Heart rate anomaly detection using contractive autoencoder for smartwatch-based health monitoring

Heart Rate Anomaly Detection Using Contractive Autoencoder for Smartwatch-Based Health Monitoring

Keerthana Mannarapparambil Sivan[1], Shanfeng Hu[1], Nauman Aslam[1], Xiaomin Chen[1], Pradorn Sureephong[2], Wit Suwit[2]
[1]Department of Computer and Information Sciences, Northumbria University, United Kingdom
[2]College of Arts, Media and Technology, Chiang Mai University, Thailand
(keerthana.sivan, shanfeng2.hu, xiaomin.chen, nauman.aslam)@northumbria.ac.uk, dorn@camt.info, suwit@kirly.org

***Abstract*— The widespread adoption of wearable devices enables continuous monitoring of physiological parameters like heart rate, offering valuable insights into health. However, consumer-grade wearable data exhibits real-world noise, variations, and discontinuities across diverse populations, posing significant challenges for anomaly detection models. This paper proposes a novel deep learning approach to address these challenges, utilizing a Contractive Autoencoder (CAE) model optimized and applied specifically to noisy temporal heart rate data from wearable devices. By incorporating a contractive regularization penalty in the loss function, the model learns more robust and stable representations of the irregular data with high accuracy. Comprehensive experiments on a real-world Fitbit dataset demonstrate the proposed CAE model accurately identifies anomalous heart rate patterns missed by traditional thresholding techniques. The research encountered key challenges in ensuring model generalizability across diverse populations with natural heart rate variations, handling missing and sparse data from unreliable real-world wearable devices, and obtaining properly labelled anomaly data for robust training. Although the current model achieved promising anomaly detection results, further extensive validation on diverse datasets is essential to fully assess its capabilities across expanded demographics and use cases. Overall, this research provides an important foundation for optimizing deep learning approaches on noisy real-world wearable data through rigorous evaluation.**

Keywords*— *contractive autoencoder, contractive loss, anomaly detection, timeseries data, heartrate anomaly, outlier detection

I. Introduction

Anomaly detection in heart rate data is a crucial aspect of modern healthcare, aiming to identify and flag any abnormal or irregular heart rate patterns that may indicate potential health problems. With the widespread adoption of wearable devices, such as smartwatches and fitness trackers, continuous heart rate monitoring has become accessible and convenient for individuals of all ages and lifestyles, providing valuable insights into cardiovascular health conditions. These wearable devices are equipped with advanced sensors that capture heart rate data continuously, enabling the tracking of heart rate variations and trends over time [1]. The heart rate data collected by wearable devices encompasses a rich and dynamic time series, often referred to as continuous or time series data. Unlike traditional sporadic measurements, continuous heart rate readings are taken at regular intervals, spanning from a few seconds to a few minutes, hours or days. This continuous data stream offers a comprehensive and granular view of an individual's heart, capturing fluctuations, patterns, and rhythms that might otherwise go unnoticed with sporadic measurements [2]. By continuously monitoring heart rate, healthcare professionals and individuals can gain insights into various heart rate parameters and assess how they evolve over different time frames.

Researchers and medical practitioners have been increasingly leveraging machine learning and data-driven techniques to develop robust anomaly detection methods for heart rate data collected from wearable devices. However, existing techniques face challenges in handling noise, sparse sampling rates, and inability to generalize across diverse populations exhibiting natural variations in normal heart rate ranges. This presents a research gap for developing robust anomaly detection models that can learn effective data representations despite real-world noise and inter-personal variations.

To address these limitations, we present a novel CAE model optimized specifically for consumer-grade wearable heart rate data. Unlike existing work exploring raw PPG or ECG signals, our approach focuses on quantified beats per minute (bpm) heart rate readily available from consumer wearables. This CAE goes beyond the standard autoencoders that solely minimize reconstruction error by incorporating a contractive regularization penalty that encourages learning invariant representations robust to noise and minor perturbations. Experiments on a real-world Fitbit dataset, discussed later in this paper, demonstrate the CAE model attains over 90% accuracy in classifying normal and anomalous heart rate patterns. This high performance highlights its capabilities in handling real-world noise compared to existing techniques. By optimizing the autoencoder architecture for wearable data characteristics, we aim to unlock the potential of widely available consumer devices for preventive monitoring through early anomaly detection. The detected anomalies could have further implications for adjusting exercise based on real-time feedback to mitigate health risks.

By integrating the CAE model into wearable devices, we aspire to contribute to the advancement of digital health in the region and empower elderly individuals to take charge of their mobility disorder prevention.

The key contributions of this research are as follows,

1. Proposing a novel CAE model for accurate and robust anomaly detection in heart rate data.

2. Demonstrating the superiority of the CAE model over traditional techniques and other ML approaches on a real-world wearable device dataset.

3. Providing a method to enable early detection of heart rate irregularities using wearable devices for preventive health strategies.

The structure of this paper is as follows: In the following section, we delve into the related work on anomaly detection and contractive autoencoders in our literature review. Subsequently, the methodology section outlines our approach, data particulars, and details regarding the model architecture. In the results section, we present quantitative evaluations and visualizations of our model's performance. Finally, we summarize our findings, discuss limitations, and outline avenues for future work in the conclusion.

