IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 9 January 2024, accepted 18 January 2024, date of publication 22 January 2024, date of current version 13 February 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3357143

RESEARCH ARTICLE

Human Activity Recognition via Temporal Fusion Contrastive Learning

INKYUNG KIM[1,2,*], **JUWAN LIM**[1,*], **AND JAEKOO LEE**[1]
[1]College of Computer Science, Kookmin University, Seongbuk-gu, Seoul 02707, Republic of Korea
[2]Hyundai Motor Company, Namyang-eup, Hwaseong-si, Gyeonggi-do 18278, Republic of Korea

Corresponding author: Jaekoo Lee (jaekoo@kookmin.ac.kr)

This work was supported in part by the National Research Foundation (NRF) under Grant RS-2023-00212484, and in part by the Institute of Information & Communications Technology Planning & Evaluation (IITP) funded by the Korean Government [Ministry of Science and ICT (MSIT)] under Grant RS-2022-00167194.

*Inkyung Kim and Juwan Lim are co-first authors.

ABSTRACT With recent advancements in wearable devices and the Internet of Things (IoT), human activity recognition (HAR) has attracted increasing interest in the wearable technology market. However, for sensor-based HAR, collecting sufficient labeled data for deep neural network learning is difficult because experts must find visually recognizable patterns in time-series data. In addition, collecting data is difficult due to privacy issues. To overcome these limitations, self-supervised learning (SSL)-based HAR methods have recently been proposed; these can learn representations without using labeled data. However, such methods only utilize sensor data and do not include the sensor wearer's biometric information. A learning method that excludes biometric information can identify typical movement patterns but cannot learn customized movement patterns effectively. Thus, in this paper, we proposed the **T**emporal **F**usion **C**ontrastive **L**earning (TFCL) method, which considers a sensor wearer's biometric information while training. Experimental results demonstrate that, when fine-tuned with biometric information, the proposed TFCL method obtained the highest F1 score of 0.9791 and 0.7433 on the DLR and MobiAct datasets, respectively. Furthermore, the results obtained when the proposed TFCL method was used to learn the representation and then applied to the downstream task were similar to or better than those obtained using supervised learning from scratch. These results indicate that representations can be learned effectively through TFCL. The experimental code can be found on GitHub at https://github.com/IKKIM00/temporal-fusion-contrastive-learning

INDEX TERMS Activity recognition, deep learning, contrastive learning, self-supervised learning, feature fusion, time series analysis, anomaly detection.

I. INTRODUCTION

A. BACKGROUND

Wearable technologies are gaining widespread interest due to various advancements in wearable devices and the Internet of Things (IoT). The demand for wearable technology has increased due to increasing interest in personal health in light of the ongoing COVID-19 pandemic [1]. Market research analysis by Grand View Research indicates that the global wearable technology market was worth USD 40.65 billion in 2020, and this market is expected to grow between 2021 and 2028 at an average annual rate of 13.8% [1]. Individuals become concerned for their health by periodically observing their health conditions in order to improve their quality of life, and human activity recognition (HAR) is a representative example. In addition, according to the Centers for Disease Control and Prevention (CDC), a public health agency of the United States, the American the elderly population is expected to increase from 52 million in 2018 to 73 million in 2030 [2]. With an aging of society, the health and well-being of the elderly has attracted increased interest. In particular, smart elderly care using sensor-based wearable devices is becoming popular as a way to improve the lifestyle of elderly people. Such devices can be used to predict and issue timely

The associate editor coordinating the review of this manuscript and approving it for publication was Angelo Trotta.

warnings for falls or other unexpected events that may pose risks to the elderly [3].

B. CHALLENGES

Deep neural network (DNN)-based deep learning methods that utilize data collected using wearable devices have been proposed previously. A large dataset with labeled data is required to train a DNN in supervised learning approach. In contrast to computer vision datasets, the amount of labeled data in most real-world time-series datasets is limited because it is more difficult to identify visually observable patterns in time-series data than in computer vision data; experts must find such patterns, which is quite expensive and time consuming. Thus, self-supervised learning (SSL), which was proposed in the computer vision field, is now being applied to time-series data.

C. RESEARCH MOTIVATION

Recently, studies have proven that SSL methods can be employed to learn representations effectively using unlabeled data. Note that semi-supervised learning only uses some labels. In contrast, SSL learns the representation without using labels. After generating a pretrained model, the learned representation is applied to a downstream task. Chen et al. [4] used a pretrained self-supervised model through the proposed SimCLR, which is a simple framework to realize contrastive learning of visual representations, and they proved that their model can perform as well as supervised models even if labeled data are lacking. In addition, contrastive learning has been verified as being capable of learning invariant representations in the computer vision field [5], [6], [7], [8].

However, the augmentation method proposed in the computer vision domain has limitations in that it cannot be applied as-is to time-series data. This is because temporal dependency is an important factor in time-series data. Thus, various augmentation methods have been proposed in recent time-series data-based SSL methods [9], [10], [11].

In addition, SSL-based methods have been proposed for HAR tasks using sensor data. However, in HAR, data labeling is difficult due to privacy and security issues in the data collection process. Furthermore, the labeling process is time intensive. Thus, the labeling process is slow relative to the data collection speed, and only a small amount of labeled data is available publicly. To overcome these limitations, recent studies have applied SSL to HAR [11], [12], [13].

D. CONTRIBUTION

Previously proposed methods only used movement information, and they excluded biometric information. When the wearer's biometric information is excluded, patterns of general movement can be learned; however, movement patterns customized to the wearer cannot be learned easily. To overcome the problems associated with the limited amount of labeled data for HAR and to learn individual movement patterns, we propose SSL-based **T**emporal **F**usion **C**ontrastive **L**earning (TFCL) method. Figure 1 shows the overall architecture of the proposed TFCL method. The experimental results demonstrate that when our TFCL is used, activity recognition is as effective as when supervised learning is used. In addition, when representations are learned with biometric information, movements can be detected more accurately than the absence of the sensor wearer's biometric information.

Our primary contributions are summarized as follows.

- We propose the TFCL method, which is a contrastive learning method for HAR based on SSL using sensor data with a small amount of labeled data.
- We quantitatively verified that the sensor wearer's biometric information is required for more accurate action recognition than the absence of the sensor wearer's biometric information.
- We propose a framework to conduct extensive experiments with two datasets that include sensor wearers' biometric information. The experimental results confirm that representations of TFCL method were learned effectively.

II. RELATED WORK

A. CONTRASTIVE LEARNING FOR COMPUTER VISION AND TIME SERIES

It has been proven that useful representations can be learned, and many advancements have been made in SSL and applied to downstream tasks in the computer vision field. For example, contrastive learning is becoming primary research topic in the machine learning field because it provides a way to learn invariant representation through advanced data augmentation. Various SSL methods that utilize contrastive learning have been proposed in the computer vision field. He et al. [6] proposed MoCo, which uses a momentum encoder to learn the representation of negative pairs acquired through a memory bank. Chen et al. [4] proposed SimCLR, which is a contrastive learning method that replaces the momentum encoder while using the negative pairs of a large batch. Grill et al. [7] developed Bootstrap Your Own Latent (BYOL), a contrastive learning method that did not use negative samples. Chen and He [8] proposed SimSiam, which is a Siamese network-based method without using negative samples and achieved state-of-the-art performance.

