IEEE Open Journal of Signal Processing

Received 12 April 2024; revised 22 June 2024; accepted 24 June 2024. Date of publication 30 July 2024; date of current version 30 August 2024. The review of this article was arranged by Associate Editor U. A Khan.

Digital Object Identifier 10.1109/OJSP.2024.3435673

Self-Supervised Learning-Based Time Series Classification via Hierarchical Sparse Convolutional Masked-Autoencoder

TING YU [1,2], KELE XU [1,2] (Member, IEEE), XU WANG [1,2], BO DING [1,2], AND DAWEI FENG [1,2]

[1]College of Computer, National University of Defense Technology, Changsha 410073, China
[2]State Key Laboratory of Complex and Critical Software Environment, National University of Defense Technology, Changsha 410073, China

CORRESPONDING AUTHOR: KELE XU (email: xukelele@163.com).

This work was supported by the National University of Defense Technology under Grant ZZCX-ZZGC-01-04.

ABSTRACT In recent years, the use of time series analysis has become widespread, prompting researchers to explore methods to improve classification. Time series self-supervised learning has emerged as a significant area of study, aiming to uncover patterns in unlabeled data for richer information. Contrastive self-supervised learning, particularly, has gained attention for time series classification. However, it introduces inductive bias by generating positive and negative samples. Another approach involves Masked Autoencoders (MAE), which are effective for various data types. However, due to their reliance on the Transformer architecture, they demand significant computational resources during the pre-training phase. Recently, inspired by the remarkable advancements achieved by convolutional networks in the domain of time series forecasting, we aspire to employ convolutional networks utilizing a strategy of mask recovery for pre-training time series models. This study introduces a novel model termed Hierarchical Sparse Convolutional Masked-Autoencoder, "HSC-MAE", which seamlessly integrates convolutional operations with the MAE architecture to adeptly capture time series features across varying scales. Furthermore, the HSC-MAE model incorporates dedicated decoders that amalgamate global and local information, enhancing its capacity to comprehend intricate temporal patterns. To gauge the effectiveness of the proposed approach, an extensive array of experiments was conducted across nine distinct datasets. The experimental outcomes stand as a testament to the efficacy of HSC-MAE in effectively mitigating the aforementioned challenges.

INDEX TERMS Time series classification, self-supervised learning, time series pre-training.

I. INTRODUCTION

The applications of time series analysis are manifestly evident across various academic disciplines, including finance [1], meteorology [2], transportation [3], and biology [4]. These instances of application vividly underscore the remarkable versatility of time series analysis. Unlike traditional machine learning methods such as K-Nearest Neighbors (KNN) [5], decision trees [6], and Support Vector Machines (SVM) [7], deep learning methods automatically learn representations from data without the need for complex feature engineering [8]. However, deep learning methods heavily rely on a large amount of labeled data, which is difficult to obtain in real-world scenarios. This difficulty has led researchers to explore learning features from unlabeled data. Self-supervised learning has made remarkable progress in computer vision (CV) [9], [10] and natural language processing (NLP) [11], [12] domains, inspiring researchers in the time series domain. Self-supervised learning has gained substantial traction within the realm of time series analysis, constituting two fundamental categories: contrastive learning-based methods and generation-based methods. To elucidate, Franceschi et al. [13] treats sub-sequences from the same series as positive samples and sub-sequences from different series as negative samples. Tonekaboni et al. (TNC) [14] considers adjacent sub-sequences as positive pairs. Eldele et al. (TS-TCC) [15] partitions samples based on augmented sequences obtained from transforming time series and learns sequence representations with contextual information. Yue et al. (TS2Vec) [16] creates positive and negative samples from both the time dimension and the instance aspect, where sequences with augmentation on the same timestamp are considered positive pairs, and sequences from the same instance are positive samples. Zhang et al. (TF-C) [17] simultaneously utilizes time-domain and frequency-domain augmentation, combining time-frequency consistency for pre-training. Regardless of their specific methodologies, contrastive learning methods such as TNC, TS-TCC, TS2Vec, and TF-C [13], [14], [15], [16], [17], all share a common characteristic: they construct positive and negative samples based on predefined assumptions about data characteristics. The demarcation between positive and negative instances introduces inherent inductive biases, which, in turn, could potentially engender model predisposition.

Another major approach to time series classification involves the use of Transformer architecture centered around masking and prediction. This approach draws its inspiration from the notable achievements of BERT [18] in the realm of representation learning for natural language processing (NLP). Zerveas et al. (TST) [19] introduced the application of Transformer masking and prediction techniques to the domain of time series data. In TST, the original sequence is ingested as input, and subsequently, the obscured segments are sequentially restored. Conversely, Cheng et al. (TimeMAE) [20] considers computational complexity and the redundancy in time series data by partitioning the time series into patches and designing classification and regression pretext tasks.

However, Transformer-based models exhibit two primary limitations. Firstly, when confronted with lengthy sequences, the intricate nature of the self-attention mechanism leads to a substantial escalation in the computational and storage resources essential for both the training and inference phases. Secondly, conventional Transformers tend to disregard the multi-scale temporal dependencies inherent in time series data. The intricate relationships and patterns within a time series fluctuate across various temporal spans. Transformers prioritize global information interaction and modeling, relatively neglecting the local characteristics at different scales. In contrast, convolutional networks typically have fewer parameters, enabling efficient parallel computation, and are well-suited for extracting local features of time series data. Moreover, recent advancements in the field of time series forecasting, exemplified by PatchMixer [21], have demonstrated the effectiveness of convolutional networks in handling time series data. At the same time, as noted by Zhao et al. [22], convolution may cause the network to focus excessively on local information while relatively neglecting long-range dependencies. To fully leverage the potential advantages of CNNs over Transformer-based methods by effectively combining global and local information while mitigating the introduction of inductive bias, we propose the Hierarchical Sparse Convolutional Masked-Autoencoder, HSC-MAE, for learning features from time series data. The pivotal contributions of this article are outlined as follows:

1) Our study adeptly combines convolutional operations with the MAE-style approach during the time series pre-training phase by employing sparse convolution. We propose a simple yet effective architecture that enables the model to learn time series features from multi-scale perspectives, further validating the feasibility of convolutional networks in the field of time series analysis.
2) We propose novel pretext tasks: including the local recovery and global reconstruction. These tasks guide the model to simultaneously focus on both local and global information without introducing additional inductive biases.
3) To evaluate the effectiveness of our approach, we conduct comprehensive experiments on nine distinct datasets. Our method achieves state-of-the-art(SOTA) performance on five of these datasets. Additionally, the average accuracy and F1 scores across all nine datasets show improvements.

The remainder of this paper is organized as follows. Section II reviews time series classification and time series self-supervised learning. Section III provides a detailed presentation of our HSC-MAE method. Section IV comprehensively outlines the experimental details and analyzes the experimental results of our method. Finally, we conclude in the last section.

II. RELATED WORK

We will briefly review related works in the areas of time series classification and self-supervised learning for time series, pertinent to our research.

A. TIME SERIES CLASSIFICATION

Traditional machine learning algorithms mostly rely on statistical and distance-based approaches. Bagnall et al. [23] achieved favorable results using nearest neighbor classifiers [5] and distance functions such as Dynamic Time Warping (DTW) [24]. Lines et al. [25] proposed HIVE-COTE, combining multiple classifiers and hierarchical voting for classification. Rocket [26] utilizes multiple random convolutional kernels for feature extraction, followed by linear classifiers to obtain results. Proximity Forest [27], similar to the random forest [28], employs distance measures to compare unclassified samples with randomly chosen samples from each class, facilitating classification. However, traditional learning methods necessitate complex feature engineering. In contrast, deep learning methods are driven by data. Ismail [29], Zheng [30], Tan et al. [31], employ convolutional networks for time series analysis. MACNN [32] integrates attention mechanisms and multi-scale CNNs for classification. Wu et al. [33] combine time-frequency attention with frame-wise self-attention for sound classification. Wu et al. [34] utilize graph neural networks to extract temporal information. However, many of these methods heavily rely on labeled data, which is often challenging to acquire.

