Self-supervised Learning for Complex Activity Recognition through Motif Identification Learning

Qingxin Xia *†, Jaime Morales† , Yongzhi Huang * , Takahiro Hara † ,
Kaishun Wu * , Hirotomo Oshima‡, Masamitsu Fukuda‡, Yasuo Namioka [2] § , Takuya Maekawa [1] † ,
*Information Hub, Hong Kong University of Science and Technology (Guangzhou), Guangzhou, China
†Information Science and Technology, Osaka University, Osaka, Japan
‡Corporate Manufacturing Engineering Center, Toshiba Corporation, Tokyo, Japan
§ Advanced Institute of Industrial Technology, Tokyo, Japan

Abstract—Owing to the cost of collecting labeled sensor data, self-supervised learning (SSL) methods for human activity recognition (HAR) that effectively use unlabeled data for pretraining have attracted attention. However, applying prior SSL to COMPLEX activities in real industrial settings poses challenges. Despite the consistency of work procedures, varying circumstances, such as different sizes of packages and contents in a packing process, introduce significant variability within the same activity class. In this study, we focus on sensor data corresponding to characteristic and necessary actions (sensor data motifs) in a specific activity such as a stretching packing tape action in an assembling a box activity, and propose to train a neural network in self-supervised learning so that it identifies occurrences of the characteristic actions, i.e., Motif Identification Learning (MoIL). The feature extractor in the network is subsequently employed in the downstream activity recognition task, enabling accurate recognition of activities containing these characteristic actions, even with limited labeled training data. The MoIL approach was evaluated on real-world industrial activity data, encompassing the state-of-the-art SSL tasks with an improvement of up to 23.85% under limited training labels. Our code is publicly available at https://github.com/qingxinxia/MoIL.git.

Index Terms—Activity recognition, self-supervised learning, wearable sensor, industrial domain

1 Introduction

Human activity recognition (HAR) using wearable devices has garnered increasing attention in various real-world domains in recent years, such as healthcare [1], [2], fitness [3], and industrials [4]. In industrial settings, commercial wearable devices equipped with various sensors, such as accelerometers, have become increasingly popular for their effectiveness in directly monitoring human hand movements and are more resistant to obstacles than cameras. Consequently, HAR research focused on human workers using wearable sensors has gained prominence in advancing Industry 4.0, particularly in areas such as process management and the automation of production lines [5]–[10].

Figure 1 shows a typical example of packaging work with complex activities. The worker performs a set of operations (activities) repetitively, with each operation consisting of a sequence of small actions. For example, an operation of "assemble box" consists of atomic actions such as folding all sides of the box, stretching a packing tape, applying the packaging tape, and flipping the box. A complete packaging task is called a period, and Figure 1 depicts two periods of acceleration data. Different from recognizing activities of daily life, complex work activity recognition in industrial settings presents several challenges: (1) **Complex sensor data**: Industrial operations consist of varied small actions, leading to highly complex sensor data, unlike the periodic waveforms seen in daily activities like walking. (2) **Highly variable sensor data**: The same operation, such as "assemble box," can vary significantly in duration and actions due to differences in the items being packed. (3) **Limited training data**: Due to the aforementioned challenges, training the HAR model for manual work requires a larger amount of labeled data. However, in the actual industrial domain, it is difficult to collect a large amount of labeled training data like image databases, such as ImageNet.

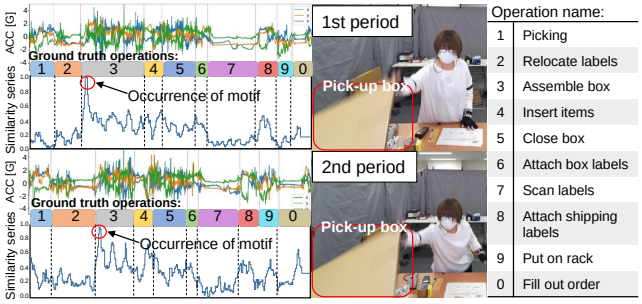


Fig. 1. Example of an occurrence of a motif corresponds to the pick-up box action in two periods. The figure shows a typical example of packaging work with complex activities. The worker performs a set of operations (activities) repetitively, with each operation consisting of a sequence of small actions. For example, an operation of 'assemble box' consists of atomic actions such as folding all sides of the box, stretching a packing tape, applying the packaging tape, and flipping the box.

Recent studies attempted to address the issue of limited training data by employing self-supervised learning (SSL),

1. This is the corresponding author.
2. This work was done while the author worked at Toshiba.

which leverages unlabeled sensor data for pretraining feature extraction layers through pretext tasks. This is advantageous because unlabeled data can be easily collected. State-of-the-art SSL methods involving masked reconstruction [11], [12] and contrastive learning [13], have been successfully applied to HAR for daily life activities [14]–[16].

However, activities in industrial domains are highly complex and variable (refer to challenges 1 and 2), so it is a challenging task to apply existing SSL to complex work activities. For instance, contrastive learning-based approaches depend on data augmentation techniques like rotation and cropping, which are derived from image processing. These techniques often lose their physical relevance when applied to time-series data. Besides, masked reconstruction-based approaches [11] involve randomly masking portions of input data and reconstructing the masked segments as a pretext task to learn local temporal dependencies in time-series data. There is no guarantee that important features are properly learned, such as features of actions that are representative of an activity regardless of the different sizes of the items to be packed. Consequently, there is a clear need for a novel SSL approach that is specifically tailored to the complex and variable nature of activities, such as in industrial domains.

In this study, we propose a new SSL approach, named Motif Identification Learning (MoIL), for complex work activity recognition in the industrial domain based on the characteristics of the complex activities. As mentioned above, the operations (complex work activities) involve several atomic actions. An atomic action exhibits a characteristic waveform in sensor data, called a **motif**. A motif that consistently occurs in an operation of each period can be helpful to roughly identify the operation. For example, the stretching packing tape action is unique and always occurs in the assembling box operation in the packaging task even though the packaging task is highly variable. Automatically detecting such motifs in the acceleration data is useful for precise activity recognition. By designing an SSL pretext task to learn latent representation regarding the characteristic motifs, we can improve the performance of the downstream task (i.e., complex activity recognition) with a small amount of annotated data.

In this study, we perform MoIL to identify occurrences of key motifs within an input time window of unlabeled acceleration data. First, we find key motifs that can be useful for the downstream task from unlabeled acceleration data. Thereafter, for each key motif, a similarity series that shows the occurrences of the corresponding key action in the acceleration data, is computed using a motif detection algorithm from each period of the original acceleration data. Figure 1 shows similarity series computed from two different periods for a motif corresponding to a pick-up box action. When the value in the similarity series is high, an action similar to the motif action occurs at that time. We then train the neural network for the pretext task (MoIL) so that the network processes an input time window of acceleration data and outputs similarity series of key motifs corresponding to the input. Thus, the network can be trained to detect key motifs that correspond to characteristic atomic actions. Therefore, the network's feature extraction layers can effectively be used in the downstream task of recognizing complex operations consisting of multiple key actions. In our method, key motifs are selected based on the following idea: (1) occurrences of a key motif, i.e., characteristic action, should be robust across different periods, (2) the relative temporal location of key motifs should be consistent, and (3) a key motif should be distinguished from other actions.

Overall, our method presents a novel pretext task that identifies occurrences of key motifs to learn activity patterns for the recognition of complex work activities effectively. To the best of our knowledge, this is the first work to design an SSL approach for complex work activity recognition based on sensor data, leveraging the intrinsic attributes of the task and time-series data at hand. We make the following contributions:

(1) We proposed a new SSL approach, MoIL, for complex activity recognition in industrial settings. To our knowledge, this is the first approach that guides a model in learning latent representation by identifying the occurrence of motifs, which can effectively improve complex activity recognition with limited labeled data.

(2) We identified the traits characteristic of industrial activities and proposed metrics for selecting good motifs for MoIL without using activity labels: robustness, consistency, and uniqueness.

(3) We discussed the performance of SSL methods that rely on data augmentation and data reconstruction when applied to complex activity sensor data with varying characteristics. Our analysis illustrates the impact of different pre-training methods on complex activity recognition.

(4) We demonstrated the effectiveness of MoIL using sensor data collected in logistics and manufacturing centers, which outperforms the state-of-the-art SSL methods with an improvement of up to 24%.

2 Related Work

2.1 HAR with Wearables

Owing to the development of sensing technologies in recent years, sensors such as accelerometers [17], gyroscopes [18], heart rate sensors [19], electrodermal activity sensors [20], and microphones [21] can be easily equipped on wearable devices to support HAR applications. These developments have raised attention in the ubicomp community regarding research on activity recognition using data collected from wearable devices.

Francisco et al. [22] introduced DeepConvLSTM, a Convolutional (Conv) and LSTM Recurrent Neural Networks to recognize daily life activities. This architecture extracts long and short terms of relationships of sensor data, which became the basic network structure of HAR with wearable sensor data. Building on the DeepConvLSTM, subsequent studies have aimed to improve HAR model performance by focusing on the characteristics of activities. For instance, Chen et al. [23] developed an online learning scheme for recognizing daily living activities, as such daily activities tend to be relatively consistent for people. Other studies enhance HAR performance based on the repetitive nature [24], biomechanical patterns [25], and enviromnents [26]. However, existing techniques mainly designed for daily life activities, which usually fail when employing on complex activity recognition [1], [27]. Therefore, there is a demand

for us to develop methods for complex activity recognition on specific domain.

2.2 HAR in Industrial Domains

Activity recognition using wearable devices has gained significant attention in industrial domains due to its wide range of potential applications [28]. However, activities in industrial domains are more complicated than daily activities [29], it is a challenging task for existing daily activity recognition approaches to achieve robust and reliable performance in industrial domains.

In situations where insufficient activity labels are provided, previous studies employed fully supervised learning methods, leveraging the work instruction document and key actions of industrial tasks to recognize work activities. For instance, Yoshimura et al. [7] utilized the work instruction document to build an activity transition map to penalize the activities that has high transition cost. Besides, several prior studies have explored motif detection algorithms [6], [8], [30] for better recognizing/understanding work activities. For example, Moral et al. [6] identified motifs according to labeled data and introduced a motif-guided U-Net model to recognize activities in a logistic center. Xia et al. [31], and recognize factory work activities in an unsupervised manner by tracking occurrences of motifs that selected based on the work instruction document. In contrast to these approaches, this study focuses on SSL for complex activities and aims to find key motifs without relying on work instruction documents or operation labels, thus offering a more flexible and scalable solution for activity recognition in industrial settings.

