JURNAL INFOTEL
Informatics - Telecommunication - Electronics
Website Jurnal : http://ejournal.st3telkom.ac.id/index.php/infotel
ISSN : 2085-3688; e-ISSN : 2460-0997

Construction of cardiac arrhythmia prediction model using deep learning and gradient boosting

Dhanar Bintang Pratama[1*], Favian Dewanta[2], Syamsul Rizal[3]
[1,2,3] School of Electrical Engineering, Telkom University
[1,2,3] Jl. Telekomunikasi, Terusan Buah Batu, Bandung 40257, West Java, Indonesia
* Corresponding email: pratama.dhanarbintang@gmail.com

Received 19 June 2021, Revised 22 July 2021, Accepted 20 August 2021

Abstract — Arrhythmia is a condition in which the rhythm of the heartbeat becomes irregular. This condition in extreme cases, can lead to fatal heart attack accidents. Therefore, to reduce heart attack risk, appropriate early treatments should be conducted right after getting results of the Arrhythmia condition, which is generated by electrocardiography ECG tools. First, however, reading ECG results should be done by qualified medical staff in order to diagnose the existence of arrhythmia accurately. This paper proposes a deep learning algorithm method to classify and detect the existence of arrhythmia from ECG reading. Our proposed method relies on Convolutional Neural Network (CNN) to extract feature from a single lead ECG signal and the Gradient Boosting algorithm to predict the final outcome of single-lead ECG reading. This method achieved an accuracy of 96.18% and minimized the number of parameters used in the CNN Layer.

Keywords – Arrhythmia, Electrocardiography, deep learning, Convolutional Neural Network, Gradient Boosting

I. INTRODUCTION

Based on data from Riset Kesehatan Dasar Indonesia in 2018, it is believed that 1.5% of the Indonesian population or 2.784.064 individuals were affected with cardiovascular disease [1]. In addition, 85% of death caused by cardiovascular disease was due to heart attack or stroke which was one of the extreme cases of fatal cardiac arrhythmia. The cardiac arrhythmia itself is when heartbeat rhythm becomes irregular, either faster or slower than normal. However, the cardiac-arrhythmia can be prevented with early detection and medication of cardiac arrhythmia with the help of correct and accurate Electrocardiography (ECG) reading.

The ECG is a method to record heart activity represented by a graph of voltage and time. This method is readily available and can be used as a diagnostic tool to help diagnose heart arrhythmia. However, reading the ECG result and determining the arrhythmia conditions is sometimes quite challenging and time-consuming even for the qualified medical staff [2],[3]. Therefore, many automated methods have been proposed to shorten and simplify the process of ECG reading with the help of the Machine Learning (ML) algorithm, specifically the Deep Learning (DL) method.

The DL, i.e., a subfield of ML, has been a popular method to solve the ECG reading problems because of its ability to extract features from raw data without human supervision. Furthermore, the DL method can provide high scalability of data analyses driven by huge size of patient datasets, including various types of features [4]–[7]. Several previous works employ the DL method and its subtypes for creating classification models. One of those works is Murat et al. [8], concluding that the CNN was more favorable among other DL subtypes due to its low computational cost and fast training-time of the CNN model.

In addition, the usage of the CNN model can further be improved by augmenting and preprocessing the datasets, as shown by Acharya et al. [9]. They show that a skewed or imbalanced dataset could negatively affect the overall performance of a CNN model, and a well-balanced dataset could result in a less biased model with higher overall performance.

Meanwhile, Kachuee et al. [10] found that the CNN model can be used as a base to train other classifications with different datasets and able to yield a satisfactory result to classify the said datasets. Thus, this model opens up the possibility of using a pre-trained model or a part of it to construct a better model.

Other than the DL-CNN methods, some proposed methods use a conjunction of the ML algorithms. For example, one of the researches conducted by Batra and Jawa [11] uses a combination of Support Vector Machine (SVMs) as a classifier and Gradient Boosting as a feature selection. The experiment shows that their maximum experimental accuracy can achieve up to 84.82%, which is rather low than the DL-CNN method. However, it is possible to improve a model when the task to classify, and feature selection is separated into two different models.