As studies began demonstrating the effectiveness of contrastive learning in the computer vision field, researchers started applying contrastive learning to time-series data. For example, Oord et al. [14] proposed contrastive predictive coding (CPC) based on a predictive coding theory proposed in the neuro-engineering field. For the encoder model, they used a one-dimensional (1D) convolutional neural network (CNN) to compress the input data and applied compressed latent representations to an autoregressive model to predict the values of multiple steps in the future. Tonekaboni et al. [10] applied contrastive learning, in which a Gaussian distribution was used to approximate a neighborhood of time series; contexts obtained from the same and other distributions were

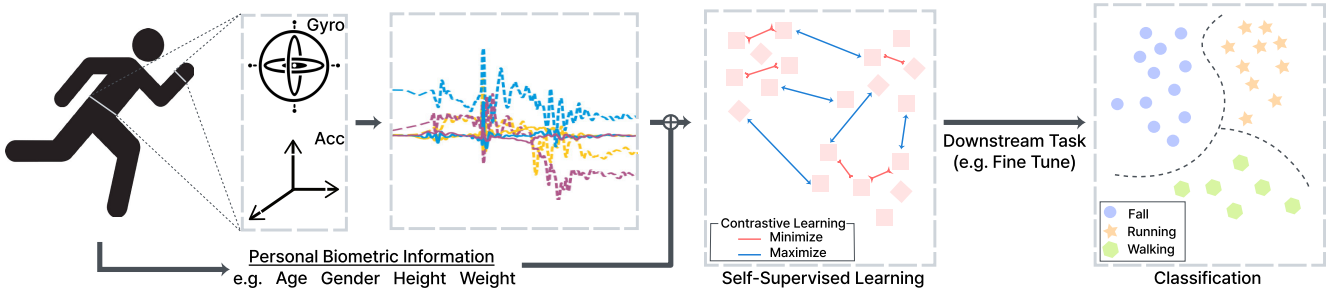

FIGURE 1. **Overview of temporal fusion contrastive learning.**

set as positive and negative samples, respectively. Eldele et al. [9] proposed a contrastive learning method using weak and strong augmentation methods.

B. CONTRASTIVE LEARNING IN HAR

Most existing HAR studies are based on supervised learning [15], [16], [17], [18], [19]. With recent developments in SSL-based on time-series data, attempts have been made to apply SSL to HAR tasks to overcome the limitations and difficulties associated with labeling data. For example, Tang et al. [11] investigated the efficiency of contrastive learning for the first time in a sensor-based HAR task. They employed the SimCLR [4] as a visual representation learning method, and a data augmentation method for time-series sensor data instead of the image augmentation operator. Additionally, eight popular time-series data augmentation methods were used to compare experimental results between augmentation methods. Khaertdinov et al. [12] also used the SimCLR framework; however, they combined a 1D CNN and the transformer's encoder part as a backbone encoder model. Furthermore, they configured their model to randomly select an augmentation method from five time-series augmentation methods. Haresamudram et al. [13] applied the CPC [14] framework to HAR. They used a 1D CNN as the encoder and applied a gated recurrent unit (GRU) for the autoregressive network.

However, all of these methods excluded the sensor wearer's human biometric information and used only the sensor data. Therefore, Considering that movement patterns vary by age, gender, height, and weight, the wearer's biometric information is emerged to provide customized activity recognition.

III. METHODS

In the following, we describe the proposed TFCL method in detail. The dataset used in our experiment contains biometric information about sensor wearers $i \in I$. Each sensor wearer i performs a specific movement for time $t \in [0, T_i]$, and each dataset contains m_x sensor inputs, where $x_{i,t} \in \mathbb{R}^{m_x}$, and m_s sensor wearer's biometric information, where $s_i \in \mathbb{R}^{m_s}$.

The overall flow of the proposed architecture is shown in figure 2. The encoder of the model consists of an observed encoder that processes sensor data information and a static encoder that processes the sensor wearers' biometric static information. Two augmentations are generated using the observed values measured through the sensor, and the generated augmentations are passed to the observed encoder. For the static input value that represents the biometric information of the sensor wearer, important information is extracted, and this extracted static information is included by adding to the input and output values of the observed encoder, respectively. Then, using the context vector c_t, which is the output value of the autoregressive model, a predicted value for $t \in [t+1, t+k]$ is generated and compared to the actual value. Using the values obtained by passing the context vector c_t through the projection head, we learn to maximize the similarity of augmentation from the same sample and minimize the similarity of augmentation from different samples.

A. TIME SERIES DATA AUGMENTATION

A total of 10 time series data augmentation methods were used in our experiment. Here, we compared the performance of various augmentation methods by applying all of the previous time series methods. We implemented eight augmentation methods used by Tang et al. [11] and Saeed et al. [20], and two methods used by Eldele et al. [9]. These 10 augmentation methods are summarized in Table 1.

For each input sample x_i, two augmentation methods are selected from $\mathcal{T}$ to create $x_i^{a_1} \sim \mathcal{T}$ and $x_i^{a_2} \sim \mathcal{T}$.

B. OBSERVED ENCODER

We used a modification of the three-block CNN proposed by Wang et al. [22] as the observed encoder. The observed encoder uses each of $x_i^{a_1}$ and $x_i^{a_2}$ with augmentations as input values. As shown in Equation 1, the observed encoder maps the given input value $\mathbf{x}_i$ to a high-order latent representation $\mathbf{z}_i = f_{obs\_enc}(\mathbf{x})$. Here, each convolutional block comprises a 1D CNN, 1D batch normalization layer, and a rectified linear unit (ReLU) [23], where only the first convolutional block performs max pooling. After passing all convolutional blocks, global average pooling is performed.

$$\begin{aligned} \mathbf{z}_i^{a_1} &= f_{obs\_enc}(\mathbf{x}_i^{a_1}, s_i), \\ \mathbf{z}_i^{a_2} &= f_{obs\_enc}(\mathbf{x}_i^{a_2}, s_i) \end{aligned} \tag{1}$$

During this process, the measured value of each sensor x_i, and s_i, i.e., the biometric information of the sensor wearer,

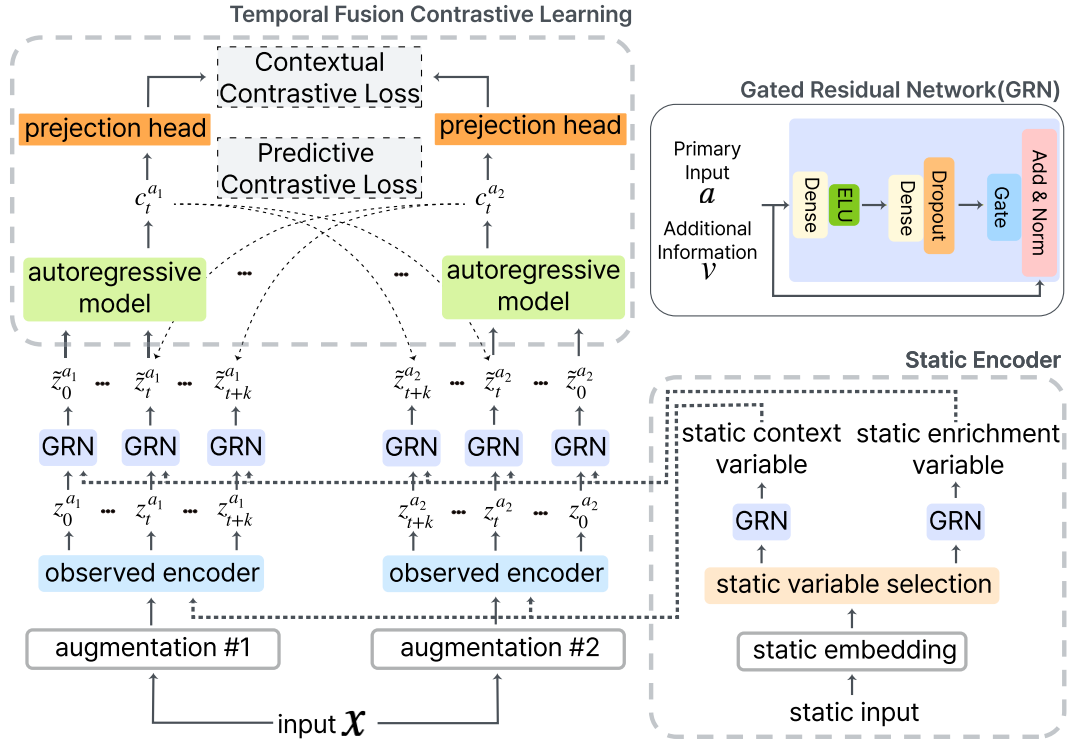

FIGURE 2. Overall architecture of the proposed method.

