LIMU-BERT: Unleashing the Potential of Unlabeled Data for IMU Sensing Applications

Huatao Xu, Pengfei Zhou, Rui Tan, Mo Li*
Nanyang Technological University
Singapore
{huatao001,pengfei.zhou,tanrui,limo}@ntu.edu.sg

Guobin Shen
Alibaba Group
China
shen.sgb@alibaba-inc.com

ABSTRACT

Deep learning greatly empowers Inertial Measurement Unit (IMU) sensors for various mobile sensing applications, including human activity recognition, human-computer interaction, localization and tracking, and many more. Most existing works require substantial amounts of well-curated labeled data to train IMU-based sensing models, which incurs high annotation and training costs. Compared with labeled data, unlabeled IMU data are abundant and easily accessible. In this work, we present LIMU-BERT, a novel representation learning model that can make use of unlabeled IMU data and extract generalized rather than task-specific features. LIMU-BERT adopts the principle of self-supervised training of the natural language model BERT to effectively capture temporal relations and feature distributions in IMU sensor measurements. However, the original BERT is not adaptive to mobile IMU data. By meticulously observing the characteristics of IMU sensors, we propose a series of techniques and accordingly adapt LIMU-BERT to IMU sensing tasks. The designed models are lightweight and easily deployable on mobile devices. With the representations learned via LIMU-BERT, task-specific models trained with limited labeled samples can achieve superior performances. We extensively evaluate LIMU-BERT with four open datasets. The results show that the LIMU-BERT enhanced models significantly outperform existing approaches in two typical IMU sensing applications.

CCS CONCEPTS

• **Human-centered computing → Ubiquitous and mobile computing systems and tools**; • **Computing methodologies → Machine learning**.

KEYWORDS

IMU, Mobile Sensing, Representation Learning, BERT

ACM Reference Format:
Huatao Xu, Pengfei Zhou, Rui Tan, Mo Li and Guobin Shen. 2021. LIMU-BERT: Unleashing the Potential of Unlabeled Data for IMU Sensing Applications. In *The 19th ACM Conference on Embedded Networked Sensor Systems (SenSys'21), November 15–17, 2021, Coimbra, Portugal.* ACM, New York, NY, USA, 14 pages. https://doi.org/10.1145/3485730.3485937

*Pengfei Zhou and Mo Li are also with Alibaba-NTU Singapore Joint Research Institute.

1 INTRODUCTION

In recent years, the proliferation of embedded and mobile devices unveils the era of Artificial Intelligence of Things (AIoT). Particularly, wearable devices have played a critical role in a wide range of applications, including human activity recognition [13, 33, 51], human-computer interaction [24], localization and tracking [14, 55, 60], and etc. Many of them highly rely on the data from Inertial Measurement Unit (IMU) sensors (i.e., accelerometer, gyroscope, and magnetometer), which are widely equipped in personal mobile devices, such as smartphones, smartwatches, and even smart earphones.

Due to the rapid development of deep learning, many works adopt deep neural networks to process IMU data [13, 21, 23, 33, 51]. Compared with manual feature engineering, deep learning algorithms can extract more effective features and gain significant performance improvements in inference. Most existing works [13, 21, 23, 33, 51], however, rely heavily on supervised learning processes where substantial amounts of labeled IMU data are required to train sensing models. The requirement of large labeled data hinders their adoption in practice for two reasons. First, labeled IMU data are scarce because it is costly and time-consuming to collect sufficient labeled IMU samples in the real-world settings. Second, the diversity in mobile devices, usage patterns, and environments results in the need for labeled data with various combinations of phone models, users, and usage scenarios to attain generalizable models.

To address the challenge of labeled data scarcity, this paper proposes a representation learning model that can leverage massive unlabeled data to extract general features through self-supervised training technique. After the representations are learned, multiple task-specific inference models can thus be trained with a small amount of labeled IMU samples. The key rationale is to learn the generalizable representations from the abundant unlabeled IMU data instead of scarce labeled data. In particular, unlabeled data can be easily collected from a variety of wearable devices, usage patterns, and scenarios.

To design such a representation learning model, we first answer the following basic question to clarify our target: *what general features are desired from the IMU data*? After scrutinizing the characteristics of IMU data, we focus on two types of features: distributions of individual measurements of IMU sensors, and temporal relations in continuous measurements. The correlation of IMU readings on three axes gives information about the attitude of the device, which is an important feature of the usage patterns. For example, the readings of the accelerometer and gyroscope become dramatically larger if the user transits to walking from standing still. The three-axis components of the accelerometer correlate differently when the device orientation varies. As a typical time-series data, temporal relations within sensory data give further information about user behaviors.

Our model design is thereafter guided by answering the following question: *how to extract those general features or representations from unlabeled IMU data?* Inspired by the emerging self-supervised techniques in natural language processing, we borrow the key framework of BERT [6] to process unlabeled IMU data and accordingly extract general features. BERT designs two novel self-supervised training methods to learn the bidirectional language representations from the unlabeled text. However, intended for natural language data processing, original BERT lacks methodology in processing IMU data, e.g., multi-modality problem of various IMU sensor readings. This paper thus devises a variety of techniques including data fusion and normalization, effective training method, structure optimization, and embeds them into the BERT framework for improved efficacy and efficiency in IMU sensing applications.

We name our model **LIMU-BERT**, which stands for a lite BERT-like self-supervised representation learning model for mobile IMU data. To demonstrate the effectiveness and generalizability of learned representations, we conduct extensive experiments with four open IMU datasets. The results demonstrate that LIMU-BERT enhanced deep learning approaches significantly outperform state-of-the-art approaches, i.e., by at least 10% in terms of both accuracy and F1-score in Human Activity Recognition (HAR) as well as Device Placement Classification (DPC) applications. In summary, this paper makes the following contributions:

- This paper devises a self-supervised approach to learn general representations from unlabeled IMU data. Based on learned representations, task-specific models can be trained with few labeled samples, which substantially reduces the supervised training overhead with labeled data.
- This paper proposes a series of adaptations and enhancements around BERT to best work with IMU data in mobile sensing applications. The proposed LIMU-BERT is lightweight, which can be accommodated in mobile devices.
- A prototype system is developed and experimentally evaluated. Extensive evaluation results show the effectiveness of LIMU-BERT in learning generalizable data representations. The codes of LIMU-BERT are made publicly available [1].

The rest of this paper is organized as follows. Section 2 presents the preliminary knowledge this paper is built on. Section 3 introduces LIMU-BERT workflow along with the design details. Section 4 provides the experiment details and evaluation results. Section 5 reviews the related works and Section 6 discusses the limitations and possible future studies. Section 7 concludes this paper.

2 PRELIMINARIES

2.1 Representation Learning

Representation learning techniques aim to extract the representations or general features from raw data. Traditional methods rely on domain expertise or prior knowledge to engineer features. Recent studies show that automatic representation learning with deep neural network is effective when provided large labeled data. It is however costly and time-consuming to gather a dataset with clean labels. To make use of large amounts of unlabeled data, more advanced models [6, 7, 19, 53] have been proposed to extract representations from unlabeled images, video, or texts. The process of learning representations from unlabeled data is called *self-supervised* learning. The literature has shown that self-supervised learning brings significant performance gains for a range of challenging tasks.

Bidirectional Encoder Representations from Transformers (BERT) [6] is one effective self-supervised learning model for Natural Language Processing (NLP), which improves the performances of NLP models to higher levels in various NLP applications. BERT features two self-supervised tasks: Masked Language Model (MLM) and Next Sentence Prediction (NSP). MLM randomly masks parts of the input text and the model is trained to predict the original identity numbers of the masked words, while NSP requires the model to determine whether two given sentences are subsequent or not. Through MLM and NSP, BERT can learn contextual relations in text data and accordingly generate effective embeddings for each word. After the self-supervised training process, pre-trained models can be connected with task-specific models to perform various NLP tasks with supervised training.

Inspired by BERT, we aim at employing a similarly effective self-supervised technique for mobile sensing. As time-series type data, IMU sensor data also contain rich contextual relations. We believe effective representations extracted from unlabeled IMU data can improve the performances of down-stream inference models and significantly reduce the label requirements at the same time.

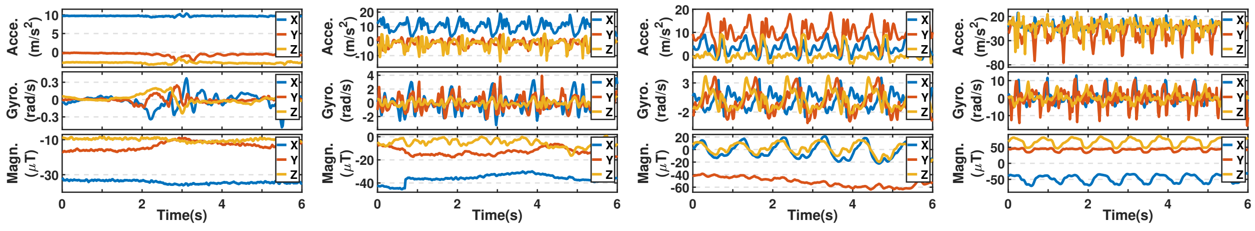
2.2 Uniqueness of IMU Sensing

BERT is designed for processing text data, which greatly differs from IMU data. To make the BERT adaptive to IMU sensing applications, we carefully examine the characteristics of IMU data and obtain the following key observations that will guide the designs of LIMU-BERT. Figure 1 provides four sets of IMU (accelerometer, gyroscope, magnetometer) readings of different human activities with different device placements. For instances, the readings in Figure 1(a) are collected when the user stands still with the smartphone placed in the bag, and those in Figure 1(c) are collected when the user walks with the smartphone placed in the pocket.