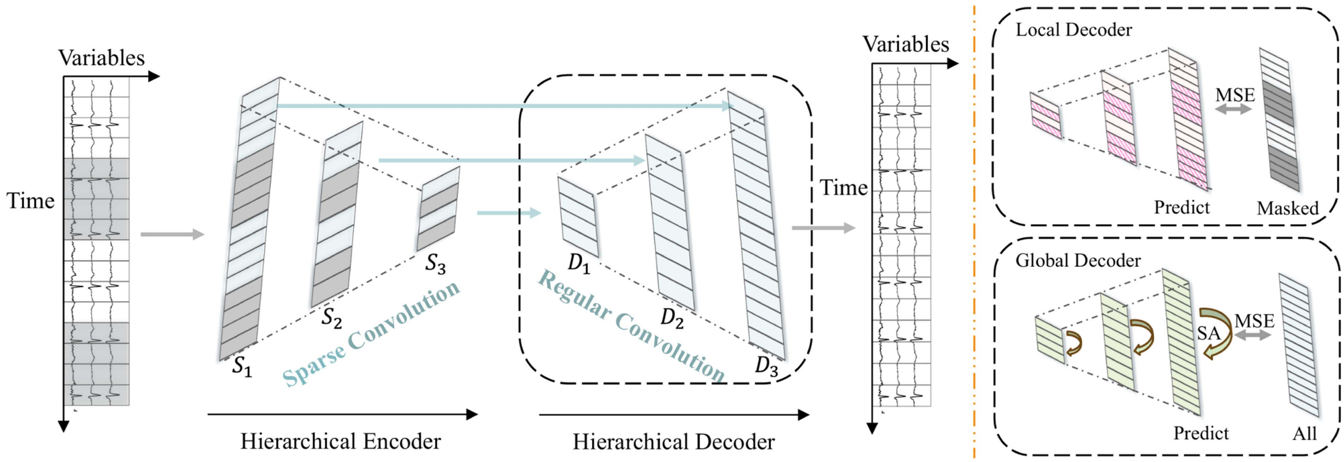


FIGURE 1. **Left: Hierarchical pre-training structure based on patch-wise sparse masking. Right: Configurations of the two pretext tasks. The local decoder recovers only the masked portions, while the global decoder reconstructs the entire sequence by applying self-attention to the features.**

B. SELF-SUPERVISED LEARNING FOR TIME SERIES

Given the difficulty in acquiring labeled data and the remarkable achievements of self-supervised pre-training in the domains of natural language understanding [35], [18] and computer vision [36], [37], [38], researchers in the field of time series have also turned their attention towards the paradigm of self-supervised learning. Reconstruction-based methods employ auto-encoders to recover the original sequences. For instance, TimeNet [39] uses recurrent neural networks for reconstruction. TST [19] employs Transformers to sequentially recover masked sequences. Similar to PatchTST [40], TimeMAE [20] utilizes Transformer to reconstruct sequences with continuous sub-sequences as basic units. Meanwhile, contrastive learning-based methods are prevalent. SelfTime [41] leverages different levels of samples and temporal relationships between sub-sequences, employing an architecture similar to SimCLR [37] for contrasting time series information. CoST [42] combines temporal consistency and frequency domain information, learning representations through a prediction-based approach. Deldari [43] maximizes shared information within continuous time intervals while minimizing shared information across non-adjacent time intervals to detect changes in time series. Hyvarinen [44] utilized logistic regression to learn temporal dependencies by contrasting sub-sequences of the original time series with randomly transformed time points. InfoTS [45] employs an information-aware augmentation approach and adaptively selects the optimal augmentation for learning time series representations. SimMTM [46] integrates mechanisms for contrastive learning and masked time-series modeling, utilizing series similarity to aggregate and reconstruct time series. Furthermore, TS-TCC [15], TS2Vec [16], TF-C [17], TNC [14], and others are not reiterated here. It is evident that the foundation of contrastive learning relies on prior assumptions about the data, such as determining whether there is an overlap between sub-sequences to distinguish positive and negative samples. This prior assumption requires human judgment, potentially introducing inductive bias. For example, TS2Vec [16] uses different samples at the same timestamp as negative samples for contrastive learning, constructing a contrastive loss. However, the assumption that different samples are necessarily dissimilar can sometimes fail, affecting the effectiveness of pre-training.

III. METHODOLOGY

A. OVERALL FRAMEWORK

Each multivariate time series sample, denoted as $\boldsymbol{X}_i$, is characterized by its sequence length T and feature dimension D. Our network aims to map $\boldsymbol{X}_i$ to $\boldsymbol{R}_i$, where $\boldsymbol{R}_i \in \mathbb{R}^{m\times k}$, with m and k respectively denoting the time and variable dimension of the representation vector. This mapping aims to ensure that $\boldsymbol{R}_i$ accurately captures the essential representation of $\boldsymbol{X}_i$, enabling general pre-training so that the model can be fine-tuned for various specific tasks. The overall architecture of HSC-MAE is illustrated in Fig. 1, which comprises the following key components: patch-wise sparse masking for pre-training, a hierarchical architecture with layered encoding and decoding, and a novel optimization strategy that captures both global and local features.

B. PATCH-WISE SPARSE MASKING FOR PRE-TRAINING

Patch Partition: Time series data often exhibit significant temporal redundancy between adjacent points, making it relatively easy to recover one point from its neighboring points [20]. Therefore, we adopted the methodologies proposed by PatchTST [40] and TimeMAE [20]. This approach involves segmenting the original time series data into non-overlapping patches using one-dimensional cross-channel convolution. Each patch is then treated as a fundamental unit for masking with a fixed probability, resulting in the transformation of each $\boldsymbol{X}_i$ into $\boldsymbol{Z}_i$ with $\boldsymbol{Z}_i \in \mathbb{R}^{l\times d}$, where l and d respectively represent the time and variable dimension of embedding.

Sparse Convolution: Unlike Transformer-based methods, convolutional networks lack built-in mechanisms for handling

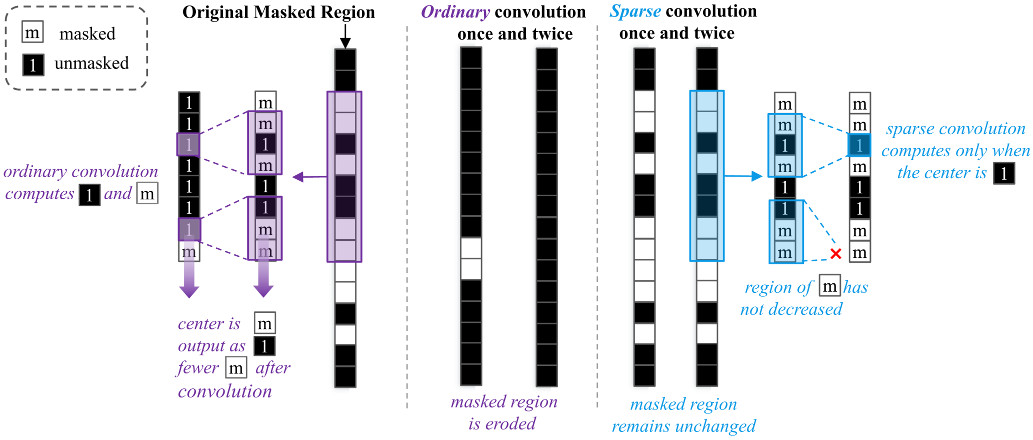


FIGURE 2. Comparison between sparse convolution and ordinary convolution. The figures are represented using binary masks, with the unmasked region denoted as 1. As depicted in the left, when using ordinary convolution, the result will be nonzero whenever the convolution kernel covers any unmasked point (value 1 position), leading to a reduction in the masked region. In contrast, as shown in the right, sparse convolution can skip masked positions, thereby avoiding the expansion of the unmasked region.

sequences of varying lengths. Directly setting masked positions to 0 and inputting them into the network carries the risk of distorting the underlying data distribution. Additionally, as multiple convolutional layers are applied, the initially masked region may gradually decrease in size [47]. To overcome these challenges, we draw inspiration from the techniques proposed in SparK [47] and ConvNeXt V2 [48]. Specifically, we employ sparse convolution for encoding during the masking stages. The comparison between ordinary convolution and sparse convolution is illustrated in Fig. 2. This choice ensures stability within the masked region and mitigates potential issues related to shifts in data distribution. Importantly, it should be noted that the integration of sparse convolution is limited to the encoder phase of the pre-training process. After pre-training is completed, sparse convolution is reverted to standard convolutional operations.

