Neural Networks (AI) – Semester Project Suggestion "Classifying pathological heartbeats from ECG signals"

Overview. This project suggestion concerns the task to identify and classify pathological heartbeats from electrocardiographic (ECG) recordings. Many things can go wrong with that powerful blood pump in your chest, and the medical relevance of reliable detection of non-normal heartbeat conditions are obvious. There is a very well documented, extensive dataset of clinical ECG measurements available which has frequently been used in machine learning studies on temporal pattern classification – the *MIT-BIH arrhythmia database* (Moody et al 2001) which is maintained at the MIT for the PhysioNet services of the NIH (Goldberger et al 2000). The data are freely available at https://physionet.org/content/mitdb/1.0.0/. The data format of this original database needs special tools to read, but there is a convenient version in csv format at https://www.kaggle.com/mondejar/mitbih-database provided by Victor Mondejar. The database consists of 48 half-hour high-resolution recordings of 47 patients with a wide spectrum of cardiologic conditions. Figure 1 shows some important classes of pathological ECG patterns.

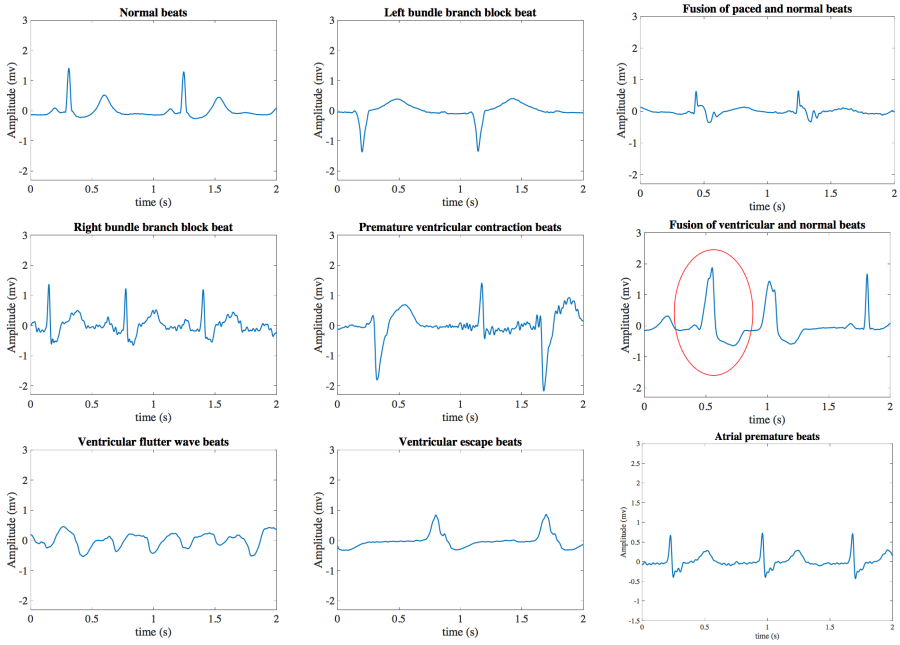

Figure 1. Examples of normal (first panel) and some pathological heartbeat ECG patterns, found in the MIT-BIH arrhythmia database. The database contains instances of several more irregularities not shown here (altogether 17 pathological types). Figure taken from Hadaeghi (2019).

This database has been used in machine learning and medical diagnostic studies in many ways. There is a rich literature about it, but there does not exist a standard objective of what, exactly, should be modeled / classified / recognized in these data. Every author seems to explore these nice data in different ways, which makes it hard to compare one's own results with a "state of the art".

The holy grail for a machine learning modeler would be to train a general-purpose recognizer which, when given a new ECG trace from any new patient as input, can identify and classify non-normal beats. This is what a highly trained human cardiologist can do. For machine learning this is currently (I would think) out of reach because different patients give quite idiosyncratic recording signals, and the available training data (not confined to this specific database) are scarce and differ according to the recording instruments that are used. Thus, a more accessible task is to train individual models per patient from data recorded from that patient with a fixed recording instrument. This is what was done in the recent work of Hadeaghi (2019) which I recommend as a starting point for your project.

