An NCDE-based Framework for Universal Representation Learning of Time Series

Zihan Liu[1,2] , **Bowen Du**[3] , **Junchen Ye**[3*] , **Xianqing Wen**[1] and **Leilei Sun**[1]
[1]SKLSDE Lab, Beihang University, Beijing, China
[2]Shanghai AI Laboratory, Shanghai, China
[3]School of Transportation Science and Engineering, Beihang University, Beijing, China
{liuzihan, dubowen, junchenye, xqwen, leileisun}@buaa.edu.cn

Abstract

Exploiting self-supervised learning (SSL) to extract the universal representations of time series could not only capture the natural properties of time series but also offer huge help to the downstream tasks. Nevertheless, existing time series representation learning (TSRL) methods face challenges in attaining universality. Indeed, existing methods relying solely on one SSL strategy (either contrastive learning (CL) or generative) often fall short in capturing rich semantic information for various downstream tasks. Moreover, time series exhibit diverse distributions and inherent characteristics, particularly with the common occurrence of missing values, posing a notable challenge for existing backbones in effectively handling such diverse time series data. To bridge these gaps, we propose CTRL, a framework for universal TSRL. For the first time, we employ Neural Controlled Differential Equation (NCDE) as the backbone for TSRL, which captures the continuous processes and exhibits robustness to missing data. Additionally, a dual-task SSL strategy, integrating both reconstruction and contrasting tasks, is proposed to enrich the semantic information of the learned representations. Furthermore, novel hard negative construction and false negative elimination mechanisms are proposed to improve sampling efficiency and reduce sampling bias in CL. Finally, extensive experiments demonstrate the superiority of CTRL in forecasting, classification, and imputation tasks, particularly its outstanding robustness to missing data.

1 Introduction

Time series is an important type of data that is ubiquitous in a wide variety of fields, including science, healthcare, finance, transportation, manufacturing, etc. Universal time series representation learning (UTSRL) could not only capture the inherent nature of time series but also greatly enhance the performance of various downstream tasks. Furthermore, UTSRL is an early yet crucial move towards enhancing the pattern recognition and reasoning capabilities inherent in time series foundational models. It lays essential groundwork for future development of artificial general intelligence that can effectively comprehend and process common time series data.

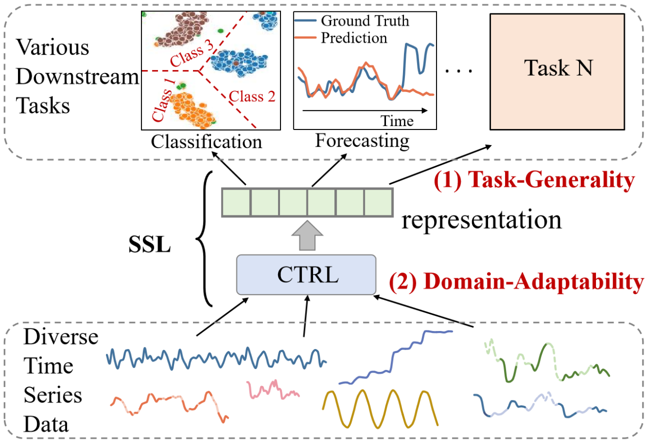


Figure 1: The goal of our CTRL framework is universal time series representation learning, with a focus on universality. The learned representations exhibit (1) Task-Generality, being applicable to various downstream tasks. and (2) Domain-Adaptability, demonstrating robustness for diverse time series data.

As illustrated in Figure 1, the universality of UTSRL is underscored by its two pivotal aspects: (1) Task-Generality: The learned representations should be applicable to a wide variety of downstream tasks, not just the specific task. (2) Domain-Adaptability: The learned representations should demonstrate robustness and stability in the face of time-series data with varied distributions or inherent characteristics. However, unfortunately, the expected universal performance are often not realized for a variety of reasons in existing time series representation learning (TSRL) methods.

First, existing prevalent self-supervised strategies, including contrastive learning (CL) and generative methods, each emphasize different aspects of feature extraction. As such, relying solely on a single self-supervised strategy often falls short in ensuring that the learned representations encompass rich and diverse semantic information, posing challenges when adapting to a wide array of downstream tasks. [Zhang *et al.*, 2023] Generative methods [Zerveas *et al.*, 2021;

[*] Corresponding Author

Chowdhury *et al.*, 2022; Nie *et al.*, 2023], like Masked Auto-Encoders (MAE), typically embed the randomly masked input into a latent space via a Transformer encoder and then reconstruct the original input signals. However, these methods model representation distribution at point-wise level, which might fail to effectively capture high-level abstractions. Notably, most transformer-based MAE methods for TSRL exhibit promising performance in forecasting tasks, but mediocre in the classification tasks [Nie *et al.*, 2023]. Contrastive learning methods force the encoder to learn representations that draw semantically similar positive samples closer while pushing negative samples further apart. This approach primarily concentrates on learning discriminative features. And many studies [Tonekaboni *et al.*, 2021; Franceschi *et al.*, 2019] focused on learning coarse-grained representations of the whole segment of the input time series, which may not be suitable for tasks requiring fine-grained representations such as forecasting and imputation.

Second, the huge diversities in distributions and underlying characteristics among different time series data pose challenges in constructing high-quality positive and negative samples for time series CL methods. These methods encourage the model to learn key invariance properties of time series through positive samples constructed by data augmentation, including sub-series consistency [Franceschi *et al.*, 2019], temporal consistency [Tonekaboni *et al.*, 2021], transformation consistency [Eldele *et al.*, 2021], contextual consistency [Yue *et al.*, 2022], etc. However, most of the TS-property-invariance are based on strong assumptions of date distribution, which could detrimentally impact Domain Adaptability. For example, sub-series consistency encourages the representation of a time series to be closer to its sampled sub-series, but it becomes vulnerable when there exist level shifts. Additionally, existing approaches typically adopt random negative sampling inspired by computer vision experiences. However, time series possess inherent characteristics, including complicated periodicity and inter-time series correlations, making randomly sampled negatives prone to containing false negatives (i.e. negative samples with similar semantics to the positive example). This compromises the semantics and generalization of the learned representations.

Third, widely used TSRL backbones like TCN [Bai *et al.*, 2018] and Transformer [Vaswani *et al.*, 2017] have their limitations in Domain-Adaptability. Time series data collected in the wild often have irregular sampling intervals and missing values, but the latest popular backbones are very challenging to deal with such irregularities. Moreover, TCN employs padding to handle variable-length data, but model performance could be affected when sequence length obviously differs from its fixed receptive field size. And recent research DLinear [Zeng *et al.*, 2023] has pointed out that the nature of the permutation invariant self-attention mechanism in Transformers inevitably results in temporal information loss.

