW+1NN**, a similarity measure approach that sets a baseline performance level. To evaluate the impact of pre-training, we consider two additional scenarios: (i) **Random Initialization (RI)**: training a linear classifier on top of randomly initialized and frozen encoders g_E^T and g_E^F; and (ii) **Supervised**: training the CDPCC framework in a fully supervised manner. Both use the same architecture as the CDPCC framework employed for contrastive learning. We focus our comparison on time series contrastive learning frameworks rather than traditional fault detection methods, as the training protocols for these approaches differ fundamentally. We refer interested readers to our previous work [9, 58], where traditional fault detection methods are evaluated on the CSTH and Arc Loss datasets.

5.1.3 Technical details

We employ a hold-out strategy for model evaluation. We conduct a random search over a predefined search space to identify a well-performing model configuration. The best-performing models in the validation phase

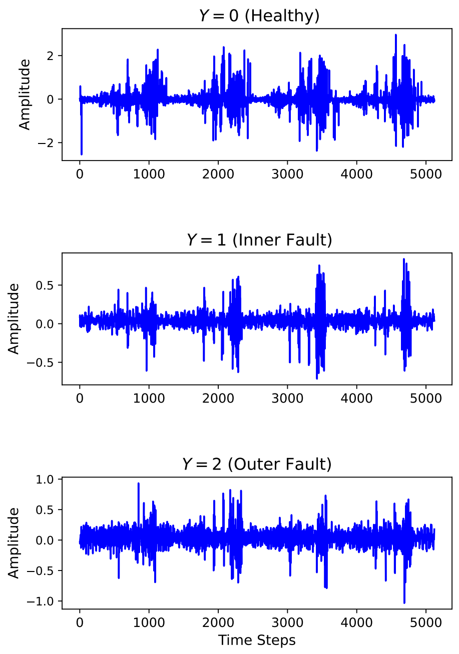

Figure 7: Visualization of time-domain vibration signals in the FD dataset. Each plot represents a vibration signal recorded under different bearing conditions: (top) healthy ($Y = 0$), (middle) inner fault ($Y = 1$), and (bottom) outer fault ($Y = 2$).

are then evaluated on the testing set, with results reported on this final set. We use a 3-block 1D convolutional architecture followed by a non-linear projection layer for both g_E^T and g_E^F. For simplicity, we opted for a standard LSTM architecture with two layers for g_{AR}^T and g_{AR}^F. The projection heads g_P^T and g_P^F are 2-layer fully connected networks without parameter sharing. Readers can find the hyperparameter settings for each dataset in the respective configuration files (`/config_files/*`) in the paper GitHub repository.

The models are trained for 100 epochs, with early stopping based on validation performance. We set $\tau = 0.2$, $\lambda_1 = 0.7$, and $\lambda_2 = 0.3$ in the loss function. Results are reported as the mean and standard deviation across five independent runs using the same data split. The standard deviation for DTW+1NN results is zero, as the model is deterministic.

5.2 Results

This subsection presents the results for downstream fault detection performance across all datasets for our proposed CDPCC and baseline methods. First, we conduct a linear evaluation experiment to evaluate the quality of the learned representations in our proposed approach compared to those learned using baseline approaches. We further examine the transferability of the learned features by conducting a transfer learning experiment.

5.2.1 Linear evaluation of learned representations

To evaluate the performance of our CDPCC model, we employ a linear evaluation protocol. In this evaluation, all parameters of the pre-trained model are frozen, and a new linear layer is added to map the representations to predictions. This linear layer is then trained on the labeled training set.

Five evaluation metrics are used to comprehensively evaluate the performance. Accuracy is calculated as the ratio of correctly predicted samples to the total number of testing samples. Precision (i.e., positive predictive value (PPV)) measures the percentage of accurately predicted faulty samples out of all faulty predictions, while recall (i.e., true positive rate (TPR)) quantifies the proportion of correctly predicted faulty samples compared to the total number of faulty samples. The false positive rate (FPR) represents the ratio of false positive predictions to the total number of normal samples. To maintain a consistent notation where greater values indicate better performance, we instead report (1 - FPR). Next, the F_1 score represents the harmonic mean of precision and recall. The comparison is performed using the CSTH and Arc Loss datasets. The experimental results are summarized in Table 2.

The linear evaluation results in Table 2 demonstrate that CDPCC outperforms state-of-the-art time series contrastive learning methods across CSTH and Arc Loss datasets. It ranks highest on the Arc Loss dataset and second on the CSTH dataset, with results close to the supervised approach. Moreover, CDPCC consistently outperforms the DTW+1NN and RI baselines by a significant margin. This indicates that contrastive pre-training allows the model to learn informative feature representations that are useful for downstream tasks.