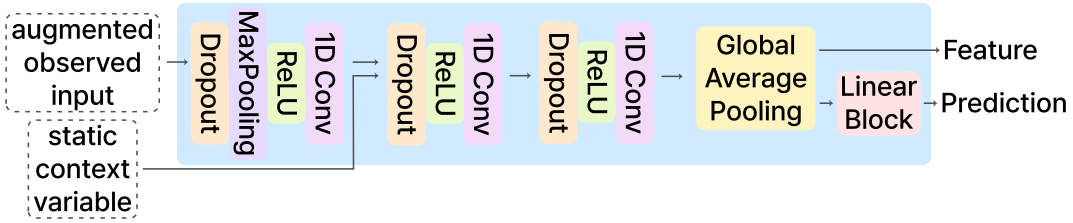

FIGURE 3. Architecture of observed encoder.

are used together as the input values to f_{obs_enc}. The, the biometric information of the sensor wearer is added with the output value $\mathbf{z}_i$ of the observed encoder f_{obs_enc} and passed through a Gated Residual Network (GRN) block. The GRN block strengthens the temporal features along with the use of static information (biometric information). The GRN block is explained in further detail in section III-C. The result according to each augmentation method is as shown in Equation 2.

$$\begin{aligned} \tilde{\mathbf{z}}^{a_1} &= GRN(\mathbf{z}^{a_1}, \mathbf{s}_i), \\ \tilde{\mathbf{z}}^{a_2} &= GRN(\mathbf{z}^{a_2}, \mathbf{s}_i) \end{aligned} \tag{2}$$

C. STATIC ENCODER

The static encoder consists of a static embedding layer, a static variable selection layer, and a GRN layer. The static variable selection and the GRN layers are constructed using the method proposed by Lim et al. [24]. First, when the static input value s_i comes in, a transformation is performed according to the type of input value. Static input value s_i can be a categorical variable or a real variable. A categorical variable can be classified according to categories, e.g., gender, and a real variable takes continuous values, e.g., height and weight. Here, embedding is performed for categorical variables, and linear transformation for continuous variables. Flattening the transformed input $\xi^j \in \mathbb{R}^{d_{\text{model}}}$ can be expressed as $\Xi = [\xi^{(1)}, \ldots, \xi^{(m_s)}]$ and the flattened output Ξ_i is taken as the input value of the static variable selection layer.

The static variable selection layer is composed of GRN blocks, as shown in figure 4. Note that the composition of the GRN block is shown in figure 2. The GRN block uses the residual connection and gating layer together to drive the flow of important information. The GRN is composed of two linear layers using the initial input $\mathbf{a}$ and additional information $\mathbf{v}$ as inputs.

$$\begin{aligned} o_1 &= ELU(\mathbf{W}_{1,\omega}\mathbf{a} + \mathbf{W}_{2,\omega}\mathbf{b} + \mathbf{b}_{1,\omega}) \\ o_2 &= \mathbf{W}_{3,\omega}o_1 + \mathbf{b}_{3,\omega} \\ GRN_\omega(\mathbf{a}, \mathbf{v}) &= LayerNorm(\mathbf{a} + GLU_\omega(o_2)) \end{aligned} \tag{3}$$

In Equation 3, ELU is the Exponential Linear Unit activation function [25] and LayerNorm means layer normalization.

TABLE 1. Augmentation method details.

Jitter	A random noise signal with 0 mean and 0.05 standard deviation is added to the data samples: $x^a = x +$ random noise
Scale	Scaling changes the magnitude of the sample using a randomly selected scalar value ρ: $x^a = x \cdot \rho$
Rotation	Rotating the axis randomly using the rotation matrix R on the 3D axis induces the representation learning irrespective of the sensor's attachment position: $x^a = x * R$
Invert	Sample value is multiplied by -1 to obtain a vertical flip or mirror image of the input signal: $x^a = -x$
Time Flip	Sample is flipped along the time axis to create a full mirror image of the original signal: $x^a = flip(x)$
Permutation	A new time series is generated by dividing the signal into four sub-intervals and subsequently permuting them together: $x^a = permute(x, \#\text{ of segments})$
Warp	A random cubic spline with four fixed points is created and used to set the deviation of the time flow rate[21]. Subsequently, stretching and warping are performed according to the cubic spline: $CS = CubicSplineInterpolation(x, 4)$ $x^a = Warp(x, CS)$
Shuffle	Random channel shuffling based on axis dimension is performed: $x^a = Shuffle(x)$
Jitter and Scale	A random variable is added to the signal, and the magnitude is scaled: $x^a = (x +$ random noise$) \cdot \rho$
Permutation and Jitter	Signal is divided into subsections; subsequently, jitter is performed: $x^a = permute(x, \#\text{ of segments}) +$ random noise

The intermediate gating layer, i.e., a Gated Linear Unit (GLU) [26], provides the flexibility to suppress unnecessary parts. The GLU layer is composed as shown in Equation 4, which is used to quantify the importance of the original input **a**.

$$GLU_\omega(\gamma) = sigmoid(\mathbf{W}_{4,\omega}\gamma + \mathbf{b}_{4,\omega}) \odot (\mathbf{W}_{5,\omega}\gamma + \mathbf{b}_{5,\omega}) \tag{4}$$

The variable selection layer is shown in Equation 5. Here, the weight $\mathbf{v}_s \in \mathbb{R}^{m_s}$ of the variable selection layer is formed by inputting Ξ to the GRN block and passing it through the softmax layer. We introduce non-linearity by passing $\chi^{(j)}$ through each GRN block. The last processed value is multiplied by a weight, and then all of the data-driven features are integrated to form the output of the variable selection layer.

$$\begin{aligned} \mathbf{v}_s &= softmax(GRN_{v_s}(\Xi)) \\ \tilde{\xi}^{(j)} &= GRN_{\tilde{\xi}_{(j)}}(\xi^{(j)}) \\ \tilde{\xi} &= \sum_{j=1}^{m_s} \mathbf{v}_s^{(j)} \tilde{\xi}^{(j)} \end{aligned} \tag{5}$$

Finally, the static context variable and static enrichment variable are extracted by applying two GRN blocks, respectively, to the output of the variable selection layer. Each GRN block extracts a static context vector, and the extracted variables perform different roles. The static context variable is used with the observation values $x_{i,t}$ collected by the sensor in f_{obs_enc} and is used to extract temporal features. The static enrichment variable is used to enhance the temporal features with the extracted latent representation by enhancing the static features of the sensor wearer.

D. TEMPORAL FUSION CONTRASTIVE LEARNING (TFCL)

The TFCL block consists of two main parts: First, next step prediction is performed using the shared information suggested by CPC [14]. Then, using the given latent representation **z**, the autoregressive model g_{ar} integrates all latent representations $z_{\leq t}$. We create the context latent representation $c_t = g_{ar}(z_{\leq t}), c_t \in \mathbb{R}^h$. Here, h represents the hidden dimension of g_{ar}. That is, the future observation value x_{t+k} is not predicted directly using the generative model $p_k(x_{t+k}|c_t)$, but the density ratio is modeled to preserve the mutual information between x_{t+k} and c_t. In order to predict the future time step, the mutual information of x_{t+k} and c_t is calculated using the log-bilinear model, and $f_k(x_{t+k}, c_t) = exp(z_{t+k}^T W_k c_t)$ is used to preserve it. Here, W_k means a linear function that maps c_t to the same dimension as z.

In our experiment, cross-view prediction is attempted using $c_t^{a_1}$ and $c_t^{a_2}$, which are generated through two augmentation methods, i.e., $c_t^{a_1}$ is used to predict the future timestep of $z_j^{a_2}, j \in [t+1, t+k]$, and $c_t^{a_2}$, $z_j^{a_1}, j \in [t+1, t+k]$ is used to predict the future timestep. For the loss function, we use the InfoNCE loss function [14], which is based on NCE loss function [27]. Through the loss function, we attempt to maximize the dot product of values from different samples while minimizing the dot product of the predicted representation from the same samples. Equation 6 is applied to each augmentation method, where $\mathcal{N}_{t,k}$ means mini-batch. The loss function is expressed as follows:

$$\mathcal{L}_{PC}^{a_1} = -\frac{1}{K}\sum_{k=1}^{K} \log \frac{\exp((z_{t+k}^{a_1})^T W_k c_t^{a_2})}{\sum_{n \in \mathcal{N}_{t,k}} \exp((z_n^{a_1})^T W_k c_t^{a_2})} \tag{6}$$

For the autoregressive model, a modified encoder structure of the transformer is used [9]. The autoregressive model consists of multiple multi-head attention (MHA) mechanisms with a multi-layer perceptron (MLP). Unlike the existing transformer [28], which passes through the MHA and proceeds with normalization, a pre-norm residual connection is applied to realize a stable gradient flow [29].