Suggestions for concrete project goals. You are (of course) free to define your own project goals. Reading papers on research based on this dataset would help you to formulate interesting objectives. Here are two suggestions which you might directly adopt or modify:

- Hadeaghi (2019) trained patient-individual RNN models to identify and classify irregular beats in the ECG signal obtained from that patient. Thus an RNN was freshly trained for each patient, using some part of the available recording from that patient as training data, and some other part for testing. The network was trained only for those arrhythmia types that actually occurred in the respective patient. Altogether 17 patients were modeled in this way. In almost all cases, the obtained results were extremely good. However, for two patients (Nr 207 and 209) the classification accuracy was much poorer than for the other patients. It would be an interesting project to study these two patients in a close-up investigation, try to improve the classification accuracy, and/or find out what characteristics in the data made these two cases more difficult to model than the other 15 patients.

- A common, but not easy to reach, goal in machine learning is *transfer learning*. The idea is to train some system, let us say a pattern classification CNN, on some dataset and task setting A, then train it on another, related task B using other data, such that some information which the CNN had learnt during the task A training can be transferred to learning task B. For instance, train an image classification CNN on a collection of photos taken from the transport domain for classifying cars vs. vans vs. buses = data and task setting A. Then take the trained network and train it further on a new collection of photos of humans with the objective to classify male vs. female persons. This kind of setting is particularly promising and useful if the dataset for A is much larger than the dataset available for B. The hope is that during training for task A, the CNN has distilled some information about perceiving low-level graphical elements in photos in general – the "early visual processing" part of the task. In order to preserve this information for task B, a standard approach is to identically re-use the first layers of the A-trained CNN (which are considered "early visual processing"), replace a few of the last layers by untrained ones, then train the hybrid half-A-trained CNN on task B data while freezing the weights in the early layers that had already been learnt.

  You could do a similar thing with the MIT-BIH arrhythmia data. Using 1-dimensional CNNs whose inputs are (for example) 1-second windows from ECG traces, train it on all patients' data, that is the entire MIT-BIH dataset, where the teacher output are classification indicator neurons for all the types of pathological heartbeats that this database includes. This task will likely prove to be (too) difficult and the results will likely not be satisfactory, but in this training process the early layers of the CNN will have learnt to distil all sorts of medically relevant features from the input patterns. Then re-train only the last few layers (maybe only the single very last one) of this pre-trained CNN on individual patients. The hope is that this CNN can benefit from the large size of the complete dataset to "understand" much about "how ECG signals generally look like", and transfer this knowledge to the individual patient. Compare the results that you get in this way with the results of CNNs trained exclusively on data of single patients.

Unbalanced data problem. One difficulty with these data is that normal heartbeats occur much more often than pathological ones. You may want to (actually you should) use some *balancing* method to prevent the network to specialize on recognizing the normal beats very well, which gives an overall low training and testing error but poor recognition of the few anomalous beats. Google "data balancing machine learning" to get a load of tutorials.

Quantifying classification performance in medical applications. In medical applications, there are several important and different ways to quantify the goodness of a pattern classifier. In particular, false negatives (incorrectly classifying a pathological pattern as normal) and false positives (classifying a normal heartbeat as pathological) are both important types of classification errors with different clinical consequences. In the literature in this field, some standard performance measures are universally used: *accuracy*, *sensitivity*, *precision*, and a compound score called the F1-score. You find the definitions in Hadaeghi (2019).

A difficulty you have to solve is that these measures are non-differentiable and cannot be used as a loss function for gradient descent optimization. You have to choose another, differentiable loss function for the network training, for instance the quadratic loss, L1-norm loss, or logistic regression (probably a good candidate). You'll probably be using a machine learning toolbox where you can easily choose between these (and more) loss functions – it will need experimentation to find out which one gives the best accuracy, sensitivity, precision, F1-score, etc.