This possibility has been proven by Hong et al. [12] in their research paper when they combine a Deep Neural Network (DNN) and gradient boosting algorithm, specifically the XGBoost. Their experiment shows that they can reach an accuracy of 91.17% and conclude that this method was more flexible and effective.

The XGBoost uses a boosted decision tree (BRT) that has a better speed and performance than other gradient boosting methods. Furthermore, the BRT improves the performances of a single model by fitting many models and combining them into one for a prediction. That is to say that the BRT is a combination of two algorithms which is a regression tree (decision tree) for classification and a regression tree for a group of models to boost and combine many models [13], [14].

According to the previous works, this paper uses a single lead ECG signal from the MIT-BIH Arrhythmia database as an input and a DL CNN method to extract features from the single-lead ECG reading and also use a gradient boosting algorithm to predict the final outcome of single-lead ECG reading with the input of intermediate output of the CNN model. Therefore, this model can increase accuracy and reduce the number of parameters used in the CNN model, resulting in higher performance of a model concerning the traditional CNN model.

II. RESEARCH METHODS

A. Dataset

The dataset used in this paper comes from a processed PhysioNet MIT-BIH Arrhythmia Databases. The raw dataset consist of ECG Recording at the sampling rate of 360 Hz from 47 different subjects, with each beat annotated by at least two cardiologists [15].

This raw dataset then goes through signal preprocessing by Kachuee et al. [10] by extracting, cropping, and zero paddings each individual beat to a certain length, making them a single labeled lead ECG beats dataset.

This labeled ECG beats dataset comprises of 109.446 samples of identical lengths beats that further differentiated into 5 classes as seen in Table 1, with the majority of samples going to class 0 with 82.8% samples and the lowest goes to label 3 with 0.7% samples. We can conclude from this that the data in the dataset is clearly bias to label 0 and highly imbalanced.

Table 1. Summary of Dataset

Classes	Heartbeat type	Number of Sample
0	Non-ectopic (N)	72.471
1	Supra ventricular ectopic (S)	2.223
2	Ventricular ectopic (V)	5.788
3	Fusion (F)	641
4	Unknown (Q)	6.431
	Total	109.446

B. Proposed Method

a) Preprocessing

Before training our model, there should be several steps of the dataset preprocessing. This step, as described below, is done to prepare and process the data so it can be used to train the model with higher accuracy and lower biases.

First, the data from dataset is split into 2 types: test data and train data. As for the proportion of the data, 80% of the total data becomes training data, and the rest becomes test data. This splitting ensures that the same dataset can be used to train and test or validate the trained model.

After that, the training data that has been split will now be resampled. The data resampling is done because the dataset was highly unbalanced and biased to class 0 heartbeat. This can cause the trained model to also be more biased to class 0 heartbeat. To solve this, we will use oversampling and undersampling. These methods are to resample the train data and eventually achieve balanced train data. In addition, the following methods can also assure that no bias occurring for the trained model.

Oversampling is a method to add new samples into minority classes to balance the dataset, and this can be done by replicating existing minority samples to a certain amount. This method will be applied to class 1 and class 3 beat.

The undersampling method is the opposite, which reduces the number of samples in the majority class by choosing a certain number of samples from the whole [16]. In this paper, undersampling will be used to class 0 and 4.

For both oversampling and undersampling, the number of samples chosen is 5000 data. This is because 5000 was considered a middle ground to not replicate too much data from the class 3 beat. However, at the same time, also maintain a big enough train data to ensure the best training. This step will result in the total train data being reduced to 25.788 data.