2.3 SSL in HAR

SSL has shown good performance in computer vision [32], natural language processing [33], and speech processing [34]. To the best of our knowledge, Saeed et al. [35] first explored SSL for HAR. They separately applied eight transformation strategies (data augmentation methods) [36] to time-series and then designed a binary classification head for each transformation with a shared encoder to identify if the input data was transformed or not. In a recent study [14], many contrastive learning frameworks, such as SimCLR [37], SimSiam [38], and BYOL [39], have been implemented for HAR with sensor data, which also applied various data transformations to acquire latent representation. The contrastive learning task is to identify whether the transformed signals originated from the same signal, which aims to learn a transformation invariant representation. However, it is not guaranteed that we can obtain representation that is useful for complex work activity recognition. Haresamudram et al. [40] proposed an end-to-end deep learning method using an autoencoder structure. This study employs a pretext task to reconstruct the original time-series from a compressed feature vector. As mentioned in the introduction, Haresamudram et al. [11] processed input sensor data by randomly masking short segments within the input. The masked data was then fed into convolutional and self-attention layers to reconstruct the original unmasked sensor data. However, since the above methods did not consider the complexity of work activities in the industrial domains, it is difficult to apply them to the work activities, which is demonstrated in the evaluation section. Thus, we propose an SSL method specifically for complex work activities in this study.

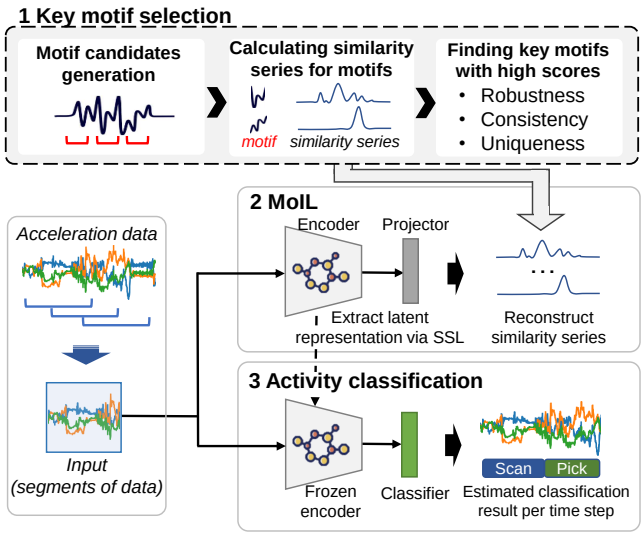


Fig. 2. Overview of the proposed framework based on MoIL.

3 SSL by Motif Identification Learning (MoIL)

Figure 2 presents the overview of the proposed approach, which is composed of three main processes: (1) key motif selection, (2) MoIL for the pretext task, and (3) activity classification in the downstream task. Firstly, we select key motifs from unlabeled data and calculate the similarity series of every period of sensor data for each key motif. Thereafter, an SSL model is pretrained by reconstructing the similarity series from unlabeled acceleration data. Finally, we apply the trained encoder in the SSL model to the downstream task for activity classification (i.e., output the estimate of activity class for each time step). In the following sections, we will explain the proposed method in detail.

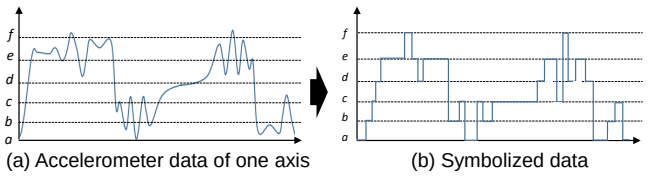
3.1 Preliminary

We collect labeled/unlabeled acceleration data from workers using a body-worn accelerometer in advance. The labeled/unlabeled data consists of a set of sensor data sequences of multiple work periods. Here, a period of data is represented as $\boldsymbol{X} = [\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_T]$, where T represents the length of the period and $\boldsymbol{x}_t$ represents the sensor values at time step t ($t \in [1, T]$). The sets of sensor data sequences for the unlabeled, labeled, and all work periods are denoted as $\boldsymbol{D}_u$, $\boldsymbol{D}_l$, and $\boldsymbol{D}_a$, respectively ($\boldsymbol{D}_a = \boldsymbol{D}_u \cup \boldsymbol{D}_l$). For every labeled period, the corresponding sequence of the activity class labels is denoted as $\boldsymbol{Y} = [\boldsymbol{y}_1, \boldsymbol{y}_2, ..., \boldsymbol{y}_T]$.

3.2 Key Motif Selection

3.2.1 Preprocessing

We preprocess the acceleration data in two steps: (1) min-max normalization, which addresses the positional discrepancies of the IMU sensor worn on the wrist, and then (2)



(a) Accelerometer data of one axis (b) Symbolized data

Fig. 3. Calculate symbolized data for one axis of acceleration data.

symbolization to efficiently find key motifs. For symbolization, we convert each numerical acceleration value into a symbol based on multiple breakpoints based on [41]. As shown in Figure 3, the acceleration values belonging to the same range are converted to the same symbols. For instance, all the acceleration values involved in the range between `a` and `b` are symbolized to `a`. Such preprocessing maintains the waveform of the acceleration data while reducing the variety of values, which could be utilized to compare action similarities efficiently. In this study, the intervals between symbols are set to be equal in the dataset.

3.2.2 Motif Candidates Generation

After symbolizing the raw sensor data, we generate a set of motif candidates $\boldsymbol{M}$ from an *initial period* $\boldsymbol{X}$ ($\boldsymbol{X}$ is randomly selected from $\boldsymbol{D}_u$, here we directly selected the first period). We use a sliding window with a fixed step across the symbolized data to extract data segments as candidate motifs. For instance, the shape of a candidate motif is (45, 3) if the size of the sliding window is 45 and the symbolized data is three-dimensional. A number of candidates of different lengths are generated using different window sizes.

3.2.3 Calculating Similarity Series

For each candidate motif m ($m \in \boldsymbol{M}$), we calculate the similarity series of m over all unlabeled periods $\boldsymbol{D}_u$, which represents the similarity of m to the sensor data segment at each timestep. Let $\boldsymbol{X}^i$ be the i-th period of data in $\boldsymbol{D}_u$ and $\boldsymbol{S}^i_m$ be the similarity series of period $\boldsymbol{X}^i$ for m. $\boldsymbol{S}^i_m$ is formulated as:

$$\boldsymbol{S}^i_m = [-d(m, g(\boldsymbol{X}^i)[1:1+|m|]), -d(m, g(\boldsymbol{X}^i)[2:2+|m|]) \\ , ..., -d(m, g(\boldsymbol{X}^i)[|\boldsymbol{X}^i|-|m|:|\boldsymbol{X}^i|])]' \quad (1)$$

where, $g(\cdot)$ denote the symbolization function (Introduced in Section 3.2.1), $|m|$ and $|\boldsymbol{X}^i|$ correspond to the length of m and $\boldsymbol{X}^i$, respectively. $d(,)$ represents a distance metric between two segments. Inspired by the Hamming distance [42], we compute the distance between two symbolized segments by counting the number of positions at which the corresponding symbols are dis-similar, enabling us to reveal the similarity of actions at each time step effectively. For example, let one axis of m be labeled as "`abc`", and let the corresponding axis of the symbolized period $g(\boldsymbol{X}^i)$ be "`bcdda`...". To calculate the similarity series, we use the sliding window with a window size equal to $|m|$ and a step length equal to 1 to generate segments of symbols from the period data. For two symbols, when the symbol distance is greater than a threshold th_d, the two symbols are dis-similar. When $th_d = 1$, the number of dis-similar symbols between "`abc`" and "`bcd`" is 0, that between "`abc`" and "`cdd`" is 2, and that between "`abc`" and "`dda`" is 3, resulting in the distance series "0,$\frac{2}{3}$,1,...". For the preceding $|m|$ symbols, we assigned the distance of the segments of symbols to the start time of the segment. For the final $|m|$ symbols, we used final obtained distance to pad the last $|m|$ distance series to ensure that the similarity series, ensuring that the similarity series and the raw sensor data have the same length. Note that we first calculate the distance series for each axis of acceleration data. Then, we sum the values at the same time step in distance series over all the axes to merge these distance series into a one-dimensional distance series. Finally, we negate the values in the distance series and use min-max normalization to map the values to [0, 1] for obtaining the similarity series.

Algorithm 1: `Finding Key Motifs`

Input: n, n': number of segments; p: number of periods; $\boldsymbol{S}$: similarity series; $\boldsymbol{M}$: all candidate motifs; $\boldsymbol{M}_R$: reference motif set; $\boldsymbol{M}_C$: complementary motif set.

$\boldsymbol{M}_R \leftarrow \emptyset, \boldsymbol{M}_C \leftarrow \emptyset$;
/* Selecting reference motifs */
for $k \leftarrow 1$ *to* n **do**
  $\boldsymbol{M}_k \leftarrow \emptyset$;
  for $\forall m \in \boldsymbol{M}$ **do**
    /* If the candidate motif locates at the k-th segment */
    if $m.occur \in [\frac{(k-1)|\boldsymbol{X}^{init}|}{n} : \frac{k|\boldsymbol{X}^{init}|}{n}]$ **then**
      d $\leftarrow m.occur - M_R^{k-1}.occur$;
      Calculate w'_k with d using the penalty function Eq. 3;
      $\boldsymbol{S}^k_{rob} \leftarrow 0$;
      /* Calculate reference score for every period */
      for $c \leftarrow 1$ *to* p **do**
        $\boldsymbol{S}^k_{rob} = \boldsymbol{S}^k_{rob} + w'_k \cdot w^c_k \cdot \max(\boldsymbol{S}^m_c)$;
      $\boldsymbol{S}^k_{rob} = \frac{\boldsymbol{S}^k_{rob}}{p}$;
      $m.score \leftarrow S^k_{rob}$;
      $\boldsymbol{M}_k \leftarrow \boldsymbol{M}_k \cup \{m\}$;
  $\boldsymbol{M}_R \leftarrow \boldsymbol{M}_R \cup \{max(\boldsymbol{M}_k.score)\}$ /* Add motif with the best score in $\boldsymbol{M}_k$ to $\boldsymbol{M}_R$ */
/* Selecting complementary motifs */
for $k \leftarrow 1$ *to* n' **do**
  $\boldsymbol{M}_k \leftarrow \emptyset$;
  for $\forall m \in \boldsymbol{M}$ **do**
    if $m.occur \in [\frac{(k-1)|\boldsymbol{X}^{init}|}{n'} : \frac{k|\boldsymbol{X}^{init}|}{n'}]$ **then**
      /* Calculate the relative time range H */
      for $j \leftarrow 1$ *to* n **do**
        if $m.occur \in [\frac{(j-1)|\boldsymbol{X}^{init}|}{n} : \frac{j|\boldsymbol{X}^{init}|}{n}]$ **then**
          $t_{m_R^{j-1}} \leftarrow \frac{\boldsymbol{M}_R^{j-1}.occur|\boldsymbol{X}^j|}{|\boldsymbol{X}^{init}|}$;
          $t_{m_R^{j}} \leftarrow \frac{\boldsymbol{M}_R^{j}.occur|\boldsymbol{X}^j|}{|\boldsymbol{X}^{init}|}$;
          $H_i = [t_{m_R^{j-1}} : t_{m_R^{j-1}}]$;
          $m.score \leftarrow S^k_{con}(m, H_i) + S^k_{uni}(m, H_i)$;
          $\boldsymbol{M}_k \leftarrow \boldsymbol{M}_k \cup \{m\}$;
  $\boldsymbol{M}_C \leftarrow \boldsymbol{M}_C \cup \{max(\boldsymbol{M}_k.score)\}$;

Output: The motifs obtained in $\boldsymbol{M}_R \cup \boldsymbol{M}_C$.

