ECG Abnormalities Recognition Using Convolutional Network with Global Skip Connections and Custom Loss Function

Tomas Vicar[1], Jakub Hejc[1,2], Petra Novotna[1], Marina Ronzhina[1], Oto Janousek [1]

[1] Department of Biomedical Engineering, Faculty of Electrical Engineering and Communications, Brno University of Technology, Brno, Czech Republic
[2] Department of Pediatric, Children's Hospital, University Hospital Brno, Brno, Czech Republic

Abstract

The latest trends in clinical care and telemedicine suggest a demand for a reliable automated electrocardiogram (ECG) signal classification methods. In this paper, we present customized deep learning model for ECG classification as a part of the Physionet/CinC Challenge 2020. The method is based on modified ResNet type convolutional neural network and is capable to automatically recognize 24 cardiac abnormalities using 12-lead ECG. We have adopted several preprocessing and learning techniques including custom tailored loss function, dedicated classification layer and Bayesian threshold optimization which have major positive impact on the model performance. At the official phase of the Challenge, our team (BUTTeam) reached the 1st place with the Challenge score of 0.696 and 0.687 on the hidden data subset and 10 % hold-out of training set, respectively.

1. Introduction

A large number of automated ECG classification methods have been reported over last decade, most of which have only been evaluated on small or homogeneous datasets. In this paper, we present deep learning model for ECG classification as a part of the Physionet/CinC Challenge 2020 [1, 2]. The Challenge data consists of 12-lead ECGs from wide range of sources and recording platforms including signals of low quality or highly variable length. As another challenging task, the data contains various numbers of reported non-exclusive abnormalities obtained with inhomogeneous annotation methods.

The aim of the model is to classify these signals in a multi-label manner into one or more of 24 given classes. Major improvements of the method were achieved by: (1) customized convolutional neural network architecture with local and global skip connections, (2) data augmentation, (3) custom loss function based on the challenge metric, and (4) class specific threshold optimization.

2. Material and Methods

2.1. Data

Training dataset is composed of 43,101 labeled recordings from 6 different sources [1, 2]. Recordings are sampled with various sampling frequencies (257, 500 or 1000 Hz) and resolution settings. The dataset includes following 24 classes (abbreviation: number of cases): 1st degree AV block (IAVB: 2,394), atrial fibrillation (AF: 3,475), atrial flutter (AFL: 314), bradycardia (Brady: 288), right bundle branch block (CRBBB: 3,085), incomplete right bundle branch block (IRBBB: 1,611), left anterior fascicular block (LAnFB: 1,806), left axis deviation (LAD: 6,086), left bundle branch block(LBBB: 1,041), low QRS voltages (LQRSV: 556), nonspecific intraventricular conduction disorder (NSIVCB: 997), pacing rhythm (PR: 299), premature atrial contraction (PAC: 1,944), premature ventricular contraction (PVC: 553), prolonged PR interval (LPR: 340), prolonged QT interval (LQT: 1,391), abnormal Q-wave (QAb: 1,013), right axis deviation (RAD: 427), sinus arrhythmia (SA: 1,240), sinus bradycardia (SB: 2,359), sinus rhythm (NSR: 20,846), sinus tachycardia (STach: 2,402), abnormal T-wave (TAb: 4,673) and T-wave inversion (TInv: 1,112).

2.2. Data preprocessing

Preprocessing pipeline consisted of three basic steps common for both training phase and inference: (1) time-domain resampling with the fixed sampling frequency 125 Hz (linear interpolation combined with decimation and anti-aliasing FIR filter, if needed), (2) correction of baseline fluctuations performed by subtracting out a 2-second wide moving average and (3) normalization of mean and standard deviation (STD) based on dataset statistics.

Additionally, augmentation tasks such as $\pm 10\%$ stretch along temporal axis (p=0.8) or $\pm 20\%$ amplification along voltage axis (p=0.8) were randomly applied during training to prevent the model from overfitting.

Long-term Holter recordings are due to possible memory issues only allowed to pass through the model during the inference. Then, divided into smaller segments, they are fed into the model as a variable-sized batch. Non-redundant binary encoded labels from the entire batch are then joined together into a single vector.

2.3. Model Architecture

In our work, we have adopted a Convolutional Neural Network (CNN) architecture based on a residual neural network (ResNet) [3], which is simple yet proven design for image multi-label classification tasks. Original 2D convolutional filters were replaced by its 1D equivalents. In order to address vanishing gradient problem efficiently, skip connections across residual layers were extended by global skip connection [4] providing a direct shortcut to the model input layer (Input Gate in Figure 1). ResNet based feature extractor consists of 6 residual blocks each of which contains 3 convolutional layers ($k = 3$). Since the data can be assigned into c non-exclusive classes, final classification layer is composed of $c = 24$ independent fully connected binary classifiers. This structure allows arbitrary combination of class labels [5] and, based on its principle, can be referred to as a Binary Units Training Technique (BUTT). Implementation details of proposed model architecture are depicted in Figure 1.