To address these gaps, we propose a novel N**C**DE-based framework for universal **T**ime series **R**epresentation **L**earning, named **CTRL**. To enhance Task-Generality, we design a dual-task SSL strategy that couples contrastive learning with reconstruction to capture diverse semantic features. For improved Domain-Adaptability, we integrate Neural Controlled Differential Equation (NCDE) [Kidger *et al.*, 2020] model within the SSL paradigm for the first time. This encoder captures the continuous evolving processes inherent in time series and effectively manages diverse time series, particularly irregular ones. Additionally, we employ a simple yet effective data augmentation technique—masking—to generate different views of the input, which avoids the necessity for complex invariance assumptions and synergizes well with our NCDE encoder. Last but not least, novel hard negative construction strategies and false negative elimination mechanisms are proposed to improve sampling efficiency and reduce sampling bias in contrastive learning. The main contributions are summarized as follows:

- *It is the first time that Neural Controlled Differential Equation (NCDE)* is innovatively introduced as a robust backbone for universal time series representation learning. It shows greater potential compared to widely-used Transformer and TCN in capturing continuous evolving processes and handling diverse time series, especially irregular/missing data.
- *A dual-task self-supervised learning strategy* is designed. It utilizes contrastive learning to learn discriminative features from time series, as well as incorporates a reconstruction task to further enrich the representation with semantic information, distinguishing it from the existing methods that employ one task only.
- *A debiased contrastive learning framework* is proposed for time series representation learning. Compared with previous work which adopts all other samples as negatives in batch, the proposed hard negative construction and false negative elimination methods could improve the quality of negatives and reduce sampling bias in time series contrastive learning.

2 Related Work

2.1 Time Series Representation Learning

Self-supervised representation learning is becoming increasingly popular. Nonetheless, in the domain of time series data, SSL has been comparatively less explored when compared to other domains, such as computer vision (CV) or natural language processing (NLP) [Tian *et al.*, 2020; He *et al.*, 2020; Chen *et al.*, 2020; Kenton and Toutanova, 2019; Joshi *et al.*, 2020; Sun *et al.*, 2019; He *et al.*, 2022]. The existing methods of TSRL can be primarily divided into two branches: one is based on Contrastive and the other is based on Generative.

(1) Contrastive. These methods construct positive and negative samples and train the models to close the distance between positive pairs and increase the distance between negative pairs. T-Loss [Franceschi *et al.*, 2019] proposes an encoder composed by dilated convolutions that admits variable-length inputs, and trains with a triplet loss employing time-based negative sampling. TNC [Tonekaboni *et al.*, 2021] trains an encoder to predict whether segments belong to the same neighborhood, with the neighborhood distribution modeled as a Gaussian distribution to capture the gradual transitions in temporal data. TS-TCC [Eldele *et al.*, 2021] encourages the consistency of different data augmentations and puts efforts to learn robust representation by means of cross-view prediction and contextual contrasting. TS2Vec [Yue *et al.*, 2022] proposes a universal framework for learning time series representations by performing hierarchical contrastive learning over augmented context views and building representation of an arbitrary sub-sequence by aggregating timestamp-level representations. CoST [Woo *et al.*, 2022] separately processes disentangled trend and seasonality parts of the original time series data to encourage discriminative seasonal and trend representations. Almost all of these methods rely on heavily data augmentation or other domain knowledge. However, the defects of these contrastive methods, such as sampling bias and spurious invariance assumption, could lower the generalizability of learned time series representations.

(2) Generative. These methods generally embed the input into a latent space via the encoder and decode the representation back to recover the input signals. TST [Zerveas *et al.*, 2021] pre-trains Transformer encoder by masking a small portion of time series and reconstructing them. PatchTST [Nie *et al.*, 2023] introduces two key components, patching and channel-independent structure, to design transformer-based model specifically for long-term time series forecasting tasks. Whereas, the representation distribution of the generative method is modeled at point-wise level, which may not effectively address the challenge of high-level abstraction in the time series data.

2.2 Neural Differential Equations

Neural differential equations are an elegant formulation combining continuous-time differential equations with the high-capacity function approximation of neural networks. They are widely recognized for their unique capacity to model complex, continuous dynamics, a feat not achievable by conventional discrete methods. [Kidger, 2022]

(1) NODE. By far the most common neural differential equation is a NODE [Chen *et al.*, 2018]:

$$z(t_1) = z(t_0) + \int_{t_0}^{t_1} f_\theta(z(t))dt, \tag{1}$$

where f_θ is the ODE function, which is implemented as a neural network to approximate $\dot{z} \overset{\text{def}}{=} \frac{dz(t)}{dt}$. Typically f_θ will be some standard simple neural architecture, such as a feedforward or convolutional network. To solve the integral problem, NODEs rely on ODE solvers, such as the explicit Euler method, the Dormand–Prince (DOPRI) method, and so on [Dormand and Prince, 1980]. In general, ODE solvers discretize time variable t and convert an integral into many steps of additions. For instance, the explicit Euler method can be written as follows in a step:

$$z(t+h) = z(t) + h \cdot f_\theta(z(t)), \tag{2}$$

where h is a pre-determined step size of the Euler method, which is usually smaller than 1. When $h = 1$, the formulation aligns with that of residual neural networks (ResNets). By taking smaller integration steps, we can directly parameterize and approximate the continuous evolution of latent states, which forms the basic idea of NODEs. In this regard, NODEs generalizes ResNets in a continuous manner.

(2) NCDE. One limitation of NODEs is that the solution of an ODE is determined by the initial condition at $z(t_0)$, leaving no direct mechanism for incorporating data that arrives later, which degrades the representation learning capability of NODEs. To this end, NCDEs [Kidger *et al.*, 2020] create path $X(t)$ from underlying time-series data, with $z(t_1)$ determined by both $z(t_0)$ and $X(t)$. NCDEs is the continuous-time limit of recurrent neural networks. NCDEs can be written as follows:

$$z(t_1) = z(t_0) + \int_{t_0}^{t_1} f_\theta(z(t))\, dX(t), \tag{3}$$

where $X(t)$ is a natural cubic spline path of underlying time-series data. Differently from NODEs, f_θ, which we call CDE function, is to approximate $\frac{dz(t)}{dX(t)}$. Whereas other methods can be used for $X(t)$, the original authors of NCDEs recommend the natural cubic spline method for its suitable characteristics to be used in NCDEs: i) it is twice differentiable, ii) its computational cost is not much, and iii) $X(t)$ becomes continuous w.r.t. t after the interpolation.

3 Preliminary

Given a set of time series $\mathcal{X} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \cdots, \boldsymbol{x}_N\}$ of N instances, and $\boldsymbol{x}_i \in \mathbb{R}^{T\times F}$ where T and F are sequence length and feature dimension respectively. The goal is to learn a nonlinear embedding function $\mathcal{F}_\theta$ that maps each $\boldsymbol{x}_i$ to a universal representation $\boldsymbol{r}_i$, applicable to various downstream tasks. The representation $\boldsymbol{r}_i = \{r_{i,1}, r_{i,2}, \cdots, r_{i,T}\}$ contains vectors $r_{i,t} \in \mathbb{R}^C$ for each timestamp t, where C is is the dimension of representation vectors.

4 Methodology

In this section, the proposed framework and all components will be stated elaborately. The overall architecture of CTRL is shown in Figure 2, which consists of three key components: masking augmentation, NCDE encoder and dual-task SSL strategy. We will clarify the merit of this particular combination and introduce the details of each component.

4.1 Masking

There are three main reasons why we use masking only for data augmentation: From the reconstruction task perspective, masking is essential. For CL, data augmentation is used to construct semantically similar positive pairs according to the consistency assumption and has a great impact on the quality of the learned representations. To minimize consistency bias, masking is adopted as a minimal form of data augmentation for time series within our framework. Last but not least, to strengthen the model robustness to irregular time series, we simulate this process by masking the original input.

