Dynamic Re-admission Prediction of Heart Patients Using Scalable Models

Author:

Bob Bontje

6202969

Utrecht University

UMC Utrecht

Supervisors:

Prof. dr. Yannis Velegrakis
Dr. Matthieu Brinkhuis

External supervisors:

Prof. dr. Folkert Asselbergs
Drs. Arjan Sammani

Master Thesis: Business Informatics

Department of Information and Computing Sciences

Utrecht University

25 februari 2021

Abstract

Heart Failure is one of the most common reasons for hospitalization of people aged over 65, causing hospitals' economic burden. A large reason for the high costs of these patients is both the high readmission costs and rate. Predicting the readmission risk at discharge helps the hospital keep patients who are not ready yet to be discharged. Traditional readmission prediction methods do not use patients' full admission window to predict readmission because of model or data restrictions. These restrictions remove many useful datapoints of patients to predict their readmission risk. Using the full admission window does significantly increase the data, hence the need for a scalable model.

This thesis aims to adapt the LSTM model to adjust to new high dimensional data efficiently and proposes extending the LSTM with a Convolutional Neural Network, extracting prominent features, and reducing the data dimensions low dimensional sequences. The method ensures the LSTM can scale efficiently with new data and increase its predictive performance.

The research approach is based on the CRISP-DM method. The literature review creates an overview of current readmission prediction methods, resulting in an optimal model for this thesis to improve. The features and data selection are made based on literature and the medical supervisor's input, after which exploratory data analysis is performed to exclude irrelevant data records. The CNN-LSTM model is created with the help of the technical supervisors. Finally, the CNN-LSTM model is evaluated using 5-fold cross-validation measuring the average AUC, training time and number of epochs. The model is compared to baseline models and the standalone parts of the CNN-LSTM regarding their predictive performance and scalability.

The temporal models utilizing the full admission window show the best predictive performance compared to the baseline models, showing the power of including the temporal dimension in predicting readmission risk. The proposed CNN-LSTM model shows to be the most scalable temporal model with slightly lower predictive power than the standalone LSTM option, which is the least scalable model option.

Acknowledgements

At the start of this thesis, I knew that applying my knowledge of deep learning in a whole new sector was going to be challenging, but I am thrilled with choosing a project in the medical field, which was a first for me. I learned a lot about many interesting new topics and met great people along the way.

My supervisor Yannis Velegrakis is the first and I would like to thank him for his flexibility, guidance and keeping me focused throughout the process. Secondly, I would like to thank Arjan Sammani for guiding me with cardiology's medical data and terminology. Special thanks to Matthieu Brinkhuis and Folkert Asselbergs for introducing me to this topic and support throughout.

I would also like to thank Ayoub Bagheri and the data science team at the UMCU for helping me with technical questions and the willingness to collaborate. Finally, I would like to thank my friends and family for helping me stay positive and motivated during this thesis project in weird and challenging times.

Contents

1. Introduction

The interest in data analytics surrounding Electronic Health Records (EHR) has surged in recent years. Using EHR data to give patients personalized care has been one of the most promising data-driven healthcare directions (Cheng et al., 2016; Jin et al., 2018; Yang et al., 2018). The data collected in EHR's hold valuable information and applied in many AI-focused projects for different medical specializations. One of these applications is predicting the readmission of heart patients. Heart Failure (HF) is one of the most common reasons for hospitalization of people aged over 65, causing hospitals' economic burden (Amarasingham et al., 2010; Yang et al., 2018). It is also expected to see an increase in patients because of the population's ageing, currently counting about 26 million adult HF patients worldwide (Ponikowski et al., 2014).

