2019

Transformation of Medical Diagnostics with Machine Learning by Considering the Example of Atrial Fibrillation Identification

Simone Schneider

Thorsten Gau

Annika Grosse

Holger Hennig,

Carmen Winter,

32ND BLED ECONFERENCE
HUMANIZING TECHNOLOGY FOR A SUSTAINABLE SOCIETY
JUNE 16 – 19, 2019, BLED, SLOVENIA, CONFERENCE PROCEEDINGS
A. Pucihar (ed. et al.)

Transformation of Medical Diagnostics with Machine Learning by Considering the Example of Atrial Fibrillation Identification

SIMONE SCHNEIDER, THORSTEN GAU, ANNIKA GROSSE,
HOLGER HENNIG & CARMEN WINTER

Abstract The paper addresses the problem of detecting one of the most common cardiac arrhythmias atrial fibrillation with artificial intelligence. The arrhythmia increases the risk of suffering from a stroke massively. Because of this, it is essential to detect atrial fibrillation early. As the arrhythmia occurs in short sequences, it is only possible to detect the disease in long-term measurements for example with electrocardiography. All common current detection techniques are calculating the R-R intervals with variations of the root mean square of successive differences. Because this approach is inflexible and expensive, a major hospital in Germany suggests the implementation of an artificial intelligence solution for atrial fibrillation detection. The aim of the paper is to study the feasibility of atrial fibrillation detection with artificial intelligence in the clinical setting of the hospital.

Keywords: • Atrial fibrilliation • Artificial intelligence • LSTM • Hospital • Machine learning • Clinicial setting • Prototype • Tensorflow • Keras • Python • IBM Cloud • Angular • TypeScript •

CORRESPONDENCE ADDRESS: Simone Schneider, IBM Deutschland GmbH, Ehningen, Germany, e-mail: simone.schneider@de.ibm.com. Thorsten Gau, IBM Deutschland GmbH, Ehningen, Germany, e-mail: tgau@de.ibm.com. Annika Grosse, IBM Deutschland GmbH, Ehningen, Germany, e-mail: annika.grosse@de.ibm.com. Holger Hennig, PhD, IBM Deutschland GmbH, Ehningen, Germany, e-mail: holger.hennig@ibm.com. Carmen Winter, PhD, Professor, Corporate State University Stuttgart, Computer Science, Stuttgart, Germany, e-mail: carmen.winter@dhbw-stuttgart.de.

DOI https://doi.org/10.18690/978-961-286-280-0.17 ISBN 978-961-286-280-0
Dostopno na: http://press.um.si

1 Introduction

The paper addresses the problem of detecting one of the most common cardiac arrhythmias atrial fibrillation with artificial intelligence. The arrhythmia increases the risk of suffering from a stroke massively. Because of this, it is essential to detect atrial fibrillation early. As the arrhythmia occurs in short sequences, it is only possible to detect the disease in long-term measurements for example with electrocardiography. All common current detection techniques (Franke-Gricksch, 2017; Gentile et al., 2017; Lee et al., 2012; McManus et al., 2013; Ricci, Morichelli, & Santini, 2009; Scully et al., 2011; Varma, Stambler & Chun, 2005) are calculating the R-R intervals of an electrocardiogram with variations of the root mean square of successive differences. The R-R interval describes the distance between two R peeks in a normal sinus rhythm (see figure 1).

This paper first discusses the methodological principles and secondly presents the conceptual foundations for the smart city conceptual model. Conclusion section summarizes and finalizes the paper.

