Executive summary for EEH Challenge
Data Analysis As A Life Saver

SenticsBrain

November 2022

1 Introduction

Our challenge was to implement a solution which would provide prevention at scale. Our main mission is to somehow prioritize patients who need of medical care urgently - not only providing the right treatment, but also provide the treatment in time, which is often a problem. We now have wearable - mainly watches, which can monitor some our health functions. There is a lot of data, but not many ways to analyze them. Using automated algorithms, we can make the work of doctors more effective - for example by automatically prioritizing which patients needs care more urgently than others. We could also point out some auto detected anomalies from such data for further examination.

Many patients also hesitate with hospital visits, which could lead to preventable deaths. By analyzing their data, we can personalize the care and catch things earlier than ever.

For PoC, we decided to work with ECG data obtained from wearable devices, mainly apple watch.

2 Our approach

2.1 Data preparation & format

We used Spark for data loading and standardization. As each measurement for each patient has different length, we needed a way to standardize those data. So from each ECG measurement, we took 500 samples of length 2000, this sampling helps with the amount of data we have for training as well as with standardization. From now on, every patient measurement is considered to have length of 2000. Each of those measurement is parsed from some original ECG measurement. As we understood it, most of the measurements came from apple watch. So in the end of standardization, we ended up with table with following columns: (Patient_id, slice), where slice is 2000 long subpart of some measurement.

2.2 Model

Given that we have standardized ECG data, we need to somehow obtain their mathematical representation. This representation is often called an embedding. Embedding is basically a point in some N-dimensional space. When we have that representation, we can measure the respective vectors distances, make clusters from them etc. Problem is to how to train the model such that it provides embeddings which also carry some information. For example, data points for one patient should be close to each other and whenever we have some abnormal measurement, this data point should be further away. The problem is that we don't have labeled data, so we don't quite know what is anomaly and what is not.

