# A Wide & Deep Transformer Neural Network for 12-Lead ECG Classification

Annamalai Natarajan, Yale Chang, Sara Mariani, Asif Rahman, Gregory Boverman, Shruti Vij and Jonathan Rubin
Philips Research North America, Cambridge, United States

## Abstract

*Cardiac abnormalities are a leading cause of death and their early diagnosis are of importance for providing timely interventions. The goal of 2020 PhysioNet/CinC challenge was to develop algorithms to diagnose multiple cardiac abnormalities using 12-lead ECG data. In this work, we develop a wide and deep transformer neural network to classify each 12-lead ECG sequence into 27 cardiac abnormality classes. Our approach combines handcrafted ECG features, which were determined to be important by a random forest model, and discriminative feature representations that are automatically learned from a transformer neural network. Our entry to the 2020 PhysioNet/CinC challenge placed $1^{st}$ out of 68 teams (team name = **prna**). Using the official generalized weighted accuracy metric for evaluation, we achieved a top score of 0.533 on the official held-out test set.*

## 1. Introduction

Cardiovascular diseases are the leading cause of death globally, resulting in 17.9 million deaths each year [1]. Early diagnosis of different cardiac abnormalities can help clinicians provide timely interventions and lead to improved clinical outcomes. The 2020 PhysioNet/CinC challenge focused on automated, open-source approaches for classifying cardiac abnormalities from 12-lead ECG [2]. Traditional approaches for cardiac abnormality classification often 1) apply signal processing techniques on the raw ECG data; 2) extract handcrafted features using domain knowledge to construct a feature vector; and 3) apply off-the-shelf machine learning models to classify the feature vector into different abnormality classes. Recently deep learning approaches have been applied to this task and achieve superior performance [3–6]. Compared to traditional approaches, deep learning-based approaches can automatically learn informative feature representations that are predictive of cardiac abnormalities in an end-to-end manner. However, it can be beneficial to incorporate expert knowledge, represented by handcrafted features, into deep learning models to enhance their performance.

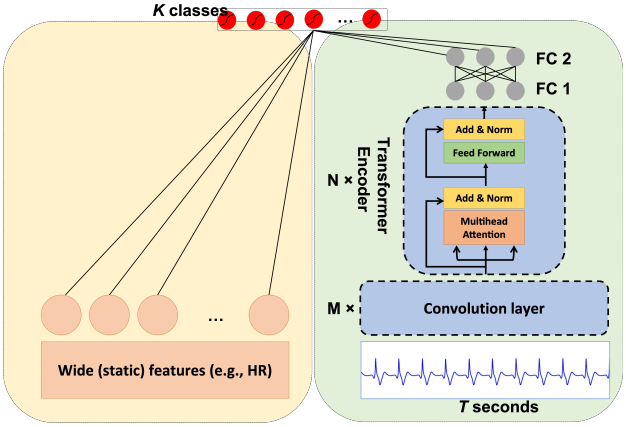


Figure 1. Architecture diagram of our wide and deep transformer neural network. The left hand side shows the the static "wide" features and the right hand side illustrates the deep network that consists of a series of convolutional layers followed by a Transformer encoder module and fully connected layers for multi-label classification.

## 2. Methods

In this work, we present a wide and deep transformer network for multi-label classification of 27 ECG findings, including for example, right and left bundle branch block, atrial fibrillation and flutter, bradycardia/tachycardia, premature beats and wave abnormalities and inversions. Our network combines static hand-crafted ECG features – so called *wide* features [7] – together with *deep* features extracted directly from the raw ECG waveform via a neural network. For encoding deep features, we first perform a series of convolution operations to learn an embedding of the raw ECG waveform. Learned embeddings are then fed into a Transformer [8] architecture, which relies entirely on a parallelizable self attention mechanism. We selected a Transformer architecture as it is faster to train than typical recurrent neural architectures that require sequential computation. A final set of fully connected layers combine both the 'wide' and 'deep' features to produce multi-label classifications of ECG findings. Figure 1 gives an overview of the complete system. In the following sec-

tions, we provide detailed explanations of each component of the system, as well as a discussion of the training and implementation details.