3.2.4 Finding Key Motifs with High Scores

To select good motifs, we calculate a score for each motif candidate m using p periods of similarity series (m is associated with p similarity series). Using the similarity series of each period, we calculate a score of the period for m. The final score for each motif candidate is calculated as the average score across the p periods. In this study, we select two types of key motifs: (i) reference motif and (ii) complementary motif, among candidate motifs to roughly determine the relative position of operations within a pe-

riod. The procedure for selecting motifs using a similarity series of p periods is described below; also, Algorithm 1 shows the algorithm for finding key motifs.

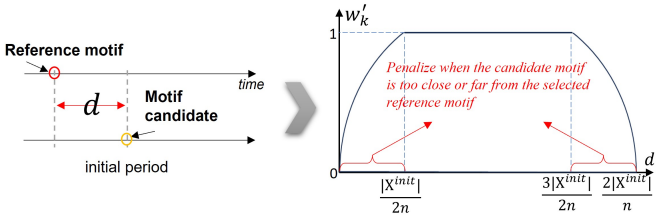


Fig. 4. The penalty function to calculate w'_k.

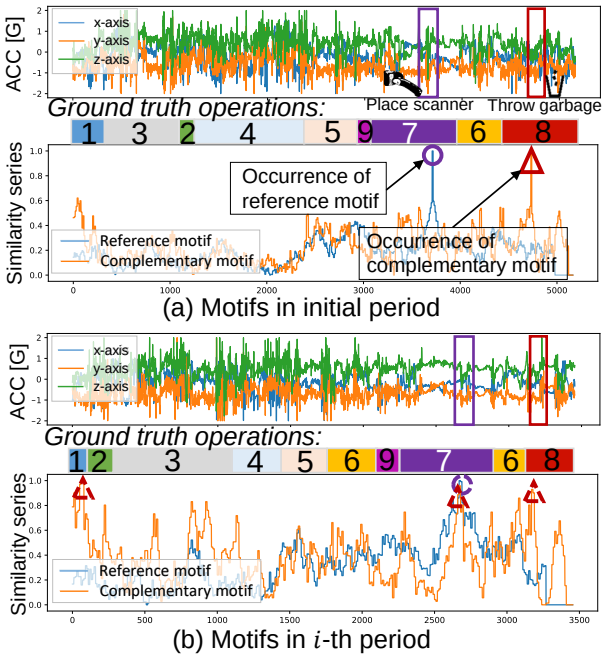


Fig. 5. Example of motifs, data is collected from worker L. Motif identified in (a) and similar actions (peaks) found in (b).

(i) Reference motif: A good reference motif candidate corresponds to a characteristic action representative of an operation in every period, such as the "pick-up box" in Figure 1. Employing the motif allows for a rough identification of the corresponding operation occurrence in a period. However, because operation labels are unavailable, we select a motif that robustly occurs at similar relative timings in unlabeled periods as a reference motif (e.g., the motif corresponding to the "pick-up box" also occurs in about one-third of the other periods). Note that, due to the highly variable sensor data patterns in different periods (as Figure 5 shows), a specific action does not always occur at the same relative timing in all the periods. Therefore, we calculate a **robust score** for each motif described below and select reference motifs with the corresponding actions that most frequently occur at similar timings in the periods.

In this method, we evenly divide the initial period into n segments. For each segment, we select a reference motif among all the candidate motifs located in the segment based on the highest **robust score** S^k_{rob} described below ($k \in [1, 2, ..., n]$), which considers if a candidate motif m occurs at similar timing in different periods and if m keeps the stable waveforms in the different periods.

$$S^k_{rob} = w'_k \frac{\sum_{c=1}^{p}(w^c_k \cdot \max(\boldsymbol{S}^m_c))}{p}. \tag{2}$$

Here, $\max(\boldsymbol{S}^m_c)$ describes the highest similarity value in the c-th period with the candidate motif m, and w^c_k (0 or 1) in Eq. 2 represents whether $\text{argmax}(\max(\boldsymbol{S}^m_c))$ locates within a relative time range corresponding to the candidate motif located in the initial period, i.e., if the candidate occurs in the c-th period at similar timing to the initial period.

However, when two reference motifs located at neighbor segments are too close in time, information obtained from the motifs will be similar. Therefore, we introduce w'_k into Eq. 2 to penalize a candidate motif that is either too close to or too far from an already-selected reference motif in the previous segment. The calculation of w'_k described below relies on $\sin(x)$, where $x \in [0, \pi]$, which is shown in Figure 4. This selection guarantees a substantial decrease in the value as the time difference between the two reference motifs approaches 0 or possible maximum distance, i.e., $\frac{2|\boldsymbol{X}^{init}|}{n}$.

$$w'_k = \begin{cases} 1 & k = 1 \text{ or } \frac{|\boldsymbol{X}^{init}|}{2n} < d < \frac{3|\boldsymbol{X}^{init}|}{2n} \\ \frac{\sin\frac{n\pi d}{2|\boldsymbol{X}^{init}|}}{\sin\frac{\pi}{4}} & d \le \frac{|\boldsymbol{X}^{init}|}{2n} \text{ or } d \ge \frac{3|\boldsymbol{X}^{init}|}{2n} \end{cases}, \tag{3}$$

where d represents the absolute time difference between the occurrence time of the already-selected reference motif in the previous segment and the candidate motif in the current segment (refer to line 8 in Algorithm 1), when d is smaller than $\frac{|\boldsymbol{X}^{init}|}{2n}$ or larger than $\frac{3|\boldsymbol{X}^{init}|}{2n}$, w'_k begins to decrease. Here, $w'_k = 1$ in the first segment because no reference motif has been selected, meaning that only the original robust score S^1_{rob} is calculated for the candidate motifs occurred in the first segment. We label the best reference motif in the k-th segment as m^k_R with the highest robust score S^k_{rob}.

(ii) Complementary motif: Because high-quality reference motifs are sparse in time, complementary motifs, which are regarded as lower-quality motifs compared to reference motifs, are introduced to support the identification of operations by reference motifs. A good complementary motif refers to an action that occurs associated with a reference motif. An example of a complementary motif is illustrated in Figure 5, which corresponds to a throwing garbage action, which occurs not only in operation 8 but other operations in the i-th period, however with the help of the reference motif occurring at operation 7, the occurrence of the subsequent operation 8 can be roughly identified. Even if the complementary motifs are of lower quality, they need to be as consistent and unique as possible in order to distinguish between different operations. Therefore, we employ the **consistency score** and **uniqueness score** to

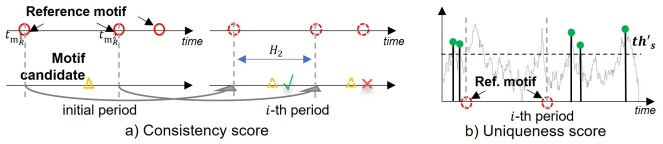


Fig. 6. Explanation of calculating consistency and uniqueness scores.

select the complementary motif, shown in Figure 6. Similar to the reference motif selection, we evenly divide the initial period into n' segments ($n' > n$) and then select the best complementary motif from the motif candidates in each segment, based on the highest scores.

As for the consistency score, since the order of operations is correlated in time, the relative location of a good complementary motif to the reference motifs should be consistent. Therefore, the consistency score is calculated to identify if the candidate motif consistently occurs between occurrences of adjacent reference motifs in the i-th period. Before calculating the score, we divide the initial period into $n+1$ ranges based on the occurrence of the reference motifs, where the occurrence time of the j-th reference motif is denoted as $t_{m_R^j}$. For a time range $[t_{m_R^{j-1}}, t_{m_R^j}]$ within which the candidate motif occurs in the initial period, we define a relative time range corresponding to $[t_{m_R^{j-1}}, t_{m_R^j}]$ as H_i in the i-th period. By using H_i, the consistency score is formulated as:

$$S_{con}^k = \frac{\sum_{i=1}^{p} P_{H_i}}{p}, \tag{4}$$

where P_{H_i} (P_{H_i} = 0 or 1) indicates whether there is a candidate motif occurs (the peak value of the similarity series $\boldsymbol{S}_i^m$ is greater than a threshold th_s=0.8) within H_i in the i-th period. This method calculates the consistency score using a coarse-grained operation order, making it applicable to minor variations of operation orders.

Regarding the uniqueness score, the action associated with the candidate motif is desired to occur as infrequently as possible within a period to better represent the occurrence of a specific operation. The uniqueness score of a candidate motif for p periods is calculated as follows:

$$S_{uni}^k = \sum_{i=1}^{p} \frac{1}{|\bar{\boldsymbol{P}}_{H_i}|}, \tag{5}$$

where $\bar{\boldsymbol{P}}_{H_i}$ is the set of peaks (peak value exceeds a threshold th_s') occurring outside the range H_i in the i-th period.

We calculate the weighted sum of the consistency and uniqueness scores for every candidate and select the highest overall scores among the candidates located in the same segments as the complementary motif. Eventually, n reference motifs and n' complementary motifs will be selected according to the above score criteria. In Section 4.4, we will evaluate the selection of motifs in detail.

3.3 MoIL

After obtaining key motifs, we introduce the procedure of MoIL, which pretrains a feature extractor (encoder) for complex activity recognition via SSL. The trained feature extractor, which learns to reconstruct similarity series, thus obtains features related to the selected key motifs, which are helpful for the downstream task. Figure 7 displays an overview of MoIL.

3.3.1 Preprocessing

Similar to the procedure in Section 3.2.1, we normalize the acceleration data. A sliding window is employed to generate sensor data segments and the corresponding similarity series of the selected key motifs, which are displayed as the pink rectangles in Figure 7.

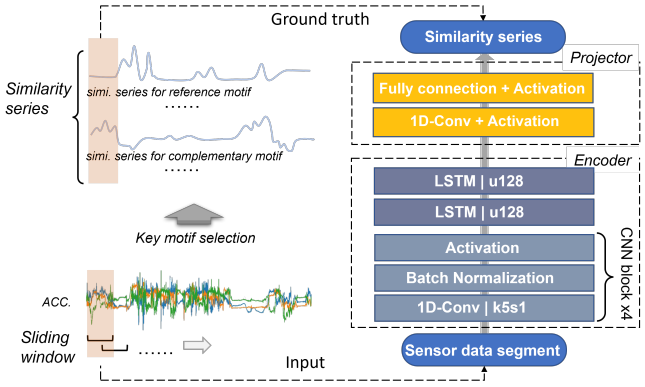


Fig. 7. Overview of MoIL.

3.3.2 SSL Model

The SSL model (MoIL) is composed of an encoder and a projector in Figure 7. The input is a sensor data segment, and the ground truth is the corresponding similarity series segment. Note that since we have $n+n'$ key motifs, the model outputs an $(n+n')$-dimensional similarity segment.

