RGC: Reliable Gesture Classification via Wearables Using GANs-Based Data Augmentation

Han Zhou[1], Ji Zhao[2], Yi Gao[1], Wei Dong[1]
[1] Zhejiang University, China and Alibaba-Zhejiang University Joint Institute of Frontier Technologies, China
[2]Shanghai Pudong Development Bank Co. ,Ltd. Shanghai Branch
Email: {zhouh,gaoy,dongw}@emnets.org, zhaoj02@spdb.com.cn

2021 IEEE 27th International Conference on Parallel and Distributed Systems (ICPADS) | 978-1-6654-0878-3/21/$31.00 ©2021 IEEE | DOI: 10.1109/ICPADS53394.2021.00114

***Abstract*—Gesture-related human-computer interaction systems have been developed with different purposes. Wearable-based gesture recognition is well studied and considered to be effective. However, unexpected body movement of a user, such as walking and turning, is one issue that affects the robustness of classification. Collecting a sufficient dataset for every unexpected movement would be time-consuming and labor-intensive. To reduce the burden on users and address the lack of training data, we proposed RGC, which is a framework for IMU data augmentation. RGC adopts Generative-Adversarial-Networks-based and rotation-based augmentation to enhance the diversity of the dataset, which helps the classifier learn more effective representations for gesture recognition. We collected a dataset of 21120 arm gesture samples under different kinds of unexpected movements to evaluate RGC. The experiments showed that our method provides an 8.8% to 1.1% accuracy improvement for different SOTA classifiers with different original dataset sizes.**

***Index Terms*—Gesture classification, Data augmentation, Generative Adversarial Networks**

I. Introduction

Designing gesture-related human-computer interaction (HCI) systems require accurate and robust classification of gestures. Among multiple sensing techniques that have been studied in this field, vision [1], wearable [2] and RF signal [3] show the most feasibility and usage. Vision-based and RF-based methods are user-friendly since they work in a non-contact manner. However, vision-based solutions have disadvantages of inability to work at non-line-of-sight scenes and insufficient mobility [1]. RF-based solutions require dedicated devices and are not robust to environmental changes [3]. Compared with them, wearable-based solutions provide mobile and low-cost use.

Among recent studies, different sensors such as IMU and PPG sensor [4], [5] are attached to a user's body for capturing the signal induced by her/his body movement. One problem that has not been solved is the noise via a user's unexpected body movements, e.g., walking, running or turning. This situation usually occurs in practice, e.g., a user moves the controller to give a command while her/he is walking in a VR game. Assuming that the classifier is trained on the clean gesture dataset (without any unexpected movements), the model accuracy will drop a lot when inferencing on noisy gestures (with unexpected movements). To solve this problem, we need to collect an abundant dataset for noisy gestures. However, collecting such a dataset is time-consuming and labor-intensive since the noise is diverse. Therefore, the challenge we face is to train a reliable classifier with insufficient data.

Data augmentation alleviated this problem by producing more data under certain immutability and let the classifier learn more effective representations from data. However, there is no mature method for the augmentation of IMU data.

In this paper, we explored the potential of using GANs [6] for IMU data augmentation and further training reliable classifiers. We modified the design of DCGAN [7] to let it supports 1D data generation, and use modified total variation loss to ensure the sample quality. We trained the model with condition input [8], which makes the category of the synthesizing samples under control.

We collected a dataset contains ten gesture classes in six conditions (non-unexpected-movement, walk, turn left/right, stand up and sit down) from 16 volunteers via three smartwatches. We evaluated our augmentation framework with three classification models. Results show that the model accuracy increased by 8.8% to 1.1% in different settings. We found that the smaller the original dataset is, the more accuracy gain will be achieved, and the augmentation enhances models based on RNN structure more.

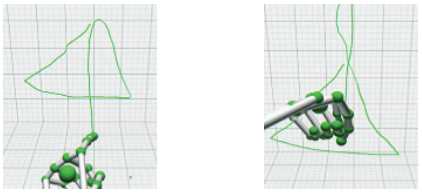
We summarize the contributions of this paper as follows:

- To tackle the problem of insufficient training samples, we proposed RGC, which is a framework for training gesture classification model with the augmented IMU dataset. The data augmentation is achieved by a cGAN [8].
- We collected an IMU dataset of 21120 arm gesture samples to evaluate RGC. Results show that RGC achieves an 8.8% to 1.1% accuracy improvement for different classifiers with different original dataset sizes.

The rest of this paper is organized as follows. Section II introduced the motivation of this paper. Section III describes the details of techniques in RGC including the GANs' training and data augmentation. Section IV shows the evaluation setup and results. Section V introduces the related work. Finally, Section VI concludes this paper.