## 2.1. Pre-processing

Public training data supplied for the 2020 PhysioNet/CinC challenge was sourced from four locations. As recordings from separate hospitals could have different sampling rates, we first upsample or downsample each recording to 500Hz. We apply an FIR (finite impulse response) bandpass filter with bandwidth between 3 - 45 Hz. Each recording is also normalized so that each channels' signal lies within the range of -1 to 1. We extract random fixed width windows from each recording. We chose $T = 15$ seconds and we apply zero-padding, if the sequence length is less than $T$ seconds.

## 2.2. Wide and Deep Transformer Network

### 2.2.1. Wide

The wide [7] component of our model allows for the inclusion of static hand-crafted features into the network architecture. We initially extracted over 300 ECG features from lead II, including heart rate variability features (time, frequency domain and non-linear), as well as morphological features.

We trained an initial random forest model to investigate feature importance. Based on these results we selected the top 20 identified features[1] The top features selected are shown in Table 1. We also add the static features of age and gender to give a total of $d_{wide} = 22$ features. Collectively, these features were fed as input to the wide network and concatenated with the learned outputs from the "Deep" portion of the model (explained in the next section).

### 2.2.2. Deep

The second component of our model consists of the following modules:

1. An embedding network that extracts information directly from the ECG waveform segment
2. A Transformer [8] encoder stack
3. A multi-label classification head.

The embedding network applies a series of convolution operations to the original ECG waveform to capture the latent space representation of the signal. An approximate 20x downsampling factor is applied to the raw waveform, resulting in a sequence of embedded representations $(x_1, \ldots, x_n)$, where $x_i \in \mathbb{R}^{\text{emb}}$. Table 2 lists the details for each convolution operation applied.

The embedded representations are summed together with positional encodings, $p = (p_1, \ldots, p_n)$, representing the order of the sequence $p_i \in \mathbb{R}^{\text{emb}}$. The result, $e = (x_1 + p_1, \ldots, x_n + p_n)$, is then fed into a Transformer module. As the objective of our network is to perform classification as opposed to sequence prediction, our Transformer module utilizes only an encoder stack made up of $N = 8$ layers. Each of the encoder layers consists of a multi-head self-attention mechanism sub-layer, followed by a fully connected feed-forward network. As in [8], we utilize skip connections [9] and layer normalization [10] within each sublayer. We choose an embedding dimension $d_{model} = 256$ and use 8 multi-head attention layers running in parallel.

We use scaled dot-product attention [8] where for each embedded representation, $x_i$, a query, key and value vector are created ($q, k, v$, respectively) by multiplying $x_i$ by learned network weight matrices, $W^Q, W^K, W^V$. The $q, k$ and $v$ vectors are stacked into matrices $Q, K, V$ and self attention is applied using Equation 1, where $d_k$ indicates the dimension of the key vector, $d_k = 64$.

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right) \quad (1)$$

Following the transformer module, global average pooling is performed column-wise, followed by a fully connected layer to produce $d_{deep}$ deep features. We choose $d_{deep} = 64$. The $d_{wide}$ features from Section 2.2.1 are concatenated together with the $d_{deep}$ features and a final fully connected layer is applied to match the number of classes, $d_{class}$.

## 2.3. Implementation Details

We excluded from consideration the 84 unscored classes and chose to classify only SNOMED CT codes that were included in the challenge evaluation metric [2], i.e $d_{class} = 27$. We utilized a standard binary cross entropy loss function averaged over each of the $d_{class}$ classes. During model training we monitored average AUC and used early stopping when validation AUC had stopped improving for 5 epochs.

We used the Noam optimization [8] procedure, which wraps an Adam optimizer ($\beta_1 = 0.9, \beta_2 = 0.98$ and $\epsilon = 10^{-9}$) within a scheduling routine that increases the learning rate linearly during a warm-up stage and subsequently decreases it, proportional to the inverse square root of the step number [8]. We chose 4000 as the number of warm-up steps. The complete model consists of 13,643,885 trainable parameters, which were initialized using Xavier uniform initialization [11]. Our model was

[1] Waveform duration was identified as the top predictive feature, however we excluded this feature, as to not bias the classifier by learning false associations between findings and sequence length.