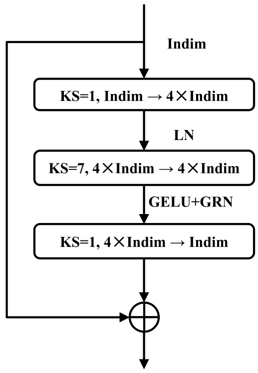


FIGURE 3. HSC-MAE Block Designs. Consistent with ConvNeXt [49], the design mimics a self-attention mechanism, following the principle of an inverted bottleneck. "Indim" represents the input dimension, and "KS" denotes the kernel size in the context of the convolution module. "LN" refers to Layer Normalization, "GELU" denotes the activation function, and "GRN" stands for Global Response Normalization, which aggregates global features for data normalization and calibration.

C. HIERARCHICAL ENCODER AND DECODER

Encoder: The HSC-MAE network employs a hierarchical encoding-decoding structure to incorporate time series patterns from multiple scales. The encoder network applies repeated convolution modules and down-sampling three times, with each down-sampling reducing the scale by a factor of 2. For input data $\boldsymbol{Z}_i \in \mathbb{R}^{l\times d}$, after each convolution block and down-sampling, the data scales become $\boldsymbol{S}_1 \in \mathbb{R}^{\frac{l}{2}\times h_1}$, $\boldsymbol{S}_2 \in \mathbb{R}^{\frac{l}{4}\times h_2}$, and $\boldsymbol{S}_3 \in \mathbb{R}^{\frac{l}{8}\times h_3}$, where h_1, h_2, and h_3 are representation dimensions at different scales. The configuration of the convolution module utilizes 1D convolution and follows the order of cross-channel, channel-split, and cross-channel convolution, with a maximum convolution kernel applied during channel-split convolution. The design of the convolutional module in HSC-MAE is illustrated in Fig. 3.

Decoder: The decoder receives features from the encoder's corresponding positions and input from smaller feature maps. For example, when computing the embedding vector $\boldsymbol{D}_2$ of the decoder's second layer, the decoder uses $\boldsymbol{D}_1$ from its first layer and $\boldsymbol{S}_2$ from the encoder's second layer. For masked positions in $\boldsymbol{S}_2$, we use learnable embeddings for padding.

D. OPTIMIZATION STRATEGY

To enable the model to comprehensively consider both global and local information, we introduce two decoders: the local decoder and the global decoder. Specifically, the architecture of the local decoder is as follows: each layer receives features from the corresponding layer of the encoder, and the final layer of the local decoder focuses on reconstructing the masked regions. Meanwhile, the global decoder processes features from the corresponding layer using self-attention operations. Taking the process of obtaining feature $\boldsymbol{D}_2$ as an example, the computation is performed as described in (1), where N_2 represents the second-layer decoder network and $\boldsymbol{S}'_2$ represents the feature obtained by replacing the masked region

Algorithm 1: Pre-Training.

Require: time series data: $X = \{X_1, X_2, \ldots, X_n\}$, downsampling layers: down_l
Ensure: minimal $Loss$
$Loss = 0$
for $X_i \in X$ **do**
  $Z_i \leftarrow \text{patchify}(X_i)$
  $S_1 \leftarrow \text{Sparse-Enc}_1(Z_i)$
  for $j \leftarrow 2$ to down_l **do**
    $S_j \leftarrow \text{Sparse-Convnet}_j(S_{j-1})$
  end for
  $S'_{down\_l} =$ Replace the masked positions in S_{down_l} with learnable embeddings.
  $\text{Dlocal}_1 \leftarrow \text{Local-dec}_1(S'_{down\_l})$
  $\text{Dglobal}_1 \leftarrow \text{Global-dec}_1(S'_{down\_l})$
  for $k \leftarrow 2$ to down_l **do**
    S'_{down_l-k} = Replace the masked positions in S_{down_l-k} with learnable embeddings.
    $\text{Dlocal}_k \leftarrow \text{Local-dec}_k(\text{Dlocal}_{k-1}, S'_{\text{down}\_l-k})$
    $\text{Dglobal}_k \leftarrow \text{Global-dec}_k(\text{Dglobal}_{k-1}, \text{SA}(S'_{\text{down}\_l-k}))$
  end for
  $X_{\text{out1}} \leftarrow \text{MLP}_1(\text{Dlocal}_{down\_l})$
  $Loss_{i1} \leftarrow \text{Mask-Region\&MSE}(X_{\text{out1}}, X_i)$
  $X_{\text{out2}} \leftarrow \text{MLP}_2(\text{Dglobal}_{down\_l})$
  $Loss_{i2} \leftarrow \text{MSE}(X_{\text{out2}}, X_i)$
  $Loss \leftarrow Loss + Loss_{i1} + Loss_{i2}$
end for

with learnable embeddings:

$$\boldsymbol{D}_2 = N_2\left(\boldsymbol{D}_1, SA\left(\boldsymbol{S}'_2\right)\right), \tag{1}$$

where $SA(\boldsymbol{S}'_2)$ can be calculated as

$$SA\left(\boldsymbol{S}'_2\right) = \text{Softmax}\left(\frac{\boldsymbol{S}'_2 \cdot \boldsymbol{S}'^{T}_2}{\sqrt{d}}\right) \cdot \boldsymbol{S}'_2. \tag{2}$$

In conclusion, following its traversal through the final decoder layer, the global decoder undertakes the task of reconstructing the entire sequence, without regard to whether it pertains to a masked region or not. It is imperative to highlight that both of these decoders employ the mean squared error (MSE) as the loss function. This architectural choice empowers the model to holistically incorporate both local and global information into its learning process when dealing with time series features. In Algorithms 1 and 2, we respectively outline the processes of pre-training and downstream classification tasks. Notably, during the classification task, the decision to freeze the pre-trained network is contingent upon whether the current evaluation pertains to linear evaluation or fine-tuning evaluation. Detailed explanations regarding linear evaluation and fine-tuning evaluation are expounded upon in the experimental section.

Algorithm 2: Downstream Classification.

Require: time series data: $X = [X_1, X_2, \ldots, X_n]$, label: $Y = [y_1, y_2, \ldots, y_n]$, pretrained model: Sparse-Encs
Ensure: minimal Loss
Loss = 0
Change Sparse-Encs into conventional ConvEncs
for $X_i \in X$ **do**
  $Z_i \leftarrow \text{patchify}(X_i)$
  $E_i \leftarrow \text{ConvEncs}(Z_i)$
end for
for all $i \in \{1, 2, \ldots, n\}$ **do**
  $X_{\text{out}} \leftarrow \text{MLP}_3(E_i)$
  $\text{Loss} \leftarrow \text{Loss} + \text{CrossEntropy}(X_{\text{out}}, y_i)$
end for

TABLE 1. Statistics of the 9 Datasets Used in the Experiments

Datasets	Dimensions	SeriesLength	Classes
HAR	9	128	6
PhonemeSpectra	11	217	39
BasicMotions	6	100	4
DuckDuckGeese	1345	270	5
EthanolConcentration	3	1751	4
ArticularyWordRecognition	9	144	25
SpokenArabicDigits	13	93	10
JapaneseVowels	12	29	9
NATOPS	24	51	6

IV. EXPERIMENTAL RESULTS

A. EVALUATION DATASETS AND EXPERIMENTAL SETTINGS

In our study, we conduct an analysis employing nine extensively utilized time series classification datasets retrieved from the UEA time series archive [50], [51]. The details of the datasets are presented in Table 1. For the HSC-MAE model, we adopt a patch size of 4 and executed three successive rounds of convolution and down-sampling blocks, with each down-sampling operation diminishing the scale by a factor of 2. Following the convolutional modules, the embedding dimensions are configured to 96, 192, and 384, correspondingly. The default mask rate is set to 0.5. The batch size for both the training and testing sets is set to 200 by default. The number of epochs for the pre-training process is 50, while the default number of epochs for the fine-tuning process is 200. In the optimization process, we employ the Adam optimizer with a default learning rate of 0.001. For both decoders, we utilize the mean squared error (MSE) function as the loss function. In the evaluation phase, we assess model performance using accuracy and F1 score as our chosen performance metrics. To assess the effectiveness of the pre-training process, we conduct experiments using both linear evaluation and fine-tuning evaluation methods. In the linear evaluation, we freeze the pre-trained parameters and exclusively fine-tuned the final classification head. Conversely, in the fine-tuning evaluation, we fine-tune the parameters of the entire network.Whenever

TABLE 2. Comparison Results of HSC-MAE and Competitive Baselines on 9 Widely-Used Datasets