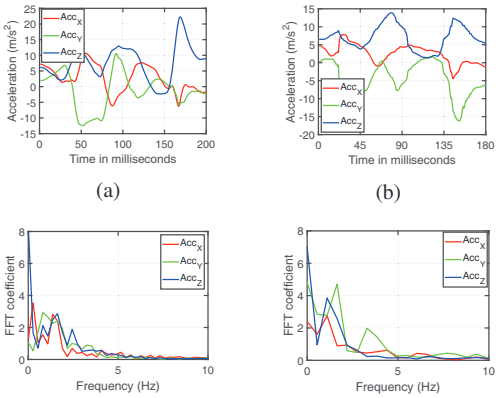
We thank all the reviewers for their valuable comments and helpful suggestions.This work is supported by the National Science Foundation of China under Grant No. 62072396 and 61872437, and Zhejiang Provincial Natural Science Foundation for Distinguished Young Scholars under No.LR19F020001. Wei Dong is the corresponding author.

2690-5965/21/$31.00 ©2021 IEEE
DOI 10.1109/ICPADS53394.2021.00114



(a) (b)

Fig. 1: Index finger trajectories (a) with none body movement, (b) when standing up. The trajectory was tracked by LeapMotion.



(a) (b) (c) (d)

Fig. 2: IMU raw data for the gesture "writing digit 4" without any unexpected movements (a), with walking induced noise (b). (c) and (d) stands for data after time-frequency transformation. The IMU readings are sampled at 100Hz, for the ease of watching we omit the 10-40Hz part which are also tiny values.

II. Motivation

We think IMU noise caused by the user's unexpected body movement is a key challenge when putting the lab-trained model into practical use. Existing studies usually collect gesture samples in a fixed scenario, where the user is standing or sitting steadily. Then, they use machine learning methods to train a classifier that would recognize the characteristics of certain gestures. However, as we mentioned, a user may involve unexpected movements such as walking or turning while performing the gesture. We believe that the impact of this noise is 1) unexpected movements induce signal changes and then the IMU data is difficult to recognize; 2) a user may perform the gesture in a more biased manner when involving other movements. For example, to maintain balance a user will slow the gesture while making a step ahead. Fig 1 displays the index finger trajectories of writing digit '4' from one user. We can see the shape in Fig 1(a) is reasonable because the user was sitting still, but the gesture in Fig 1(b), which was performed while the user was standing up, is not shaped like '4' at all. Fig 2 shows the raw acceleration in time and frequency domains of the same gesture writing digit '4', Fig 2(a) and (c)

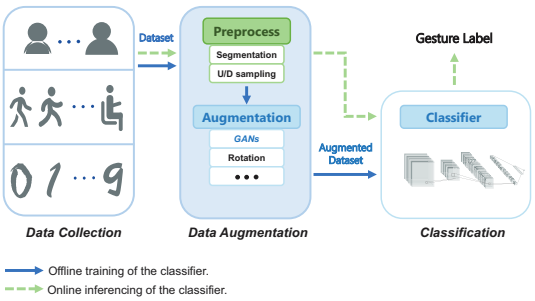


Fig. 3: Framework overview of RGC. In the training stage, we collect samples of different user performing different gestures involving different unexpected movement, and then use GANs to generate an augmented dataset, finally we train the classifier with the augmented dataset. In the inferencing stage, we use the pre-trained classifier to recognize the gesture.

stand for the clean sample, (b) and (d) for the noisy examples with walking-induced noise. Once again, we see very different shapes and values in the time and frequency domain signal.

To evaluate the effect of noise caused by unexpected movements, we conducted classification experiments of the noisy gestures. The gestures contain writing digits '0' to '9'. A state-of-the-art classifier Deepsense [9] was adopted for recognition. Results showed a 19% absolute accuracy drop when introducing noise of walking, from 95% to 76%. Besides, the noise of standing up caused a 32% accuracy drop, sitting down caused 24%, turning left and right caused 28% and 29%.

In the literature, filtering is considered an effective method to prevent noise [4], [5]. However, it is not suitable for this problem since the noise and signals are distributed in the same frequency band. In Fig 2 (c) and (d), we can see the FFT coefficients approach zero when the frequency is above 5Hz, which means regular arm gestures and unexpected body movements are mainly composed of such low-frequency components.

To address this practical problem, people need to further expand the collection of datasets. We use N_{base} to represent the base sample number we collect for each user each gesture, N_{user} and N_{ges} to the number of involved users and gestures. If we used to collect $N_{base} * N_{user} * N_{ges}$ samples to satisfy the classifier training requirement, now we need to collect $N_{base} * N_{user} * N_{ges} * N_{unexpected}$ samples, where $N_{unexpected}$ represent the number of classes of unexpected movements. Collecting such a dataset requires more time and participation of the user than before. In order to reduce the burden on users, we explore the potential to use GANs-based methods to augment the dataset.

III. Methodology

A brief overview of the proposed RGC framework is shown in Fig 3 , RGC consists of three components: data collection, data augmentation and classification component. In section,

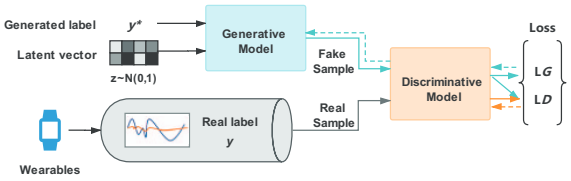


Fig. 4: Workflow of the contional GAN.

we will describe the techniques in detail including the pre-process, cGAN structure and training.

