# Few-Shot Time Series Classification

Ninduwara K.G.M.
*Department of Computer Science and Engineering*
*University of Moratuwa*
Colombo, Sri Lanka
maneth.21@cse.mrt.ac.lk

Uthayasanker Thayasivam
*Department of Computer Science and Engineering*
*University of Moratuwa*
Colombo, Sri Lanka
rtuthaya@cse.mrt.ac.lk

***Abstract*—Time series classification remains a challenging task in machine learning, particularly when labeled data is scarce. While self-supervised learning methods like Temporal Neighborhood Coding (TNC) have demonstrated success in learning robust representations from unlabeled time series data, their performance in few-shot learning scenarios remains underexplored. This paper investigates the integration of TNC with few-shot classification methods to address extreme data scarcity conditions. We evaluate seven classification approaches on synthetic multivariate time series data generated from Hidden Markov Models, including baseline methods (k-NN, linear classifier, standard prototypical networks) and four enhanced prototypical variants that introduce learnable transformations and adaptive mechanisms. Experiments across 1-shot to 20-shot scenarios in a 4-way classification task demonstrate that TNC representations exhibit strong linear separability and natural clustering properties. The Metric-Prototypical Networks method, which employs a learnable distance metric, achieved the highest performance with 77.4% accuracy in the 5-shot setting and 76.9% in the 10-shot setting, outperforming both baseline approaches and other enhanced variants. Our results indicate that combining self-supervised temporal representation learning with appropriate few-shot classification strategies provides an effective solution for time series classification under severe data constraints, with learnable metrics offering advantages over fixed Euclidean distances in the embedding space.**

***Index Terms*—time series classification, few-shot learning, self-supervised learning, temporal neighborhood coding, prototypical networks, contrastive learning**

## I. Introduction

Time Series Classification (TSC) is a critical task in machine learning with wide-ranging applications, from medical diagnosis using EEG signals to financial forecasting and industrial sensor monitoring. Historically, methods like Dynamic Time Warping (DTW) have been effective but are often computationally intensive. The advent of deep learning has introduced powerful models like Recurrent Neural Networks (RNNs) and Convolutional Neural Networks (CNNs) that can automatically learn hierarchical features from raw time series data.

However, the performance of these supervised models is heavily reliant on large, meticulously labeled datasets, which are often expensive or impractical to acquire. To address this limitation, self-supervised learning has emerged as a powerful paradigm. Self-supervised learning (SSL) methods learn meaningful data representations from unlabeled data, which can then be used for downstream tasks with much less labeled data.

A state-of-the-art SSL method in the time series domain is Temporal Neighborhood Coding (TNC). TNC utilizes a contrastive learning objective to learn robust representations by ensuring that embeddings from a signal's temporal neighborhood are closer to each other than to embeddings from other, dissimilar signals.

While TNC has proven effective for representation learning, its performance under data-scarce conditions, specifically in few-shot learning scenarios, remains an area ripe for exploration. In a few-shot setting, a model must learn to classify new categories given only a handful of examples. This project aims to bridge this gap by enhancing the TNC framework to improve its performance on few-shot time series classification tasks.

The primary contribution of this work is the improved TNC architecture that integrates Prototypical Networks for few-shot classification.

## II. Related Work

Unsupervised representation learning for time series has recently gained attention as a critical research area in machine learning. Unlike supervised learning, which requires large amounts of labeled data, unsupervised learning focuses on extracting meaningful patterns without labels. This is especially valuable in real-world applications such as healthcare, where obtaining high-quality annotations is costly or impractical. Time series data present unique challenges such as high dimensionality, non-stationarity, and variable lengths, making representation learning both difficult and important. The following section reviews the major categories of methods and key state-of-the-art contributions.

### A. Autoencoder-Based Methods

Early research applied autoencoders and sequence-to-sequence models to time series. These models jointly train an encoder and decoder to reconstruct the input, forcing the encoder to capture compressed representations. Examples include Choi et al. [1], Amiriparian et al. [2], and Malhotra et al. [3]. Variational autoencoders (VAEs) were later introduced to encourage disentangled and interpretable features. However, direct reconstruction of complex signals proved challenging, leading to a shift toward methods that learn representations without reconstruction objectives.