II. Related work

Several studies have explored different machine learning techniques to detect anomalies in heartrate data and were able to accurately detect anomalies with high sensitivity and specificity. Several commercial wearable devices, such as Fitbit and Apple Watch, have also implemented algorithms for anomaly detection in heart rate data [3]. For instance, Alugubelli et al. [4] emphasized the potential of wearable devices in remote heart rate and heart rate variability monitoring, discussing different wearable devices' accuracy, limitations, and future research directions. In a study by Liu et al. [5], a convolutional autoencoder was employed to estimate COVID-19 symptoms and anomalies. Furthermore, Abir et al. [6] introduced a deep learning framework with CNN, Variational Autoencoder, and LSTM components for detecting COVID-19 based on smartwatch data and successfully detected COVID-19 for 74% of subjects, demonstrating utility as a supplementary screening tool. The approach leverages continuously collected heart rate and step count data from consumer wearables.

Beyond heart rate anomaly detection, machine learning models have been applied to predict chronic obstructive pulmonary disease (COPD) based on physiological time series patterns [7]. Additionally, researchers explored using Fitbit-assessed behaviour as a predictor for readmission of postsurgical cancer inpatients, building a predictive machine learning model with Fitbit activity data [8].

In the domain of CAE, researchers have leveraged gradient-based activation penalties and sparse activations to reflect data's intrinsic properties [9]. CAEs have found application in diverse tasks, including cloud Intrusion Detection [10], document clustering [11], recognition of pilots' Fatigue Status [12], Online spike sorting [13], and more. For ECG denoising, Banerjee et al. [14] proposed a convolutional sparse contractive autoencoder incorporating sparsity, contractive regularization, and L2 norm.

Some prior studies developing anomaly detection models for wearable devices have utilized ECG data for training and evaluation. ECG provides detailed waveform data useful for research purposes. For example, Zhong et al. [16] proposed an unsupervised approach using convolutional autoencoders and Gaussian mixture models to estimate beat-to-beat heart rate from ECG data from wearable sensors. Carrera et al. [26] introduced an online anomaly detection system using a Variational autoencoder architecture tailored for ECG time-series data from wearable sensors. Their approach combining beat segmentation and adaptive thresholding could effectively perform online ECG monitoring and detect anomalous heartbeats.

However, consumer smartwatches mostly rely on photoplethysmography (PPG) sensors which measure blood volume changes to estimate heart rate. The PPG sensors output periodic heartbeat waveforms, which can be processed to derive a quantified beats-per-minute (bpm) value. Prior studies like Gu et al. [15] developed lightweight convolutional neural network to detect anomalies directly from the raw PPG waveform data.

While studies have used ECG and PPG data, for many commercial wearables only the quantified bpm heart rate is available to users, not the raw PPG waveform. Our work focuses specifically on analysing the bpm heart rate data readily available from consumer wearables. This quantified bpm data provides a direct measurement of heart rate in beats per minute.

We propose a CAE model designed to work with the bpm heart rate time series data for effective anomaly detection. We formulate anomaly detection as a supervised classification task, by labelling data to train the CAE model to distinguish between normal and anomalous heart rate sequences. This provides an end-to-end approach optimized specifically for heart rate time series characteristics, unlike unsupervised techniques explored in some prior work.

Existing anomaly detection techniques using deep learning, such as Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs), have faced challenges related to expert feature engineering, handling high-dimensional data, and interpretability issues [17]. Anomaly detection remains an active research area for healthcare applications. This work proposes a novel CAE architecture for heart rate anomaly detection from wearable bpm data in a supervised framework. By providing robust anomaly detection, this approach has promising implications for advancing digital health monitoring.

III. Methodology

A. *Dataset and preprocessing*

This research utilizes a public dataset of heart rate time series data collected from Fitbit users. The dataset was generated via a survey distributed on Amazon Mechanical Turk between March 12, 2016 and May 12, 2016 [25]. Fourteen Fitbit users consented to share their personal heart rate tracker data. The data is structured with each row representing a heart rate measurement at a specific timestamp.

There are 3 columns:

- Id - An identifier representing each unique user in the dataset. This allows heart rate measurements to be grouped by user.
- Time - The timestamp indicates when each heart rate reading was taken, in YYYY-MM-DD HH:MM:SS format.
- Value - The heart rate measurement in bpm recorded at each timestamp.

In total, the dataset contains 1,048,576 heart rate measurements across the 14 users. The time intervals between measurements range from 1 second to over 1 hour. This variable sampling rate is common in real-world wearable device datasets.

After conducting an initial analysis to comprehend the data distribution, continuity, and underlying patterns, several significant observations surfaced. Firstly, it became apparent that the dataset exhibited an imbalance, suggesting an uneven distribution of samples for each user. Secondly, the data displayed discontinuities, with missing observations at specific time points. Lastly, the data length varied among users, leading to differing numbers of data points for each individual. These findings highlight the importance of meticulous preprocessing and data handling to enable robust analysis.

To better understand the data, we visualized heart rate plotted against time (Fig. 1). This plot would display a fluid line reflecting fluctuations influenced by physiological factors. Patterns like gradual shifts, oscillations, and spikes/drops can be revealed. An anomaly could be represented by a sudden spike or drop-in heart rate that stands out from surrounding points (Fig. 2). This irregularity often indicates a potential health issue or error.