A large reason for the high costs of these patients is both the high readmission costs and rate. In the United States, an average one-year Medicare spending is $35465 for non-readmitted patients. Readmitted patients cost an average of $56856, which is over 60% higher (Zheng et al., 2019). In addition to these costs, a study shows that 83% of their experiment group is hospitalized at least once, and 43% at least four times (Dunlay et al., 2009). HF patients also have a high mortality rate, with 17 - 45% of patients dying within one year of admission and the majority dying within five years (Ponikowski et al., 2014; Savarese & Lund, 2016). All these factors explain why the readmission of heart patients is a significant burden for hospitals. It has shown that hospital admission is a vital prognostic factor for an increased rate of mortality. The more ill a patient is, the more often the patient will be hospitalized because of the illness (Ponikowski et al., 2014). Therefore, it is crucial to predict patients' readmission rate and provide them with the necessary care. Patients also gradually show factors of HF days or weeks before an actual HF emerges (Ponikowski et al., 2016). These gradual indicating factors are stored in a temporal sequence in EHR. They can provide the necessary data and input for a Long Short Term Memory (LSTM) model to predict the readmission rate.

Deep Learning (DL) models are a growing support tool in healthcare. The performance for predictive analysis on time series data gives domain experts insight into a specific application for patient care (Choi et al., 2016; Pham et al., 2017; Razavian et al., 2016). Current prediction models in healthcare primarily use traditional statistical approaches such as linear regression or Cox Regression. These models show worse results than DL for readmission prediction and do not grasp all information from the EHR data because of the data's simplification (Brüngger & Blozik, 2019; F. Wang et al., 2014). Recently, more research has been executed using DL, such as Recurrent Neural Networks (RNN). These models can use the temporal dimension in EHR data, which allows them to adapt to patients' dynamically changing variables. An addition to RNN's is the LSTM. This model expands on RNN by making it possible to "remember" datapoints longer during the analysis, which combats the vanishing gradient problem of RNN (Gers et al., 1999). This improvement made it a valuable model for EHR data analysis.

A current focus in DL is to create a model that can adapt to the data over time (Faghri et al., 2017; Mayer & Jacobsen, 2020). DL models such as LSTM are traditionally trained on a dataset once and then used in production, but with constantly changing data it is possible that these trained models can get out of date. Therefore more DL models are being created with a focus on scalability (Wen, 2019). The result would be a trained, continuously model learning from new data to give a more accurate prediction. Training an LSTM model on new data is a slow process due to the model's depth (Wen, 2019). This thesis focuses on adapting the current baseline LSTM model to adapt efficiently to new high-dimensional data, advancing the healthcare domain because EHR data keeps growing with each patient's treatment. So, scalable LSTM models can provide new predictions based on new data in a dynamic setting.

1.1 Motivation

Trying to predict readmission risk has been a long-standing problem. There is no study with a CNN-LSTM model predicting readmission based on EHR data, even though this model can handle high dimensional temporal data. Older studies modify the data to a simplified version making the data applicable for their analysis (Felker et al., 2004; Keenan et al., 2008; Zolfaghar et al., 2013). This method leads to information loss of the clinical data points in EHR data, therefore limiting their studies' impact. This thesis intends to solve this problem with a CNN-LSTM model and researches its applicability for high-dimensional many to one time series classification. Currently, less is known about the use of the model to high dimensional time series data. Opening a new technique for many domains, with this thesis potentially contributing to patients' health care. Additionally, the impact of dynamic predicting is researched in this thesis. Finding out if continuously predicting on most recent patient data is beneficial compared to traditional learning.

1.2 Objective

Using the design science template of (Wieringa, 2014), the objective of this thesis is:

- To improve the prediction of readmission risk of heart failure patients, which is achieved using the novel data structure of detailed admission windows.
- By creating a CNN-LSTM model in a dynamic setting to handle high dimensional time series data continuously updating over time. The created CNN-LSTM model shows to benefit from the best characteristics of the standalone models.
- That satisfies predictive performance and scalability criteria, which is shown in the results making the CNN-LSTM the most scalable model with only a small reduction in predictive performance.
- So that doctors can predict readmission risk more accurate, potentially contributing to society by improving care for heart failure patients. The results show the CNN-LSTM model to outperform baseline models making the detailed admission windows combined with the CNN-LSTM a more accurate and scalable option than baseline methods.

1.3 Research questions

The following main question is created to accomplish the objective state above:

> *"How can the readmission risk of a patient with heart failure be predicted dynamically, using a scalable Long Short Term Memory model based on Electronic Health Records?"*

The main question is decomposed into three different sub-questions stated below.