Since time series has a lower information density and higher redundancy, the very short masked sequences (e.g., of 1 masked time-point) in the input can be easily referred from neighboring values, which makes the task trivial and thus the representation may not carry important abstract information. Here we utilize a complex randomization strategy proposed

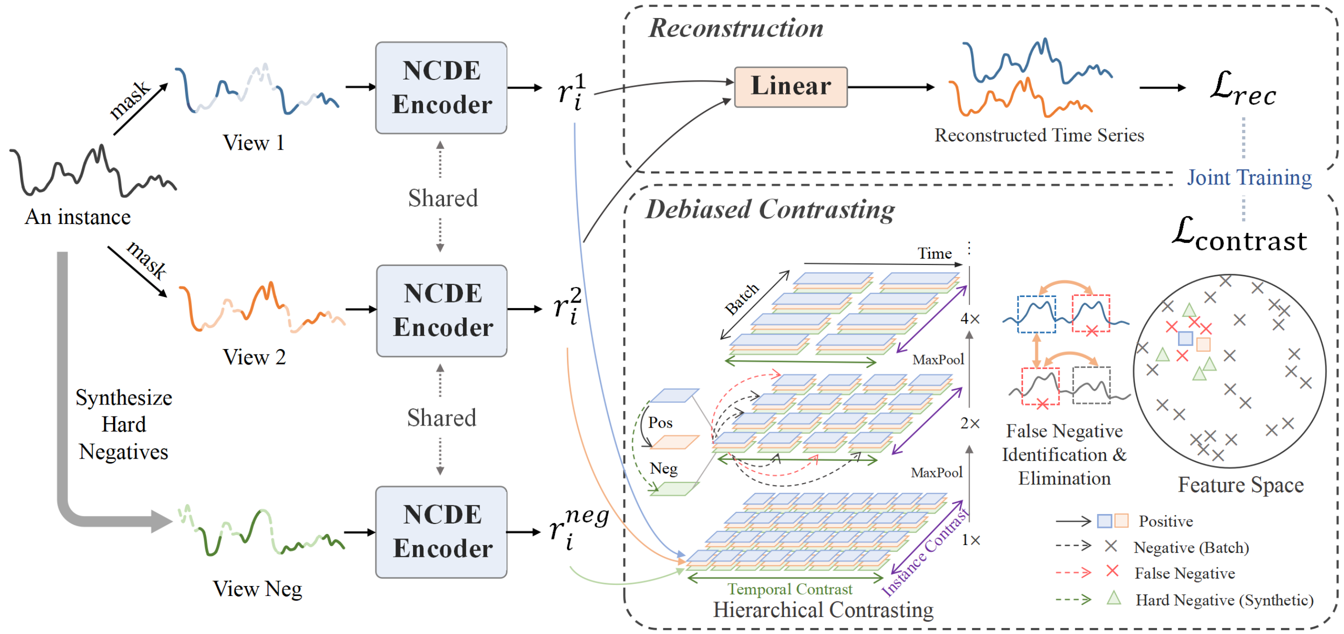


Figure 2: The overview architecture of CTRL. The different views of each time series generated by masking augmentation are fed into the NCDE encoder to obtain their corresponding representations. Then the representations flow into two branches to perform two self-supervised tasks: (1) reconstruction task, (2) contrastive task. Furthermore, we construct hard negatives to improve sampling efficiency, as well as identify and eliminate false negatives to reduce sampling bias in contrastive learning.

by [Zerveas *et al.*, 2021] to resolve the problem. Specifically, masks $Mask \in \{0,1\}^{T\times F}$ are created independently for each view of a given training sample $\boldsymbol{x}_i \in \mathbb{R}^{T\times F}$, where T represents the sequence length and F is the feature dimension. We use the state transition method to alternate masked and unmasked along the time axis and the mask ratio is r_m, such that each masked segment has a length that follows a geometric distribution with mean l_m. In our work, we adopt a higher masking ratio $r_m = 0.5$ coupled with an increased length for continuous masking at $l_m = 5$ to largely eliminate redundancy and prevent the model from focusing only on low-level semantic information.

4.2 NCDE Encoder

We employ Neural Controlled Differential Equations (NCDE) [Kidger *et al.*, 2020] as our backbone. As the continuous time analogue of an RNN, NCDE is designed to learn the evolving process behind time series. The continuous hidden states captured by NCDE are consistent with representation learning. In contrast, Transformer , originally proposed for NLP, treats time series values as tokens and deals with time series in a discrete manner. However, we believe that the underlying process of time series develops in continuous time. Another crucial reason for choosing NCDE as our encoder is its remarkable ability to handle diverse time series, particularly those with irregular intervals or missing values. Furthermore, NCDE's powerful mathematical properties for dealing with such irregularities bring many advantages to our dual-task SSL strategy. For reconstruction, there is no need for additional processing on missing values after masking, such as complementing with 0 [Yue *et al.*, 2022] or applying linear interpolation [Zerveas *et al.*, 2021], which may introduce deviations. This is because the NCDE can directly operate on irregularly sampled and partially observed time series. For CL, the NCDE encoder, when combined with masking augmentation, can construct effective semantically similar positive pairs and avoid the need for prior knowledge and overly strong assumptions about TS-property-invariance.

Following the application of our masking augmentation, we define the view as $x = ((t_0, x_0), \cdots, (t_T, x_T))$ for simplicity, annotated with observation time $t_0 < \cdots < t_T$. Here, $x_j \in (\mathbb{R} \cup \{*\})^F$, where $*$ represents either missing values or those masked by $Mask$. We employ the natural cubic spline method for interpolating these discrete time series, thereby building a continuous path $X(t)\colon [t_0, t_T] \to \mathbb{R}^{F+1}$, that satisfies the condition $X(t_j) = (t_j, x_j)$ for each observation x_j at its respective time-point t_j. And for other non-observed time-points, the natural cubic spline algorithm interpolates nearby observed data. Subsequently, the NCDE is driven by X and can be defined as

$$\begin{aligned} r_{t_0} &= \zeta_{\theta_2}(t_0, x_0), \\ r_t &= r_{t_0} + \int_{t_0}^{t} f_{\theta_1}(r_s)dX_s \quad t \in (t_0, t_T], \end{aligned} \tag{4}$$

where dX_s denotes a Riemann–Stieltjes integral. The hidden state $r_t \in \mathbb{R}^C$ reflects an evolving belief about the time series, updated continuously as observations $X(t)$ are made. Here, r_{t_0} is the initial hidden state, and C defines the size of hidden state. The integrand $f_{\theta_1}\colon \mathbb{R}^C \to \mathbb{R}^{C\times(F+1)}$ and $\zeta_{\theta_2}\colon \mathbb{R}^{F+1} \to \mathbb{R}^C$ are both neural networks depending on learnable parameters θ_1, θ_2. In our work, f_{θ_1} and ζ_{θ_2} are taken to be simple feedforward neural networks.

The output of the NCDE model portrays a continuous evolution over time. Consequently, we extract representations at the observation time-points, i.e., $\{r_1, r_2, \cdots, r_T\}$.

4.3 Dual-Task Training

To obtain informative features for various downstream tasks, we design a dual-task training strategy which combine reconstruction and contrastive learning tasks. The final form of the loss function in our framework is defined as:

$$\mathcal{L}_{dual} = \mathcal{L}_{rec} + \lambda \mathcal{L}_{contrast}, \tag{5}$$

where λ is a trade-off between two tasks.