Preprocessing. ECG data often are corrupted with 50 Hz signals resulting from the power supply of the recording instrument. Furthermore, such recordings often have slow drift components (the signal generally rises up and down, an artifact from the measurement procedure). I recommend to follow Hadaeghi (2019) and pass all timeseries through a frequency bandpass filter with a suitable lower and higher cutoff to cancel both the 50 (or 60?) power-line oscillations and slow drifts. Hadaeghi used a Butterworth filter with cutoffs at 0.4 and 45 Hz.

Network type. This task can be tackled both with feedforward and recurrent NNs.

- If you use feedforward nets, the default kind of inputs are fixed-width windows from the training/testing timeseries, possibly subsampled. You can also replace this input by, or enrich it with, descriptive features that you define, for instance peak amplitude, peak gradient, variance or entropy measured in the input window. You will have as many output neurons as there are anomality classes that are relevant for the task, plus one output neuron for normal beats. The teacher signal per output unit can be defined in several ways. For instance, it could be a binary 0-1 signal which you set to 1 if and only if the input window captures a full pathological beat of the respective anomality class. You can also go for graded teachers with values in the continuous range [0, 1] depending of what fraction of an anomalous heartbeat is found in the input window.

- If you use RNNs, the input signal is the timeseries of a recording, possibly sub-sampled or otherwise preprocessed. The output signal to be trained has as many channels as there are pathological heartbeat types that you want to identify, plus one for "normal". It is not obvious and indeed important for good results how to define the teacher signal for a given channel. For instance, it could be a binary signal that jumps to 1 exactly at those time points where the expert annotation of the data flags the signal as pathological. But it could also be a ramp signal which climbs from 0 to 1 throughout the duration of a pathological beat pattern. This might be motivated by the fact that at the beginning of a heartbeat pattern it is less clear what type it is than it is at the end.

  Using RNNs seems natural for this task because this is a timeseries processing task after all. But RNNs are generally more difficult to train, and there is no easy way for transfer learning in RNNs (this is in fact a topic of current active research).

  Hadeaghi used a special kind of neural networks, *Echo State Networks*, which we will treat in the lecture only toward the end of the course (skip forward in the lecture notes if you start your project early, which would be a good idea…) Echo state networks are in some ways easier to handle than LSTM or other RNNs. An in-depth practical guide to using them cleverly is Lukoševičius (2012).

Report writing hints are given in the the document "SemesterProjectInstructions.pdf" (on the Nestor page "Hands-on miniprojects") for some guidance concerning the report writing.

Deliverables: as specified in the "SemesterProjectInstructions.pdf" (report, link to code or code).

References

Goldberger, A. L., Amaral, L. A. N., Glass, L., Hausdorff, J. M., Ivanov, P. Ch., Mark, R. G., Mietus, J. E., Moody, G.B., Peng, C-K., Stanley, H. E. (2000), *PhysioBank, PhysioToolkit, and PhysioNet: Components of a New Research Resource for Complex Physiologic Signals.* Circulation 101(23):e215-e220 [Circulation Electronic Pages; http://circ.ahajournals.org/content/101/23/e215.full

Hadaeghi, F. (2019), Reservoir Computing Models for Patient-Adaptable ECG Monitoring in Wearable Devices. Online report, https://arxiv.org/pdf/1907.09504

Lukoševičius, M. (2012): *A Practical Guide to Applying Echo State Networks.* In: G. Montavon, G. B. Orr, and K.-R. Müller (eds.) Neural Networks: Tricks of the Trade, 2nd ed. Springer LNCS 7700, pp 659-686. Online and code samples at http://minds.jacobs-university.de/pubs/ .

Moody, G.B., Mark, R. G. (2001), *The impact of the MIT-BIH Arrhythmia Database*. IEEE Eng in Med and Biol 20(3):45-50. http://ecg.mit.edu/george/publications/mitdb-embs-2001.pdf