1. *What makes a machine learning model scalable?*
   - *1.1. How can the LSTM model be adapted to perform in a scalable setting?*

     To fully understand what makes a model scalable, an analysis is performed on scalability, the achievability, and the importance thereof. The knowledge obtained is used to adapt the LSTM model's shortcomings to make the model work in a scalable setting, answering question 1.1.

2. *What are the current methods of predicting the readmission risk of patients?*
   By reviewing the current literature surrounding readmission prediction, an overview is created to analyze the current studies. This overview confirms our proposed model's novelty and is used to extract the most promising features in deliberation with the doctors at the UCMU.

3. *How does the adapted LSTM model perform compared to other (baseline) models concerning scalability and predictive performance?*

The adapted LSTM model is validated in an experiment using k-Fold cross-validation, comparing baseline methods to the new model. This validation results in an understanding of how the model improved regarding its predictive performance and scalability. The metrics used to analyze these two factors are the AUC score for predictive performance. The combination of training time and the number of epochs is used to evaluate the models' scalability.

1.3.1 Contribution

Based on the objective and by answering the research questions, the thesis contributes to the scientific body of knowledge by creating a new machine learning model by extending the LSTM model with a CNN for scalability of high dimensional temporal sequence data. This adaption is compared to baseline methods and related machine learning approaches. The comparison is made on their predictive performance by measuring their AUC, the number of epochs and training times. The addition of a CNN model should reduce the total model's training time without losing significant predictive performance. Next to using a new model to predict readmission risk, a unique body of data is used from the UNRAVEL Research Data Platform (Sammani et al., 2019). This data body combines ten years of unique features from real-world HER, which is unique compared to previous research.

1.4 Outline of and justification for a chosen research method

During this research, the design science method is used as the overarching method. Design science is the design and investigation of artefacts in context. The artefacts of research are designed to interact with a problem context to improve an aspect in that context. The artefact in this research is the new scalable model in the context of readmission prediction. The design science method is part of the Engineering Cycle, consisting of five different steps: problem investigation, treatment design, treatment validation, treatment implementation, and implementation evaluation. In a design science project, research is restricted to the first four tasks of the engineering Cycle: problem investigation, treatment design, treatment validation, and treatment implementation (Wieringa, 2014).

The CRoss Industry Standard Process for Data Mining (CRISP-DM) framework is used in addition to the design cycle. CRISP-DM is a specific method created for data-focused research, and therefore better fits this thesis's context. The steps in CRISP-DM do overlap with the design cycle but provide in-depth steps for data-focused research compared to the design cycle. The blue squares in Figure 1 are the global description of the steps taken. These steps are generally executed in a specific order that allows for iterative progress throughout the thesis. The CRISP-DM framework and overlap with design science are explained below and consist of the following steps (Pete Chapman et al., 2000):

Problem investigation

1. *Business Understanding*: This step focuses on understanding the project's objectives and requirements from the different stakeholder's points of view. A project plan is created based on these objectives, and the business question is converted to a data mining problem definition. Within this step, the literature review is done. Using the snowball technique, relevant papers are selected and reviewed to form the context of the problem. Based on the literature and the medical supervisor's input, a problem statement is defined, solving the current literature gap. This step's output is the problem description, problem statement, and literature review and are explained in chapters 2 and 3.

2. *Data Understanding*: This step focusses on selecting and collecting the relevant data while also investigating the possible data problems valuable subsets. This step will analyze the raw EHR data of the UMCU and get familiar with the structure and content. A selection of patients and admission is made in collaboration with the medical supervisor, finding all unplanned heart failure admissions. Based on the problem statement and literature, a selection of features is constructed only to extract the relevant medical records. To complete the data understanding step, an export is made from the UNRAVEL data platform of

the patients selected. These patients are linked to their medical records, creating an initial dataset of the patients and their medical state. Explanatory data analysis is executed to determine the prediction classes' distribution, find missing values, outliers, and check for normality of the features. The results of this step are described in chapter 4.