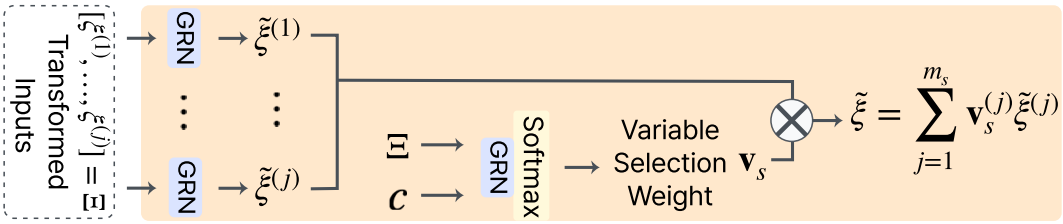

FIGURE 4. Variable selection layer architecture.

TABLE 2. Description of datasets used in experiments (A: accelerometer, G: gyroscope).

Dataset	Sensor Type	#Class	Type of Biometric Information				
			Person ID	Age	Gender	Height	Weight
DLR [31]	A, G	5	✓	✓	✓	✓	
MobiAct [32]	A, G	20	✓	✓	✓	✓	✓

The autoregressive model proceeds in the following order. First, we use $\tilde{z}_{\leq t}$ in the linear function to map the feature to the size of the hidden dimension: $\phi = W_{Trans}\tilde{z}_{\leq t}$. In the autoregressive model, in addition to the input value ϕ, we use a combination of tokens $c \in \mathcal{R}^h$ inspired by the BERT [30]: $\psi_0 = [c; \phi]$. Then, ψ_0 is passed through the L transformer block. Finally, the 0^{th} value of the final output is set as the context vector $c_t = \psi_L^0$ and then used as the input value of the projection head layer.

Next, we add a non-linear projection head to learn discriminative representations. As suggested by Chen et al. [4] in SimCLR, we apply non-linear transformation to the context. Also, using normalized temperature scaled cross entropy (NT-Xent) loss [33], we attempt to maximize the similarity of two augmented views from the same sample and minimize the similarity of augmented views from different samples.

Two augmentations are performed within a given number of mini-batches $\mathcal{N}$; thus, each mini-batch has a total of $2\mathcal{N}$ contexts. For the context vector c_t^k, $c_t^{k^+}$ means a positive sample for c_t^k. That is, $c_t^{k^+}$ is a context vector augmented differently in the same sample as c_t^k, and $(c_t^k, c_t^{k^+})$ is a positive pair. On the other hand, $(2\mathcal{N} - 2)$ contexts except for two positive pairs are considered negative samples. The combination of $c_t^{k^-}$ and c_t^k extracted from $(2\mathcal{N} - 2)$ is a negative pair. Equation 7 defines the NT-Xent loss function [33].

$$\mathcal{L}_{NT} = -\sum_{i=1}^{\mathcal{N}} \log \frac{\exp(\text{sim}(\mathbf{c}_t^k, \mathbf{c}_t^{k^+})/\tau)}{\sum_{m=1}^{2\mathcal{N}} \mathbb{1}_{[m\neq k]} \exp(\text{sim}(\mathbf{c}_t^k, c_t^m)/\tau)} \tag{7}$$

where $\text{sim}(\mathbf{u}, \mathbf{v}) = \mathbf{u}^T\mathbf{v}/\|\mathbf{u}\|\|\mathbf{v}\|$ denotes the cosine similarity between two vectors $\mathbf{u}$ and $\mathbf{v}$. $\mathbb{1}_{[m\neq k]} \in \{0, 1\}$ is an indicator function only for $m \neq k$, and τ is called the annealing schedule.

The overall loss function uses predictive contrastive loss and NT-Xent loss [33] according to each augmentation and is defined as follows:

$$\mathcal{L} = \lambda_1(\mathcal{L}_{PC}^{a_1} + \mathcal{L}_{PC}^{a_2}) + \lambda_2(\mathcal{L}_{NT}) \tag{8}$$

where λ_1 and λ_2 are scalar hyper-parameter values that determine the weight of each error.

IV. EXPERIMENTAL DESIGN

A. USED DATASETS

In this section, we describe the datasets used in the experiment and the preprocessing method. We used the publicly available MobiAct [32] and DLR [31] datasets. Table 2 details the experimental datasets, the sensor used to collect the data, the number of classes in each dataset, and the type of biometric information.

1) DLR DATASET

The DLR dataset [31] was collected using inertial measurement unit (IMU) sensors attached to the subject's belt. Data were acquired from 19 subjects and included a total of 7 classes. We divided the data by the actions performed by each subject and included the subject's personal biometric information. Each file is labeled with a single subject performing multiple actions and the actions performed at each sample point. The sensor data included a three-axis accelerometer, three-axis gyroscope, and three-axis magnitude. In terms of personal biometric information, this dataset includes gender, age, and height information.

Data preprocessing was performed in the following order. First, the data for 50%, 17%, and 33% of people were used as training, validation, and testing datasets, respectively. For the sensor data, the three-axis accelerometer and three-axis gyroscope data were used. After annotating the data by behavior for each file, data exceeding the average time of all sequences(20 s) was cut into subsequences.

2) MOBIACT DATASET

The MobiAct dataset [32] was collected using a smartphone placed in the pocket of the subjects' pants. Here, the data were collected from 67 subjects and included 4 types of falls and 16 activities of daily living (ADL). The files were divided according to the actions performed by each subject, and gender, age, height, and weight are provided as personal

TABLE 3. Experimental results on downstream tasks with and without biometric information (W/: with, W/O: without, Info.: information).

Dataset	Data Format	Tasks	F1 Score	Precision	Recall	Accuracy
DLR	W/ biometric Info.	Fine Tuning	**0.9791**	**0.9800**	**0.9892**	**0.9714**
		Training a Linear Classifier	**0.8726**	**0.8700**	0.7754	0.7341
		Supervised Learning	**0.9450**	**0.9300**	**0.9399**	**0.9309**
	W/O biometric Info.	Fine Tuning	0.9678	0.9700	0.9842	0.9571
		Training a Linear Classifier	0.8643	**0.8700**	**0.8582**	**0.8726**
		Supervised Learning	0.8863	0.8800	0.8875	0.8852
MobiAct	W/ biometric Info.	Fine Tuning	**0.7433**	0.7443	**0.7708**	0.8627
		Training a Linear Classifier	**0.6575**	**0.6603**	**0.6842**	**0.8029**
		Supervised Learning	**0.6913**	**0.7754**	**0.7813**	**0.6967**
	W/O biometric Info.	Fine Tuning	0.7356	**0.7583**	0.7533	**0.8821**
		Training a Linear Classifier	0.5499	0.6149	0.5794	0.6527
		Supervised Learning	0.6640	0.7722	0.6966	0.6900

information about the human body. The sensor data included three-axis accelerometer data, three-axis gyroscope data, and three-axis orientation sensor data.

To preprocess the MobiAct dataset, we first excluded subjects' personal biometric information. We used the data from 48, 7, and 9 subjects as training, validation, and testing datasets, respectively. With the MobiAct dataset, the time length of each file varied; thus, if data exceeded the average time of all sequences (35 *s*), the length was trimmed.

B. IMPLEMENTATION

We performed three subtasks in our experiment, i.e., **Fine Tuning**, **Training a Linear Classifier** for downstream tasks, and **Supervised Learning**. The **Fine Tuning** was performed using a pre-trained encoder by SSL. We retrained all weights of the encoder in response to the task-specific gradients computed from the labeled data. The **Training a Linear Classifier** involved retraining only the linear layer as a task-specific layer (except for the last linear layer in the pre-trained encoder) after freezing all weights. The **Supervised Learning** involved training the encoders from scratch without using pre-trained encoder. In terms of evaluation metrics, the F1 score, precision, recall, and accuracy metrics were used. For F1 score, the macro-averaged F1 score was applied to observe the effect on an imbalanced dataset.

