Classification of ECG Signals with Different Lead Systems Using AutoML

Matteo Bodini, Massimo W Rivolta, Roberto Sassi

Dipartimento di Informatica "Giovanni Degli Antoni", Università degli Studi di Milano, Milan, Italy

Abstract

The PhysioNet 2021 challenge asks participants to develop automated techniques for classifying cardiac abnormalities (CA) from both 12-lead electrocardiogram (ECG) and reduced-lead settings. We investigated on the feasibility of applying Automated Machine Learning (AutoML) approaches to build ECG classifiers. Standard ECG preprocessing was applied beforehand to the ECG (filtering and resampling). Three different AutoML frameworks were executed on the 88,000+ ECGs made available by the challenge organizers. The optimal combination of preprocessing and ML algorithms were found by the AutoML frameworks. We finally assessed the frameworks' classification performance, the effect of the number of employed leads, and the effect of extending the frameworks training time. The classifiers proposed by our team "BiSP_Lab" obtained a challenge score up to 0.35 on the private test set. The AutoML frameworks showed comparable performance. The worst score was obtained on the 12-lead system, while the best on the 6-lead one. Significantly extending the training time seemed to not improve the test score. AutoML frameworks showed promising performance on the private test set, suggesting their potential for classification of CA. Future works are towards testing further AutoML approaches, and better determining the impact of the available training time on the classification performance.

1. Introduction

The 12-lead electrocardiogram (ECG) is a fundamental clinical tool for diagnosing several cardiac abnormalities (CA) [1]. The automatic interpretation and correct diagnosis of CA largely increases the odds that their treatments become successful [2]. However, most of the available algorithms which interpret ECGs provide the diagnosis of a relatively small amount of CA. Thus, to cover up for their actual huge amount, several algorithms need to be implemented with the hassle of merging their predictions [2].

Similar issues yet occurred in Computer Vision, where designing algorithms to classify hundreds of classes became unfeasible. Further, in case the formulation of the underlying problem would change, *e.g.* by considering a new class, all such algorithms would need a time-consuming phase of re-train. Within this context, Machine Learning (ML) algorithms and Deep Neural Networks (DNN) have been introduced to interpret ECGs. DNN tackle the problem by defining a mathematical model flexible to changes and easier to update when new classes become available.

A few studies relied on DNN for ECG classification and reached promising results. Ribeiro *et al.* [3] trained a DNN on $>$2,000,000 12-lead ECGs, to distinguish among 7 different CA. Hannun *et al.* [4] designed a DNN relying on $>$95,000 single-lead ECG to detect 12 different CA.

The Computing in Cardiology 2021 challenge [5] extended the previous year's challenge [6] by asking participants to develop automated techniques for detecting and classifying CA from both 12-lead ECG recordings and reduced-lead ECG recordings by means of automatic algorithms. The challenge organizers provided a dataset containing 88,000+ recordings of clinical ECGs, collected from multiple sources, along with their diagnosis.

Despite the promising results of ML, the proper parameters and hyperparameters must be carefully selected to let ML algorithms perform at their best. In the case of DNN, the architectures are time-consuming to design, thus often borrowed from other domains. They usually need a huge amount of data for training, while being susceptible to class imbalance. To address such drawbacks, we investigated on Automated Machine Learning (AutoML) approaches to automatically find the optimal parameters and hyperparameters for ML algorithms, and architectures for DNN models, to distinguish CA from ECGs.

2. Materials and Methods

2.1. Dataset

The dataset made available for the challenge was composed of 12-lead, 6-lead, 4-lead, 3-lead, and 2-lead ECGs in WFDB format, labeled with one or more CA, among 133 possible ones [5]. The dataset was obtained merging data from seven institutions in four countries across three continents. The ECG recordings lasted from 6 seconds to 30 minutes and sampling rates ranged from 257 Hz to 1000 Hz, where the majority was sampled at 500 Hz. A total

number of 88,253 ECG signals was available. The performance of the submitted classifiers were assessed using an expert-based scoring metric provided by the challenge organizers which assessed the classifiers' performance only relying on a subset of 27 selected CA [6]. A private test set of 42,902 ECGs handled by the challenge organizers was used for evaluating the proposed algorithms. A maximum of 72h was allowed for training time and 24h for testing.