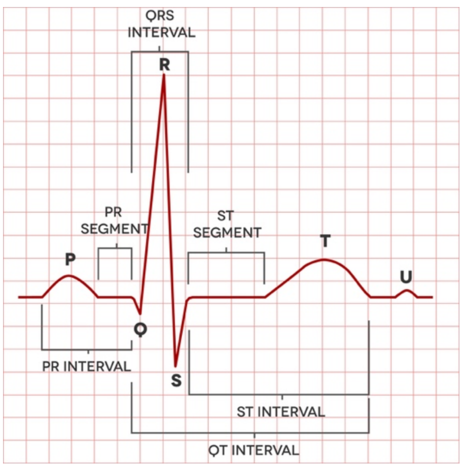

Figure 1: Normal Sinus Rythm ECG with R-R Interval Ilustration (US National Library of Medicine, 2013)

Compared to a normal sinus rhythm the distance between two R-R is unsteady (see figure 2). Also, the P wave in front of the R peek is completely missing (Birbaumer & Schmidt, 2006; Foster, 2007; Price, 2012).

Because this approach is inflexible and expensive, a major hospital in northern Germany suggests the implementation of an artificial intelligence solution for atrial fibrillation (AF) detection. The aim of the paper is to study the feasibility/possibility of atrial fibrillation detection with artificial intelligence in the clinical setting of the hospital. Therefore, the paper discusses the implementation of a prototype as well as a concept for an integration of the artificial intelligence solution in the hospital. The content of the paper is acquired during a bachelor thesis identically named. The following paragraphs are excerpts of the bachelor thesis slightly shortened or adjusted. At the end of every paragraph there is a reference to the corresponding chapter of the thesis.

Figure 2: Atrial Fibrillation ECG compared to Normal Sinus Rhythm ECG (Lifeline, 2015)

1.1 Requirements for the solution

The aim of the artificial intelligence atrial fibrillation detection is a full integration into the current processes in the hospital with no paper-based actions remaining. The process will start with the patient being monitored by a medical device (see figure 3). This is performed with an electrocardiography (ECG) monitor that is able to record long-term ECGs up to 24 hours. After the recording of the patient over the full period, the data is transferred to the AF prediction system. The prediction system evaluates the data with an artificial intelligence approach into three already existing categories: low risk for AF, high risk for AF and the presence of manifest AF. The report of the evaluation is cached to a tentative result which can be reviewed by a medical doctor on a user interface. If the prediction of the system is different from the doctor's opinion, the result is corrected by the doctor, and the prediction of the system is overwritten. After the doctor's review, the prediction is no longer a report. Instead, it is a result which has to be added to the patient's health record in the hospital information system (HIS). Different from the current solution in the hospital, the AF prediction system and the HIS are able to communicate directly which enables an automatic integration of the result. Apart from the integration in the HIS, it is also possible to use the verified diagnosis of the doctor for retraining the model. It ensures an improvement of the algorithm over time adjusted to the needs of the hospital (see thesis chapter 5.1).

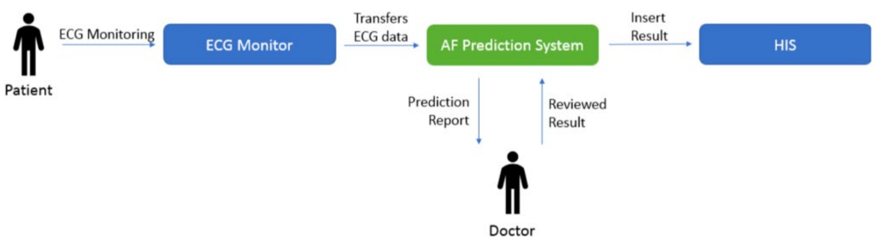

Figure 3: System Context Diagram

2 Implementation Prototype

The previous section specifies the system context and use cases for an overall solution integrated into the hospital environment. As there is currently no clinical data of the hospital available, the implementation of an overall solution is not possible. Therefore, the following sections illustrate the implementation of an AF identification prototype to demonstrate the capabilities and feasibility of an executable solution in production.

The deployment of the prototype mainly consists of three phases concerning *data processing*, *modeling* and *building the web application* as a user interface. Figure 4 is used to give an overview of all associated components and techniques. The components of the phases will be described in detail in the following section (see thesis chapter 5.2).

