Robust and Task-Aware Training of Deep Residual Networks for Varying-Lead ECG Classification

Hansheng Ren*, Miao Xiong*, Bryan Hooi

^{1} National University of Singapore

Abstract

In PhysioNet/Computing in Cardiology Challenge 2021, we developed an ensemble model based on different epochs of ResNet to classify cardiac abnormalities from 12 6, 4, 3, 2 lead electrocardiogram (ECG) signals. Epochs are chosen based on validation performance on CPSC and Georgia datasets. In order to adapt to the specially designed challenge score, We designed a multi-task loss to combine the benefit of binary cross-entropy loss and challenge loss. Besides, we also integrated a subsample frequency feature into the model to learn from the signals. To gain a better generalization ability, mixup and weighted loss are introduced.

We submitted our model in the official phase, and our final model achieved a challenge validation score of 0.637 in 12 leads, 0.622 in 6 leads, 0.623 in 4 leads, 0.621 in 3 leads, and 0.625 in 2 leads. Our team achieves the best ranking in 2 leads, as 5th in the official ranking.

1. Introduction

Electrocardiography is a commonly used, non-invasive technique for recording electrical changes and examining the heart's physiological activities. The record, named electrocardiogram (ECG), shows the series of waves that relate to the electrical impulses that occur during each heartbeat. In the PhysioNet/Computing in Cardiology Challenge 2021[1], we are required to develop models for electrocardiograms of 12, 6, 4, 3, 2 leads to identify the cardiac abnormalities present in these recordings. In this paper, we will show our proposed methods and what we have tried for this challenge for inspiration.

2. Datasets

The Challenge data include over 88,000 annotated twelve-lead ECG recordings from eight sources, including CPSC, CPSC2, PTB, PTB-XL, Georgia, INCART, ChapmanShaoxing, and Ningbo, with sampling frequency ranging from 257Hz to 1000Hz, and data length ranging from less than 10 seconds to 30 minutes.

2.1. Data Processing

The challenge data was sourced from eight organizations, and recordings from different sources have distinct features. As an initial step, we want to process all samples into fixed length and fixed sample frequency. Based on the observation that most recordings are 500Hz, we decide to upsample or downsample all recordings into 500Hz. By analyzing the dataset statistics, we find that most samples are around ten seconds, and the INCART dataset with 30-minute-long signals only contains 74 recordings and does not appear in the test dataset. We decide to sample all recordings to samples with lengths equal to 4096 instead of splitting them into multiple patches. Zero padding is introduced for recordings shorter than 4096, and for recordings longer than 4096, we randomly select the start point to fetch a fixed-length patch. Besides the recording itself, we also utilize some auxiliary information, including age and sex.

In order to decide whether the model has been overfitted, a validation set is needed. However, different datasets have different distributions. In order to gain a better generalization on the test dataset, we have tried two modes of the validation set. One is to treat every information source equally and randomly split 80% of this dataset as a training dataset and 20% as the validation set. Another is to put more weight on CPSC and Georgia, splitting 20% of these two datasets as the validation set and leaving the remaining as training datasets.

3. Method

After obtaining the preprocessed signals, we train our model using multi-task learning to align better with clinical realities. We adopt a normal binary cross-entropy loss and a loss based on the challenge score metric that points out which misdiagnoses are more harmful, named chal-

*These authors contributed equally and are co-first authors.

lenge loss. For one input signal, we first feed them into our modified 1D-resnet18 or 1D-effientnet model to get its embedding and then use its embedding for two independent linear layers to generate two predictions. These two predictions are used to calculate binary cross-entropy loss and challenge loss separately.

3.1. Model Architecture

For Our model architecture, we tried two types of widely used deep learning architecture for generating our recording embeddings, 1D-Resnet18 and 1D-Efficientnet, with details as below.

1D-Resnet18 Except the design of multi-task loss, We also modify the Resnet18 [2] in a way following the last year challenge's winning solution[3]:one convolutional layer followed by N = 8 residual blocks (ResBs) with large kernel and dropout layer, each of which contains two convolutional layers and a squeeze and excitation (SE) block introduced by [4]. After obtaininng the embedding of signals, a fully connected layer is introduced to analyze auxiliary information, including patient age and gender, and join in recording embedding into two independent dense layers for different loss functions.

