Virtual Data Generation for Complex Industrial Activity Recognition : An Approach Based on Interpolation Algorithms and Data Augmentation Techniques

Atsushi Yanagisawa* [1]
[1]Kyoto University

Abstract

In this study, we propose a virtual data generation method tailored for the OpenPack Challenge, which leverages acceleration sensor data along with associated operation and action labels. Our approach integrates simple yet effective data augmentation techniques (e.g., jitter, axis switching, and time warping) with advanced interpolation algorithms (including various forms of RBF and FFT-based methods) to generate high-quality virtual data. Experimental findings indicate that data-driven methods specifically adapted to the characteristics of sensor signals can substantially improve HAR model performance, offering a promising solution in scenarios where large-scale data collection is impractical. Specifically, by grouping the data according to the action and operation labels and applying an RBF-based interpolation algorithm to each group, an F1 score of 0.6379 was achieved. However, employing more advanced data generation algorithms such as GANs may further improve performance.

1 Introduction

Human Activity Recognition (HAR) has emerged as a key research area in Human-Computer Interaction (HCI) [1]. One goal of activity recognition is to provide information on a user's behavior that allows computing systems to proactively assist users with their tasks [2].

HAR is applied in various fields including industrial and medical domains, and is also used for Animal Behavior Understanding [3]. In this study, we address the challenge utilizing the OpenPack dataset [4][5], which extends HAR concepts to industrial settings, particularly logistics

[1]yanagisawa.atsushi.62t@st.kyoto-u.ac.jp

centers where manual labor processes such as assembly and packaging remain essential. While the previously mentioned applications focus on general activity recognition, industrial environments present unique challenges due to variations in task details, item sizes, and quantities, despite their reliance on human workers to meet dynamic demand shifts.

The OpenPack challenge specifically tasks us with developing a virtual data generative method to enhance HAR performance within constrained computational resources.

To achieve this, our research sets the objective; Within the constrained capacity of 500MB, we generate virtual data aimed at enhancing HAR model performance by employing techniques such as data augmentation and implementation.

2 Method

In this study, we propose a virtual data generation method aimed at improving Human Activity Recognition (HAR) performance by leveraging the acceleration sensor data provided in the challenge utilizing the OpenPack dataset [5][4]. The target data include acceleration signals (atr01/acc_x, atr01/acc_y, atr01/acc_z, atr02/acc_x, atr02/acc_y, atr02/acc_z), an operation label (`operation`), an auxiliary action state (`action`), and a timestamp (`timestamp`).

In our approach, we conducted two primary strategies: data augmentation and implementation techniques. For data augmentation, we apply methods such as axis switching, flipping, scaling, and others that are designed to respect the inherent characteristics of sensor data. In the implementation aspect, we employ techniques including linear interpolation and statistical scaling to generate the virtual data.

The rationale for choosing these simple data augmentation and interpolation techniques is as follows:

- **Adaptation to Task Characteristics:** The HAR model processes contiguous blocks of 300 rows and uses the most frequent operation label as the ground truth. Because data near the boundaries may involve label transitions, it is essential to generate virtual data that captures both uniform segments and transitions—a challenge that conventional methods like TimeGAN struggle to overcome.
- **Computational Efficiency:** Given the 500MB data capacity and the strict time constraints of the competition, the data generation method must be highly efficient. This requirement favors the use of simple, fast techniques rather than resource-intensive GAN-based approaches.
- **Limitations of Existing Methods:** For instance, TimeGAN has been proposed for time series data generation; however, it has notable disadvantages. TimeGAN is unable to adapt to time series that exhibit different labels (in this case, Operation labels) at different times within a single sequence, and its operational speed is relatively slow [6].

The data generated by the methods described below are used to train a HAR model with a specified architecture, and the accuracy is evaluated based on the model's F1 score.

2.1 Development and Evaluation of Data Augmentation Techniques

We implemented several data augmentation techniques that consider the physical properties of sensor data. The following methods were applied to each subject's data.

2.1.1 Switch Axis

To promote features that are independent of sensor orientation, we switched the order of the x, y, and z axes. With three axes, there are $3! = 6$ permutations. Denote the transformed data as $\mathbf{x}'$, $\mathbf{y}'$, and $\mathbf{z}'$. The possible combinations are:

1. $[x', y', z'] = [x, y, z]$ (original data)
2. $[x', y', z'] = [x, z, y]$
3. $[x', y', z'] = [y, x, z]$
4. $[x', y', z'] = [y, z, x]$
5. $[x', y', z'] = [z, x, y]$
6. $[x', y', z'] = [z, y, x]$

2.1.2 Flip

To reduce directional dependency in time-series data, we perform a temporal reversal where the time order of the sequence is inverted. This transformation is applied to each variable.

2.1.3 Scaling

We applied scaling transformations to each channel of the input data independently. The scaling factors were randomly sampled from a normal distribution centered at 1.0 with a standard deviation of 0.3, ensuring that the change was at least 5% from the original.

2.1.4 Time Warping

Time warping perturbs temporal locations by smoothly distorting the time intervals between samples[7]. This transformation involves generating distorted time steps through non-linear deformation of the time axis, applying interpolation to remap the signal, and thus simulating variations in movement speed.

2.1.5 Jitter

Gaussian noise is added to each data point to simulate sensor noise. This technique is also considered as a way of simulating additive sensor noise[7]. pecifically, in our implementation, we added Gaussian noise with a mean of 0.1

2.1.6 Magnitude Warping

2[7].