Models	Ours		SimMTM (NeurIPS23)		InfoTS (AAAI23)		TimeMAE		TS2Vec (AAAI22)		TF-C (NeurIPS22)		TST (KDD21)		TS-TCC (IJCAI21)	
Metric	Accuracy	F1 score	Accuracy	F1 score	Accuracy	F1 score	Accuracy	F1 score	Accuracy	F1 score	Accuracy	F1 score	Accuracy	F1 score	Accuracy	F1 score
Linear Evaluation																
PhonemeSpectra*	**25.56**	**24.02**	15.24	14.76	12.46	12.01	19.25	18.18	10.82	10.09	9.36	7.83	8.93	7.71	8.88	7.54
BasicMotions	**100.00**	**100.00**	97.50	97.49	95.00	94.99	95.00	94.99	85.00	84.96	92.50	92.44	95.00	94.95	**100.00**	**100.00**
DuckDuckGeese	**58.00**	**58.49**	56.00	55.15	**58.00**	57.94	**58.00**	57.42	56.00	54.03	42.00	40.66	56.00	55.96	39.29	40.87
EthanolConcentration	**37.26**	**35.41**	32.69	29.17	27.75	27.80	36.50	28.71	28.89	26.24	33.08	31.60	33.84	28.65	30.41	27.53
HAR*	**93.31**	**93.45**	87.41	87.36	92.26	92.12	91.31	92.15	78.16	77.43	66.17	61.28	87.39	87.18	77.63	77.09
ArticularyWordRecognition	92.33	92.27	96.67	96.66	97.33	97.31	87.00	87.08	97.33	97.29	32.33	28.96	**97.34**	**97.33**	96.12	95.65
SpokenArabicDigits*	92.31	92.28	**96.13**	**96.13**	93.68	93.69	95.76	95.75	94.65	94.64	71.17	70.94	96.03	96.02	88.42	88.41
JapaneseVowels	94.32	94.00	94.59	94.39	97.56	97.50	87.02	85.70	**98.91**	**99.04**	41.62	37.15	98.10	98.00	75.53	77.02
NATOPS	80.56	80.60	82.22	82.58	91.67	91.70	75.56	75.63	**91.78**	**91.79**	63.88	61.74	88.89	88.79	72.83	73.33
Average	**74.85**	**74.50**	**73.16**	**72.63**	**73.97**	**73.90**	**71.71**	**70.62**	**71.28**	**70.61**	**50.23**	**48.07**	**73.50**	**72.73**	**65.46**	**65.27**
Relative Improvement			**2.31%**	**2.57%**	**1.19%**	**0.82%**	**4.38%**	**5.49%**	**5.01%**	**5.51%**	**49.00%**	**55.00%**	**1.83%**	**2.43%**	**14.35%**	**14.14%**
Fine-tuning Evaluation																
PhonemeSpectra*	**33.82**	**33.61**	16.19	14.88	13.89	13.48	20.30	18.83	10.07	8.63	9.81	8.92	8.82	7.79	8.47	8.00
BasicMotion	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	**100.00**	92.50	92.13	90.00	89.99	**100.00**	**100.00**	**100.00**	**100.00**
DuckDuckGeese	64.00	63.98	66.00	64.01	64.00	64.66	68.00	67.17	**70.00**	**69.10**	44.00	43.17	68.00	68.43	53.57	53.06
EthanolConcentration	40.68	38.14	35.36	27.30	33.33	33.17	**42.59**	**38.40**	28.90	19.00	34.60	31.96	31.93	25.92	29.76	29.62
HAR*	**98.50**	**98.49**	93.56	93.52	93.65	93.57	95.11	95.10	86.98	86.69	68.47	64.16	94.35	94.33	89.22	89.04
ArticularyWordRecognition	**99.00**	**99.00**	98.33	98.34	87.66	84.62	98.33	98.33	**99.00**	98.99	35.67	31.93	98.00	97.98	98.44	97.99
SpokenArabicDigits*	**99.82**	**99.82**	99.40	99.40	98.37	98.36	99.20	99.20	99.11	99.10	69.80	69.76	99.06	99.06	98.71	98.71
JapaneseVowels	98.11	98.15	98.64	98.51	98.91	98.86	98.10	97.99	**99.45**	**99.55**	42.16	37.75	98.92	98.70	97.87	97.53
NATOPS	93.33	93.23	89.44	88.67	96.11	96.08	95.00	94.98	**95.72**	**95.75**	65.00	64.88	95.56	95.52	92.93	93.32
Average	**80.81**	**80.49**	**77.44**	**76.07**	**76.21**	**75.87**	**79.63**	**78.89**	**75.75**	**74.33**	**51.06**	**49.17**	**77.18**	**76.41**	**74.33**	**74.14**
Relative Improvement			**4.35%**	**5.81%**	**6.03%**	**6.10%**	**1.48%**	**2.03%**	**6.68%**	**8.29%**	**58.27%**	**63.70%**	**4.70%**	**5.33%**	**8.71%**	**8.56%**

* indicates the results which are adopted from TimeMAE [20]

We conduct pre-training and fine-tuning on the same dataset. The experimental results are presented in %. The highlighted sections in red indicate the optimal results, while the highlighted sections in blue represent the second-best results.

feasible, we utilized the hyperparameters provided in the original papers for the comparative methods. All experiments are conducted using the PyTorch 1.7.1 library on a computer equipped with a GeForce RTX 3060 GPU supporting NVIDIA CUDA.

B. ONE-TO-ONE PRE-TRAINING RESULTS

In the forthcoming experiments, we endeavor to address the following eight research questions (RQs).

RQ1: Can HSC-MAE be more accurate and effective compared to competitive models?

Table 2 presents the evaluation metrics across nine datasets, and a comparative analysis reveals the overarching superiority of fine-tuning evaluation over linear evaluation within each model architecture. Given that fine-tuning evaluation is predicated upon supervised training, its performance surpasses that of the linear adjustment of classification heads, in alignment with anticipated outcomes. Conversely, the performance of the linear fine-tuning approach substantiates the efficacy of pre-training.

Firstly, let us consider the scenario of linear evaluation. In the experimental results of linear evaluation, our method HSC-MAE achieves SOTA performance across five datasets. It is noteworthy that, on average, the time series lengths of these five datasets are relatively long, enabling our pre-training approach to conveniently leverage information captured from a multiscale perspective. For instance, the EthanolConcentration (EC) dataset exhibits a lengthy time span of 1751, posing challenges for Transformer-based models due to their high computational resource requirements, ultimately affecting their efficiency. In contrast, HSC-MAE adeptly integrates multiscale temporal dependencies, enhancing its ability to detect broad temporal patterns, as evidenced by its accuracy of 37.26% on the EC dataset. On the other hand, the shorter time spans of the remaining four datasets may limit HSC-MAE's capability to capture multiscale features. Nonetheless, HSC-MAE consistently demonstrates robust performance, highlighting its proficiency in capturing local information. HSC-MAE achieves a particularly outstanding accuracy of 25.56% on the PhonemeSpectra dataset, surpassing even the supervised learning performance of other models. Clearly, compared to contrastive learning methods such as TS2Vec, TF-C, and TS-TCC, HSC-MAE exhibits greater improvements, particularly outperforming TF-C by a significant margin of 49.00%. This observation underscores the versatility of the MAE training paradigm. While contrastive methods continue to achieve SOTA on a limited number of datasets, this may be attributed to their inherent priors aligning well with dataset characteristics. However, their scalability remains constrained.

The experiments on linear evaluation above demonstrate that a simple mask recovery strategy during the pre-training process can yield satisfactory results. Next, we consider the performance of fine-tuning evaluation. It can be observed that HSC-MAE continues to lead in terms of average accuracy. This suggests that employing convolutional networks as the backbone is a reasonable approach. However, for the JapaneseVowels and NATOPS datasets, with their relatively shorter time lengths, the network's effectiveness in extracting multiscale features from them is slightly inferior to the optimal solution. The DuckDuckGeese dataset, possibly due to its large dimensions, may result in the single-encoder structure showing some inadequacy in handling dimensional information. However, overall, the architecture of convolutional networks is capable of addressing the majority of dataset scenarios.

C. ONE-TO-MANY EVALUATION

RQ2: Can the network learn general features of time series?