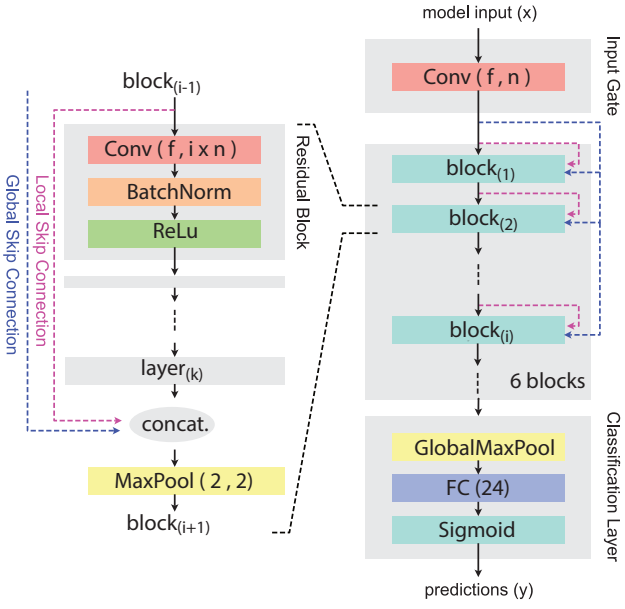

Figure 1: Architecture of proposed model. i – block number of ResNet; k – number of layer in residual block; n – number of filters in first layer

In order to use a mini-batch optimization, collected data samples were padded by zeros to equal length. This procedure may adversely affect extracted features if global or adaptive pooling are applied over temporal axis. To address this problem the zero-padded equivalent is cut out just after leaving ResNet layers. Global Max Pooling is then performed over temporal dimension and pooled tensor is fed into BUTT classifier.

2.4. Loss Function

Cross-entropy is commonly used as a loss function for classification tasks [6]. In the case of class imbalanced data, weighted variant of cross-entropy (WCE) or generalised dice loss function [7] can be introduced instead to provide better stability of a model during an onset phase of training. In multi-label classification problems WCE can be defined as follows:

$$WCE = -\sum_{c} w_c^+ t_c \log(p_c) + w_c^- (1 - t_c) \log(1 - p_c), \tag{1}$$

where c are individual classes, $t_c \in \{0, 1\}$ are binary encoded labels, $p_c \in [0, 1]$ are model output scores, and w_c^+ and w_c^- are weights for positive and negative classes, respectively.

Weights are inversely proportional to its frequency in the training dataset, specifically, $w_c^+ = N/N_c^+$ and $w_c^- = N/N_c^- = N/(N - N_c^+)$. N is the total number of samples, N_c^+ and N_c^- are the numbers of positive and negative samples within given class.

2.5. Custom Loss Function

The official Challenge Metric (CM) is based on modified confusion matrix $\mathbf{A}$, where each entry a_{ij} (confusion) is weighted according to given clinical importance w_{ij}:

$$CM = \sum_{ij} a_{ij} w_{ij}. \tag{2}$$

On this basis, we have derived a custom loss function, which includes the same clinical importance measures. In a multi-label case, modified confusion matrix $\mathbf{A}$ can be expressed by a matrix multiplication:

$$\mathbf{A} = \mathbf{L}^{\mathbf{T}} (\mathbf{R} \oslash \mathbf{N}), \tag{3}$$

where operator $\oslash$ represents point-wise division, $\mathbf{L}$ and $\mathbf{R}$ are $N \times c$ binary matrices, formed by one-hot encoded ground truth labels and thresholded model output scores, respectively. $\mathbf{N}$ is $N \times c$ normalizing matrix factoring in number of unique labels for each sample from $\mathbf{L}$ and $\mathbf{R}$ according to:

$$\mathbf{N} = ((\mathbf{L}|\mathbf{R})\mathbf{1}_{\mathbf{c}\times\mathbf{1}})\mathbf{1}_{\mathbf{1}\times\mathbf{c}}, \tag{4}$$

where $\mathbf{1_{c\times 1}}$ is $c \times 1$ all-ones matrix and symbol | stands for bit-wise binary OR operator. Operator OR as well as binary matrix R can now be replaced with its continuous equivalent using raw output scores p_c as follows:

$$\mathbf{N} = ((\mathbf{L} + \mathbf{R} - \mathbf{L} \odot \mathbf{R}))\mathbf{1_{c\times 1}})\mathbf{1_{1\times c}}, \quad (5)$$

where operator $\odot$ represents point-wise multiplication. This makes the metric differentiable, thus it can be used as a loss function.