Next, CDPCC achieves superior performance relative to instance-level contrastive learning methods (e.g., SimCLR and T-FC). Unlike models that encode an entire time series into a single embedding, CDPCC preserves localized temporal variations, which are critical for fault detection tasks. The superior performance of segment-level contrastive learning methods (e.g., CDPCC, CPC, and TCC) relative to instance-level contrastive learning methods suggests that local temporal features are more valuable than global features in time series for fault detection tasks.

Despite CPC, TCC, and CDPCC use predictive contrastive learning (i.e., predicting future embeddings from the past), CDPCC achieves better performance. CPC and TCC predict future embeddings within the same domain (time domain), while CDPCC performs cross-domain prediction between the time and frequency domains. This forces the model to learn consistent and informative representations in both domains. Furthermore, CDPCC surpasses T-FC, which only aligns time and frequency representations without explicitly incorporating predictive objectives. The predictive contrastive objective in CDPCC enhances feature robustness, leading to better fault detection performance.

Finally, SimCLR relies on manual augmentations to generate contrastive pairs. These augmentations introduce arbitrary transformations that may not always preserve the underlying time series semantics, leading to degraded representations. In contrast, CDPCC leverages the natural transformation between time and frequency domains to generate contrastive pairs, which makes CDPCC more robust and eliminates the need for manual augmentation tuning.

5.2.2 Transfer learning experiment

We conduct a transfer learning experiment to evaluate the transferability of the features learned by our CDPCC model. The evaluation is performed using the FD-A and FD-B datasets in a transfer learning setting. Specifically, we pre-train the models on the FD-A dataset (source dataset) and then fine-tune and test them

Table 2: Performance Comparison of CDPCC and baseline approaches on CSTH and Arc Loss datasets. Bold values indicate the best performance in each metric.

Model	Accuracy (%)	PPV (%)	TPR (%)	1-FPR (%)	F_1 (%)
CSTH					
DTW+1NN	81.17 ± 0.00	96.38 ± 0.00	64.89 ± 0.00	97.55 ± 0.00	77.56 ± 0.00
RI	87.72 ± 0.74	90.71 ± 1.61	82.93 ± 1.72	92.13 ± 1.59	86.62 ± 0.83
Supervised	**99.55 ± 0.18**	**99.71 ± 0.25**	**99.40 ± 0.23**	**99.71 ± 0.25**	**99.55 ± 0.18**
SimCLR	82.42 ± 2.40	86.74 ± 2.72	76.37 ± 4.11	88.18 ± 2.67	81.16 ± 2.76
CPC	97.32 ± 0.60	98.69 ± 1.10	95.88 ± 0.84	98.71 ± 1.10	97.26 ± 0.60
TCC	97.25 ± 0.43	98.04 ± 0.74	96.08 ± 0.78	98.06 ± 0.74	97.05 ± 0.46
T-FC	95.59 ± 0.65	98.79 ± 0.22	92.07 ± 1.20	98.86 ± 0.19	95.31 ± 0.73
CDPCC	99.29 ± 0.18	99.53 ± 0.23	99.04 ± 0.30	99.53 ± 0.23	99.29 ± 0.18
Arc Loss					
DTW+1NN	64.96 ± 0.00	65.52 ± 0.00	64.31 ± 0.00	65.63 ± 0.00	64.91 ± 0.00
RI	68.13 ± 1.50	65.47 ± 1.40	78.57 ± 3.97	57.41 ± 3.88	71.37 ± 1.69
Supervised	75. 62 ± 0.60	71.86 ± 1.06	85.14 ± 3.06	65.93 ± 2.81	77.89 ± 0.88
SimCLR	66.28 ± 1.51	62.17 ± 0.85	83.93 ± 3.27	47.84 ± 1.42	71.41 ± 1.65
CPC	69.82 ± 1.12	69.13 ± 1.54	76.56 ± 7.45	64.76 ± 6.08	72.41 ± 2.45
TCC	74.88 ± 0.47	71.19 ± 2.32	**85.15 ± 5.82**	64.39 ± 6.21	77.35 ± 1.18
T-FC	69.29 ± 0.90	67.13 ± 0.55	77.79 ± 1.24	61.07 ± 0.80	72.06 ± 0.76
CDPCC	**75.88 ± 0.67**	**72.95 ± 1.64**	83.22 ± 3.68	**68.39 ± 3.92**	**77.96 ± 0.94**

on the FD-B dataset (target dataset). In this comparison, we exclude the DTW+1NN and random initialization models. The performance is evaluated using two metrics: accuracy and macro-averaged F_1 score (MF_1).