One major advantage of self-supervised methods is their ability to learn generic representations of data. Therefore, we

TABLE 3. One-to-Many Evaluation of HSC-MAE

Metrics	ACC			F1 score		
Datasets	Transfer	Linear	Fine-tuning	Transfer	Linear	Fine-tuning
PhonemeSpectra	10.35	25.56	33.82	10.13	24.02	33.61
BasicMotions	97.50	100.00	100.00	97.49	100.00	100.00
DuckDuckGeese	56.00	58.00	64.00	55.32	58.49	63.98
EthanolConcentration	40.68	37.26	40.68	39.83	35.41	38.14
ArticularyWordRecognition	98.33	92.33	99.00	98.33	92.27	99.00
SpokenArabicDigits	98.04	92.31	99.82	98.05	92.28	99.82
JapaneseVowels	98.11	94.32	98.11	98.02	94.00	98.15
NATOPS	86.67	80.56	93.33	86.43	80.60	93.23

"Transfer" indicates pre-training on the HAR dataset, freezing the pretrained network, and adjusting only the final classification head on other datasets. "Linear" denotes pre-training on the respective dataset itself, freezing the pre-trained network, and adjusting the classification head. "Fine-tuning" represents pre-training on the respective dataset and fine-tuning the entire network. The experimental results are presented in %.

assess the transferability of pre-trained models to observe whether the model has learned some universal representations from the data. Specifically, we conduct pre-training on one dataset and then freeze the parameters of the pre-trained network. Subsequently, we perform fine-tuning on another dataset, adjusting only the final classification head without changing the parameters of the pre-trained network. To ensure alignment between inputs from different datasets, a simple linear mapping is applied before feeding the data into the pre-trained network. We use the HAR dataset with the largest data volume for pre-training and perform fine-tuning on the remaining eight datasets individually. The results of transfer learning are compared with those of separately pre-trained and linear evaluation, as well as fine-tuning evaluation on each dataset, demonstrating the model's ability to learn generic representations. The results are presented in Table 3. It can be observed that over half of the datasets among the eight demonstrate better results than mere pre-training on their own datasets. The results of transfer learning are close to those of fine-tuning evaluation similar to supervised learning. The reason for this might be the relatively similar features between these datasets and the HAR dataset. Meanwhile, the HAR dataset learns richer representations, alleviating the issue of insufficient pre-training due to limited data for datasets like NATOPS. However, on datasets such as PhonemeSpectra, the performance of transfer learning is not as good as separate pre-training, which is reasonable. In cases with sufficient data, individual pre-training and fine-tuning for a specific dataset lead to the learning of more unique, dataset-specific features by the network. In conclusion, the above analysis indicates that our model can provide assistance to the target task when pre-trained on other tasks, especially when the dataset for the target task is limited. Our model exhibits the potential to be applied across different domains, making it a candidate for a universal model.

D. ANALYSIS OF PRE-TRAINING SETS WITH DIFFERENT PROPORTIONS

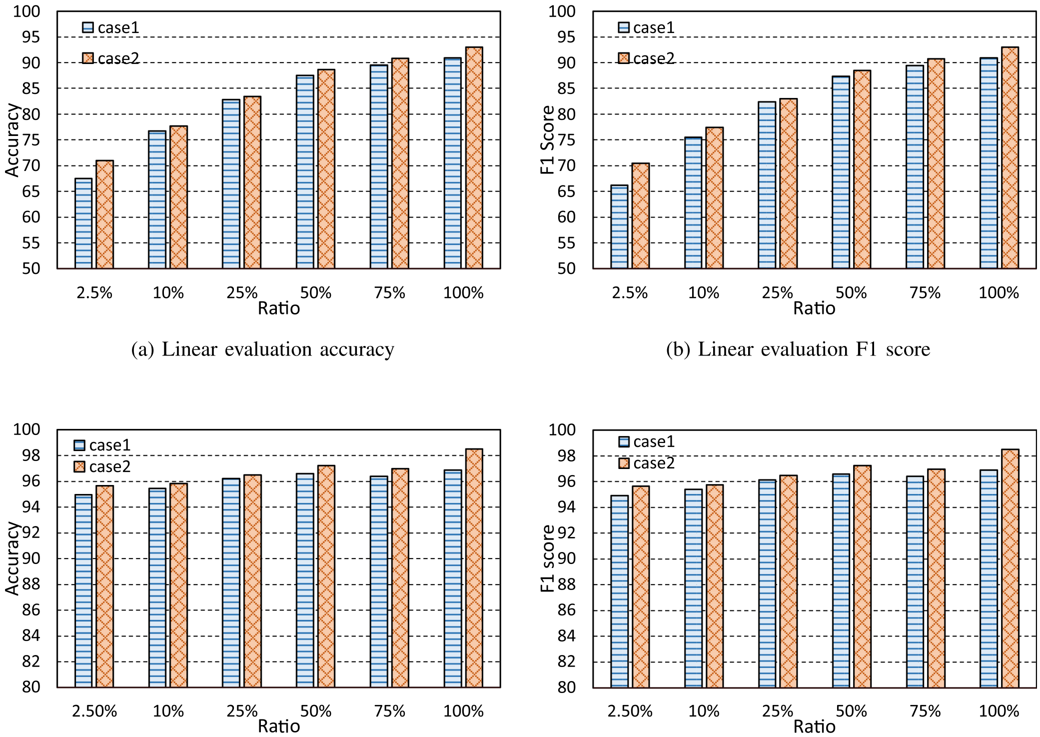
RQ3: Can the increase in data volume lead to an enhancement in model performance?

Subsequently, we aim to analyze whether an increase in the scale of data can lead to an enhancement in the performance of the model. The process of collecting time series data is relatively straightforward, but the labeling process is highly complex, requiring a substantial investment of human and material resources. Therefore, if the model can better leverage a larger-scale dataset to learn more nuanced features, rather than experiencing saturation, thereby avoiding the wastage of a larger dataset, it holds significant implications. In this context, we conducted experiments utilizing the largest-scale HAR dataset. We employed various proportions (2.5%, 10%, 25%, 50%, 75%, 100%) of the training set for pre-training. Subsequently, we conducted fine-tuning under two modes: freezing the pre-trained network, referred to as linear evaluation, and training the entire network, referred to as fine-tuning evaluation. Furthermore, during fine-tuning, we utilized different proportions (75%, 100%) of the training set to observe the model's performance. The experimental results, as depicted in Fig. 4, reveal that with an increase in the proportion of the pre-training dataset, the overall trend of the model's performance shows improvement, particularly in the case of the linear evaluation mode, where the improvement is more pronounced. Even in the fine-tuning evaluation mode, which is equivalent to supervised training, a larger pre-training dataset yields some performance improvement. This could be attributed to the fact that the pre-training process under a larger dataset provides better initial parameters for subsequent optimization processes. The aforementioned analysis suggests that our model holds promise for performance improvement by acquiring a larger pre-training dataset, thereby offering foundational assistance to the target task as a foundation model.

E. ABLATION STUDY

RQ4: Is the design of combining local decoder and global decoder appropriate and effective?

In this study, we conducted a series of ablation experiments aimed at validating the indispensability of both local and global decoders within our framework. The obtained results, as illustrated in Fig. 5, unequivocally demonstrate that the collaborative utilization of these two decoders leads to a significant enhancement in overall detection performance. Furthermore, the performance of using only the local decoder



(a) Linear evaluation accuracy

(b) Linear evaluation F1 score

(c) Fine-tuning accuracy

(d) Fine-tuning F1 score

FIGURE 4. **The performance of HSC-MAE across different proportions of the pre-training dataset. The performance of our model is described in terms of accuracy and F1 score (*y*-axis) under different proportions of pre-training data (*x*-axis) and various fine-tuning modes. In this context, Case 1 denotes the utilization of a 75% training set during fine-tuning, while Case 2 signifies the application of a 100% training set during fine-tuning.**