Treatment design

3. *Data Preparation*: Here, all activities to construct the final dataset are executed. These activities include cleaning, deduplication, selection, and converting data. During this thesis, a clean dataset is created from the raw EHR data gathered from the databases of the UMCU. Instead of having medical records per patient, the medical records need to be linked to individual admissions. Each patient's admission is exported and transformed into an admission window. The medical records data is cleaned by removing empty values and outliers. The admission windows are then filled with medical records of the patient on the corresponding days. The features and rows are then structured into a temporal data set for the final models. Observations are correctly labelled, with their prediction targets either being readmitted <30 days or readmitted >30 days. This step's output is a temporal and non-temporal dataset to be used for the models in the experiment and is described in chapter 4.

4. *Modelling:* Modelling the data is done in two ways, as the modelling techniques in this thesis require two different data formats. The data is constructed in a temporal and a non-temporal dataset. The non-temporal dataset is used for the baseline model, while the temporal data set is used for the CNN-LSTM method and its standalone parts. The CNN-LSTM model is created with the help of the technical supervisors and the data science team of the UMCU. The optimal structure is found with the use of experiments while constantly changing the hyperparameters and optimizations techniques such as batch normalization, dropout, and regularization. The other models, compared to the CNN-LSTM, are also selected and constructed. The baseline models are the DT and MLP using the non-temporal dataset. The temporal models are the CNN and LSTM standalone as well as the CNN-LSTM combination.

Treatment validation:

5. *Evaluation*: The project now results in multiple models of seemingly high quality, and this step puts the models to the test. The models are applied to a set of tests regarding the business understanding step; after that, a decision will be made regarding optimal model choice. An experiment is set up to evaluate the models based on their scalability and predictive performance. The evaluation is done by using 5-fold Cross-validation (CV) on the datasets created in the data preparation phase, using the average measures over the five folds. The CNN-LSTM model is compared to baseline models and related models such as a DT, MLP, CNN, and LSTM. Evaluation of the predictive performance is done by analyzing the AUC measure. The scalability is measured by analyzing the training times and the total number of epochs used for training. The results are visualized and shared with the technical and medical supervisors to include their expertise in the evaluation.

Treatment Implementation:

6. *Deployment*: This step is not performed in this thesis, as is explained in chapter 1.4.1.

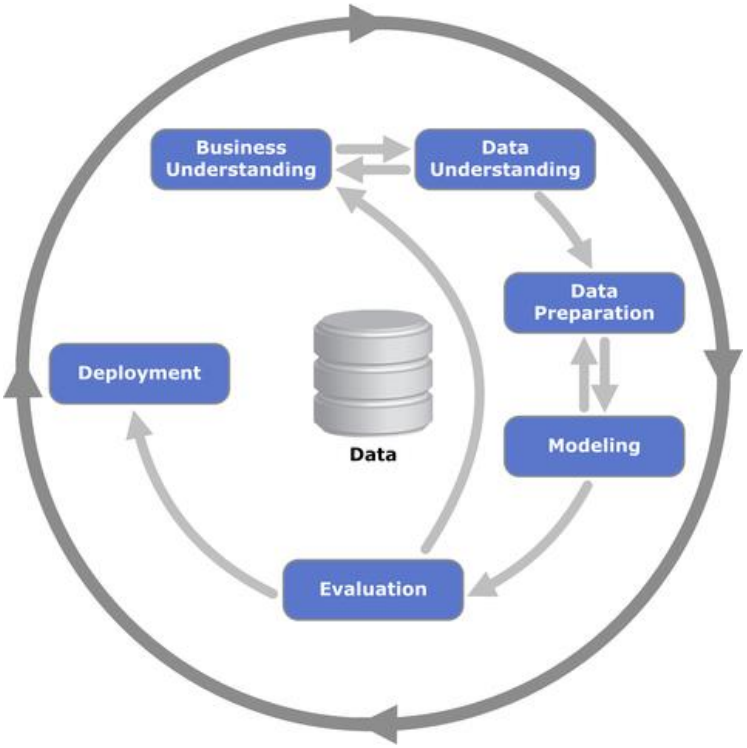


Figure 1 CRISP-DM (Pete Chapman et al., 2000)

1.4.1 Adaptation to this thesis.

The steps of CRISP-DM can be applied to the activities of this research. The first activity of performing a literature review will answer sub-question one and two and give a complete understanding of the business and data. An experiment is set up with this information, answering sub-question three by performing data preparation, modelling, and evaluation.