In this work, the encoder consists of four CNN blocks and two RNN blocks. Each CNN block has a 1D convolutional layer containing 64 kernels with a kernel size of 5 and a stride of 1, a Batch Normalization layer, and the ReLU activation function. The rationale for using the convolutional layers is that the information on the local temporal dependencies of the sensor data can be extracted through the layers. The output of the i-th block is fed into the $i+1$-th block, which is described as follows:

$$\mathbf{X}_u^{B(i+1)} = \text{Sigmoid}(\text{BatchNorm}(\text{1DConv}(\mathbf{X}_u^{B(i)}))), \tag{6}$$

where $\mathbf{X}_u^{B(i)}$ is the input of the i-th CNN block ($i = 1, 2, 3, 4$); $\text{1DConv}(\cdot)$ denotes the 1D convolutional layer; $\text{BatchNorm}(\cdot)$ and $\text{Sigmoid}(\cdot)$ denote the Batch Normalization and the Sigmoid activation function, respectively. Here, the Sigmoid activation function is selected for the feature extractor because it has an S-shaped curve that is smooth and bounded (ranging from 0 to 1), effectively mapping input data to probabilities [43]. This ensures that the extractor captures relevant information related to the key action .

To extract long temporal dependencies of the sensor data, two RNN blocks are used. Each RNN block has a bidirectional LSTM layer with 128 units. The output of the last CNN block is fed into the first RNN block, and the output of the first RNN block is fed into the second RNN block, which is described as follows.

$$\mathbf{X}_u^{B(i+1)} = \text{LSTM}(\mathbf{X}_u^{B(i)}), \tag{7}$$

where $\text{LSTM}(\cdot)$ denotes the RNN block ($i = 5, 6$).

The projector adjusts the output shape of the encoder module to be suitable for reconstructing similarity series. Herein, the projector is composed of a 1D convolutional layer with the same settings as the mentioned 1DConv(·); a ReLU activation function; a fully connected layer with the

dimension of the output features of the layer equal to the number (dimensions) of similarity series for the key motifs; a ReLU activation function which makes the model training more efficient. The structure is described as follows:

$$\mathbf{X}_u^{B(i+2)} = \text{ReLU}(\text{1DConv}(\mathbf{X}_u^{B(i+1)})), \tag{8}$$

$$\hat{\boldsymbol{S}}_N = \text{ReLU}(\text{Linear}(\mathbf{X}_u^{B(i+2)})), \tag{9}$$

where $\text{ReLU}(\cdot)$ denotes the ReLU activation function, $\text{Linear}(\cdot)$ denotes the fully connected layer and $\hat{\boldsymbol{S}}_N$ is an estimates of a N-dimensional similarity series segment ($N = n + n'$).

3.3.3 Network Training

We employ the mean squared error (MSE) as the loss function for the SSL task in MoIL, which is shown as follows:

$$L_{ssl} = \frac{1}{N}\sum_{j=1}^{N}\frac{1}{l}\sum_{i=t}^{t+l}(\hat{s}_{i,j} - s_{i,j})^2, \tag{10}$$

where $\hat{s}_{i,j}$ and $s_{i,j}$ are the i-th prediction and ground truth values in the similarity series of the j-th key motif, respectively. l is the segment length. Model parameters are optimized via the Adam optimizer [44].

3.4 Activity Recognition

Here, we introduce how the encoder (feature extractor) trained by MoIL can be applied to the downstream task, which is predicting the activity class at each time step. The neural network of the downstream task consists of an encoder directly copied from the SSL model and a classifier. The classifier is an MLP module consisting of three linear layers of 256, 128, and C units, where C is the number of activity classes. A Batch Normalization and activation function are applied consecutively between each layer. During the model training, we freeze the learned weights of the encoder module from MoIL and optimize the parameters of only the classifier. Given a window of sensor data $\boldsymbol{X}^{[t:t+l]}$ as the input and $\boldsymbol{Y}^{[t:t+l]} = [\boldsymbol{y}_t, \boldsymbol{y}_{t+1}, ..., \boldsymbol{y}_{t+l}]$ as the ground truth label, the classifier is trained to output estimates that have minimum errors to $\boldsymbol{Y}^{[t:t+l]}$. Note that in this study, we recognize the class label for each time step in order to clearly identify the boundary of operations. We train the downstream model using cross-entropy loss using the Adam optimizer. The objective function is formulated as follows:

$$L_{ar} = -\sum_{i=t}^{t+l}\sum_{c=1}^{C}\boldsymbol{y}_{ic}\log(p_{ic}), \tag{11}$$

where $\boldsymbol{y}_{ic}$ is a one-hot vector corresponding to the i-th prediction of class c, and p_{ic} represents the prediction of x_l^i belonging to class c. The idea of training the classifier only is to compare the performance of feature extraction via MoIL against other state-of-the-art SSL frameworks when using limited data annotations.

4 Evaluation

4.1 Datasets

We select four datasets containing the most common tasks in logistics and factories, which continue to hold widespread significance in industries. We provide an overview of the four datasets in Table 1, and describe the characteristics of every dataset as follows.

TABLE 1
Overview of the datasets
("AVG" refers to "average"; "SD" refers to "standard deviation")

Dataset	Worker	Period number	Operation number	AVG period duration (s)	SD of duration
OpenPack	A	20	11	107.88	28.06
	B	20	11	152.17	37.84
	C	20	11	121.01	24.84
	D	20	11	148.79	45.27
	E	20	11	124.39	26.66
	F	20	11	151.36	47.08
	G	20	11	122.93	31.91
	H	20	11	119.17	31.76
	I	20	11	153.61	34.78
	J	20	11	93.44	18.31
Logi	K	48	8	107.00	108.29
	L	86	9	72.77	47.35
TestBoard	M	12	8	127.42	5.01
Skoda	N	59	11	80.73	3.09

4.1.1 OpenPack Dataset [45]

The workers were asked to repeatedly perform a packaging task several times by following the work instruction document (Scenario 1, all 10 workers). Acceleration data from both wrists were collected using Empatica E4 wristband with a sampling rate of 30Hz.

4.1.2 Logi Dataset

This private dataset contains workers performing packaging tasks in a real logistics center. Data is collected using a smartwatch (Sony SmartWatch3 SWR50) worn on the workers' dominant hands, which has a sampling rate of 30 Hz. Unlike the OpenPack dataset, the order and number of operations in the Logi dataset vary for each period.

4.1.3 TestBoard Dataset

This private dataset contains a worker performing testing circuit board tasks in a factory. Data is collected using a smartwatch (Sony SmartWatch3 SWR50) worn on the worker's dominant hand with a sampling rate of 30 Hz.

4.1.4 Skoda Dataset [46]

This dataset contains a single worker working on an assembly line in car manufacturing. We used the acceleration data of the worker's wrists and downsampled the data to 30 Hz. The dataset splits every work period into time-series sensor data corresponding to individual activity instances (i.e., operation). Because data of different activities (i.e., operations) in the dataset were stored in separate files, in this work, we concatenated the activities to form periods of data based on their timestamps following the approach used in [7].

4.2 Experimental Settings & Evaluation Metrics

We utilized the normalized acceleration data described in Section 3.3.1. A sliding window was applied to generate segments of time-series data, which were inputs for the pretext and downstream tasks. The window size was set to 6 seconds (180 data points). For the training set, the sliding window step was 3 seconds, creating an overlap of 50% with

TABLE 2
Experimental parameters.

Motif selection	n=3, n'=10, th_d=1, th_s=0.8, th'_s=0.8, p=$\lvert \boldsymbol{D}_a \rvert$, Motif lengths={45, 90, 180, 270}
Model training	Batch size: 1000, L2 regression: 1e-4, SSL learning rate: 1e-4, Classifier learning rate: 1e-3, SSL pretraining #.epochs: 1000, Downstream training #.epochs: 50

the original data. For the test set, the step of the sliding window was the same length as the window size. Note that, to maintain a fair comparison, all the experimental settings for training and downstream evaluation as well as the structure of classifier (refer to Section 3.4) are consistent in this study.

The activity recognition accuracy was calculated as the accuracy, macro F1-measure, and weighted macro F1-measure of all the time steps (i.e., data points) in the test set. We executed five runs for each experiment, altering the random seed each time to randomly select periods to get the training and test sets, and the standard deviation ($\pm$) was calculated according to the results of the five experiments. The hyper-parameters used in this study are displayed in Table 2.

4.3 Baselines

In this work, we evaluated MoIL against several state-of-the-art baseline SSL methods, employing the "pretrain-then-finetune" approach to assess the efficacy of the representations learned through SSL. Among the baselines, Multi-task self-supervision, SimCLR, BYOL, CPC, and TS-TCC utilize data augmentation strategies, whereas Autoencoder and Masked reconstruction rely on data reconstruction techniques.

Multi-task self-supervision [35]. This is a multi-task SSL approach that applies eight signal transformations and assigns a binary classification task for each transformation, i.e., transformed or not. We re-implemented this network based on the paper [35] with the number of units and kernel size the same as used in that paper.

SimCLR [37]. This is one of the state-of-the-art models of contrastive learning for SSL. It identifies different augmented views of the same input signal from the other signals. In this experiment, we selected the "time warped" and "permuted" augmentations described in [35] for the best performance.

BYOL [39]. This approach comprises two autoencoders called online and target networks. Each network is trained with different augmentations of the same input signal, where the augmentations have the same settings as SimCLR. BYOL learns representation by minimizing the MSE loss of both networks.

CPC [47]. CPC employs an LSTM network designed to predict subsequent time-series segments. This method facilitates the learning of latent representations for these segments by leveraging InfoNCE loss [48] with various transformations.

TS-TCC [49]. This method employs temporal and contextual contrasts to learn time-series representations using weak and strong augmentations.

Autoencoder [40]. This task contains an "encoder-decoder" structure, which aims to reconstruct the original signal using the MSE loss.

Masked reconstruction [11]. This task is similar to the BERT model, which learns latent representation by reconstructing the masked input signals.

TABLE 3
Accuracy, macro F1-measure, and weighted macro F1-measure (%) of the SSL methods for every dataset when employing leave-one-worker-out experiments. ("$\pm$" represents the standard deviation. Results with the best performance are shown in bold.)