2.6. Class Specific Threshold Optimization

Mapping a raw score of class membership onto a class label with the default threshold value of 0.5 does not always guarantee the best model performance with respect to a given metric. Thus, proposed model transforms raw output scores using a set of class-specific thresholds τ_c. Each τ_c was estimated via Python implementation [8] of Bayesian Optimization [9] with respect to CM loss function. Optimization was performed on a validation part of the dataset.

2.7. Model Training

Weights and biases of convolutional layers were initialized with Xavier [10] and constant (c=0) initialization, respectively. Model training was performed by Adam optimizer ($\beta_1 = 0.9$; $\beta_2 = 0.999$) [11] with decoupled weight decay regularization ($\lambda = 10^{-5}$) [12], modified learning rate schedule ($\alpha_0 = 10^{-3}$) and mini-batch size of 32. Learning rate schedule consisted of two cycles with decaying learning rate strategy $0.1\alpha_0$ every 30 epochs in each cycle. During the first cycle, the model was trained with the WCE, while in the second cycle custom CM Loss was used to retrain the model.

To bridge the generalization gap, we have adopted a Stochastic Weight Averaging algorithm [13] which captures model weights w_m in the end of every epoch and then sets new model with weights w_{SWA} as a running average of w_m from the last m captured models.

2.8. Ensemble Modeling

Combination of multiple models can decrease variance and may produce a more generalized output [14]. Final class label is thus given by a majority vote of 3 bootstrap aggregated ensembles, each fitted to a 90 % subset of the training data. The number of ensembles has been chosen to meet the computational limits of the Challenge.

3. Results and Discussion

Model performance was internally validated with a hold-out method using 10 % randomly sampled subset of the training data. Final evaluation has been performed on a subset of the Challenge hidden test data. A comparison of a base model with various hyper-parameter setting using the Challenge Metric (CM) and the Dice Coefficient (DC) [15] is listed in Table 1. Results of the best performing model on hold-out and hidden subset are listed in Table 2.

Table 1: The results of models with various hyper-parameters and without (w/o) applied customization on the hold-out (validation) subset (n - number of filters in first layer ($i \times n$ for $i-th$ layer) ; k – number of convolutional layers in block; f - filter size).

Model	CM	DC
Small model ($n = 12$; $k = 3$, $f = 7$)	0.665	0.542
Deep model ($n = 12$, $k = 12$, $f = 7$)	0.667	0.544
Wide model ($n = 48$, $k = 3$, $f = 7$)	**0.687**	**0.571**
Smaller filters ($n = 48$, $k = 3$, $f = 3$)	0.679	0.570
Larger filters ($n = 48$, $k = 3$, $f = 11$)	0.677	0.558
w/o augmentation (Wide model)	0.646	0.559
w/o CM Loss (Wide model)	0.636	0.552
w/o global skip con. (Wide model)	0.679	0.563

Table 2: The results of the best performing model on the hold-out validation subset and the hidden subset.

Model	Validation CM	Hidden Subset CM
Wide model	**0.687**	**0.696**

The best results were reached with the "Wide" model in which the number of convolutional filters (kernel size = 7) were preferred to a model depth. Model widening was performed by simply increasing the number of filters within each convolutional layer. This has led to the largest improvement of the DC by 0.03. Any further increment in the model depth by adding more convolution layers did not improve the model performance as well as either reduction or expansion of filter size.

Another improvement has been achieved by using the CM Loss function. Efficiency of the CM Loss is greatly dependent on mini-batch size and has been proven to cause an unstable learning (from scratch) of the model. However, when used as a secondary loss function during cycling learning rate schedule, it has led to an increase of DC by 0.02.

Data augmentation strategies mentioned previously showed an increment in DC by 0.01. On the other hand, some of preprocessing methods (e.g. baseline drift removal, temporal shifting, noise addition or simulation of electrode switch) have had no effect neither on the model performance nor on generalization capability most likely due to a sufficient diversity of the dataset.

Classification results for individual classes are shown

in Figure 2. The best performance (DC $>$ 0.8) has been reached for AF, SNR and PR. DC $>$ 0.6 was obtained in more than 45 % of classes. It should be noted that the model performance is still substandard in some of well distinguishable pathologies such as PVC, PAC, Brady, etc. This may be primarily caused by low occurrence of those classes in the dataset, and also by a presence of annotation inconsistencies (typically confusion of PVC for PAC).

