A Comparative Analysis of Arrhythmia Detection Using Convolutional neural network and Support Vector Machines in ECG Signals

Kornkamol Sampaongern
kornkamol.sampaongern@city.ac.uk

1. Introduction

Arrhythmias, irregular heart rhythms, are considered as significant health concerns, affecting up to 5% of the population and contributing to thousands of fatalities annually. Early detection is crucial for intervention and treatment planning. This study compares the performance of Convolutional Neural Network (CNN) and Support Vector Machines (SVM) algorithms in classifying various types of Arrhythmias, exploring different configurations to achieve accurate classification.

2. Dataset and Initial Data Analysis

This study obtained the ECG Arrhythmia Dataset from Kaggle [1], derived from MIT-BIH arrhythmia database. The dataset contains 91,935 records with 187 time points sampled at 125 Hz. The dataset is divided into five classes: N (Normal and Escape Rhythms), S (Atrial and Supra-ventricular Premature Contractions), V (Ventricular Arrhythmias), F (Fusion Complexes), Q (Paced and Unclassifiable Patterns), as described in [2].

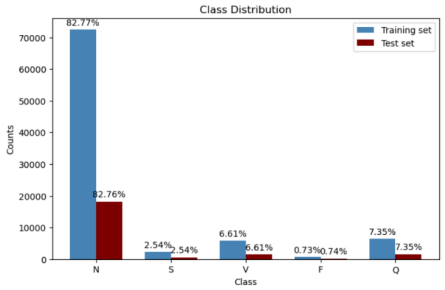

Fig 1: Class distribution on train and test set

Normalization is unnecessary as signal amplitudes range from 0 to 1. Also,there is no missing data, as it was zero-padded to ensure consistent lengths. Class distribution indicates imbalanced data in both the training and test sets, with Class N comprising over 80% of the dataset, while the remaining classes each represent less than 10%.

3. The summary of implementing models

3.1 Support Vector Machines (SVM)

SVM is a supervised machine learning algorithm, primarily used for binary classification tasks by finding an optimal decision boundary or hyperplane to maximize the margin between the nearest data points of different classes, which are called support vectors. SVM is flexible and can handle both linear and non-linear tasks, by leveraging versatile kernel tricks that map the complex input data into higher dimensional space where the classes can be separable. While SVMs may face challenges with very large datasets due to memory requirements and the complexity of kernel matrix computations, they are advantageous for handling high-dimensional data with the ability to focus on relevant support vectors, allowing them to generalize well even in high-dimensional spaces.

3.2 Convolutional neural network (CNN)

CNN is a type of deep learning algorithm, widely used for grid-like data such as image recognition and time-series data. Comprising convolutional, pooling, and fully connected layers, CNNs apply filters to extract relevant features from input data, subsequently reducing dimensions of feature maps and making decisions based on these features. Through the backpropagation process, CNNs iteratively adjust parameters to minimize loss during training. Unlike SVM, CNN can automatically learn hierarchy representation and spatial patterns from raw data. Their complex structures enable them to handle large and intricate datasets effectively, while techniques like dropout and weight decay help prevent overfitting. However, CNNs are computationally intensive, particularly for large datasets.

Also, CNNs are not suitable for small datasets, as they are likely to overfit due to the model memorizing limited features from the input data rather than learning generalized patterns.

4. Hypothesis statement

Comparing performance between SVM and CNN in classifying ECG data, it is believed that CNN may outperform SVM, as suggested by findings in previous studies [2], [3], [4], [5]. This could be attributed to the ability of CNN to learn hierarchical representations and spatial patterns which is beneficial in analyzing time series data such as ECG signals. Additionally, SVM may require shorter computational time during parameter tuning and training with its simpler architecture.

However, It is also expected that CNN's predictive time could be faster than SVM, due to its utilization of CUDA GPU parallel processing capabilities. Moreover, considering the risk of overfitting, it is suggested that SVM could have more risk of overfitting in these extensive records compared to CNN, as the ability of CNN to learn complex patterns, potentially leading to more robust generalization.

5. Description of the choice of training and evaluation methodology

This section provides a detailed description of the experimental procedures, including the utilized models, architectures, parameters, and evaluation.