As reported in Table 3, contrastive learning methods outperform the supervised approach in the transfer learning experiment. This is because contrastive learning methods learn more generalizable representations compared to supervised learning, which tends to extract dataset-specific features. Moreover, CDPCC is the best-performing model among all contrastive learning methods.

Notably, T-FC and CDPCC achieve the highest performance, showing that incorporating time and frequency domain information improves transferability (generalizability). Unlike other contrastive learning methods, which construct positive pairs based only on the temporal axis, T-FC and CDPCC promote time-frequency consistency, ensuring that representations remain invariant across datasets. While T-FC aligns time and frequency representations, CDPCC further improves cross-domain learning by introducing a predictive contrastive task between the time and frequency domains. This forces the model to learn representations that are not only aligned but also predictive across domains, leading to superior generalization.

Table 3: Transfer learning experiment results. Results are reported on FD-B dataset. Bold values indicate the best performance in each column.

Model	Accuracy (%)	MF_1 (%)
Supervised	60.87 ± 4.37	56.77 ± 6.42
SimCLR	54.39 ± 5.62	57.99 ± 12.86
CPC	79.33 ± 1.33	84.63 ± 1.01
TCC	84.97 ± 1.14	88.98 ± 0.83
T-FC	89.34 ± 3.79	91.62 ± 8.26
CDPCC	**89.86 ± 0.52**	**92.55 ± 0.38**

5.3 Analysis

In this subsection, we evaluate the performance of the CDPCC model under various conditions. We first examine its effectiveness with limited labeled data, then analyze the impact of key hyperparameters, and finally explore the contributions of different model components.

5.3.1 Few-labeled data scenarios

In this section, we evaluate the effectiveness of the CDPCC model in scenarios where only a small amount of labeled data is available. We compare the performance of two classifiers: one trained directly on raw data features and another trained on features extracted using the proposed CDPCC model. Both classifiers are trained on varying amounts of labeled data, specifically 1%, 5%, 10%, 25%, 50%, 75%, and 100% of the available training set. This experiment uses a simple 3-block 1D CNN model as the classifier.

Supervised Baseline. First, we examine the performance of the supervised 1D-CNN classifier across different sizes of labeled data for both the CSTH and Arc Loss datasets. On the CSTH dataset, the classifier achieves 79.6% accuracy with just 1% of labeled data, improving to 98.1% when the entire dataset is used. On the Arc Loss dataset, the classifier starts at 65.5% accuracy with 1% labeled data and reaches 72.5% with the entire dataset. These results are shown in Figure 8 (blue curve).

CDPCC Model. Next, we explore whether the CDPCC model improves data efficiency compared to the supervised baseline. We follow the same training approach as before, but this time, the 1D-CNN is trained on features extracted by the CDPCC model rather than directly on raw data. The feature extractor from the CDPCC model remains fixed during the training process, and the 1D-CNN classifier is trained until convergence. The results are presented in Figure 8 (orange curve).

Training the 1D-CNN classifier on CDPCC-extracted features results in noticeable improvements in accuracy. For instance, with just 1% of labeled data, the CDPCC-based classifier achieves 86.2% accuracy on the CSTH dataset and 67.2% on the Arc Loss dataset, representing a 6.6% and 1.7% improvement over the baseline, respectively. Even with the entire dataset, the CDPCC-based classifier continues to outperform the supervised baseline. Furthermore, the CDPCC model demonstrates better data efficiency; for example, with 50% of the labeled data, it surpasses the supervised baseline classifier trained on the full dataset for both CSTH and Arc Loss datasets. Similarly, with only 1% of labeled data, the CDPCC-based classifier achieves better performance than the baseline using 5% of the labels (i.e., a 5× gain in data efficiency).

5.3.2 Sensitivity analysis

Here, we perform sensitivity analysis experiments to examine the influence of two key hyperparameters: the total number of frames, K, and the number of predicted future frames, $(K - k_{past})$. The parameter K is related to the time series windowing module, while k_{past} is critical in the cross-domain predictive contrasting module.

