Small Convolutional Network for Arrhythmia Classification in Electrocardiogram Signals

Jorge Torres[1], Julio David Buldain[2]

[1] Accenture
e-mail: 651240@unizar.es
[2] Department of Electronic and Communication Engineering
University of Zaragoza, 50018, Zaragoza, Spain
e-mail: buldain@unizar.es

Abstract

Our main goal is to achieve a small neural network capable of identifying electrocardiogram segments with arrhythmic beats which can be embebed in a wearable device for real time monitoring an analysis.

Introduction

Electrocardiograms (ECG) are electrical signals that can be used to analyse heart activity. They have been broadly extended due to being a noninvasive technique that can record valuable information about heart activity, which is really helpful to diagnose heart pathologies.

Neural networks have been trained to analyze and classify heart pathologies from electrocardiograms signals with great accuracy, but they are frequently composed by large and complex models that require high processing power to execute, making them not suitable for portable devices with low specs, such as wearables.

Several studies have shown that new architectures can reduce neural networks complexity, decreasing the model size or the number of operations required to perform a task. For example, MobileNet [1] shown that separable layers could be used instead of traditional convolutional layers.

Separable layers perform a depthwise convolution to each input channel, followed by a 1x1 convolution, named pointwise convolution, in order to combine the outputs from the depthwise convolution. This operation drastically reduces the computational cost required by convolutional layers, while outputting almost the same accuracy.

Using these layers, Vernon et al. [2] proved that EEGNet model, a compact convolutional neural network (CNN), can achieve state-of-the-art results with a really limited number of parameters.

This model uses a two dimensional convolutional layer to process a 64 EEG records as input. Then, a depthwise layer followed by a separable layer are used to produce the outputs to be classified by a fully connected layer to determine the EEG class, achieving a great accuracy but with two order of magnitude fewer parameters than models with comparable classification results.

We wanted to test if this architecture could be used to create a small model capable of analyzing ECG signals and suitable for being embebed into a wearable device for applications such as patient monitorization.

To train and test our model, we used the ECG signals from the MIT-BIH Arrhythmia Database [3], which has great popularity in ECG analysis studies and it is continuously revised and annotated by independent cardiologists. It contains 48 half-hour ECG signals taken from two different derivations.

Methodology

We first normalized each signal from the MIT-BIH database, and then we resampled the signals from 360Hz to 512Hz. We extracted segments with 1.5 seconds duration from both derivations. We joined both derivation sections to create segments with 768 samples and 2 channels.

These segments were used to generate training, validation and test batches. For training we used 60k samples, selecting 10% to perform validation after each epoch, and 10k samples reserved for testing the trained models.

Once the models were trained, we calculated the ROC curve using the predicted values from the test samples. Then, we obtained the optimal cut-point to use as the baseline to classify segments between arrhythmic and not arrhythmic.

Revista "Jornada de Jóvenes Investigadores del I3A", vol. 9 (Actas de la X Jornada de Jóvenes Investigadores del I3A – 21 de octubre de 2021). ISSN 2341-4790.

Architecture

As first step, we wanted to test if the EEGNet model could be adapted to ECG characteristics, so we used a 2 channel signal as an input instead of the 64 channel input required for EEGs.

We also modified the size of the filters according to the ECG time characteristics, applied selu as layer activation and added a second fully connected layer.

Adam was chosen as the training optimizer with binary cross-entropy to check training performance, using accuracy as metric.

We tried several models, exploring from 8 to 128 filters in the convolutional layer, and from 8 to 40 filters in the separable layer. We also explored different number of hidden neurons in the dense layer.

Once the models were trained, we calculated their ROC curve, which reported 90% AUC along all models. Using as baseline the cut-point, we obtained the classification accuracy in test batches which was near 78% for all models.

With this architecture, as Fig. 1 shows, the smallest model only uses 4.8k parameters, while reporting a 78% accuracy in arrhythmia classification.

Stacked Separable Model

After seeing that different models achieved almost the same results, we tested if stacking separable layers could improve the classification accuracy. The model, shown in Fig. 2, uses three separable layers appended after the separable layer from the previous model.

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 2, 768, 8)	1024
batch_normalization (BatchNo	(None, 2, 768, 8)	32
depthwise_conv2d (DepthwiseC	(None, 1, 768, 16)	32
activation (Activation)	(None, 1, 768, 16)	0
average_pooling2d (AveragePo	(None, 1, 192, 16)	0
separable_conv2d (SeparableC	(None, 1, 192, 24)	896
activation_1 (Activation)	(None, 1, 192, 24)	0
average_pooling2d_1 (Average	(None, 1, 24, 24)	0
flatten (Flatten)	(None, 576)	0
dense (Dense)	(None, 5)	2885
dense_1 (Dense)	(None, 1)	6

Total params: 4,875
Trainable params: 4,859
Non-trainable params: 16

Fig. 1. The image shows architecture implemented, using a convolutional layer with a depthwise and a separable layers to compose a model with less than 5k parameters.

Also, we removed the average pooling layers, applying strides to convolutional and separable layers to decrease the output size.

This model improved the results from the previous architecture and reached an 82% accuracy in arrhythmia classification with test samples.

Conclusions and Future Work

We have trained neural networks with less than 20k parameters capable of classifying electrocardiogram segments between arrhythmic and not arrhythmic with 82% accuracy.

These models are suitable for being embebed in small devices such as wearables with low processing power.

Our next steps are focused on training and exploring different architectures that can improve the classifying accuracy without increasing the size or complexity of the network.

REFERENCES

1. Howard, A.G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., & Adam, H. (2017). MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. ArXiv, abs/1704.04861.

2. Lawhern, V.J., Solon, A.J., Waytowich, N.R., Gordon, S.M., Hung, C.P., & Lance, B. (2018). EEGNet: A Compact Convolutional Network for EEG-based Brain-Computer Interfaces. Journal of neural engineering, 15 5, 056013.

3. MIT-BIH Arrhythmia Database [Internet]. Harvard-MIT Division of Health Sciences and Technology.

1980- [cited 2021 Sep]. Available from: https://www.physionet.org/content/mitdb/1.0.0/

Fig. 2. Designed architecture for the stacked separable layers model, used strides instead of average pooling to decrease the output parameters from the previous layer.

```
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_2 (Conv2D)            (None, 2, 384, 8)         1024
_________________________________________________________________
batch_normalization_2 (Batch (None, 2, 384, 8)         32
_________________________________________________________________
depthwise_conv2d_2 (Depthwis (None, 1, 384, 16)        32
_________________________________________________________________
separable_conv2d_14 (Separab (None, 1, 192, 64)        2048
_________________________________________________________________
separable_conv2d_15 (Separab (None, 1, 96, 64)         6144
_________________________________________________________________
separable_conv2d_16 (Separab (None, 1, 48, 32)         3072
_________________________________________________________________
separable_conv2d_17 (Separab (None, 1, 24, 16)         768
_________________________________________________________________
flatten_2 (Flatten)          (None, 384)               0
_________________________________________________________________
dense_4 (Dense)              (None, 5)                 1925
_________________________________________________________________
dense_5 (Dense)              (None, 1)                 6
=================================================================
Total params: 15,051
Trainable params: 15,035
Non-trainable params: 16
```