Approach	Accuracy	Macro F1	Weighted macro F1
OpenPack dataset			
Multi-task	41.16±5.76	36.25±4.97	41.16±5.87
SimCLR	39.47±5.44	36.20±5.60	39.70±5.31
BYOL	40.62±5.90	36.70±4.80	41.00±5.52
CPC	42.34±7.51	36.68±6.21	43.03±7.49
TS-TCC	40.86±9.44	37.02±8.97	41.27±9.50
Auto.	47.82±6.13	**42.83±5.55**	47.96±6.19
Masked rec.	42.16±8.43	37.14±8.23	42.15±8.33
DeepConvLSTM	46.67±6.27	41.78±5.58	47.25±6.03
MoIL	**48.22±6.44**	42.81±5.79	**48.22±7.67**
Logi dataset			
Multi-task	18.51±1.47	14.68±0.74	17.89±0.66
SimCLR	18.47±2.49	15.30±3.07	17.90±3.17
BYOL	18.64±2.14	15.55±2.82	18.06±2.40
CPC	19.83±1.53	17.26±0.80	19.52±2.01
TS-TCC	19.02±0.89	15.10±1.18	18.84±0.98
Autoencoder	20.59±1.10	16.36±2.42	18.93±1.51
Masked rec.	15.13±1.21	12.86±0.62	14.68±1.40
DeepConvLSTM	**21.35±2.11**	**17.54±2.01**	20.56±2.58
MoIL	20.87±0.57	17.27±1.52	**20.76±1.59**

We also compared MoIL with the supervised baseline **DeepConvLSTM** that shares the same encoder structure as MoIL, which allows for a direct comparison of performance while controlling for architectural differences.

4.4 Results

4.4.1 User-independent Model Performance

We conducted the user-independent experiment to investigate whether the latent representation learned from other workers is helpful for a new worker. This experiment provides an intuitive feeling about how significant individual differences are even when performing identical tasks, and illustrate the importance of collecting every worker's label data.

Setup: For each dataset, we assessed the performance of MoIL based on the SSL assessment paper [14]. We conducted the leave-one-worker-out experiment by randomly selecting a worker from the dataset as the test set while utilizing the rest of the workers as the training set. Initially, the training set is employed for pre-training. Following the

pre-training phase, we freeze the model and then fine-tune it for the downstream task using the training set's labels. The model's performance is assessed using the test set. In addition, we applied a pure supervised model, DeepConvLSTM, as a baseline with the same network structure as MoIL to estimate the classification performance of the network. The parameters of DeepConvLSTM were randomly initialized and trained from scratch.

Explanation: Table 3 shows the activity recognition results for each SSL method. As for the F1-measures, because both datasets have more than 8 types of activities and the amount of activities is severely imbalanced, the results in terms of macro F1-measure are relatively lower than the other two metrics. In detail, the data reconstruction-based methods, including Autoencoder, Masked reconstruction, and MoIL exhibited superior performance compared to approaches reliant on data augmentations. This disparity may stem from the fact that general data augmentations might not be as efficient in guiding the model to learn meaningful features for complex activities during the pretraining phase. Additionally, while MoIL demonstrates improved accuracy on two datasets compared to most SSL baseline methods and has performance relatively equal to that of the pure supervised method, its superiority is not markedly obvious. This may be because work activities can vary significantly from person to person, even when performing identical tasks, indicating that a user-dependent model is needed for complex work activity recognition.

It is also important to analyze the statistical significance of the difference between the performance of the SSL methods. Previous metrics, such as t-tests, operate under the assumption that each experiment is independent and their confidence levels are not the probability that a method is better than another, but only the probability of getting the observed differences assuming the methods are equal. In this work, we will use their Bayesian approach to analyze our results [50]. We use a region of practical equivalence (rope) of 0.5% in weighted macro F1-measure, that is, we want to know the probability that the difference between the model's performance is within 0.5% (p(rope)), less than that (p(left)) or more (p(right)). According to Figure 8, we compared the performance of MoIL to the best-performing SSL baselines on OpenPack and Logi datasets, respectively. We observed that MoIL has a 35.32% probability of being at least 0.5% better than Autoencoder and a 43.82% probability of similar performance, indicating that MoIL has likely a comparable performance to Autoencoder. On the other hand, MoIL has a 90.76% probability of being at least 0.5% better than CPC, indicating that MoIL outperforms the other baselines when the time series is very complicated.

4.4.2 User-dependent Model Performance

We conducted a user-dependent experiment to investigate the performance of SSL methods when using a small amount of labeled data of individual workers.

Setup: For each worker, we randomly select 80% of periods for all the periods into the training set and the remaining 20% of periods into the test set. The pre-training, fine-tuning, and test phases are the same as the setting in the user-independent experiment.

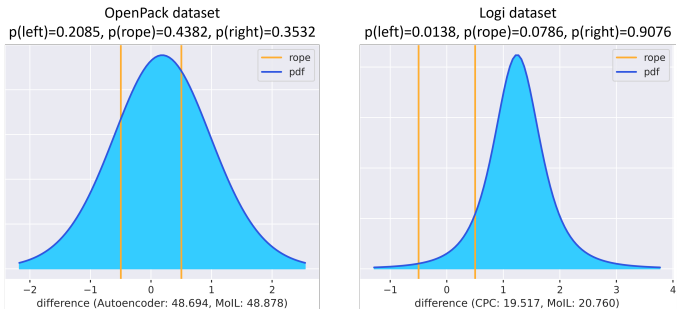


Fig. 8. Posterior of the Bayesian correlated t-test for the difference between the weighted macro F1 obtained by SSL baselines and MoIL.

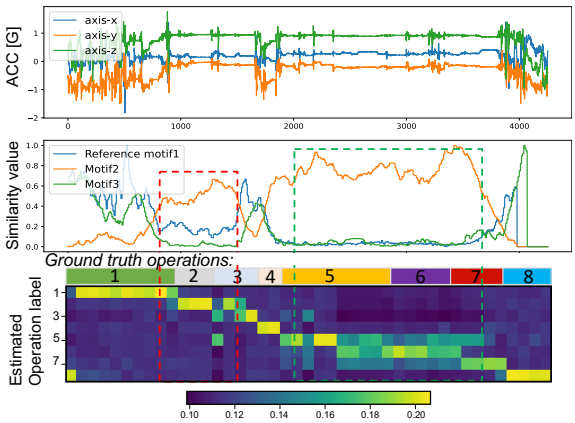


Fig. 9. An example period of worker M (TestBoard) by MoIL.

Explanation: Table 4 shows the average performance of every worker in each dataset. MoIL outperformed the other SSL methods among all workers. The result indicated that MoIL can effectively learn latent representation for activity recognition using unlabeled data of the worker despite the variation of workers and tasks. The factory work tasks (TestBoard and Skoda) showed better performance than the packaging tasks. This could be because the operations performed in the factories are more consistent (low standard deviation shown in Table 1). **MoIL outperformed other methods on the OpenPack, Logi, TestBoard, and Skoda datasets, with improvements of 2.78%, 11.17%, 23.85%, and 1.60%, respectively.** Although the F1-measure on the Logi dataset was relatively low because the occurrence and duration of operations greatly differ in every period, by identifying the corresponding key actions (Figure 5), features extracted by MoIL are more effective at identifying the complex operations than the other baselines. Besides, MoIL exhibited a good performance on similar operations. As an example shown in Figure 9, MoIL demonstrated its ability to differentiate between activities in the red and green rectangles, which involve very similar and small actions, by utilizing three reference motifs. Because the similarity series associated with motifs revealed similarities between key actions and other actions that encapsulate rich information for operation recognition, pretraining in MoIL led to an improvement of over 20% compared to other baselines. Figure 10 depicts the performance of MoIL compared to Autoencoder (best SSL baseline with data reconstruction)

TABLE 4
Average accuracy, macro F1-measure, and weighted macro F1-measure (%) of the SSL methods for each worker in every dataset. ("±" represents the standard deviation. Results with the best performance are shown in bold.)

		Multi-task	SimCLR	BYOL	TS-TCC	CPC	Autoencoder	Masked rec.	MoIL
OpenPack	Accuracy	47.21±6.37	52.41±6.70	51.30±7.56	53.77±10.49	54.60±10.59	70.56±5.16	56.34±11.48	**73.27±5.58**
	Macro F1	43.86±6.74	49.46±7.03	47.82±7.51	48.64±11.77	47.63±10.72	68.99±5.87	53.81±8.76	**71.67±5.42**
	Weighted F1	46.88±6.61	51.54±8.06	51.04±7.76	51.47±11.45	51.94±11.10	70.34±5.20	56.73±8.87	**73.12±5.60**
Logi	Accuracy	35.94±4.14	39.28±2.87	37.99±4.48	45.42±3.63	39.17±6.20	41.00±2.57	40.47±5.09	**56.03±5.83**
	Macro F1	29.50±3.21	34.05±2.74	32.19±4.13	30.01±2.49	23.26±2.58	35.29±2.30	31.95±4.96	**48.83±4.58**
	Weighted F1	35.11±4.55	39.39±2.76	37.81±4.51	39.67±3.92	35.46±4.75	40.47±2.45	39.65±5.92	**55.64±5.60**
TestBoard	Accuracy	43.43±3.43	52.44±2.93	50.19±4.45	53.33±3.65	35.56±4.42	47.72±2.63	42.06±1.25	**75.07±4.39**
	Macro F1	40.11±3.21	50.02±3.27	46.97±3.94	48.65±4.62	26.10±5.20	43.85±1.11	35.67±1.94	**74.08±3.27**
	Weighted F1	41.14±3.76	50.75±3.02	48.26±4.05	50.26±3.86	30.05±5.91	45.30±1.60	37.61±1.53	**74.60±4.30**
Skoda	Accuracy	76.29±0.78	75.30±2.10	73.78±2.27	94.65±1.11	71.74±2.51	85.43±1.98	78.42±1.06	**96.34±0.78**
	Macro F1	74.81±0.90	73.71±1.66	72.26±3.47	93.63±1.45	65.05±4.69	83.48±1.90	75.25±1.46	**95.87±0.68**
	Weighted F1	75.67±0.96	74.56±2.06	73.11±2.82	94.72±1.14	69.64±2.62	85.19±2.04	78.01±1.15	**96.32±0.74**

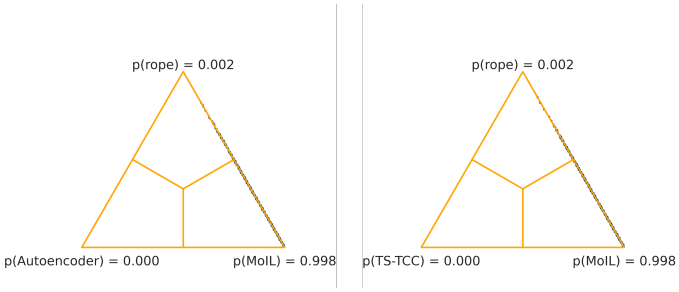


Fig. 10. Posterior of the Bayesian correlated t-test for the difference between the weighted macro F1 obtained by SSL baselines and MoIL between datasets.

and TS-TCC (best SSL baseline with data augmentation) across all datasets. We can see that MoIL outperforms Autoencoder by more than 0.5% of weighted macro F1 with a probability of 99.8%, while there is less than 0.2% that MoIL and Autoencoder perform equally. On the other hand, MoIL also outperforms TS-TCC by more than 0.5% of weighted macro F1 with a probability of 99.8%.

4.4.3 *Latent Representation Acquired by MoIL*

We conduct this experiment to further illustrate why pre-trained models using MoIL can enhance performance in downstream activity recognition tasks with a limited amount of labeled data.