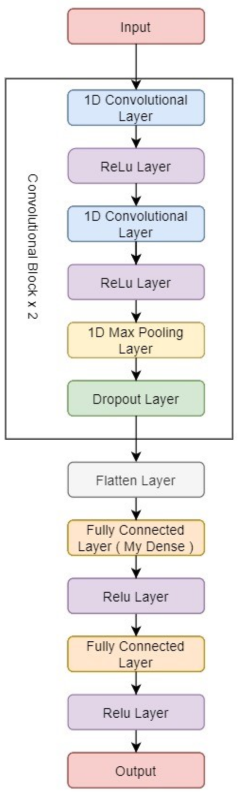

Fig. 1. Architecture of the proposed CNN Model

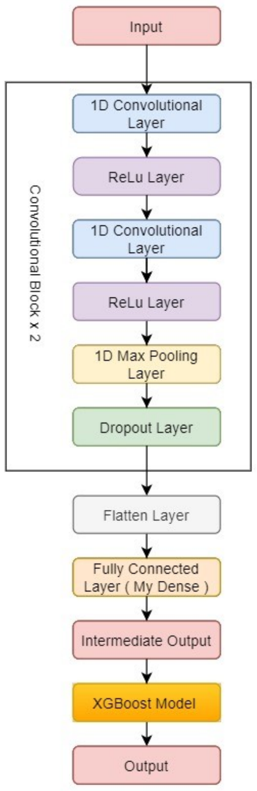

Fig. 2. Architecture of the proposed CNN XGBoost Model

b) CNN Model Architecture

After preprocessing the dataset, we can build the architecture of our CNN model. There will be two main blocks in our CNN architecture, which are the convolutional block and the fully connected block.

Figure 1 shows the proposed CNN model after training the dataset with several architectures and parameters. The convolutional block comprises 2 1D Convolutional Layers, which include the convolutional layer and its activation function, which is ReLu layer, followed by a 1D Max Pooling layer, and finished with the dropout layer.

For the fully connected block, the output of the convolutional block will first be flattened then it will be passed to the dense layer. The first dense layer will be called My Dense. As for the second dense layer it will eventually result from the output of classification. The parameters used in CNN model training can be seen in Table 2.

Table 2. Parameters of CNN Training

Parameter	Values
Optimizer	Adam Optimizer (Learning Rate = 0.001)
Loss	Sparse Categorical Cross Entropy
Metrics	Accuracy
Epoch	20

c) CNN-XGBoost Model Architecture

In training the XGBoost model, the output of the My Dense layer will be fed into the XGBoost model. However, unlike the proposed CNN method as shown in Figure 1, the process is done by cutting the CNN model training process up to My dense layer, in which this kind of CNN model is called an intermediate model. Then, the output of the intermediate model, i.e., intermediate output, is fed into the XGBoost model, as shown in Figure 2. Eventually, the parameters used in the proposed XGBoost model are listed in Table 3.

Table 3. Parameters of XGBoost Training

Parameter	Values
Booster	Gbtree
Objective	Multi: Softprob
Learning Rate	0.3
n_estimators	100

d) Performance Evaluation

After the model has been successfully developed, there should be an evaluation to conclude whether the developed model has reached a satisfactory result or need to be developed further. This step can be done by predicting the test data that has been split in the previous preprocessing. The parameters that can help evaluate the model are Accuracy, Recall, Precision, and F1-Score. Those

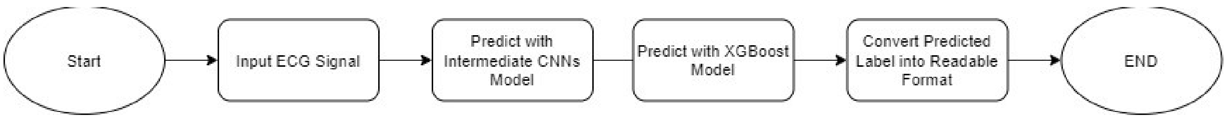
Start
Input ECG Signal
Predict with Intermediate CNNs Model
Predict with XGBoost Model
Convert Predicted Label into Readable Format
END