A. Pre-processing

1) Data segmentation: Before training any model, we identify the IMU data segment that corresponding to the gesture. To acquire the exact start and end of the gesture, we use a threshold-based method. If the arm stays still before the gesture starts, the accelerometer readings should submit to local gravitational acceleration, and the gyroscope readings should be zero. We choose a window w to slide over the continuous data. Mean and standard deviation are computed for the acceleration and angular velocity norm separately. Every time the window slide one point forward, and we judge the latest point p as a 'still' point if

$$\begin{aligned} &\mid \overline{w} - [G, 0] \mid < T_1, \\ &\sigma(w) - [\sigma_{acc}, \sigma_{gyro}] < T_2, \\ &p - [G, 0] < T_3. \end{aligned} \tag{1}$$

We use G to represent the norm of gravitional acceleration ($9.8m/s^2$), and $\sigma_{acc}, \sigma_{gyro}$ stands for the standard deviation of the sensor random noise. If we find that

$$\mid \overline{w} - [G, 0] \mid > T_4, \tag{2}$$

it means the gesture has begun. We go back to pick the latest 'still' point as the start of the gesture. Because we track the still points using a more strict threshold, we could get a more accurate start point from still to moving compared with methods which usually use a loose threshold to track the moving point. This method helps us find a more complete trace of the gesture. After the gesture begins, the next still point we find means the gesture has ended. We use the index of this still point minus half of the window as the end index of the gesture.

The above method works well for segmenting the gestures without unexpected movement, but for the others not. We find that each axis of sensors is affected to varying degrees, dropping and picking certain axes would still let us get an accurate segmentation. Using gestures with walking noise as an example, we observe that acceleration data of all axes are too noisy to find the start and end. However, the gyroscope readings only suffer in Z-axis which is induced by arm swinging. We can use Y and Z-axis readings of the gyroscope to achieve segmentation.

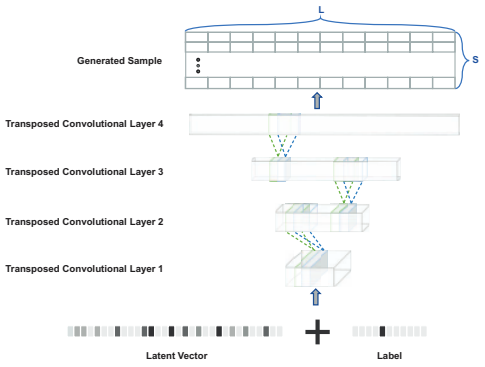


Fig. 5: Generative model structure of cGAN for IMU data generation.

B. cGAN-Based Data Augmentation

1) **cGAN workflow:** An overview of the cGAN workflow is shown in Fig 4. Original GANs [6] were designed to learn the mapping from a distribution Z (usually a Gaussian distribution) to another distribution X, which belongs to the training dataset (e.g., images, audio, etc.). Generally, GANs consist of a generative model (generator) and a discriminative model (discriminator). The generative model is trained to generate fake samples which are hard to be distinguished from real data, while the discriminative model is trained to determine whether a sample is real or synthesized. The way we train GANs is what so-called adversarial training. Specifically, during training the discriminative model tries to classify the input from real data as real, and the input from the generative model as fake. On the contrary, the generative model tries to fool the discriminative model by generating more real samples.

The original GANs show a great ability to synthesize data, but it is hard to be directly applied in data augmentation since we prefer to generate labeled data. cGAN [8] was proposed to control the generation process. As shown in 4, the input of the generative model and discriminative model both contain the label of the data. The discriminative model now distinguishes 1) if the data is real or generated; 2) if the data matches the label. The generative model also needs to synthesize data which matches the label. After the training of cGAN, we can control the classes of generated samples by feeding specific labels to the generative model.

2) ***Model input****:* The discriminative model accepts the IMU data and the label as input. After pre-processing, we use up-sampling and down-sampling via 1D signal interpolation in the time domain to make the signal have the same length L. The IMU signal becomes an $S \times L$ tensor where S represents the sensor dimension. For example, S is six if we use both 3-axes accelerometer and gyroscope. The label is a one-hot vector of length N_ges. For the generative model, the input is a latent vector z sampled from the Gaussian distribution Z and the label. The vector is of length n and with random values in (0,1). The label is also the one-hot vector as described above.

*3) **Generative model**:* The generative model tries to map low-dimensional latent vectors z to high-dimensional sequential IMU data $\hat{x} = G(z)$. We choose the generative model design with transposed convolutional structured [10] as basic blocks.

We tried to use the 2D transposed convolutional structure design for generating IMU data, but the result is not good. The reason could be 1) this structure is more sensitive for image-like 2D inputs, the $S \times L$ IMU data has no effective shape features to learn from; 2) at the end of the generation, data of each IMU axis should be processed separately, or it's hard for the network to perfect details of generation. This is not like when we try to extract features from IMU data for classification, where we use 2D and even 3D convolutional structures to capture the inside time correlation. Therefore, we choose 1D transposed convolutional structure since it could prevent the issues we mentioned above.