The fundamental limitation of autoencoder-based approaches lies in their pixel-level reconstruction objective, which may not align with downstream task requirements. While reconstruction forces the encoder to preserve information, it does not guarantee that the learned representations are discriminative or semantically meaningful for classification tasks. Additionally, autoencoders struggle with high-frequency noise and complex temporal dynamics, often learning to reproduce noise rather than extracting meaningful patterns. These limitations motivated the development of alternative self-supervised objectives that do not rely on explicit reconstruction.

### B. Contrastive Learning Approaches

Contrastive learning has become a dominant self-supervised strategy. The key idea is to bring representations of similar samples (positives) closer, while pushing apart dissimilar ones (negatives). Although initially popular in computer vision, adapting contrastive frameworks to time series required careful treatment of temporal dependencies and augmentations.

Contrastive Predictive Coding (CPC) [4] introduced the idea of predicting future latent representations from past contexts. CPC works well on structured datasets like Human Activity Recognition but struggles with highly non-stationary signals. The model uses a unidirectional architecture that predicts multiple future steps, maximizing mutual information between context and future representations. While CPC demonstrated strong performance on audio and vision tasks, its application to time series revealed limitations in handling abrupt state changes and long-range dependencies.

Triplet Loss (T-Loss) [5] proposed a triplet-based sampling strategy tailored for time series, using causal dilated convolutions for variable-length inputs. While effective in simple settings, T-Loss often fails when states are generated from similar dynamics. The method samples anchor, positive, and negative triplets based on temporal proximity, but this simple heuristic can introduce false negatives when similar states occur at distant time points.

Temporal Neighborhood Coding (TNC) [6] addressed these issues directly. TNC defines temporal neighborhoods as segments with locally stationary properties and employs a debiased contrastive loss inspired by Positive-Unlabeled learning. By treating non-neighboring samples as unlabeled rather than strictly negative, TNC reduces false negatives and learns smoother temporal embeddings. The encoder in TNC can be an RNN or CNN depending on the dataset. TNC demonstrated superior performance over CPC and T-Loss, particularly in medical applications where non-stationarity is common. The key innovation of TNC is its use of statistical tests like the Augmented Dickey-Fuller test to dynamically determine temporal neighborhoods, allowing the model to adapt to local signal properties rather than relying on fixed window sizes.

TS-TCC [7] extended contrastive methods with a dual-module framework that combines temporal and contextual contrasting, often with a Transformer backbone. This model outperformed CPC and SimCLR, especially in few-label and transfer scenarios. The temporal contrasting module learns representations by distinguishing between different temporal augmentations of the same instance, while the contextual contrasting module operates across different instances. This dual approach provides complementary supervision signals that improve representation quality.

TS2Vec [8] pushed the field further by introducing hierarchical contrastive learning at multiple semantic levels (timestamp, instance). It demonstrated strong improvements across classification, forecasting, and anomaly detection, achieving state-of-the-art performance on both UEA and UCR benchmarks. TS2Vec applies contrastive learning across different temporal scales, from individual timestamps to entire sequences, enabling the model to capture both fine-grained and global temporal patterns. The hierarchical approach allows the model to learn representations that are robust to different types of temporal variations.

BTSF [9] incorporated both temporal and spectral features via bilinear fusion, arguing that spectral properties are often ignored in prior work. This approach improved performance across tasks and emphasized the value of frequency-domain information. By combining time-domain and frequency-domain representations, BTSF captures complementary aspects of temporal dynamics that pure time-domain methods may miss.

SimCLR, originally developed for computer vision, has also been adapted for time series by applying various augmentation techniques such as jittering, scaling, rotation, and permutation. However, the effectiveness of these augmentations varies significantly across different types of time series data, and careful selection is required to avoid destroying semantic information.