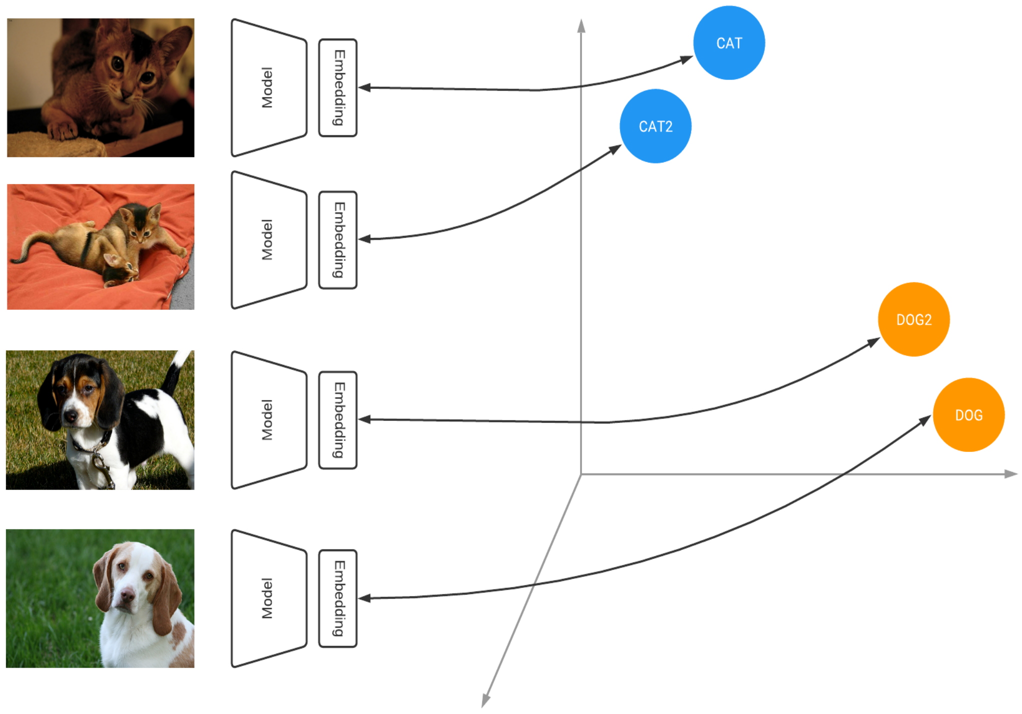

Figure 1: Similarity model example

When using this problem using autoencoders [2], there is something called reconstruction loss. What that means is the 'normal' measurement can be compressed to some latent space and then reconstructed to the original measurement without much loss of an information. But you still need labeled data for that.
Working with previous idea, we came up with an idea of using similarity models [1]. This is quite new concept. What it means is that the model is trained such that measurements for one patient should be close to each other and measurements for different patients should be further away. This is also called a triplet loss. [3] We then expect to have a compact cluster for each patient. If there is some measurement, which the model was not able to include to this compact cluster, we than consider it as an anomaly. Example of such similarity model can be seen in figure 1. Instead of cats or dogs, we have patient's ECG measurements - our 'cat' is some patient one a individual cat pictures are patients ECG measurements to make the analogy clearer.

2.3 Experimental

Figure 2 shows clusters from different patients in 2D space obtained using principal component analysis (original embeddings vector space has 64 dimensions).

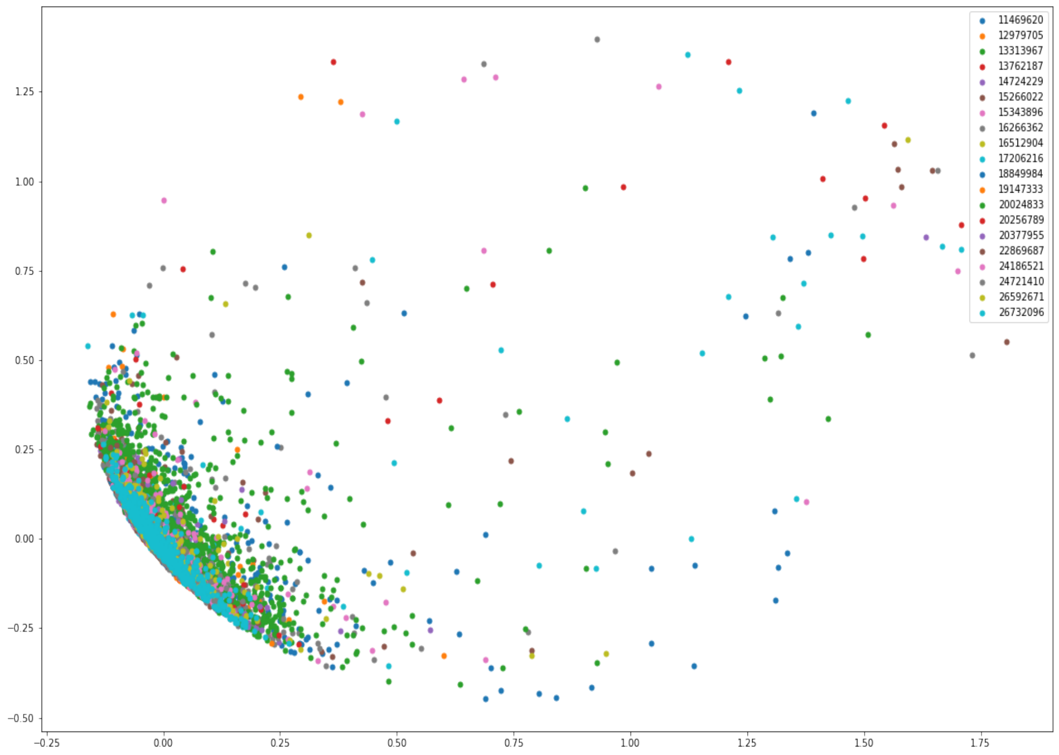

Figure 2: Different patients embeddings obtained by PCA

From those embeddings, we can see that there is one big cluster, and some data points further away from them - those could be considered anomalies. We then evaluated every individual patients cluster. Some example is in figure 3.

For this patient, we than run our algorithm for outlier detection (it's a simple heuristics considering outliers by their distance to the cluster centroid). Along with that, we plot some 'normal' measurements = measurements closer to the centroid. Result of that can be seen in figure 4. We can see, that the outlier (the data point furthest away from the centroid) is clearly different from the normal measurements. Based on that, we can sent the alert and it's up to the expert to check the relevance of that alert.