To compare performance with that of baseline models, we used three existing contrastive learning methods for the HAR task. Fine tuning and training a linear classifier for downstream tasks were carried out using the encoder, an autoregressive model. Here, all hyperparameters were set the same as that for the proposed TFCL, and the encoder and autoregressive models were modified using the common methods in the previous studies. Detailed hyperparameter settings are presented in Table 4.

For the SimCLR proposed by Tang et al. [11] to HAR, three convolutional blocks consisting of a 1D convolutional layer, ReLU, and dropout layer were used for the encoder, and two linear layers were used for the autoregressive model. The contrastive SSL approach for sensor-based human activity recognition (CSSHAR), proposed by Khaertdinov et al. [12], used three convolution blocks consisting of a 1D convolutional layer, 1D batch normalization, ReLU, and the encoder part of the transformer [28]. The autoregressive model for CSSHAR comprises ReLU and a linear layer. For SimCLR-based HAR and CSSHAR, the results were measured using only encoders in the fine tuning and training a linear classifier for downstream tasks. In the case of CPC-based HAR, which was proposed by Haresamudram et al. [13], three convolutional blocks consisting of a 1D convolutional layer, ReLU, and a dropout layer were used as the encoder. For the autoregressive model, a GRU [34] was used.

TABLE 4. Hyperparameter settings (Kernel size for 1D CNN).

Dataset	Kernel size	h	d_{model}	k	Batch size	Learning rate	λ_1	λ_2
DLR	50	128	256	20	32	0.0001	1.5	0.7
MobiAct	22	128	256	20	32	0.001	1.5	0.7

V. EXPERIMENTAL RESULTS

To prove the effectiveness of the proposed method, an experiment was conducted according to the presence and absence of the biometric information of the sensor wearer. Table 3 shows the results when the subjects' biometric information was included and excluded in each dataset. The results in Table 3 are arranged based on the highest performance among the best combination of time series data augmentations presented in *III-A*. Here, fine tuning and training a linear classifier for downstream tasks, and supervised learning task were carried out, and the cases with the best results are shown in bold. It can be seen that the DLR and MobiAct datasets show better performance for all evaluation metrics when the biometric information was included.

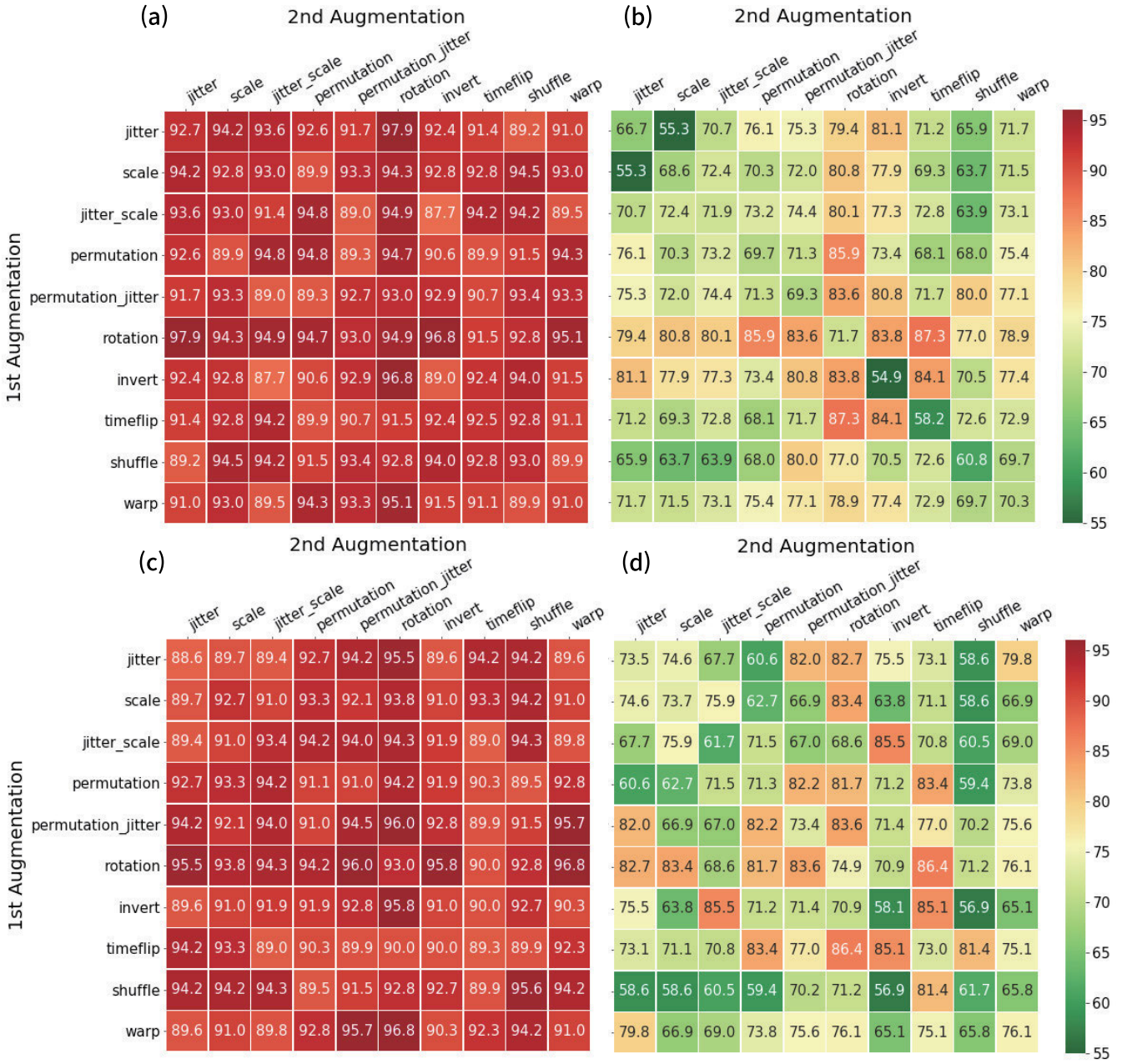

FIGURE 5. Macro F1 score results for different augmentation combinations on the DLR dataset. Diagonal matrix values refer to values tested using only one augmentation method. (a) F1 score of fine tune downstream task with biometric information of sensor wearer. (b) F1 score of train linear downstream task with biometric information of sensor wearer. (c) F1 score of fine tune downstream task without biometric information of sensor wearer. (d) F1 score of train linear downstream task without biometric information of sensor wearer.

With the DLR dataset, when biometric information was included, the F1 scores were 0.9791, 0.8726, and 0.9450 for each task, and these results are better than those obtained when the biometric information was not included. The highest performance difference appeared in supervised learning, which confirms that biometric information has a significant effect even with supervised learning. In the experiment with biometric information as static information, we found that the training a linear classifier task exhibited the best performance when the *rotation* and *time flip* augmentation methods were used, and for fine tuning task, the F1 score obtained with the *jitter* and *rotation* augmentation methods was 0.9791(best performance). When biometric information was not used in the fine tuning task, the combination of the *rotation* and *warp* augmentation methods performed the best in the same way as the train linear.