| Rank | Feature |
|---|---|
| 1 | $HR_{min}$ |
| 2 | T wave multiscale permutation entropy $\sigma$ |
| 3 | $HR_{max}$ |
| 4 | T wave multiscale permutation entropy median |
| 5 | RMSSD |
| 6 | P wave correlation coefficient |
| 7 | RR interval median |
| 8 | Heart rate $\mu$ |
| 9 | RR interval intra-cluster distance, cluster 3 |
| 10 | RR interval Fisher information |
| 11 | pNN60 |
| 12 | SWT decomposition level 4 entropy |
| 13 | RR interval intra-cluster distance, cluster 2 |
| 14 | Heart rate activity |
| 15 | $\Delta$ $RR_{min}$ |
| 16 | T wave permutation entropy $\sigma$ |
| 17 | P wave sample entropy $\sigma$ |
| 18 | SWT decomposition level 3 entropy |
| 19 | Median p wave approximate entropy |
| 20 | R peak approximate entropy |

Table 1. Top 20 features as discovered from a Random Forest model, $\mu, \sigma$ are mean and standard deviation, $\Delta$RR is the difference of RR intervals and SWT stands for stationary wavelet transform.

| Conv. layer | Input size | Output size | Kernel size | Stride | Padding |
|---|---|---|---|---|---|
| 1 | 12 | 128 | 14 | 3 | 2 |
| 2 | 128 | 256 | 14 | 3 | 0 |
| 3 | 256 | 256 | 10 | 2 | 0 |
| 4 | 256 | 256 | 10 | 2 | 0 |
| 5 | 256 | 256 | 10 | 1 | 0 |
| 6 | 256 | 256 | 10 | 1 | 0 |

Table 2. Convolution layer settings

trained on the data-sets that were made publicly available for the 2020 PhysioNet/CinC challenge and no other external data sources were used. Models were training using PyTorch on two P100 GPUs with a batch size of 128. Each epoch took roughly 5 minutes to train and convergence typically took place within 30 epochs. Table 3 summarizes the selected hyper-parameters.

## 3. Results

We begin by first splitting the data-set into 10 folds using iterative stratification [12]. We train and evaluate our models using 10-fold *nested* cross-validation, whereby eight folds are used for model training, one for validation and the remaining acts as a held-out test set. This is repeated ten times to produce ten trained models and the

| **Hyper-Parameter** | **Value** |
|---|---|
| **Global** | |
| ECG window size (secs) | 15 |
| Sampling frequency (Hz) | 500 |
| Batch size | 128 |
| Wide feature size, $d_{wide}$ | 22 |
| Deep feature size, $d_{deep}$ | 64 |
| Number of classes, $d_{class}$ | 27 |
| **Transformer** | |
| Number of encoding layers | 8 |
| Embedding size, $d_{model}$ | 256 |
| Number of heads | 8 |
| Dimension of feed forward layer | 2048 |
| Size of key, query and value vectors, $d_k, d_q, d_v$ | 64 |
| Dropout | 0.1 |
| **Fully connected layers** | |
| FC 1 size | 64 |
| FC 2 size | 27 |
| Dropout | 0.2 |

Table 3. A listing of hyper-parameters selected to train the wide and deep neural network model for ECG finding classification.

| Fold | 10 seconds | 15 seconds |
|---|---|---|
| 1 | 0.519 | 0.452 |
| 2 | 0.508 | 0.486 |
| 3 | 0.459 | 0.481 |
| 4 | 0.530 | 0.581 |
| 5 | 0.484 | 0.587 |
| 6 | 0.508 | 0.565 |
| 7 | 0.484 | 0.566 |
| 8 | 0.484 | 0.556 |
| 9 | 0.493 | 0.525 |
| 10 | 0.484 | 0.532 |
| **Average** | **0.496 $\pm$ 0.021** | **0.533 $\pm$ 0.046** |

Table 4. 10-fold nested cross validation results showing challenge scores for different window sizes.

challenge score of each is averaged to get an estimate about how well the model is performing. Cross validation results are shown in Table 4.

We submitted an ensemble of the trained models to the challenge server. Due to computational and training time limitations we restricted our final submitted model to an ensemble of the top three performing models from Table 4. Each model in the ensemble uses a separate threshold to make binary decisions for each of the $d_{class} = 27$ classes. Optimal thresholds were computed on the models' validation set to maximize the challenge metric. A final

| Ranking | Team name | Test Database 1 | Test Database 2 | Test Database 3 | Full Test Set |
|---|---|---|---|---|---|
| $1^{st}$ | **prna** | 0.761 | 0.558 | 0.492 | **0.533** |
| $2^{nd}$ | Between_a_ROC_and_a_heart_place | 0.845 | 0.639 | 0.412 | 0.520 |
| $3^{rd}$ | HeartBeats | 0.852 | 0.649 | 0.396 | 0.514 |

Table 5. The official Top 3 results on the full test set of the PhysioNet/CinC challenge

classification decision is then made by combining the binary decisions from each classifier. In particular, we used the strategy whereby a class was predicted to be positive, if any classifier in the ensemble predicted that class to be positive. The final results are shown in Table 5. Our entry, ***prna***, achieved the top score $(0.533)$ on the full test set. Table 5 compares our approach to the $2^{nd}$ and $3^{rd}$ place finishers in the challenge.