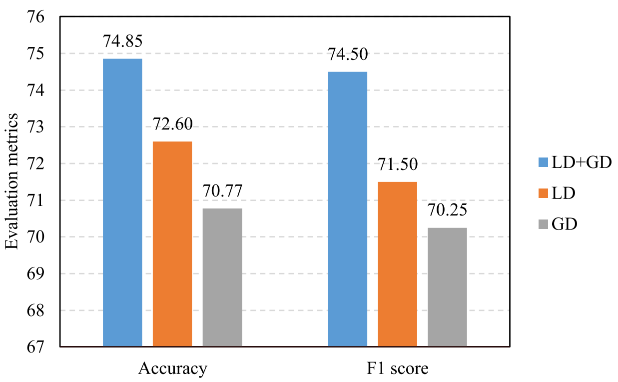


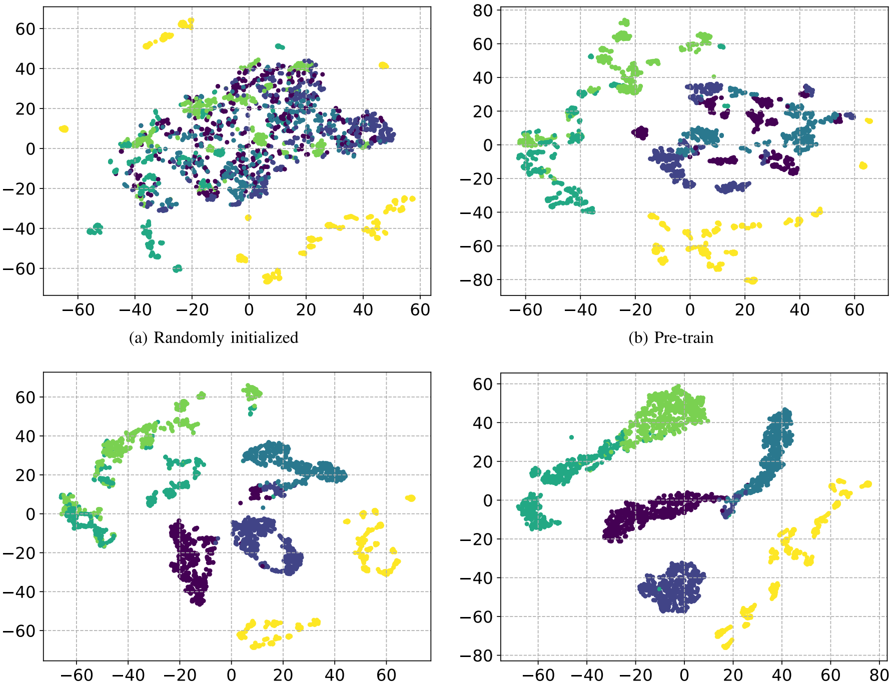
FIGURE 5. **Average accuracy and F1 scores across the 9 datasets. LD stands for local decoder, and GD stands for global decoder.**

surpasses that of using only the global encoder. This outcome underscores the feasibility of employing convolutional networks to focus on local temporal characteristics. Moreover, the integration of a global decoder aids in capturing global information, thereby endowing the network with richer features.

F. T-SNE ANALYSIS

RQ5: Can the pre-trained features be distinguishable?

Next, we will visualize the effects of pre-training using a t-SNE [52] approach on the HAR dataset to validate the distinguishing ability between different time series classes. The t-SNE approach is an algorithm used for dimensionality reduction and visualization of high-dimensional data. It maps data points to a two- or three-dimensional space while preserving local structures and similarities. Fig. 6(a) displays the outcomes of the randomly initialized encoder, while Fig. 6(b) demonstrates the performance of the pre-trained encoder without any fine-tuning. Fig. 6(c) represents the results of linear evaluation, and Fig. 6(d) showcases the outcomes of fine-tuning evaluation. It can be observed that, without any adjustment to the pre-training process, as depicted in Fig. 6(b), the network has already preliminarily distinguished and clustered various types of points. This indicates that our network has initially learned the features of time series. After a simple linear evaluation process, as shown in Fig. 6(c), points of different types further distance themselves from each other, and points of the same type become more closely related. This suggests that, following the initial learning of data features during the pre-training process, the network can be effectively adapted to downstream tasks with relatively small additional costs. At last, as shown in Fig. 6(d), the results of the fine-tuning evaluation demonstrate that the network, under supervised conditions, performs well in achieving the target task. The inherent design of the network structure enables the capture of time series features. Naturally, as fine-tuning evaluation is akin to supervised training, it exhibits the best classification performance. In conclusion, our pre-training



(a) Randomly initialized

(b) Pre-train

(c) Linear evaluation

(d) Fine-tuning evaluation

FIGURE 6. t-SNE visualization on the HAR dataset. Different colors represent different classes.

paradigm effectively aids in feature separation, thereby generating distinct clusters in visualization.

G. ANALYSIS OF THE IMPACT OF MODEL SIZE ON MODEL PERFORMANCE

RQ6: Can larger models, extended training durations, and the design of an inverted bottleneck contribute to an improvement in model performance?

A series of studies suggest that larger-scale self-supervised pre-trained models and longer training durations hold the potential to achieve superior model performance. Thus, we aim to investigate whether similar characteristics apply to our time series pre-training model. Additionally, given that our network, similar to ConvNeXt [49], adopts an inverted bottleneck design, we seek to understand the impact of the inverted bottleneck design on model effectiveness. Consequently, we validate the performance of the model under different parameters by varying the number of layers of the convolutional module ($l = 3, l = 6$), the presence or absence of the inverted bottleneck, and the number of training epochs. The results are presented in Table 4. Firstly, both linear evaluation and fine-tuning evaluation consistently show that larger models and extended training times generally lead to performance improvements. However, for smaller datasets, such as DuckDuckGeese, employing a relatively small model during the pre-training phase proves to be adequate. This implies that the enhancement of pre-trained model performance depends not only on the model itself but also on factors such as data scale and features. Furthermore, it is observed that during pre-training, larger models require more extensive training times for optimal performance. Taking the example of linear evaluation under $l = 6$, inverted, epoch $= 25$ for PhonemeSpectra and JapaneseVowels, despite the model's enlargement compared to $l = 3$, the limited number of training rounds results in suboptimal performance. In addition, in most cases, the model's performance under the inverted bottleneck design surpasses that of the model under the bottleneck design. However, it is noteworthy that the inverted bottleneck design necessitates a longer training time. In summary, when faced with larger-scale data, larger-scale models, increased training duration, and the inverted bottleneck design hold more promise for achieving superior performance.

H. ANALYSIS OF THE NUMBER OF LAYERS IN THE HIERARCHICAL ENCODER AND DECODER

RQ7: How does the number of down-sampling layers affect the model's performance?

Our model adopts an overall design in the style of U-Net. Therefore, we aim to investigate the impact of the number of down-sampling and up-sampling operations on the model. The experimental results are presented in Table 5. Generally, it can be observed that a network with three down-sampling operations is sufficient. If the down-sampling operations are too few, the model may struggle to capture an adequate amount of information from a multi-scale perspective. On the other hand, excessive down-sampling operations may result in insufficient information about the inherent characteristics of the time series, leading to a decline in performance.

TABLE 4. Model Performance Under Various Model Sizes

Evaluation	Datasets	Metrics	l = 3 Bottleneck epoch=25	l = 3 Bottleneck epoch=50	l = 3 Inverted epoch=25	l = 3 Inverted epoch=50	l=6 Bottleneck epoch=25	l=6 Bottleneck epoch=50	l=6 Inverted epoch=25	l=6 Inverted epoch=50
Linear Evaluation	HAR	Accuracy	91.61	90.36	90.46	92.53	91.78	92.56	92.73	92.98
		F1 score	91.56	90.28	90.44	92.56	91,73	92.56	92.97	92.98
	PhonemeSpectra	Accuracy	6.94	6.56	15.86	23.23	17.56	18.19	14.79	23.91
		F1 score	3.90	2.78	13.47	22.32	14.48	15.84	10.99	21.46
	JapaneseVowels	Accuracy	93.51	93.78	89.45	90.00	92.16	93.24	85.95	93.78
		F1 score	93.14	93.33	89.00	89.48	91.63	92.79	84.35	93.36
	DuckDuckGeese	Accuracy	54.00	62.00	54.00	56.00	54.00	54.00	62.00	58.00
		F1 score	54.31	61.91	54.19	55.25	53.79	53.00	61.99	58.49
Fine-tuning Evaluation	HAR	Accuracy	96.57	96.53	95.49	95.92	97.11	96.64	96.02	98.50
		F1 score	96.55	96.46	95.44	95.91	97.14	96.59	96.05	98.49
	PhonemeSpectra	Accuracy	25.79	26.90	28.92	28.57	27.13	27.29	30.54	32.98
		F1 score	25.19	25.68	27.70	27.40	26.24	26.61	29.99	32.46
	JapaneseVowels	Accuracy	97.56	97.29	97.83	98.10	97.29	98.10	98.10	98.11
		F1 score	97.55	97.18	97.77	97.99	97.36	98.06	98.06	98.15
	DuckDuckGeese	Accuracy	64.00	62.00	60.00	62.00	64.00	64.00	62.00	68.00
		F1 score	63.83	62.49	59.96	60.98	63.01	63.91	61.83	67.43

Here, 'l' represents the repetition count of the convolutional module, and 'epoch' signifies the number of training rounds during pre-training. 'Bottleneck' indicates the design of the convolutional module as a bottleneck design, while 'Inverted' represents the convolutional module's design as an inverted bottleneck. The highlighted portions in red indicate the best results. The experimental results are presented in %.