2.2. Preprocessing

We only relied on the dataset provided for the challenge, and on signals labeled with the CA considered in the challenge scoring metric defined by the organizers. Standard ECG preprocessing was applied beforehand to raw ECGs, including filtering and resampling. ECGs were downsampled or upsampled to 125 Hz according to their actual sampling rate and filtered with a bandpass Butterworth filter (3^{rd} order, zero phase, and pass-band: $0.67 - 30$ Hz) to reduce powerline interference, baseline wandering and high frequency noise. For each lead system, only the first 10 seconds of ECG were considered. In case the length was inferior to 10 seconds, zero padding was performed. The available ECGs were randomly split into training and validation sets with 70/30 ratio for each lead system, with stratification (*i.e.*, the class distribution of the training set matched the one of the validation set).

2.3. The AutoML Frameworks

We adopted three different AutoML frameworks to build the required ECG classifiers, *i.e.* auto-sklearn, AutoKeras, and the Tree-Based Pipeline Optimization Tool (TPOT). The AutoML frameworks are capable of automatically selecting the optimal preprocessing steps, which include data preprocessing and feature preprocessing, and ML algorithm, along with its trainable parameters and hyperparameters, for the problem and dataset at hand [7].

The auto-sklearn framework is an AutoML system based on the Python scikit-learn library [8]. It relies on 15 ML classifiers, 4 data preprocessing methods, and 14 feature preprocessing methods, giving rise to a high-dimensional hypothesis space. Onto such space, auto-sklearn defines a Combined Algorithm Selection and Hyperparameter optimization (CASH) problem and it relies on Bayesian Optimization to discover a top-performing ML pipeline. With the term "pipeline", we hereinafter refer to the preprocessing (data and feature preprocessing) and adopted ML/DNN classifier, along with its parameters and hyperparameters, to solve the problem at hand.

The AutoKeras framework is an AutoML tool specific for DNN, based on the Python Keras library [9]. It exploits the concept of network morphism, which retains the functionality of a DNN while changing its underlying architecture. Bayesian optimization is leveraged by AutoKeras to guide the network morphism in searching of the optimal DNN architecture for the considered problem and dataset. To efficiently explore the search space, the authors of the AutoKeras framework developed a custom neural network kernel along with a tree-structured optimization algorithm.

The TPOT framework automatically constructs and optimizes ML pipelines relying on the well-known evolutionary computation technique of genetic programming (GP) [10]. At the beginning of every TPOT run, a fixed number of pipelines is generated to constitute what is usually called in GP as population. GP is used to evolve the set of pipelines that acted on the dataset, and a portion of those is retained relying on their classification performance. The top-performing pipeline is retained when TPOT reaches convergence or after a user-defined number of runs.

2.4. Experiments on AutoML Frameworks

The AutoML frameworks were trained on the available dataset with the aim of 1) comparing the performance among the three considered frameworks; 2) assessing the effect of the number of employed leads on the final classification performance; 3) assessing the effect of extending the training time at disposal of the AutoML frameworks.

The input features were set as the reduced 10 seconds ECGs for each lead system, and the respective validation sets were employed by the frameworks to select the optimal ML pipeline. By default, AutoML frameworks use the validation loss of the employed ML/DNN algorithm as a score for selecting the best pipeline. For each AutoML framework, we set the challenge score defined by organizers as scoring function to measure the performance of the created pipelines. Each AutoML instance which follows comes with a parenthesized name to easily refer to it.