1D-Efficient net We also try to use the modified 1D EfficientNet [5] as the main model architecture during the official phase. This model is first published on [5] and its one-dimensional PyTorch implementation originates from this repo.

Challenge loss Since our classification task considers the clinical reality that making some misdiagnoses is more harmful than others and some misdiagnoses are partially acceptable, we design a special challenge loss to adapt our model. Besides, we design a special task-aware loss function to teach our model different severity of making misdiagnoses during training. Specially speaking, we define continuous relaxation of the scoring metric and optimizing the resulting loss function directly in our end-to-end framework so the model can align better with clinical realities as captured in the scoring metric. Specifically, denote the weight of score metric as w_{ij}, the model prediction as x, and the label as y. The challenge loss is calculated by:

$$norm = max(1, sum(max(x, y))) \tag{1}$$

$$z_{ij} = y_i x_j w_{ij} \tag{2}$$

$$ChallengeLoss = \frac{sum(z)}{norm} \tag{3}$$

This challenge loss is indeed the challenge scoring metric generalized to non-binary input cases. So after feeding the recordings to the model during training, we can get a generalized challenge score between model scalar outputs and labels without any threshold and then use it as a loss to supervise backward. We also implemented a batch-style computing function of this challenge loss to make its speed acceptable for training our model.

BCEloss We also adopt the binary cross-entropy loss since it's basically a multi-label classification problem.

Weight Loss for Imbalanced Class We also tried a modified BCE loss for the data imbalance problem. Figure 1 shows the positive labels of the ground truth for every class. It demonstrates that the data imbalance problem is significant in most classes. In order to solve this problem, we have tried two solutions: one is to use the weighted binary cross-entropy loss; another is to use an imbalanced data sampler. The former approach slightly increases the final performance while the latter does not work (we leave it in the discussion part). We define the weight manually, different from the standard way of using a positive/negative ratio to compute the weights. Similar to Figure 1, we calculate the validation set accuracy for every class and try to increase the weight for classes in which our proposed model performs poorly.

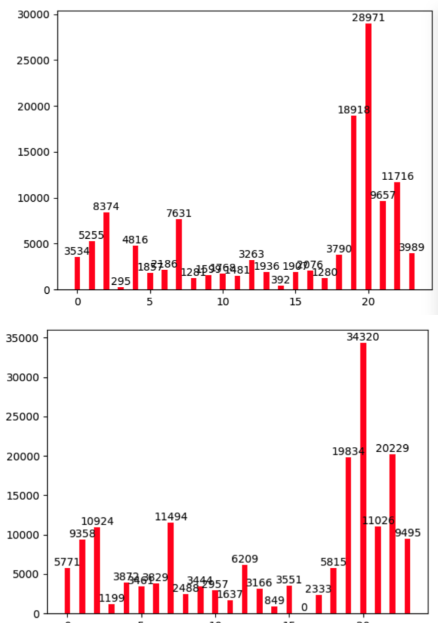

Figure 1. Comparison of positive samples in ground truth labels(**Left**) and prediction results(**Right**). The result is given by the main model (resnet18 with local threshold optimization and multi-loss).

Mixup In order to enhance the generalization ability, we add mixup [6] to the training process. Firstly, we sample two datapoints (x_i, y_i) and (x_j, y_j) from the training dataset. Then we contruct a new mixed datapoint $x_{ij} = \alpha x_i + (1 - \alpha)x_j$ with new label $y_{ij} = \alpha y_i + (1 - \alpha)y_j$. Model trained on these mixed datapoints tends to learn soft decision boundary between classes.

3.2. Implementation Setup

In this section, we introduce three training techniques we used for our model: loading parameters of 12-lead trained model to model with fewer leads, threshold optimization and model ensemble.

loading parameters of 12-lead trained model to model with fewer leads We observe that if we adopt modified ResNet model architecture when we train fewer leads model, we can reuse almost all the parameters from 12 leads except the first convolution layer since the following layers' parameter layout is the same. So we first train a 12 lead model. And when we are going to train a model with fewer than 12 leads, we initialize the model's parameters by the trained 12 lead model's parameter, except the first convolutional layer.