TABLE 5. The Impact of Different Numbers of Down-Sampling Layers on Model Performance

Methods		Linear Evaluation			Fine-tuning Evaluation		
Datasets	Metrics	down_l=2	down_l=3	down_l=4	down_l=2	down_l=3	down_l=4
HAR	Accuracy	90.36	92.98	92.53	95.93	98.50	98.00
	F1 score	90.20	92.98	92.66	95.91	98.49	98.02
PhonemeSpectra	Accuracy	20.49	23.91	20.40	30.33	32.98	27.23
	F1 score	18.35	21.46	19.40	29.53	32.46	26.63
JapaneseVowels	Accuracy	92.43	93.78	90.81	98.11	98.11	96.76
	F1 score	91.87	93.36	90.04	98.16	98.15	96.75
DuckDuckGeese	Accuracy	58.00	58.00	50.00	60.00	68.00	68.00
	F1 score	57.15	58.49	47.70	60.01	67.43	68.41

The highlighted sections indicate the optimal results. The experimental results are presented in %.

TABLE 6. The Results of Different Mask Ratios on HSC-MAE

Evaluation	Datasets	Metrics	0.2	0.3	0.4	0.5	0.6	0.7	0.8
FineLast	HAR	Accuracy	92.81	93.31	91.92	92.98	90.36	91.61	89.78
		F1 score	92.89	93.45	91.96	92.98	90.34	91.61	89.67
	PhonemeSpectra	Accuracy	6.74	25.56	23.02	23.91	22.15	22.93	20.78
		F1 score	2.73	24.02	21.02	21.46	19.76	21.83	19.82
	JapaneseVowels	Accuracy	94.32	90.27	93.24	93.78	89.45	90.00	94.05
		F1 score	94.00	89.28	92.74	93.36	88.93	89.57	93.54
	DuckDuckGeese	Accuracy	56.00	52.00	58.00	58.00	52.00	50.00	42.00
		F1 score	55.82	51.43	57.00	58.49	51.95	49.82	42.61
FineALL	HAR	Accuracy	97.86	97.93	97.89	98.50	97.76	96.94	96.40
		F1 score	97.88	97.91	97.88	98.49	97.71	96.93	96.32
	PhonemeSpectra	Accuracy	30.45	33.82	33.58	32.98	32.81	33.61	32.29
		F1 score	30.02	33.61	32.02	32.46	31.34	33.27	31.45
	JapaneseVowels	Accuracy	97.83	97.56	98.10	98.11	97.83	97.83	97.56
		F1 score	97.67	97.46	97.84	98.15	97.59	97.95	97.47
	DuckDuckGeese	Accuracy	60.00	62.00	64.00	62.00	64.00	56.00	64.00
		F1 score	60.22	61.81	63.98	62.28	63.85	56.16	63.88

The highlighted sections indicate the optimal results. The experimental results are presented in %.

I. ANALYSIS OF THE IMPACT OF DIFFERENT MASK RATIOS ON MODEL PERFORMANCE

RQ8: What is the impact of different mask ratios on model performance?

Time series inherently possesses a certain degree of information redundancy. Therefore, our HSC-MAE network considers a small segment of the sequence as the fundamental unit for masking and recovery. Consequently, we investigate the impact of different mask ratios on model performance, with results presented in Table 6. It can be observed that, when employing a higher mask ratio, the model's performance remains reasonable, indicating that time series may inherently contain a substantial amount of redundant information that can be relatively easily reconstructed from the unmasked portions. However, for achieving optimal results, a moderately low mask ratio proves to be more effective. This approach avoids the loss of details associated with excessively high mask ratios and prevents the reconstruction task from being overly simplistic, ensuring the model learns the distinctive features of the time series.

V. CONCLUSION

In the domain of time series self-supervised training, the challenge introduced by the potential bias inherent in constructing positive and negative pairs has led to the adoption of generative approaches based on mask prediction, akin to the masked autoencoder paradigm. Recognizing the challenges posed by the handling of extended sequences and intricate multi-scale dependencies encountered by prominent Transformer-based techniques, we introduce HSC-MAE into the realm of time series analysis. HSC-MAE considers the redundancy within time series by utilizing short segments as the basic units for masking. Through the fusion of global and local information at multiple scales, HSC-MAE exhibits exceptional performance in benchmark evaluations, thereby underscoring the untapped potential of convolutional-based MAE paradigms.

REFERENCES

[1] Y. Wu, J. M. Hernández-Lobato, and G. Zoubin, "Dynamic covariance models for multivariate financial time series," in *Proc. Int. Conf. Mach. Learn.*, 2013, pp. 558–566.

[2] S. K. Grange and D. C. Carslaw, "Using meteorological normalisation to detect interventions in air quality time series," *Sci. Total Environ.*, vol. 653, pp. 578–588, 2019.

[3] D. A. Tedjopurnomo, Z. Bao, B. Zheng, F. M. Choudhury, and A. K. Qin, "A survey on modern deep neural network for traffic prediction: Trends, methods and challenges," *IEEE Trans. Knowl. Data Eng.*, vol. 34, no. 4, pp. 1544–1561, Apr. 2020.

[4] C. Fan, Y. Peng, S. Peng, H. Zhang, Y. Wu, and S. Kwong, "Detection of train driver fatigue and distraction based on forehead EEG: A time-series ensemble learning method," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 8, pp. 13559–13569, Aug. 2022.

[5] T. Cover and P. Hart, "Nearest neighbor pattern classification," *IEEE Trans. Inf. Theory*, vol. IT-13, no. 1, pp. 21–27, Jan. 1967.

[6] J. R. Quinlan, *C4. 5: Programs for Machine Learning*. Amsterdam, The Netherlands: Elsevier, 2014.

[7] B. E. Boser, I. M. Guyon, and V. N. Vapnik, "A training algorithm for optimal margin classifiers," in *Proc. 5th Annu. Workshop Comput. Learn. Theory*, 1992, pp. 144–152.

[8] X. Han et al., "Pre-trained models: Past, present and future," *AI Open*, vol. 2, pp. 225–250, 2021.

[9] D. Mizrahi et al., "4 M: Massively multimodal masked modeling," in *Proc. Int. Conf. Adv. Neural Inf. Process. Syst.*, 2024, vol. 36.

[10] K. He, X. Chen, S. Xie, Y. Li, P. Dollár, and R. Girshick, "Masked autoencoders are scalable vision learners," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 16000–16009.

[11] S. Liu, B. Wu, X. Xie, G. Meng, and Y. Liu, "ContraBERT: Enhancing code pre-trained models via contrastive learning," in *2023 IEEE/ACM 45th Int. Conf. Softw. Eng.*, 2023, pp. 2476–2487.

[12] X. Qiu, T. Sun, Y. Xu, Y. Shao, N. Dai, and X. Huang, "Pre-trained models for natural language processing: A survey," *Sci. China Technol. Sci.*, vol. 63, no. 10, pp. 1872–1897, 2020.

[13] J.-Y. Franceschi, A. Dieuleveut, and M. Jaggi, "Unsupervised scalable representation learning for multivariate time series," in *Proc. Int. Conf. Adv. Neural Inf. Process. Syst.*, 2019, vol. 32, pp. 4652–4663.

[14] S. Tonekaboni, D. Eytan, and A. Goldenberg, "Unsupervised representation learning for time series with temporal neighborhood coding," in *Proc. Int. Conf. Learn. Representations*, 2021.

[15] E. Eldele et al., "Time-series representation learning via temporal and contextual contrasting," in *Proc. 30th Int. Joint Conf. Artif. Intell.*, 2021, pp. 2352–2359.

[16] Z. Yue et al., "TS2VEC: Towards universal representation of time series," in *Proc. AAAI Conf. Artif. Intell.*, 2022, vol. 36, pp. 8980–8987.

[17] X. Zhang, Z. Zhao, T. Tsiligkaridis, and M. Zitnik, "Self-supervised contrastive pre-training for time series via time-frequency consistency," in *Proc. Int. Conf. Adv. Neural Inf. Process. Syst.*, 2022, vol. 35, pp. 3988–4003.

[18] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proc. 2019 Conf. North Amer. Chapter Assoc. Comput. Linguistics: Hum. Lang. Technol.*, 2019, vol. 1, Art. no. 2.

[19] G. Zerveas, S. Jayaraman, D. Patel, A. Bhamidipaty, and C. Eickhoff, "A transformer-based framework for multivariate time series representation learning," in *Proc. 27th ACM SIGKDD Conf. Knowl. Discov. Data Mining*, 2021, pp. 2114–2124.