Reconstruction

The representations r_i^1, r_i^2 are passed through a linear layer respectively to reconstruct the raw input time series:

$$\hat{x}_{i,t}^k = r_{i,t}^k \boldsymbol{W} + \boldsymbol{b}, \tag{6}$$

where $\boldsymbol{W} \in \mathbb{R}^{C \times F}$ and $\boldsymbol{b} \in \mathbb{R}^F$ are a trainable weight and a bias of the linear layer. We calculate Mean Absolute Error between the ground truth x_i and reconstruction $\hat{x}_i$ for masked and unmasked part of the data, respectively. The hyperparameter α $(0 < \alpha < 1)$ is used to control the relative weights between the two losses:

$$\begin{aligned} \mathcal{L}_{rec}^k &= \alpha \mathcal{L}_{masked}^k + (1-\alpha) \mathcal{L}_{unmasked}^k, \\ \mathcal{L}_{rec} &= \mathcal{L}_{rec}^1 + \mathcal{L}_{rec}^2, \end{aligned} \tag{7}$$

Debiased Contrasting

We employ hierarchical contrastive loss [Yue *et al.*, 2022] to facilitate multi-scale information learning, achieved through max pooling on the learned representations along the time axis. Moreover, we propose a debiased contrastive learning framework aiming to boost the efficiency of negative sampling and reduce sampling bias.

Hard Negative Construction (HNC). Recent studies [Kalantidis *et al.*, 2020] have shown that hard negative samples (i.e. true negative samples that are close to the anchor) make it more challenging for the encoders to learn distinguishing features and lead to higher performance. However, the exploration of hard negative has been relatively limited in time series domain. To facilitate better learning, we propose the following two methods to generate hard negatives.

(1) Temporal shuffling: Changing temporal orders may lead to a different evolutionary tendency which is a vital feature in time series. Thus, instead of using the shuffling strategy to generate positive samples as in previous work [Eldele *et al.*, 2021], we treat the shuffled time series as negative samples. Specifically, the time series instance is divided into several segments $x_i = \{seg_1, seg_2, \cdots, seg_M\}$, where M is the number of segments. And then half of the segments are randomly selected for shuffling. In this way, half of the segments remain in their original position, while their contextual information has changed. The purpose of this is to make the generated negatives much harder.

(2) Instance Mixing: This method incorporates anchor sample features into negative samples, rendering them more intricate to discriminate and improving the model's sensitivity to differentiate between positive and negative samples. Given a time series instance x_i and another instance x_j drawn randomly from the batch, the mixing example can be constructed as follows:

$$x_i^{(mix)} = \beta x_i + (1-\beta) x_j, \quad \beta \sim Beta(b, 1), \tag{8}$$

where β is a mixing parameter that determines the contribution of each time series in the new sample, and to avoid the issue of fixed parameter weighting and introduce fixed bias, beta distribution $\beta \sim Beta(b, 1)$ is utilized. It is worth noting that the value of b is recommended to be greater than 1 so that the original time series x_i can make up a larger proportion of the new sample than x_j.

False Negative Elimination (FNE). Besides the synthetic negatives above, we adopt other in-batch samples as negatives, following previous works. We design two strategies to identify false negatives, the top-k strategy and the threshold strategy. For each anchor x, x' and $\mathbb{B}^x$ denote its positive sample and the set of its negatives, respectively.

As for top-k strategy, we assume that the first k samples with the highest similarity to the anchor may be false negative samples. Thus, the false negatives of the anchor screened by top-k strategy can be formulated as:

$$\Psi^x = \{x^- | x^- \in top(sim(x, x^-), k), x^- \in \mathbb{B}^x\}, \tag{9}$$

where $sim(u, v)$ is a similarity function and $top(\mathbb{S}, k)$ denotes a set containing the maximum k elements of set $\mathbb{S}$.

As for threshold strategy, we believe that the similarity between negative pairs is typically lower than that between positive pairs, aligning with the training objectives of CL. Thus, we can utilize the similarity of positive pairs as a benchmark, considering negatives with high similarity to the anchor as potential false negatives:

$$\Phi^x = \{x^- | sim(x, x^-) > \phi \times sim(x, x'), x^- \in \mathbb{B}^x\}, \tag{10}$$

where ϕ is a hyperparameter of the similarity threshold.

Since the true labels or semantic similarities of time series are unavailable, we directly utilize the dot product of the representations to measure the similarity between samples, i.e. $sim(x, x^-) = r \cdot r^-$, where r and r^- is the representations of x and its negative x^-. However, it boils down to a chicken-and-egg problem: we want to learn good semantic representations, but may demand certain semantic similarity from the start. Therefore, we delay the false negative elimination until after a period of model training, which we found to be more beneficial for performance improvement. The top-k strategy may be preferred given information about the approximate number of false negatives, while threshold may be better suited when a dynamic adaptation is expected. In addition, we also consider a strategy that combines top-k and threshold, which not only allows for adaptive adjustments but also guarantees a minimum count for negative samples.

We perform false negative elimination in the calculation of both temporal contrastive loss and instance-wise contrastive loss. Let i be the index of the input time series sample and t be the timestamp. $r_{i,t}$ and $r'_{i,t}$ denote the representations for the same timestamp t but from two augmentations of x_i.

These two contrastive loss for $r_{i,t}$ can be both formulated as:

$$\ell_{type}^{r_{i,t}} = -\log \frac{\exp(r_{i,t} \cdot r'_{i,t})}{\exp(r_{i,t} \cdot r'_{i,t}) + \sum\limits_{r^- \in \boldsymbol{\Omega}_{type}^{r_{i,t}}} \exp(r_{i,t} \cdot r^-)}, \tag{11}$$

where $type$ indicates whether it is temporal or instance-wise, and $\boldsymbol{\Omega}_{type}^{r_{i,t}}$ represents the set of negative sample representations r^- corresponding to $r_{i,t}$ in the $type$ contrative loss.

As for temporal contrastive loss, to learn discriminative representations over time, we take those at different timestamps from the same time series as negatives, i.e., $\mathbb{B}_{temp}^{r_{i,t}} = \{r_{i,t'}\}_{t'=1,t'\neq t}^{T} \cup \{r'_{i,t'}\}_{t'=1}^{T}$, where T is sequence length. Building on this, we employ the top-k strategy to eliminate false negative samples, resulting in $\boldsymbol{\Omega}_{temp}^{r_{i,t}} = \mathbb{B}_{temp}^{r_{i,t}} \setminus \Psi_{temp}^{r_{i,t}}$.

As for instance-wise contrastive loss, we use representations of other time series in the same batch and synthetic hard negative samples $r_{i,t}^{neg}$ at same timestamp t as negative samples, i.e., $\mathbb{B}_{inst}^{r_{i,t}} = \{r_{j,t}\}_{j=1,j\neq i}^{B} \cup \{r'_{j,t}\}_{j=1}^{B} \cup \{r_{j,t}^{neg}\}_{j=1}^{B}$, where B is the batch size. We then filter the false negatives through the combined top-k and threshold strategy, i.e., $\boldsymbol{\Omega}_{inst}^{r_{i,t}} = \mathbb{B}_{inst}^{r_{i,t}} \setminus (\Phi_{inst}^{r_{i,t}} \cap \Psi_{inst}^{r_{i,t}})$.