In Fig 5 we plot the main architecture of the generative model. We first concatenate the input random latent vector and label, then it passes four sequential 1D transposed convolutional layers since we treat the IMU signal as the multi-channel 1D sequential signal. Specifically, the signal at every axis of every sensor, e.g., the signal of the accelerometer's X-axis, represents the one channel 1D sequential signal. The large kernel and stride increase the perception field of the transposed convolutional block, and let the model learn the long-range correlations inside in signal. The long-range correlation is more useful since it represents the complete characteristics of the gesture and potential unexpected movement. We reduced the kernels into S at the last transposed convolutional layer and set the length of the output equals to L. That is, we get an output of shape $S \times L$ that will be treated as the data of a gesture sample. For each but not the last transposed convolution layer, we use a rectified linear unit (ReLU) [11] as the activation function and a Dropout layer [12] to enhance randomness.

*4) **Discriminative model**:* The discriminative model tells if a sample is a real gesture or synthesized from the generator, and what category the sample belongs to. It requires the discriminative model to have an ability to extract effective features. Therefore we inherit the design ideas from the SOTA human activity recognition classifier [9].

Fig 6 shows the structure of the discriminator. We employed 1D, 2D and 3D convolutional blocks to extract features from the IMU data. Followed by each convolutional layer, we use a ReLU [11] as the activation function and a Dropout Layer [12] to enhance randomness. As stated in [13], we do not use any batch normalization [14] in the discriminative model. Besides, the input label vector is fed to a fully connected layer for embedding. We concatenate these two parts and then use another fully connected layer to finally output $D(x)$. $D(x)$ is a vector of length $1 + N_{ges}$, where the first digit represents the probability that the sample is real, the remaining digits represent the probability that the sample belongs to a certain gesture category.

*5) **cGAN Training**:* The training and gradient backpropagation process of GAN are illustrated in Fig 6. we train our model using the wGAN-GP loss [13] which measures the Wasserstein Distance (or called Earth-Mover Distance) as

$$W(P_X, P_Z) = \frac{1}{K} \sup_{\|f\|_L \leq K} E_{x \sim P_X}[f(x)] - E_{x \sim P_Z}[f(x)] \quad (3)$$

where P_X and P_Z represent two distributions and f represents a 1-Lipschitz function. To minimize the Wasserstein Distance, the loss function for the discriminative model and generative model are

$$\begin{aligned} L_D &= -E_{x \sim P_X}[D(x)] + E_{x \sim P_Z}[D(x)] + \lambda Penalty(\nabla D(x)), \\ L_G &= -E_{x \sim P_Z}[D(x)]. \end{aligned} \quad (4)$$

In experiments, even when the Wasserstein Distance decreases, the generated sample is not ideal in terms of appearance. We found that the generated samples show distinct high-frequency noise. We think the cGAN tends to learn global features while relaxes its learning of local features. We could not choose a low pass filter to process the data since the up/down sampling changes the original frequency distribution of the signal. Therefore, we add a new loss function to prevent glitches. We choose the *Total Variation* (TV) loss which is well applied in image denoise [15]. The TV of an image is defined as

$$TV = \sum_{i,j} ((x_{i,j+1} - x_{i,j})^2 + (x_{i+1,j} - x_{i,j})^2)^{\frac{\beta}{2}} \quad (5)$$

where i, j are the pixel indexes and β is a hyperparameter. In our case, the TV of IMU signal should be defined as

$$TV_{IMU} = \sum_s \sum_l (x_{l+1}^s - x_l^s)^\beta \quad (6)$$

where s represents the sensor axis and l represents the time index. We use the TV loss to punish the generative model if its output has high TV, and the generative model loss function now becomes

$$L_G = -E_{x \sim P_Z}[D(x)] + \gamma E_{x \sim P_Z}(TV(x)). \quad (7)$$

where γ is a hyperparameter to control the value of TV loss.

C. Rotation-Based Data Augmentation

We also adopted manual rotation to the IMU signal for data augmentation. This idea is inspired by SHOW [16] which rotates IMU signal when recognizing the user's arm gestures for alphabet input. SHOW [16] only considers manual rotation in Y-axis of the smartwatch to prevent the IMU signal bias caused by the improper wearing position of the user. We think more rotation conditions should be considered. When the user's arm is freely moved in the space, the arm could have plenty orientation choices for the pre-defined gesture. We manually rotated the collected sample a (x, y, z) degrees for X, Y and Z-axis, respectively. We could get the rotation matrix S from (x, y, z). The augmentation for the sample is

$$\begin{aligned} Acc_r &= S * Acc, \\ Gyro_r &= S * Gyro. \end{aligned} \quad (8)$$

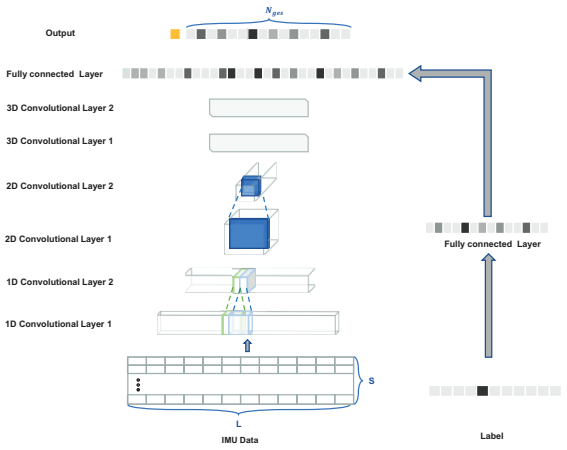


Fig. 6: Discriminative model structure of cGAN.