2.2 Development of Data Generation Methods via Interpolation

In order to generate new data while preserving the characteristics of the real data, we developed and evaluated a series of linear interpolation techniques. These methods were selected with the objective of efficiently increasing the dataset size while maintaining the intrinsic structure of the original data.

2.2.1 Overall Procedure for Virtual Data Generation

The following processing flow was executed:

1. Group the data according to either the `Action` labels.
2. For each group, determine the quantity and positions of the data to be generated.
3. Apply the selected filters to remove noise from the data.
4. Apply an appropriate interpolation algorithm to construct a continuous function.
5. Extract new data points at uniform intervals from the constructed function.
6. Round the values as necessary to obtain the final virtual data.

In addition, filtering and interpolation were applied individually to each explanatory variable with respect to the timestamp, thereby ensuring that the generated virtual data retain the characteristics of the original data.

2.2.2 Grouping Data by Action

we combined both `operation` and `action` labels—data points are considered to belong to the same group only if both labels match. This approach, henceforth referred to as grouping by Action, provides greater temporal consistency at shorter time scales compared to operation-based segmentation alone, and facilitates the learning of more detailed activity patterns. This granular grouping serves as the foundation for our subsequent data augmentation and interpolation methods.

2.2.3 Determining the Quantity and Position of Generated Data(Statistical Scaling)

Normal Case In the absence of specific instructions, the following procedure was adopted:

- Obtain the minimum and maximum timestamps for each group, denoted as $t_{\min}$ and $t_{\max}$, respectively.
- For each data point in the group with a timestamp t, calculate a relative timestamp t' using the following equation:

$$t' = t - t_{\min} \tag{1}$$

The computed t' serves as the timestamp for the data points obtained after performing linear interpolation.

Statistically Varied Case In this approach, the following procedure was adopted:

Preprocessing

1. Sample the number of data points among groups sharing the same label (e.g., groups formed based on the same Operation).
2. Compute the mean and standard deviation of the number of data points.

Data Generation

1. Obtain the minimum and maximum timestamps for each group, denoted as $t_{\min}$ and $t_{\max}$, respectively.
2. For each data point in the group with a timestamp t_i, compute the relative timestamp t'_i using Equation 2. Let the maximum relative timestamp be $t'_{\max}$ (note that the minimum is 0).

   $$t'_i = t_i - t_0 \tag{2}$$

   where t_0 is the minimum timestamp in the group.
3. Determine the number of new data points to be generated by assuming that the Normal distribution follows one of the following distributions, using the computed mean and standard deviation:

   Let the resulting (integer) number of data points be n. If $n \leq 0$, set $n = 1$.
4. After performing linear interpolation, compute the timestamp for the new data points using:

$$t'_{\text{new}} = \frac{t'_{\max} \times i}{n}, \quad i = 0, 1, 2, \ldots, n-1. \tag{3}$$

Our data generation approach is based on a fundamental assumption:

Activities with identical `operation` and `action` labels exhibit uniform motion patterns that differ primarily in their temporal execution.

In other words, the same action performed multiple times may vary in duration but maintains consistent motion characteristics.

To accommodate this temporal variance, we generate virtual data by randomly varying the number of data points sampled within each labeled group. This approach effectively simulates the natural variability in execution speed that occurs when humans perform the same activity multiple times. By manipulating the sampling density rather than altering the motion pattern itself, we preserve the essential kinematic signature of each activity while introducing realistic temporal variations.

The newly generated data points are denoted by t_i.

2.2.4 High-Quality Data Generation Combining Filtering Processes

To improve the quality of the sensor data, several filtering processes were applied to each group prior to interpolation. These filters are designed to reduce noise and remove outliers, thus enabling the generation of higher-quality virtual data. The filtering algorithms used are described below:

- **Moving Average Filter** The moving average filter smooths the signal by replacing each data point with the average of its neighboring points within a sliding window of length M. For a discrete signal $x[n]$, the filtered output $y[n]$ is given by:

  $$y[n] = \frac{1}{M} \sum_{k=0}^{M-1} x[n-k].$$

  This method is effective in reducing high-frequency noise and is a standard technique in signal processing.

- **RANSAC Filtering** RANSAC (RANdom SAmple Consensus) is a robust method for estimating the parameters θ of a model in the presence of outliers. For data points (x_i, y_i), the inlier set is defined as:

  $$\mathcal{I} = \{ i \mid |y_i - f(x_i; \theta)| < \epsilon \},$$

  where ϵ is a pre-determined error threshold. The model parameters are estimated by maximizing the number of inliers .

- **Wavelet Filter** The wavelet filter decomposes the signal into a series of wavelet basis functions, allowing for both time and frequency localization. A continuous signal $x(t)$ can be represented as:

  $$x(t) = \sum_j \sum_k c_{j,k}\, \psi_{j,k}(t),$$

  where $\psi_{j,k}(t)$ are the wavelet functions at different scales j and positions k, and $c_{j,k}$ are the corresponding coefficients. Noise can be reduced by thresholding these coefficients .

- **Savitzky–Golay Filter** The Savitzky–Golay filter performs a local polynomial regression to smooth the data while preserving features like peaks and edges. For a window of length $2m+1$, the output at the central point is computed as:

$$y[n] = \sum_{k=-m}^{m} c_k \, x[n+k],$$

2.2.5 Selection and Implementation of Interpolation Algorithms

The following interpolation algorithms were implemented and compared for data interpolation. Each method uses specific mathematical formulations to estimate the values between known data points.