First, we evaluate model performance on the CSTH and Arc Loss datasets by varying the total number of frames K from 5 to 15. A smaller K results in larger frames, meaning each frame contains more time steps. The left-hand side of Figure 9 shows how K affects prediction accuracy. As observed, using a higher number of frames can negatively impact accuracy, likely due to the reduced information in each frame,

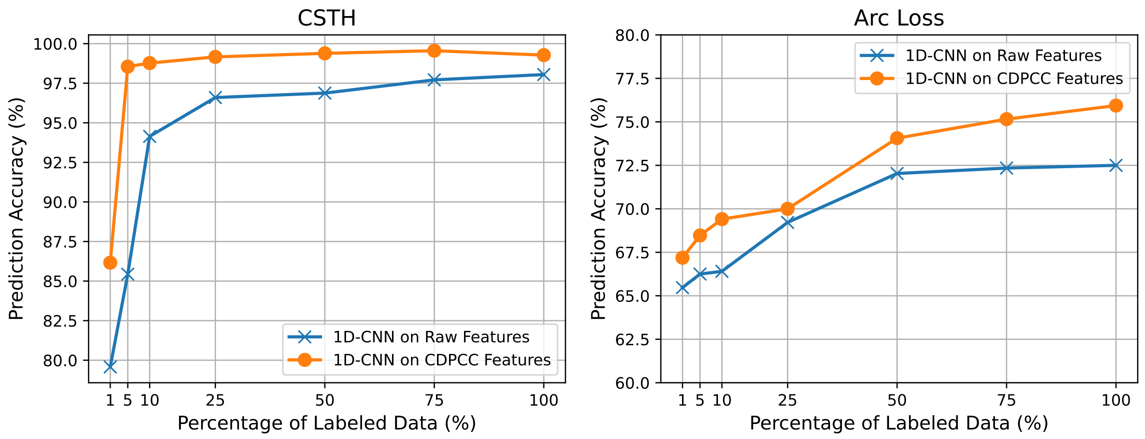

Figure 8: Comparison of prediction accuracy between 1D-CNN models trained on raw features and CDPCC features across the CSTH and Arc Loss datasets on varying amounts of labeled data.

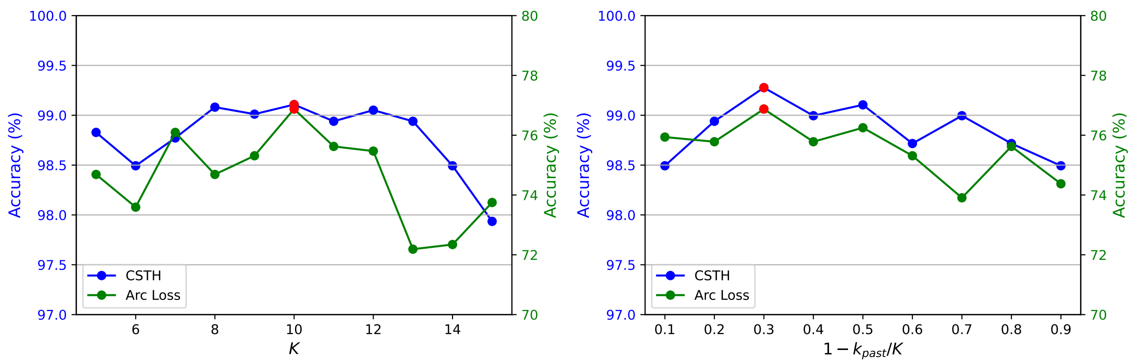

Figure 9: Sensitivity analysis experimental results on CSTH (blue) and Arc Loss (green) datasets. Left: The impact of the total number of frames K on prediction performance. Right: The effect of the number of future frames on prediction performance. Red dots highlight the highest accuracy values for each dataset.

leading to potential spectral leakage. We find that $K = 10$ yields the best results; hence, we use this value in all subsequent experiments.

Next, the right-hand side of Figure 9 shows the impact of k_{past} on performance, with the x-axis representing the ratio of future frames to the total number of frames, $(1 - k_{past}/K)$. The results indicate that increasing this percentage generally improves performance. However, predicting too large a percentage can negatively affect performance, as it reduces the amount of past data available to train the autoregressive model. We find that predicting 30% of the total frames provides optimal performance, and therefore, we set $(1 - k_{past}/K)$ to 30% in our experiments (i.e, $k_{past} = 0.7K$).

5.3.3 Ablation study

To examine the contribution of each component in CDPCC, we conduct an ablation study on the CSTH dataset. We compare the full CDPCC model with three variations: (i) w/o predictive contrasting: removes the cross-domain predictive contrasting losses ($\mathcal{L}_P^{T\to F}$ and $\mathcal{L}_P^{F\to T}$), (ii) w/o contextual contrasting: removes the cross-domain contextual contrasting loss ($\mathcal{L}_C$), and (iii) w/o projection layers: removes the projection layers (g_P^T and g_P^F), computing the contrastive loss directly on the contexts (c^T and c^F) rather than the projections (z^T and z^F). Table 4 shows the results. The results indicate that each component is crucial for the CDPCC, as removing any of them leads to a noticeable decrease in accuracy.