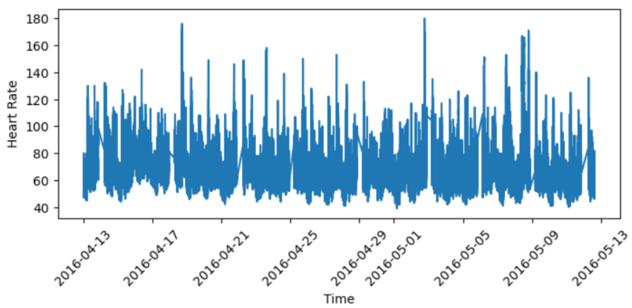

Figure 1 The temporal heartrate data for a specific individual over a specific period, ranging from 2016-04-12 to 2016-05-12, in its raw, unprocessed form.

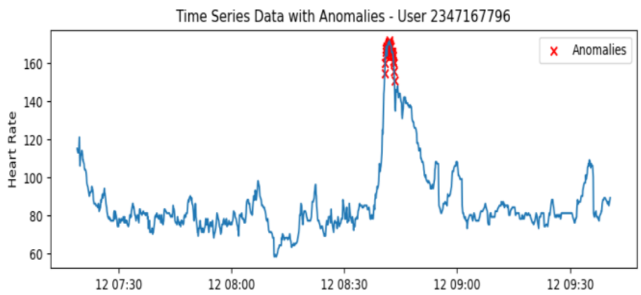

Figure 2: An example of an identified irregular or unusual variation in the heart rate of an individual, indicating a potential anomaly.

To ensure the quality and suitability of the data for the machine learning algorithm, a pre-processing step was carried out. This involved several key actions, including noise removal, handling missing values, data normalization, and converting the data into a suitable format. By performing these necessary adjustments, the data was prepared optimally for accurate and effective analysis using the machine learning algorithm.

The raw data was initially filtered to include the complete data records from just five users spanning the entire time duration. The data was then resampled to 10 second intervals for each user, with missing values imputed via forward fill. To smooth out noise, a rolling 30-sample average was applied to the heart rate readings for each user. The sequences were then sliced to contain only complete intervals of 5 minutes. This process yields a substantial volume of data with a consistent length, ensuring adequacy for training, validation, and testing purposes. Thresholds of 60 to 140 bpm were set to identify anomaly labels - any heart rates outside this range were labelled as 'Anomaly' while others were labelled as 'Normal'. The dataset was split into training, validation and testing sets for modelling.

Training an autoencoder using normal data is essential to establish a baseline understanding of regular patterns and features within the dataset. This process allows the autoencoder to learn the inherent structures and representations of normal data, which in turn enables it to distinguish anomalies or deviations when presented with unfamiliar or anomalous instances. Therefore, the dataset has been partitioned in a manner where 80% of the normal data is allocated for training purposes. The remaining data has been combined and then divided into two subsets: validation and test data.

Ultimately, all datasets underwent normalization to establish a uniform scale, preventing any individual feature from overshadowing the model training process. Normalization enhances convergence and optimizes model performance. The Min-Max Scaler was employed to rescale numerical data into a predefined range, typically bounded between 0 and 1. This rescaling conserves the inherent relationships among data points while ensuring uniformity across all features.

This multi-step preprocessing pipeline transformed the raw variable-rate data into structured sequenced samples, engineered relevant anomaly labels, and normalized features. It successfully addresses issues observed during exploratory data analysis, such as data imbalance, missing values, and variable lengths, ensuring the model's accuracy even with noisy or incomplete data.

B. Autoencoder

An autoencoder is a type of artificial neural network architecture used for learning efficient data representations in an unsupervised manner [20]. It contains an encoder that maps input data to a latent representation, and a decoder that reconstructs the input. Autoencoders can be used for various applications, including dimensionality reduction, feature learning, noise reduction, and anomaly detection [20-24].

The encoding process transforms the input x into a hidden representation y through the encoding function f, and can be mathematically represented as,

$$y = f(x) = \phi(W_x + b_h) \tag{1}$$

Here, the hidden layer is denoted as h. W_x are the weights applied to the input, b_h is the bias term, and φ is the activation function.

The decoding process then maps the hidden representation y back to a reconstructed input r through the decoding function g, the decoding operation can be represented as,

$$r = g(y) = \phi^0(W_y + b_r) \tag{2}$$

Here, W_y are the weights applied to y, b_r is the decoding bias, and ϕ^0 is the activation function. In summary, the encoder f transforms the input to a hidden representation y, which the decoder g then uses to reconstruct the input as r.

The autoencoder training minimizes the reconstruction error between input and output. The reconstruction error, also called the reconstruction loss, measures how well the autoencoder can reproduce the original input after it has been encoded and decoded [22]. If we have a dataset of inputs $D_i = [x_1, x_2, x_3, ... x_n]$, then the cost function with reconstruction error R can be expressed as,

$$J_{AE}(\theta) = \sum_{x \epsilon D_i} R(x, r) \tag{3}$$

The reconstruction error $R(x, r)$ is the mean squared error of the input x and output y and can be represented as,

$$R(x, r) = \| x - r \|^2 \tag{4}$$

C. Contractive Autoencoder

The CAE introduces a novel explicit regularization term into the traditional autoencoder cost function. This contractive penalty sets the CAE apart from standard autoencoders that solely minimize reconstruction error between the input and reconstructed output. The contractive term provides a unique form of regularization that specifically promotes robustness and stability in the learned feature representations [22].

By penalizing the Frobenius norm of the Jacobian matrix of encoder activations, the CAE cost function uniquely penalizes the model's sensitivity to minor perturbations in the input data. This encourages the model to discover encodings that are invariant and unaffected by small changes or noise in the inputs. In effect, the contractive regularization enables the model to focus on robust features that represent the underlying causal factors rather than superficial noise patterns.