The final step of the CRISP-DM will not be performed in this research for the same reason treatment implementation is not covered. This is because the thesis will show the possible improvement of the CNN-LSTM model in an experimental setting. Further steps are necessary to decide if the model is optimal for deployment, such as external validations and clinical trials. These additional steps and deployment are outside the scope of this thesis.

1.5 Thesis outline

The structure of the thesis is as follows. Chapter 2 describes the domain preliminaries and provides the necessary background information, context and finally specifies the problem statement. Chapter 3 presents the literature review showing the relevance of the study and the gap in current research. Following is chapter 4, which describes the process steps taken to create the final dataset and build the prediction models' framework. The prediction models' results are explained in chapter 5, including the thesis's discussion and limitations. Finally, all sources are displayed in chapter 6 and all appendixes supporting the text in the thesis are shown in chapter 7.

2. Domain preliminaries

The next chapter includes the relevant literature surrounding the domain of heart patient readmission prediction. The chapter focusses on describing heart failure, EHR data, and gives an overview of popular readmission prediction techniques.

2.1 Heart failure description and current techniques for risk prediction

This chapter describes Heart Failure (HF) and creates an overview of the current readmission prediction techniques.

2.1.1 Definition of Heart Failure

The term HF is a clinical syndrome caused by a cardiac abnormality. It describes a situation when the heart is not able to circulate blood throughout the body. Different causes can be attributed to this phenomenon, for example, the cardiac muscle's inability to pump blood (i.e., muscle weakness or rhythm disturbance) or by obstructions in the outflow tract (aortic valve stenosis) or by systemic high blood pressures. There are many causes for HF, such as high blood pressure (systemic high blood pressure), infections (cardiac muscle inflammation), or an unhealthy lifestyle (myocardial infarction) (Ponikowski et al., 2014). Classical symptoms leading to heart failure are having fluids in the lungs (dyspnea), tiredness, edemas around the ankles and an increased heart rate.

HF can be separated as Acute Heart Failure (AHF) and Chronic Heart Failure (CHF). As the names indicate, the difference between the two is how quickly HF occurs. The European Society of Cardiology guidelines describe AHF to happen suddenly and CHF to develop slowly over time. When a patient has suffered from AHF, the patient is classified as stable when their symptoms and signs have remained unchanged for at least one month (Ponikowski et al., 2016). When a stable CHF patient is deteriorating, it is classified as decompensated. The decompensation may happen quickly or slowly but often leads to hospital admission, which is an event of considerable prognostic importance (Ponikowski et al., 2016).

Another option of differentiating HF patients is by measuring the ratio of blood ejected between the systole and diastole in the left ventricle after each beat: The Left Ventricular Ejection Fraction (LVEF). A difference is made between three categories concerning the LVEF. First, patients with a high LVEF percentage are typically considered to have Heart Failure with Preserved Ejection Fraction (HFpEF). HFpEF is also known as diastolic HF, which occurs when the heart cannot relax between each heartbeat. Secondly, patients with a low LVEF are classified as Heart Failure with Reduced Ejection Fraction (HFrEF) or Systolic HF, meaning the heart cannot pump enough blood through the body. A third group is between these classes having mid-range ejection fraction (HFmrEF). These patients are described as being in the grey area (Ponikowski et al., 2016). A patient is put in a class based on their LVEF score. An overview of each class and its corresponding LVEF score is given in Table 1. A split is done because of the difference in morbidity, mortality, and treatment options for each class.

This thesis focuses on readmission risk prediction for CHF patients with both diastolic and systolic HF options and are selected based on their diagnoses and corresponding ICD10 codes. CHF patients are selected because of the slow showing aspects of CHF are shown in EHR data and can be used as a prognostic factor for the prediction model (SAITO et al., 2020).

