Cardiac Abnormality Detection in 12-lead ECGs With Deep Convolutional Neural Networks Using Data Augmentation

Lucas Weber[1], Maksym Gaiduk[1], Wilhelm Daniel Scherz[1], Ralf Seepold[1, 2]

[1] HTWG Konstanz, Konstanz, Germany
[2] I.M. Sechenov First Moscow State Medical University, Moscow, Russian Federation

Abstract

A residual neural network was adapted and applied to the Physionet/Computing data in Cardiology Challenge 2020 to detect 24 different classes of cardiac abnormalities from 12-lead. Additive Gaussian noise, signal shifting, and the classification of signal sections of different lengths were applied to prevent the network from overfitting and facilitating generalization. Due to the use of a global pooling layer after the feature extractor, the network is independent of the signal's length. On the hidden test set of the challenge, the model achieved a validation score of 0.656 and a full test score of 0.27, placing us 15th out of 41 officially ranked teams (Team name: UC_Lab_Kn). These results show the potential of deep neural networks for application to raw data and a complex multi-class multi-label classification problem, even if the training data is from diverse datasets and of differing lengths.

1. Introduction

Cardiovascular disease is the leading cause of death worldwide [1]. Due to the severe impact on the healthcare system, early and efficient detection of cardiovascular diseases is desirable [2]. The electrocardiogram (ECG) is a useful tool for detecting different cardiovascular diseases [3]. The Physionet/Computing in Cardiology Challenge 2020 [4] focused on automated, reproducible approaches for classifying cardiac abnormalities from 12-lead ECGs. Our best entry applied a convolutional neural network (CNN) to detect different abnormalities automatically. Our works main focus was on different methods to facilitate the network's generalization performance, mainly implemented using different generic data augmentation methods. Also, we adapted the network to be independent of different input sizes.

2. Methods

2.1. Data preprocessing

The overall challenge dataset contained overall 43101 signals of varying lengths from five seconds up to 30 minutes. The signals vary in the sample rate and the gain value, which maps the saved values to their physical value counterparts. Due to this variety, all signals have been resampled to most common frequency of 500 Hz using Fourier resampling. In the following, all given amounts of samples correspond to signals with a sampling rate of 500 Hz. In order to accomplish comparable signal amplitudes, all signals were divided by their associated gain value. After these initial steps, the signals were filtered in two consecutive steps to deal with outliers and unnecessary frequencies.

Outlier detection Due to unknown measurement perturbations, multiple occurrences of high valued outliers in the data range in duration from a few samples to a few hundred samples. Because of their high numerical values, these perturbations would significantly interfere with the applied convolutions' training and recognition process. We apply a uniform filter with a length of 50 samples (0.1 s) to each lead, effectively generating each signal's baseline. We replace every sample that differs more than 2.5 mV from this baseline with the signal mean. After this step, we also apply a median filter with a length of 5 samples (0.01 s) to smooth the signal.

Frequency filtering Most of the diagnostic information of the typical ECG is embedded in a frequency range from 0.5 Hz up to 50 Hz. We apply a second-order butter-worth bandpass filter with cut-off frequencies at 0.5 and 45 Hz to extract this frequency information.

2.2. Network structure

Deep residual neural networks [5] have proven themselves in various computer vision tasks. Due to their skip connections, even profound networks can be trained. In our algorithm, we also apply a residual network with a