Unlike common regularization techniques like early stopping and dropout, the contractive penalty directly builds invariance and robustness into the optimization process through the cost function. This novel regularization approach improves generalizability and stability compared to basic autoencoders trained only to minimize reconstruction error $R(x, r)$. The contractive term also guides the model to identify salient features that are insensitive to input noise.

The total CAE cost function can be represented as,

$$J_{CAE}(\theta) = \sum_{x \epsilon D_i} (R(x, r) + \lambda ||J_f(x)||_f^2) \tag{5}$$

Here, λ controls the weighting of the contractive term and $J_f(x)$ is the Jacobian matrix of hidden layer activations with respect to x.

D. Proposed Method

The input data is sequence of 30 time steps. Hence, the input dimension for each sample fed into the encoder is 30. This input is passed through a sequence of dense layers in a stacked architecture that progressively compresses the data into a reduced encoding. The decoder portion then reconstructs the data back to the original input dimension of 30.

Encoder: The encoder consists of three dense layers with 240, 120, and 60 neurons, respectively, with each layer decreasing in size (Fig. 3). This stacked compression approach reduces the dimensionality of the data. Each layer employs the Rectified Linear Unit (ReLU) activation function. ReLU provides nonlinear behaviour while avoiding problems like vanishing gradients. It leads to better generalization, faster training, and more interpretable models compared to other activation functions. To enhance stability and robustness of the learned encoding, L2 regularization is applied to the first dense layer, encouraging the acquired representations to be more reliable. Additionally, a dropout layer follows the third dense layer, serving as a regularization technique to mitigate overfitting during the training process. The encoder produces a final output with an encoding dimension of 30 neurons, capturing the compressed and informative encoding of the heart rate data. This reduced encoding dimension controls model complexity.

Decoder: The decoder portion mirrors the encoder architecture in reverse order to reconstruct the original input

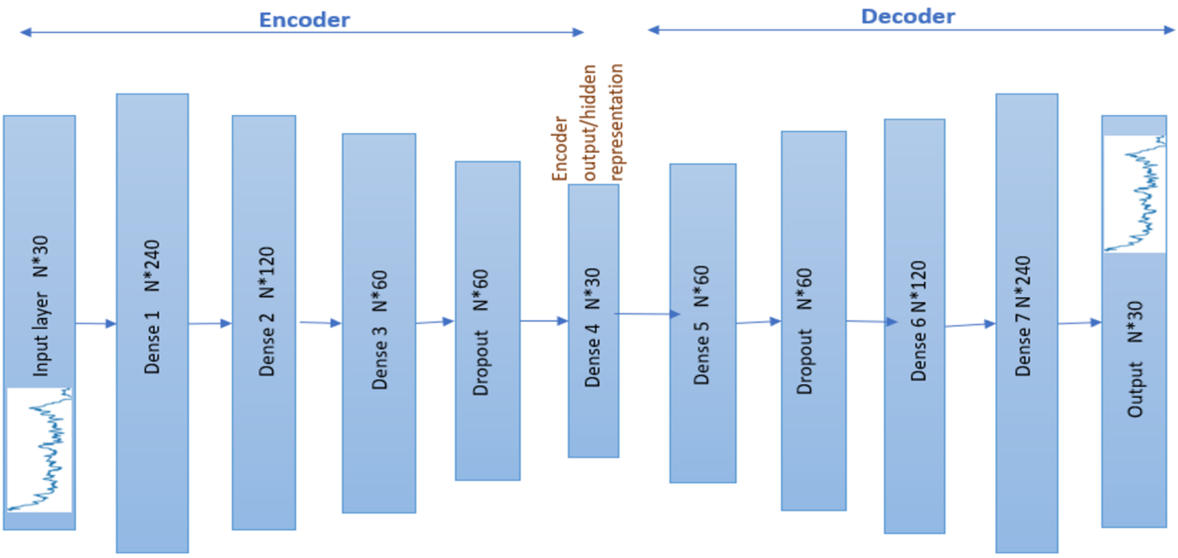

Figure 3: Detailed CAE architecture with dense layers in encoder and decoder. N denotes the number of samples

data. It comprises three dense layers with 60, 120, and 240 neurons, respectively, following the same ReLU activation and dropout configuration as in the encoder (Fig. 3). The last dense layer contains neurons matching the original input size, allowing the model to map the encoded representations back to the original data dimensions. The decoder effectively learns how to reverse the dimensionality reduction performed by the encoder.

Loss Function: The loss function for the contractive autoencoder has two key components - the reconstruction loss and the contractive regularization loss.

The reconstruction loss measures how well the model reconstructs the input data after it has been encoded and decoded. This is calculated as the mean squared error (MSE) between the input data and the reconstructed output. MSE compares the input heart rate sequences to the reconstructions and computes the average squared differences. A lower MSE indicates the model has learned to accurately reconstruct the inputs.

The contractive loss provides regularization to improve robustness and stability of the learned representations. It is calculated by first obtaining the Jacobian matrix of the model outputs with respect to the inputs using gradient tape. This Jacobian contains the gradients that measure how each output changes with respect to small changes in the input. The Frobenius norm of this Jacobian matrix gives the contractive regularization term. This penalty encourages the model to learn encodings that are contractive, meaning small changes in input only cause small changes in the encoding.

The final loss function is a weighted sum of the MSE reconstruction loss and the contractive regularization loss.