In addition, we evaluate the choice of the autoregressive model architecture by replacing the LSTM with GRU and Transformer models of similar parameter sizes. Both replacements result in a significant drop in accuracy, suggesting that LSTM is a better architecture choice for CDPCC.

Table 4: Ablation results on the CSTH dataset

	Avg. Accuracy
CDPCC	**99.29%**
w/o Predictive Contrasting ($\lambda_1 = 0$)	97.70% (-1.59%)
w/o Contextual Contrasting ($\lambda_2 = 0$)	98.45% (-0.84%)
w/o Projection Layers (g_P^T and g_P^F)	98.39% (-0.90%)
Autoregressive model g_{AR} architectures	
LSTM	**99.29%**
$\to$ GRU	98.49% (-0.80%)
$\to$ Transformer	86.95% (-12.34%)

6 Conclusion

In this paper, we propose CDPCC, a novel contrastive learning framework designed to extract informative latent representations from time series data. CDPCC is specifically designed to capture the cross-domain dynamics between time and frequency features of time series signals. The framework first splits the time series into non-overlapping frames, applying FFT to each frame to create its spectral view. The cross-domain predictive contrasting module then learns correlations and dynamic patterns between the time and frequency domains. Additionally, we propose a cross-domain contextual contrasting module to capture discriminative features. Experimental results demonstrate that a linear classifier trained on the features learned by CDPCC performs comparably to fully supervised models. Moreover, CDPCC proves highly efficient in few-labeled and transfer learning scenarios—achieving superior performance with only 50% of labeled data compared to fully supervised training on the entire labeled dataset.

Acknowledgement

We gratefully acknowledge the financial support from the Natural Sciences and Engineering Research Council of Canada (NSERC).

References

[1] M. Khan, A. Haleem, and M. Javaid, "Changes and improvements in industry 5.0: A strategic approach to overcome the challenges of industry 4.0," *Green Technologies and Sustainability*, vol. 1, no. 2, p. 100 020, 2023, ISSN: 2949-7361.

[2] Y.-J. Park, S.-K. S. Fan, and C.-Y. Hsu, "A review on fault detection and process diagnostics in industrial processes," *Processes*, vol. 8, no. 9, 2020, ISSN: 2227-9717. DOI: 10.3390/pr8091123.

[3] D. Miljković, "Fault detection methods: A literature survey," in *2011 Proceedings of the 34th International Convention MIPRO*, 2011, pp. 750–755.

[4] E. Russell, L. Chiang, and R. Braatz, *Data-Driven Methods for Fault Detection and Diagnosis in Chemical Processes.* Jan. 2000, ISBN: 978-1-4471-1133-7. DOI: 10.1007/978-1-4471-0409-4.

[5] V. Venkatasubramanian, R. Rengaswamy, K. Yin, and S. N. Kavuri, "A review of process fault detection and diagnosis: Part i: Quantitative model-based methods," *Computers Chemical Engineering*, vol. 27, no. 3, pp. 293–311, 2003, ISSN: 0098-1354. DOI: https://doi.org/10.1016/S0098-1354(02)00160-6.

[6] V. Venkatasubramanian, R. Rengaswamy, and S. N. Kavuri, "A review of process fault detection and diagnosis: Part ii: Qualitative models and search strategies," *Computers Chemical Engineering*, vol. 27, no. 3, pp. 313–326, 2003, ISSN: 0098-1354. DOI: https://doi.org/10.1016/S0098-1354(02)00161-8.

[7] V. Venkatasubramanian, R. Rengaswamy, S. N. Kavuri, and K. Yin, "A review of process fault detection and diagnosis: Part iii: Process history based methods," *Computers Chemical Engineering*, vol. 27, no. 3, pp. 327–346, 2003, ISSN: 0098-1354. DOI: https://doi.org/10.1016/S0098-1354(02)00162-X.

[8] S. Qiu *et al.*, "Deep learning techniques in intelligent fault diagnosis and prognosis for industrial systems: A review," *Sensors*, vol. 23, no. 3, 2023, ISSN: 1424-8220. DOI: 10.3390/s23031305.

[9] I. Yousef, A. Tulsyan, S. L. Shah, and R. B. Gopaluni, "Visual analytics for process monitoring: Leveraging time-series imaging for enhanced interpretability," *Journal of Process Control*, vol. 132, p. 103 127, 2023, ISSN: 0959-1524. DOI: https://doi.org/10.1016/j.jprocont.2023.103127.

[10] L. Ericsson, H. Gouk, C. C. Loy, and T. M. Hospedales, "Self-supervised representation learning: Introduction, advances, and challenges," *IEEE Signal Processing Magazine*, vol. 39, no. 3, pp. 42–62, May 2022, ISSN: 1558-0792. DOI: 10.1109/msp.2021.3134634.