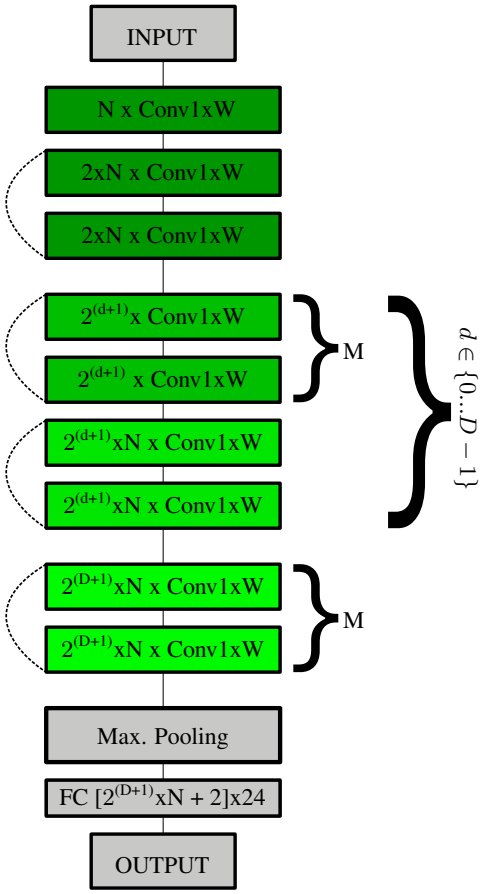

Figure 1. Sketch of the applied neural network.

few changes to its structure. As sketched in figure 1, the applied network has an initial starting block composed of three convolutions, the latter two having skip connections. After that, there are D repetitions of M normal blocks, consisting of two convolutions with stride one each, plus a bottleneck layer where the stride is three, and the number of features gets doubled every repetition to downsize the signal. After this repeatable structure, there follow M normal blocks. After this layer, there is global max-pooling for each feature, which enabling the network to be independent of different signal lengths and will be explained in detail in section 2.3. After this, a fully connected layer with one hidden layer acts as a classifier. All applied convolutions have the same width W, and the padding calculates as $W/2 - 0.5$. Dashed lines symbolize skip connections. Our changes to a residual neural network's standard structure are twofold and mostly concerned with the network being independent of signal length during training.

1. No intermediate pooling layers, and batch norm.
2. One global pooling layer.

All hidden layers utilize exponential linear unit (ELU) as an activation function; only the final output of the fully connected layer is processed with a sigmoid function as activation in order to generate probability like values in the range of zero to one. The network input is a 12-lead ECGs as an array with the dimension of $[12x\mathrm{S}]$, with S corresponding to the number of samples. The fully connected layer received the $2^{(D+1)} \times$ N features plus age and sex of the patient coded as integers. The features from the convolutional part of our network and the patient information are concatenated. If either one of these has no information in the header file, they are coded as -1. Our final submission had a depth D of three, a normal block repetition number M of five, and a kernel width W of five.

2.3. Input size independence

Due to the nature of the convolution, convolutional layers can be applied to inputs of different sizes. So the initialization and definition of those layers are independent of the given input size. Only the fully connected layers, which are widely used as classifiers at the end of neural networks, are dependent on the convolutional layers' output size, which are commonly seen as feature extractors. Following this definition as feature extractors, the final output of our convolutional part can be seen as $2^{(D+1)} \times$ N features and their matching probability at different positions in the signal. With a similar idea as in [6], we apply a global pooling layer, instead of the standard average pooling layer, that extracts the maximum probability of each feature matching somewhere in the signal, in essence, keeping the highest matching value for every feature. This approach makes the application of batch normalization very difficult. We forward each signal individually for training, accumulate the gradients, and divide the final loss by our batch size before triggering the backpropagation (similar to how most loss functions deal with batch size reduction).

2.4. Data augmentation methods

Data augmentation has been shown to improve neural network training and generalization performance significantly [7]. Data augmentation aims to multiply training data and reduce overfitting by adding different perturbations to the network input. We applied five different methods to improve model performance (see figure 2). Most methods need to be parameterized in a suitable way to prevent losing necessary information. All our methods can be applied during runtime with reasonable computational overhead. The augmentation methods can be listed as follows:

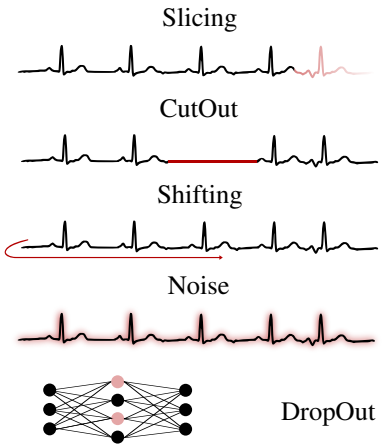

Figure 2. Visualization of applied data augmentation methods.