3.1 Data partitioning

The dataset was initially divided into training and testing sets by the source provider [1], with 21,892 records (20% of total) reserved for testing, and 87,554 records (80% of the total) for training. The test set is kept unseen throughout the experiment until the final evaluation to compare algorithm performance. Within the training set, a further partition was made for validation purposes in CNN model to evaluate model performance during the backpropagation process. This validation subset contained 20% of the training data. Consequently, the training data used in CNN model remained at 70,043 records, while SVM utilized all 87,554 records for training.

3.2 Hyperparameter tuning

In the hyperparameter tuning process, gridsearch is employed alongside stratified 5-fold cross-validation to identify the optimal parameters for both CNN and SVM, ensuring an equitable distribution of data. The combination of parameters will be evaluated on average of 5 different subsets of data. Then an F1 score will be averaged across the folds to evaluate overall performance of each combination of parameters to obtain the optimal parameters for the model by balancing between minimizing error and preventing overfitting.

After selecting the parameters, both models are trained using the optimal parameters on the training set. To address overfitting, different strategies are employed. In addition to the optimized regularization parameter (C) obtained from grid search, SVM model is trained with the maximum allowable number of iterations. Meanwhile, CNN model applies early stopping criteria, focusing on the validation set to stop training when the validation loss exceeds a predefined threshold. These techniques aim to mitigate the risk of overfitting and enhance the generalization capabilities of the models.

3.3 Evaluation

Finally, the best trained models for both SVM and CNN are evaluated using a test set. The evaluation matrices include accuracy, precision, recall, and F1 score. Additionally, the confusion matrix is utilized to provide detailed information of correct and incorrect predictions for each class. Time regarding parameter selection, training, and prediction are also taken into account. Furthermore, the ROC curve, along with AUC score is used to assess the models' ability to discriminate among different classes.

6. Choice of parameters and experimental results

6.1 Choice of parameters during hyper parameter tuning through gridsearch

Table 1 presents a subset of parameter combinations from the selection process through 5-fold gridsearch.

Although CNN model involves numerous parameters in the structure, this study selects to tune hidden size and dropout rate using gridsearch due to limited computational resources. However, other related parameters undergo thorough exploration and selection through sophisticated experiments. Conversely, all parameters of SVM, including kernel type, regularization parameter (C), and gamma value are optimized using gridsearch. The results reveal that the optimal parameters for the CNN model are a hidden size of 128 and a dropout rate of 0.1, achieving the highest F1 score of 0.94. Meanwhile, the optimal parameters for the SVM model achieve an F1 score of 0.9, employing the RBF kernel with a C value of 100 and a gamma value of 1.

CNN			SVM			
Hidden Size	**Dropout**	**Val F1 macro**	**Kernel**	**C**	**Gamma**	**Val F1 macro**
64	0.1	0.93	Linear	10	0.1	0.19
128	**0.1**	**0.94**	Linear	100	0.1	0.20
256	0.1	0.92	Linear	1000	0.1	0.20
64	0.2	0.93	Linear	10	1	0.19
128	0.2	0.93	Linear	100	1	0.20
256	0.2	0.93	Linear	1000	1	0.20
64	0.3	0.93	RBF	10	0.1	0.40
128	0.3	0.93	RBF	100	0.1	0.65
256	0.3	0.92	RBF	1000	0.1	0.62
64	0.4	0.92	RBF	10	1	0.89
128	0.4	0.92	**RBF**	**100**	**1**	**0.90**
256	0.4	0.92	RBF	1000	1	0.89

Table 1: Hyper parameter tuning through gridsearch 5-folds cross validation

6.2 Model architecture : CNN

Figure 2 illustrates the network architecture proposed in this study with five convolutional blocks. Each block consists of a 1D convolutional layer with a kernel size of 3 and padding. Following the convolutional layer, there is a batch normalization layer, GELU activation function, and a max-pooling layer with a kernel size of 3 and a stride of 2. The output is then passed through an adaptive max-pooling layer and a flatten layer. These local features are then fed into a fully connected layer with batch normalization and GELU activation, consisting of 128 hidden neurons. To mitigate overfitting, a dropout layer with a dropout rate from gridsearch of 0.1 is applied before the data is passed to the second fully connected layer with Softmax to produce outputs for classification.