Threshold optimisation After getting the scalar outputs of our model, we choose the scalar output generated by challenge loss to convert it to a binary prediction by applying thresholds. Since we did 80/20 data split after data preprocessing, we can use the validation data to conduct a grid-thresholds search. Similar to [], We search a global threshold (the same threshold for all classes) first and then do a local threshold search for each class. It's worth mentioning that if we directly use the divided 20 percentage validation set to search the model threshold and make the model selection, it seems not optimal. So we do an 80/20 data split for the whole training data first, and then in the validation set we duplicate recordings from the Georgia dataset 21 times and recordings from the CPSC dataset by seven times to make a validation with a more reasonable data distribution for threshold search and model selection.

Model ensemble During the official phase, we submit several ensemble versions of training models to the challenge server. We have two types of ensemble: the first one is the majority vote of models, that is, each model produces a binary prediction based on their own saved threshold searched by our self-divided validation set for a test recording; another is that we calculate the mean of these model's corresponding searched thresholds and apply it to the model's scalar outputs.

4. Results

The results of our submitted models' 2 lead challenge scores are shown in table 3. We denote the default version as ResNet18 trained using BCEloss plus 0.1 challenge loss, without model ensemble and mixup technique. Then, sub version 1 denotes ResNet18 with only global threshold search; sub 2 is ResNet18 with local threshold search; sub3 is ResNet18 trained by mixup technique with 0.3 weight and global threshold search. Sub 4 is ResNet18 trained by weighted loss mention above and with local threshold search. Sub 5 is 1D EffientNet with local threshold search. Sub 6 is ResNet18 trained by mixup technique with 0.15 weight and local threshold search. sub7 is ResNet18 with local threshold search and parameters initialized 12-lead ResNet18 trained by 0.1 challenge loss and BCE loss. Sub 8 is the ResNet18 with no validation set, trained on all given data and use hand-set threshold 0.1, ensemble models from 20 epoch to 30 epoch with majority vote strategy. Sub9 is ResNet18 with local threshold search on validation set while ensemble models from 20 epoch to 30 epoch with mean threshold ensemble strategy mentioned above.

Table 1. submitted models challenge scores on 2-lead

Methods	challenge score on 2 leads
sub1	0.612
sub2	0.621
sub3	0.598
sub4	0.602
sub5	0.578
sub6	0.623
sub7	0.560
sub8	0.162
sub9	0.625

We also have some versions tested on all kinds of lead sets, shown in table 2. And in this table version 1 means Resnet18 trained by 0.1 challenge loss and BCE loss, and load parameters from 12-lead resnet18 to fewer leads of model, version 2 means Resnet18 trained by 0.1 challenge loss and BCE loss and using mixup training technique with weight 0.15. version 2 means Resnet18 trained by 0.1 challenge loss and BCE loss, ensemble models from 20 epoch to 30 epochs using mean thresholds.

Table 2. submitted models challenge scores on 12/6/4/3/2 leads

Methods	12 leads	6 leads
4 leads	3 leads	2 leads
version 1	0.631	0.603
0.597	0.574	0.560
version 2	0.620	0.615
0.615	0.623	0.620
version 3	0.637	0.622
0.623	0.621	0.625
version 3	0.638	0.599
0.622	0.623	0.614

5. Discussions

During the exploration of finding a suitable classifier for this challenge, we have tried many different approaches. Some do increase the performance, while some are not.