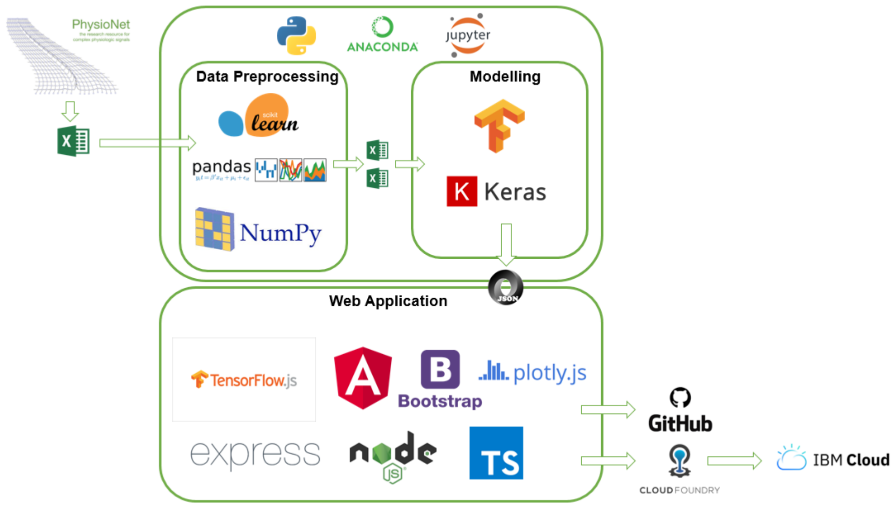

Figure 4: Architecture overview prototype

2.1 Data Preprocessing

Because of the lack of clinical data at the beginning of the project an alternative for the implementation of the prototype has to be found. According to many comparable studies (Limam & Precioso, 2017; PhysioNet, 2016; Shashikumar, Shah, Li, Clifford & Nemati, 2017; Sujadevi, Soman, & Vinayakumar, 2018; Yuan, Yan, Zhou, Bai, & Wang, 2016) PhysioNet provides an open source collection of physiologic signals including atrial fibrillation samples. Therefore, the MIT-BIH AF database as well as the MIT-BIH normal sinus rhythm database from PhysioNet are used to replace the currently missing clinical data (see thesis chapter 5.1.2).

From eighteen up to ten-hour examples respectively one hour is exported as a CSV file. One CSV file has three columns regarding a time stamp and two ECG measurements in millivolts which result in a different positioning of the ECG sensors. Every AF CSV file has 900.000 points of measurement in an interval of four milliseconds. As healthy samples, the MIT-BIH normal sinus rhythm database with eighteen samples is used and also exported as CSV files from one hour. The structure of the file has the same columns but differs in the intervals with a point of measurement only every eight milliseconds. It reduces the overall amount of monitored points to 450.000.

For preprocessing of the data, the open-source Python environment Anaconda in association with the web-based notebook environment Jupyter is used. For reading and manipulating of the CSV files the Python libraries pandas, NumPy and scikit-learn are in use. From both CSV files only the first ECG measurement column is extracted. As a result of the step of preprocessing a stacked array of both AF and healthy data with values assigned to milliseconds fitting to the demanded data format of the model is provided. Additionally, two classes are labeled concerning healthy with zero and AF with one.

The scikit-learn library also provides an interpolation function which is able to find values of new points. It is applied to gain an equal frequency of healthy and AF data (see thesis chapter 5.2.2). After the preprocessing step one labeled dataset is available. Because a training dataset and a test dataset is needed, the set is split randomly with the test dataset consisting of 20 percent of the original data. Then again both datasets are exported as CSV files for further usage.