Acc represents the collected accelerometer readings and $Gyro$ represents the gyroscope readings. The choice of (x, y, z) are submitted to Gaussian distribution $\mathcal{N} \sim (0, \sigma)$, we used ten degrees for σ in RGC. The rotation-based augmentation is used during the training of the classifier, every time a training sample is used, we put a random rotation S on it.

IV. Evaluation

A. Experiment Setup

We employed 16 volunteers to collect gesture data, including 4 females and 12 males. All of the volunteers are university students. In our experiments, we use arm gesture recognition as an example to evaluate RGC. Therefore, we let the volunteers wear a smartwatch on her/his wrist to collect the IMU data of the arm gesture. Because all of the volunteers are right-handed, they chose to wear the smartwatch on the left wrist as most right-handed people do. One LG smartwatch and two Huawei Watch 2 were used for data collection. We developed an Android App for the smartwatch that can help the volunteers complete data collection independently. When the App is activated for recording gestures, it continuously records the 6-axis IMU data (accelerometer and gyroscope readings) at a 100 Hz sampling frequency.

We chose a set of gestures, which are writing digits 0 to 9 in the air, to evaluate RGC. We chose these gestures because they have certain representativeness and practical significance. Existing studies have tried to use wearables to recognize text input (the alphabet) and sign language [4], [16], we think that the digit is also an effective and common representation for controlling and communication. We considered five kinds of common and representative unexpected movements in our experiments, which are walking, turning left/right, standing up and sitting down. Fig 7 displays the setup.

The volunteers were first asked to perform the gesture when she/he is standing/sitting still, which means no unexpected movement, and the digit was repeated at least 20 times. Then

(a) (b) (c) (d) (e)

Fig. 7: Experiment setup of RGC. We collect gesture samples when the volunteer is (a)walking (b)turning left (c)turning right (d)standing up (e)sitting down and none movement.

the volunteers were asked to perform the gestures under different unexpected movements, each digit under each unexpected movement was repeated also at least 20 times. In addition, we collect the IMU data at volunteers' daily life, which represents the null gesture class. We randomly sample some segments from the data and added them to the dataset. Finally we got $16 * 11 * 6 * 20 = 21120$ gesture samples.

First, we separated the dataset into five parts equally according to the gesture, user and unexpected movement, which means every part has the same composition. Then, every time, we chose two parts as the training set, while the rest three parts as the testing set, we used RGC's method to augment the training set. After that, we trained the classifier on the original dataset and the augmented dataset to evaluate the performance gain of RGC. We did it $C_5^2 = 10$ times for cross-validation.

Since we also wanted to study the effect of the training set size. We picked different portions of the training set as the new training set while removing the rest. We achieved augmentation based on this reduced training set and then evaluated the classifier trained on such a dataset. For a certain training set size, we also did it 10 times for cross-validation.

The cGAN was trained on the training set for 2000 epochs with different batch sizes according to the dataset size. After training, we let the generative model generates $k * N$ samples where N is the original dataset size and k is a variable which is according to the original training set size. If the original training set is small, we generated more samples. As for the training of the classifier, we trained it for 400 epochs over the training set. Because the augmented dataset size N_{aug} is bigger than that of the original dataset, which is N, training of same epochs means that different total amount of data passing through the two classifiers. To maintain fairness of training, we assumed N is the dataset size of the augmented dataset and switched to the next dataset after every epoch.

B. Classifiers choices

DeepSense. As stated in [9], DeepSense performs well in HAR tasks. It divides the input sensing data into chunks, then uses CNNs and RNNs to extract features from the signal. The original DeepSense takes both the raw data and Discrete Fourier transform (DFT) coefficients as input, and here we omitted the frequency domain parts since RGC only achieves augmentation for the raw data.

BiLSTM. Model of RNN architecture shows a strong ability to learn representations from sequential data. According to [4],

the authors used a three-layer-bidirectional-LSTM structure for sign language recognition via wearables and made great progress. We also built such a three-layer-bidirectional-LSTM in the evaluation of RGC.

ResNet-IMU. CNN structure is well applied in image-related tasks, and it also shows an ability to classify IMU data [9], [17], [18]. We implemented the ResNet-IMU using residual blocks [19]. The model is composed of 1) two layers of 1D residual blocks; 2) two layers of 2D residual blocks; 3) two layers of 3D residual blocks; 4) a fully connected layer.