## 4. Discussion and Conclusions

We made several observations about the performance of our wide and deep transformer network:
1. Bradycardia was trickier to score than just merely using $HR < 60$ bpm. Some cases that looked like bradycardia were actually not labeled as such by the classifier.
2. Low QRS voltages were most often confused with atrial fibrillation, T wave abnormality or sinus rhythm. It is possible that the max-min normalization performed in the data pre-processing step may have influenced this.
3. T wave abnormality or inversion was sometimes confounded by noise.
4. Atrial flutter was often mixed up with atrial fibrillation.
5. Prolonged QT intervals were sometimes over detected (when the rhythm was actually normal), this is likely due to difficult delineation of the fiducial points in borderline cases.

In addition to the Transformer module in the architecture described above, we also experimented with using recurrent neural networks instead (in particular, GRU). We found that the Transformer module did not necessarily lead to better performance than an RNN module, but that it did allow for faster model training and iteration. Finally, the Noam optimization [8] procedure was observed to be crucial for achieving a reasonable challenge score when using the Transformer module and we observed large differences in final scores achieved if it was not used. In future work, we would like to utilize information from the remaining 88 classification labels, which were not included in the challenge evaluation metric, to take advantage of any potential label interactions.

## References


[1] Cardiovascular diseases. `https://www.who.int/health-topics/cardiovascular-diseases/#tab=tab_1`.

[2] Alday EAP, Gu A, Shah A, Robichaux C, An-Kwok, Wong I, Liu C, Liu F, Rad AB, Elola A, Seyedi S, Li Q, Sharma A, Clifford GD, Reyna. MA. Classification of 12-lead ECGs: the PhysioNet/Computing in Cardiology Challenge 2020. Physiological Measurement 2020;.

[3] Hannun AY, Rajpurkar P, Haghpanahi M, Tison GH, Bourn C, Turakhia MP, Ng AY. Cardiologist-level arrhythmia detection and classification in ambulatory electrocardiograms using a deep neural network. Nature Medicine 2019; 25(1):65.

[4] Ribeiro AH, Ribeiro MH, Paixão GM, Oliveira DM, Gomes PR, Canazart JA, Ferreira MP, Andersson CR, Macfarlane PW, Meira Jr W, et al. Automatic diagnosis of the short-duration 12-lead ECG using a deep neural network: The CODE study. arXiv preprint arXiv190401949 2019;.

[5] Hong S, Zhou Y, Shang J, Xiao C, Sun J. Opportunities and challenges in deep learning methods on electrocardiogram data: A systematic review. arxiv 2019. arXiv preprint arXiv200101550 ;.

[6] Parvaneh S, Rubin J, Babaeizadeh S, Xu-Wilson M. Cardiac arrhythmia detection using deep learning: A review. Journal of electrocardiology 2019;57:S70–S74.

[7] Cheng HT, Koc L, Harmsen J, Shaked T, Chandra T, Aradhye H, Anderson G, Corrado G, Chai W, Ispir M, et al. Wide & deep learning for recommender systems. In Proceedings of the 1st Workshop on Deep Learning for Recommender Systems. 2016; 7–10.

[8] Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, Kaiser Ł, Polosukhin I. Attention is all you need. In Advances in Neural Information Processing Systems. 2017; 5998–6008.

[9] 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.

[10] Ba JL, Kiros JR, Hinton GE. Layer normalization. arXiv preprint arXiv160706450 2016;.

[11] Glorot X, Bengio Y. Understanding the difficulty of training deep feedforward neural networks. In Proceedings of the Thirteenth International Conference on Artificial Intelligence and Atatistics. 2010; 249–256.

[12] Szymański P, Kajdanowicz T. A scikit-based Python environment for performing multi-label classification. arXiv e prints February 2017;.



Address for correspondence:

Jonathan Rubin
222 Jacobs St Cambridge, MA 02141, United States
jonathan.rubin@philips.com