The two losses are complementary to each other. For example, given traffic data of a city, instance contrast may learn the road-specific characteristics, while temporal contrast aims to mine the dynamic trends over time. The overall contrast loss is defined as:

$$\mathcal{L}_{contrast} = \frac{1}{2BT} \sum_i \sum_t \left(\ell_{temp}^{r_{i,t}} + \ell_{temp}^{r'_{i,t}} + \ell_{inst}^{r_{i,t}} + \ell_{inst}^{r'_{i,t}} \right). \tag{12}$$

5 Experiments

In this section, we verify the superiority of our representation learning framework through extensive experiments in forecasting, classification and imputation tasks. Notably, CTRL attains the top average rank in all three tasks, demonstrating its outstanding universality. The source code is publicly available at https://github.com/LiuZH-19/CTRL.

5.1 Datasets

For time series forecasting task, we utilize four popular real-world datasets: Exchange Rate [Lai *et al.*, 2018], Wind [Wu *et al.*, 2020], Weather[1], and ILI [Wu *et al.*, 2021].

For time series classification task, we select 18 datasets from the UCR, UEA Time Series Classification Archive [Dau *et al.*, 2019; Bagnall *et al.*, 2018]. This selection is in line with the classification datasets utilized in TST [Zerveas *et al.*, 2021] and TimesNet [Wu *et al.*, 2022]. Additionally, we include 8 univariate time series datasets. These datasets exhibit diverse characteristics, including variations in the number, dimension, sequence length, and the number of classes.

For time series imputation task, we select the datasets from the electricity and weather scenarios, including ETT [Zhou *et al.*, 2021] and Weather, where the data-missing problem happens commonly. To compare the model capacity under different proportions of missing data, we randomly mask the time points in the ratio of $\{12.5\%, 25\%, 37.5\%, 50\%\}$.

5.2 Baselines

We present a comprehensive comparison of the well-acknowledged and advanced models. The selected baselines are categorized into two groups: (1) Representation Learning Methods: T-Loss [Franceschi *et al.*, 2019], TNC [Tonekaboni *et al.*, 2021], TS-TCC [Eldele *et al.*, 2021], TS2Vec [Yue *et al.*, 2022], CoST [Woo *et al.*, 2022], TST [Zerveas *et al.*, 2021] and PatchTST [Nie *et al.*, 2023]. (2) End-to-End Supervised Methods: LSTM [Sutskever *et al.*, 2014], TCN [Bai *et al.*, 2018], Informer [Zhou *et al.*, 2021], Autoformer [Wu *et al.*, 2021], FEDformer [Zhou *et al.*, 2022], DLinear [Zeng *et al.*, 2023] and TimesNet [Wu *et al.*, 2022].

5.3 Reproduction Details for CTRL

The representation dimension C is set to 320. The batch size B is set to 128 by default, unless limited by memory. The learning rate is 0.001. The number of optimization iterations is determined empirically based on the dataset size. For masking data augmentation, we set the mask ratio r_m to 0.5 and the average length of continuous masking l_m to 5. The parameter α for the reconstruction loss is set to 0.8. The loss term trade-off parameter λ is tuned from the set $\{$0.01, 0.05, 0.1, 0.5, 1$\}$. Regarding our NCDE encoder, a fixed set of hyper-parameters is determined empirically regardless of datasets. The integrand f_{θ_1} is a feedforward network with 5 fully connected layers and 64 hidden channels. And we use a tangent hyperbolic function as the final nonlinearity for f_{θ_1}. ζ_{θ_2} is implemented as a fully connected network with 2 layers, where the hidden channel size is set to 128.

For hard negative construction, we typically set the beta-distribution parameter β to either 2 or 4, and the number of segments M to either 4 or 8. In the false negative identification process, we tune ϕ within the range [0.96, 0.99], and set k to be 20% to 40% of the total number of instances. These hyper-parameters for debiased contrasting should be chosen carefully according to the nature of the time series data. For example, in time series data with longer sequences, the value of M should be greater than in those with shorter sequences when applying shuffling.

We conduct 5 repetitions of all experiments using different random seeds, and report the average evaluation metrics. To ensure complete reproducibility, we have included detailed settings for each experiment in our public code.

5.4 Time Series Forecasting

The evaluation results on MSE for forecasting are presented in Table 1. In general, CTRL attains the best average rank of 1.688 across all dataset-horizon results and excels with either the top or second-best performance in most cases. Furthermore, our representations only need to be learned once for each dataset and can be directly applied to various horizons with linear regressions, including short-term and long-term, which demonstrates the universality of the representations. Even in comparison to the latest TSRL methods like CoST [Woo *et al.*, 2022] and PatchTST [Nie *et al.*, 2023],

[1] https://www.ncei.noaa.gov/data/local-climatological-data/

Dataset	H	TS-TCC	TNC	TS2Vec	CoST	PatchTST	CTRL
Avg. Rank		5.063	4.563	4.063	2.118	3.438	**1.688**
Exchange Rate	96	0.540	0.231	0.277	0.129	0.111	$\underline{0.107}$
	168	1.018	0.475	0.574	0.245	0.196	$\underline{0.182}$
	336	1.584	1.099	1.010	0.641	0.409	$\underline{0.403}$
	672	4.345	2.581	3.711	1.822	$\underline{0.966}$	*1.526*
Wind	96	0.754	0.561	0.512	0.511	0.605	$\underline{0.508}$
	168	0.722	0.559	0.512	0.511	0.601	$\underline{0.507}$
	336	0.719	0.550	0.502	0.500	0.551	$\underline{0.496}$
	672	0.717	0.551	0.505	0.503	0.566	$\underline{0.499}$
Weather	96	0.463	0.459	0.448	$\underline{0.423}$	0.494	*0.446*
	168	0.504	0.507	0.494	$\underline{0.465}$	0.542	*0.483*
	336	0.544	0.549	0.533	$\underline{0.499}$	0.578	*0.520*
	672	0.585	0.600	0.574	$\underline{0.544}$	0.647	*0.566*
ILI	6	1.917	3.080	4.139	2.064	$\underline{1.086}$	*1.654*
	12	2.613	3.609	4.580	2.437	$\underline{1.820}$	*2.263*
	48	4.782	4.849	4.619	2.512	$\underline{2.183}$	2.624
	60	4.798	4.882	4.943	2.741	$\underline{2.091}$	2.850

Table 1: Time series forecasting results compared to other representation learning methods on MSE. H denotes the forecasting horizon. The best and second-best MSE results are highlighted in $\underline{\text{underlined}}$ and *italicized* formatting, respectively.

tailored for time series forecasting, CTRL remains competitively strong. CoST, built upon the TS2Vec [Yue *et al.*, 2022] framework, captures more relevant features by learning a composition of trend and seasonal features. Meanwhile, PatchTST, as indicated by its name, employs patching to time series based on TST [Zerveas *et al.*, 2021], thereby enhancing the performance of transformer-based models in long-term forecasting. However, it's important to note that both CoST and PatchTST have limitations in terms of universality, which will be discussed in later sections.