2.2 Modeling

As Deep Learning algorithms outperform other techniques when large datasets are in use, and there is no need for feature extraction, it is entirely fitting for complex problems like sequence classification (Mahapatra, 2018). One common deep learning algorithm class are recurrent neural networks (RNN) which add the aspect of memory to algorithms. With atrial fibrillation sometimes occurring only in a few seconds of a 24-hour record and the frequent detection refers to the R-R intervals, it is essential that the algorithm is able to remind the previous sequences of the ECG for comparison. Because traditional RNNs are only able to remind a few steps back, in this use case the long short-term memory (LSTM) algorithm which is able to keep information for extended periods is used (see thesis chapter 5.2.3). The algorithm decides between essential or less essential information with selectively remembering or forgetting, which enables it to have a long-term memory. With this, it keeps the essential aspects of the time steps that need to remember this information (Olah, 2015; Kanber, 2018; Srivastava, 2017).

Regarding the LSTM the study of Sujadevi, Soman & Vinayakumar (2018) is also using the LSTM algorithm to detect atrial fibrillation in PhysioNet sample data. As the from-scratch implementation of an LSTM is out of the scope, we are using the TensorFlow Deep Learning framework Keras for the implementation of the LSTM as Sujadevi, Soman & Vinayakumar (2018) do as well. With this it is possible to prove the reliability of their accuracy of 100 percent. Similar to the preprocessing of the data, the model is implemented in a Jupyter notebook. It is a sequential model with a linear stack of layers with an LSTM layer that accepts an input dimension of 15000 corresponding to the dimension of our training dataset. An additional dropout helps to prevent overfitting by dropping fractions of the input. The output layer is a dense layer. As an activation function, the sigmoid function is applied. After the network architecture of the Keras paradigm the model needs a fitting which determines the test and training set as well as the epoch the model has to run through. In this case, the model runs through 20 epochs. As a loss function a binary cross entropy is used. After fitting and compiling the model is trained with the training dataset prepared during the data preprocessing. With the second dataset on which the model has not been trained, the model is tested. The last step of modeling is saving the trained model with the generated weights for further usage in a JSON format needed for TensorFlow.js (see thesis chapter 5.2.4).

2.3 Web application

For visualization a web application written in *TypeScript*, which is an extension of *JavaScript* and uses the *express.js* server-side framework with *Node.js* as a runtime container, is developed. The Front-End (see figure 5) is implemented in *Angular 6* and *Bootstrap*. From the previously generated datasets eight examples are chosen to be shown in the prototype. These are converted from the CSV files to a JSON file which consists of arrays of the samples. Plotly.js enables a dynamic zoomable plot of each sample. By clicking a button, a predict function is called. This function uses the *Tensorflow.js* package for making predictions with the model. The previously trained model can be accessed by the function as a JSON file which also has references to the weights that have been calculated during training. Based on the current classification scheme of the hospital, the prediction is divided into three types: low risk for AF, high risk for and the presence of manifest AF. The design of the front-end is plain to guarantee a fast understanding for the doctors. The web application is deployed in the IBM Cloud using a *Node.js runtime* and *Cloud Foundry* (see thesis chapter 5.2.5).

Figure 5: Front End prototype web application

2.4 Evaluation of the prototype

Even though these databases are also used in further studies (Limam & Precioso, 2017; Shashikumar, Shah, Li, Clifford & Nemati, 2017; Sujadevi, Soman, & Vinayakumar, 2018; Yuan, Yan, Zhou, Bai, & Wang, 2016), there are reluctance to believe in the quality of the data. First of all, the AF and normal sinus rhythm database differ in their frequency with AF having 250 Hz and Normal Sinus Rhythm having 125 Hz which was solved with interpolation for the healthy database. It cannot be guaranteed that the interpolation has found the matching points for the missing ones. Moreover, the positioning of the leads of the sensors seems to be different for AF. Plotting sequences of the AF database not only shows a different positioning of the leads, furthermore some of the plots look like the underlying data is broken. Because on this data the training is based, later the model could not feature the AF but rather the disturbances or the varying frequency. As the PhysioNet data is only for testing purposes of the feasibility of a prototype, it is well for first experiences regarding AF identification and testing the compilation of the components of a future application. In order to train the model for production clinical data where the origin is clearly known is absolutely and urgently needed.