Auto-sklearn was tested by setting 2.5h of training time for each lead system (auto-sklearn #1), and setting a proportional training time to the number of leads of $(2.5h \times \#\text{leads})$ for each lead configuration (auto-sklearn #2). The whole set of classifiers, feature preprocessing methods, and data preprocessing methods was considered.

AutoKeras was tested relying on a training time of $(2.5h \times \#\text{leads})$ for each lead configuration and using the full set of pipeline elements at disposal (AutoKeras #1). Next, the hypothesis space was reduced to consider only DNN composed of Convolutional, Dense, ResNet, and Xception layers, and by mantaining the same training time of the previous configuration (AutoKeras #2).

TPOT was tested under two configurations with a training time of $(2.5h \times \#\text{leads})$. The TPOT Default configuration was used to search over a broad range of pipeline elements, where some of them may take a long time to run, especially on large datasets (TPOT #1). Then, the TPOT Light configuration was tested, in which TPOT searched

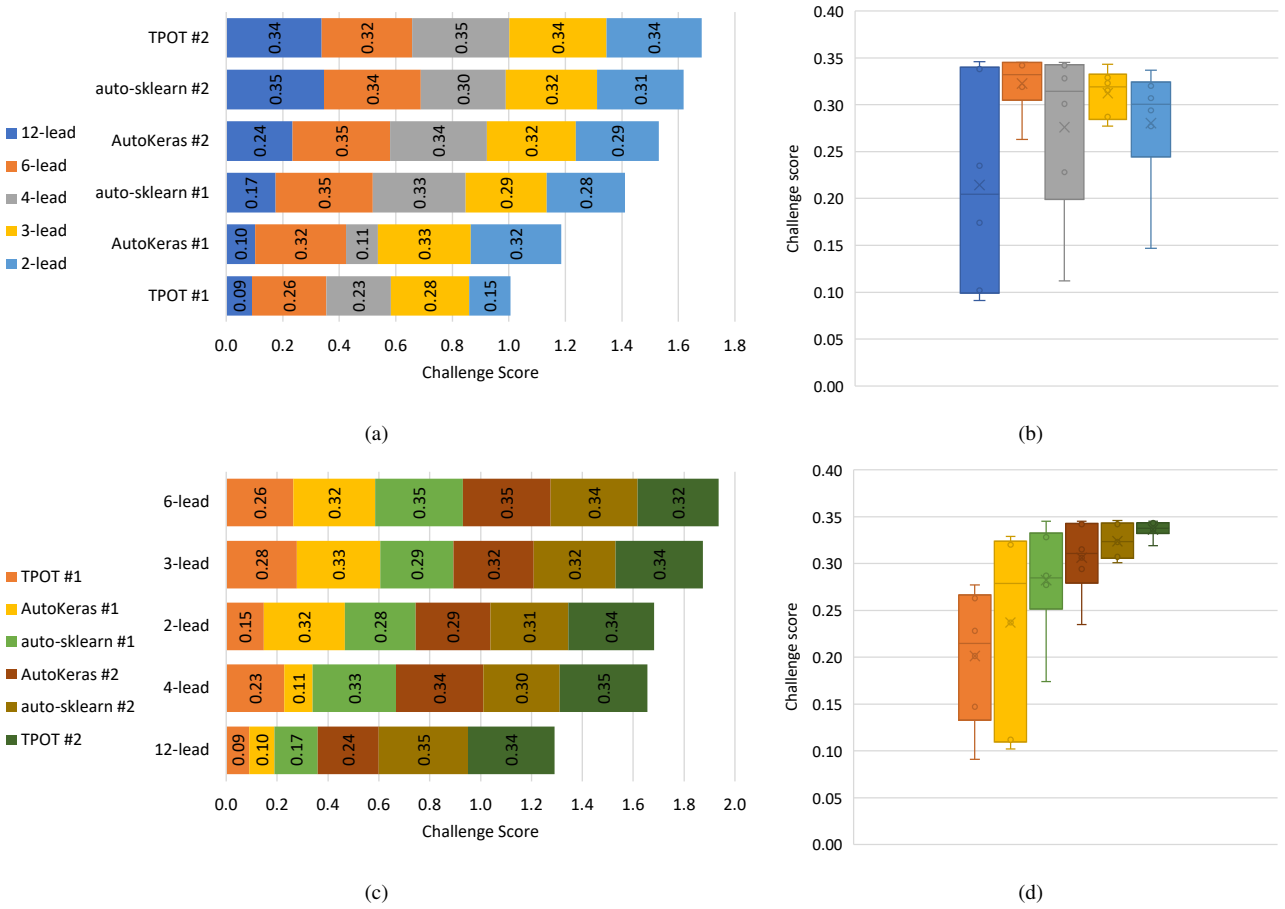

Figure 1: (a) The cumulative sum of challenge scores of AutoML instances on each lead system. (b) The box-plots of the challenge scores computed over each lead system. (c) The cumulative sum of challenge scores obtained over each lead system by the AutoML instances. (d) The box-plots of challenge scores computed over each AutoML instance.

over a restricted range of simple and fast-running pipeline element to find quick and simple ML pipelines (TPOT #2).

To assess the effect of extending the training time at disposal of the AutoML frameworks, as a final test we considered only the 3-lead system and we trained each AutoML framework for 70h onto such system (while we used pre-trained ML models for the remaining lead systems in the instance submission phase). For each AutoML system, the settings of the top performance instance were selected.

3. Results

To compare the performance among the considered AutoML frameworks, we computed in the Figure 1a the cumulative sum of challenge scores obtained by the AutoML instances on each lead configuration. The sub-bars report the score obtained by an AutoML instance on a specific lead system. To assess the overall variability of all the AutoML instances over lead systems, we reported in the Figure 1b the box-plots of challenge scores computed over each lead system.