5.5 Time Series Classification

The evaluation results of classification task are shown in Table 2. CTRL performs an average rank of 2.656 and an average improvement of 2.2% classification accuracy, setting it apart from all other models. These results demonstrate the power and robustness of CTRL to capture high-level semantics of time series. Furthermore, we observe that TST and PatchTST, the transformer-based models trained by reconstruction task, encounter challenges in capturing high-level features in excessively long sequences, such as InsectWingbeatSound and SelfRegulationSCP1. T-Loss [Franceschi *et al.*, 2019], TS-TCC [Eldele *et al.*, 2021], and TNC [Tonekaboni *et al.*, 2021] impose strong inductive biases to select positive pairs, which ultimately restricts their generalization. Regarding CoST, as an enhancement to TS2Vec, it notably improves forecasting tasks but falls short of outperforming TS2Vec in classification tasks, highlighting the challenge of excelling in diverse tasks. Additionally, due to its frequency domain computations, CoST requires consistency in input lengths between the pre-training and inference phases, constraining its universality.

5.6 Time Series Imputation

Table 3 presents the results of imputation task. The imputation task necessitates the model to uncover underlying temporal patterns from irregular and partially observed time series. CTRL attains the state-of-the-art performance in this challenging task, with an average MSE improvement of **32.3%**.

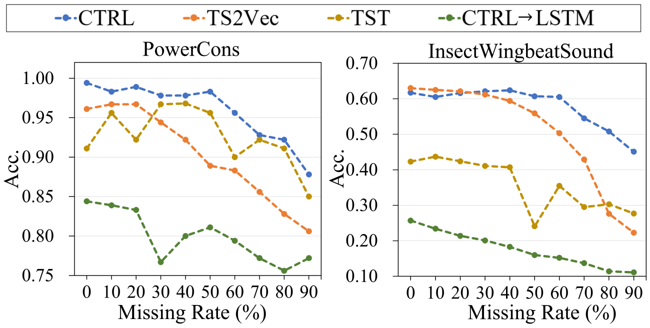


Figure 3: Accuracy scores with respect to the missing ratios.

5.7 Robustness to Missing Data

To further investigate the robustness of CTRL to missing data, we conduct classification experiments on two datasets and compare CTRL with two other SOTA models (TS2Vec, TST) and one variant (CTRL→LSTM) that replaces the NCDE encoder with LSTM. We randomly mask out observations for both training set and test set with specific missing rates of timestamps. Figure 3 illustrates that CTRL maintains steady performance when feeding data with a large proportion of missing values, while the other models all begin to decrease. Notably, even with 50% missing values, the accuracy of CTRL drops by only 1.1% and 1.6% on PowerCons and InsectWingbeatSound, respectively.

The robustness of CTRL against missing data can be attributed to two main factors: (1)The NCDE demonstrates powerful mathematical properties that enable it to effectively handle irregular time series, operating directly on irregularly sampled and partially observed data. In contrast, TS2Vec fills in missing values with 0, while TST uses linear interpolation, both of which may introduce deviations. (2) We employ a high mask ratio ($r_m = 0.5$) and longer continuous mask length ($l_m = 5$) to mask raw data during the self-supervised training stage, thereby enhancing our model's ability to infer representations under incomplete contexts. In contrast, TST has a mask ratio of 0.15 with $l_m = 3$, while TS2Vec randomly masks timestamps on the embedding layer instead of continuous segments. Although the mask ratio of TS2Vec is also 0.5, it utilizes a much simpler approach to mask.

In conclusion, CTRL, as a universal framework, exhibits exceptional robustness to missing data. This characteristic is crucial in real-world applications where data incompleteness is common and underscores the practical value of CTRL.

5.8 Ablation Study

In this section, We conduct comprehensive ablation studies to validate the effectiveness of our key components, which try to answer the following four research questions (RQs).

- **RQ1:** How does the proposed dual-task SSL strategy effectively improve performance and generalizability?
- **RQ2:** Is our masking augmentation effective and suitable for our NCDE-based framework?

Method	DTW	TST	TS-TCC	TNC	T-Loss	TS2Vec	CoST	PatchTST	CTRL
EthanolConcentration	0.323	0.262	0.285	0.297	0.205	0.308	0.296	0.289	*0.319*
FaceDetection	0.529	0.534	0.544	0.536	0.513	0.501	0.534	0.501	0.547
FingerMovements	0.530	0.560	0.460	0.470	0.580	0.480	0.470	0.530	0.593
Heartbeat	0.717	0.746	0.751	0.746	0.741	0.683	0.725	0.741	0.722
JapaneseVowels	0.949	0.978	0.930	0.978	0.989	0.984	0.972	0.965	0.976
PEMS-SF	0.711	0.740	0.734	0.699	0.676	0.682	0.791	0.798	0.751
SelfRegulationSCP1	0.775	0.754	0.823	0.799	0.843	0.812	0.816	0.768	0.864
SelfRegulationSCP2	0.539	0.550	0.533	0.550	0.539	0.578	0.539	0.533	*0.567*
SpokenArabicDigits	0.963	0.923	0.970	0.934	0.905	0.988	0.977	0.904	0.975
UWaveGestureLibrary	0.903	0.575	0.753	0.759	0.875	0.906	0.903	0.438	0.847
Chinatown	0.957	0.936	0.983	0.977	0.951	0.965	0.971	0.921	*0.979*
ECG5000	0.924	0.928	0.941	0.937	0.933	0.935	0.942	0.957	0.937
ElectricDevices	0.602	0.676	0.686	0.700	0.707	0.721	0.644	0.562	0.732
InsectWingbeatSound	0.355	0.266	0.415	0.549	0.597	0.630	0.592	0.254	*0.617*
MelbournePedestrian	0.791	0.741	0.949	0.942	0.944	0.959	0.944	0.936	*0.951*
PowerCons	0.878	0.911	0.961	0.933	0.900	0.961	0.959	0.889	0.994
DodgerLoopDay	0.500	0.200	-	-	-	0.562	0.585	0.313	*0.575*
DodgerLoopGame	0.877	0.696	-	-	-	0.841	0.894	0.674	0.899
Avg. Accuracy (excl. DodgerLoop*)	0.715	0.693	0.732	0.738	0.744	0.756	0.755	0.687	**0.773**
Avg. Rank	6.313	6.125	4.750	4.781	5.250	3.875	4.438	6.813	**2.656**

TS-TCC, TNC and T-Loss cannot handle datasets with missing values, including DodgerLoopDay and DodgerLoopGame.

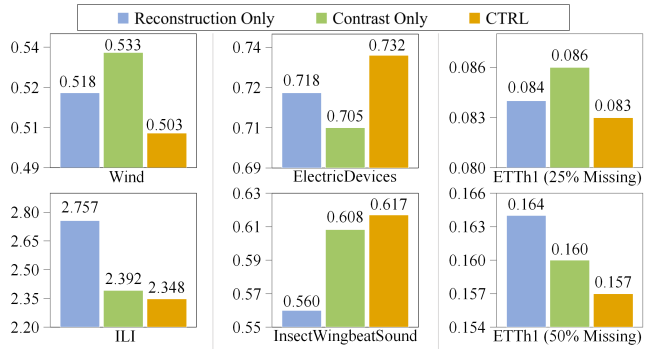
Table 2: Time series classification results compared to other representation learning methods. The best and second-best accuracy results are highlighted in underlined and *italicized* formatting, respectively.