Fig. 3. System Process Diagram

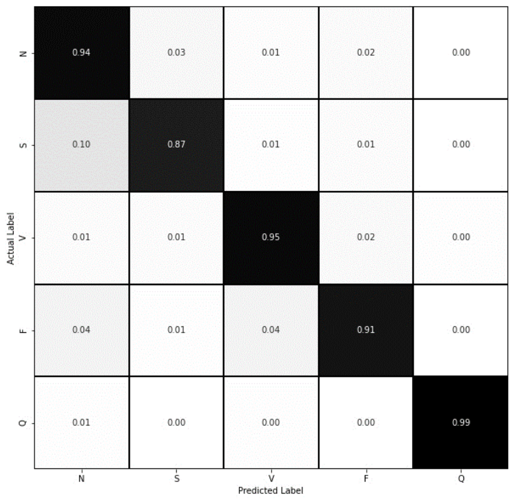

Fig. 4. CNN Model Confusion Matrix

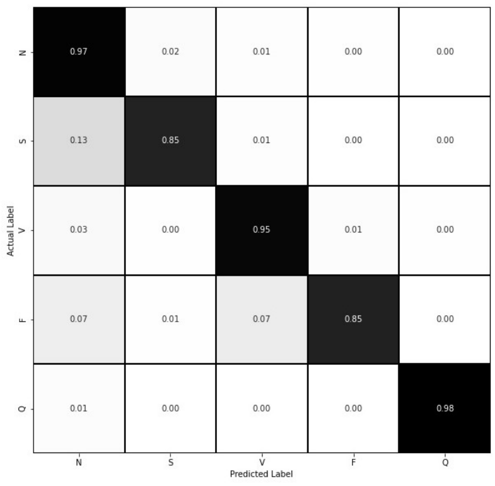

Fig. 5. CNN XGBoost Model Confusion Matrix

parameters can be calculated by using the following equations:

$$Accuracy = \frac{TP + TN}{TP + FP + FN + TN} \quad (1)$$

$$Precision = \frac{TP}{TP + FP} \quad (2)$$

$$Recall = \frac{TP}{TP + FN} \quad (3)$$

$$F_1\ Score = \frac{2 \times Precision \times Recall}{Precision + Recall} \quad (4)$$

III. RESULTS

The following data have been acquired from several iterations with different convolutional blocks, learning rates, and integration of the XGBoost model.

Table 4 shows a comparison of accuracy from different models with different numbers of convolutional blocks. It can be concluded that 2 convolutional blocks with the addition of the XGBoost model are the most efficient number and should be used as the main classification model.

We can further see the impact of XGBoost integration from the classification report of the main model in Table 5 for the pure CNN model and Table 6 for the CNN XGBoost model, specifically for class 3 which increase considerably in the CNN XGBoost model from pure CNN model.

Table 4. Comparison of Accuracy

	Accuracy	
	CNN Model (Fig.1)	**CNN XGBoost Model (Fig.2)**
2 Conv. Blocks	94.33 %	96.18 %
3 Conv. Blocks	92.09 %	95.50 %
4 Conv. Blocks	90.11 %	95.86 %
5 Conv. Blocks	81.75 %	93.37 %

Table 5. Classification Report of CNN Model (Fig.1)

Classes	**Precision**	**Recall**	**F1-Score**
0	0.99	0.94	0.97
1	0.49	0.87	0.63
2	0.86	0.95	0.90
3	0.30	0.91	0.45
4	0.95	0.99	0.97

Table 6. Classification Report of CNN XGBoost Model (Fig.2)

Classes	**Precision**	**Recall**	**F1-Score**
0	0.99	0.97	0.98
1	0.58	0.85	0.69
2	0.88	0.95	0.91
3	0.57	0.85	0.68
4	0.96	0.98	0.97

In Figure 3, we can see the prediction process starting from an input of raw ECG signal. Next, we can do signal preprocessing to make the ECG signal suitable for the input of the intermediate CNN model. Then, the output of the intermediate CNN model will be treated as input for the XGBoost model that will predict the final label. Finally, we can further convert the label into readable a format.

Figure 4 and Figure 5 are the confusion matrix of the Pure CNN model and CNN XGBoost model, respectively. From these figures, we see the difference in prediction between the CNN and the CNN XGBoost models and measure the precision, recall, and f1-score.

We can also see the main CNN model training evaluation in Figure 6 for the accuracy and Figure 7 for the loss. Those figures show that the increase of accuracy and loss in training is stable and we can conclude that 20 epoch is suitable for training the model. Furthermore, we can also see the different, learning rate made for this model in Figure 8, where the learning rate of 0,001 perform better than the default learning rate of the Adam optimizer which is 0,01, as such we can conclude the learning rate of 0,001 is better for this model.