Fusion matters. In Figure 1(a), gyroscope readings have an evident fluctuation while accelerometer readings are steadier since gyroscope is more sensitive to the movement. The impact of gyroscope fluctuation can be mitigated if the changes of accelerometer readings are considered. In other words, cross referencing of multiple sensors can provide more information and improves the overall performance, which has been shown by prior work [36]. Thus, aligned with the recent research interest of multi-modal sensor fusion [23, 49, 51], the representation learning model should support the data fusion with multiple IMU sensors, which is not the design objective of the original BERT for NLP.

Distribution matters. By comparing individual measurements of accelerometer and gyroscope in the three activities (i.e., standing, walking, and running), we find their ranges greatly vary. For example, the gyroscope readings are within $(-5, 5)$ when the user walks

[1]https://github.com/dapowan/LIMU-BERT-Public



(a) Still, Bag (b) Walk, Bag (c) Walk, Pocket (d) Run, Pocket

Figure 1: IMU measurements of daily activities with different device placements.

and are distributed between -15 and 15 when the user runs. The relations among the 3-axis readings also change when the device is placed differently. It is obvious that accelerometer readings on y-axis have largest values compared to other two axes if the device is in a bag but they are differently related when the device is placed in the pocket. Therefore, we believe the distribution of IMU readings contains rich information, which is one feature LIMU-BERT should capture. We argue that any transformation, which may destruct the distribution information of raw IMU data, should not be applied before feeding them into neural network if we want to capture general features.

Context matters. As Figure 1(a) shows, the occasional fluctuations are inevitable since human activities are complicated. However, walk and run exhibit obvious periodical patterns on the IMU data, which is a reliable feature that distinguishes them from standing still. The detailed periodical features (e.g., stride frequency) may further distinguish walk and run. Such periodical pattern further varies when the device is placed differently. By observing the IMU readings when the user is walking in Figure 1(b) and Figure 1(c), we find that readings of the smartphone in the pocket have more significant changes. This is likely because the smartphone orientation changes frequently as the leg moves compared to it is in the bag. In summary, temporal relations also play an important role in representation learning for IMU data, which will likely benefit from a BERT-like design.

Efficiency matters. LIMU-BERT targets processing IMU data collected from mobile devices in real time. Mobile devices have limited computation capability and battery capacity. The base model of BERT has about 110 million parameters, which is too heavy for mobile devices. Therefore, a lightweight and efficient design is needed.

2.3 Potential Applications

Many mobile devices are equipped with IMU sensors (i.e., accelerometer, gyroscope, and magnetometer), which are widely used in various applications including estimating the attitudes of mobile devices [34, 59], tracking users' arm [24, 35], and implementing secure text pin [3]. They recently enable human activity recognition [13, 33, 51], which has received high research attention. It seeks profound high-level knowledge about human activities. The motion information extracted from IMU data can also be used for indoor localization and tracking [38, 39, 50, 56]. For those applications, we find that device placements play an important role, e.g., Ear-AR [50] and RMPCA [5] need to know when the smartphones are in the pant pockets before estimate the heading direction or step length.

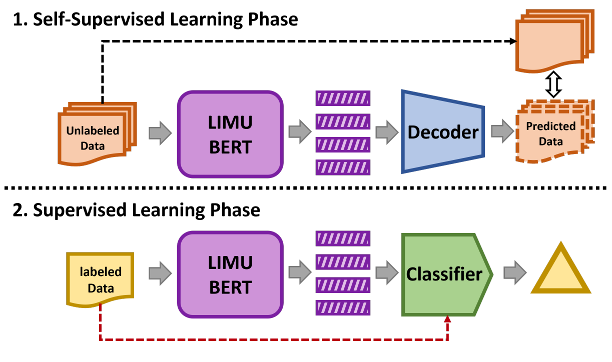


Figure 2: Framework overview.

For these influential applications of IMU sensors, LIMU-BERT aims at extracting general features to achieve superior performance with limited labeled samples. This paper will demonstrate the efficacy of LIMU-BERT using the following applications as vehicles, i.e., Human Activity Recognition (HAR) and Device Placement Classification (DPC).

3 DESIGN

3.1 Overview

Before diving into the details, the overview of our framework is presented in Figure 2, which consists of self-supervised and supervised learning phases. There are three major components in our framework, including *LIMU-BERT*, *decoder*, and *classifier*. The *LIMU-BERT* takes the unlabeled IMU data as input and outputs high-level representations or features (the striped rectangles in Figure 2). The decoder reconstructs the unlabeled data based on the learned features. The classifier trained with a small amount of labeled representations aims to accomplish a task-specific application, such as HAR.

Self-supervised learning. In this phase, we mask partial readings of unlabeled samples and feed them into LIMU-BERT. LIMU-BERT and the decoder jointly predict the original values of masked readings by learning the temporal relations among IMU data. The objective of self-supervised learning process is to fully utilize a large amount of unlabeled data and accordingly extract general features.

Supervised learning. Next, we transfer the LIMU-BERT model and connect it with a classifier. In this phase, all parameters of the LIMU-BERT are frozen and only the classifier is trained with limited labeled representations that have been processed by the LIMU-BERT. At run time after the supervised learning, the LIMU-BERT and classifier are deployed together to estimate the task-specific results for IMU sensor data.

In the following subsections, we introduce the detailed designs of our framework along with the structures of the three components.

3.2 Fusion and Normalization

According to the first observation mentioned in Section 2.2, LIMU-BERT should handle multiple sensor data. But the readings of IMU sensors have different distributions, which can be seen in Figure 1. And such differences would affect the model performance based on our experiments. The original BERT, however, does not consider the normalization issue since the input data are well-normalized features (e.g., one-hot vectors). Therefore, the sensor measurements need to be properly normalized before being fed into LIMU-BERT.

Common normalization methods are min-max normalization or mean-variance normalization. Some existing normalization approaches apply data transformations, e.g., DeepSense [51] applies Fourier transform and replaces raw data with frequency domain features, which are free from the normalization issue. Those methods, however, result in the losses of distribution information and may negatively affect the quality of the general representations. To this end, we design a simple but effective normalization method on accelerometer and magnetometer readings to narrow the range differences and not severely alter their distributions, which can be expressed by:

$$acc_i = \frac{acc_i}{9.8\ m/s^2},\ mag_i = \frac{\alpha \cdot mag_i}{\sqrt{\sum mag_i^2}},\ i \in \{x, y, z\} \tag{1}$$

where acc_i and mag_i denote the measurements in the i-axis of the accelerometer and magnetometer, respectively. The α is a weight scaling the range of the magnetometer readings, which is set to 2 in LIMU-BERT. The magnetometer reading are normalized with the magnitude since it is easily affected by environments (such as electronic devices nearby). But the relations among the 3-axis readings of magnetometer remain. We keep the distribution of gyroscope readings because they are naturally small values. By Equation 1, all IMU sensor readings primarily distribute in the similar ranges (e.g., $[-8, 8]$).

The normalized IMU readings are then cut by a fixed window $X \in \mathbb{R}^{S_{dim} \times L}$, where L is the number of normalized readings in each window and S_{dim} is the dimension of collected IMU features. For instance, S_{dim} is six if only accelerometer and gyroscope data are gathered since each sensor has three dimensions. The first three features correspond to the three axes of accelerometer while the last three correspond to the three axes of gyroscope. An IMU sample X is also called an IMU sequence in our paper. The $\boldsymbol{X}$ denotes a tensor or matrix and $\boldsymbol{X}_{ij}$ denotes the element in the i-th row and the j-th column in the $\boldsymbol{X}$. Let bold upper-case letters $\boldsymbol{X}^u$ and $\boldsymbol{X}^l$ represent unlabeled and labeled samples, respectively. Each labeled sample $\boldsymbol{X}^l_{[i]} \in \boldsymbol{X}^l$ corresponds to a true label $\boldsymbol{Y}_{[i]} \in \boldsymbol{Y}$, where $\boldsymbol{Y}$ is the label set.

A critical characteristic of IMU sensors is that the number of features S_{dim} is small (e.g., six). To extend the dimension of features and fuse IMU sensors, we project the normalized sensor data X into a higher space by:

$$\boldsymbol{I} = \text{Proj}\,(X) = \boldsymbol{W} \times \boldsymbol{X} \tag{2}$$

$\boldsymbol{W}$ is a matrix of size $H_{dim} \times S_{dim}$, where H_{dim} is the hidden dimension larger than S_{dim}. From Equation 2, sensor readings collected at the same measurement time are fused together in the elements of one column in $\boldsymbol{I} \in \mathbb{R}^{H_{dim} \times L}$. Compared with the normalized sensor data X, $\boldsymbol{I}$ contains more implicit features. Note that this projection is implemented by a linear layer. Next, LIMU-BERT leverages Layer Normalization [1] to normalize the fused features corresponding to the same measurement time:

$$\boldsymbol{I}'_{ij} = \text{LayerNorm}\,(\boldsymbol{I}) = \frac{\boldsymbol{I}_{ij} - E(\boldsymbol{I}_{\cdot j})}{\sqrt{Var(\boldsymbol{I}_{\cdot j}) + \epsilon}} \cdot \gamma + \beta \tag{3}$$

where ϵ is a small value while γ and β are learning hyper parameters. The $E(\boldsymbol{I}_{\cdot j})$ and $Var(\boldsymbol{I}_{\cdot j})$ denote the mean and variance of the elements in the j-th column, respectively. By learning the parameters of the normalization layer, the neural network can dynamically normalize implicit features of IMU sensors.