- **Linear Interpolation** Linear interpolation estimates intermediate values by connecting two adjacent data points with a straight line. Given two points (x_i, y_i) and (x_{i+1}, y_{i+1}), the interpolated value $y(x)$ for $x \in [x_i, x_{i+1}]$ is computed as:

$$y(x) = y_i + \frac{y_{i+1} - y_i}{x_{i+1} - x_i}(x - x_i).$$

- **Radial Basis Function (RBF) Interpolation**
  In RBF interpolation, the interpolant $s(x)$ is constructed as a linear combination of radially symmetric basis functions centered at the data points:

$$s(x) = \sum_{j=1}^{N} \lambda_j \, \phi(\|x - x_j\|),$$

where $\{x_j\}_{j=1}^{N}$ are the interpolation nodes, and λ_j are coefficients determined by enforcing the interpolation condition:

$$s(x_i) = f_i, \quad i = 1, 2, \ldots, N.$$

This leads to the linear system:

$$A\lambda = f, \quad \text{with} \quad A_{ij} = \phi(\|x_i - x_j\|).$$

Common choices for the radial basis function $\phi(r)$ include:

$$\begin{aligned}
\text{Inverse:} \quad & \phi(r) = \frac{1}{1+r^2}, && (4)\\
\text{Multiquadric:} \quad & \phi(r) = \sqrt{1+r^2}, && (5)\\
\text{Gaussian:} \quad & \phi(r) = \exp(-r^2), && (6)\\
\text{Linear:} \quad & \phi(r) = r, && (7)\\
\text{Cubic:} \quad & \phi(r) = r^3. && (8)
\end{aligned}$$

This approach provides a flexible framework for approximating multivariate functions.

- **Akima Interpolation**
  Akima interpolation is designed to minimize oscillations in regions where the data change rapidly. It first computes the slopes between consecutive data points:

$$m_i = \frac{y_{i+1} - y_i}{x_{i+1} - x_i},$$

  and then estimates the derivative at each point x_i by taking a weighted average of neighboring slopes:

$$d_i = \frac{|m_{i+1} - m_i|\, m_{i-1} + |m_{i-1} - m_{i-2}|\, m_i}{|m_{i+1} - m_i| + |m_{i-1} - m_{i-2}|}.$$

  A cubic polynomial is then fitted in each interval $[x_i, x_{i+1}]$ using the computed derivatives, ensuring a smooth transition across segments .

- **PCHIP Interpolation (Piecewise Cubic Hermite Interpolating Polynomial)**
  PCHIP interpolation constructs a piecewise cubic polynomial that preserves the monotonicity and shape of the data by ensuring the continuity of both the function and its first derivative. In each interval $[x_i, x_{i+1}]$, the interpolant $p_i(x)$ is defined as:

$$\begin{aligned} p_i(x) = {} & y_i H_1(t) + y_{i+1} H_2(t) \\ & + d_i (x_{i+1} - x_i) H_3(t) \\ & + d_{i+1} (x_{i+1} - x_i) H_4(t), \end{aligned}$$

  where

$$t = \frac{x - x_i}{x_{i+1} - x_i},$$

  and the Hermite basis functions are given by:

$$\begin{aligned} H_1(t) &= 2t^3 - 3t^2 + 1, \\ H_2(t) &= -2t^3 + 3t^2, \\ H_3(t) &= t^3 - 2t^2 + t, \\ H_4(t) &= t^3 - t^2. \end{aligned}$$

  The derivatives d_i are computed in a manner that preserves the monotonicity of the data. This method is particularly useful for avoiding overshoots and preserving the overall shape of the data .

- **FFT Interpolation**
  FFT-based interpolation utilizes the Fourier transform to enhance the resolution of the signal. For a discrete signal $x[n]$, where $n = 0, 1, \ldots, N-1$, the Discrete Fourier Transform (DFT) is computed as:

$$X[k] = \sum_{n=0}^{N-1} x[n]\, e^{-j \frac{2\pi}{N} kn}, \quad k = 0, 1, \ldots, N-1.$$

To increase the resolution, the spectrum $X[k]$ is zero-padded to a length $M > N$, resulting in $\tilde{X}[k]$, and the interpolated signal is obtained by the inverse FFT:

$$\tilde{x}[n] = \frac{1}{M} \sum_{k=0}^{M-1} \tilde{X}[k]\, e^{j\frac{2\pi}{M}kn}, \quad n = 0, 1, \ldots, M-1.$$

In order to mitigate spectral leakage and reduce edge effects, a window function $w[n]$ is applied to the original signal:

$$x_w[n] = x[n]\, w[n].$$

In this study, the following window functions were evaluated:

- **Hann Window:** $w[n] = 0.5\left(1 - \cos\left(\frac{2\pi n}{N-1}\right)\right)$,
- **Welch Window:** $w[n] = 1 - \left(\frac{2n-N+1}{N-1}\right)^2$,
- **Blackman-Harris Window:**

$$\begin{aligned} w[n] = a_0 - a_1 \cos\left(\frac{2\pi n}{N-1}\right) \\ + a_2 \cos\left(\frac{4\pi n}{N-1}\right) \\ - a_3 \cos\left(\frac{6\pi n}{N-1}\right), \end{aligned}$$

with $a_0 = 0.35875$, $a_1 = 0.48829$, $a_2 = 0.14128$, and $a_3 = 0.01168$.

This FFT-based interpolation scheme, with its windowing strategy, is described in standard signal processing texts.

These algorithms were chosen to compare the effectiveness of interpolation techniques with different properties (such as smoothness, locality, and computational efficiency) in order to identify the method most suitable for the sensor data.

2.2.6 Rounding Data to Four Decimal Places

Each data point was rounded to four decimal places by rounding the fifth decimal place. The reasons for this processing are as follows:

- The original data is recorded with precision up to the fourth decimal place, and the generated data is matched to this level of precision.
- Data size is reduced, which facilitates the generation of a larger dataset within the competition's constraint of 500MB.