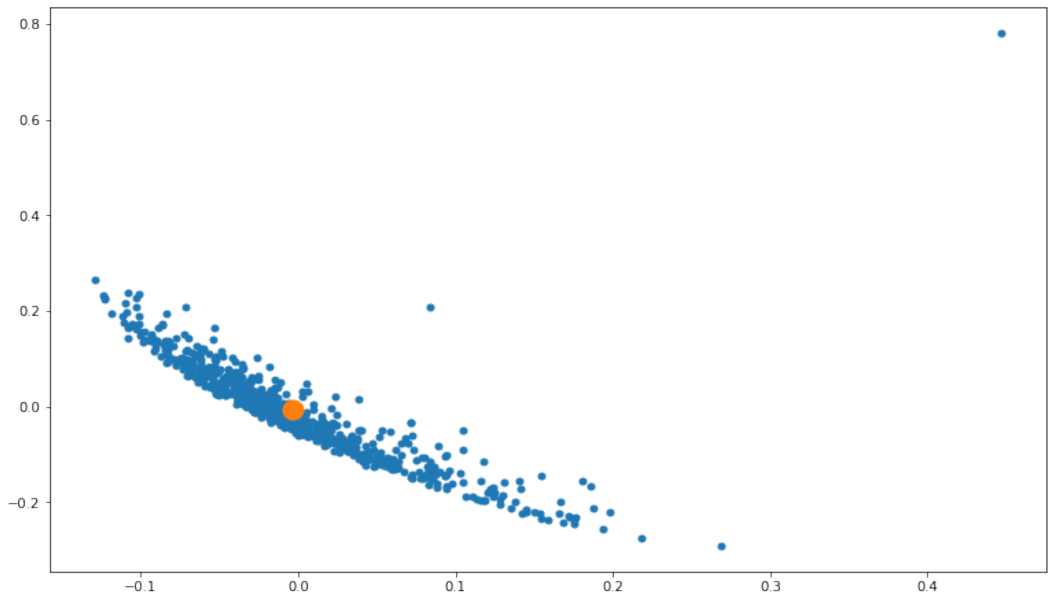

Figure 3: Individual patient cluster with cluster centroid

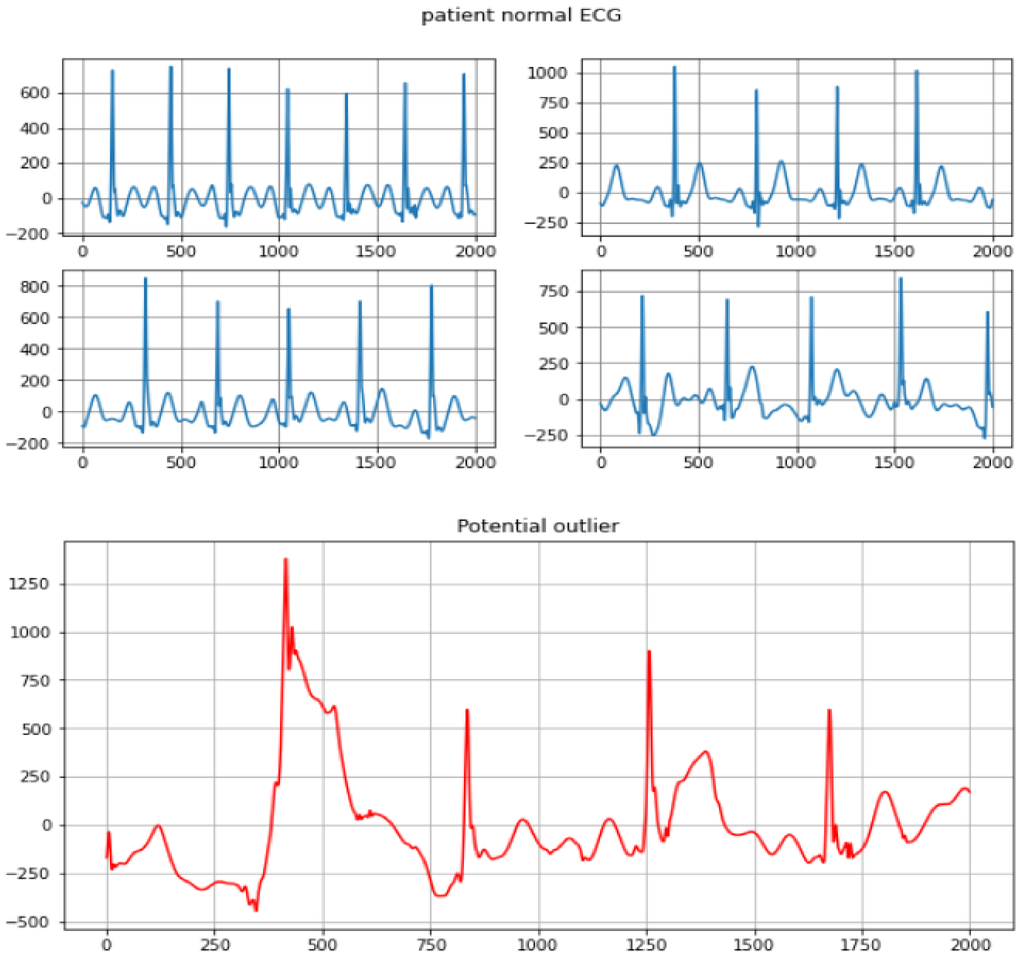

Figure 4: Patients normal ECG and potential outliers as detected by the model

Some more examples of normal ECG vs. outliers are shown below. Most of the time, no outlier is detected based on our metric - meaning that the patient has a normal measurements and no alerting is needed. However there are patients where the ECG data shows some abnormal activity for that individual patient and these are the cases the model can identify.