[11] U. Ozbulak *et al.*, *Know your self-supervised learning: A survey on image-based generative and discriminative training*, 2023. arXiv: 2305.13689 [cs.CV].

[12] A. Jaiswal, A. R. Babu, M. Z. Zadeh, D. Banerjee, and F. Makedon, *A survey on contrastive self-supervised learning*, 2021. arXiv: 2011.00362 [cs.CV].

[13] U. Ozbulak *et al.*, *Know your self-supervised learning: A survey on image-based generative and discriminative training*, 2023. arXiv: 2305.13689.

[14] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, *A simple framework for contrastive learning of visual representations*, 2020. arXiv: `2002.05709 [cs.LG]`. [Online]. Available: `https://arxiv.org/abs/2002.05709`.

[15] S. Gidaris, P. Singh, and N. Komodakis, "Unsupervised representation learning by predicting image rotations," *ArXiv*, vol. abs/1803.07728, 2018.

[16] M. Noroozi and P. Favaro, "Unsupervised learning of visual representations by solving jigsaw puzzles," in *Computer Vision – ECCV 2016*, B. Leibe, J. Matas, N. Sebe, and M. Welling, Eds., Cham: Springer International Publishing, 2016, pp. 69–84, ISBN: 978-3-319-46466-4.

[17] R. Zhang, P. Isola, and A. A. Efros, *Colorful image colorization*, 2016. arXiv: `1603.08511 [cs.CV]`.

[18] A. van den Oord, Y. Li, and O. Vinyals, "Representation learning with contrastive predictive coding," *ArXiv*, vol. abs/1807.03748, 2018.

[19] P. H. Le-Khac, G. Healy, and A. F. Smeaton, "Contrastive representation learning: A framework and review," *IEEE Access*, vol. 8, pp. 193 907–193 934, 2020, ISSN: 2169-3536. DOI: `10.1109/access.2020.3031549`.

[20] P. Kumar, P. Rawat, and S. Chauhan, "Contrastive self-supervised learning: Review, progress, challenges and future research directions," *International Journal of Multimedia Information Retrieval*, vol. 11, no. 4, pp. 461–488, 2022, ISSN: 2192-662X. DOI: `10.1007/s13735-022-00245-6`.

[21] Z. Yue *et al.*, "Ts2vec: Towards universal representation of time series," in *AAAI Conference on Artificial Intelligence*, 2021.

[22] L. Yang and linda Qiao, "Unsupervised time-series representation learning with iterative bilinear temporal-spectral fusion," in *International Conference on Machine Learning*, 2022.

[23] L. Zhang, S. Frank, J. Kim, X. Jin, and M. Leach, "A systematic feature extraction and selection framework for data-driven whole-building automated fault detection and diagnostics in commercial buildings," *Building and Environment*, vol. 186, p. 107 338, 2020, ISSN: 0360-1323. DOI: `https://doi.org/10.1016/j.buildenv.2020.107338`. [Online]. Available: `https://www.sciencedirect.com/science/article/pii/S0360132320307071`.

[24] M. Altaf, T. Akram, M. A. Khan, M. Iqbal, M. M. I. Ch, and C.-H. Hsu, "A new statistical features based approach for bearing fault diagnosis using vibration signals," *Sensors*, vol. 22, no. 5, 2022, ISSN: 1424-8220. DOI: `10.3390/s22052012`. [Online]. Available: `https://www.mdpi.com/1424-8220/22/5/2012`.

[25] S. Velliangiri, S. Alagumuthukrishnan, and S. I. Thankumar joseph, "A review of dimensionality reduction techniques for efficient computation," *Procedia Computer Science*, vol. 165, pp. 104–111, 2019, 2nd International Conference on Recent Trends in Advanced Computing ICRTAC -DISRUP - TIV INNOVATION , 2019 November 11-12, 2019, ISSN: 1877-0509. DOI: `https://doi.org/10.1016/j.procs.2020.01.079`. [Online]. Available: `https://www.sciencedirect.com/science/article/pii/S1877050920300879`.

[26] M. Noruzi Nashalji, M. Aliyari Shoorehdeli, and M. Teshnehlab, "Fault detection of the tennessee eastman process using improved pca and neural classifier," in *Soft Computing in Industrial Applications*, X.-Z. Gao, A. Gaspar-Cunha, M. Köppen, G. Schaefer, and J. Wang, Eds., Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 41–50.