On the MobiAct dataset, we confirmed that performance was better when human biometric information was included in most of all tasks. In the case of the MobiAct dataset, when the human biometric information was used, the invert and

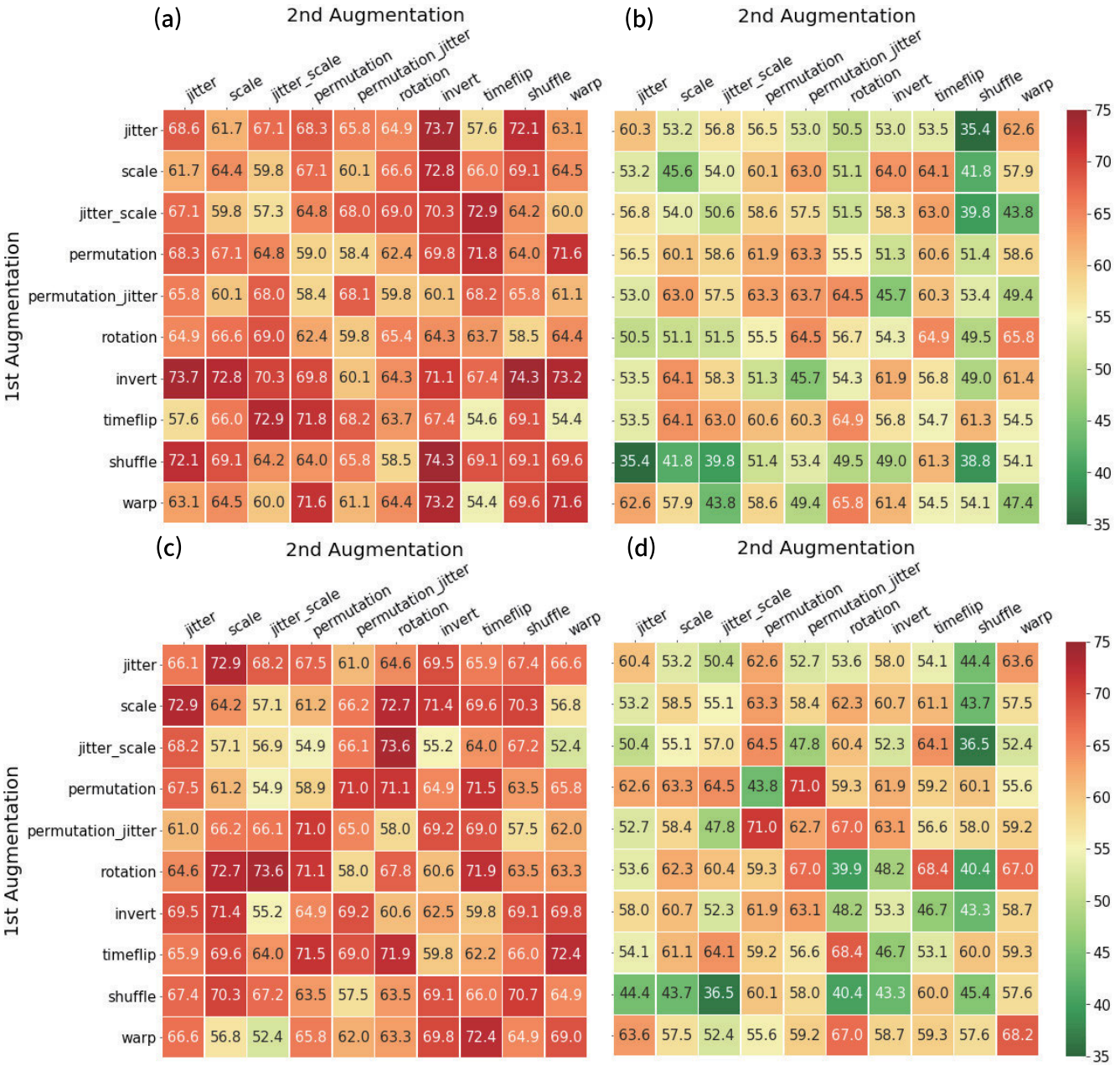

FIGURE 6. Macro F1 score results for different augmentation combinations on the MobiAct dataset. Diagonal matrix values refer to values tested using only one augmentation method. (a) F1 score of fine tune downstream task with biometric information for sensor wearer. (b) F1 score of train linear downstream task with biometric information for sensor wearer. (c) F1 score of fine tune downstream task without biometric information for sensor wearer. (d) F1 score of train linear downstream task without biometric information for sensor wearer.

shuffle augmentation methods exhibited the best performance in the fine tuning task. With training a linear classifier task, the best performance was obtained with the rotation and warp augmentation methods. In the case of learning without human biometric information, it showed the highest performance when the *jitter and scale* and *rotation* augmentation methods were used in fine tuning task, and when *permutation* and *jitter* augmentation methods were used with the training a linear classifier task.

Next, we compared the performance of the proposed method against other models proposed in previous HAR tasks using SSL. As existing models, we used SimCLR on HAR [11], which applies SimCLR to HAR as proposed by Tang et al., CSSHAR [12], and CPC [14] applied to HAR, CPC on HAR [13]. Table 5 summarizes the experimental results. As can be seen from Table 5, the proposed method outperformed the existing models in terms of all evaluation metrics on all tasks.

On the DLR dataset, the F1 score obtained by the proposed TFCL method on the fine tuning task was 0.9791, which shows the best performance of all methods. The F1 score of fine tuning task was higher by 15.2 percentage points($p.p$)

TABLE 5. Experimental results on DLR and MobiAct datasets with other SSL based HAR models.

Dataset	Method	Tasks	F1 Score	Precision	Recall	Accuracy
DLR	TFCL(*ours*)	Fine Tuning	**0.9791**	**0.9800**	**0.9892**	**0.9714**
		Training a Linear Classifier	**0.8726**	**0.8700**	**0.8726**	**0.8726**
		Supervised Learning	**0.9450**	**0.9300**	**0.9399**	**0.9309**
	SimCLR on HAR [11]	Fine Tuning	0.8752	0.8600	0.8714	0.8840
		Training a Linear Classifier	0.3707	0.3700	0.3966	0.3930
		Supervised Learning	0.7901	0.8000	0.8460	0.7675
	CSSHAR [12]	Fine Tuning	0.8610	0.8600	0.8647	0.8589
		Training a Linear Classifier	0.7470	0.7700	0.7958	0.7269
		Supervised Learning	0.8863	0.8800	0.8875	0.8852
	CPC on HAR [13]	Fine Tuning	0.7425	0.7000	0.7642	0.7331
		Training a Linear Classifier	0.5812	0.5900	0.6158	0.5946
		Supervised Learning	-	-	-	-
MobiAct	TFCL(*ours*)	Fine Tuning	**0.7433**	**0.7443**	**0.7708**	**0.8627**
		Training a Linear Classifier	**0.6575**	**0.6603**	**0.6842**	**0.8029**
		Supervised Learning	**0.6913**	**0.7754**	**0.7813**	**0.6967**
	SimCLR on HAR [11]	Fine Tuning	0.4491	0.5250	0.4856	0.4639
		Training a Linear Classifier	0.2121	0.2746	0.2348	0.2311
		Supervised Learning	0.5378	0.6058	0.6061	0.5481
	CSSHAR [12]	Fine Tuning	0.0105	0.0985	0.0299	0.0508
		Training a Linear Classifier	0.0234	0.1115	0.0253	0.0642
		Supervised Learning	0.0159	0.1050	0.0093	0.0542
	CPC on HAR [13]	Fine Tuning	0.1009	0.2294	0.1162	0.1250
		Training a Linear Classifier	0.0520	0.1454	0.0506	0.0825
		Supervised Learning	-	-	-	-

than average of existing methods. With training a linear classifier task, the F1 score obtained by the proposed TFCL method was 0.8726, which was 30.6 *p.p* higher on average than existing methods. Supervised learning task also had the highest F1 score of TFCL at 0.9450. Among the compared existing methods, SimCLR on HAR demonstrated the best performance on fine tuning and supervised learning tasks, and CSSHAR showed the best performance on the training a linear classifier task.

On the MobiAct dataset, the proposed method performed better on all tasks compared to the other existing models. For fine tuning task, the F1 score of TFCL was 0.7433, which was 55.46 *p.p* higher on average than that of the existing methods. In training a linear classifier and supervised learning tasks, TFCL performed the best with 0.6575 and 0.6913 compared with other existing models, which was 56.16 *p.p* and 45.94 *p.p* higher on average than the existing models, respectively.

Next, we compared the performance according to the change in the augmentation method. Figure 5 shows the experimental results obtained by combining 10 augmentation methods on the DLR dataset. Here, we found that the proposed method including the sensor wearer's biometric information was superior in 30 out of 55 augmentation combinations for both the fine tuning and training a linear classifier tasks. For the fine tuning task, the averages F1 score with and without biometric information were 0.9252 and 0.9232, respectively. When the biometric information was included, the maximum F1 score was 0.9791 with the combination of *jitter* and *rotation* augmentation methods.

TABLE 6. Computational costs of model size (MB) and average time per epoch (s); linear indicates training a linear classifier tasks.