3.3 Learning Representations

Based on our observations of IMU data, both distributions of individual readings and temporal relations among continuous readings are important features. Thus, self-supervised method should be able to extract both two types of representations. After analyzing the two self-supervised tasks (i.e., MLM and NSP) in BERT, we find the NSP task is not suitable for IMU data. The reason is that learning whether two IMU sequences are subsequent does not bring great benefits to the model due to the frequent transitions of human daily activities. On the other hand, we notice that the MLM task, which masks some subsequences of a sequence and the models are trained to predict the masked subsequences, has dual intentions for IMU representation learning. First, after the MLM training process, the classifier is able to reconstruct the masked readings based on the corresponding representations, which means the features learned by LIMU-BERT must contain distribution information. Second, LIMU-BERT is required to generate representations for the masked readings and such process forces it to learn the contextual relations in IMU data. In summary, we argue that MLM is beneficial to extracting our target features from IMU data.

The MLM task in BERT masks the text subsequences with only one token since many words have independent meaning (e.g., *apple, water*). However, the nature of human mobility leads to similar IMU sensor data across adjacent measurements in time, which can be observed in Figure 1. Therefore, the model will easily degrade to reconstructing the masked readings by mirroring neighboring readings if only the one-sample subsequences are masked. Longer subsequences should be masked in order to provide a challenging condition and thus train an effective model. To decide the length of a masked subsequence, we implement a Span Masking mechanism [15], which samples the length of masked subsequence (denoted by l) from a geometric distribution $Geo(p)$ clipped at l_{max}:

$$P(l = k) = (1 - p)^{k-1} p,\ s.t.\ l \in [1, l_{max}] \tag{4}$$

Algorithm 1: Span mask algorithm

```
Input  : IMU sequence $X$, sequence length $L$, probability of
         success $p$, masked ratio $p_r$, mask probability $P_m$
Output : Masked IMU sequence $X$, masked position set $I$
1  $M_{max} = L \times p_m$, $m = 0$, $I = \emptyset$;
2  sample $p_m$ from $U[0, 1)$;
3  while $m < M_{max}$ do
4      sample $s$ from $U[0, L)$;
5      if $s \notin I$ then
6          sample $l$ from $Geo(p)$;
7          $l = \min(l, M_{max} - m)$, $e = \min(s + l, L)$;
8          for $j = s$ to $e$ do
9              $I = I \bigcup \{j\}$, $m = m + 1$;
10             if $p_m < P_m$ then
11                 $X_{\cdot j} = 0$;
12             end
13         end
14     end
15 end
```

where the probability of success p is set to 0.2 and l_{max} is set to 10. By masking longer subsequences, LIMU-BERT is challenged to learn the temporal relations in IMU data more effectively and accordingly reconstruct the masked subsequences from the context. In addition, the IMU readings of each input sequence are gathered at a longer period of time (i.e., 6 seconds), which contain richer temporal information.

The details of the mask method is summarized in Algorithm 1. The $U[a, b)$ in line 2 and 4 represents discrete uniform distribution with an interval of $[a, b)$. The M_{max} is the maximum number of the masked readings in one IMU sequence and the equation in line 7 guarantees that a total of M_{max} readings are masked each time. The s and e are the start and end indexes of each subsequence. In line 2, we sample a p_m uniformly and randomly from [0, 1) and the IMU sequence would be masked only if $p_m < P_m$. In other words, the masking is performed with a probability of P_m. The reason is that the input data are not unmasked in the supervised learning phase, leading to the differences between the input data of two learning phases. To tackle this issue, LIMU-BERT can learn how to deal with both unmasked and masked data through probabilistic masking. All values of the selected readings are replaced with 0 in line 11. The masked ratio p_r and masking probability P_m are set to 0.15 and 0.8, respectively. The masked position set I will be used in the loss function.

3.4 Lightweight Model

Different from BERT, LIMU-BERT must be lightweight enough to run on mobile devices. At the same time, lightweight LIMU-BERT should be able to learn general representations from unlabeled IMU data. To achieve this goal, we make the following customizations.

First, given the sampling rate of IMU sensors, the number of readings collected in six seconds is large. For example, the sequence length L is 600 if the sampling rate is 100 Hz, which is larger than the maximum sequence length (i.e., 512) in the original BERT. Large

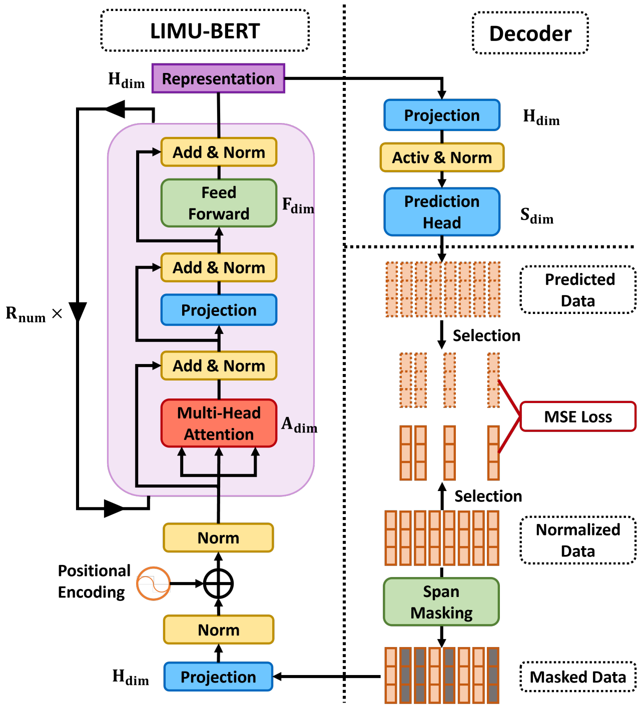


Figure 3: Self-supervised training workflow.

sequence length would increase model complexity significantly. Thus, we adopt a much smaller sampling rate (i.e., 20 Hz) compared with the existing works [8, 33, 51], and accordingly decrease the length of the input IMU sequences. Extensive experiments show that 20 Hz is enough for our target applications. In addition, the representations dimension H_{dim} of LIMU-BERT is smaller than that of the original BERT (e.g., 1024) due to the small number of features for IMU sensors, which helps shrink the model size.

Second, LIMU-BERT adopts a cross-layer parameter sharing mechanism [19] to improve parameter efficiency. LIMU-BERT consists of multiple encoder layers, where only the parameters in the first encoder layer are trained. The parameters of the first layer are shared with other layers. This mechanism reduces the number of parameters of LIMU-BERT significantly.

Third, we treat the IMU data reconstruction problem as a regression task rather than a classification task because the IMU features are continuous variables. Instead of handling large number of categories in the classification task, regression model can avoid a heavy output layer and simplify the decoder considerably.

3.5 LIMU-BERT Design

LIMU-BERT. Putting all designs above together, the detailed workflow of self-supervised process is illustrated in Figure 3. The objective of LIMU-BERT is to generate representations for unlabeled IMU data, which can be formulated as:

$$E = f_{enc}(X^u) \tag{5}$$

where E is a $H_{dim} \times L$ matrix. In the beginning, normalized data X needs to be masked before being fed into LIMU-BERT. The first projection and norm component together implement the sensor fusion and normalization design in Equation 2 and Equation 3. Note that all normalization components (i.e., the yellow rectangles in Figure 3) represent layer normalization. Next, positional encoding [43] is added into the input data to make full use of the order information. After the second layer normalization layer, the hidden features are expressed as follows:

$$\boldsymbol{H}^{\{0\}}_{\cdot j} = \text{LayerNorm}\left(\boldsymbol{I}'_{\cdot j} + PE(j)\right) \tag{6}$$

where $PE(\cdot)$ is the positional embedding function, which maps an order (column) index to a vector with length of H_{dim}. All positional embeddings are trainable variables. An attention-enteric block (i.e., the purple rectangle in Figure 3) then takes $\boldsymbol{H}$ as input and repeats for R_{num} times before outputs the final representations. All components in this block are identical and this process implements cross-layer parameter sharing mechanism. There are three residual blocks in the attention-enteric block, which can be expressed as:

$$\begin{aligned} \boldsymbol{H}^{\{r\}} &= \text{LayerNorm}\left(\text{FeedForward}\left(\boldsymbol{P}^{\{r-1\}}\right) + \boldsymbol{P}^{\{r-1\}}\right) \\ \boldsymbol{P}^{\{r-1\}} &= \text{LayerNorm}\left(\text{Proj}\left(\boldsymbol{A}^{\{r-1\}}\right) + \boldsymbol{A}^{\{r-1\}}\right) \\ \boldsymbol{A}^{\{r-1\}} &= \text{LayerNorm}\left(\text{MultiAttn}\left(\boldsymbol{H}^{\{r-1\}}\right) + \boldsymbol{H}^{\{r-1\}}\right) \end{aligned} \tag{7}$$

where r is an integer in $[1, R_{num}]$. The MultiAttn$(\cdot)$ is a self-attention layer [43] with A_{num} attention heads. The hidden dimensions of query, key, value in the attention layer are H_{dim}. The Proj$(\cdot)$ represents a fully connected layer, whose input and output dimensions are both H_{dim}. The FeedForward$(\cdot)$ consists of two fully-connected layers with a hidden dimension of F_{dim} and its input and output dimensions are the same as those of Proj$(\cdot)$. There is a Gaussian Error Linear Unit (GELU) [9] activation function between two fully-connected layers. Finally, we can get representations $\boldsymbol{E} = \boldsymbol{H}^{\{R_{num}\}}$ for a masked IMU sequence $\boldsymbol{X^m}$. In LIMU-BERT, R_{num} and H_{dim} are set to 4. According to the previous designs, the L is set to 120 under a sampling rate of 20Hz.