[27] T. J. Rato and M. S. Reis, "Fault detection in the tennessee eastman benchmark process using dynamic principal components analysis based on decorrelated residuals (dpca-dr)," *Chemometrics and Intelligent Laboratory Systems*, vol. 125, pp. 101–108, 2013, ISSN: 0169-7439. DOI: https://doi.org/10.1016/j.chemolab.2013.04.002. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0169743913000592.

[28] Y. Zhang, W. Du, Y. Fan, and L. Zhang, "Process fault detection using directional kernel partial least squares," *Industrial & Engineering Chemistry Research*, vol. 54, no. 9, pp. 2509–2518, 2015. DOI: 10.1021/ie501502t. eprint: https://doi.org/10.1021/ie501502t.

[29] J. Dong, K. Zhang, Y. Huang, G. Li, and K. Peng, "Adaptive total pls based quality-relevant process monitoring with application to the tennessee eastman process," *Neurocomputing*, vol. 154, pp. 77–85, 2015, ISSN: 0925-2312. DOI: https://doi.org/10.1016/j.neucom.2014.12.017. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0925231214016816.

[30] Z. Chen, S. X. Ding, K. Zhang, Z. Li, and Z. Hu, "Canonical correlation analysis-based fault detection methods with application to alumina evaporation process," *Control Engineering Practice*, vol. 46, pp. 51–58, 2016, ISSN: 0967-0661. DOI: https://doi.org/10.1016/j.conengprac.2015.10.006. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0967066115300332.

[31] Z. Chen, S. X. Ding, T. Peng, C. Yang, and W. Gui, "Fault detection for non-gaussian processes using generalized canonical correlation analysis and randomized algorithms," *IEEE Transactions on Industrial Electronics*, vol. 65, no. 2, pp. 1559–1567, 2018. DOI: 10.1109/TIE.2017.2733501.

[32] J. Yu and Y. Zhang, "Challenges and opportunities of deep learning-based process fault detection and diagnosis: A review," *Neural Comput. Appl.*, vol. 35, no. 1, pp. 211–252, Nov. 2022, ISSN: 0941-0643. DOI: 10.1007/s00521-022-08017-3. [Online]. Available: https://doi.org/10.1007/s00521-022-08017-3.

[33] Z. Zhu *et al.*, "A review of the application of deep learning in intelligent fault diagnosis of rotating machinery," *Measurement*, vol. 206, p. 112 346, 2023, ISSN: 0263-2241. DOI: https://doi.org/10.1016/j.measurement.2022.112346.

[34] O. Fink, Q. Wang, M. Svensén, P. Dersin, W.-J. Lee, and M. Ducoffe, "Potential, challenges and future directions for deep learning in prognostics and health management applications," *Engineering Applications of Artificial Intelligence*, vol. 92, p. 103 678, 2020, ISSN: 0952-1976. DOI: https://doi.org/10.1016/j.engappai.2020.103678. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0952197620301184.

[35] N. Amruthnath and T. Gupta, "A research study on unsupervised machine learning algorithms for early fault detection in predictive maintenance," in *2018 5th International Conference on Industrial Engineering and Applications (ICIEA)*, 2018, pp. 355–361. DOI: 10.1109/IEA.2018.8387124.

[36] K. Yan, J. Huang, W. Shen, and Z. Ji, "Unsupervised learning for fault detection and diagnosis of air handling units," *Energy and Buildings*, vol. 210, p. 109 689, 2020, ISSN: 0378-7788. DOI: https://doi.org/10.1016/j.enbuild.2019.109689. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0378778819320134.

[37] L. Schmarje, M. Santarossa, S.-M. Schröder, and R. Koch, "A survey on semi-, self-and unsupervised learning for image classification," *IEEE Access*, vol. PP, pp. 1–1, May 2021. DOI: 10.1109/ACCESS.2021.3084358.

[38] R. Balestriero *et al.*, *A cookbook of self-supervised learning*, 2023. arXiv: 2304.12210 [cs.LG]. [Online]. Available: https://arxiv.org/abs/2304.12210.

[39] M. Caron, P. Bojanowski, A. Joulin, and M. Douze, *Deep clustering for unsupervised learning of visual features*, 2019. arXiv: 1807.05520 [cs.CV].

[40] Z. Wu, Y. Xiong, S. Yu, and D. Lin, *Unsupervised feature learning via non-parametric instance-level discrimination*, 2018. arXiv: 1805.01978 [cs.CV]. [Online]. Available: https://arxiv.org/abs/1805.01978.

[41] C. Doersch, A. Gupta, and A. A. Efros, "Unsupervised visual representation learning by context prediction," in *2015 IEEE International Conference on Computer Vision (ICCV)*, 2015, pp. 1422–1430. DOI: 10.1109/ICCV.2015.167.