In this experiment, PyTorch and Skorch are utilized. The optimization is carried out using the SGD optimizer with a learning rate of 0.05, momentum of 0.9 to accelerate convergence, and weight decay at 0.0001 to prevent overfitting.

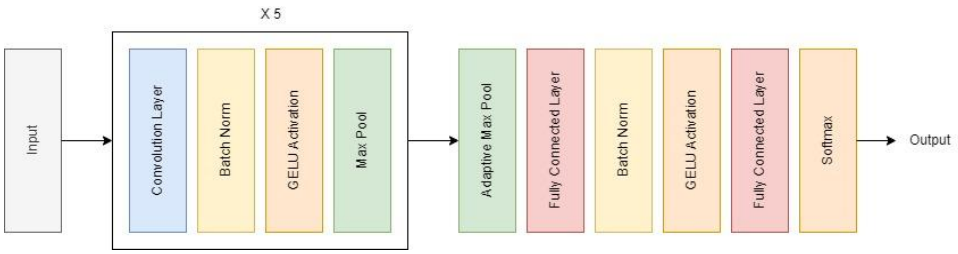

Fig 2: Convolution Neural Network Structure

The model is trained with a batch size of 128 and a maximum of 100 epochs to minimize computational resources. A reduced learning rate scheduler is employed to adjust the learning rate during training for improved performance and convergence at factor 0.5.

Additionally, early stopping is utilized to prevent overfitting, stopping training when the validation loss does not improve for 15 consecutive iterations. The CrossEntropyLoss function is used as the loss function during training.

6.3 Model architecture : SVM

While SVM is originally designed for binary classification tasks, it can also be adapted for multi-class classification using the OVR approach where each class has a binary classifier to distinguish it from all other classes.

SVM relies on three key parameters to customize the model to different datasets and problem characteristics, including kernel, C, and gamma. Throughout the experiment, the Support Vector Classification library from Scikit-learn is employed, with the following parameters obtained through grid search: RBF kernel, C of 100 and gamma of 1.

6.4 Experimental results

This section presents the results obtained from the experiments for both models, including evaluation matrices, time duration, confusion matrices, and ROC curve.

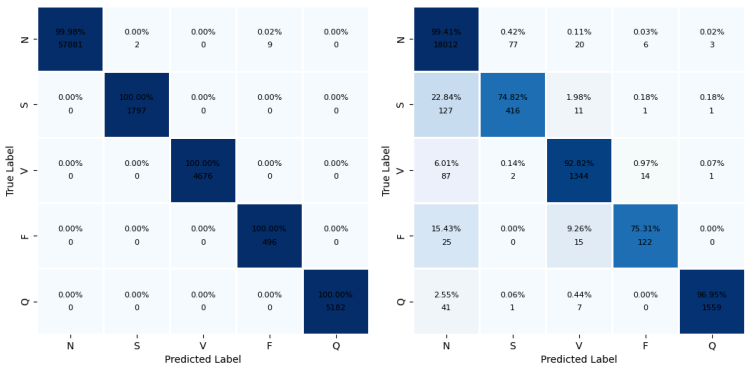

Fig 3: SVM matrix on train(left) and test(right) set

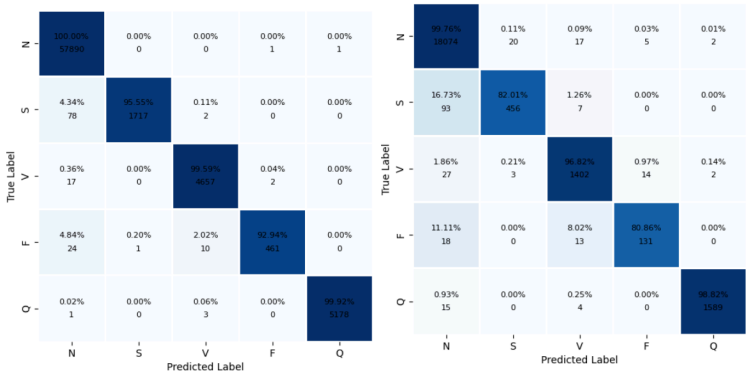

Fig 4: CNN matrix on train(left) and test(right) set

	SVM				CNN			
Class	**Accuracy**	**Precision**	**Recall**	**F1**	**Accuracy**	**Precision**	**Recall**	**F1**
N	0.98	0.98	0.99	0.99	0.99	0.99	1.00	0.99
S		0.84	0.75	0.79		0.95	0.82	0.88
V		0.96	0.93	0.94		0.97	0.97	0.97
F		0.85	0.75	0.80		0.87	0.81	0.84
Q		1.00	0.97	0.98		1.00	0.99	0.99
Macro avg		**0.93**	**0.88**	**0.90**		**0.96**	**0.92**	**0.94**