Hyperparameters of the three classifiers are determined by a random hyperparameter searching method.

C. Effectiveness

In this section, we evaluated the performance of the proposed methods. First, we evaluate how much performance gain is provided by our data augmentation, especially when the original dataset size is different. Then we study the effect of RGC on different classifiers. We discussed the model accuracy over different gestures. We also evaluated the generality of the trained classifier via leave-one-out cross-validation. We watched the generated gesture samples to see if mode collapse occurs.

1) Accuracy gain.: Here we evaluated the accuracy improvement of RGC. We used Deepsense [9] model as our classifier. As mentioned in Sec IV-A, every time, we chose 40% of the dataset as the original training set and the rest 60% as the test set. The cGAN was trained based on the training set and the generated samples are added to the dataset to train the classifier.

Since augmentation methods usually bring obvious benefits when the original dataset is small [20], we set the original training set size as a variable to see the impact. For a gesture of a certain gesture class, a certain user and a certain unexpected movement, we call it a *base sample* which contains information of a triad $[gesture, user, unexpectedmovement]$. We have $11 * 16 * 6 = 1056$ kinds of base samples, and there are 20 samples of each base sample in the whole dataset, while 8 samples in the training set. Every time, for each certain base sample property, which means a certain $[gesture, user, unexpectedmovement]$, we took $m = 1, 2, 4, 8$ samples from the training set to form the new training set. For example, when $m = 1$, the training set size is $1156 * 1 = 1156$. Then we built the augmented dataset of size $k * N$, where $k = 8, 4, 2, 1$ as $m = 1, 2, 4, 8$ and N is the training set size.

Fig 8a displays the accuracies of the classifier trained 1)without any augmentation; 2) with rotation-based augmentation; 3) with cGAN-based augmentation; 4)with both cGAN and rotation-based augmentation (RGC). We can see that the augmentation via RGC brings accuracy increase, which are 6.8%, 4.5%, 2.6% and 1.1% in sequence. The results prove the assumption that when the original dataset is small the generated samples will bring more diversity and let the classifier learn more general features. cGAN-based augmentation proposes a 6.4% to 1.1% absolute accuracy increase while that of rotation-based augmentation is 0.8% to 0.0%, which means the former contributes most to the accuracy increase. Compared with only using cGAN for augmentation, the combination of cGAN and rotation leads to a small improvement when the dataset is small, i.e., $m = 1$ *or* 2. When $m \leq 4$, the combination brings no more benefits than only using cGAN. When the base sample number is eight, we see the highest accuracy which is 94.4% with a standard deviation 0.53%.

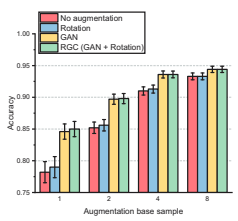
To summarize, we think RGC brings accuracy increase for training a gesture classifier. Especially when there is a lack of data, using RGC is significantly superior.

2) Improvement of different classifiers.: We added new classifiers LSTM and ResNet-IMU that are mentioned in Sec IV-B, while other experiment settings stay the same as Sec IV-C1 described.

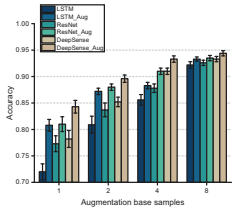
Fig 8b plots the classifiers' accuracies trained with or without RGC. We can see that RGC brings benefits over all three classifiers, where the accuracy gain varies from 8.8% to 1.1%. DeepSense shows superiority over the three classifiers since it always has the highest accuracy. ResNet shows comparable performance as DeepSense, and the accuracy gains are also similar between these two models. However, LSTM shows a different situation. LSTM presents the lowest original accuracy (trained without RGC) among the three models and the highest accuracy gain. When $m = 1$, the accuracy of LSTM is only 72.0%. After training with the augmented dataset, it increases to 80.8%, which is basically equal to that of ResNet. From another aspect, the original accuracy difference of LSTM and the other two reduces much when m is increasing, i.e., the training set is increasing. We concluded that among the three classifiers, LSTM is least good at learning robust features when trained with an insufficient dataset and thus can be improved by RGC most significantly.

3) Generality: To evaluate the generality of the trained model, we adopted leave-one-user-out experiments. Every time we chose one volunteer as the test volunteer, the training set removed all samples of this volunteer while the test set only kept samples of this volunteer. In other words, this volunteer is unseen in the training set. Otherwise, the dataset setting remains unchanged as Sec IV-C1. The classifier we used is DeepSense.