Decoder. The aim of the decoder is to reconstruct the original values of the masked IMU sequences with the representations generated by LIMU-BERT, which can be formulated as:

$$\hat{\boldsymbol{X}}^{\boldsymbol{u}} = f_{dec}(\boldsymbol{E}) \tag{8}$$

f_{dec} consists of three components: a projection, an activated and normalization layer, and a prediction head. The decoder can be formulated as follows:

$$\begin{aligned} \hat{\boldsymbol{X}}^{\boldsymbol{u}} &= \text{LayerNorm}\left(\text{Pred}\left(\boldsymbol{D}\right)\right) \\ \boldsymbol{D} &= \text{Proj}\left(\text{GELU}\left(\boldsymbol{E}\right)\right) \end{aligned} \tag{9}$$

where the Pred$(\cdot)$ and the Proj$(\cdot)$ denotes single fully-connected layer with unit numbers of S_{dim} and H_{dim} respectively. Finally, we get the reconstructed IMU sequence $\hat{\boldsymbol{X}}^{\boldsymbol{u}}$ from the masked IMU sequence.

Training. As we mentioned earlier, the reconstruction problem is regarded as a regression task. Therefore, the loss function in self-supervised phase is defined as follows:

$$loss = \frac{1}{|X^u|}\sum_{i=1}^{|X^u|} \text{MSE}\left(\text{Select}\left(\boldsymbol{X}^{u}_{[i]}\right), \text{Select}\left(\hat{\boldsymbol{X}}^{u}_{[i]}\right)\right) \tag{10}$$

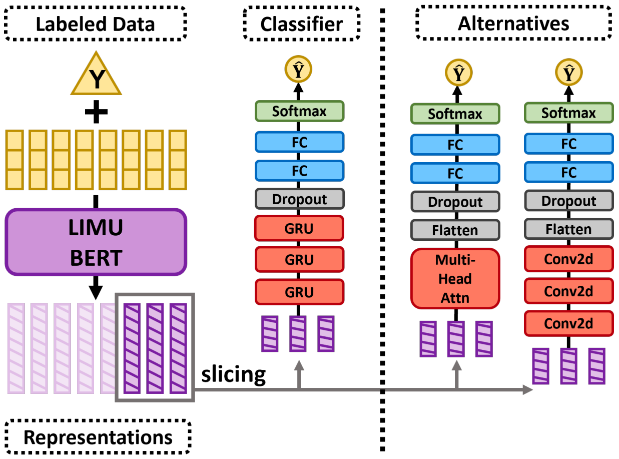


Figure 4: Supervised training workflow.

where MSE$(\cdot)$ denotes Mean Square Error (MSE) function and $|X^u|$ is the number of unlabeled samples. Select$(\cdot)$ represents the selection process in Figure 1, which selects the readings corresponding to the masked position set I in Algorithm 1. In other words, only the reconstruction losses of the masked IMU subsequences are computed. In the self-supervised training process, the parameters in LIMU-BERT and decoder are updated with Adam [16] optimizer.

3.6 Task-specific Classifier Design

Classifier. After the LIMU-BERT is trained with unlabeled data, it can be utilized to generate representations for labeled IMU data. Based on the learned representations and their corresponding labels, we can design task-specific models with supervised training. The whole process is illustrated in Figure 4. Note that the IMU sequences are not masked in the supervised training phase. Since the time period (i.e., 6 seconds) of the input IMU sequence is relatively longer, we can slice the subsequences of the representations to design the task-specific classifier if a fine granularity is needed. The sequence length of the classifier is denoted as L_c. Note that slicing is an optional step here.

In our framework, we design a lightweight classifier with Gated Recurrent Unit (GRU) [4] as illustrated in Figure 4. It contains three stacked GRU layers with hidden sizes of 20, 20, and 10, respectively. The input size of first GRU layer is H_{dim}. Upon the GRU layers, only the hidden features at the last position are fed into dropout layer with a drop rate of 0.5, which aims at reducing over-fitting. Next, two fully-connected layers with 10 hidden units are constructed before the softmax layer. The final output size is identical to the number of classes in the targeted task. The GRU classifier is very lightweight since only limited labeled samples are available.

In addition to the GRU classifier, we can design classifiers with other neural network structures, such as Convolutional Neural Network (CNN) [18] or Multi-head Attention [43]. We propose two alternative classifiers in Figure 4. The hidden dimension of the attention component in the first alternative is 36. For CNN-based classifier, it adopts three 2D-convolutional layers, which consists of 8, 16, and 4 feature maps, respectively, with kernel sizes of (3,3).

There is a max pooling layer with (3,3) kernel and (2,1) stride after each convolutional layer. Other components are the same as those in the GRU classifier.

Training. The cross entropy loss is adopted to support multi-class classification. Note that only the parameters in task-specific classifiers are updated with Adam [16] optimizer in the supervised training phase.

4 EVALUATION

4.1 Methodology

4.1.1 Datasets. We quantitatively evaluate the effectiveness of LIMU-BERT with four publicly available datasets, which have been commonly used in previous works [33, 49, 51]. Those datasets cover a wide variety of device types, activities, users, and environments.
HHAR. HHAR dataset [40] contains readings from accelerometers and gyroscopes from 9 users with 6 different activities (*biking, sitting, standing, walking, upstairs, and downstairs*), and 6 types of mobile phones (3 models of Samsung Galaxy and one model of LG), which are carried by the users around their waist. The sampling rate differs from 100 Hz to 200 Hz.
UCI. For UCI [32] dataset, 30 volunteers aged from 19 to 48 years performed 6 basic activities (*standing, sitting, lying, walking, walking downstairs, and walking upstairs*) with a smartphone (Samsung Galaxy S II) on the waist. Raw accelerometer and gyroscope readings were collected at a constant sampling rate of 50 Hz.
MotionSense. MotionSense [25] includes accelerometer and gyroscope time-series data collected by an iPhone 6s. A total of 24 participants diverse in gender, age, weight, and height performed 6 activities (*downstairs, upstairs, walking, jogging, sitting, and standing*) with the smartphone in their front pockets. All data were collected at a 50 Hz sample rate.
Shoaib. Shoaib et al. [36] collected data of seven physical activities (*walking, sitting, standing, jogging, biking, walking upstairs, and walking downstairs*). During data collection, 10 male participants were equipped with five Samsung Galaxy SII (i9100) smartphones placed on five body positions (*right pocket, left pocket, belt, upper arm, and wrist*). Accelerometer, gyroscope, and magnetometer readings were collected at the rate of 50 samples per second.

4.1.2 Preprocessing. For all datasets, we first down-sample to 20 Hz[2] and slice the continuous IMU data into the window with a length of 120 measurements. There is no overlapping between any two windows. Each sample is labeled with an activity type, and device placement (if applicable). The key attributes of the four datasets are summarized in Table 1.

We randomly divide each dataset into training (80%), validation (10%), and test (10%) sets. The training set is further divided into 1% as labeled set and 99% as unlabeled set. The ratio of the number of samples in labeled set to that in training set is called the **labeling rate** in our paper, which equals to 1% unless otherwise specified. In self-supervised training phase, the training set is used to jointly train the LIMU-BERT and the decoder, while the validation set is used to select the models. In the supervised training phase, the task-specific classifiers are trained with the labeled set and selected by the validation set. In our experiment setting, only 1% of labeled samples are utilized to train task-specific classifiers, which is mimic the practical application scenario where only limited labels are available. The trained classifiers are finally evaluated on the test set. When dividing the training set, we ensure that each class has the same amount of labeled samples. The class imbalance problem [11] is not considered in our paper. We argue that the its impact to the task-specific classifiers is limited if only a small amount of labeled data is required.

[2]High sampling rates lengthen the input sequence and thus increase the model complexities, which may cause severe over-fitting. Therefore, the sampling rate is set to 20 Hz for the balancing the performance and efficiency.

Table 1: Datasets summary. (A=accelerometer, G=gyroscope, M=magnetometer)

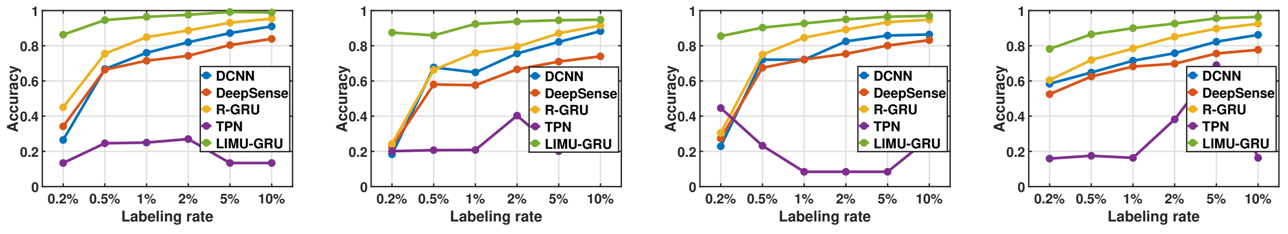
Dataset	Sensor	Activity	User	Placement	Sample
HHAR	A,G	6	9	-	9166
UCI	A,G	6	30	-	2088
MotionSense	A,G	6	24	-	4534
Shoaib	A,G,M	7	10	5	10500

4.1.3 Models in comparison. Our approach is evaluated by comparing it with other alternative models, which are based on state-of-the-art machine models in HAR.
LIMU-GRU. LIMU-GRU is a classification model implemented based on our framework. The input data are the representations learned by LIMU-BERT. The classifier is constructed with GRU introduced in Section 3.6.
DCNN [49]. It designs a deep CNN-based model to automate feature learning from the raw IMU inputs for the HAR problem, which outperforms many traditional method, including support vector machine (SVM) and deep belief network (DBN). To make DCNN adaptive to our setting, we slightly adjust the size of input and output layers.
DeepSense [51]. Different from other models, DeepSense applies Fourier Transform to the raw IMU data and feeds the frequency domain features (i.e., magnitude and phase pairs) to the neural network. It adopts a deep learning structure to fuse data from multiple sensors and accordingly extract temporal features. Since the input setting in DeepSense is different from ours, we implement the CNNs in DeepSense with minor adaptions (e.g., smaller kernels).
TPN [33]. To the best of our knowledge, TPN is the only study that is able to learn general features from unlabeled data before training task-specific models. A multi-task temporal CNN is trained to recognize several transformations applied on the input data. The temporal CNN is then transferred to the HAR classification model. We make TPN adapt to the window size by reducing the kernel sizes in the CNN layers.
R-GRU. To show the effectiveness of representations learned by LIMU-BERT, we implement a baseline model that directly applies the GRU classifier on the raw IMU data.