### C. Architectures for Time Series Representation Learning

Different frameworks adopt different encoders. Recurrent Neural Networks (RNNs) are intuitive but suffer from vanishing gradients and limited parallelization. Long Short-Term Memory (LSTM) networks and Gated Recurrent Units (GRUs) were developed to address the vanishing gradient problem, enabling the capture of longer-range dependencies. However, RNNs remain fundamentally sequential, limiting their computational efficiency on modern parallel hardware.

Dilated Convolutional Neural Networks (CNNs), used in T-Loss and TS2Vec, capture long-range dependencies more efficiently. Dilated convolutions exponentially expand the receptive field without increasing the number of parameters, making them particularly suitable for long time series. Residual connections in deep CNNs further improve gradient flow and enable the training of very deep architectures. Temporal Convolutional Networks (TCNs) combine dilated convolutions with residual connections, achieving strong performance across various time series tasks while maintaining computational efficiency.

Transformers have also been explored, such as in TS-TCC, but CNN backbones often remain more efficient for long time series. The self-attention mechanism in Transformers enables direct modeling of long-range dependencies, but the quadratic complexity with respect to sequence length poses computational challenges. Various modifications have been proposed to reduce this complexity, including sparse attention patterns and local attention windows. Despite these innovations, for very long time series with thousands of time steps, CNN-based architectures often provide better trade-offs between performance and computational cost.

Recent work has also explored hybrid architectures that combine the strengths of different encoder types. For example, some methods use CNNs for local feature extraction followed by Transformers for global dependency modeling, or employ RNNs for temporal encoding with attention mechanisms for selective information aggregation.

### D. Evaluation Metrics and Benchmarks

Representation learning methods are typically evaluated on downstream tasks:

**Classification:** Accuracy and AUPRC on datasets like UCR and UEA archives. The UCR Time Series Classification Archive contains univariate datasets spanning diverse domains including medical diagnostics, motion tracking, sensor readings, and synthetic data. The UEA archive extends this with multivariate time series datasets. Standard evaluation protocols involve training linear classifiers or nearest-neighbor methods on frozen representations to assess their quality independently of classifier complexity.

**Clustering:** Silhouette score and Davies-Bouldin index measure how well the learned representations naturally group similar time series without supervision. Higher silhouette scores indicate better-separated and more compact clusters, while lower Davies-Bouldin indices indicate better cluster separation.

**Forecasting:** Mean Squared Error (MSE) on benchmarks such as ETT and Weather. Forecasting tasks evaluate whether learned representations capture predictive temporal patterns. The Electricity Transformer Temperature (ETT) dataset and various weather datasets are commonly used benchmarks.

**Anomaly Detection:** Precision, Recall, and F1 score on datasets like Yahoo KPI and SWaT. Anomaly detection tests whether representations can distinguish normal patterns from abnormal events, which is critical for applications like industrial monitoring and cybersecurity.

**Transfer Learning:** Performance when representations learned on one dataset are applied to different but related datasets, measuring generalization capability across domains.

### E. Few-Shot Learning in Time Series

Few-shot learning (FSL) is another critical direction, addressing scenarios where only a handful of labeled samples are available per class. FSL methods are typically grouped into:

**Metric-Based Learning:** Prototypical Networks [10], which compute class prototypes and classify queries based on distance. Matching Networks extend this by incorporating attention mechanisms and episodic training. Relation Networks learn to compare samples using a neural network rather than fixed distance metrics. These methods are particularly effective when the embedding space naturally clusters similar examples together.

**Optimization-Based Learning:** Model-Agnostic Meta-Learning (MAML), which adapts model parameters with a few gradient steps. MAML learns initial parameters that can be quickly fine-tuned to new tasks with minimal data. Extensions like Reptile and ANIL simplify the meta-learning objective while maintaining competitive performance. These methods require task distributions during meta-training and optimize for rapid adaptation rather than direct classification.