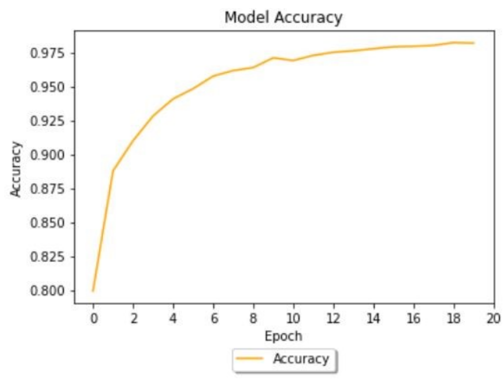

Fig. 6. Main CNN Model Accuracy Evaluation

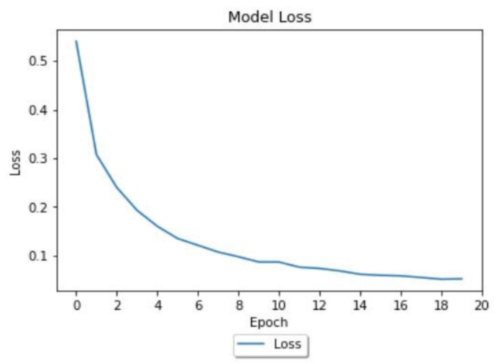

Fig. 7. Main CNN Model Loss Evaluation

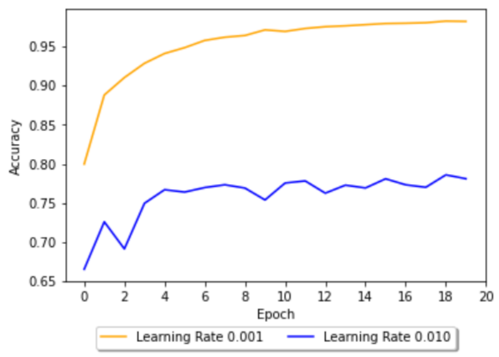

Fig. 8. Main CNN Model Accuracy Evaluation for different Adam Learning Rate

IV. DISCUSSION

Based on the result that has been shown previously. Integration of gradient boosting algorithm, specifically the XGBoost, is shown to boost the overall performance of a pure CNN model. This can be seen in Table 4. A few iterations of the pure CNN model with different number of convolutional blocks shown to has an increase in accuracy with the most significant increase for the 5 convolutional block which increase from 81.75 % to 93.37 %.

We can also see the overall increase in performance in Table 6, which show that integration of XGBoost could also increase the overall classification score, including accuracy, precision, recall, and f1-score.

Furthermore, we could also refer to Figures 3 and 4, where the confusion matrix for both the pure CNN model and the XGBoost CNN model are presented, and notice that the number of correct predictions is increased. Conversely, the number of false predictions is decreased.

V. CONCLUSSION

Based on the research that has been done in this paper, we can conclude that the addition of the XGBoost model into the CNN model can improve the overall performance of a constructed model. With the most improvement achieved by a sub-optimal CNN model, this can help improve the bad CNN model or a CNN model that was designed to be lighter for the purpose of low-computational device hence lowering the number of parameters used in the model.

In the future, it would be interesting if this method could be developed and used to improve a model. It is designed to be used on a low-computational device to see the improvement that it can achieve and the effect of computational need that may raise by adding an XGBoost model.

ACKNOWLEDGMENT

Authors would like to thank the lecturer who help and guide author in the completion of this research paper and Telkom University which have supported the research and publication of this paper.

REFERENCES

[1] Badan Penelitian dan Pengembangan Kesehatan, "Laporan_Nasional_RKD2018_FINAL.pdf," *Badan Penelitian dan Pengembangan Kesehatan*. p. 198, 2018, [Online]. Available: http://labdata.litbang.kemkes.go.id/images/download/laporan/RKD/2018/Laporan_Nasional_RKD2018_FINAL.pdf.

[2] S. L. Oh, E. Y. K. Ng, R. S. Tan, and U. R. Acharya, "Automated diagnosis of arrhythmia using combination of CNN and LSTM techniques with variable length heart beats," *Comput. Biol. Med.*, vol. 102, pp. 278–287, 2018, doi: 10.1016/j.compbiomed.2018.06.002.