Training: The CAE model is trained by minimizing a combined loss function consisting of reconstruction error and a contractive penalty term. The optimization is performed using the Adam adaptive gradient optimizer, which computes individual adaptive learning rates for each parameter. A small learning rate of 0.00001 is initially set to control the size of update steps during training.

The model is trained by processing batches of data, where the batch size is set to 32 samples based on memory considerations and training efficiency. The number of training epochs is determined based on the size of the available training data, with smaller batches and more epochs allowing for more update steps to optimize the parameters.

The training process is implemented in Keras using the model.fit() method, which handles the underlying optimization loop. For each batch, the input data is passed through the encoder and decoder layers to reconstruct the output. The loss function then compares this reconstruction to the original input to calculate the mean squared reconstruction error. Additionally, the loss function computes the Frobenius norm of the Jacobian matrix of the encoder activations as a contractive regularization penalty. The combined reconstruction and contractive losses are differentiated using backpropagation to determine the gradients with respect to the model parameters.

These gradients are provided to the Adam optimizer, which uses them to update the model weights and biases to minimize the loss function. By iteratively repeating this process across many batches of data and epochs, the model is trained until the reconstruction error and contractive penalty are minimized, fitting the model to efficiently compress and reconstruct the input data while regularizing the internal encodings.

Callbacks like EarlyStopping are used to monitor the validation loss and halt training once the loss stops improving. The customized training process allows the CAE to learn an effective representation of the input data.

IV. Comparative Analysis

The research approach taken to detect anomalies in heartrate can be described as an inductive approach. This involves inferring a general rule from a specific set of data (the training data) and applying it to new, unseen data to identify any anomalies. The approach involves an iterative and self-reflective process [19] of designing, building, and testing a system to detect anomalies in the heart rate data, evaluating its performance, and finetuning model hyperparameters to optimize its performance. The proposed model was implemented in Anaconda3, Tensorflow 2.12, and python 3.9 (CPU version) environments. The model was built, trained and validated on a laptop computer that featured an Intel Core i5-1135G7 CPU clocked at 2.4 GHz. The data used for training is the data collected from Fitbit users. The data was pre-processed to standardize into a fixed length format and labelled into normal and anomalies based on threshold values. Around 80% of the normal data was used for training, while the remaining data was merged and used for validation and testing purposes. This partitioning ensures a robust evaluation of the model's performance.

The CAE model is compiled using the Adam optimization algorithm to minimize the defined custom loss function, comprising reconstruction error and contractive regularization penalties. Model training proceeds by feeding the training data in batches to gradually update the weights to minimize loss. The batch size is set to 32 samples and training runs for 50 epochs. The scaled training data is input as both the input and target so that the model learns to reconstruct the input. Validation data is provided to assess model performance during training. The callback *EarlyStopping* is used to halt training early if validation loss stops improving after 5 epochs. This helps prevent overfitting. The training history tracks loss metrics at each epoch. By iterating through the full training set in small batches and applying gradient-based optimization, the model minimizes the combined loss function and learns to effectively compress the input data into a stable encoding while accurately reconstructing the original input.

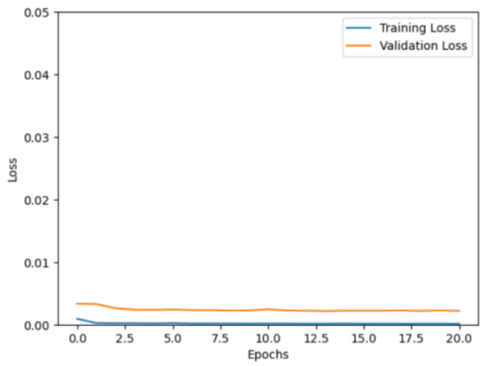

Figure 4: Training and validation loss vs Epochs

The CAE model demonstrates stable optimization and excellent performance based on the training logs. The training loss decreases steadily from an initial value of around 0.0009 down to 0.00014. This steady decrease indicates the model successfully learns to reconstruct the normal heart rate sequences. The validation loss also drops initially, reaching 0.0022-0.0023 where it plateaus. The small gap between the ending training loss of 0.00014 and validation loss of ~0.0022 shows the model generalizes well without overfitting, as depicted in Figure 4.

Training time per epoch remains consistent in the 10-14 millisecond range, demonstrating reliable model optimization. Though training continues for 50 epochs, the unchanging validation loss after 20 epochs suggests early stopping could trigger shortly after. Achieving a low validation reconstruction error of ~0.0022 highlights the model has learned robust data representations. The stable training curve, lack of overfitting, and strong validation performance validate the contractive autoencoder's ability to effectively compress the input data into an informative encoding that can accurately reconstruct the original data despite noise and variations in the real-world consumer dataset.

Figure 5 visualizes the model's reconstruction capability on a sample normal heart rate sequence. The first plot depicts the input sequence fed into the model. The second plot reveals the sequence reconstructed by the model from the encoded representations. The remarkably close alignment between the input and output sequences demonstrates that the model has successfully learned the fundamental patterns characterizing normal heart rate data, enabling it to accurately regenerate the input sequence despite compressing it into a low-dimensional encoding. This exemplifies the model's proficiency in extracting meaningful representations to precisely recreate normal heart rate sequences.