**Data Augmentation Approaches:** Which expand limited labeled data through transformations. For time series, augmentation strategies include time warping, magnitude scaling, jittering, window slicing, and mixup techniques that interpolate between examples. However, aggressive augmentation can destroy temporal semantics, requiring careful validation that augmented samples remain realistic and preserve class labels.

**Hallucination-Based Methods:** Generate synthetic samples from limited examples using generative models or learned transformations. These approaches aim to expand the support set but risk introducing unrealistic or biased samples if the generative model is not well-calibrated.

In time series, metric-based approaches are particularly attractive due to their simplicity and robustness under small data regimes. They do not require multiple gradient steps per episode, making them computationally efficient. Additionally, the episodic training paradigm naturally aligns with real-world deployment scenarios where models encounter new classes with minimal examples.

Recent work has begun exploring few-shot learning specifically for time series. ProtoTime combines prototypical networks with learned time warping to handle temporal misalignment. TapNet introduces attentive prototypes that weight different time steps based on their discriminative importance. However, most few-shot time series methods still rely on supervised pre-training rather than self-supervised representations, limiting their applicability when large labeled pre-training datasets are unavailable.

### F. Research Gap

Despite strong progress, most existing self-supervised time series models, including TNC, were not explicitly designed for few-shot classification. Their typical evaluation relies on training a linear classifier on top of frozen embeddings. While this works when ample labels are available, it becomes suboptimal under extreme data scarcity. There is a clear opportunity to integrate few-shot methods such as Prototypical Networks directly into frameworks like TNC. This combination would better exploit learned representations in low-label settings, providing a more sample-efficient solution to Time Series Classification.

The literature shows a clear evolution from reconstruction-based methods to advanced contrastive learning and hybrid approaches that incorporate temporal, contextual, and spectral information. Methods like TNC, TS2Vec, and BTSF represent the cutting edge of unsupervised time series representation learning. However, the integration of these powerful encoders with few-shot classification strategies remains underexplored.

Existing self-supervised methods typically evaluate their representations by training linear classifiers with abundant labeled data, which does not reflect realistic deployment scenarios where annotations are scarce. While some work has explored semi-supervised learning with partially labeled data, the extreme case of few-shot learning—where only 1-5 examples per class are available—has received limited attention in the context of self-supervised time series representations.

Furthermore, most few-shot time series classification methods assume supervised pre-training on related tasks, which may not be feasible in specialized domains like rare disease diagnosis or emerging industrial processes. The combination of self-supervised representation learning with few-shot classification offers a promising solution: learn general temporal patterns from abundant unlabeled data, then rapidly adapt to new classification tasks with minimal labels.

Addressing this gap forms the motivation for our project, which enhances TNC by replacing its standard linear classifier with a Prototypical Network head, aiming to improve performance in few-shot time series classification tasks. Additionally, we explore enhanced prototypical methods that introduce learnable components to better leverage the specific properties of TNC representations, including learnable distance metrics, feature transformations, and adaptive architectures.

## III. Methodology

The proposed approach [11] extends the Temporal Neighborhood Coding (TNC) framework by incorporating few-shot classification strategies. The methodology comprises two principal phases: self-supervised representation learning through pre-training, followed by few-shot classification using both baseline and enhanced methods. This two-stage pipeline enables the evaluation of learned representations under realistic low-data scenarios.

### A. Self-Supervised Representation Learning

The first phase focuses on learning robust temporal representations without supervision through contrastive learning. This pre-training stage establishes the foundation for subsequent few-shot classification tasks.

*1) **Dataset Generation**:* The dataset was synthetically generated to emulate long, non-stationary multivariate time series commonly encountered in real-world applications. Each sequence contained 2000 measurements with three features, governed by four latent states sampled from a Hidden Markov Model (HMM). Within each state, data were generated using either a Gaussian Process with different kernels or a Nonlinear Auto-Regressive Moving Average (NARMA) model. To better resemble realistic signals, two features were maintained as correlated throughout the sequence, introducing dependencies that challenge representation learning.