Setup: We split the training and test sets of every worker following the settings of the user-dependent experiment. In the case of MoIL, we used the training set for pre-training and extracted the latent representation from the encoder's output, as illustrated in Figure 7. Conversely, we trained a supervised learning model named DeepConvLSTM from scratch on the same training set. **This model has the same architecture as the MoIL encoder but replaces the projector with a new classifier, as depicted in Figure 7, we chose this model to eliminate the effects of model structure**. We also extracted the latent representation from the output of this encoder.

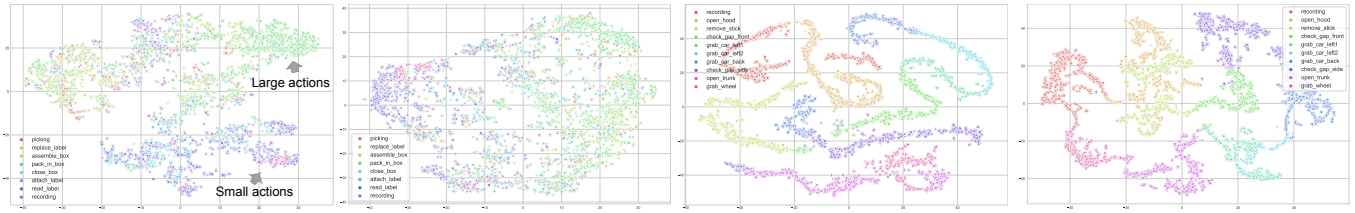
Explanation: As shown in Figure 11, we mapped the latent representation into a 2D space through t-SNE [51], which is an unsupervised technique for visualizing multi-dimensional data that maps high-dimensional data to low-dimensions. The clusters of data points in the Skoda dataset were clear for both MoIL and the supervised method, even though MoIL did not use ground truth operation labels in the training. Interestingly, the number of clusters in Figure 11 (c) did not strictly equal the number of operation classes, and some operations, such as "recording" (in red color), were divided into several clusters. This might be because at least two motifs were selected in this operation and the learned latent representation was closely associated with motifs. In contrast, clusters from the latent representation for the Logi dataset were confused, likely because actions performed during every operation were more complicated than the ones in Skoda. However, the clusters for the latent representation of worker K, which was pre-trained by MoIL, were clearer than the result of DeepConvLSTM because the similarity series contains similarities in actions, which are helpful in roughly identifying operations.

4.4.4 *Effect of Amount of Labeled Data*

This experimental setup was designed to evaluate the impact of increasing the amount of labeled training data on the performance of MoIL compared to both SSL and supervised learning baselines. By doing so, we aimed to demonstrate the potential advantages of MoIL, particularly in scenarios where labeled data is scarce or costly to obtain.

Setup: For each worker, we randomly select some periods for training and the rest periods to test. We increased the amount of labeled training data from 1 to 5 periods and performed five runs by changing the random seeds. We selected the best-performing SSL baselines that utilized distinct strategies: data augmentations (TS-TCC) and data reconstruction (Autoencoder). We also compared MoIL with the supervised baseline (DeepConvLSTM) that shares the same encoder structure as MoIL as described before.

Explanation: Figure 12 shows the average F1-measure for each dataset when training with different amounts of labeled data. On average, MoIL surpassed the SSL baselines across all datasets when the amount of labeled data available for fine-tuning the downstream tasks was limited. This outcome suggests that MoIL, through its pre-training process of reconstructing similarity series, effectively learned



(a) Worker K (Logi), MoIL (b) Worker K (Logi), DeepConvLSTM (c) Worker N (Skoda), MoIL (d) Worker N (Skoda), DeepConvLSTM

Fig. 11. t-SNE visualization of the learned representation (training process of MoIL does not use any labels).

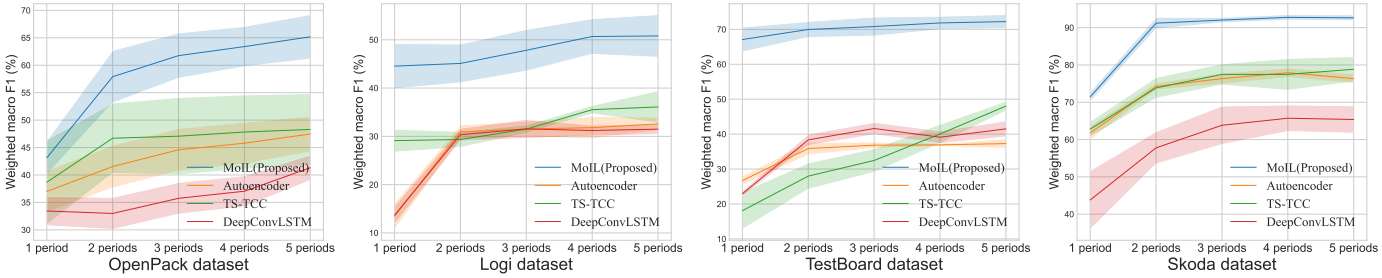


Fig. 12. Average weighted macro F1-measure of methods for each dataset.

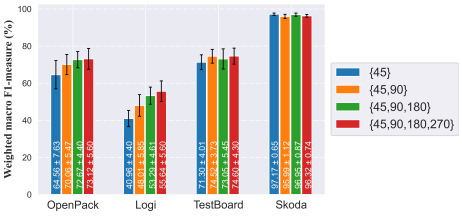


Fig. 13. The results of weighted macro F1-measure when using different window sizes to generate candidate motifs.

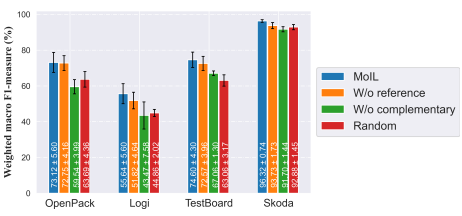


Fig. 14. The results of weighted macro F1-measure when using different motif selection criteria.

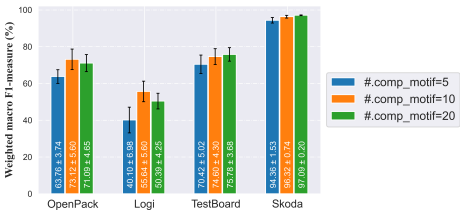


Fig. 15. The results of weighted macro F1-measure when using different numbers of complementary motifs.

worker-related features. Consequently, this made it easier to fine-tune models for downstream tasks involving the same worker. Specifically, MoIL outperformed the supervised baseline (i.e., DeepConvLSTM) on all datasets, especially for the Logi dataset, which shows relatively complex sensor data, when using one period of labels. This result highlighted that training a supervised learning model for complex activity recognition requires a sufficient number of labels. However, a pretext task (MoIL) using unlabeled data can significantly enhance activity recognition performance in downstream tasks, even when only limited labels are available. This is because MoIL learns latent representation through good motifs closely associated with the occurrence of specific operations, which is helpful for operation recognition in the downstream task. (Similar experimental settings for the OpenPack and Skoda datasets could be found in [45] of Table 9 and [7] of Table 9, respectively. MoIL outperformed the state-of-the-art supervised learning models such as LOS-Net in [7], [45] even though these methods relied on instruction documents in recognition.)

4.4.5 *Effect of Motif Sizes*

We investigate the effect of the sliding window size when generating candidate motifs, i.e., the motif length.

Setup: We select different window sizes of 45, 90, 180, and 270 data points to represent short and long-duration actions in all datasets. We follow the motif generation and scoring algorithms to get the best motifs and follow the settings of the user-dependent experiment to calculate activity recognition results.

Explanation: As shown in Figure 13, the result of generating candidate motifs using various window sizes is better than using a single window size in most of the datasets. We found that a smaller window size can generate better candidate motifs when the hand movement is not frequently changing, such as in TestBoard and Skoda datasets. However, the effect of the window size with the OpenPack and Logi datasets varies. The actions in these two datasets are more complicated, and using a small window cannot achieve the best performance. In other words, various window sizes are required to generate candidate motifs when the actions are complicated. In this study, we use 4 window sizes for all datasets to generate candidate motifs.

4.4.6 *Effect of Motif Scores*

Here, we discuss the impact of the two categories of motifs on training the latent representation in MoIL. We adopted the leave-one-method-out strategy to measure the effectiveness of each motif type. We follow the experiment settings of

TABLE 5
Computation time (minutes) of key motif selection for every worker in OpenPack dataset.

	A	B	C	D	E	F	G	H	I	J
Total duration	36	51	40	50	41	50	41	40	51	31
Computation time	30	42	26	39	28	49	33	30	37	16

the user-dependent model, and the detailed implementation of each method is described below.

MoIL. The proposed method.

W/o reference. The proposed method without the use of the reference motifs when training MoIL.

W/o complementary. The proposed method without the use of complementary motifs when training MoIL.

Random. The proposed method with the random selection of $n + n'$ candidate motifs among all the candidates.

Figure 14 shows the activity recognition accuracy for four datasets. Overall, MoIL achieved the best performance among all the methods, indicating the effectiveness of motif selection. The "W/o complementary" yielded poorer results than the "W/o reference", even though the quality of the complementary motif was not as high as that of the reference motif. This implied that the performance of MoIL could be affected not only by the quality of motifs but also by the number of motifs. However, in the case of the Test-Board dataset, the weighted macro F1-measure of the "W/o complementary" was higher than that of the "Random", even though the number of motifs was four times smaller. This result suggested that if high-quality reference motifs were selected, MoIL could achieve better performance with a limited number of motifs.

4.4.7 Influence of Number of Motifs

We observed that selecting high-quality reference motifs is helpful for downstream activity recognition tasks as shown in Section 4.4.6. Here, we evaluate the effect of the number of complementary motifs (relatively lower quality) selected to pretrain the SSL model in MoIL.

Setup: We select different amounts of complementary motifs in all datasets and calculate downstream activity recognition results following the settings of the user-dependent experiment.

Explanation: Figure 15 shows that selecting 5 complementary motifs to pre-train the SSL model results in the lowest performance on all datasets because only a few motifs cannot track sufficient information to identify every operation. When the number of motifs is increased to 10 and 20, the weighted macro F1-measure becomes consistent and achieves the best result in all datasets. The differences between the F1-measure for 10 and 20 motifs in all datasets are only 2.03%, 5.25%, -1.18%, and -0.77%, respectively. This is because key motifs were already densely distributed throughout a period. In other words, key motifs located in close areas provide similar activity information, which is not helpful for pretraining the SSL model.

4.4.8 Computation Time

The key motif selection process involves generating motif candidates, calculating similarity series, and identifying key motifs with high scores. Table 5 presents the computation cost of the key motif selection method for the OpenPack dataset. On average, the computation time is approximately 0.8 times the total duration of the sensor data. As the duration of the sensor data increases, the computation time for key motif selection also increases, with the most time-consuming task being the calculation of similarity series for each period. This result suggests that further optimization of the key motif selection method is needed to reduce computational costs.

TABLE 6
Comparison of MoIL with the method that applies similarity series as input to DeepConvLSTM for the OpenPack dataset in a user-dependent setting.