3 Result

This section presents the experimental results of our virtual data generation methodology for wearable accelerometer data. We evaluated the performance of generated data using F1 scores as the primary metric. For consistency across all experiments, identical training, validation, and test datasets were used throughout the evaluation process, ensuring fair comparisons between different methods.

Due to time constraints, our study was limited to a single trial, and thus further validation is needed before generalizing these findings.

3.1 Data Augmentation Approach

Switch Axis Table 1 shows performance comparison of axis-switching combinations. The best performing patterns are:

Accelerometer 1	Accelerometer 2	F1 Score
1	1	**0.6169**
1	3	0.5891
1	2	0.5782
6	3	0.5635
3	5	0.5582

Table 1: Performance Comparison of Top 5 Axis-Switching Patterns

The results indicate that simply duplicating the data (pattern 1-1, i.e., no axis switching) yielded the highest performance.

Other Data Augmentation Techniques Table 2 summarizes the F1 scores obtained using various data augmentation methods: Among

Method	F1 Score
Jitter	**0.6351**
Flip	0.1619
Permute	0.5811
Scale	0.5791
Time Warp	0.5528
Magnitude Warp	0.5779

Table 2: Performance Comparison of Various Data Augmentation Techniques

these, the jitter method achieved the highest F1 score. Figure 1 illustrates examples of different data augmentation techniques applied to the same 100 consecutive data points compared to original data. The confusion matrix of predicted values versus ground truth for our best-performing data augmentation model (Jitter with F1 score of 0.6351) is shown in Figure 6. This visualization helps to understand which classes the model excels at recognizing and where misclassifications occur most frequently with the jitter-augmented data.

3.2 Interpolation Techniques

Interpolation Algorithms We evaluated various interpolation algorithms to generate virtual data. For all experiments in this paragraph, we first extracted the original timestamp values for each action group (corresponding to Normal Case) and used these as the foundation for generating new data points. No filters were applied to the data in this initial evaluation. Table 3 shows the performance comparison of different interpolation algorithms when applied to action-based groupings: Here, RBF interpolation with the inverse kernel achieved the highest performance. Interestingly, FFT-based methods performed better in this grouping compared to the operation-based grouping, likely due to the more homogeneous and periodic nature of the action groups. Figure 3 illustrates the effect of various interpolation algorithms on a representative sample of 100 consecutive data points from the acc_x signal of the atr01 sensor. The visualization compares how different methods transform the original signal.

Method	F1 Score
Linear Interpolation	**0.6340**
RBF Interpolation	
Inverse	**0.6358**
Multiquadric	0.5805
Gaussian	0.5243
Linear	0.5563
Cubic	0.5768
FFT Interpolation	
Hann Window	0.5852
Welch Window	0.5946
Blackman-Harris Window	0.5255
PCHIP	**0.6202**
Akima	0.5976

Table 3: Performance Comparison of Interpolation Algorithms (Action-Based Grouping)

Statistical Scaling Table 4 summarizes the performance of all interpolation methods without applying any filters, using the normal distribution for statistical scaling:

Figure 4 visualizes the impact of statistical scaling on the interpolated data, displaying 100 consecutive data points from the acc_x signal of the atr01 sensor.

Based on Table 4, RBF interpolation with the Inverse kernel achieved the highest F1 score of 0.6358 among all tested interpolation methods when applied to action-based grouping with statistical scaling based on a normal distribution.

Filter Comparison Table 5 presents the performance when various filters were applied to the action-based groups, using the two best-performing interpolation methods from the previous experiment:

Figure 5 visualizes the impact of different filters on the interpolated data, displaying 100 consecutive data points from the acc_x signal of the

Method	F1 Score
Linear Interpolation	0.5790
RBF Interpolation	
Inverse	0.6358
Multiquadric	0.5853
Gaussian	0.6084
Linear	0.6062
Cubic	0.6188
FFT Interpolation	
Hann Window	0.5724
Welch Window	0.5756
Blackman-Harris Window	0.5133
PCHIP	0.6202
Akima	0.5774

Table 4: Performance Comparison of Interpolation Methods with Rounding (Action-Based, Normal Distribution Assumed)

Filter	PCHIP	RBF (Inverse)
Moving Average	0.5906	0.5599
RANSAC	0.5515	0.6034
Wavelet	0.5851	0.5762
Savitzky-Golay	0.5706	**0.6379**
None	0.6202	0.6358

Table 5: Filter Comparison for Action-Based Grouping

atr01 sensor.

The combination of the Savitzky-Golay filter with **RBF (inverse)** yielded the highest F1 score of 0.6379. The confusion matrix of predicted values versus ground truth for our best-performing model (RBF interpolation with inverse kernel and Savitzky-Golay filter) is shown in figure 6.

4 Discussion

4.1 Data Augmentation

The results from Tables 1 and 2 indicate that, in many cases, simply duplicating the data (i.e., applying no axis switching) yielded the best performance. This suggests that axis switching, which was expected to enhance model generalizability, actually reduced performance in our specific case. Among the augmentation techniques, jitter produced the highest F1 score.

The superior performance of jitter augmentation deserves careful consideration. There are multiple possible explanations for this observation. One hypothesis is that the addition of noise through jitter may have coincidentally created training samples that closely resembled the test data distribution. Alternatively, the introduced noise might have effectively canceled out existing noise in the sensor data, resulting in cleaner, more generalizable patterns for the model to learn.