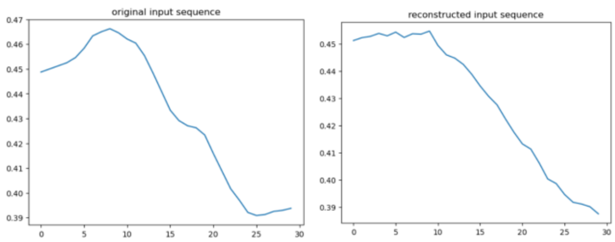

Figure 5: CAE model reconstruction of a random heart rate sequence. The first plot shows the input sequence fed into the model. The second plot shows the sequence reconstructed by the CAE model from the learned representations.

To evaluate the trained autoencoder's generalization, its reconstruction competence is validated on an unseen test set. The model reconstructs each test sequence, then the deviation between the original and predicted sequences is quantified using mean squared error. A threshold of 0.00007, identified through iterative testing of multiple values, classifies the sequences as either normal or anomalous based on the reconstruction error. Test sequences accurately reconstructed below the set threshold are designated as normal patterns learned by the model. Meanwhile, sequences poorly reconstructed and exceeding the threshold are identified as unlearned anomalies. By comparing these predicted labels to the true labels, the threshold's efficacy in distinguishing anomalies is assessed. This iterative process determined 0.00007 as the optimal threshold for segregating normal and anomalous heart rate sequences based on reconstruction error, maximizing classification performance.

The evaluation results are presented in the confusion matrix and classification report, as depicted in Figure 6 and Figure 7. The confusion matrix aggregates the predictions of the model on the test set versus the actual true labels of the test data. It provides the counts of true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN). These counts are then used to calculate the metrics as follows:

Accuracy: Accuracy refers to the overall correctness of the model's predictions. It is the ratio of the number of correct predictions to the total number of input samples.

$$Accuracy = (TP + TN) / (TP + FP + TN + FN) \quad (5)$$

Precision: Precision refers to the correctness of the model when it predicts the positive class. It is the ratio of the number of true positives to all positive predictions.

$$Precision = TP / (TP + FP) \quad (6)$$

Recall: Recall, also called sensitivity, refers to the model's ability to find all relevant positive samples. It is the ratio of true positives to all actual positive samples.

$$Recall = TP / (TP + FN) \quad (7)$$

F1 Score: F1 Score is the harmonic mean of precision and recall. It balances both metrics into a single measure of a model's performance.

$$F1 = 2 * (Precision * Recall) / (Precision + Recall) \quad (8)$$

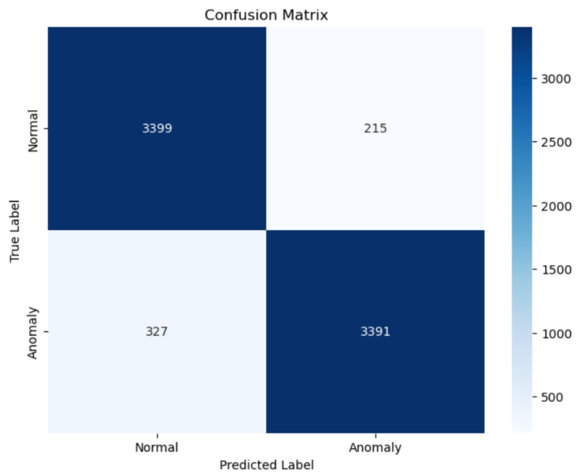

Figure 6: Confusion matrix demonstrating performance of CAE model in classifying normal vs. anomalous heart rate sequences.

In summary, accuracy measures overall performance, precision measures correctness of positive predictions, recall measures finding all positive samples, and F1 score balances precision and recall into a single metric. These are important metrics for evaluating classification models.

The classification report takes the raw counts from the confusion matrix and turns them into handy precision, recall and F1 metrics for each class. This provides a detailed breakdown of model performance across different output categories, as depicted in Table 1.

The CAE model achieves an overall accuracy of 93% in discriminating between normal and anomalous heart rate sequences. The precision of 0.94 for detecting anomalies indicates a low false positive rate, while the recall of 0.91

shows the model correctly identifies the vast majority of anomalies.

Table 1: Performance matrix for classification of anomalous versus normal heart rate sequences.

	Precision	Recall	F1-Score
Anomaly	0.94	0.91	0.93
Normal	0.91	0.94	0.93
Accuracy			0.93

The F1-score, balancing precision and recall, reaches 0.93 for anomalies. Similarly for normal sequences, precision of 0.91 and recall of 0.94 result in a strong 0.93 F1-score. The balanced performance across classes is further evidenced by the macro-average F1 of 0.93. Additionally, with support of 3718 anomalies and 3614 normal sequences, the model generalizes well to both classes. The consistent metrics across anomaly and normal categories, also reflected in the weighted average scores, demonstrate the model's competence in recognizing both normal and abnormal sequence patterns. The high accuracy, precision, recall and F1-scores validate the effectiveness of the autoencoder-based anomaly detection approach on unseen heart rate data.

Figure 7 visualizes the variation in reconstruction error on the test set between normal and anomalous sequences. The normal heart rate patterns, depicted in green, display a tight clustering of lower reconstruction errors, confirming the model's proficiency in accurately reconstructing these regular sequences. However, the anomalous sequences represented in red demonstrate a broader spread of predominantly larger error values. The distinct error distributions confirm the model's identification of anomalies based on their poor reconstruction compared to normal samples.