		TS-TCC	TS2Vec	CoST	PatchTST	CTRL
	Avg. Rank	5.000	3.533	3.067	2.400	**1.000**
ETTm1	12.5%	0.186	0.130	0.085	0.140	0.036
	25.0%	0.210	0.156	0.113	0.085	0.037
	37.5%	0.247	0.184	0.142	0.071	0.040
	50.0%	0.283	0.215	0.177	0.072	0.044
	Avg. MSE	0.232	0.171	0.129	0.092	0.039
ETTh1	12.5%	0.330	0.233	0.166	0.257	0.083
	25.0%	0.357	0.271	0.204	0.193	0.095
	37.5%	0.429	0.319	0.245	0.200	0.116
	50.0%	0.493	0.396	0.304	0.207	0.157
	Avg. MSE	0.402	0.305	0.230	0.214	0.113
Weather	12.5%	0.240	0.201	0.199	0.206	0.157
	25.0%	0.246	0.211	0.212	0.184	0.160
	37.5%	0.272	0.228	0.229	0.180	0.168
	50.0%	0.290	0.246	0.250	0.188	0.179
	Avg. MSE	0.262	0.222	0.223	0.186	0.166

Table 3: Time series imputation results compared to other representation learning methods on MSE.

- **RQ3:** How does CTRL benefit from NCDE encoder?
- **RQ4:** How does the proposed debiased contrasting framework influence the performance of CTRL and other time series contrasting learning methods?

RQ1. As shown in Figure 4, we conduct an ablation study of two parts of our training loss on different datasets across three benchmark downstream tasks. *Reconstruction Only* and *Contrast Only* train the encoder independently using either the reconstruction or contrastive learning task, respectively.



(a) Forecasting (MSE↓) (b) Classification (Acc.↑) (c) Imputation (MSE↓)

Figure 4: Ablation study of dual-task SSL strategy.

In all observed results, CTRL's dual-task SSL strategy consistently surpasses the single-task SSL approaches. This clearly validates the effectiveness of integrating both reconstruction and contrastive learning tasks to acquire richer semantic information, indeed enhancing the performance and generalizability of the learned representations.

RQ2. Furthermore, we investigate the impact of varying data augmentation techniques on CTRL. Figure 5 displays the classification accuracy outcomes for various data augmentation on ElectricDevices dataset. Replacing our masking strategy with timestamp-level random masking (→ *Bernoulli Mask*) leads to decreased performance, indicating that excessively short masking length diminishes the model's ability to capture essential abstract information. Additionally, incorpo-

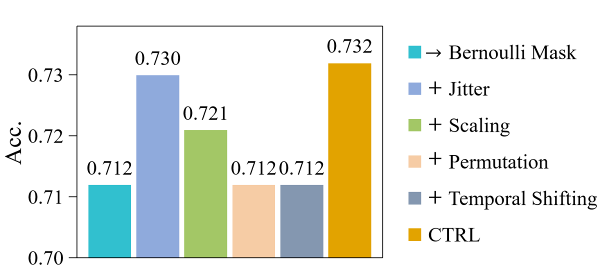


Figure 5: Ablation study of different augmentations.

	Metrics	Dilated CNN	LSTM	Transf.	NCDE
	Param.	637K	413K	416K	116K
Elec.	Acc.	0.726	0.725	0.570	**0.732**
Power.	Acc.	0.978	0.844	0.833	**0.994**
Exch.	MSE	1.031	0.671	0.411	**0.371**

Table 4: Ablation study of various backbone encoders.

	Random	+ FNE	+ HNC	+ Both (Ours)
CTRL	0.707	0.717	0.718	**0.722** (+2.1%)
TS2Vec	0.683	0.710	0.702	**0.719** (+5.3%)

Table 5: Ablation study of debiased contrasting.

rating augmentations used in other CL methods, such as jitter, scaling, permutation [Eldele *et al.*, 2021], and temporal shifting [Liu *et al.*, 2021], also results in performance degradation. As previously mentioned, these augmentations assume certain invariance assumptions that do not hold for diverse and ever-changing distributions of time series. This emphasizes the robustness of our masking strategy and its compatibility with our NCDE-based framework, as additional augmentations appear to adversely affect performance.

RQ3. To justify our choice of the backbone, we replace the NCDE with the latest popular backbones, including Dilated CNN [Yue *et al.*, 2022; Woo *et al.*, 2022], LSTM [Tonekaboni *et al.*, 2021] and Transformer [Zerveas *et al.*, 2021], whose settings refer to the default values in their public code. The classification results on ElectricDevices and PowerCons and forecasting results on Exchange Rate are shown in Table 4. Remarkably, the NCDE exhibits superior performance despite its notably fewer parameters.

RQ4. We also perform an ablation study to analyse the efficacy of our debiased contrastive learning framework. We name different variants of the negative sampling method in CL as follows: *Random* utilizes other samples within the batch as negatives, a practice most frequently used. *+FNE (False Negative Elimination)* builds upon the *Random* method by integrating our false negative elimination technique. *+HNE (Hard Negative Construction)* enhances *Random* by incorporating our hard negative construction strategy. *+Both* includes constructing hard negatives and executing false negative elimination on all negative samples. Table 5 presents classification accuracy results from different variants on the Heartbeat dataset. The results show that each operation, whether FNE or HNC, independently contributes to enhancement, and their combined application results in superior overall performance. Moreover, our debiased contrastive learning framework demonstrates broad applicability to other time series contrastive learning methods. When applied to TS2Vec model, our *+Both* variant yields a 5.3% performance improvement compared to its original version (*Random*). Considering the periodic nature of time-series data and inter-time series correlations, random negative sampling often results in numerous false negatives. Hence, false negative elimination becomes crucial for time series data. Experiments indicate that our FNE takes up only 0.413% of total training time. At the same time, our hard negative construction method supplements the negative sample set to prevent the quantity of negatives from diminishing excessively.

Tasks	Forecasting	Classification	Imputation
Metric	Avg. MSE	Avg. Acc.	Avg. MSE
LSTM	2.105	0.542	0.989
TCN	1.587	0.723	0.516
Informer	1.550	0.741	0.071
Autoformer	0.613	0.622	0.051
FEDfromer	0.519	0.661	0.061
DLinear	**0.354**	0.726	0.093
TimesNet	0.416	**0.747**	**0.027**
CTRL	0.412	0.727	0.039

Table 6: CTRL results compared to supervised methods.

5.9 Compared to Supervised Methods

We further compare CTRL with end-to-end supervised methods. Table 6 summarizes the average metrics of forecasting results on the Exchange Rate dataset, classification results on 8 datasets, and imputation results on the ETTm1 dataset. CTRL obtains remarkable performance, ranking second in forecasting and imputation tasks, while also displaying competitive performance in classification tasks. These strongly support the representation learning capability of CTRL.

6 Conclusion

In this work, we proposed, for the first time, an NCDE-based framework for universal representation learning of time series, named CTRL. The evaluation of the learned representations on time series classification, forecasting and imputation tasks demonstrated the universality and effectiveness of CTRL. Remarkably, CTRL showed superior stability when handling missing data compared to existing SOTA methods. The universality of CTRL makes it a promising candidate with ample potential for various applications and future research endeavors.

Acknowledgements

This work was supported by the National Key R&D Program of China (2021ZD0111201).

References

[Bagnall *et al.*, 2018] Anthony Bagnall, Hoang Anh Dau, Jason Lines, Michael Flynn, James Large, Aaron Bostrom, Paul Southam, and Eamonn Keogh. The uea multivariate time series classification archive, 2018. *arXiv preprint arXiv:1811.00075*, 2018.