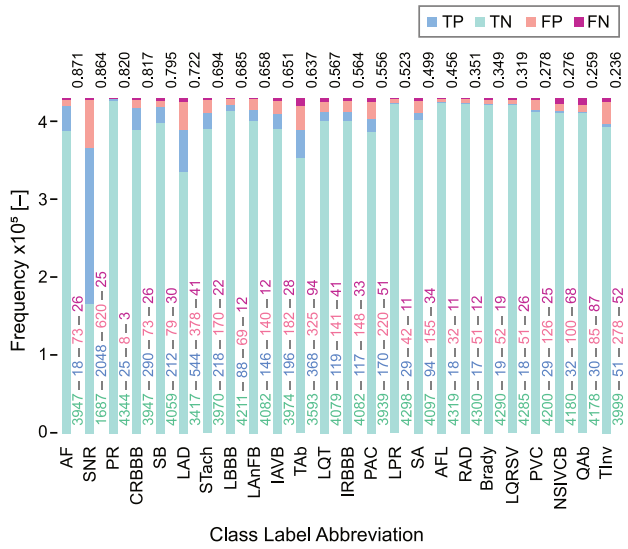

Figure 2: The results for individual classes. Total counts of true positive (TP), true negative (TN), false positive (FP) and false negative (FN) cases are given on the right side of each bar. Individual DC is given at the top of the related bar. Abbreviation for pathologies are listed in Section 2.1 Data.

4. Conclusions

Our ResNet based CNN architecture with BUTT as dedicated classification layer and custom CM Loss function has met the main aim of the Challenge and is capable to automatically recognize 24 of given pathologies. At the official phase the Challenge, our team (BUTTeam) reached the 1st place with the Challenge score of 0.696 and 0.687 on the hidden data subset and 10 % hold-out of training set (validation set), respectively.

The code is available at: `https://github.com/tomasvicar/BUTTeam`.

References

[1] Goldberger AL, Amaral LA, Glass L, Hausdorff JM, Ivanov PC, Mark RG, Mietus JE, Moody GB, Peng CK, Stanley HE. Physiobank, physiotoolkit, and physionet: components of a new research resource for complex physiologic signals. circulation 2000;101(23):e215–e220.

[2] Alday EAP, Gu A, Shah A, Liu C, Sharma A, Seyedi S, Rad AB, Reyna M, Clifford G. Classification of 12-lead ecgs: the physionet/computing in cardiology challenge 2020. physionet. Physiological Measurement 2020;(Under Review).

[3] He K, Zhang X, Ren S, Sun J. Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition. 2016; 770–778.

[4] Vicar T, Novotna P, Hejc J, Ronzhina M, Smisek R. Sepsis detection in sparse clinical data using long short-term memory network with dice loss. In 2019 Computing in Cardiology (CinC). IEEE, 2019; Page–1.

[5] Zhang ML, Zhou ZH. A review on multi-label learning algorithms. IEEE transactions on knowledge and data engineering 2013;26(8):1819–1837.

[6] Goodfellow I, Bengio Y, Courville A, Bengio Y. Deep learning, volume 1. MIT press Cambridge, 2016.

[7] Sudre CH, Li W, Vercauteren T, Ourselin S, Cardoso MJ. Generalised dice overlap as a deep learning loss function for highly unbalanced segmentations. In Deep learning in medical image analysis and multimodal learning for clinical decision support. Springer, 2017; 240–248.

[8] Nogueira F. Bayesian Optimization: Open source constrained global optimization tool for Python, 2014–. URL `https://github.com/fmfn/BayesianOptimization`.

[9] Snoek J, Larochelle H, Adams RP. Practical bayesian optimization of machine learning algorithms. In Advances in neural information processing systems. 2012; 2951–2959.

[10] Glorot X, Bengio Y. Understanding the difficulty of training deep feedforward neural networks. In Proceedings of the thirteenth international conference on artificial intelligence and statistics. 2010; 249–256.

[11] Kingma DP, Ba J. Adam: A method for stochastic optimization. arXiv preprint arXiv14126980 2014;.

[12] Loshchilov I, Hutter F. Decoupled weight decay regularization. arXiv preprint arXiv171105101 2017;.

[13] Izmailov P, Podoprikhin D, Garipov T, Vetrov D, Wilson AG. Averaging weights leads to wider optima and better generalization. arXiv preprint arXiv180305407 2018;.

[14] Hansen LK, Salamon P. Neural network ensembles. IEEE transactions on pattern analysis and machine intelligence 1990;12(10):993–1001.

[15] Powers DMW. Evaluation: From precision, recall and f-measure to roc., informedness, markedness & correlation. Journal of Machine Learning Technologies 2011;2(1):37–63.

Address for correspondence:

Tomas Vicar
Department of Biomedical Engineering,
Faculty of Electrical Engineering and Communication,
Brno University of Technology,
Technicka 12, 616 00 Brno, Czech Republic.
vicar@vutbr.cz