*2) **Encoder Training**:* The encoder was optimized using the contrastive objective from the TNC framework. The model learned to distinguish temporally neighboring segments (positive pairs) from distant segments (negative pairs), where neighborhoods were dynamically defined using the Augmented Dickey-Fuller (ADF) test for stationarity detection. A bidirectional single-layer RNN encoder was employed to capture temporal dependencies in both forward and backward directions. The architecture encoded windows of size 50 into 10-dimensional representation vectors, carefully balancing the trade-off between capturing sufficient state information and maintaining temporal locality.

*3) **Pre-training Outcome**:* After training convergence, the encoder produced embeddings that clearly separated the underlying signal types, achieving a class separation ratio of 1.648. This metric indicates strong discriminative power in the learned representation space. The encoder weights were subsequently frozen and utilized for all downstream few-shot classification experiments, ensuring consistent feature extraction across different methods.

### B. Few-Shot Classification

To evaluate the quality and transferability of the learned TNC representations, a comprehensive few-shot classification framework was designed. The primary objective was to assess the encoder's ability to generalize to new classification tasks with minimal labeled data, which is central to the promise of self-supervised representation learning. Seven distinct classification approaches were implemented, ranging from simple baselines to sophisticated meta-learning techniques.

*1) **Baseline Methods**:* Three baseline methods were employed to establish performance benchmarks and evaluate fundamental properties of the learned embedding space.

*a) **Linear Classifier**:* A logistic regression model was trained directly on the frozen TNC feature representations, serving as a baseline for assessing linear separability of the embeddings. The model employed scikit-learn's default L2 regularization with regularization strength C=1.0 to prevent overfitting in low-data regimes. Training was performed using the Limited-memory BFGS (L-BFGS) optimizer with a maximum of 1000 iterations, which ensures stable convergence with small datasets and provides insights into the linear structure of the learned embedding space. Each trial used a different random seed to ensure reproducible yet varied sampling of the few-shot support sets.

*b) **k-Nearest Neighbors**:* A non-parametric classifier was implemented where labels are assigned based on the majority vote of the $k$ nearest neighbors in the TNC feature space. Euclidean distance in the 10-dimensional embedding space was used as the similarity metric, with the optimal value of $k$ determined through cross-validation. This approach evaluates whether semantically similar time series naturally cluster together in the learned representation space without requiring any model training.

*c) **Standard Prototypical Networks**:* This baseline few-shot method computes class prototypes as the mean of support set embeddings for each class. Query samples are then classified based on their Euclidean distance to the nearest prototype. This approach directly tests the natural clustering properties of the TNC feature space without requiring any task-specific adaptation or fine-tuning.

*2) **Enhanced Prototypical Methods**:* To leverage the characteristics of TNC representations more effectively, four enhanced prototypical methods were developed. These methods introduce learnable components that adapt to the specific properties of temporal embeddings.

*a) **Linear Prototypical Networks**:* This method introduces a learnable two-layer neural transformation that projects TNC features into an optimized prototype space before computing class centroids. The transformation consists of sequential linear layers with ReLU activation and dropout regularization, using Xavier normal initialization for stable training. The network learns to transform features into a representation space where Euclidean distances between query examples and class prototypes are more discriminative, exploiting the linear separability of TNC embeddings through learned feature transformation with a fixed temperature parameter for logit conversion.

*b) **Metric Prototypical Networks**:* Rather than relying on fixed Euclidean distance, this approach employs a learnable neural network that computes distance scores between query samples and class prototypes. The network comprises three fully connected layers with ReLU activation and dropout regularization, followed by a sigmoid activation producing distance values between 0 and 1. The architecture concatenates query and prototype features as input, enabling task-specific distance modeling that can capture non-linear relationships in the embedding space through learned distance metrics rather than similarity scores.

*c) **Hybrid Few-Shot Classifier**:* This method combines prototypical and linear classifiers through a learnable parameter-based fusion mechanism. The fusion uses a single sigmoid-activated parameter for static weighted combination of both approaches. The linear branch employs a two-layer neural network with ReLU activation and dropout, while the prototypical branch uses feature transformation before computing distance-based logits with learnable temperature. This hybrid approach leverages complementary strengths through fixed weighted fusion.