4.1.4 Implementation. LIMU-BERT and other baseline models are implemented with Python and PyTorch [28]. They are trained in a server equipped with 4 NVIDIA GEFORE 2080Ti GPUs, 128 GB memory, and an Intel(R) Core(TM) i9-9820X 3.30GHz CPU. The learning rate and batch size in both self-supervised and supervised training phases are the same, which are 0.001 and 128, respectively.

Table 2: Performance comparison on HAR with 1% labeled data.

Dataset	HHAR		UCI		MotionSense		Shoaib		Average	
Metric	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1
DCNN	0.760	0.736	0.649	0.625	0.721	0.637	0.715	0.718	0.711	0.679
DeepSense	0.715	0.688	0.576	0.544	0.722	0.650	0.682	0.683	0.674	0.641
R-GRU	0.849	0.832	0.760	0.741	0.846	0.806	0.785	0.787	0.810	0.792
TPN	0.250	0.151	0.208	0.068	0.084	0.026	0.163	0.040	0.176	0.071
LIMU-GRU	**0.964**	**0.962**	**0.924**	**0.923**	**0.927**	**0.899**	**0.900**	**0.899**	**0.929**	**0.921**



(a) HHAR **(b) UCI** **(c) MotionSense** **(d) Shoaib**

Figure 5: Accuracy comparison on HAR at different labeling rates.

To ensure a fair evaluation, all classifiers and baseline models are trained with the same training hyper-parameters and labeled set in the supervised training process. LIMU-BERT and TPN are pre-trained with the unlabeled samples in the training set for 3,200 epochs. The GRU classifier and baseline models are trained with unlabeled samples in the label set for 700 epochs. All the models utilize the same datasets for self-supervised training or supervised training, validation, and testing. The default input data are accelerometer and gyroscope readings. The sequence length of classifiers L_c is 20 unless otherwise specified.

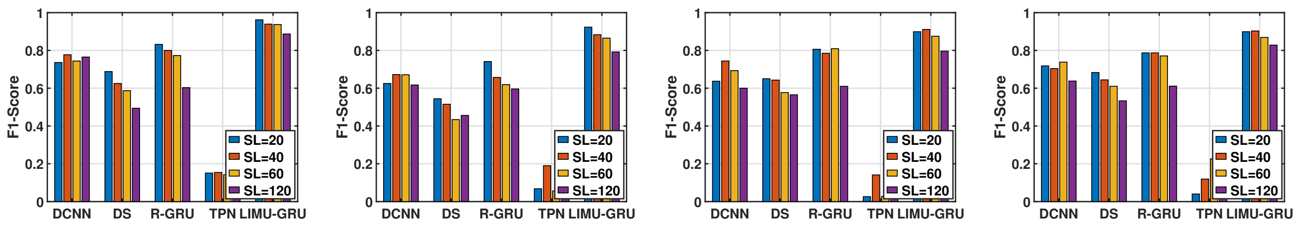
4.1.5 Application and metrics. We compare all models on two applications: Human Activity Recognition (HAR) and Device Placement Classification (DPC). In the HAR task, models are trained to recognize human activities (e.g. recognize *standing, sitting,lying, walking, walking downstairs, or walking upstairs* on the HHAR dataset) with IMU data. Similarly, models process IMU data and determine the placement of the device (i.e., determine *right pocket, left pocket, belt, upper arm, or wrist* on the Shoaib dataset) in the DPC task. As both two applications are classification tasks, we adopt accuracy and macro F-score for performance comparison.

4.2 Evaluation of Human Activity Recognition

4.2.1 Overall performances. Table 2 gives the comparative performances of LIMU-GRU and other baseline models in HAR application. The labeling rate is 0.01, i.e., all models only utilize 1% labeled samples of training set. According to the results, LIMU-GRU outperforms other baseline models by a large margin (at least 10%) in all cases, which demonstrates the effectiveness of LIMU-BERT. In general, LIMU-GRU achieves nearly 0.90 accuracy and F1-score with all datasets. The performances of DCNN and DeepSense are even worse than R-GRU, a simpler model. We suspect that may be due to over-fitting issue when only limited labeled samples can be leveraged. The performances of TPN is the worst, probably due to the reason that TPN is not designed to handle multi-modality data with readings from a variety of IMU sensors. TPN classifier does not converge correctly, though it has a self-supervised phase for the temporal CNN. In addition, the performance gap between R-GRU and LIMU-GRU clearly suggests that the representations learned by LIMU-BERT are effective (over 10% improvement on accuracy and F1-score).

In summary, the performance gain of LIMU-GRU is significant, thanks to the effective and generalizable features extracted by LIMU-BERT. The results also show that 20 Hz sampling rate of IMU sensors already delivers high performance for the HAR task.

4.2.2 Varying labeling rate. In this experiment, we investigate the performances of DCNN, DeepSense, R-GRU and LIMU-GRU at different labeling rates, varying from 0.2% to 10%. Figure 5 depicts the comparison results with all datasets. Most models are able to achieve higher accuracies as the labeling rate increases. The results show that LIMU-GRU consistently outperforms the baseline in all cases. The performance gaps between LIMU-GRU and other models are higher when the labeling rate is smaller. For examples, LIMU-GRU obtains accuracies of 0.863, 0.875, and 0.855 at labeling rate 0.2% on HHAR, UCI, and MotionSense dataset, respectively. All other models, however, only achieve accuracies below 0.5. The result suggests that LIMU-BERT is able to effectively learn general representations from the unlabeled data, and the down-stream GRU classifier achieves higher accuracy with learned representations. The performance gain is significant especially when the labeled data samples are fewer.



(a) HHAR **(b) UCI** **(c) MotionSense** **(d) Shoaib**

Figure 6: Accuracy comparison on HAR with different sequence lengths (SL).

Table 3: Performance comparison on DPC.

Labeling rate	0.2%		0.5%		1%		2%		5%		10%		Average	
Metric	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1
DCNN	0.486	0.484	0.561	0.564	0.657	0.652	0.781	0.778	0.842	0.839	0.900	0.897	0.705	0.702
DeepSense	0.463	0.457	0.537	0.529	0.600	0.594	0.669	0.663	0.765	0.761	0.819	0.815	0.642	0.637
R-GRU	0.613	0.589	0.719	0.716	0.832	0.830	0.901	0.901	0.941	0.941	0.964	0.964	0.828	0.824
TPN	0.321	0.184	0.421	0.319	0.404	0.266	0.488	0.391	0.336	0.216	0.391	0.257	0.394	0.272
LIMU-GRU	**0.753**	**0.746**	**0.886**	**0.885**	**0.920**	**0.921**	**0.948**	**0.949**	**0.969**	**0.970**	**0.984**	**0.984**	**0.910**	**0.909**

4.2.3 *Varying sequence length.* Since the representations are learned from longer IMU sequences, there is a question about whether the performance improvement of LIMU-GRU comes from the use of longer sequences in the self-supervised training phase rather than the representations learned from the unlabeled data. To investigate this, we conduct experiments by varying IMU sequence length from 120 to 20. For DCNN, we change the number of neural unit in the first fully-connected layer and make it able to work with different sequence lengths. Other models are naturally adaptive to different sequence lengths. Figure 6 plots the performances of all models, where DeepSense is abbreviated as DS. We find that the model performances are not positively related to the sequence length. In all sequence length settings, LIMU-BERT performs the best. There are several reasons to explain the result: First, longer IMU sequences would increase the model complexity (e.g., DCNN), possibly leading to over-fitting and may not necessarily yield higher accuracy. Second, when the labeled IMU measurements are limited, the increase of sequence length corresponds to fewer labeled samples, which may lead to severer label scarcity issue in the supervised training process. Overall, longer IMU sample sequence do not always lead to higher performance for the tested models.

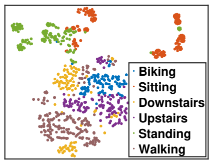
4.3 Evaluation of Device Placement Classification

In this evaluation, we quantitatively analyze the performance of LIMU-GRU in the DPC task. We change labeling rate from 0.2% to 10% and compare LIMU-GRU with other models in Table 3. The experiments are conducted on the Shoaib dataset as only Shoaib has ground truths for device placement. Similar to case of HAR, LIMU-GRU outperforms all baseline models in all cases on the DPC. The performance margins of accuracies and F1-scores are significant, especially when labeling rate is small. The average accuracy and F1-score of LIMU-GRU across all test cases are 0.910 and 0.909, respectively. As for comparisons, the accuracies and F1-score of all other models are lower than 0.828. The performance gain of LIMU-BERT over the rest models increases from 2% for 10% labeling rate all the way to 14% when the labeling rate reduces to 0.2%. The results show that the representations learned by LIMU-BERT significantly benefit the DPC application as well.