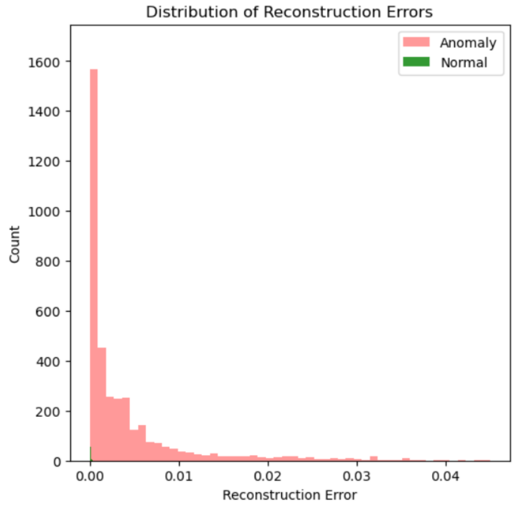

Figure 7: Distribution of reconstruction errors for normal and anomalous heart rate sequences in test data.

The receiver operating characteristic (ROC) curve and area under the curve (AUC) metric are computed to evaluate the model's anomaly detection performance, as depicted in Figure 8. The true labels are converted to numeric values to compute the ROC curve.

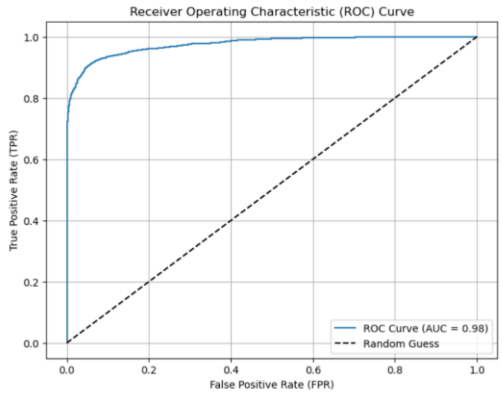

Figure 8: ROC curve highlighting model's tradeoff between true and false positives.

The model reconstructs the test set and calculates the reconstruction error per sample. These errors and true labels are used to derive the AUC, quantifying the model's separation of anomalies. A high AUC of 0.98 indicates excellent anomaly detection. The roc_curve() function generates false positive and true positive rates for different thresholds to plot the ROC curve. The curve's proximity to the upper left corner demonstrates strong performance. Additionally, the random guess line assists interpretation. Together, the ROC curve and high AUC of 0.98 confirm the effectiveness of the autoencoder-based anomaly detector on the heart rate data.

V. Conclusion

This study presented a novel CAE model for heart rate anomaly detection, showcasing its potential in handling noise and variability challenges in widely available consumer-grade wearable data. The CAE demonstrates high accuracy, precision, and recall in anomaly detection from noisy wearable data. The model was trained on a dataset of normal heart rate samples, and its performance was evaluated on a separate test dataset which contains both normal and anomalous data. The results demonstrated that the CAE model achieved a high accuracy of 90% on the test data, with balanced precision and recall scores for both the 'Anomaly' and 'Normal' classes. The low loss values during training indicated that the model effectively learned the underlying representations in the data and achieved excellent reconstruction performance. The area under the ROC curve reached 0.98, further validating anomaly detection proficiency. The reconstruction error distributions clearly differentiated anomalies on unseen test data. Building on these promising anomaly detection capabilities, this study lays the foundation for ancillary innovations like personalized interventions based on predicting future heart rate trajectories.

However, the evaluation was limited to only one dataset from a single source. More robust validation on larger and more diverse datasets is needed to fully assess the model's capabilities. Additionally, the labelling of anomalies was done manually by setting a threshold, which may miss outliers that fall within the normal range.

While the presented model showed promise, future work should focus on exploring alternative neural network architectures and incorporating advanced techniques, such as recurrent or attention-based models, may further enhance its performance. One exciting direction for future research is extending the CAE model for time-series forecasting to enable proactive healthcare monitoring and intervention. With continuous advancements in anomaly detection and data availability, the CAE model lays the groundwork for innovative approaches in healthcare anomaly detection and personalized medical interventions. However, obtaining properly labelled anomaly data posed a key challenge. Thus, the next research phase demands collecting and labelling real-world data.

Overall, the model has demonstrated its potential as an effective tool for heart rate anomaly detection, offering valuable contributions to the field of healthcare monitoring.

VI. Acknowledgement

We gratefully acknowledge the generous support and funding provided by the Digihealth Asia project (https://digihealth-asia.eu/), which has been instrumental in enabling us to conduct this impactful research.

VII. References

[1] C. S. McLachlan and H. Truong, “A narrative review of commercial platforms offering tracking of heart rate variability in corporate employees to detect and manage stress,” Journal of Cardiovascular Development and Disease, vol. 10, no. 4, p. 141, 2023. doi:10.3390/jcdd10040141

[2] G. M. Peters et al., “Detecting patient deterioration early using continuous heart rate and respiratory rate measurements in hospitalized COVID-19 patients,” Journal of Medical Systems, vol. 47, no. 1, 2023. doi:10.1007/s10916-022-01898-w

[3] P. C. Demkowicz et al., Physician responses to Apple Watch-detected irregular rhythm alerts: A case-based survey, 2022. doi:10.1101/2022.08.02.22278237

[4] N. Alugubelli, H. Abuissa, and A. Roka, “Wearable devices for remote monitoring of heart rate and heart rate variability—what we know and what is coming,” Sensors, vol. 22, no. 22, p. 8903, 2022. doi:10.3390/s22228903