*d) **Adaptive Prototypical Networks**:* This method dynamically adjusts prototype computation and feature transformation based on the number of available shots. The approach incorporates multiple learnable linear transformations with ReLU activation and dropout regularization, using different transformation pathways for varying shot numbers. Xavier normal weight initialization ensures stable training, and the method adapts its architectural complexity based on shot availability rather than modulating reliance on individual examples versus class statistics during prototype formation.

*3) **Experimental Protocol**:* All methods were evaluated using the standard N-way, K-shot episodic protocol, which simulates realistic few-shot learning scenarios. The experimental design ensures rigorous and reproducible evaluation across all classification approaches.

*a) **Task Configuration**:* The focus was placed on 4-way classification tasks, distinguishing between four distinct signal types: Periodic Gaussian Process, NARMA-5, Squared Exponential Gaussian Process, and NARMA-3. This configuration represents a challenging multi-class scenario that tests the discriminative power of learned representations.

*b) **Episode Structure**:* Each evaluation episode randomly selected 4 classes from the available signal types. For each selected class, K support examples were sampled for training and 15 query examples were sampled for testing. This structure ensures consistent evaluation across all methods while maintaining independence between support and query sets.

*c) **Data Availability Scenarios**:* Experiments were conducted across five different shot settings: 1-shot, 3-shot, 5-shot, 10-shot, and 20-shot. These variations simulate realistic scenarios with varying levels of labeled data availability. The 1-shot scenario represents the most challenging extreme low-data condition, while the 20-shot setting provides more substantial adaptation data for methods that can leverage larger support sets.

*d) **Statistical Validation**:* Each configuration was evaluated over 50 independent episodes with different random class and sample selections.

*e) **Feature Consistency**:* All methods utilized identical 10-dimensional embeddings extracted from the frozen TNC encoder. This design choice ensures that performance differences are attributable solely to the classification methods rather than variations in feature quality or extraction procedures.

*f) **Evaluation Metrics**:* Classification accuracy served as the primary performance metric, computed as the proportion of correctly classified query samples. This was complemented by confusion matrix analysis and per-class accuracy breakdowns to identify potential biases across different signal types and to understand method-specific strengths and weaknesses.

This comprehensive methodology provides a rigorous framework for assessing both the intrinsic quality of TNC representations and the effectiveness of diverse few-shot classification strategies for temporal data analysis.

## IV. Experimental Results

Table I presents the few-shot classification accuracy across all evaluated methods and shot configurations. The results demonstrate the effectiveness of learned TNC representations and reveal interesting trade-offs between different classification approaches.

TABLE I
FEW-SHOT CLASSIFICATION ACCURACY ON SIMULATED DATASET

| **Method** | **1-shot** | **3-shot** | **5-shot** | **10-shot** | **20-shot** |
|---|---|---|---|---|---|
| Prototypical Networks | 0.566 | 0.699 | 0.718 | 0.733 | 0.765 |
| k-NN Baseline | 0.357 | 0.621 | 0.659 | 0.704 | 0.725 |
| Linear Baseline | 0.555 | 0.691 | 0.721 | 0.751 | 0.778 |
| Linear-Prototypical | 0.578 | 0.727 | 0.724 | 0.743 | 0.752 |
| Metric-Prototypical | 0.587 | 0.729 | 0.774 | 0.769 | 0.772 |
| Hybrid-Prototypical | 0.580 | 0.691 | 0.740 | 0.727 | 0.756 |
| Adaptive-Prototypical | 0.531 | 0.672 | 0.705 | 0.719 | 0.648 |

Metric-Prototypical Networks achieved the best overall performance across most configurations, with particularly strong results in the 5-shot (0.774) and 10-shot (0.769) scenarios. The learnable similarity metric effectively captured task-specific relationships in the embedding space.