Table 1: LVEF score per type of Heart Failure.

LVEF	Type of HF
>= 50%	Preserved ejection fraction (HFpEF)
40 – 49%	Mid-range ejection fraction (HFmrEF)
< 40%	Reduced ejection fraction (HFrEF)

2.2 EHR data description and challenges

The following section describes the EHR data structure and gives an example of how to use this data. The data challenged are also described.

2.2.1 EHR data

The adoption rate of EHR data in hospitals has been growing in recent years (Adler-Milstein et al., 2015). The EHR data makes it possible to store patient data in a (semi)-structured form. The EHR data is then used throughout the hospital and used for patient recruitment, diagnoses, and analysis. EHR data is gathered from multiple different data sources, such as manually entered diagnoses from doctors and automated export from patient health systems (Sammani et al., 2019). These data sources can be separated by Electronic Medical Records (EMR) and Sensor Data (SD). Where EMR data is high-dimensional data consisting of structured and unstructured data such as lab results and textual description of patients, SD is collected from different medical devices such as ECG sensors and stream patient data in real-time. These sensors generate massive and various types of data streams (Lee et al., 2017).

The EHR data structure is set up to store data in a temporal sequence, resulting in all interactions with a patient being stored on a timeline. Figure 2 shows an example of a timeline. As the Figure shows, it records all medical events on a specific timestamp in sequential order (Sammani et al., 2019). This thesis uses data from the UNRAVEL research data platform. UNRAVEL is a database of selected heart patients at the UMCU. An overview of UNRAVEL and its sources of patient data is shown in Appendix 2.

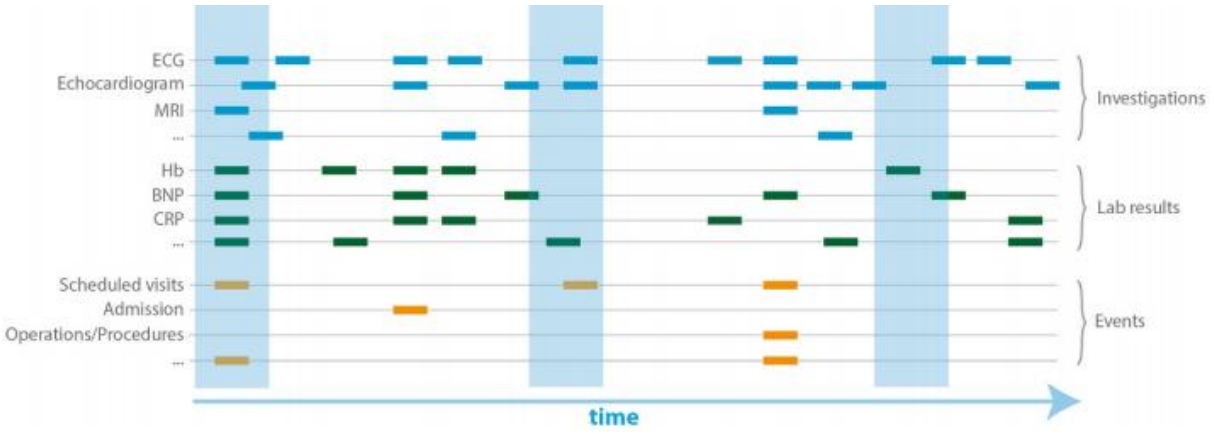


Figure 2: Temporal sequence of EHR data (Sammani et al., 2019)

2.2.2 Challenges

Working with EHR data has proven to be a complicated process because the extraction of a patient's individual characteristics is an ambiguous process with many data factors that hinder the ease of extraction. (Lee et al., 2017) describes the following challenges for using EHR data.

High dimensionality: As shown in Figure 2, for each timestamp, many possible events are registered. The codes used to identify these events are based on the ICD-10 standard (Organization, 1993). The standard has more than 14000 diagnosis codes. However, one patient is usually only diagnosed with a minuscule subset of codes, which creates a high dimensional vector per patient which is also sparse, resulting in a model with many parameters and increased complexity. Two techniques used to combat these challenges: feature selection and feature creation. Feature selection only picks out the most crucial feature; feature creation embeds features into a lower-dimensional space.