	Accuracy	Macro F1	Weighted macro F1
MoIL	**73.27±5.58**	**71.67±5.42**	**73.12±5.60**
DeepConvLSTM (simi. series as input)	22.76±3.81	20.28±3.00	22.87±3.61

4.4.9 Different Ways to Use Similarity Series

Following the experimental setup in Section 4.4.2, we examine whether directly using similarity series as input for the activity recognition network is sufficient. Table 6 presents the results of MoIL and a comparison method using similarity series as input, with DeepConvLSTM (the same network structure as MoIL) as the network structure. It is clear that although the similarity series contains information about key actions, it is insufficient to identify operations directly from it. This result indicates that the original sensor data still holds rich information for identifying operations. However, using the similarity series as a pseudo label for pretraining the network can help the model quickly focus on important features and reduce the number of labels required for the downstream task.

5 Discussion

In discussing the results of this study, we must consider the additional challenges that may be encountered when applying MoIL to real industrial environments.

During the data collection phase, a major issue is distribution differences caused by differences in wearable devices and the device position. This issue may potentially lead to inconsistencies in the data, which can directly affect motif selection and the calculation of similarity series. Therefore, this study assumes that data from the same individual is collected as consistently as possible through the same type of device, and that the device's wearing position remains as constant as possible. In the future, to enhance the robustness of the method, we need to design methods to deal with noise caused by different devices and wearing positions. Besides, there may be missing data due to signal interference in real industrial settings, which can be addressed by re-sending data or employing interpolation techniques in our future work.

In terms of pseudo label generation (calculating similarity series), we hope to improve computational speed by porting the algorithm to a compiled language, such as C++.

Compiled languages like C++ have long been proven to be more than 30 times faster [52] than interpreted languages, such as Python, thus speeding up the calculation of similarity series. On the other hand, we found that the similarity series generated by different initial periods varies. When the operation order and sensor data of the initial period are similar to those of other periods, the results of HAR do not differ significantly; conversely, significant differences can lead to a decline in HAR results. In the future, we will consider optimizing the selection method for the initial period to further improve the performance of HAR.

During the model training phase, computational speed and space occupation become important considerations. As the volume of data continuously increases, models become more complex, requiring us to balance the model performance and efficiency. The feature extractor in this study is based on CNN and RNN models, and we found that RNN models have more parameters and take longer to train. Therefore, we consider using more efficient model structures, such as MobileNet [53], as a feature extractor in the future.

During the model inference phase, real-time performance becomes a key consideration. In many industrial applications, such as automated production lines or real-time monitoring systems, the model needs to be able to quickly and accurately infer the current activity so that the system can respond in a timely manner. To achieve this, we can reduce the model's parameters through knowledge distillation [54] while ensuring the accuracy of the model. Meanwhile, we could also apply network quantization [55] to effectively reduce both the model size and computation cost in a resource-constrained environment.

Finally, future research could focus on improving the model's ability to predict erroneous operations. In industrial settings, predicting and preventing erroneous operations are crucial for ensuring production safety and improving efficiency. By deeply analyzing the causes of erroneous operations and how to avoid these errors through model prediction, greater value can be provided to industrial applications.

In summary, complex work activity recognition still presents a series of challenges in real industrial settings, requiring a significant amount of research to address issues in real industrial scenarios.

6 Conclusion

We presented MoIL, a new SSL approach for sensor data representation learning focusing on complex work activity recognition in the industrial domain. We first selected key motifs representing characteristic actions without using operation labels. Then, the SSL task was performed to identify the occurrence of the motifs to effectively learn useful latent representation regarding complex activity recognition. We exhaustively evaluated our approach and demonstrated that MoIL outperformed state-of-the-art SSL baselines on various industrial tasks. Finally, visualizations of extracted features explained the characteristics of the latent representation learned by MoIL. This work provides a future direction toward developing an SSL for complex time-series through data similarity that can handle self-supervised feature extraction involving the characteristics of the data.

References

[1] S. Inoue, P. Lago, T. Hossain, T. Mairittha, and N. Mairittha, "Integrating activity recognition and nursing care records: The system, deployment, and a verification study," *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.*, vol. 3, no. 3, Sep 2019.

[2] Y. Xiao, J. Zhu, S. Zhang, X. Liu, and S. Guo, "Fall-attention: An attention-based fall detection method for adjoint activities," *IEEE Transactions on Mobile Computing*, vol. 23, no. 07, pp. 7895–7909, jul 2024.

[3] L. Ray, B. Zhou, S. Suh, L. Krupp, V. Rey, and P. Lukowicz, "Text me the data: Generating ground pressure sequence from textual descriptions for har," in *2024 IEEE International Conference on Pervasive Computing and Communications Workshops and other Affiliated Events (PerCom Workshops)*. Los Alamitos, CA, USA: IEEE Computer Society, mar 2024, pp. 461–464. [Online]. Available: https://doi.ieeecomputersociety.org/10.1109/PerComWorkshops59983.2024.10503379

[4] T. Maekawa, D. Nakai, K. Ohara, and Y. Namioka, "Toward practical factory activity recognition: Unsupervised understanding of repetitive assembly work in a factory," ser. UbiComp '16. New York, NY, USA: Association for Computing Machinery, 2016, p. 1088–1099.

[5] M. Javaid, A. Haleem, R. P. Singh, S. Rab, and R. Suman, "Significance of sensors for industry 4.0: Roles, capabilities, and applications," *Sensors International*, vol. 2, p. 100110, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2666351121000310

[6] J. Morales, N. Yoshimura, Q. Xia, A. Wada, Y. Namioka, and T. Maekawa, "Acceleration-based human activity recognition of packaging tasks using motif-guided attention networks," in *Proceedings of the 2022 IEEE International Conference on Pervasive Computing and Communications (PerCom)*, 2022, pp. 1–12.

[7] N. Yoshimura, T. Maekawa, T. Hara, A. Wada, and Y. Namioka, "Acceleration-based activity recognition of repetitive works with lightweight ordered-work segmentation network," *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.*, vol. 6, no. 2, Jul 2022.

[8] Q. Xia, J. Korpela, Y. Namioka, and T. Maekawa, "Robust unsupervised factory activity recognition with body-worn accelerometer using temporal structure of multiple sensor data motifs," vol. 4, no. 3, Sep 2020.

[9] Y. Huang, K. Chen, L. Wang, Y. Dong, Q. Huang, and K. Wu, "Lili: liquor quality monitoring based on light signals," in *Proceedings of the 27th Annual International Conference on Mobile Computing and Networking*, 2021, pp. 256–268.

[10] Y. Huang, K. Chen, J. Zhao, L. Wang, and K. Wu, "Beverage deterioration monitoring based on surface tension dynamics and absorption spectrum analysis," *IEEE Transactions on Mobile Computing*, vol. 23, no. 5, pp. 3722–3740, 2023.

[11] H. Haresamudram, A. Beedu, V. Agrawal, P. L. Grady, I. Essa, J. Hoffman, and T. Plotz, "Masked reconstruction based self-supervision for human activity recognition," in *Proceedings of the 2020 International Symposium on Wearable Computers*, ser. ISWC '20. New York, NY, USA: Association for Computing Machinery, 2020, p. 45–49.

[12] G. Zerveas, S. Jayaraman, D. Patel, A. Bhamidipaty, and C. Eickhoff, "A transformer-based framework for multivariate time series representation learning," in *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, ser. KDD '21. New York, NY, USA: Association for Computing Machinery, 2021, p. 2114–2124.

[13] H. Qian, T. Tian, and C. Miao, "What makes good contrastive learning on small-scale wearable-based tasks?" ser. KDD '22. New York, NY, USA: Association for Computing Machinery, 2022, p. 3761–3771.

[14] H. Haresamudram, I. Essa, and T. Plotz, "Assessing the state of self-supervised human activity recognition using wearables," *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.*, vol. 6, no. 3, Sep 2022.

[15] C. I. Tang, I. Perez-Pozuelo, D. Spathis, S. Brage, N. Wareham, and C. Mascolo, "Selfhar: Improving human activity recognition through self-training with unlabeled data," *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.*, vol. 5, no. 1, Mar 2021.

[16] Y. Jain, C. I. Tang, C. Min, F. Kawsar, and A. Mathur, "Collossl: Collaborative self-supervised learning for human activity recognition," *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.*, vol. 6, no. 1, Mar 2022.

[17] Y. Dong, J. Liu, Y. Gao, S. Sarkar, Z. Hu, J. Fagert, S. Pan, P. Zhang, H. Y. Noh, and M. Mirshekari, "A window-based sequence-to-one approach with dynamic voting for nurse care activity recognition using acceleration-based wearable sensor," in *Adjunct Proceedings of the 2020 ACM International Joint Conference on Pervasive and Ubiquitous Computing and Proceedings of the 2020 ACM International Symposium on Wearable Computers*, ser. UbiComp-ISWC '20. New York, NY, USA: Association for Computing Machinery, 2020, p. 390–395.

[18] X. Dong, Z. Han, Y. Nishiyama, and K. Sezaki, "Detecting single-hand riding with integrated accelerometer and gyroscope of smartphone," in *Adjunct Proceedings of the 2021 ACM International Joint Conference on Pervasive and Ubiquitous Computing and Proceedings of the 2021 ACM International Symposium on Wearable Computers*, ser. UbiComp '21. New York, NY, USA: Association for Computing Machinery, 2021, p. 19–20.

[19] A. Reiss and D. Stricker, "Introducing a new benchmarked dataset for activity monitoring," in *Proceedings of the 2012 16th Annual International Symposium on Wearable Computers (ISWC)*, ser. ISWC '12. USA: IEEE Computer Society, 2012, p. 108–109.

[20] I. Mohino-Herranz, R. Gil-Pita, M. Rosa-Zurera, and F. Seoane, "Activity recognition using wearable physiological measurements: Selection of features from a comprehensive literature study," *Sensors*, vol. 19, no. 24, 2019.

[21] E. Garcia-Ceja, C. E. Galvn-Tejada, and R. Brena, "Multi-view stacking for activity recognition with sound and accelerometer data," *Information Fusion*, vol. 40, no. C, p. 45–56, Mar 2018.

[22] F. J. Ordóñez and D. Roggen, "Deep convolutional and lstm recurrent neural networks for multimodal wearable activity recognition," *Sensors*, vol. 16, no. 1, 2016.

[23] Y. Chen, Y. Gu, X. Jiang, and J. Wang, "Ocean: A new opportunistic computing model for wearable activity recognition," in *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing: Adjunct*, ser. UbiComp '16. New York, NY, USA: Association for Computing Machinery, 2016, p. 33–36.

[24] S. S. Alia and P. Lago, "Daily routine recognition from longitudinal, real-life wearable sensor data for the elderly," in *2024 International Conference on Activity and Behavior Computing (ABC)*, 2024, pp. 1–9.