## V. Conclusion and Future Work

This work successfully demonstrates the integration of Temporal Neighborhood Coding with few-shot classification methods for time series analysis under data scarcity. Through evaluation of seven classification approaches, we established that TNC representations possess strong discriminative properties for few-shot learning. The Metric-Prototypical Networks method achieved the best performance with 77.4% accuracy in the 5-shot setting, demonstrating that learnable distance metrics effectively capture task-specific relationships in the embedding space better than fixed Euclidean distances.

Our results confirm that combining self-supervised pretraining with sophisticated few-shot strategies provides an effective solution for time series classification when labeled data is limited. Enhanced prototypical methods generally outperformed baselines in low-shot regimes, though the Adaptive-Prototypical method showed performance degradation at 20-shot, suggesting potential overfitting with excessive architectural complexity.

Several important limitations present opportunities for future research. First, this study relied entirely on synthetically generated data from Hidden Markov Models to train the TNC encoder due to computational resource constraints. Future work should validate the approach on real-world datasets such as ECG signals, Human Activity Recognition data, or other medical and industrial sensor recordings to assess practical generalization capabilities. Second, the few-shot evaluation was not conducted on standardized benchmarks. Future research should incorporate the UEA Time Series Classification Archive to rigorously evaluate transfer learning capabilities and cross-domain performance. Testing scenarios where the encoder is pre-trained on one dataset and evaluated on entirely different signal types would provide stronger evidence of the method's utility.

With greater computational resources, future investigations could explore the application of this framework to more challenging few-shot time series classification scenarios, including cross-domain transfer where models pre-trained on one type of temporal signal are evaluated on entirely different domains. Additional promising directions include investigating time series-specific data augmentation techniques tailored for few-shot scenarios, exploring optimization-based meta-learning approaches like MAML that explicitly optimize for rapid task adaptation, and developing hierarchical few-shot methods that leverage multi-scale temporal representations. Furthermore, analyzing the interpretability of learned prototypes and distance metrics would provide insights into what temporal patterns and dynamics are critical for successful few-shot classification in different application domains.RetryClaude can make mistakes. Please double-check responses.

## References


[1] Choi, J., Chiu, C., & Sontag, D. (2016). Learning low-dimensional representations of medical time series data using autoencoder-based models. Proceedings of the Machine Learning for Healthcare Conference.

[2] Amiriparian, S., Schmitt, M., & Weninger, F. (2017). Audio-visual emotion recognition via multi-channel autoencoders. Proceedings of the International Conference on Affective Computing and Intelligent Interaction.

[3] Malhotra, P., Vig, L., Shroff, G., & Agarwal, P. (2017). Long Short Term Memory networks for anomaly detection in time series. Proceedings of the European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning (ESANN).

[4] Oord, A. v. d., Li, Y., & Vinyals, O. (2018). Representation learning with contrastive predictive coding. arXiv preprint arXiv:1807.03748.

[5] Franceschi, J.-Y., et al. (2019). Unsupervised representation learning for time series with triplet loss. Proceedings of the 36th International Conference on Machine Learning (ICML).

[6] Tonekaboni, S., et al. (2021). Temporal Neighborhood Coding (TNC): Unsupervised representation learning for non-stationary time series. arXiv preprint arXiv:2106.00750.

[7] Eldele, E., et al. (2021). TS-TCC: A framework for unsupervised representation learning of time series. Proceedings of the AAAI Conference on Artificial Intelligence.

[8] Yue, C., Zhao, R., & Li, J. (2021). TS2Vec: Towards universal representation of time series. Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining(KDD).

[9] YYang, Z., & Hong, B. (2022). Unsupervised time-series representation learning via iterative bilinear temporal-spectral fusion (BTSF). IEEE Transactions on Neural Networks and Learning Systems.

[10] Snell, J., Swersky, K., & Zemel, R. (2017). Prototypical networks for few-shot learning. Advances in Neural Information Processing Systems (NeurIPS).

[11] Ninduwara K.G.M. "TNC-Few-Shot" GitHub. Available: https://github.com/ManethNin/TNC-Few-Shot Accessed: Oct. 20, 2025.