Table 2: Performance metrics for SVM and CNN

	Parameter selection	Training	Prediction
SVM	**1.37 h**	**0.07 h**	**50.77 s**
CNN	**33.67 h**	**0.25 h**	**0.015 s**

Table 3: Time Durations for Model Training Process

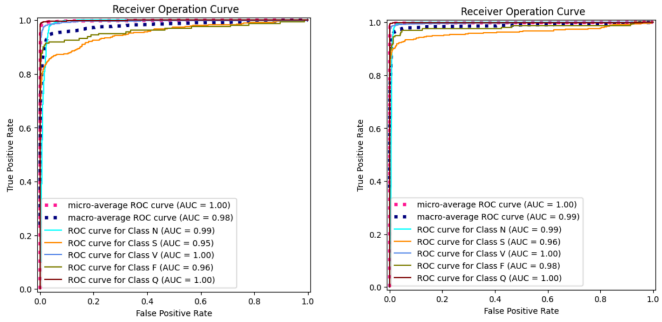

Figure 5, 6: ROC Curve for SVM (left), CNN (right)

7. Analysis and critical evaluation of results

7.1 Comparison on algorithms

The results obtained from parameter selection in Table 1, underscore the significant impact of parameter combinations on SVM performance with value varying drastically. While the linear kernel achieves a 20% F1 score, the RBF kernel significantly enhances performance to 90%.

This improvement can be attributed to the non-linear nature of the data, which necessitates more flexible decision boundaries provided by the RBF kernel. Lower gamma leads to a smoother boundary, potentially causing underfitting, which results in a lower F1 score of 65%. Similarly, a lower c value allows for more misclassifications, also leading to underfitting. Conversely, hidden neuron size and dropout rate in CNN model have minimal impact on model performance, but higher dropout rates lead to slight performance decline due to learning capacity limitations. Also, higher hidden neuron size has no significant improvement.

The confusion matrix of the SVM model, shown in figure 3, indicates signs of overfitting in the training set, with nearly 100% correct across all classes. In the test set, while SVM accurately identifies the majority class N, as well as classes Q and V, with 99%, 97%, and 93% correct respectively. However, it struggles with classes S and F, which have the least training data, comprising only 2% and less than 1% respectively. This results in mispredictions of around 25%, even with the application of class weights to make models pay more attention to minority classes.

In contrast, the confusion matrix shown in Figure 4 for CNN model reveals minimal drops in performance compared to the training set, with approximately a 10% decrease in accuracy for classes S and F, achieving 82% and 80% respectively in the test set. Additionally, CNN demonstrates superior performance over SVM in other classes, achieving accuracy rates of 100%, 99%, and 97% for classes N, Q, and V respectively. The CNN model's ability to learn from hierarchical features and its more complex architecture allow it to effectively handle complexity and variation in intricate patterns, particularly in highly imbalanced datasets, resulting in better generalization to unseen data compared to SVM.

Table 2 displays evaluation metrics for SVM and CNN, including accuracy, precision, recall, and F1 score. Both models achieve high accuracy, with SVM at 98% and CNN at 99%. While both models perform well for the majority class N, CNN outperforms SVM in all classes, achieving overall accuracy 99% and F1-score at 94%, compared to 98% accuracy and F1 score at 90% for SVM.

Additionally, ROC curve with OVR strategy is used to assess the model’s quality by visualizing the trade-off between true positive and false positive rates.

As shown in figure 5, both models show low false positive rate (close to 0), and high true positive rate (close to 1), indicating proficient discrimination among classes. Although both models demonstrate high AUC scores across all classes, the CNN model shows superior AUC scores, particularly in minority classes, with 0.96 for class S and 0.98 for class F, compared to 0.95 and 0.96 respectively in the SVM model. This underscores the higher ability of CNN for class differentiation.