[42] Y. LeCun and I. Misra, *Self-supervised learning: The dark matter of intelligence*, Facebook AI Blog, 2020. [Online]. Available: https://ai.facebook.com/blog/self-supervised-learning-the-dark-matter-of-intelligence/.

[43] S. Deldari, H. Xue, A. Saeed, J. He, D. V. Smith, and F. D. Salim, *Beyond just vision: A review on self-supervised representation learning on multimodal and temporal data*, 2022. arXiv: 2206.02353 [cs.LG].

[44] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick, *Momentum contrast for unsupervised visual representation learning*, 2020. arXiv: 1911.05722 [cs.CV].

[45] X. Chen, H. Fan, R. Girshick, and K. He, *Improved baselines with momentum contrastive learning*, 2020. arXiv: 2003.04297 [cs.CV].

[46] J.-B. Grill *et al.*, *Bootstrap your own latent: A new approach to self-supervised learning*, 2020. arXiv: 2006.07733 [cs.LG].

[47] M. Caron, I. Misra, J. Mairal, P. Goyal, P. Bojanowski, and A. Joulin, *Unsupervised learning of visual features by contrasting cluster assignments*, 2021. arXiv: 2006.09882 [cs.CV].

[48] T. Mikolov, K. Chen, G. Corrado, and J. Dean, *Efficient estimation of word representations in vector space*, 2013. arXiv: 1301.3781 [cs.CL]. [Online]. Available: https://arxiv.org/abs/1301.3781.

[49] N. Reimers and I. Gurevych, *Sentence-bert: Sentence embeddings using siamese bert-networks*, 2019. arXiv: `1908.10084 [cs.CL]`. [Online]. Available: `https://arxiv.org/abs/1908.10084`.

[50] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, *A simple framework for contrastive learning of visual representations*, 2020. arXiv: `2002.05709 [cs.LG]`. [Online]. Available: `https://arxiv.org/abs/2002.05709`.

[51] F. Falck, S. K. Sarkar, S. Roy, and S. L. Hyland, "Contrastive representation learning for electroencephalogram classification," in *ML4H@NeurIPS*, 2020. [Online]. Available: `https://api.semanticscholar.org/CorpusID:229781944`.

[52] E. Eldele *et al.*, *Time-series representation learning via temporal and contextual contrasting*, 2021. arXiv: `2106.14112 [cs.LG]`. [Online]. Available: `https://arxiv.org/abs/2106.14112`.

[53] X. Zhang, Z. Zhao, T. Tsiligkaridis, and M. Zitnik, *Self-supervised contrastive pre-training for time series via time-frequency consistency*, 2022. arXiv: `2206.08496 [cs.LG]`. [Online]. Available: `https://arxiv.org/abs/2206.08496`.

[54] T. T. Um *et al.*, "Data augmentation of wearable sensor data for parkinson's disease monitoring using convolutional neural networks," in *Proceedings of the 19th ACM International Conference on Multimodal Interaction*, ser. ICMI '17, ACM, Nov. 2017. DOI: `10.1145/3136755.3136817`.

[55] K. Gupta, T. Ajanthan, A. van den Hengel, and S. Gould, *Understanding and improving the role of projection head in self-supervised learning*, 2022. arXiv: `2212.11491 [cs.LG]`. [Online]. Available: `https://arxiv.org/abs/2212.11491`.

[56] N. F. Thornhill, S. C. Patwardhan, and S. L. Shah, "A continuous stirred tank heater simulation model with applications," *Journal of Process Control*, vol. 18, no. 3, pp. 347–360, 2008, Festschrift honouring Professor Dale Seborg, ISSN: 0959-1524. DOI: `https://doi.org/10.1016/j.jprocont.2007.07.006`.

[57] I. Yousef, L. D. Rippon, C. Prévost, S. L. Shah, and R. B. Gopaluni, "The arc loss challenge: A novel industrial benchmark for process analytics and machine learning," *Journal of Process Control*, vol. 128, p. 103 023, 2023, ISSN: 0959-1524. DOI: `https://doi.org/10.1016/j.jprocont.2023.103023`.

[58] L. Rippon *et al.*, "Representation learning and predictive classification: Application with an electric arc furnace," *Computers Chemical Engineering*, vol. 150, p. 107 304, 2021, ISSN: 0098-1354. DOI: `https://doi.org/10.1016/j.compchemeng.2021.107304`.

[59] C. Lessmeier, J. K. Kimotho, D. Zimmer, and W. Sextro, "Condition monitoring of bearing damage in electromechanical drive systems by using motor current signals of electric motors: A benchmark data set for data-driven classification," in *European Conference of the Prognostics and Health Management Society*, 2016.