It is important to note that jitter applies random perturbations drawn from a normal distribution to each time point in the series. Consequently, the augmented data varies significantly between trials. To properly assess jitter's effectiveness, multiple experimental runs with different random seeds would be necessary—a limitation in our current study, which was constrained to a single trial.

While jitter is commonly applied to both time series and image data, its effectiveness varies depending on data characteristics—sometimes improving accuracy and sometimes degrading it. In image classification domains, literature reviews indicate that other augmentation techniques such as rotation typically outperform jitter [8]. Some studies have reported that jitter augmentation fails to improve performance in Parkinson's Disease (PD) motor state classification because it introduces rapid fluctuations that can mimic symptoms , leading to classification confusion [1]. However, research by Iwana et al. [9] on 128 datasets from the 2018 UCR Time Series Archive showed that jitter generated data highly similar to the original time series in certain tasks, resulting in accuracy that surpassed other augmentation methods. This suggests that jitter's unusual effectiveness in our study, while not common, has precedent in specific time series applications.

The unusually high performance of jitter in our specific case might represent a statistical outlier rather than a generalizable finding. However, because jitter involves random perturbations, its performance may vary depending on the number of trials. Due to time constraints, our study was limited to a single trial, and thus further validation is needed before generalizing these findings.

4.2 Interpolation Techniques

Interpolation Algorithms RBF interpolation with a linear kernel outperformed other methods. In contrast, FFT-based methods performed poorly, likely because they assume periodicity, which the sensor data in operation groups do not consistently exhibit. This can be visually confirmed in Figures 3 and 4, which show the results of different interpolation methods applied to acceleration data. The FFT-based interpolation methods clearly fail to accurately preserve the signal characteristics, particularly at transitions and peaks in the data.

Filter Processing Interestingly, applying filters generally did not improve performance and in some cases even decreased it. This finding aligns with our jitter augmentation results—while jitter (adding controlled noise) improved performance, filters (which reduce noise) tended to worsen it. This suggests that a certain level of noise in the data may actually be beneficial for model training, possibly by preventing overfitting or by creating more robust feature representations. However,The combination of the Savitzky-Golay filter with RBF (inverse) yielded the best performance

(F1 score of 0.6379), demonstrating that appropriate smoothing at a finer segmentation level can be particularly beneficial. The combination of RBF interpolation with the Savitzky-Golay filter likely achieved superior performance due to their complementary strengths. As can be seen from the form of the equation (Eq. 4), RBF interpolation, particularly with an inverse kernel, emphasizes local information by rapidly reducing influence with distance from data points, enabling natural interpolation that preserves signal characteristics. The Savitzky-Golay filter, meanwhile, performs polynomial fitting within a fixed window to effectively reduce high-frequency noise while preserving critical signal features such as peaks and trends—unlike simple moving averages which tend to distort these important characteristics.[10]

When these methods are combined, the Savitzky-Golay filter first removes high-frequency noise from the original time series, allowing the subsequent RBF interpolation to operate on cleaner data, resulting in smoother and more representative interpolation curves. This integration maintains the continuity and essential patterns of the original data while generating new samples free from meaningless noise, enabling the model to recognize genuine patterns more accurately. This synergistic effect explains the superior F1 score achieved by this particular combination. The effectiveness of the Savitzky-Golay filter has been demonstrated in other domains beyond wearable sensor data. For instance, in studies using data sets obtained from the analysis of a pharmaceutical drug and its degradation products by liquid chromatography–mass spectrometry (LC–MS), applying the Savitzky-Golay filter significantly improved the signal-to-noise ratio (S/N)[11].

Model Performance(Confusion Matrix Analysis) Figures 4and 6 present the confusion matrices for our best-performing models: the Savitzky-Golay filter with RBF interpolation and jitter augmentation with $\sigma = 0.1$, respectively. Examining these matrices reveals minimal differences between the two approaches in terms of class-wise prediction patterns. A notable challenge across both models is the poor recognition of class 1000, indicating that further improvements are needed to enhance the model's ability to distinguish this particular class from others.

Despite achieving the highest overall F1 scores in our experiments, both methods demonstrate similar misclassification patterns, suggesting that they may be encountering fundamental limitations in feature representation for certain activity classes. This consistent performance ceiling across different approaches indicates that more sophisticated modeling techniques or feature engineering may be required to achieve substantial improvements in classification accuracy for challenging classes.

4.3 Limitations and Future Work

Our research has several limitations that could be addressed in future work. First, we did not explore combinations of data augmentation and interpolation techniques, which might yield synergistic improvements in performance.

Additionally, while our methods were effective, recent advances in artificial intelligence for time series generation, such as Diffusion Models and Recurrent Adversarial Networks (RAN), could potentially generate even more realistic and diverse wearable sensor data. For example, research exists that applies Recurrent Generative Adversarial Networks (RGANs) for time series data generation[12]. These AI-based approaches might better capture the complex spatiotemporal characteristics of human activity data, potentially leading to further improvements in HAR model performance.

5 Conclusion

In this study, we improved HAR model performance for the Challenge by developing an efficient virtual data generation method for acceleration sensor data. Our approach combined simple data augmentation (e.g., jitter, which performed best, though data duplication also yielded comparable results) with advanced interpolation algorithms particularly RBF interpolation using linear and inverse kernels to generate new data while preserving temporal characteristics. However, advanced AI-based approaches such as RAN hold the potential to generate data that could further enhance the performance of HAR models.