In terms of computational time, CNN requires extensive resources, especially during parameter tuning with grid search and training through backpropagation. This is primarily due to its complex architecture and numerous parameters, leading to longer processes compared to SVM. However, CNN demonstrates faster prediction times than SVM, due to its utilization of CUDA GPU parallel processing.

While the CNN model requires more computational resources compared to SVM, its superior generalization and performance, particularly for minority classes, make it the preferred choice for this task. Given the critical consequences of mispredictions in medical contexts, where misidentifying Arrhythmia types can lead to significant loss and incorrect treatment plans, prioritizing model performance is necessary.

7.2 Comparison on existing works on arrhythmia ECG classification

Table 3 provides a comparative overview of existing studies on arrhythmia classification utilizing the data from MIT-BIH arrhythmia database, focusing on average accuracy. Notably, the proposed SVM and CNN models in this study outperform previous methodologies, achieving higher accuracy. Consistently, the CNN models demonstrate superior accuracy compared to the SVM models, aligning with the results obtained in this study.

The proposed CNN architecture, with five convolutional layers and two fully connected layers, surpasses the structures with 2 convolutional layers and 1 fully connected layer proposed by Kachuee et al. [6] and the residual CNN model by Ullah et al. [7]. This highlights the effectiveness of the proposed architecture in accurately classifying Arrhythmia ECG signals, resulting in superior performance on this challenging classification task.

Author	Classes	Model	Accuracy
Melgani et al. [4]	6	SVM	88.14%
Joshi et al. [5]	16	SVM	86.00%
Kachuee et al. [6]	5	1D-CNN	95.90%
Ullah et al. [7]	5	1D-CNN	97.38%
Proposed	**5**	**SVM**	**97.99%**
Proposed	**5**	**1D-CNN**	**98.90%**

Table 4: Comparison on existing studies on Arrhythmia ECG classification

8. Lessons learned and future work

An important consideration in model selection for specific tasks is balancing between model simplicity and performance. While SVM offers simplicity and lower computational resources, CNN provides superior performance, which is more important in medical context. Addressing the challenge of imbalanced data requires careful attention to model options, regularization techniques, and fine-tuning parameters to prevent overfitting while optimizing error minimization and generalization. Also, choosing appropriate evaluation metrics is crucial. Although both models show similar accuracies in this study, leveraging other evaluation tools like confusion matrices and ROC curves can provide deeper insights into performance differences.

In future work, a deeper understanding of the arrhythmia ECG pattern will be essential to effectively identify noise and outliers. For SVMs, employing ensemble methods by training multiple SVMs on different subsets of data could be beneficial. Conversely, for CNNs, comprehensive hyperparameter tuning is essential. While this study focused on tuning only two parameters due to resource constraints, CNNs typically require fine tuning across numerous variables to optimize performance for specific tasks. Furthermore, limited data in the minority class is a crucial issue. Leveraging transfer learning to utilize pre-trained models, trained on large datasets can significantly enhance pattern recognition in smaller datasets, effectively addressing this issue.

9. References

[1] “ECG Heartbeat Categorization Dataset,” Kaggle, May 31, 2018. https://www.kaggle.com/datasets/shayanfazeli/heartbeat/data

[2] F. Melgani and Y. Bazi, “Classification of electrocardiogram signals with support vector machines and particle swarm optimization,” IEEE Transactions on Information Technology in Biomedicine, vol. 12, no. 5, pp. 667–677, Sep. 2008, doi: 10.1109/titb.2008.923147.

[3] Joshi, N. P., & Topannavar, P. S. (2014, May). Support vector machine based heartbeat classification. In Proc. of 4th IRF Int. Conf (pp. 140-144)

[4] A. Ullah, S. U. Rehman, S. Tu, R. M. Mehmood, F. Fawad, and M. Ehatisham-Ul-Haq, “A Hybrid Deep CNN Model for Abnormal Arrhythmia Detection Based on Cardiac ECG Signal,” Sensors, vol. 21, no. 3, p. 951, Feb. 2021, doi: 10.3390/s21030951.