[20] M. Cheng, Q. Liu, Z. Liu, H. Zhang, R. Zhang, and E. Chen, "TimeMAE: Self-supervised representations of time series with decoupled masked autoencoders," 2023, *arXiv:2303.00320*.

[21] Z. Gong, Y. Tang, and J. Liang, "PatchMixer: A patch-mixing architecture for long-term time series forecasting," 2023, *arXiv:2310.00655*.

[22] L. Zhao et al., "Purified contrastive learning with global and local representation for hyperspectral image classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 62, 2024, Art. no. 5520414.

[23] J. Lines and A. Bagnall, "Time series classification with ensembles of elastic distance measures," *Data Mining Knowl. Discov.*, vol. 29, pp. 565–592, 2015.

[24] T. K. Vintsyuk, "Speech discrimination by dynamic programming," *Cybernetics*, vol. 4, no. 1, pp. 52–57, 1968.

[25] J. Lines, S. Taylor, and A. Bagnall, "HIVE-COTE: The hierarchical vote collective of transformation-based ensembles for time series classification," in *2016 IEEE 16th Int. Conf. Data Mining*, 2016, pp. 1041–1046.

[26] A. Dempster, F. Petitjean, and G. I. Webb, "Rocket: Exceptionally fast and accurate time series classification using random convolutional kernels," *Data Mining Knowl. Discov.*, vol. 34, no. 5, pp. 1454–1495, 2020.

[27] B. Lucas et al., "Proximity forest: An effective and scalable distance-based classifier for time series," *Data Mining Knowl. Discov.*, vol. 33, no. 3, pp. 607–635, 2019.

[28] T. K. Ho, "Random decision forests," in *Proc. IEEE 3rd Int. Conf. Document Anal. Recognit.*, 1995, vol. 1, pp. 278–282.

[29] H. I. Fawaz et al., "Inceptiontime: Finding alexnet for time series classification," *Data Mining Knowl. Discov.*, vol. 34, no. 6, pp. 1936–1962, 2020.

[30] Y. Zheng, Q. Liu, E. Chen, Y. Ge, and J. L. Zhao, "Exploiting multi-channels deep convolutional neural networks for multivariate time series classification," *Front. Comput. Sci.*, vol. 10, pp. 96–112, 2016.

[31] C. W. Tan, A. Dempster, C. Bergmeir, and G. I. Webb, "Multirocket: Multiple pooling operators and transformations for fast and effective time series classification," *Data Mining Knowl. Discov.*, vol. 36, no. 5, pp. 1623–1646, 2022.

[32] W. Chen and K. Shi, "Multi-scale attention convolutional neural network for time series classification," *Neural Netw.*, vol. 136, pp. 126–140, 2021.

[33] B. Wu and X.-P. Zhang, "Environmental sound classification via time–frequency attention and framewise self-attention-based deep neural networks," *IEEE Internet Things J.*, vol. 9, no. 5, pp. 3416–3428, Mar. 2021.

[34] Z. Wu, S. Pan, G. Long, J. Jiang, X. Chang, and C. Zhang, "Connecting the dots: Multivariate time series forecasting with graph neural networks," in *Proc. 26th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2020, pp. 753–763.

[35] J. Sarzynska-Wawer et al., "Detecting formal thought disorder by deep contextualized word representations," *Psychiatry Res.*, vol. 304, 2021, Art. no. 114135.

[36] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick, "Momentum contrast for unsupervised visual representation learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 9729–9738.

[37] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," in *Proc. Int. Conf. Mach. Learn.*, 2020, pp. 1597–1607.

[38] Z. Liu et al., "Swin transformer: Hierarchical vision transformer using shifted windows," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2021, pp. 10012–10022.

[39] P. Malhotra, V. TV, L. Vig, P. Agarwal, and G. Shroff, "TimeNet: Pre-trained deep recurrent neural network for time series classification," *Esann*, 2017.

[40] Y. Nie, N. H. Nguyen, P. Sinthong, and J. Kalagnanam, "A time series is worth 64 words: Long-term forecasting with transformers," in *Proc. Int. Conf. Learn. Representations*, 2023.

[41] H. Fan, F. Zhang, and Y. Gao, "Self-supervised time series representation learning by inter-intra relational reasoning," 2020, *arXiv:2011.13548*.

[42] G. Woo, C. Liu, D. Sahoo, A. Kumar, and S. Hoi, "Cost: Contrastive learning of disentangled seasonal-trend representations for time series forecasting," in *Proc. Int. Conf. Learn. Representations*, 2021.

[43] S. Deldari, D. V. Smith, H. Xue, and F. D. Salim, "Time series change point detection with self-supervised contrastive predictive coding," in *Proc. Web Conf. 2021*, 2021, pp. 3124–3135.

[44] A. Hyvarinen and H. Morioka, "Nonlinear ICA of temporally dependent stationary sources," in *Proc. 20th Int. Conf. Artif. Intell. Statist.*, 2017, pp. 460–469.

[45] D. Luo et al., "Time series contrastive learning with information-aware augmentations," in *Proc. AAAI Conf. Artif. Intell.*, 2023, vol. 37, pp. 4534–4542.

[46] J. Dong, H. Wu, H. Zhang, L. Zhang, J. Wang, and M. Long, "SimMTM: A simple pre-training framework for masked time-series modeling," in *Proc. Int. Conf. Adv. Neural Inf. Process. Syst.*, 2023, pp. 29996–30025.

[47] K. Tian et al., "Designing BERT for convolutional networks: Sparse and hierarchical masked modeling," in *Proc. 11th Int. Conf. Learn. Representations*, 2023.

[48] S. Woo et al., "ConvNext V2: Co-designing and scaling convnets with masked autoencoders," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2023, pp. 16133–16142.

[49] Z. Liu, H. Mao, C.-Y. Wu, C. Feichtenhofer, T. Darrell, and S. Xie, "A convnet for the 2020s," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 11976–11986.

[50] A. Bagnall et al., "The UEA multivariate time series classification archive, 2018," *Statistics*, 2018.

[51] D. Anguita et al., "A public domain dataset for human activity recognition using smartphones," *Esann*, vol. 3, 2013, Art. no. 3.

[52] L. V. d. Maaten and G. Hinton, "Visualizing data using t-SNE," *J. Mach. Learn. Res.*, vol. 9, no. 11, pp. 2579–2605, 2008.

TING YU received the bachelor's degree in computer science from the Harbin Institute of Technology, Harbin, China, in 2020. She is currently working toward the master's degree with the College of Computer, National University of Defense Technology, Changsha, China. Her research interests include time series analysis and artificial intelligence for IT operations.

KELE XU (Member, IEEE) received the Doctoral degree from Université Pierre et Marie CURIE (UPMC), Paris, France, in 2016. He is curently an Associate Professor with the National University of Defence Technology, Changsha, China. He has coauthored more than 100 publications in peer-reviewed journals and conference proceedings, including IEEE/ACM TRANSACTIONS ON AUDIO, SPEECH, AND LANGUAGE PROCESSING, IEEE TRANSACTIONS ON EMERGING TOPICS IN COMPUTATIONAL INTELLIGENCE, IEEE TRANSACTIONS ON ARTIFICIAL INTELLIGENCE, IEEE TRANSACTIONS ON MEDICAL IMAGING, IEEE TRANSACTIONS ON GEOSCIENCE AND REMOTE SENSING, IEEE TRANSACTIONS ON COGNITIVE AND DEVELOPMENTAL SYSTEMS, *Journal of the American Statistical Association*, ICML, AAAI, IJCAI, ASE, ACM MM, and ICASSP.

XU WANG received the bachelor's degree in economics from the Southwest University of Finance and Economics, Chengdu, China, in 2017. She is currently working toward the master's degree with the National University of Defense Technology, Hunan, China. Her main research interests include time series analysis and artificial intelligence for IT operations.

BO DING received the Ph.D. degree in computer science from the National University of Defense Technology, Changsha, China, in 2010. He is currently a Professor with the State Key Laboratory of Complex and Critical Software Environment, National University of Defense Technology. His research interests include distributed computing and complex software systems.

DAWEI FENG received the Ph.D. degree in computer science from the University of Paris-Sud, Gif-sur-Yvette, France, in 2014. He is currently an Associate Professor with the National Laboratory for Parallel and Distributed Processing, National University of Defense Technology, Changsha, China. His research interests include intelligent systems and distributed computing.