References

[1] A. Bulling, U. Blanke, and B. Schiele, "A tutorial on human activity recognition using body-worn inertial sensors," *ACM Comput. Surv.*, vol. 46, no. 3, Jan. 2014. [Online]. Available: https://doi-org.kyoto-u.idm.oclc.org/10.1145/2499621

[2] D. Abowd, A. K. Dey, R. Orr, and J. Brotherton, "Context-awareness in wearable and ubiquitous computing," *Virtual Reality*, vol. 3, no. 3, pp. 200–211, 1998. [Online]. Available: https://doi.org/10.1007/BF01408562

[3] T. Maekawa, Q. Xia, R. Otsuka, N. Yoshimura, and K. Tanigaki, "Recent trends in sensor-based activity recognition," in *2023 24th IEEE International Conference on Mobile Data Management (MDM)*, 2023, pp. 36–38.

[4] N. Yoshimura, J. Morales, T. Maekawa, and T. Hara, "Openpack: A large-scale dataset for recognizing packaging works in iot-enabled logistic environments," pp. 90–97, 2024.

[5]

[6] J. Yoon, D. Jarrett, and M. Van der Schaar, "Time-series generative adversarial networks," *Advances in neural information processing systems*, vol. 32, 2019.

[7] T. T. Um, F. M. J. Pfister, D. Pichler, S. Endo, M. Lang, S. Hirche, U. Fietzek, and D. Kulić, "Data augmentation of wearable sensor data for parkinson's disease monitoring using convolutional neural

networks," in *Proceedings of the 19th ACM International Conference on Multimodal Interaction*, ser. ICMI '17. New York, NY, USA: Association for Computing Machinery, 2017, pp. 216–220. [Online]. Available: https://doi.org/10.1145/3136755.3136817

[8] L. Taylor and G. Nitschke, "Improving deep learning using generic data augmentation," *CoRR*, vol. abs/1708.06020, 2017. [Online]. Available: http://arxiv.org/abs/1708.06020

[9] B. K. Iwana and S. Uchida, "An empirical survey of data augmentation for time series classification with neural networks," *PLOS ONE*, vol. 16, no. 7, p. e0254841, Jul. 2021. [Online]. Available: http://dx.doi.org/10.1371/journal.pone.0254841

[10] M. KhodAgholi and M. BAgheri, "Seismic data random noise attenuation using llsp smoothing." *Bollettino di Geofisica Teorica ed Applicata*, vol. 61, no. 2, 2020.

[11] M. Fredriksson, P. Petersson, M. Jörntén-Karlsson, B.-O. Axelsson, and D. Bylund, "An objective comparison of pre-processing methods for enhancement of liquid chromatography–mass spectrometry data," *Journal of Chromatography A*, vol. 1172, no. 2, pp. 135–150, 2007. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0021967307017104

[12] A. Hoelzemann, N. Sorathiya, and K. Van Laerhoven, "Data augmentation strategies for human activity data using generative adversarial neural networks," in *2021 IEEE International Conference on Pervasive Computing and Communications Workshops and other Affiliated Events (PerCom Workshops)*, 2021, pp. 8–13.

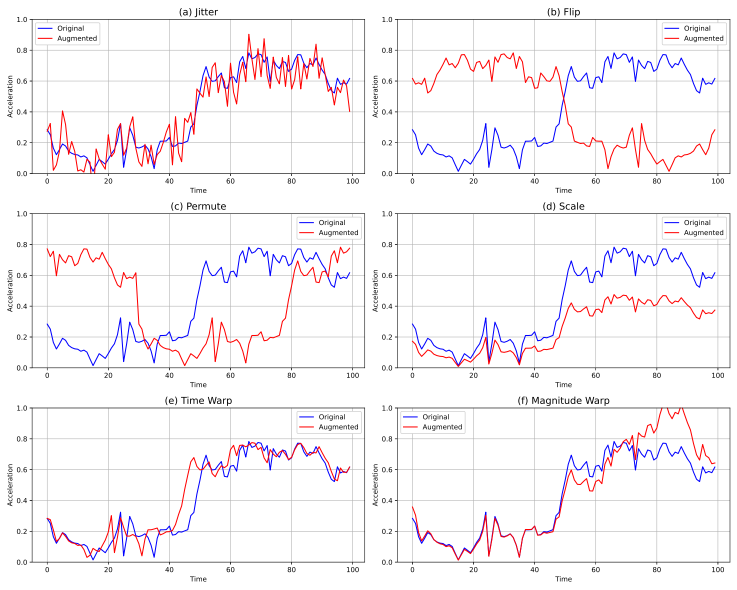
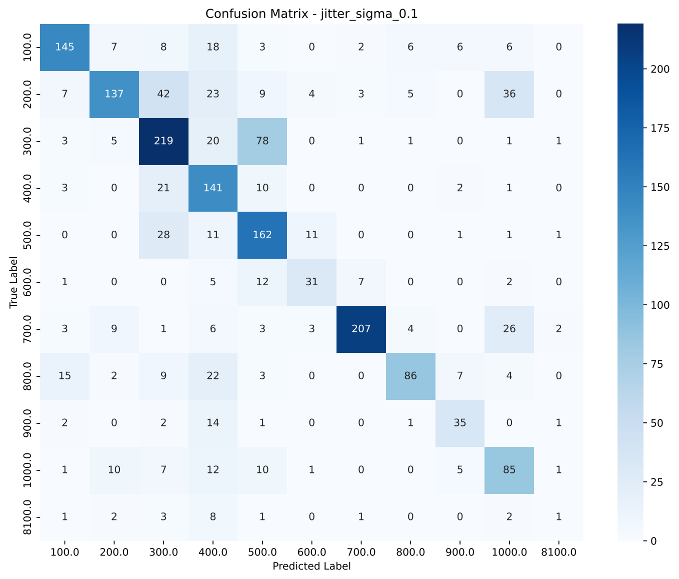