Sparsity: EHR data is also sensitive to sparsity due to high dimensionality. The high dimensionality creates sparsity on the Y-axis of the sequential data matrix. However, there is also a sparsity of data on the X-axis. The X-axis or time

axis has sparse data because of the interval of diagnoses that are executed. Diagnoses are made by doctors or other healthcare professionals that are not scheduled in a structured interval, and sometimes the diagnosis is not entered because of human error. Multiple techniques are available to impute new data, such as mean imputation or forward/ back-filling.

Irregularity: Two main factors appear in the concept of irregularity. The first is that EHR data is only recorded for the time that a patient is hospitalized. So, the data does not include readings from the time spent at home in between hospitalization. Second, even when a patient is hospitalized, not all the diagnosis events are done in the same interval, even for the same illness. The irregularity can be combated using overlapping features, time adjustment, or transforming the data so that all events fit a one-time window.

Selection bias: The bias problem comes primarily from patient sampling, which appears when the sampling rate is dependent on patients' states and dependent on doctors' judgment of the patients. One reason for this is because ill patients are sampled more than healthy patients.

Temporality: (Cheng et al., 2016) also states temporality as a challenge. The reason is that the constant of new data input created by new diagnoses made on patients results in an evolving patient record which then becomes more valuable than old records of patients. Using the most recent data is a significant challenge; a continuous update is needed for the data to analyze the patient's temporality accurately.

The proposed model will focus on solving the first three challenges by using a CNN's capabilities to reduce the dimensions by extracting the most relevant features into convolutional layers. These features are then forwarded to the LSTM to analyze their temporal linkage, and a final prediction is made. The challenge of temporality is handled using the new model in a dynamic setting, continuously retraining the most relevant data. The patients that are selected for this study all come from the UMCU hospital. The results might not be generalizable with all hospitals, because the UMCU is a tertiary hospital, which means that the UMCU primarily handles specialized patient cases. External validation is necessary to achieve generalizability, but this is outside the scope of this thesis.

The next section gives an overview of popular ML techniques, including the new model and explains their capability of handling the sequential EHR data.

2.3 Machine learning techniques for risk prediction

Many different analysis techniques are used to predict HF patients' readmission risk, as is shown in Table 2. These techniques show different results when applied to EHR data for readmission prediction. This thesis focusses on analytics approaches that can perform on sequential data. The main difference in the approaches is how the models input the EHR data. Which is either done by using one vector as input or by using a sequence of data. The following section describes the baseline methods used in healthcare and ML approaches that focus on sequential data.

2.3.1 Single vector input

As discussed in chapter 2.2, the specific characteristics of EHR are too complex for specific analysis methods. An aggregation of the data is done to create a compatible dataset for specific analysis methods. This aggregation is done by combining multiple feature vectors or sequences into one. An example of this is summing or averaging the vectors. A disadvantage of this method is that the individual vector's information is lost combined with their temporal relationship. A visual representation is shown in Figure 3.

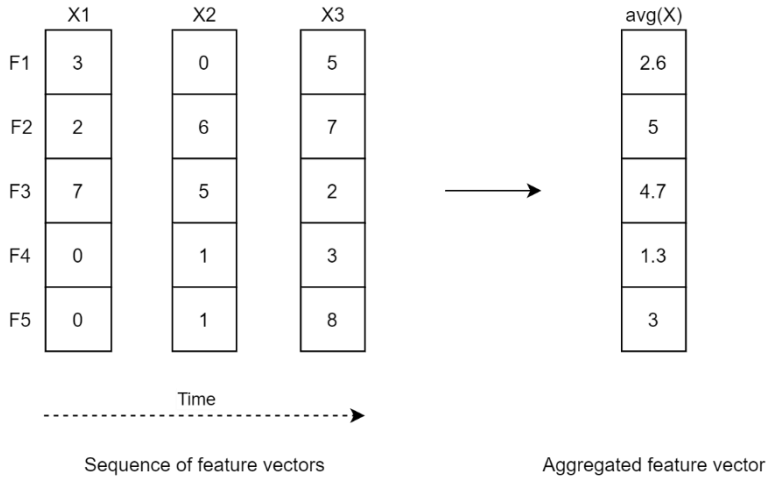


Figure 3: The left sequences are aggregated and stored as averages in the final feature vector.