To quantify the effect of the number of employed leads on the final classification performance, similarly to Figure 1a, we computed in the Figure 1c the cumulative sum of the challenge scores obtained over each lead system by the AutoML instances. For each lead system, the sub-bars report the score obtained by an AutoML instance. To assess the variability of a specific AutoML instance over lead systems, we reported in the Figure 1d the box-plots of challenge scores computed over each lead system.

Finally, after training auto-sklearn #2, AutoKeras #1, and TPOT #2 for 70h relying onto the 3-lead ECGs, we respectively obtained 0.32, 0.33, and 0.35 challenge scores.

4. Discussion

As shown in the Figure 1a, TPOT #2 was the best among the six instances in terms of cumulated score, and

it reached the highest score values in three out of five lead systems (up to 0.35 with 4-leads). The worst performance were provided by TPOT #1, since it reached the lowest challenge score values in four out of five lead systems. The Figure 1d shows that TPOT #2 and TPOT #1 were the ones with the highest and lowest median challenge score value computed across lead systems, respectively of 0.34 and 0.21. The instance TPOT #2 showed the lowest interquartile range (IQR) of 0.01, while the highest IQR of 0.14 was reached by TPOT #1. The highest IQR obtained by TPOT #1, associated with the lowest median score value, suggests that in this case the used AutoML configuration may be weak in classifying CA, since it searched into a limited hypothesis space.

The Figure 1c shows that the 12-lead configuration is the one where instances obtained the lowest performance. On the other hand, the 6-lead system showed the highest cumulated score. The Figure 1b shows that the 6-lead and the 12-lead systems were the ones that showed respectively the highest and lowest median score value computed across AutoML instances, respectively of 0.33 and 0.20. The 6-lead system obtained the lowest IQR of 0.04, while the highest IQR of 0.24 was obtained by the 12-lead system. The results on the 3-lead are comparable the to 6-lead system, as it is reasonable to expect since the second system is a linear combination of former. The highest IQR obtained by the 12-lead system, associated with the lowest median score value, suggests that in this case AutoML frameworks might need further training time to match the performance obtained in the case of less numerous lead systems.