Dataset	SSL		Fine Tuning		Linear	
	Size	Time	Size	Time	Size	Time
DLR	17.203(MB)	166s	11.589(MB)	80s	11.589(MB)	80s
MobiAct	16.708(MB)	587s	11.099(MB)	85s	11.099(MB)	84s

Figure 6 shows the results of an experiment that combined 10 augmentation methods on the MobiAct dataset. The experimental results show that among 55 augmentation combinations, 34 combinations exhibited better performance in fine tuning task when biometric information was included. The F1 score with and without biometric information in fine tuning task were 0.6564 and 0.6528, respectively. Performance was slightly better when the biometric information was included. For training a linear classifier task, when the biometric information was included, the performance was better in 25 out of 55 combinations. The combination of augmentation methods that showed the best performance

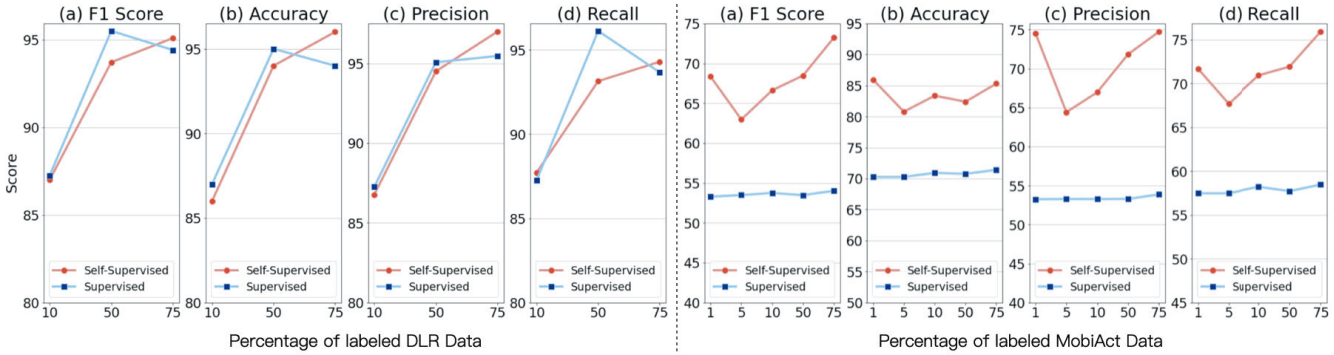

FIGURE 7. Performance results of transfer learning and supervised learning according to the ratio of the used DLR and MobiAct dataset.

TABLE 7. Comparative performance analysis with static inputs (s) and different annealing schedule (τ).

	Age	Gender	Height	F1 Score	Precision	Recall	Accuracy
s	✓			0.9510	0.9482	0.9543	0.9500
		✓		0.9338	0.9544	0.9229	0.9400
			✓	0.9528	0.9467	0.9629	0.9500
	✓	✓		0.9693	0.9735	0.9657	0.9700
		✓	✓	0.9162	0.9230	0.9114	0.9200
	✓		✓	0.9205	0.9478	0.9086	0.9300
	✓	✓	✓	**0.9791**	**0.9800**	**0.9892**	**0.9714**
τ		0.1		0.9559	0.9795	0.9429	0.9600
		0.2		**0.9791**	0.9800	**0.9892**	**0.9714**
		0.3		0.9489	0.9526	0.9457	0.9500
		0.4		0.9600	0.9600	0.9600	0.9600
		0.5		0.9600	0.9600	0.9600	0.9600
		0.6		0.9118	0.9098	0.9143	0.9100
		0.7		0.9510	0.9482	0.9543	0.9500
		0.8		0.9678	**0.9842**	0.9571	0.9700
		0.9		0.9600	0.9600	0.9600	0.9600
		1.0		0.9693	0.9735	0.9657	0.9700

for fine tuning task was *invert* and *shuffle*. *rotation* and *warp* augmentation methods were used in training a linear classifier task. The computational costs associated with the best combination of data augmentation for each dataset and task are presented in Table 6. According to the cost measurements, despite the DLR dataset having generally larger model parameters than MobiAct, the great volume of data in MobiAct resulted in long average times per epoch.

Table 7 presents the findings of the experiments conducted on the DLR dataset. These experiments explored performance variations based on the composition of static inputs (s) and different annealing schedules (τ). The upper section of the table illustrates the performance impact of including static input variables such as age, gender, and height. The inclusion of all variables yielded the highest F1 score of 0.9791, suggesting a positive effect on performance.

The lower section of the table evaluates the performance across various annealing schedules (τ), indicating fluctuations in F1 scores and other performance metrics with different τ settings. Notably, a τ value of 0.2 showed the best performance, with an F1 score of 0.9791, outperforming other τ settings. These results underscore the significant influence of the composition of static(biometric) information and adjustment of the annealing schedule on the model's performance.

TABLE 8. Result of transfer learning on each dataset.

Train Dataset	Test Dataset	F1 Score	Precision	Recall	Accuracy
DLR	DLR	0.9791	0.9800	0.9892	0.9714
MobiAct → DLR	DLR	0.8997	0.9024	0.8978	0.8900
MobiAct	MobiAct	0.7433	0.7443	0.7708	0.8627
DLR → MobiAct	MobiAct	0.6931	0.7426	0.7208	0.7900

Next, we examine the efficiency of transfer learning between different dataset domains by using SSL of the proposed TFCL method. Table 8 shows the results of fine tuning task using TFCL method on each dataset. This experiment was conducted to investigate whether representation learning was possible even if retraining was attempted after representation learning using another dataset domain. From experimental results, when representations learned using the MobiAct dataset were retrained using the training data of the DLR dataset, the F1 score was 0.8997, which was 7.94 $p.p$ lower than when both the training and validation sets used the DLR dataset. When presentations learned using the DLR dataset were transferred to the MobiAct dataset, the F1 score was 0.6931, which was 5.02 $p.p$ lower than when both training and validation sets were performed using the MobiAct dataset. Comparing the case with and without transfer learning, we found that the proposed TFCL method guarantee the peformance of transfer learning between different dataset domains. In the case transfer learning from MobiAct to DLR datasets, transfer learning of TFCL from diverse and sufficient source dataset guarantees better performance.

Finally, the results of the transfer learning and the supervised learning were compared according to the ratio of training dataset. After we conducted various experiments changing the ratio of training dataset, we attempted to confirm the effect of transfer learning of TFCL.

Figure 7 shows the differences between the supervised learning and transfer learning of TFCL according to the ratio of the DLR and MobiAct datasets, respectively. In both figures, the horizontal axis indicates the ratio of data used, and the vertical axis indicates the resulting value. As can be seen, transfer learning of TFCL exhibited similar or superior performance on both datasets compared to supervised learning. With DLR dataset, because the data are small, the experiment was conducted using only 10%, 50%, and 75% of the total data. As a result of the experiment, transfer learning of TFCL showed better performance on most results. With MobiAct dataset, the experiment was conducted by adjusting the five dataset ratios, and transfer learning of TFCL demonstrated better results in all experiments. Through these experiments, we were able to prove that transfer learning of TFCL is more effective in learning than supervised learning even when only a small proportion of the data is used.

VI. CONCLUSION

From the proposed TFCL method, we evaluated the effect on HAR performance of a sensor wearer's biometric information using SSL. The proposed TFCL method uses a sample to generate two augmented samples and takes each sample to perform contrastive learning. First, the sensor wearer's biometric information is extracted effectively through a static encoder and used as the input values in the observed encoder. Then, important static information is extracted again, with the latent representation from the observed encoder. We use the extracted context vector to predict future latent representations to learn temporal features. Finally, by comparing each augmentation method, we ensure that learning is performed such that the contextual similarity of representation is maximized for the same sample. The experimental results showed that better performance was obtained on all datasets when the biometric information was included. Furthermore, the proposed mothod outperformed other existing models in terms of most evaluation metrics. We have proven that transfer learning of TFCL can be performed efficiently with a sensor wearer's biometric information, even if the amount of labeled data is small.

Although the experimental results demonstrated that the biometric information inclusion consistently enhanced the TFCL performance across all datasets and that the proposed method surpassed other existing models in most evaluation metrics, TFCL has a critical limitation. The complex transformer-based architecture of the TFCL model lacks transparency, leading to potential issues with trust and reliability.