[Bai *et al.*, 2018] Shaojie Bai, J Zico Kolter, and Vladlen Koltun. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. *arXiv preprint arXiv:1803.01271*, 2018.

[Chen *et al.*, 2018] Ricky TQ Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. Neural ordinary differential equations. *Advances in neural information processing systems*, 31, 2018.

[Chen *et al.*, 2020] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR, 2020.

[Chowdhury *et al.*, 2022] Ranak Roy Chowdhury, Xiyuan Zhang, Jingbo Shang, Rajesh K Gupta, and Dezhi Hong. Tarnet: Task-aware reconstruction for time-series transformer. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Washington, DC, USA*, pages 14–18, 2022.

[Dau *et al.*, 2019] Hoang Anh Dau, Anthony Bagnall, Kaveh Kamgar, Chin-Chia Michael Yeh, Yan Zhu, Shaghayegh Gharghabi, Chotirat Ann Ratanamahatana, and Eamonn Keogh. The ucr time series archive. *IEEE/CAA Journal of Automatica Sinica*, 6(6):1293–1305, 2019.

[Dormand and Prince, 1980] John R Dormand and Peter J Prince. A family of embedded runge-kutta formulae. *Journal of computational and applied mathematics*, 6(1):19–26, 1980.

[Eldele *et al.*, 2021] Emadeldeen Eldele, Mohamed Ragab, Zhenghua Chen, Min Wu, Chee Keong Kwoh, Xiaoli Li, and Cuntai Guan. Time-series representation learning via temporal and contextual contrasting. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence*, pages , 2352–2359, 2021.

[Franceschi *et al.*, 2019] Jean-Yves Franceschi, Aymeric Dieuleveut, and Martin Jaggi. Unsupervised scalable representation learning for multivariate time series. *Advances in neural information processing systems*, 32, 2019.

[He *et al.*, 2020] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9729–9738, 2020.

[He *et al.*, 2022] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16000–16009, 2022.

[Joshi *et al.*, 2020] Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S Weld, Luke Zettlemoyer, and Omer Levy. Spanbert: Improving pre-training by representing and predicting spans. *Transactions of the Association for Computational Linguistics*, 8:64–77, 2020.

[Kalantidis *et al.*, 2020] Yannis Kalantidis, Mert Bulent Sariyildiz, Noe Pion, Philippe Weinzaepfel, and Diane Larlus. Hard negative mixing for contrastive learning. *Advances in Neural Information Processing Systems*, 33:21798–21809, 2020.

[Kenton and Toutanova, 2019] Jacob Devlin Ming-Wei Chang Kenton and Lee Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT*, pages 4171–4186, 2019.

[Kidger *et al.*, 2020] Patrick Kidger, James Morrill, James Foster, and Terry Lyons. Neural controlled differential equations for irregular time series. *Advances in Neural Information Processing Systems*, 33:6696–6707, 2020.

[Kidger, 2022] Patrick Kidger. On neural differential equations. *arXiv preprint arXiv:2202.02435*, 2022.

[Lai *et al.*, 2018] Guokun Lai, Wei-Cheng Chang, Yiming Yang, and Hanxiao Liu. Modeling long-and short-term temporal patterns with deep neural networks. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 95–104, 2018.

[Liu *et al.*, 2021] Xu Liu, Yuxuan Liang, Yu Zheng, Bryan Hooi, and Roger Zimmermann. Spatio-temporal graph contrastive learning. *arXiv preprint arXiv:2108.11873*, 2021.

[Nie *et al.*, 2023] Yuqi Nie, Nam H. Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. A time series is worth 64 words: Long-term forecasting with transformers, 2023.

[Sun *et al.*, 2019] Yu Sun, Shuohuan Wang, Yukun Li, Shikun Feng, Xuyi Chen, Han Zhang, Xin Tian, Danxiang Zhu, Hao Tian, and Hua Wu. Ernie: Enhanced representation through knowledge integration. *arXiv preprint arXiv:1904.09223*, 2019.

[Sutskever *et al.*, 2014] Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*, pages 3104–3112, 2014.

[Tian *et al.*, 2020] Yonglong Tian, Dilip Krishnan, and Phillip Isola. Contrastive multiview coding. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XI 16*, pages 776–794. Springer, 2020.

[Tonekaboni *et al.*, 2021] Sana Tonekaboni, Danny Eytan, and Anna Goldenberg. Unsupervised representation learning for time series with temporal neighborhood coding. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*, 2021.

[Vaswani *et al.*, 2017] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[Woo *et al.*, 2022] Gerald Woo, Chenghao Liu, Doyen Sahoo, Akshat Kumar, and Steven Hoi. Cost: Contrastive learning of disentangled seasonal-trend representations for time series forecasting. *arXiv preprint arXiv:2202.01575*, 2022.

[Wu *et al.*, 2020] Sifan Wu, Xi Xiao, Qianggang Ding, Peilin Zhao, Ying Wei, and Junzhou Huang. Adversarial sparse transformer for time series forecasting. *Advances in Neural Information Processing Systems*, 33, 2020.

[Wu *et al.*, 2021] Haixu Wu, Jiehui Xu, Jianmin Wang, and Mingsheng Long. Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. *Advances in Neural Information Processing Systems*, 34:22419–22430, 2021.

[Wu *et al.*, 2022] Haixu Wu, Tengge Hu, Yong Liu, Hang Zhou, Jianmin Wang, and Mingsheng Long. Timesnet: Temporal 2d-variation modeling for general time series analysis. *arXiv preprint arXiv:2210.02186*, 2022.

[Yue *et al.*, 2022] Zhihan Yue, Yujing Wang, Juanyong Duan, Tianmeng Yang, Congrui Huang, Yunhai Tong, and Bixiong Xu. Ts2vec: Towards universal representation of time series. *Proceedings of the AAAI Conference on Artificial Intelligence*, 36(8):8980–8987, 2022.

[Zeng *et al.*, 2023] Ailing Zeng, Muxi Chen, Lei Zhang, and Qiang Xu. Are transformers effective for time series forecasting? *Proceedings of the AAAI conference on artificial intelligence*, 37(9):11121–11128, 2023.

[Zerveas *et al.*, 2021] George Zerveas, Srideepika Jayaraman, Dhaval Patel, Anuradha Bhamidipaty, and Carsten Eickhoff. A transformer-based framework for multivariate time series representation learning. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pages 2114–2124, 2021.

[Zhang *et al.*, 2023] Kexin Zhang, Qingsong Wen, Chaoli Zhang, Rongyao Cai, Ming Jin, Yong Liu, James Zhang, Yuxuan Liang, Guansong Pang, Dongjin Song, et al. Self-supervised learning for time series analysis: Taxonomy, progress, and prospects. *arXiv preprint arXiv:2306.10125*, 2023.

[Zhou *et al.*, 2021] Haoyi Zhou, Shanghang Zhang, Jieqi Peng, Shuai Zhang, Jianxin Li, Hui Xiong, and Wancai Zhang. Informer: Beyond efficient transformer for long sequence time-series forecasting. *Proceedings of the AAAI conference on artificial intelligence*, 35(12):11106–11115, 2021.

[Zhou *et al.*, 2022] Tian Zhou, Ziqing Ma, Qingsong Wen, Xue Wang, Liang Sun, and Rong Jin. Fedformer: Frequency enhanced decomposed transformer for long-term series forecasting. In *International Conference on Machine Learning*, pages 27268–27286. PMLR, 2022.