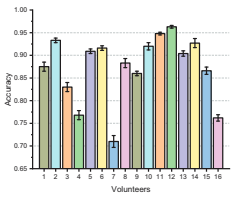
Fig 8c draws the accuracies of every volunteer. Compared with the original average accuracy, we can see an accuracy drop for most users. However, for 50% of users the accuracy is still above 90%, and for 75% of users the accuracy is above 85%. Only for three users the accuracy dropped below 80%. In the worst case, the absolute accuracy drops 22% to 72%. We analyzed the data and found that the drop is caused by certain gestures, for example, writing digits '0' and '6' of volunteer 7 show a less than 30% recognition rate. We believe it is because this volunteer has unique habits of drawing these two digits so that these samples cannot be recognized via the classifier trained on the rest volunteers' data. Since accuracies for most users do not drop or have a reasonable drop, we think the results prove the generality of the trained model over different users. We also think it is always good to add the data of the



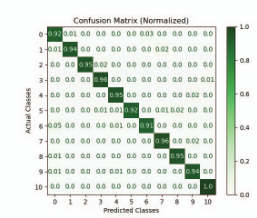
(a) Recognition performance of the same classifier trained with different augmentation methods.



(b) Recognition performance of three different classifiers trained on 1) the original training set; 2) the augmented one.



(c) Recognition performance across different users.



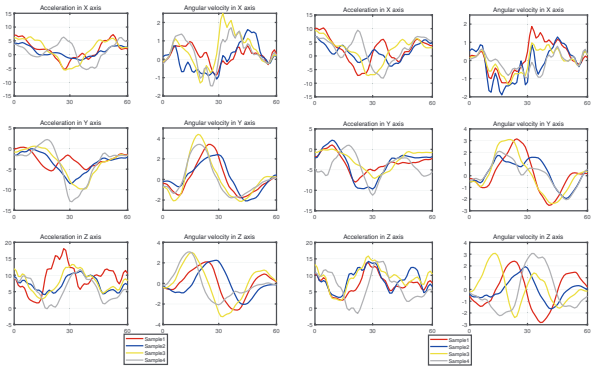
(d) Recognition performance across different gestures.

target user to the training set.

We also evaluated the classification performance across different gestures. In Fig 8d we plot the normalized confusion matrix based on the recognition results in Sec IV-C1 ($m = 8$). In this figure we omitted the parts whose normalized ratio is below 0.01 for a more clear display, those omitted numbers represent very few mismatches across every categories. From the figure, we can see accuracies of every gesture are above 90%. For writing digits 2, 3, 4, 7 and 8, the accuracies are above 95%. Writing digits 0 and 6 tend to be misclassified as each other, which reduces the recognition rate of both of them. We believe this is because of the similar shape of the two digits. It is worth noting that the recognition rate for null gesture is 100%, which means our unintended gesture is not likely to be classified as a gesture. However, a few samples of writing '3' are recognized as null gesture.

To summarize, the trained classifier shows generality over different users and gestures.

4) Generated samples by GANs: Fig 9(a) displays four real sample of writing digit '0' and Fig 9(b) displays four generated samples by the cGAN. First, we can see the real samples and the generated samples are similar in the shape of the curve at each axis. Second, we could clearly notice differences among the generated samples. They have different peaks and valleys, even the number of peaks are different. Therefore, the cGAN is not only trying to remember some certain samples but learn from the gesture distribution, and there is no mode collapse in our cGAN training. Third, we can see the generated samples have smooth curves which means the TV loss suppressed the generation of burrs very well.



(a) (b)

Fig. 9: 6-axis IMU data of gesture writing digit '0' which belongs to (a) real gestures; (b) generated samples by the cGAN.

V. RELATED WORK

Researchers have put a lot of efforts into wearable-based gesture recognition, and some works try to address the problem of sensor noises from unexpected movements.

Gesture recognition with unexpected movements. SignSpeaker [4] uses LSTM [21] to recognize sign language and achieves an 99% accuracy when the user is sitting still. It leverages an average filter to avoid the influence of unexpected movements, but when users are walking or turning, the accuracy still decreases around ten percents. ViBand [22] collects high frequency bio-acoustic signals to recognize finger-level gestures, and it uses a high-pass filter which prevents the low frequency noise caused by unexpected movements. However, this methods can't be applied to arm-level gestures since the signal and the noise are in the same frequency band as mentioned in Sec II. Float [5] combines both PPG and IMU signals to detect air-tap gesture, and system performance shows no significant decrease in a walking scenario. The limitation is that this work only evaluated the tapping gesture but not rich gestures. SynchroWatch [23] measures the magnetic field change of a smartwatch and a ring on the thumb and use it to classify hand gestures. This method could avoid the affect by unexpected movements since it uses magnetic sensing, but the requirement of special devices and common indoors magnetic disturbance will limit its usage in practice. Existing study also used an additional device enhance the recognition robustness to noise caused by unexpected movements. It assumes another smart device such as a smart phone is in the user's pocket, so noises of the smartwatch IMU can be canceled through subtracting the IMU readings of the smartphone. This method requires somehow strong assumption since the user may not carry the phone all day, and the phone in the pocket may not be so tightly attached to the torso which means the sensor readings could be different from true values of the body.

To summarize, existing studies tried to solve the problem of

unexpected movement via 1)filtering, and 2) another supporting device, which only works for certain gestures and scenes. In this paper, we would like to collect the gesture data under different unexpected movements, and coping with the lack of data.