The next step of the prototype is the deployment of a Machine Learning model regarding an LSTM. According to the quality measures for algorithms the LSTM model achieves the following values:

- Loss: 0,53
- Accuracy: 0,87
- Precision: 0,88
- Recall: 0,88
- F1 Score: 0,88

The accuracy of the model is well for a first try but needs further optimization also owing to the high loss of 0,53 which generally should have a value towards zero. Compared to other Deep Learning solutions in AF identification Shashikumar, Shah, Li, Clifford & Nemati (2017) have reached a bit higher accuracy value with 91,8 percent in using CNNs. Limam & Precioso (2017) on the other hand, describe an F1 score of 0,77 for their Calculational RNN solution. They highlight that this value could be increased with more training data which could also be useful for the model of the prototype. At last the current solution from Schaefer, Leussler, Rosin, Pittrow & Hepp (2014) and Gentile, et al (2017) used in the hospital achieves a recall of 0,99 which is a nearly perfect solution. With this, the model has to be improved to measure up to the state-of-the-art solution. With obtaining clinical data, the model will be retrained and further optimized to guarantee a comparable quality in AF detection with the Machine Learning solution.

3 Overall solution integration in a clinical setting

The following section illustrates the integration of the AF detection system in the hospital environment. Even though a production-ready solution is not yet implemented, an outlook of the AF prediction system integration system can be given. Therefore figure 6 depicts the operational overview of the system in a production-ready solution.

Similar to the current state of the art solution the process starts with the monitor measuring the patient's ECG. For access of the data, the monitor API (see 2.7) from the hospital environment is addressed. After the complete measuring, the data is transmitted to the IBM Cloud (figure 6, step 1). It is performable either over https or over messaging protocols like Message Queuing Telemetry Transport or with distributed streaming platforms like Apache Kafka. Apart from the data a patient identification number for a later allocation of the data to a patient, is added. With this, the datasets are anonymized, as the identification number is not traceable for external systems.

In the IBM Cloud, the data has to be transferred to a service conducting the preprocessing and evaluation (figure 6, step 2). The cognitive service is divided into Watson Studio where the data preprocessing is implemented, with Watson Machine Learning on top for training and evaluation of the model (figure 6, step 3). Watson Studio also supports the usage of TensorFlow and Keras in notebooks. Moreover, Watson Studio enables increasing capabilities with CPUs and GPUs for training models at scale. For later retraining of the model, there is also a continuous learning feature available.

After transmitting the data to the cognitive service, at first data preprocessing is executed to adjust the data format from the monitor for fitting to the required data format of the LSTM model. Similar to the prototype it can be implemented in a Jupyter Notebook in Watson Studio with the difference of now being able to utilize the processing power of the IBM Cloud instead of a local machine in the prototype. As a result, one dataset from every monitored patient in the required format for the model is received. After preprocessing every dataset is inserted separately into the model of the previously trained LSTM.
The model can be implemented in a notebook with Keras or TensorFlow. It has to be trained and optimized with labeled clinical data to ensure the origin of the training data. The accuracy of the new algorithm should be in range with the current solution. Apart from that, the recall of the model has to be high to ensure that a minimum of sick patients get overlooked. Also, a high precision value is required. Because if there are a lot of false positives, the doctors need to examine a lot of actual unnecessary datasets.