Figure 1: Comparison of Data Augmentation Techniques Applied to acc_x from atr01 Sensor



	100.0	200.0	300.0	400.0	500.0	600.0	700.0	800.0	900.0	1000.0	8100.0
100.0	145	7	8	18	3	0	2	6	6	6	0
200.0	7	137	42	23	9	4	3	5	0	36	0
300.0	3	5	219	20	78	0	1	1	0	1	1
400.0	3	0	21	141	10	0	0	0	2	1	0
500.0	0	0	28	11	162	11	0	0	1	1	1
600.0	1	0	0	5	12	31	7	0	0	2	0
700.0	3	9	1	6	3	3	207	4	0	26	2
800.0	15	2	9	22	3	0	0	86	7	4	0
900.0	2	0	2	14	1	0	0	1	35	0	1
1000.0	1	10	7	12	10	1	0	0	5	85	1
8100.0	1	2	3	8	1	0	1	0	0	2	1

Figure 2: Confusion Matrix of the Best Model (Jitter with $\sigma = 0.1$) on the Test Set

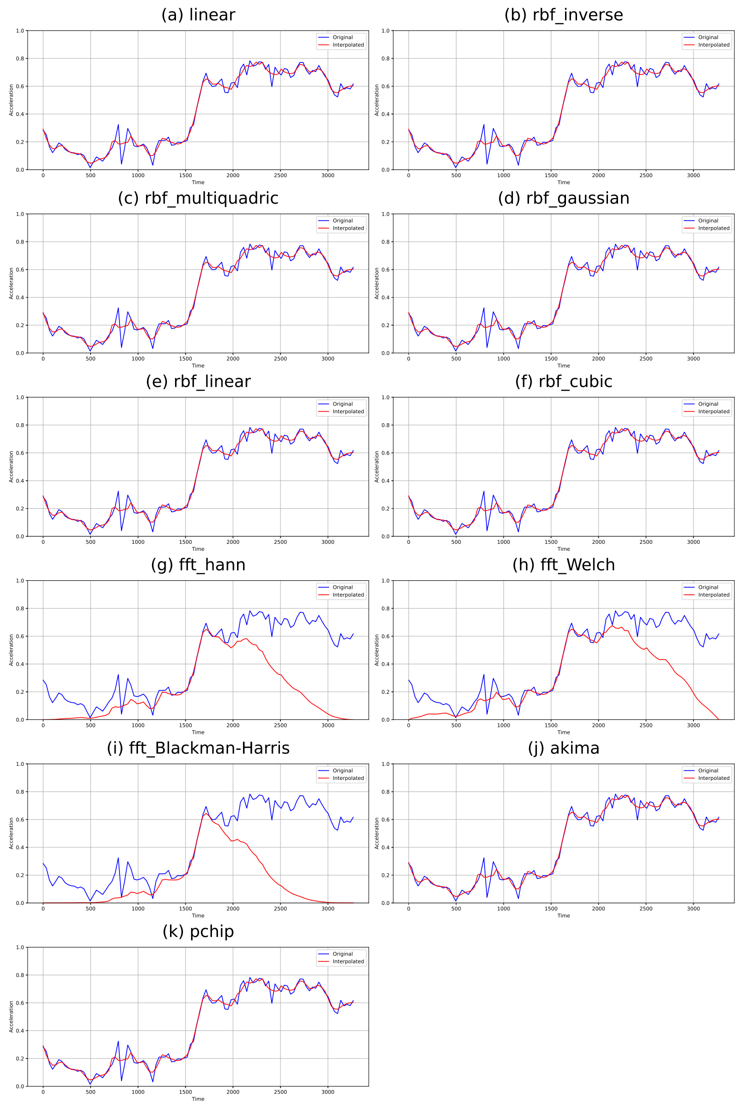


Figure 3: Comparison of Interpolation Algorithms Applied to acc_x from atr01 Sensor