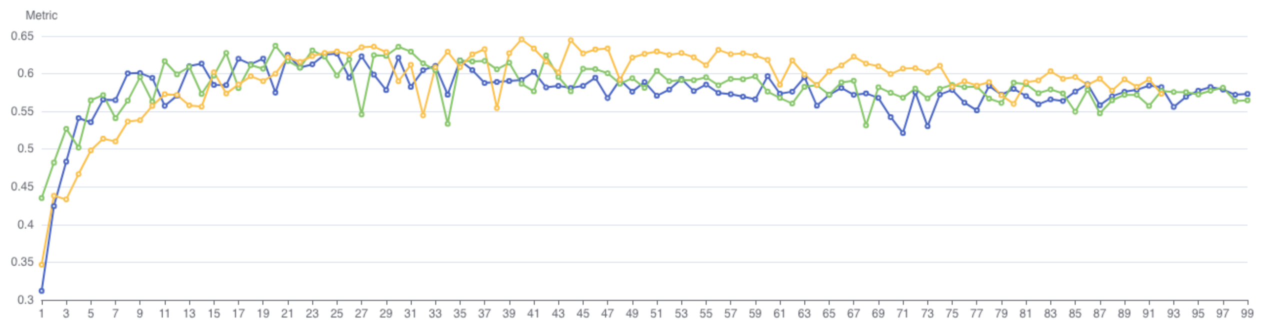

Figure 2. Validation accuracy during every training epoch. The blue plot represents the model trained on resnet18, the green plot denotes the model trained no resnet34, while the yellow one represents the model trained on resnet152. It can be seen that after 20-30 epochs the validation set performance has been decreased, indicating the occurance of overfitting.

Even though some methods do not work, we believe it might shed light on improving the performance.

Rule-based Method We do this exploration by starting from the 'Bradycardia' disease, whose definition is simple enough for a non-expert to identify. Figure 1 shows our default model's performance on every class, and it indicates that our model is not performing well at identifying Bradycardia. Based on Wiki, Bradycardia is a condition typically defined wherein an individual has a resting heart rate of under 60 beats per minute (BPM) in adults, although some studies use a heart rate of less than 50 BPM. [7] Inspired by this, we compute every recording's bpm and assign positive labels to every sample with BPM lower than 60. Unfortunately, we find that many recordings with bpm less than 60 and 50 are not labeled as Bradycardia, making this algorithm fail to improve performance.

6. Conclusions

During this challenge, we tried two types of widely used deep learning architecture for generating feature embeddings of every recordings: 1D-Resnet18 and 1D-Efficientnet. Besides, we utilize the multi task weighted loss to adapt to the specially designed challenge score, and mixup to increase the generalization ability. To better classify, we also integrate model ensemble, threshold optimisation and pre-train technique. All these tricks contribute to our final performance and make us as the 5th in 2 leads in the official ranking.

Acknowledgments

Give any acknowledgments here.

References

[1] Reyna MA, Sadr N, Alday EAP, Gu A, Shah AJ, Robichaux C, Rad AB, Elola A, Seyedi S, Ansari S, et al. Will two do? varying dimensions in electrocardiography: The physionet/computing in cardiology challenge 2021. Computing in Cardiology 2021;48:1–4.

[2] He K, Zhang X, Ren S, Sun J. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In Proceedings of the IEEE international conference on computer vision. 2015; 1026–1034.

[3] Zhao Z, Fang H, Relton SD, Yan R, Liu Y, Li Z, Qin J, Wong DC. Adaptive lead weighted resnet trained with different duration signals for classifying 12-lead ecgs. In 2020 Computing in Cardiology. IEEE, 2020; 1–4.

[4] Hu J, Shen L, Sun G. Squeeze-and-excitation networks. In Proceedings of the IEEE conference on computer vision and pattern recognition. 2018; 7132–7141.

[5] Tan M, Le Q. Efficientnet: Rethinking model scaling for convolutional neural networks. In International Conference on Machine Learning. PMLR, 2019; 6105–6114.

[6] Zhang H, Cisse M, Dauphin YN, Lopez-Paz D. mixup: Beyond empirical risk minimization. arXiv preprint arXiv171009412 2017;.

[7] Bradycardia. Bradycardia, 2010. URL `https://en.wikipedia.org/wiki/Bradycardia`. [Online; accessed 31-August-2021].

Address for correspondence:

Hansheng Ren
3 Research Link, Singapore 117602
hanshengren@u.nus.edu

Miao Xiong
3 Research Link, Singapore 117602
miao.xiong@u.nus.edu