The experimented AutoML frameworks obtained intermediate classification performance with respect to other teams, reaching up to 0.35 challenge score. Since the class distribution of the available dataset was not balanced, a cost-sensitive learning approach was leveraged to face the class imbalance problem: we executed the AutoML frameworks to find optimal pipelines relying on the challenge score function, instead of the standard loss functions of the employed ML/DL algorithms. The misclassification costs defined for CA by expert physicians helped in learning the few represented class, by considering their misclassification cost within the knowledge domain.

Further aspects of AutoML frameworks may be explored in the future. A wide number of AutoML tools is arising in the recent literature and more AutoML frameworks may be tested to address ECG classification, even if recent works suggest that their performance is relatively similar [7]. Next, even if the impact of increasing the training time did not significantly improve the performance in the case of the 3-lead system (not more than 0.03 of the challenge score), a systematic assessment of performance against training time may be investigated even for other lead configurations. A full analysis was not performed due to the limited number of instance submissions available, that was 10. However, the missed improvement in performance may be in line with results of recent works, which showed that most of AutoML frameworks tend to converge to similar performance in a few hundreds of iterations [7].

Differently from previous works where ML algorithms where often applied without a deep exploration of ML pipelines, and designs of DNN architectures were inspired from other domains, we tested AutoML approaches to manage the proper choice of the optimal ML pipeline, and at the same time to face the class imbalance problem with the aim of cost-sensitive learning. The approach seemed suitable to deal efficiently with the challenging problem of ECG classification and the unbalanced dataset available.

References

[1] Kligfield P, Gettes LS, Bailey JJ, et al. Recommendations for the standardization and interpretation of the electrocardiogram. J Am Coll Cardiol 2007;49(10):1109–1127.

[2] Schläpfer J, Wellens HJ. Computer-interpreted electrocardiograms. J Am Coll Cardiol 2017;70(9):1183–1192.

[3] Ribeiro AH, Ribeiro MH, Paixão GMM, et al. Automatic diagnosis of the 12-lead ECG using a deep neural network. Nat Commun 2020;11:1760.

[4] Hannun AY, Rajpurkar P, Haghpanahi M, et al. Cardiologist-level arrhythmia detection and classification in ambulatory electrocardiograms using a deep neural network. Nat Med 2019;25(1):65–69.

[5] Reyna MA, Sadr N, Perez Alday EA, et al. Will two do? Varying dimensions in electrocardiography: the PhysioNet/Computing in Cardiology challenge 2021. In Comput in Cardiol 2021, volume 48. IEEE, 2021; 1–4.

[6] Perez Alday EA, Gu A, Shah A, et al. Classification of 12-lead ECGs: the PhysioNet/Computing in Cardiology challenge 2020. Phys Meas 2021;41(12):124003.

[7] Zöller M, Huber MF. Benchmark and survey of automated machine learning frameworks. J Artif Int Res 2021;70:409–472.

[8] Feurer M, Klein A, Eggensperger K, et al. Efficient and robust automated machine learning. In Proceedings of the 28th NIPS. MIT Press, 2015; 2962–2970.

[9] Jin H, Song Q, Hu X. Auto-keras: An efficient neural architecture search system. In Proceedings of the 25th SIGKDD. ACM, 2019; 1946–1956.

[10] Le TT, Fu W, Moore JH. Scaling tree-based automated machine learning to biomedical big data with a feature set selector. Bioinformatics 2020;36(1):250–256.

Address for correspondence:

Matteo Bodini
Dipartimento di Informatica “Giovanni Degli Antoni”,
Università degli Studi di Milano,
Via Celoria 18, 20133 Milan, Italy,
matteo.bodini@unimi.it