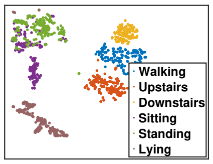
4.4 Micro-benchmark

We conduct micro-benchmark experiments to inspect LIMU-BERT and evaluate its sensitivity to various system settings.

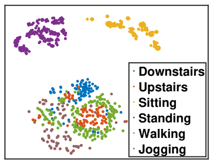
4.4.1 *Representation visualization.* To understand the effectiveness of the representations learned by LIMU-BERT, we adopt t-distributed Stochastic Neighbor Embedding (t-SNE) [42] to visualize the learned high-dimensional representations in 2D space. For each dataset, 1000 samples are selected randomly and their relevances are depicted in Figure 7. First four sub-figures depict the LIMU-BERT outputs of the four datasets on the HAR task, and Figure 7(e) depicts that on the DPC task. The clusters show the high correlations among the learned representations in all datasets. It is obvious that samples belonging to the same activity class exhibit high clustering effect. An interesting finding is that representations of dynamic activities (*walking, jogging, upstairs, downstairs, and etc.*) are likely to be close, which is in line with our understanding that those activities require many motions and thus introduce fluctuations on IMU sensor data. The representations in the Shoaib dataset are less concentrated compared with other datasets for activity labels. We believe it is mainly because Shoaib dataset contain device placement diversity and the learned representations contains general information about both activity and device placement.



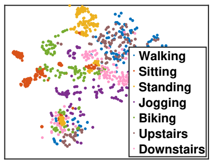
(a) HHAR, HAR



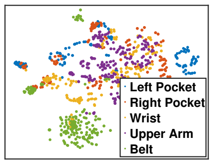
(b) UCI, HAR



(c) MotionSense, HAR



(d) Shoaib, HAR



(e) Shoaib, DPC

Figure 7: Representation visualization with t-SNE.

Table 4: Performance comparison of classifiers.

Application	HAR		HAR		HAR		HAR		DPC		Average	
Dataset	HHAR		UCI		MotionSense		Shoaib		Shoaib		-	
Metric	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1
LIMU-CNN	0.952	0.946	0.883	0.882	0.895	0.858	0.849	0.850	0.884	0.884	0.893	0.884
LIMU-ATTN	0.928	0.923	0.915	0.913	0.909	0.874	0.809	0.810	0.812	0.811	0.875	0.866
LIMU-LSTM	0.953	0.949	0.913	0.915	0.913	0.880	0.890	0.891	**0.921**	**0.921**	0.918	0.911
LIMU-GRU	**0.964**	**0.962**	**0.924**	**0.923**	**0.927**	**0.899**	**0.900**	**0.899**	0.920	**0.921**	**0.927**	**0.921**

4.4.2 Varying classifier. As we explained in Section 3.6, other than GRU, other commonly used neural network components can also be adopted following LIMU-BERT for task-classifier design. We investigate how 2D-CNN [18], Multi-head Attention [43], and Long Short-term Memory (LSTM) [10] as alternatives perform with the representations generated by LIMU-BERT. The detailed designs of them are provided in Section 3.6. The settings of LSTM are identical to those of GRU. The labeling rate and sequence length are 1% and 20, respectively. Table 4 compares the performances of the four varieties of classifiers on the four datasets and two applications. The GRU gives the highest performance, and we find that the other alternatives also achieve reasonably good performance. The results suggest that representations produced by LIMU-BERT can be beneficial to many neural network models for classification tasks. Their performances are higher than all previously proposed baseline models as shown in Table 2, which also justifies the reliability of learned representations.

4.4.3 Varying sensors. We examine how LIMU-BERT performs on different input sensor combinations. We try to feed only accelerometer readings, both accelerometer and gyroscope readings, and all IMU sensors (including magnetometer) into LIMU-BERT. The performances of corresponding LIMU-GRU are depicted in Figure 8, where A, G, and M denote the inclusion of accelerometer, gyroscope, and magnetometer, respectively. The F1-scores of all sensor combinations indicate that the performance gain introduced by extra gyroscope readings is significant, especially in the UCI dataset. The reason is that gyroscope sensor can provide more information about the motions of the mobile device. On the other hand, magnetometer readings do not bring much benefit in our target applications. The possible reasons include it is not as sensitive to device motions as other two sensors are and it may easily be affected by the environment. Magnetometer readings, nevertheless does do not degrade the performances of LIMU-GRU. In general, the results suggest that LIMU-BERT is able to work well on multiple IMU sensors.

4.4.4 Varying normalization method. We evaluate the effectiveness of our normalization method as proposed in Section 3.2. We compare with (1) feeding the raw IMU data into LIMU-BERT directly, as well as (2) adopting mean-variance method to normalize raw data. Figure 9 depicts their performances, where the results are grouped under different datasets and applications. We find that feeding raw data can achieve similar accuracies as those of our normalization method in the MotionSense and Shoaib dataset. However, its accuracy degrades significantly in other cases, such as in the HHAR-HAR and UCI-HAR, which suggests that the performances of raw data are not stable. For mean-variance normalization method, the classifiers cannot extract effective features from learned representations and thus accuracies are very low, which is in line with our hypothesis that any normalization method that destructs the distribution information of raw IMU data may lead to performance degradation. In contrast, the normalization method adopted in LIMU-BERT outperforms using raw data by 5.78% on average and achieves the highest overall accuracy. The result indicates the improved stability of LIMU-BERT normalization.

4.4.5 Varying masking approach. Figure 10 displays the comparative performances of different masking approaches including the approach adopted in original BERT (referred to as *single mask* in this evaluation) and span masking with different probability of success p as specified in Equation 4. The p in Equation 4 determines the expected value of the geometric distribution, which affects the overall masked subsequence length. The span mask with smaller p is more likely to mask longer subsequences. The results show that span masking obtains higher average accuracy than the single masking, which suggests that masking longer subsequence may be more effective for IMU representation learning. The setting of

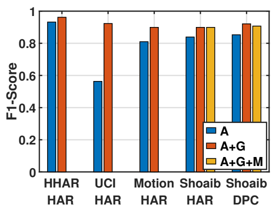


Figure 8: F1-score comparison of sensor combinations.

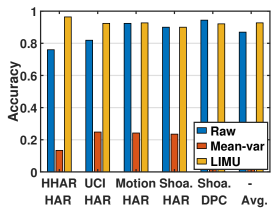


Figure 9: Accuracy comparison of normalization methods.

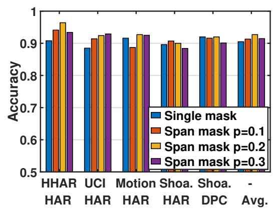


Figure 10: Accuracy comparison of different mask approaches.

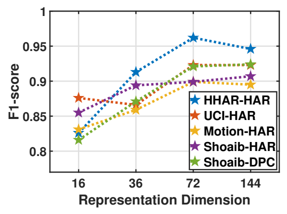


Figure 11: F1-score comparison of representation dimensions.

Table 5: Accuracies across datasets.

Source Dataset	Target Dataset and Application					
	HHAR HAR	UCI HAR	Motion HAR	Shoaib HAR	Shoaib DPC	Avg. -
HHAR	0.964	0.862	0.872	0.845	0.832	0.875
UCI	0.865	0.924	0.879	0.843	0.820	0.866
Motion	0.883	0.879	0.927	0.847	0.852	0.878
Shoaib	0.879	0.847	0.869	0.900	0.925	0.884
Merged	0.905	0.932	0.901	0.895	0.883	0.903

Table 6: Efficiency comparison.

Model	Parameters	Size	Train Time	Infer. Time
DCNN	17 K	77 KB	4 ms	6 ms
DeepSense	13 K	73 KB	8 ms	6 ms
TPN	105 K	501 KB	38+2 ms	6 ms
R-GRU	5 K	24 KB	4 ms	18 ms
LIMU-BERT*	189 K	766 KB	36 ms	18 ms
LIMU-BERT	62 K	255 KB	27 ms	14 ms
LIMU-GRU	9 K	39 KB	6 ms	18 ms

p = 0.2 achieves the best overall performances and we adopt it in LIMU-BERT.

4.4.6 Varying representation dimension. We study the impact of representation dimension H_{dim} on the final performance. Different values are applied and their F1-scores are illustrated in Figure 11. In general, it is clear that the F1-scores increase when we increase H_{dim} under 72. But F1-scores decrease if H_{dim} increases to 144. This is because H_{dim} is highly related to the complexities of LIMU-BERT and the GRU classifier. Larger representation dimension improves the goodness of fitting but may cause over-fitting, which is a typical trade-off. It also affects the efficiency of our models on mobile devices. We set H_{dim} to 72, which achieves best performances in the series of experiments. Due to the characteristics of IMU features, the dimensions here are much smaller than those (e.g., 1024) in the original BERT, which significantly reduces the model size.

4.4.7 Varying dataset. We examine how LIMU-BERT performs across different datasets, e.g., when the LIMU-BERT trained on the HHAR dataset is applied on the MotionSense dataset (abbreviated as Motion). We also create a *merged* dataset by combining the unlabeled data in the four datasets. A LIMU-BERT is trained on the merged dataset and then tested on each of the four datasets. Table 5 shows that the LIMU-BERT trained on any dataset achieves average accuracy higher than 0.850, which suggests that the features generated by LIMU-BERT are generalizable and can be transferred to varied datasets. The performances in the cross-dataset cases degrade due to the dataset diversity (i.e., user, device, and behavior diversity). The LIMU-BERT trained on the merged dataset achieves the best performance, which suggests that LIMU-BERT can better handle dataset diversity and extract general features when trained on more extensive unlabeled data. This further shows the good capability of LIMU-BERT in unleashing the potential of the abundant unlabeled data in IMU sensing applications.