1. Signal slicing.
2. CutOut [7].
3. Channel shifting.
4. Additive noise.
5. Dropout [8].

Signal slicing Due to the network's ability to deal with different input sizes, we can feed the network with different parts of every signal, which also have different numbers of samples. So the network sees different parts of each signal every epoch. In our view, this prevents the network from overfitting, i. e. memorizing, certain parts of a signal. We create slices with a maximum length of 10000 samples (20 s) to prevent memory shortage in the GPU during training. The slicing samples are the same for all channels to keep the timing relations between the leads. The minimum signal length is either the signal's length if it is shorter than 4000 samples or exactly 4000 samples. During testing, we feed the network the whole signal. If the signal is longer than 10000 samples, we slice the signal in equal duration parts and predict each slice class. The overall score for this signal is then computed as the mean overall predictions.

CutOut [7] is a gold standard augmentation technique for many computer vision tasks. Despite the simplicity, it has improved performance on almost all applications. The central concept is to mask parts of the input data and counteract memorizing and overfitting. During training, we mask every lead independently with a probability of 35 %. The masking size is random and calculated independently for every channel with a length of zero samples to a maximum of 20 % of the signal length. The mask value is equivalent to the signal mean to introduce as little discontinuities as possible to the signal course.

Shifting Since the ECG is an approximately periodically signal, we apply random length signal shifting to the ECG. It is implemented as a circular motion, where parts that are shifted out at the end will be patched in at the beginning of the signal. The main parameter is the maximum shifting length. We applied two different versions of this method. In one version, all channels are shifted the same random amount of samples. For this version, we shift for a maximum amount of 500 samples. The second version applies the shifting to each channel independently. Here the maximum amount of shifted samples are parametrized as five.

Additive noise Adding small noise to the network's input is standard for most applications [7]. We apply additive white gaussian noise with a maximum amplitude of 20 μV.

DropOut [8] is a standard method to counteract overfitting by dropping single neurons in the hidden layers of fully connected classifiers. We apply DropOut with a probability of 50 %.

2.5. Network training

We trained our models using a Tesla K40 GPU (Amazon Webservices (AWS)) employing PyTorch as a deep learning framework. An Adam optimizer guided our optimization with a learning rate of $1e-4$ and a weight decay of $1e-5$. We applied the binary cross-entropy function as a loss function to deal with the multi-class-multi-label classification problem. Due to our unique signal length independence, we could not apply batched training but fed every signal separately through the network, while accumulating the gradients and the loss. After reaching the desired batch size of 64, we divided the loss by this amount, i.e., calculating the mean, and made an optimizer step. After that, we reset the gradients for every layer. We saved the model parameters based on the sum of AUROC and AUPRC during training on a left out validation dataset. We used 90 % of the data for training and 10 % for validation for our best model.

2.6. Network evaluation

As already mentioned, we evaluated the models during training using AUROC and AUPRC. These values measure model performance without the dependency on a decision threshold, while other values like accuracy and the challenge metric are snapshots of the model capability for one threshold. The threshold is a limit value, which decides for one class, whether the sample is positively classified or not. As there is one output probability for every class, the threshold relates to the minimum class probability that a sample needs to be classified positively. We chose 0.071 as the threshold over all classes by maximizing the challenge metric on the validation data set for the cross-validated models. The challenge metric is a newly developed metric by Perez Alday et al. [4] which weighs

diagnosis based on severity and similarity in treatment for different abnormalities.