Considerably, future research should focus on incorporating eXplainable AI (xAI) techniques to provide transparency and enhance interpretability. Developing methods to unravel the decision-making process of such complex models will improve their trustworthiness and pave the way for a deeper understanding of how biometric data influences learning processes. This direction of research is crucial for advancing the field and ensuring that these sophisticated models can be utilized effectively and responsibly in real-world applications.

REFERENCES

[1] *Wearable Technology Market Size, Share & Trends Analysis Report by Product (Wrist-Wear, Eye-Wear & Head-Wear, Foot-Wear, Neck-Wear, Body-Wear), by Application, by Region, and Segment Forecasts, 2021–2028*, Grand View Research, San Francisco, CA, USA, 2021.

[2] *Expected NUM of Elderly, Injuries and Falls*, Centers for Disease Control and Prevention, Atlanta, GA, USA, 2020.

[3] M. Z. Uddin and A. Soylu, "Human activity recognition using wearable sensors, discriminant analysis, and long short-term memory-based neural structured learning," *Sci. Rep.*, vol. 11, no. 1, pp. 1–15, Aug. 2021.

[4] 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.

[5] R. D. Hjelm, A. Fedorov, S. Lavoie-Marchildon, K. Grewal, P. Bachman, A. Trischler, and Y. Bengio, "Learning deep representations by mutual information estimation and maximization," 2018, *arXiv:1808.06670*.

[6] 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. (CVPR)*, Jun. 2020, pp. 9726–9735.

[7] J. Grill, "Bootstrap your own latent—A new approach to self-supervised learning," in *Proc. Adv. Neural Inf. Process. Syst. (NeurIPS)*, 2020, pp. 21271–21284.

[8] X. Chen and K. He, "Exploring simple Siamese representation learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 15745–15753.

[9] E. Eldele, M. Ragab, Z. Chen, M. Wu, C. K. Kwoh, X. Li, and C. Guan, "Time-series representation learning via temporal and contextual contrasting," 2021, *arXiv:2106.14112*.

[10] S. Tonekaboni, D. Eytan, and A. Goldenberg, "Unsupervised representation learning for time series with temporal neighborhood coding," 2021, *arXiv:2106.00750*.

[11] C. Ian Tang, I. Perez-Pozuelo, D. Spathis, and C. Mascolo, "Exploring contrastive learning in human activity recognition for healthcare," 2020, *arXiv:2011.11542*.

[12] B. Khaertdinov, E. Ghaleb, and S. Asteriadis, "Contrastive self-supervised learning for sensor-based human activity recognition," in *Proc. IEEE Int. Joint Conf. Biometrics (IJCB)*, Aug. 2021, pp. 1–8.

[13] H. Haresamudram, I. Essa, and T. Plötz, "Contrastive predictive coding for human activity recognition," *Proc. ACM Interact., Mobile, Wearable Ubiquitous Technol.*, vol. 5, no. 2, pp. 1–26, Jun. 2021.

[14] A. van den Oord, Y. Li, and O. Vinyals, "Representation learning with contrastive predictive coding," 2018, *arXiv:1807.03748*.

[15] Y. Guan and T. Plötz, "Ensembles of deep LSTM learners for activity recognition using wearables," *Proc. ACM Interact., Mobile, Wearable Ubiquitous Technol.*, vol. 1, no. 2, pp. 1–28, Jun. 2017.

[16] Y. Zhao, R. Yang, G. Chevalier, X. Xu, and Z. Zhang, "Deep residual bidir-LSTM for human activity recognition using wearable sensors," *Math. Problems Eng.*, vol. 2018, pp. 1–13, Dec. 2018.

[17] S. Yao, S. Hu, Y. Zhao, A. Zhang, and T. Abdelzaher, "DeepSense: A unified deep learning framework for time-series mobile sensing data processing," in *Proc. 26th Int. Conf. World Wide Web*, Apr. 2017, pp. 351–360.

[18] S.-M. Lee, S. Min Yoon, and H. Cho, "Human activity recognition from accelerometer data using convolutional neural network," in *Proc. IEEE Int. Conf. Big Data Smart Comput. (BigComp)*, Feb. 2017, pp. 131–134.

[19] R. Xi, M. Hou, M. Fu, H. Qu, and D. Liu, "Deep dilated convolution on multimodality time series for human activity recognition," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2018, pp. 1–8.

[20] A. Saeed, T. Ozcelebi, and J. Lukkien, "Multi-task self-supervised learning for human activity detection," *Proc. ACM Interact., Mobile, Wearable Ubiquitous Technol.*, vol. 3, no. 2, pp. 1–30, Jun. 2019.

[21] H. Li, X. Wan, Y. Liang, and S. Gao, "Dynamic time warping based on cubic spline interpolation for time series data mining," in *Proc. IEEE Int. Conf. Data Mining Workshop*, Dec. 2014, pp. 19–26.

[22] Z. Wang, W. Yan, and T. Oates, "Time series classification from scratch with deep neural networks: A strong baseline," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, May 2017, pp. 1578–1585.
[23] A. F. Agarap, "Deep learning using rectified linear units (ReLU)," 2018, *arXiv:1803.08375*.
[24] B. Lim, S. O. Arik, N. Loeff, and T. Pfister, "Temporal fusion transformers for interpretable multi-horizon time series forecasting," 2019, *arXiv:1912.09363*.
[25] D.-A. Clevert, T. Unterthiner, and S. Hochreiter, "Fast and accurate deep network learning by exponential linear units (ELUs)," 2015, *arXiv:1511.07289*.
[26] Y. N. Dauphin, "Language modeling with gated convolutional networks," in *Proc. Int. Conf. Mach. Learn.*, 2017, pp. 933–941.
[27] Z. Ma and M. Collins, "Noise contrastive estimation and negative sampling for conditional models: Consistency and statistical efficiency," 2018, *arXiv:1809.01812*.
[28] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Proc. Adv. Neural Inf. Process. Syst.*, 30, 2017, pp. 6000–6010.
[29] Q. Wang, B. Li, T. Xiao, J. Zhu, C. Li, D. F. Wong, and L. S. Chao, "Learning deep transformer models for machine translation," 2019, *arXiv:1906.01787*.
[30] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," 2018, *arXiv:1810.04805*.
[31] K. Frank, M. J. Vera Nadales, P. Robertson, and T. Pfeifer, "Bayesian recognition of motion related activities with inertial sensors," in *Proc. 12th ACM Int. Conf. Adjunct Papers Ubiquitous Comput. Adjunct*, Sep. 2010, pp. 445–446.
[32] G. Vavoulas, C. Chatzaki, T. Malliotakis, M. Pediaditis, and M. Tsiknakis, "The MobiAct dataset: Recognition of activities of daily living using smartphones," in *Proc. Int. Conf. Inf. Commun. Technol. Ageing Well e-Health*, 2016, pp. 143–151.
[33] K. Sohn, "Improved deep metric learning with multi-class N-pair loss objective," in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, pp. 1857–1865.
[34] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," 2014, *arXiv:1412.3555*.

INKYUNG KIM received the B.S. degree in political science and international relationship with a minor in computer science from Kookmin University, South Korea, in 2020, and the M.S. degree from the College of Computer Science, Kookmin University, in 2022. She is currently a Research Engineer with Hyundai Motor Company, South Korea. Her research interests include time series data analysis, healthcare data analytics, and machine learning.

JUWAN LIM received the B.S. degree from the College of Computer Science, Kookmin University, South Korea, in 2022, where he is currently pursuing the M.S. degree. His research interests include time series data analysis, machine learning, deep learning, and artificial intelligence (AI).

JAEKOO LEE received the Ph.D. degree in electrical and computer engineering from Seoul National University, South Korea, in 2018. From 2011 to 2013, he was with LG Electronics Research and Development Campus, Seoul, South Korea. He was a Data Scientist with SK Telecom Corporation, in 2018. He is currently an Assistant Professor with the College of Computer Science, Kookmin University, South Korea. His research interests include artificial intelligence (AI), machine learning, deep learning, and data science and their applications.