[5] S. Liu et al., “Fitbeat: Covid-19 estimation based on wristband heart rate using a contrastive convolutional auto-encoder,” Pattern Recognition, vol. 123, p. 108403, 2022. doi:10.1016/j.patcog.2021.108403

[6] F. F. Abir et al., “PCovNet+: A CNN-VAE anomaly detection framework with LSTM embeddings for smartwatch-based COVID-19 detection,” Engineering Applications of Artificial Intelligence, vol. 122, p. 106130, 2023. doi:10.1016/j.engappai.2023.106130

[7] Yang Xie et al., “Prediction of chronic obstructive pulmonary disease exacerbation using physiological time series patterns,” 2013 35th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC), 2013. doi:10.1109/embc.2013.6611114

[8] S. Bae, A. K. Dey, and C. A. Low, “Using passively collected sedentary behavior to predict hospital readmission,” Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing, 2016. doi:10.1145/2971648.2971750

[9] S. Rifai, P. Vincent, X. Muller, X. Glorot, and Y. Bengio, "Contractive auto-encoders: Explicit invariance during feature extraction," In Proceedings of the 28th International Conference on Machine Learning (ICML-11), pages 833–840, 2011.

[10] W. Wang, X. Du, D. Shan, R. Qin, and N. Wang, “Cloud intrusion detection method based on stacked contractive auto-encoder and support vector machine,” IEEE Transactions on Cloud Computing, vol. 10, no. 3, pp. 1634–1646, 2022. doi:10.1109/tcc.2020.3001017

[11] B. Diallo et al., “Deep embedding clustering based on contractive autoencoder,” Neurocomputing, vol. 433, pp. 96–107, 2021. doi:10.1016/j.neucom.2020.12.094

[12] E. Q. Wu, X. Y. Peng, C. Z. Zhang, J. X. Lin, and R. S. Sheng, “Pilots’ fatigue status recognition using deep contractive Autoencoder Network,” IEEE Transactions on Instrumentation and Measurement, vol. 68, no. 10, pp. 3907–3919, 2019. doi:10.1109/tim.2018.2885608

[13] M. Radmanesh, A. A. Rezaei, M. Jalili, A. Hashemi, and M. M. Goudarzi, “Online spike sorting via deep contractive autoencoder,” Neural Networks, vol. 155, pp. 39–49, 2022. doi:10.1016/j.neunet.2022.08.001

[14] R. Banerjee, A. Mukherjee, and A. Ghose, “Noise cleaning of ECG on edge device using convolutional sparse contractive Autoencoder,” 2022 IEEE International Conference on Pervasive Computing and Communications Workshops and other Affiliated Events (PerCom Workshops), 2022. doi:10.1109/percomworkshops53856.2022.9767313

[15] M. Gu et al., “A lightweight convolutional neural network hardware implementation for wearable heart rate anomaly detection,” Computers in Biology and Medicine, vol. 155, p. 106623, 2023. doi:10.1016/j.compbiomed.2023.106623

[16] J. Zhong et al., “Convolutional autoencoding and gaussian mixture clustering for unsupervised beat-to-beat heart rate estimation of electrocardiograms from wearable sensors,” Sensors, vol. 21, no. 21, p. 7163, 2021. doi:10.3390/s21217163

[17] A. Blázquez-García, A. Conde, U. Mori, and J. A. Lozano, “A review on outlier/anomaly detection in time series data,” ACM Computing Surveys, vol. 54, no. 3, pp. 1–33, 2021. doi:10.1145/3444690

[18] Julenaranguren, “Bellabeat - case study,” Kaggle, https://www.kaggle.com/code/julenaranguren/bellabeat-case-study (accessed Aug. 6, 2023).

[19] K. Lewin, “Action research and minority problems (1946).,” Resolving social conflicts and field theory in social science., pp. 143–152. doi:10.1037/10269-013

[20] D. Bank, N. Koenigstein, and R. Giryes, Autoencoders, 2020. doi:10.1017/9781108955652.006

[21] M. Aamir, N. Mohd Nawi, F. Wahid, and H. Mahdin, “A deep contractive autoencoder for solving multiclass classification problems,” Evolutionary Intelligence, vol. 14, no. 4, pp. 1619–1633, 2020. doi:10.1007/s12065-020-00424-6

[22] S. Rifai, P. Vincent, X. Muller, X. Glorot, and Y. Bengio, Contractive Auto-Encoders: Explicit Invariance During Feature Extraction, 2011.

[23] G. E. Hinton, R. S. Zemel. Autoencoders, minimum description length, and Helmholtz free energy. Advances in Neural Information Processing Systems 6. J. D. Cowan, G. Tesauro and J. Alspector (Eds.), Morgan Kaufmann: San Mateo, CA.

[24] L. Yasenko, Y. Klyatchenko, and O. Tarasenko-Klyatchenko, “Image noise reduction by denoising autoencoder,” 2020 IEEE 11th International Conference on Dependable Systems, Services and Technologies (DESSERT), 2020. doi:10.1109/dessert50317.2020.9125027

[25] Julenaranguren, “Bellabeat - case study,” Kaggle, https://www.kaggle.com/code/julenaranguren/bellabeat-case-study (accessed Sep. 8, 2023).

[26] D. Carrera, B. Rossi, P. Fragneto, and G. Boracchi, “Online anomaly detection for long-term ECG monitoring using wearable devices,” Pattern Recognition, vol. 88, pp. 482–492, 2019. doi:10.1016/j.patcog.2018.11.019