3. Results

We evaluated our model locally using five-fold cross-validation on the training data. The model training on our hardware took about twelve hours per model. On the hidden test set, the network achieved a challenge metric of 0.27. The detailed results on the hidden test can be found in Table 1. Besides a single network, we also employed the models gained by cross-validation as an ensemble model, where we merged the decisions using a mean. We kept 10 % of testing data as a validation set for local testing and trained four models using four-fold cross-validation. Unfortunately, we could not test the ensemble on the hidden test set due to runtime restrictions for model training. Compared to our model trained with full data augmentation, we also trained the model without any data augmentation, but still with signal length independence. The results can be seen in Table 2.

Table 1. Detailed results of our model on the hidden test set of the challenge.

Validation	Set 1	Set 2	Set 3	**Full**
0.656	0.840	0.300	0.190	**0.270**

Table 2. Mean of the results from the five-fold cross-validation for our models with and without data augmentation. The last row states the results of the ensemble.

	AUROC	AUPRC	Challenge Metric
w/o aug.	0.943	0.567	0.602
with aug.	0.947	0.577	0.611
Ensemble	0.970	0.611	0.642

4. Discussion and Conclusion

We have presented a network capable of detecting different cardiac abnormalities in a diverse dataset of 12-lead ECG data. We showed that a residual neural network with minor modifications is capable of learning features from raw ECG signals with varying lengths. Our model reached the 15th of 41 officially ranked entries (Team name: UC_Lab_Kn). As reflected in the results, data augmentation improved generalization performance. Future research will be twofold. On the one hand, we will look further into data augmentation methods and their impact on training. On the other hand, further comparison with other network structures, e.g., fully convolutional networks independent of different input sizes, will be done. Also, we observed that ensembles of networks with different subsets of training data were able to push classification performance. Ensemble models are known to enhance performance from other machine learning techniques [9].

Acknowledgments

This research was partially funded by the EU Interreg V-Program "Alpenrhein-Bodensee-Hochrhein": Project "IBH Living Lab Active and Assisted Living", grants ABH40, ABH41 and ABH66 and a SJR grant from the HTWG Konstanz.

References

[1] Virani SS et al. Heart Disease and Stroke Statistics—2020 Update: A Report From the American Heart Association. Circulation 2020;141(9):E139–E596. ISSN 0009-7322.

[2] Ribeiro AH et al. Automatic Diagnosis of the 12-Lead ECG using a Deep Neural Network. Nature Communications 2020;11(1):1760. ISSN 2041-1723.

[3] Kligfield P et al. Recommendations for the Standardization and Interpretation of the Electrocardiogram. Circulation 2007;115(10):1306–1324. ISSN 0009-7322.

[4] Perez Alday EA et al. Classification of 12-lead ECGs: the PhysioNet/Computing in Cardiology Challenge 2020. Physiol Meas 2020;2020.08.11.20172601.

[5] He K et al. Deep Residual Learning for Image Recognition. In CVPR, volume 2016-Decem. IEEE. ISBN 978-1-4673-8851-1. ISSN 10636919, 2016; 770–778.

[6] He K et al. Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition. In Lecture Notes in Computer Science, volume 8691. ISBN 9783319105772, 2014; 346–361.

[7] Shorten C , Khoshgoftaar TM. A Survey on Image Data Augmentation for Deep Learning. Journal of Big Data 2019; 6(1):60. ISSN 2196-1115.

[8] Gal Y , Ghahramani Z. Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning. 33rd International Conference on Machine Learning ICML 2016 2015;3:1651–1660.

[9] Biau G. Analysis of a Random Forests Model. Journal of Machine Learning Research 2010;13(5):1063–1095. ISSN 1532-4435.

Address for correspondence:

Lucas Weber

Alfred-Wachtel-Str. 8, 78462 Konstanz, Baden-Württemberg, Germany

Lucas.Weber@htwg-konstanz.de