[5] M. Kachuee, S. Fazeli and M. Sarrafzadeh, "ECG Heartbeat Classification: A Deep Transferable Representation," 2018 IEEE International Conference on Healthcare Informatics (ICHI), New York, NY, USA, 2018, pp. 443-444, doi: 10.1109/ICHI.2018.0009

Appendix 1 – glossary

Term	Definition
Electrocardiogram (ECG)	A record of the heart's electrical activity over a period of time.
Gaussian Error Linear Unit (GELU)	An activation function known for its smooth gradient characteristics.
Stochastic gradient descent (SGD)	An optimization algorithm for training models by updating parameters using mini-batch gradients.
One-vs-Rest (OVR)	A strategy to handle multiclass classification problem by having multiple binary classifiers with each focused on separating one class from others.
Radial basis function (RBF)	A kernel function commonly in SVM enables non-linear classification.
Receiver operating characteristic (ROC)	A visualization evaluating a model's ability to distinguish between different classes.
Area under the ROC Curve (AUC)	A metric from the ROC curve that quantifies the model's ability to distinguish between classes.

Appendix 2 – Intermediate results and Implementation details

- **Additional model architecture**

This paper originally experimented with the Residual CNN structure and parameters, proposed by M. Kachuee et al. [7], as structure shown in Figure 7.

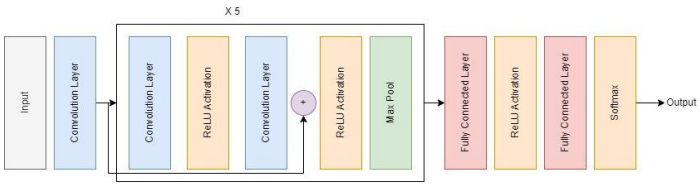

Fig 7: Residual Convolution Neural Network Structure by M. Kachuee et al.

While the input data is the same to this study, certain preprocessing steps, such as augmentation details, were not provided in their paper. Furthermore, the authors augmented the entire dataset, including both train and test sets, whereas this study only augmented the train set to preserve the integrity of the test set, ensuring it accurately represents real ECG signals. Additionally, the authors structured their model using Keras and TensorFlow, while this study utilized PyTorch and Skorch, potentially leading to differences in model performance, as outlined in Figure 8.

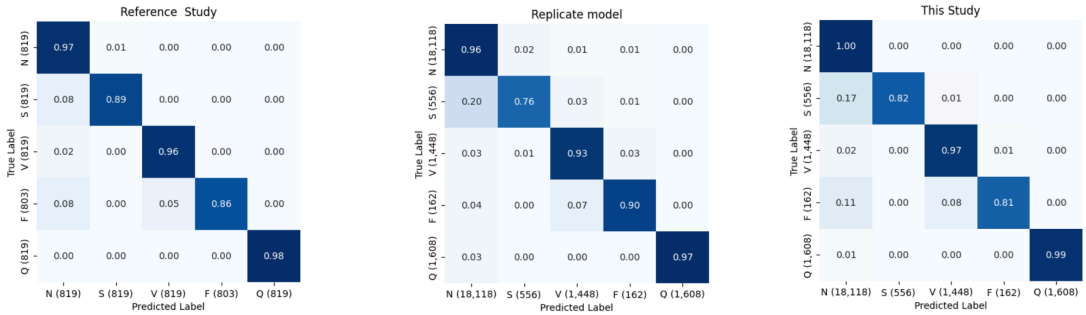

Fig 8: Confusion matrices from reference study [7] (left), replicated experiment (center), proposed 1D-CNN structure (right)

By replicating the structure and parameters, the model cannot capture the pattern of class S effectively. In response, this study studies a new CNN architecture comprising five convolutional layers and two architectures, detailed in the report. This new architecture generally demonstrates superior performance on most classes, as shown in figure 8 (right figure).

- **Additional model architecture**

Throughout this study, various preprocessing techniques and model structures were explored to optimize performance in detecting ECG patterns as shown in table 3.