As a result, the prediction, as well as the dataset and the patient identification number, are transferred to the Node.js runtime. The results will be visualized in a web front-end similar to the prototype (figure 6, step 4). The predicted datasets are cached as long as they are not finally verified by a doctor and can be selected on the user interface. The front-end of the Node.js runtime can be displayed in a browser as well as mobile devices in the hospital environment (figure 6, step 5 + 6). The verification of the doctor in the front-end triggers the endorsement of the diagnosis in the hospital HIS. Along with this, the identification number is allocated to a patient in the HIS. The process is executed only in the hospital environment to ensure the privacy of the patient (figure 6, step 7). Apart from the endorsement in the HIS, it is possible to send the verification back to the cognitive service in the IBM Cloud. With this information, it is possible to retrain the model in the Watson Machine Learning service for continuous improvement of the model. According to this, cases that have less or even not been covered in the training dataset are also predictable over time.

The new overall solution integration offers an AF prediction system with no paper-based actions left. Another point concerning the verification of a doctor is that it also underlines that Artificial Intelligence has not the purpose of replacing humans. Instead, AI is just supporting them in their daily labor (see thesis chapter 5.3).

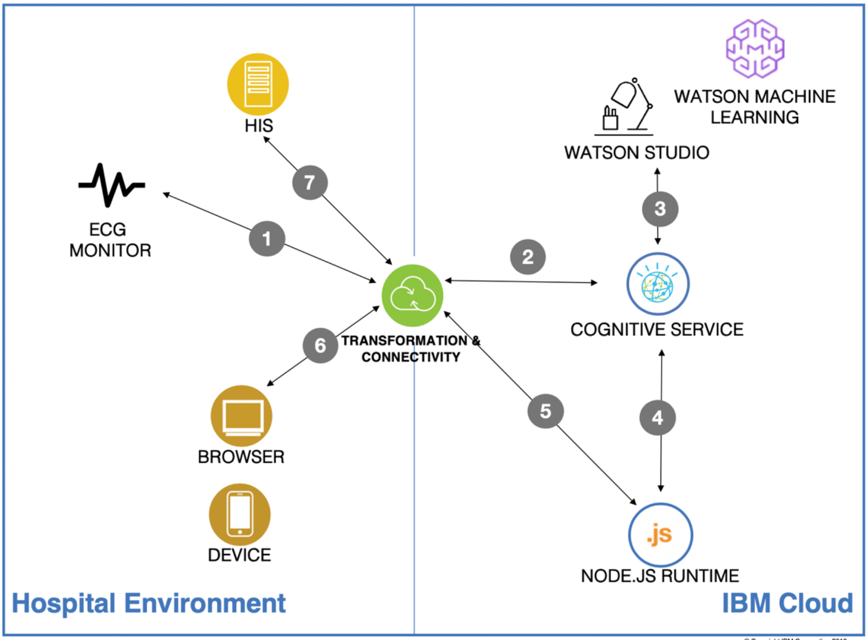

Figure 6: Operational Model Overall Solution

4 Conclusion and future work

For future work the most essential step is collecting a clinical dataset for AF and healthy ECGs to gain a clinical training dataset. According to this, the clinical data has to be evaluated by more than one doctor to ensure that the resulting training dataset is reliable. For labeling, there is the idea to provide a mobile app and to gamify the process to motivate the doctors to label as much data as possible. After gathering enough labeled data, the preprocessing and modeling for the in-production solution can be implemented, and the model can finally be trained. Against the prototype where no optimization is performed, the in-production model has to reach an accuracy which is in range with the current solution. Apart from that, it needs a high recall value to ensure that a minimum of diseased patients get overlooked. Another important aspect is the implementation of a retraining function in the Watson Machine Learning service.

Apart from the training of the model with clinical data and the integration of the AF prediction system discussed in the overall solution in 3, the system and the new process with all practical steps being digitized needs to be tested in a clinical setting before the current solution is detached. The testing should exhibit if doctors, employees and patients accept the Machine Learning approach, and if the process itself is interacting well with the hospital environment.
Another idea for a future implementation leads to the monitor being able to measure many other parameters apart from the ECG. Breathing, oxygen saturation, electroencephalography, temperature and blood pressure could also be monitored. Regarding this, the Machine Learning algorithm could be extended also to feature other parameters and not only the ECG. Considering more vital parameters of a patient could enable finding coherence between the parameters and define the AF detection more precisely.