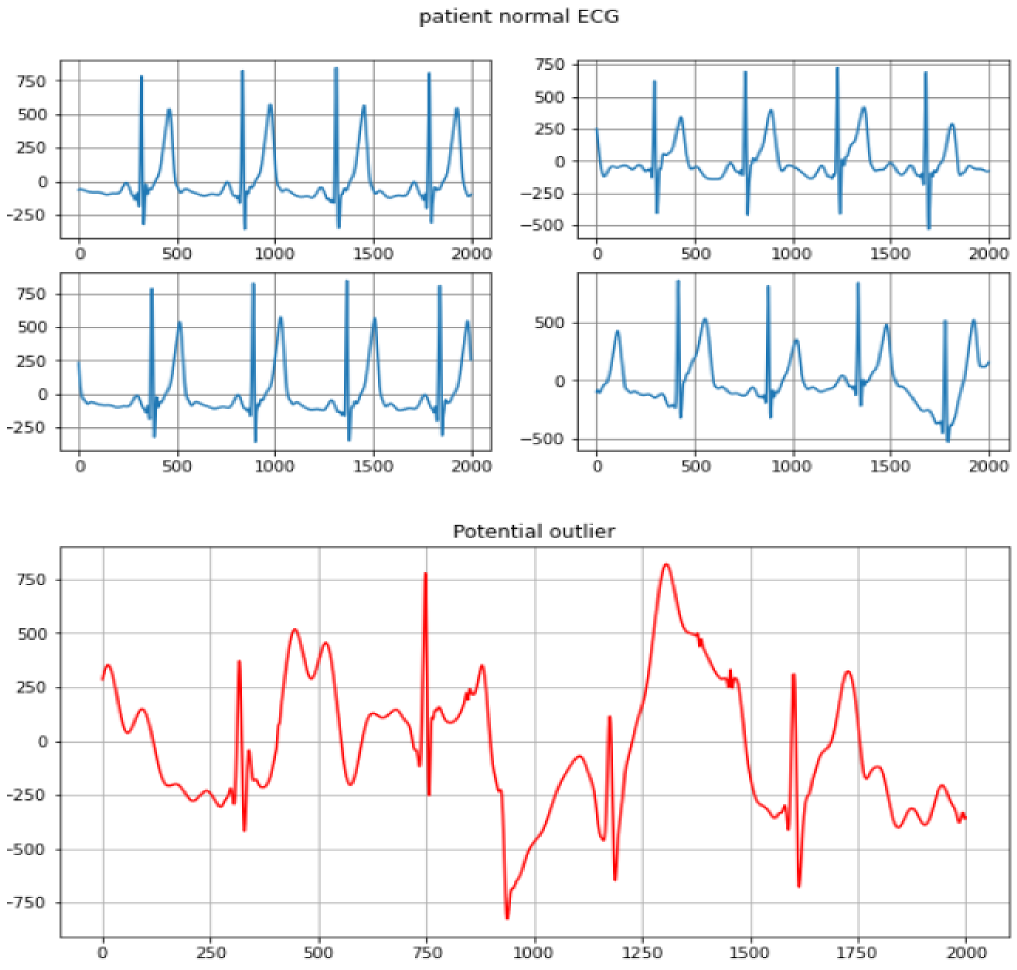

Figure 5: Patients normal ECG and potential outliers as detected by the model

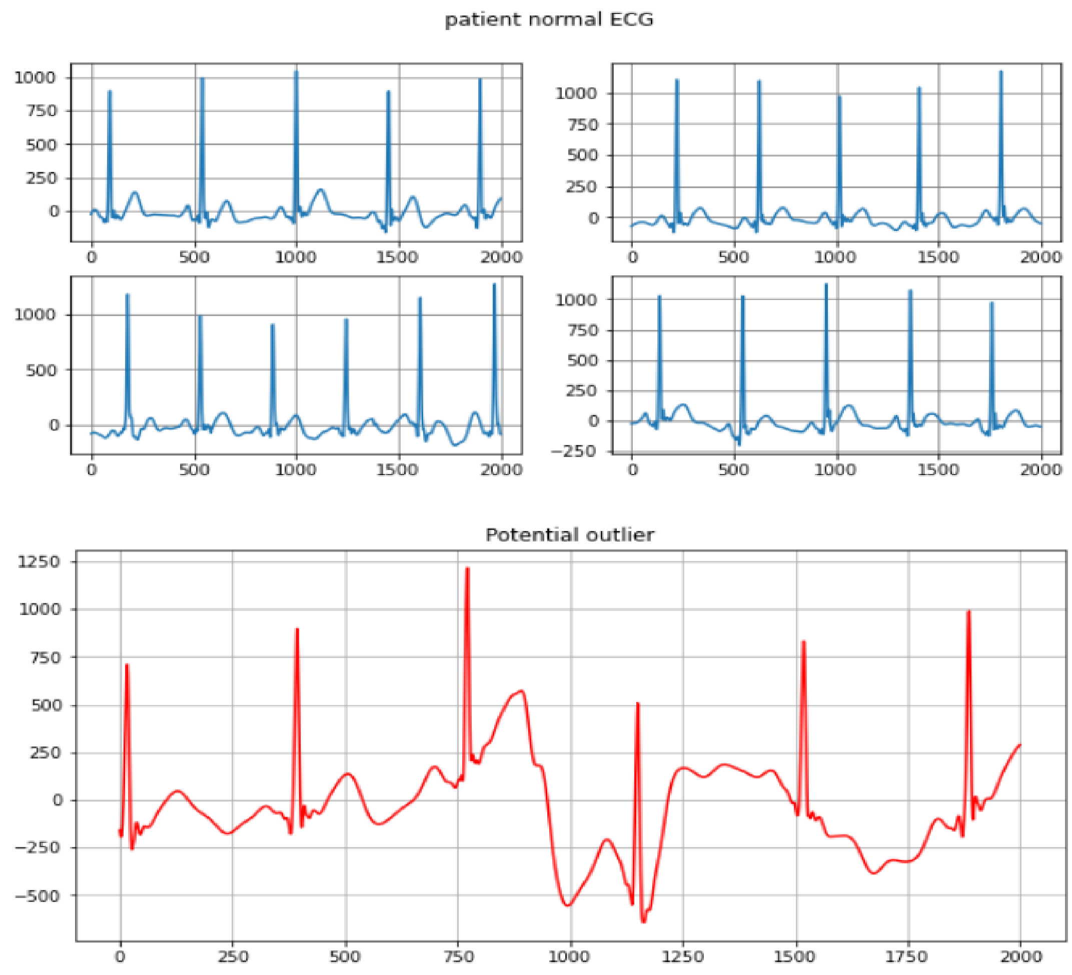

Figure 6: Patients normal ECG and potential outliers as detected by the model

2.4 Discussion

By those experiments, we checked the model potential, that is able to detect anomalies based on similarity models. Of course further examination is needed, but it seems like a very promising path. Something to consider for further testing is to check the models recall (if there are anomalies that were considered as normals). We cannot check this now as the data are unlabeled. But we certainly can say that the model has high precision (when classified as an anomaly, it is most of the time really different from the normal).

References

[1] https://blog.tensorflow.org/2021/09/introducing-tensorflow-similarity.html

[2] https://arxiv.org/abs/2003.05991

[3] https://paperswithcode.com/method/triplet-loss