Approach	Accuracy	Precision	Recall	F1
CNN + Balancing	0.09	0.28	0.22	0.06
CNN + Class weight	0.97	0.80	0.93	0.85
CNN + Noise	0.98	0.84	0.88	0.91
CNN + ReLU	0.96	0.82	0.91	0.85
CNN + LeakyReLU	0.95	0.79	0.92	0.84
SVM + Balancing	0.97	0.87	0.89	0.88
SVM + PCA	0.98	0.92	0.88	0.90
SVM + LDA	0.46	0.29	0.32	0.24

Table 3: Additional approaches with evaluation matrices on test set

The SVM results indicate challenges in handling imbalanced data, particularly minority classes. To address this, balancing methods with oversampling minority classes and downsampling majority classes were employed. However, it led to decreased performance, due to information loss and overfitting. SMOTE is also tested and shows lower performance, likely because synthetic data may not accurately represent true ECG patterns. Additionally, feature reduction techniques, such as Principal component analysis (PCA) and Linear Discriminant Analysis (LDA) are explored to enhance SVM performance. Despite setting the number of components based on explained variance, applying LDA with 4 components results in the worst accuracy, possibly due to feature selection bias and dimensionality reduction effects. Similarly, 100 components with PCA showed slightly decreased performance.

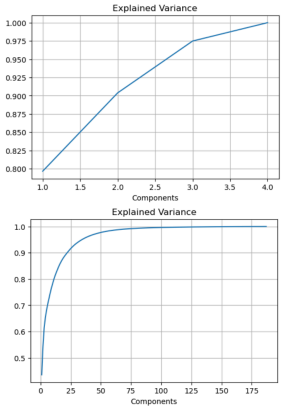

Fig 9: Explained Variance in LDA and PCA (below)

Similarly, balancing in CNN model results in lower accuracy due to information loss and overfitting. To make the model more generalize, gaussian noise is used to augment ECG source data resulting in slightly decreased performance. This can be attributed to signal distortion, leading to information loss.

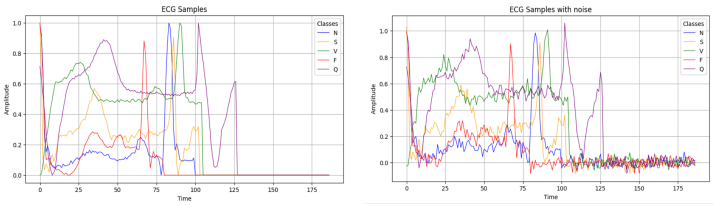

Fig 10: ECG signals before and after augmentation

Unlike SVM, providing class weight in CNN does not enhance the performance. This causes the model to prioritize minimizing the loss related to minority class, resulting in increased fluctuations during training. Consequently, this disrupts the learning process and may lead to suboptimal solutions. As shown in figure 11.

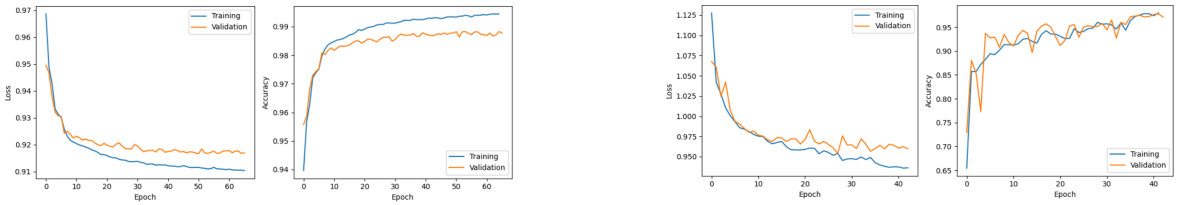

Fig 11: Validation loss and accuracy during training process before (left), and after (right) applied class weights

Additionally, GELU activation function proves to be suitable for this task, significantly outperforming ReLU and LeakyReLU. This is because GELU maintains non-zero gradients for negative inputs, preventing vanishing gradient problems and also provides smoother gradient flow, leading to more effective training and better performance. The model architecture was selected through experimentation, with the optimal configuration determined to be 5 convolutional and 2 fully connected layers. Fewer convolutional layers led to decreased accuracy, while more layers did not improve performance. Similarly, having more than 2 fully connected layers caused overfitting, while fewer layers resulted in inaccurate predictions. Additionally, parameters such as the optimizer, learning rate, scheduler, momentum, and weight decay were selected based on their performance in experiments.