The availability of measurements for other activities of the body like breathing and electroencephalography also enables letting the algorithm not only classify into AF or not. Rather more diseases like congestive heart failure could be detected during the evaluation of the ECG. That it is also possible to detect other diseases then AF is already proven in other studies like Choi, Schuetz, Stewart & Sun (2016). Technically the algorithm would then no longer feature on a binary classification but instead perform classification into different groups. The results of the classification can also be shown on the web-application which then summarizes the different findings in the ECG.

Having measurements of different diseases in the web application leads to the idea of a platform for evaluating different diseases with Machine Learning algorithms. The predictions would be central in one application similar to a dashboard where the doctors are able to verify the results and to fast analyze coherences between the diseases. Reutilizing one application or service for different use cases also saves costs in evaluation and time in administration concerning the direct integration into the HIS. With upcoming ideas for electronic health records, the integration of the verified diagnosis into such a record would enable a history of detected diseases from a patient. Especially when changing a doctor, information about the previous medical records will be helpful in considering previously detected diseases (see thesis chapter 6.4).

To sum up the prototype again illustrates the poosibility of an artificial intelligence AF detection and takes it further into a clinical setting. The new solution will not only digitize the process of AF detection in abandoning the paper-based steps but will also help the hospital to save a lot of monthly costs as the current solution is much more expensive then the artificial intelligence concept. According to this, more at-risk patients can get tested and with this possibly more strokes will be prevented. This is not only very beneficial for the patient him or herself but also in the interest of health insurance companies, because a stroke patient produces a huge amount of costs. To conclude, the idea of bringing artificial intelligence AF detection in a clinical setting should be further followed up, as the approach is definitely feasible in helping to find AF patients more easily and efficient.

Corresponding thesis

"Transformation of Medical Diagnostics with Machine Learning by Considering the Example of Atrial Fibrillation Identification", Schneider S., Hamburg, 2018

References

Birbaumer, N., & Schmidt, R. F. (2006). Das Elektrokardiogramm. In N. Birbaumer, & R. F. Schmidt, Biologische Psychologie (pp. 203-204). Springer Medizin Verlag Heidelberg.

Choi, E., Schuetz, A., Stewart, W. F., & Sun, J. (2016). Using recurrent neural network models for early detection of heart failure onset. Journal of American Medical Informatics Association, 24(2), 361–370.

Foster, D. B. (2007). Electrocardiographic Waveforms. In D. B. Foster, Twelve-Lead Electrocardiography (pp. 5 - 11). Springer-Verlag London.

Franke-Gricksch, N. (2017). RMSSD – der HRV-Wert für die Erholungsfähigkeit. HRC - Herzratenvariabilität. Retrieved 8-10-2018, from https://xn--hrv-herzratenvariabilitt- dcc.de/2017/09/rmssd-der-hrv-wert-fuer-die-erholungsfaehigkeit/

Gentile, C., Molon, G., Hepp, T., Valente, M., Thijs, V., & Adami, A. (2017). R-R variability to identify patients at high risk of atrial fibrillation in acute ischemic stroke. Third European Stroke Organisation Conference. Prague.

Kanber, B. (2018). Long Short-Term Memory. In B. Kanber, Hands-on Machine Learning with JavaScript. Packt Publishing.

Lee, J., Reyes, B. A., McManus, D. D., Maitas, O., & Chon, K. H. (2012). Atrial Fibrillation Detection Using an iPhone 4S. IEEE Transactions on Biomedical Engineering, 60(1), 203 - 206.