[3] Ö. Yıldırım, P. Pławiak, R. S. Tan, and U. R. Acharya, "Arrhythmia detection using deep convolutional neural network with long duration ECG signals," *Comput. Biol. Med.*, vol. 102, pp. 411–420, 2018, doi: 10.1016/j.compbiomed.2018.09.009.

[4] A. Esteva *et al.*, "A guide to deep learning in healthcare," *Nat. Med.*, vol. 25, no. 1, pp. 24–29, 2019, doi: 10.1038/s41591-018-0316-z.

[5] G. Sannino and G. De Pietro, "A deep learning approach for ECG-based heartbeat classification for arrhythmia detection," *Futur. Gener. Comput. Syst.*, vol. 86, pp. 446–455, 2018, doi: 10.1016/j.future.2018.03.057.

[6] A. Isin and S. Ozdalili, "Cardiac arrhythmia detection using deep learning," *Procedia Comput. Sci.*, vol. 120, pp. 268–275, 2017, doi: 10.1016/j.procs.2017.11.238.

[7] Z. Ebrahimi, M. Loni, M. Daneshtalab, and A. Gharehbaghi, "A review on deep learning methods for ECG arrhythmia classification," *Expert Syst. with Appl. X*, vol. 7, p. 100033, 2020, doi: 10.1016/j.eswax.2020.100033.

[8] F. Murat, O. Yildirim, M. Talo, U. B. Baloglu, Y. Demir, and U. R. Acharya, "Application of deep learning techniques for heartbeats detection using ECG signals-analysis and review," *Comput. Biol. Med.*, vol. 120, no. April, p. 103726, 2020, doi: 10.1016/j.compbiomed.2020.103726.

[9] U. R. Acharya *et al.*, "A deep convolutional neural network model to classify heartbeats," *Comput. Biol. Med.*, vol. 89, pp. 389–396, 2017, doi: 10.1016/j.compbiomed.2017.08.022.

[10] M. Kachuee, S. Fazeli, and M. Sarrafzadeh, "ECG heartbeat classification: A deep transferable representation," *Proc. - 2018 IEEE Int. Conf. Healthc. Informatics, ICHI 2018*, pp. 443–444, 2018, doi: 10.1109/ICHI.2018.00092.

[11] A. Batra and V. Jawa, "Classification of Arrhythmia Using Conjunction of Machine Learning Algorithms and ECG Diagnostic Criteria," *Int. J. Biol. Biomed.*, vol. 1, pp. 1–7, 2016.

[12] S. Hong *et al.*, "Combining deep neural networks and engineered features for cardiac arrhythmia detection from ECG recordings," *Physiol. Meas.*, vol. 40, no. 5, p. 054009, 2019, doi: 10.1016/j.snb.2007.07.003.

[13] B. R. Manju and A. R. Nair, "Classification of Cardiac Arrhythmia of 12 Lead ECG Using Combination of SMOTEENN, XGBoost and Machine Learning Algorithms," *Proc. 2019 Int. Symp. Embed. Comput. Syst. Des. ISED 2019*, pp. 48–55, 2019, doi: 10.1109/ISED48680.2019.9096244.

[14] J. Elith, J. R. Leathwick, and T. Hastie, "A working guide to boosted regression trees," *J. Anim. Ecol.*, vol. 77, no. 4, pp. 802–813, 2008, doi: 10.1111/j.1365-2656.2008.01390.x.

[15] G. B. Moody and R. G. Mark, "The impact of the MIT-BIH arrhythmia database," *IEEE Eng. Med. Biol. Mag.*, vol. 20, no. 3, pp. 45–50, 2001, doi: 10.1109/51.932724.

[16] M. S. Shelke, P. R. Deshmukh, and P. V. K. Shandilya, "A Review on Imbalanced Data Handling Using Undersampling and Oversampling Technique," *Int. J. Recent Trends Eng. Res.*, vol. 3, no. 4, pp. 444–449, 2017, doi: 10.23883/ijrter.2017.3168.0uwxm.