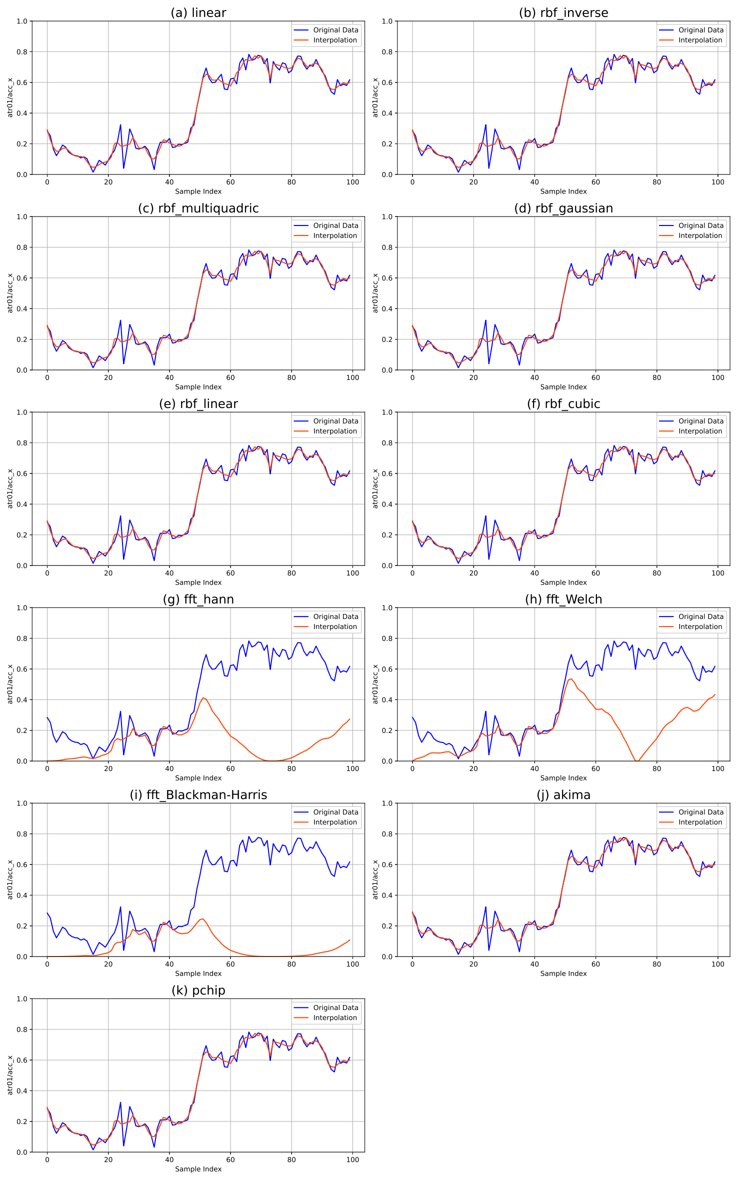


Figure 4: Comparison of Interpolation Algorithms with Statistical Scaling Applied to acc_x from atr01 Sensor

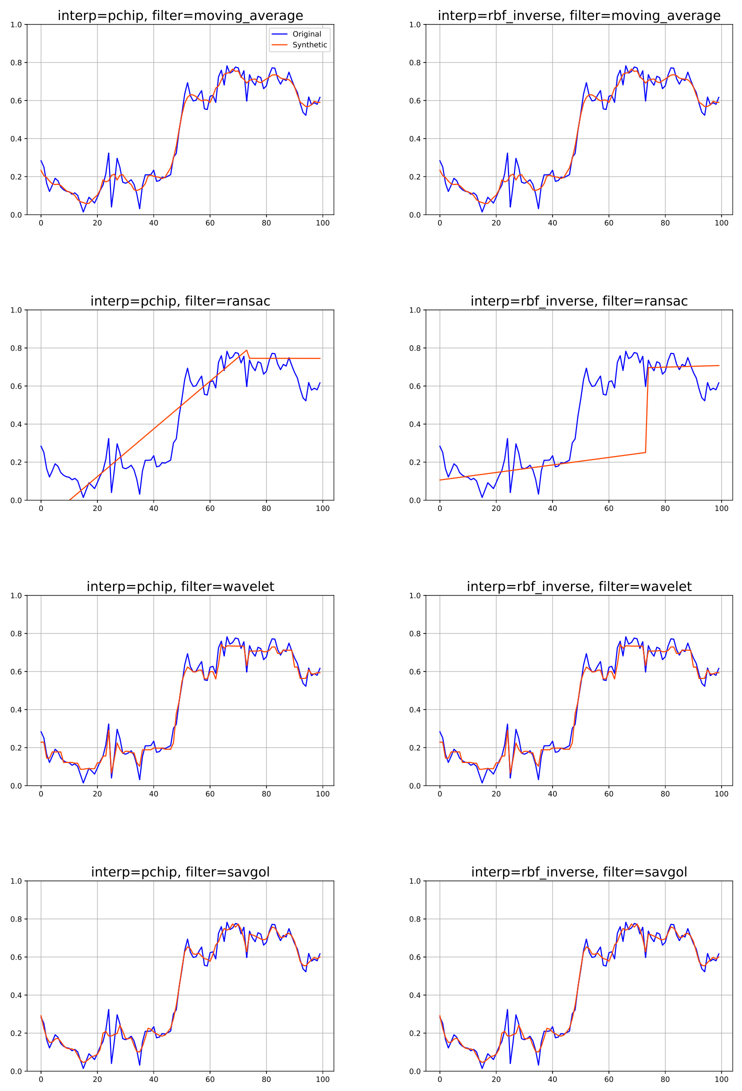


Figure 5: Comparison of Filters Applied to Interpolated Data (acc_x from atr01 Sensor)

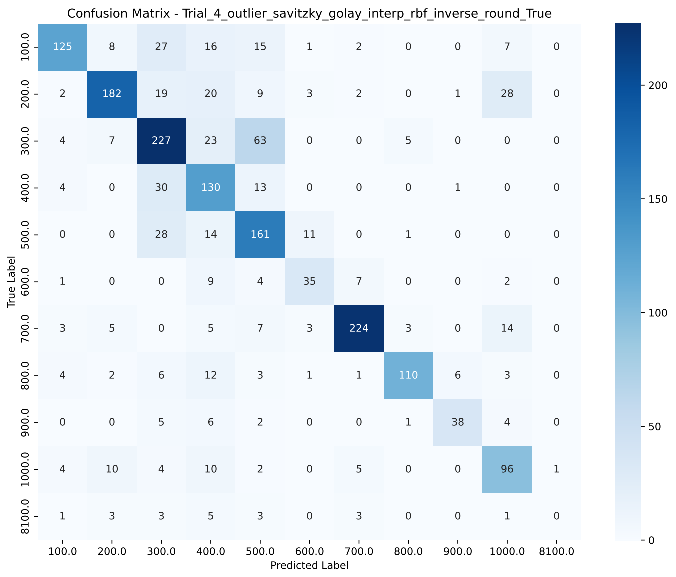


Figure 6: Confusion Matrix of the Best Model (Savitzky-Golay Filter with RBF Interpolation) on the Test Set