Lifeline (2015). Herzrhythmusstörungen - Vorhofflimmern (AF, AFib). lifeline - das Gesundheitsportal. Retrieved 8-15-2018, from https://www.lifeline.de/herz-gefaesse/herzkrankheiten/vorhofflimmern-af-afib-id35663.html

Limam, M., & Precioso. (2017). Atrial Fibrillation Detection and ECG Classification based on Convolutional Recurrent Neural Network. Université Cote d'Azur, 1 - 4.

Mahapatra, S. (2018). Why Deep Learning over Traditional Machine Learning? Towards Data Science. Retrieved 8-21-2018, from https://towardsdatascience.com/why-deep-learning- is-needed-over-traditional-machine-learning-1b6a99177063

McManus, D. D., Lee, J., Maitas, O., Esa, N., Pidikiti, R., Carlucci, A., . . . Chon, K. H. (2013). A novel application for the detection of an irregular pulse using an iPhone 4S in patients with atrial fibrillation. Heart Rhythm, 10(3), 315-319.

Olah, C. (2015). Understanding LSTM Networks. Github. Retrieved 8-2-2018,from http://colah.github.io/posts/2015-08-Understanding-LSTMs/

Physionet. (2016). PhysioNet: A Resource for Research and Education. Physionet. Retrieved 8-6-2018, from https://physionet.org/resource.shtml

Price, D. D. (2012). How to read an Electrocardiogram (ECG). Part One: Basic principles of the ECG. The normal ECG. southsudanmedicaljournal. Retrieved 7-30-2017, from http://www.southsudanmedicaljournal.com/archive/may-2010/how-to-read-an- electrocardiogram-ecg.-part-one-basic-principles-of-the-ecg.-the-normal-ecg.html

Ricci, R. P., Morichelli, L., & Santini, M. (2009). Remote control of implanted devices through Home MonitoringTM technology improves detection and clinical management of atrial fibrillation. EP Europace, 11(1), 54–61.

Schaefer, J. R., Leussler, D., Rosin, L., Pittrow, D., & Hepp, T. (2014). Improved Detection of Paroxysmal Atrial Fibrillation Utilizing a Software-Assisted Electrocardiogram Approach. PLOS one, 9(2).

Scully, C. G., Lee, J., Meyer, J., Gorbach, A. M., Granquist-Fraser, D., Mendelson, Y., & Chon, K. H. (2011). Physiological Parameter Monitoring from Optical Recordings With a Mobile Phone. IEEE Engineering in Medicine and Biology Society, 59(2), 303 - 306.

Shashikumar, S. P., Shah, A. J., Li, Q., Clifford, G. D., & Nemati, S. (2017). A Deep Learning Approach to Monitoring and Detecting Atrial Fibrillation using Werable Technology. IEEE EMBS International Conference on Biomedical & Health Informatics (BHI). Orlando: IEEE.

Srivastava, P. (2017). Essentials of Deep Learning: Introduction to Long Short Term Memory. Analyticsvidhya. Retrieved 8-3-2018, from https://www.analyticsvidhya.com/blog/2017/12/fundamentals-of-deep-learning- introduction-to-lstm/

Sujadevi, V., Soman, K., & Vinayakumar, R. (2018). Real-Time Detection of Atrial Fibrillation from Short Time Single Lead ECG Traces Using Recurrent Neural Networks. Springer International Publishing, 683, 212-221.

US National Library of Medicine (2013). Sick sinus syndrome. Genetics Home Reference. Retrieved 8-29-2018, from https://ghr.nlm.nih.gov/condition/sick-sinus-syndrome

Varma, N., Stambler, B., & Chun, S. (2005). Detection of Atrial Fibrillation by Implanted Devices with Wireless Data Transmission Capability. Pacing and clinical Electrophysiology of the World Societ of Arryhthmias, 28(1), 133 - 136.

Yuan, C., Yan, Y., Zhou, L., Bai, J., & Wang, L. (2016). Automated atrial fibrillation detection based on deep learning network. IEEE International Conference on Information and Automation (ICIA). Ningbo: IEEE.