GANs. One recently proposed framework for the generation of artificial data are generative adversarial networks [6] which showed groundbreaking results for the generation of artificial images. Originally, vanilla GANs suffered heavily from training instability and were restricted to low resolution images. A lot of advancement in regard to stability and the quality of the generated images has been made with different regularization methods [13], [24]–[26]. GANs also allow the intentional manipulation of specific properties in generated samples [7] and therefore could prove to be a useful tool in understanding the original data distribution used for training the GAN. GANs have mainly been developed and applied to the generation of images and only a few studies investigating time series were conducted; recently they showed promising results for the generation of artificial audio [27]. The generation of artificial IMU signals would have applications in many different areas including activity/gesture classification, but to our best knowledge no research regarding the generation of raw IMU signals with GANs has been published at this time.

VI. Conclusion

In this paper we proposed RGC, which is a framework targeting practical gesture classification via wearables. We reconsidered the problem of IMU noises caused by unexpected movements of the user. When facing the challenge that collecting gesture samples is time-consuming, we proposed a GANs-based data augmentation method to enhance the diversity of the dataset and improve the recognition. We evaluated RGC using arm gestures recognition experiments. The results showed that RGC provides accuracy gain varies from 8.8% to 1.1% with different dataset sizes and classifiers.

References

[1] Siddharth S Rautaray and Anupam Agrawal. Vision based hand gesture recognition for human computer interaction: a survey. *Artificial intelligence review*, 43(1):1–54, 2015.

[2] Jian Wu, Lu Sun, and Roozbeh Jafari. A wearable system for recognizing american sign language in real-time using imu and surface emg sensors. *IEEE journal of biomedical and health informatics*, 20(5):1281–1290, 2016.

[3] Sheng Tan and Jie Yang. Wifinger: leveraging commodity wifi for fine-grained finger gesture recognition. In *Proceedings of the 17th ACM international symposium on mobile ad hoc networking and computing*, pages 201–210. ACM, 2016.

[4] Jiahui Hou, Xiang-Yang Li, Peide Zhu, Zefan Wang, Yu Wang, Jianwei Qian, and Panlong Yang. Signspeaker: A real-time, high-precision smartwatch-based sign language translator. 2019.

[5] Ke Sun, Yuntao Wang, Chun Yu, Yukang Yan, Hongyi Wen, and Yuanchun Shi. Float: one-handed and touch-free target selection on smartwatches. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*, pages 692–704. ACM, 2017.

[6] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

[7] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.

[8] Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014.

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

[10] Vincent Dumoulin and Francesco Visin. A guide to convolution arithmetic for deep learning. *arXiv preprint arXiv:1603.07285*, 2016.

[11] Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *Icml*, 2010.

[12] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958, 2014.

[13] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C Courville. Improved training of wasserstein gans. In *Advances in neural information processing systems*, pages 5767–5777, 2017.

[14] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.

[15] Nicolas Dey, Laure Blanc-Feraud, Christophe Zimmer, Pascal Roux, Zvi Kam, Jean-Christophe Olivo-Marin, and Josiane Zerubia. Richardson–lucy algorithm with total variation regularization for 3d confocal microscope deconvolution. *Microscopy research and technique*, 69(4):260–266, 2006.

[16] Xinye Lin, Yixin Chen, Xiao-Wen Chang, Xue Liu, and Xiaodong Wang. Show: Smart handwriting on watches. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 1(4):151, 2018.

[17] Wenjin Tao, Ze-Hao Lai, Ming C Leu, and Zhaozheng Yin. Worker activity recognition in smart manufacturing using imu and semg signals with convolutional neural networks. *Procedia Manufacturing*, 26:1159–1166, 2018.

[18] Haohua Huang, Pan Zhou, Ye Li, and Fangmin Sun. A lightweight attention-based cnn model for efficient gait recognition with wearable imu sensors. *Sensors*, 21(8):2866, 2021.

[19] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[20] Antreas Antoniou, Amos Storkey, and Harrison Edwards. Data augmentation generative adversarial networks. *arXiv preprint arXiv:1711.04340*, 2017.

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

[22] Gierad Laput, Robert Xiao, and Chris Harrison. Viband: High-fidelity bio-acoustic sensing using commodity smartwatch accelerometers. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology*, pages 321–333. ACM, 2016.

[23] Gabriel Reyes, Jason Wu, Nikita Juneja, Maxim Goldshtein, W Keith Edwards, Gregory D Abowd, and Thad Starner. Synchrowatch: One-handed synchronous smartwatch gestures using correlation and magnetic sensing. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 1(4):158, 2018.

[24] Xudong Mao, Qing Li, Haoran Xie, Raymond YK Lau, Zhen Wang, and Stephen Paul Smolley. Least squares generative adversarial networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2794–2802, 2017.

[25] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In *International conference on machine learning*, pages 214–223. PMLR, 2017.

[26] Naveen Kodali, Jacob Abernethy, James Hays, and Zsolt Kira. On convergence and stability of gans. *arXiv preprint arXiv:1705.07215*, 2017.

[27] Chris Donahue, Julian McAuley, and Miller Puckette. Synthesizing audio with generative adversarial networks. *arXiv preprint arXiv:1802.04208*, 1, 2018.