4.4.8 Computation overhead. Table 6 compares the models in our framework with baseline models in terms of the number of parameters, model size, training time and inference time. The LIMU-BERT aims at learning a general representation from unlabeled data accumulated over time, and can thus be continuously trained in the cloud while the mobile devices only run the task-specific models for inference. Therefore, the training is performed on the the server (see in Section 4.1.4) and the training time estimates the time used to train one mini-batch (128 samples). The inference time is the execution time for inferring one sample (120 IMU readings) on a Samsung Galaxy S8 (SM-G9500 equipped with Octa-core CPU and 4 GB RAM). Each experiment is repeated 1000 times to obtain the average time. The input data are randomly sampled from the UCI dataset. The training time of TPN denotes the pre-training and supervised training time. LIMU-BERT* represents the the LIMU-BERT without cross-layer parameter sharing and the decoder. The total number of parameters in LIMU-BERT (H_{dim} = 72) is about 62,000, which is much smaller than that of the original BERT. The results indicate that cross-layer parameter sharing mechanism does help reduce the model size and improve the overall efficiency. On the other hand, GRU classifier has only about 9,000 parameters, which is very lightweight. Although LIMU-BERT generally incurs slightly more inference time than some other approaches, the overhead is comparable and affordable for most COTS smartphones.

5 RELATED WORK

Applying deep learning techniques with IMU sensors in mobile devices (i.e., smartphones, smartwatches, earphones, and etc.) facilitates many ubiquitous applications, such as human activity recognition [13, 23, 33, 51], human-computer interaction [24, 58], user authentication [2, 31, 48, 51], indoor and tracking [14, 41], and etc. Compared with vision-based [30], wireless-based [12, 21, 46, 52, 54, 57] systems, mobile-based [13, 23, 33, 47, 51] methods are more ubiquitous and achieve promising performances.

However, most models [13, 23, 41, 51, 52] are trained with a large amount of labeled samples and require great manual labeling efforts, which may not be practical in reality. MetaSense [8] employs meta-learning to make a base model adaptive to the target user domain with few labeled samples. But it still needs large labeled dataset (leave-one-user-out among ten users) to train a base model. EI [12] proposes a novel feature extraction framework, which extracts environment or subject independent features from both labeled and unlabeled data. Nevertheless, it needs multiple types of labels (i.e., activity labels and environment or subject labels), which imposes stronger labeling requirements. It is also challenging to balance different losses in the adversarial training process in [12]. Therefore, there exists a gap in designing a label-free representation learning method for IMU sensing applications.

As an emerging line of research topic, self-supervised learning approaches have been widely studied to reduce dependence on labeled data in deep learning. The self-supervised learning approaches do not use human-annotated datasets and automatically learn representations (e.g., visual features) from the target data. Many surrogate tasks [6, 15, 19, 20, 27, 29] have been proposed to learn spatial and temporal relations from images, videos, and text. These self-supervised learning paradigms have proved that extracting high-level representations tremendously help down-stream transfer and semi-supervised learning models. These approaches, however, may not apply to IMU sensing applications due to the multi-modality nature of IMU data and the constrained resources of mobile devices in accommodating complicated models.

To the best of our knowledge, TPN [33] is the only work that borrowed the idea of self-supervised learning and apply it to IMU sensing. It extracts the accelerometer representations by jointly learning to solve multiple self-supervised tasks. A temporal CNN is trained to to recognize several transformations (e.g., scale or rotate) applied on the raw data. In the supervised training phase, the features learned by the temporal CNN are utilized for HAR model. Nevertheless, TPN is designed to process only accelerometer readings and is thus limited in fully unleash the potential the multi-modality IMU sensors. According to our experiment result, the performance of TPN is far below what LIMU-BERT achieves.

This paper explores the feasibility of self-supervised representations learning for the IMU data collected at mobile devices. Different from TPN, LIMU-BERT is able to handle multiple sensor data thanks to its special design in normalization and fusion. LIMU-BERT targets at two types of features (i.e., distributions of individual measurements of IMU sensors and temporal relations in continuous measurements) and learn them by adaptive MLM self-supervised task. The learned representations significantly reduce the label requirements of the down-stream classifiers.

6 DISCUSSION AND FUTURE WORK

Model transferability. As shown in Table 5, the performance of LIMU-BERT slightly degrades when transferring across datasets. One main reason is that the four datasets are collected with diverse devices, placements, users, and environments. The diversities cause the domain shifts among the datasets and affect the generalizability of learned representations. However, the results in Table 5 also suggest that LIMU-BERT is able to cope with those diversities when the activity types of two datasets are similar. To mitigate the impact of domain shifts and extract more general features, LIMU-BERT might be further improved by techniques like denoised autoencoder [44] or data augmentation [37] and we leave it as a future work.
Irrelevant event detection. Being another popular IMU sensing application, the irrelevant or anomaly detection task has been recently concerned [22, 45]. As a typical generative self-supervised model, LIMU-BERT is sensitive to the rare samples and may fail to extract effective features from them. Therefore, the performance of LIMU-BERT in detecting irrelevant or anomaly events depends on the frequency of occurrence of the related events during the sensing process.
User privacy. In most IMU sensing scenarios, the sensor data are collected on mobile devices and supposed to be uploaded to the cloud for the self-supervised learning process, which may cause privacy issue. Emerging training frameworks like federated learning [17, 26] may be introduced in the training of LIMU-BERT for protecting the user privacy. An alternative method is to further reduce the size of the models so that we can directly train them on mobile phones without the need of data transmission.

Other future works include the investigation of how the representations learned by LIMU-BERT may facilitate other mobile applications, e.g., indoor localization [39] or device orientation estimation [59].

7 CONCLUSION

In this paper, we present a lite BERT-like representation learning model for mobile IMU sensor data, which makes use of unlabeled data and accordingly extracts generalizable features instead of task-specific features. Extensive experimental evaluation demonstrates the learned representations by LIMU-BERT can boost the performances of down-stream models significantly with few labeled samples. With LIMU-BERT, the labeling efforts in real IMU-based sensing applications can be greatly reduced.

ACKNOWLEDGMENTS

We sincerely thank the anonymous shepherd and reviewers for their insightful comments and suggestions. This research is supported by the National Research Foundation, Singapore under its Industry Alignment Fund – Pre-positioning (IAF-PP) Funding Initiative, Alibaba Group through Alibaba Innovative Research (AIR) Program and Alibaba-NTU Singapore Joint Research Institute (JRI), and NTU CoE SUG. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not reflect the views of National Research Foundation, Singapore, Alibaba Group, and NTU.

REFERENCES

[1] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. 2016. Layer Normalization. arXiv:1607.06450 [stat.ML]

[2] Cheng Bo, Lan Zhang, Xiang-Yang Li, Qiuyuan Huang, and Yu Wang. 2013. Silentsense: silent user identification via touch and movement behavioral biometrics. In *Proceedings of the 19th annual international conference on Mobile computing & networking*. 187–190.

[3] Wenqiang Chen, Lin Chen, Yandao Huang, Xinyu Zhang, Lu Wang, Rukhsana Ruby, and Kaishun Wu. 2019. Taprint: Secure text input for commodity smart wristbands. In *The 25th Annual International Conference on Mobile Computing and Networking*. 1–16.

[4] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555* (2014).

[5] Zhi-An Deng, Guofeng Wang, Ying Hu, and Di Wu. 2015. Heading estimation for indoor pedestrian navigation using a smartphone in the pocket. *Sensors* 15, 9 (2015), 21518–21536.

[6] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805* (2018).

[7] Basura Fernando, Hakan Bilen, Efstratios Gavves, and Stephen Gould. 2017. Self-supervised video representation learning with odd-one-out networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 3636–3645.

[8] Taesik Gong, Yeonsu Kim, Jinwoo Shin, and Sung-Ju Lee. 2019. Metasense: few-shot adaptation to untrained conditions in deep mobile sensing. In *Proceedings of the 17th Conference on Embedded Networked Sensor Systems*. 110–123.

[9] Dan Hendrycks and Kevin Gimpel. 2016. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415* (2016).

[10] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780.

[11] Nathalie Japkowicz and Shaju Stephen. 2002. The class imbalance problem: A systematic study. *Intelligent data analysis* 6, 5 (2002), 429–449.

[12] Wenjun Jiang, Chenglin Miao, Fenglong Ma, Shuochao Yao, Yaqing Wang, Ye Yuan, Hongfei Xue, Chen Song, Xin Ma, Dimitrios Koutsonikolas, et al. 2018. Towards environment independent device free human activity recognition. In *Proceedings of the 24th Annual International Conference on Mobile Computing and Networking*. 289–304.

[13] Wenchao Jiang and Zhaozheng Yin. 2015. Human activity recognition using wearable sensors by deep convolutional neural networks. In *Proceedings of the 23rd ACM international conference on Multimedia*. 1307–1310.

[14] Yonghang Jiang, Zhenjiang Li, and Jianping Wang. 2018. Ptrack: Enhancing the applicability of pedestrian tracking with wearables. *IEEE Transactions on Mobile Computing* 18, 2 (2018), 431–443.

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

[16] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[17] Jakub Konečný, H Brendan McMahan, Felix X Yu, Peter Richtárik, Ananda Theertha Suresh, and Dave Bacon. 2016. Federated learning: Strategies for improving communication efficiency. *arXiv preprint arXiv:1610.05492* (2016).

[18] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems* 25 (2012), 1097–1105.

[19] Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2019. Albert: A lite bert for self-supervised learning of language representations. *arXiv preprint arXiv:1909.11942* (2019).

[20] Hsin-Ying Lee, Jia-Bin Huang, Maneesh Singh, and Ming-Hsuan Yang. 2017. Unsupervised representation learning by sorting sequences. In *Proceedings of the IEEE International Conference on Computer Vision*. 667–676.