[25] Q. Zhang, "Deep learning of biomechanical dynamics in mobile daily activity and fall risk monitoring," in *2019 IEEE Healthcare Innovations and Point of Care Technologies, (HI-POCT)*, 2019, pp. 21–24.

[26] N. Hernandez, J. Lundström, J. Favela, I. McChesney, and B. Arnrich, "Literature review on transfer learning for human activity recognition using mobile and wearable devices with environmental technology," *SN Computer Science*, vol. 1, no. 2, p. 66, 2020.

[27] Z. R. Tusar, M. Islam, and S. Sharmin, "Accelerometer based complex nurse care activity recognition using machine learning approach," in *Adjunct Proceedings of the 2021 ACM International Joint Conference on Pervasive and Ubiquitous Computing and Proceedings of the 2021 ACM International Symposium on Wearable Computers*, ser. UbiComp '21. New York, NY, USA: Association for Computing Machinery, 2021, p. 452–457.

[28] M. Aehnelt, E. Gutzeit, and B. Urban, "Using activity recognition for the tracking of assembly processes : Challenges and requirements," 2014.

[29] S. Feldhorst, M. Masoudenijad, M. ten Hompel, and G. A. Fink, "Motion classification for analyzing the order picking process using mobile sensors," in *Proceedings of the 5th International Conference on Pattern Recognition Applications and Methods*, ser. ICPRAM 2016. Setubal, PRT: SCITEPRESS - Science and Technology Publications, Lda, 2016, p. 706–713.

[30] F. M. Rueda, R. Grzeszick, G. A. Fink, S. Feldhorst, and M. ten Hompel, "Convolutional neural networks for human activity recognition using body-worn sensors," *Informatics*, vol. 5, p. 26, 2018.

[31] Q. Xia, A. Wada, J. Korpela, T. Maekawa, and Y. Namioka, "Unsupervised factory activity recognition with wearable sensors using process instruction information," *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.*, vol. 3, no. 2, Jun 2019.

[32] S. Gidaris, P. Singh, and N. Komodakis, "Unsupervised representation learning by predicting image rotations," in *Proceedings of the International Conference on Learning Representations*, 2018.

[33] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Minneapolis, Minnesota: Association for Computational Linguistics, jun 2019, pp. 4171–4186.

[34] Y.-A. Chung and J. Glass, "Generative pre-training for speech with autoregressive predictive coding," in *Proceedings of the ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2020, pp. 3497–3501.

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

[36] T. T. Um, F. M. J. Pfister, D. Pichler, S. Endo, M. Lang, S. Hirche, U. Fietzek, and D. Kulic, "Data augmentation of wearable sensor data for parkinson's disease monitoring using convolutional neural networks," in *Proceedings of the 19th ACM International Conference on Multimodal Interaction*, ser. ICMI '17. New York, NY, USA: Association for Computing Machinery, 2017, p. 216–220.

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

[38] X. Chen and K. He, "Exploring simple siamese representation learning," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2021, pp. 15 750–15 758.

[39] J.-B. Grill, F. Strub, F. Altche, C. Tallec, P. Richemond, E. Buchatskaya, C. Doersch, B. Avila Pires, Z. Guo, M. Gheshlaghi Azar, B. Piot, k. kavukcuoglu, R. Munos, and M. Valko, "Bootstrap your own latent - a new approach to self-supervised learning," in *Proceedings of the Advances in Neural Information Processing Systems*, H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, Eds., vol. 33. Curran Associates, Inc., 2020, pp. 21 271–21 284.

[40] H. Haresamudram, D. V. Anderson, and T. Plotz, "On the role of features in human activity recognition," in *Proceedings of the 23rd International Symposium on Wearable Computers*, ser. ISWC '19. New York, NY, USA: Association for Computing Machinery, 2019, p. 78–88.

[41] J. Lin, E. Keogh, S. Lonardi, and B. Chiu, "A symbolic representation of time series, with implications for streaming algorithms," in *Proceedings of the 8th ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery*, ser. DMKD '03. New York, NY, USA: Association for Computing Machinery, 2003, p. 2–11. [Online]. Available: https://doi.org/10.1145/882082.882086

[42] B. Waggener, W. N. Waggener, and W. M. Waggener, *Pulse code modulation techniques*. Springer Science & Business Media, 1995.

[43] C. Nwankpa, W. Ijomah, A. Gachagan, and S. Marshall, "Activation functions: Comparison of trends in practice and research for deep learning," *arXiv preprint arXiv:1811.03378*, 2018.

[44] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[45] N. Yoshimura, J. Morales, T. Maekawa, and T. Hara, "Openpack: A large-scale dataset for recognizing packaging works in iot-enabled logistic environments," *arXiv preprint arXiv:2212.11152*, 2022.

[46] P. Zappi, T. Stiefmeier, E. Farella, D. Roggen, L. Benini, and G. Troster, "Activity recognition from on-body sensors by classifier fusion: sensor scalability and robustness," in *Proceedings of the 2007 3rd international conference on intelligent sensors, sensor networks and information*. IEEE, 2007, pp. 281–286.

[47] H. Haresamudram, I. Essa, and T. Plötz, "Contrastive predictive coding for human activity recognition," *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.*, vol. 5, no. 2, jun 2021. [Online]. Available: https://doi.org/10.1145/3463506

[48] A. v. d. Oord, Y. Li, and O. Vinyals, "Representation learning with contrastive predictive coding," *arXiv preprint arXiv:1807.03748*, 2018.

[49] 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," *arXiv preprint arXiv:2106.14112*, 2021.

[50] A. Benavoli, G. Corani, J. Demšar, and M. Zaffalon, "Time for a change: a tutorial for comparing multiple classifiers through bayesian analysis," *Journal of Machine Learning Research*, vol. 18, no. 77, pp. 1–36, 2017. [Online]. Available: http://jmlr.org/papers/v18/16-305.html

[51] L. v. d. Maaten and G. Hinton, "Visualizing Data using t-SNE," *Journal of Machine Learning Research*, vol. 9, no. 86, pp. 2579–2605, 2008.

[52] F. Zehra, M. Javed, D. Khan, and M. Pasha, "Comparative analysis of c++ and python in terms of memory and time," *Preprints*, December 2020. [Online]. Available: https://doi.org/10.20944/preprints202012.0516.v1
[53] D. Sinha and M. El-Sharkawy, "Thin mobilenet: An enhanced mobilenet architecture," in *2019 IEEE 10th annual ubiquitous computing, electronics & mobile communication conference (UEMCON)*. IEEE, 2019, pp. 0280–0285.
[54] J. Gou, B. Yu, S. J. Maybank, and D. Tao, "Knowledge distillation: A survey," *International Journal of Computer Vision*, vol. 129, no. 6, pp. 1789–1819, 2021.
[55] J. Yang, X. Shen, J. Xing, X. Tian, H. Li, B. Deng, J. Huang, and X.-s. Hua, "Quantization networks," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 7308–7316.

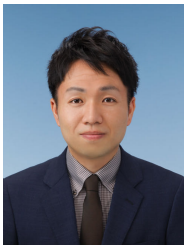
Qingxin Xia is currently a full-time researcher at the Hong Kong University of Science and Technology (Guangzhou) (HKUSTGZ) and a visiting researcher at Osaka University. She received her Ph.D. from Osaka University in 2021. Her research interests include ubiquitous computing and mobile sensing. She has published papers at top conferences of the mobile/ubiquitous computing research community, such as IMWUT, ISWC, PerCom, etc.

Jaime Morales is currently pursuing a Ph.D at Osaka University. He received a M.S. From Osaka University and his B.S. from the Monterrey Institute of Technology(Mexico). His research interests include IoT-based machine learning, industrial applications for IoT sensing, and workplace health monitoring with wearable devices. He has published in primary conferences and journals such as IEEE Percom, IEEE PMC, ACM IMWUT, etc.

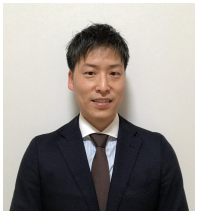
Yongzhi Huang is currently a Ph.D. student at HKUSTGZ, with a focus on advanced technologies in smart sensing, mobile computing, and IoT. His research includes next-generation health sensing, HCI, and wireless networks. He has published papers at top conferences including MobiCom, MobiSys, UbiComp, CHI, etc. He is also a prolific inventor with 12 pending patents in China, reflecting his contributions to the field of smart technologies and IoT.

Kaishun Wu (Fellow, IEEE) is the Associate Vice President for Research at HKUSTGZ. He is also a full professor of the DSA & IoT Thrust Area under the Information Hub. He received his Ph.D. degree in computer science and engineering from HKUST in 2011. He is an active researcher with more than 200 papers published in major international academic journals and conferences. He received the 2012 Hong Kong Young Scientist Award, the 2014 Hong Kong ICT Awards: Best Innovation, and the 2014 IEEE ComSoc Asia-Pacific Outstanding Young Researcher Award. He is an IET Fellow.

Takahiro Hara received the B.E, M.E, and Dr.E. degrees in Osaka University in 1995, 1997, and 2000, respectively. Currently, he is a full Professor at the Department of Multimedia Engineering, Osaka University. He has published more than 550 Journal and international conference papers in the areas of databases, mobile computing, distributed systems, WWW, social computing, and wireless networking. He served as a General Chair of SRDS'14 and Mobiquitous'16 and '22. He served and is serving as a Program Chair of MDM'06, 10, 18, and 21, Mobiquitous'13, and SRDS'12. He served and is serving as a Program Committee Member of more than 200 international conferences including top-ranked ones such as VLDB, WWW, and CIKM. He is a distinguished scientist of ACM and a senior member of IEEE.

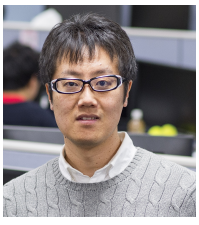


Hirotomo Oshima received his B.S. and M.S. degrees from Keio University, Japan, in 2018 and 2020 respectively. He is currently a researcher at Toshiba Corporation. His research interests include action recognition and production planning. He is a member of the Japan Society of Mechanical Engineers(JSME) and the Information Processing Society of Japan(IPSJ).



Masamitsu Fukuda is currently a researcher at Toshiba Corporation. His research interests include action recognition and production planning.

Yasuo Namioka received his B.S. and M.S. degrees from Center University, Japan in 1988 and 1990, and his Ph.D. degree from Osaka University, Japan, in 2003. From 1990 to 2024, he was a researcher at the Toshiba Cooperation Innovation Center in Japan. He is currently a professor at the Advanced Institute of Industrial Technology, Japan. His research interests include Big Data, Task and Activity Recognition, Mixed Reality, and Database Cardinality.



Takuya Maekawa received his B.S., M.S., and Ph.D. degrees from Osaka University, Japan, in 2003, 2004, and 2006, respectively. He is currently a distinguished professor at Osaka University. His research interests include ubiquitous and mobile sensing, web data mining, and information retrieval. He has published 28 full papers at top conferences of the mobile/ubiquitous computing research community (IMWUT, ISWC, PerCom) and 4 papers in the biologging community (Nature Communications and PNAS).