[21] Xinyu Li, Yanyi Zhang, Ivan Marsic, Aleksandra Sarcevic, and Randall S Burd. 2016. Deep learning for rfid-based activity recognition. In *Proceedings of the 14th ACM Conference on Embedded Network Sensor Systems CD-ROM*. 164–175.

[22] Jian Liu, Hongbo Liu, Yingying Chen, Yan Wang, and Chen Wang. 2019. Wireless sensing for human activity: A survey. *IEEE Communications Surveys & Tutorials* 22, 3 (2019), 1629–1645.

[23] Shengzhong Liu, Shuochao Yao, Jinyang Li, Dongxin Liu, Tianshi Wang, Huajie Shao, and Tarek Abdelzaher. 2020. GIobalFusion: A Global Attentional Deep Learning Framework for Multisensor Information Fusion. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 4, 1 (2020), 1–27.

[24] Yang Liu, Zhenjiang Li, Zhidan Liu, and Kaishun Wu. 2019. Real-time arm skeleton tracking and gesture inference tolerant to missing wearable sensors. In *Proceedings of the 17th Annual International Conference on Mobile Systems, Applications, and Services*. 287–299.

[25] Mohammad Malekzadeh, Richard G Clegg, Andrea Cavallaro, and Hamed Haddadi. 2019. Mobile sensor data anonymization. In *Proceedings of the international conference on internet of things design and implementation*. 49–58.

[26] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. 2017. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*. PMLR, 1273–1282.

[27] Ishan Misra, C Lawrence Zitnick, and Martial Hebert. 2016. Shuffle and learn: unsupervised learning using temporal order verification. In *European Conference on Computer Vision*. Springer, 527–544.

[28] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. 2017. Automatic differentiation in pytorch. (2017).

[29] Deepak Pathak, Pulkit Agrawal, Alexei A Efros, and Trevor Darrell. 2017. Curiosity-driven exploration by self-supervised prediction. In *International Conference on Machine Learning*. PMLR, 2778–2787.

[30] Ronald Poppe. 2010. A survey on vision-based human action recognition. *Image and vision computing* 28, 6 (2010), 976–990.

[31] Zhen Qin, Lingzhou Hu, Ning Zhang, Dajiang Chen, Kuan Zhang, Zhiguang Qin, and Kim-Kwang Raymond Choo. 2019. Learning-aided user identification using smartphone sensors for smart homes. *IEEE Internet of Things Journal* 6, 5 (2019), 7760–7772.

[32] Jorge-L Reyes-Ortiz, Luca Oneto, Albert Samà, Xavier Parra, and Davide Anguita. 2016. Transition-aware human activity recognition using smartphones. *Neurocomputing* 171 (2016), 754–767.

[33] Aaqib Saeed, Tanir Ozcelebi, and Johan Lukkien. 2019. Multi-task self-supervised learning for human activity detection. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 3, 2 (2019), 1–30.

[34] Sheng Shen, Mahanth Gowda, and Romit Roy Choudhury. 2018. Closing the gaps in inertial motion tracking. In *Proceedings of the 24th Annual International Conference on Mobile Computing and Networking*. 429–444.

[35] Sheng Shen, He Wang, and Romit Roy Choudhury. 2016. I am a smartwatch and i can track my user's arm. In *Proceedings of the 14th annual international conference on Mobile systems, applications, and services*. 85–96.

[36] Muhammad Shoaib, Stephan Bosch, Ozlem Durmaz Incel, Hans Scholten, and Paul JM Havinga. 2014. Fusion of smartphone motion sensors for physical activity recognition. *Sensors* 14, 6 (2014), 10146–10176.

[37] Connor Shorten and Taghi M Khoshgoftaar. 2019. A survey on image data augmentation for deep learning. *Journal of Big Data* 6, 1 (2019), 1–48.

[38] Yuanchao Shu, Cheng Bo, Guobin Shen, Chunshui Zhao, Liqun Li, and Feng Zhao. 2015. Magicol: Indoor localization using pervasive magnetic field and opportunistic WiFi sensing. *IEEE Journal on Selected Areas in Communications* 33, 7 (2015), 1443–1457.

[39] Yuanchao Shu, Kang G Shin, Tian He, and Jiming Chen. 2015. Last-mile navigation using smartphones. In *Proceedings of the 21st annual international conference on mobile computing and networking*. 512–524.

[40] Allan Stisen, Henrik Blunck, Sourav Bhattacharya, Thor Siiger Prentow, Mikkel Baun Kjærgaard, Anind Dey, Tobias Sonne, and Mads Møller Jensen. 2015. Smart devices are different: Assessing and mitigatingmobile sensing heterogeneities for activity recognition. In *Proceedings of the 13th ACM conference on embedded networked sensor systems*. 127–140.

[41] Scott Sun, Dennis Melamed, and Kris Kitani. 2021. IDOL: Inertial Deep Orientation-Estimation and Localization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 35. 6128–6137.

[42] Laurens Van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *Journal of machine learning research* 9, 11 (2008).

[43] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *arXiv preprint arXiv:1706.03762* (2017).

[44] Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. 2008. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th international conference on Machine learning*. 1096–1103.

[45] Hao Wang, Daqing Zhang, Yasha Wang, Junyi Ma, Yuxiang Wang, and Shengjie Li. 2016. RT-Fall: A real-time and contactless fall detection system with commodity WiFi devices. *IEEE Transactions on Mobile Computing* 16, 2 (2016), 511–526.

[46] Wei Wang, Alex X Liu, Muhammad Shahzad, Kang Ling, and Sanglu Lu. 2015. Understanding and modeling of wifi signal based human activity recognition. In *Proceedings of the 21st annual international conference on mobile computing and networking*. 65–76.

[47] Tianzhang Xing, Qing Wang, Chase Q. Wu, Wei Xi, and Xiaojiang Chen. 2020. DWatch: A Reliable and Low-Power Drowsiness Detection System for Drivers Based on Mobile Devices. *ACM Trans. Sen. Netw.* 16, 4, Article 37 (Sept. 2020), 22 pages.

[48] Xiangyu Xu, Jiadi Yu, Yingying Chen, Qin Hua, Yanmin Zhu, Yi-Chao Chen, and Minglu Li. 2020. TouchPass: towards behavior-irrelevant on-touch user authentication on smartphones leveraging vibrations. In *Proceedings of the 26th Annual International Conference on Mobile Computing and Networking*. 1–13.

[49] Jianbo Yang, Minh Nhut Nguyen, Phyo Phyo San, Xiaoli Li, and Shonali Krishnaswamy. 2015. Deep convolutional neural networks on multichannel time series for human activity recognition.. In *Ijcai*, Vol. 15. Buenos Aires, Argentina, 3995–4001.

[50] Zhijian Yang, Yu-Lin Wei, Sheng Shen, and Romit Roy Choudhury. 2020. Ear-AR: indoor acoustic augmented reality on earphones. In *Proceedings of the 26th Annual International Conference on Mobile Computing and Networking*. 1–14.

[51] Shuochao Yao, Shaohan Hu, Yiran Zhao, Aston Zhang, and Tarek Abdelzaher. 2017. Deepsense: A unified deep learning framework for time-series mobile sensing data processing. In *Proceedings of the 26th International Conference on World Wide Web*. 351–360.

[52] Yinggang Yu, Dong Wang, Run Zhao, and Qian Zhang. 2019. RFID based real-time recognition of ongoing gesture with adversarial learning. In *Proceedings of the 17th Conference on Embedded Networked Sensor Systems*. 298–310.

[53] Xiaohua Zhai, Avital Oliver, Alexander Kolesnikov, and Lucas Beyer. 2019. S4l: Self-supervised semi-supervised learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 1476–1485.

[54] Yi Zhang, Zheng Yang, Guidong Zhang, Chenshu Wu, and Li Zhang. 2021. XGest: Enabling Cross-Label Gesture Recognition with RF Signals. *ACM Trans. Sen. Netw.* 17, 4, Article 37 (Sept. 2021), 23 pages.

[55] Yi Zhao, Zimu Zhou, Wang Xu, Tongtong Liu, and Zheng Yang. 2020. Urban Scale Trade Area Characterization for Commercial Districts with Cellular Footprints. *ACM Trans. Sen. Netw.* 16, 4, Article 42 (Sept. 2020), 20 pages.

[56] Yuanqing Zheng, Guobin Shen, Liqun Li, Chunshui Zhao, Mo Li, and Feng Zhao. 2017. Travi-navi: Self-deployable indoor navigation system. *IEEE/ACM transactions on networking* 25, 5 (2017), 2655–2669.

[57] Yue Zheng, Yi Zhang, Kun Qian, Guidong Zhang, Yunhao Liu, Chenshu Wu, and Zheng Yang. 2019. Zero-effort cross-domain gesture recognition with Wi-Fi. In *Proceedings of the 17th Annual International Conference on Mobile Systems, Applications, and Services*. 313–325.

[58] Han Zhou, Yi Gao, Xinyi Song, Wenxin Liu, and Wei Dong. 2019. LimbMotion: Decimeter-level Limb Tracking for Wearable-based Human-Computer Interaction. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 3, 4 (2019), 1–24.

[59] Pengfei Zhou, Mo Li, and Guobin Shen. 2014. Use it free: Instantly knowing your phone attitude. In *Proceedings of the 20th annual international conference on Mobile computing and networking*. 605–616.

[60] Pengfei Zhou, Yuanqing Zheng, and Mo Li. 2012. How long to wait? Predicting bus arrival time with mobile phone based participatory sensing. In *Proceedings of the 10th international conference on Mobile systems, applications, and services*. 379–392.