Logistic Regression

Logistic regression (LR) is one of the most used traditional methods used for healthcare predictions, as shown in Table 2, where the early techniques predominantly use LR as their method. LR is also one of the most used methods for predicting a binary classification. The sigmoid output of LR makes it always be between 0 and 1. Based on the output, it predicts the object probability of corresponding to one of the classes. An output probability of <0.50 is assigned to class 0 and >0.50 to class 1. An LR input is the aggregated feature vector where each feature is combined with a weight resulting in a prediction score, as shown in Figure 4.

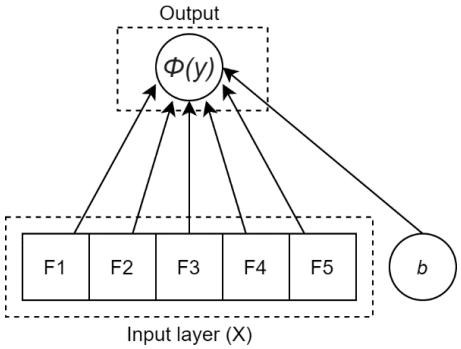


Figure 4: Logistic regression where (F) are the features and combined with the bias (b) resulting in a sigmoid activation function.

Cox Proportional Hazards Regression

Cox Proportional Hazards Regression (CHPR) is an extension of survival analysis. Survival analysis is used to model the time it takes for an event to happen. It is developed by medical researchers to measure a population's lifespan (Walters, 2009) and is a traditional prediction method in the medical field. Other fields have taken the CHPR method and use it for predicting the duration of the client to stay in a store or the time to failure of a machine. These examples have an assumed survival function that describes the predicted survival over time, where survival in this thesis is not being readmitted.

CHPR extends on survival analysis by using multiple variables for its analysis, making it like multiple regression analysis but with the dependent variable being the hazard function at a specific time. The model's input is the feature vector, just like LR, and each feature is connected to a coefficient that increases or decreases the log of the hazard ratio (Fox & Weisberg, 2011). These partial hazard ratios are combined with the baseline hazard of time and show the probability of survival or total hazard at a given time, as is shown in Figure 5:

$$\overbrace{h(t|x)}^{\text{hazard}} = \overbrace{b_0(t)}^{\text{baseline hazard}} \overbrace{\exp\left(\sum_{i=1}^{n} b_i(x_i)\right)}^{\text{partial hazard}}$$

Figure 5: Cox Proportional Hazards Regression formula (Wikipedia)

Each input feature's partial hazard ratio has a positive or negative influence on the total hazard depending on its score. When the partial hazard ratio is 1, it does not affect the probability of survival. A partial hazard ratio higher than 1 translates into a positive effect on the survival probability, which extends the event from happening. The opposite happens when the partial hazard ratio is lower than 1.

A notable flaw of CPHR is that the model can only identify risk factors that increase or decrease the probability of survival. The model does not calculate the actual baseline hazard, making it impossible to perform a personalized prediction by inputting a patient's values. This limitation was combatted by taking the risk factors and combining them with the marginal distribution of readmission and was first introduced by (Chin & Goldman, 1997).

Support Vector Machine

Support Vector Machines (SVM) is a robust algorithm used for regression and classification problems. SVM's are designed initially to be a binary classifier just as LR, but they can also predict multinomial problems. An SVM's goal is to find the optimal hyperplane for separating the classes by finding the maximum distance between the two data points of both classes. A plane is created in the gap with the maximum distance and the corresponding margin. These margins are set on each class's data point closest to the hyperplane and are called the support vectors. Real data sets usually do not have one gap in classes for the hyperplane to be placed. That is why a soft margin method was introduced by (Farhat, 1992). This method splits the data set as optimal as possible where one class fits the highest degree (Bellazzi & Zupan, 2008).

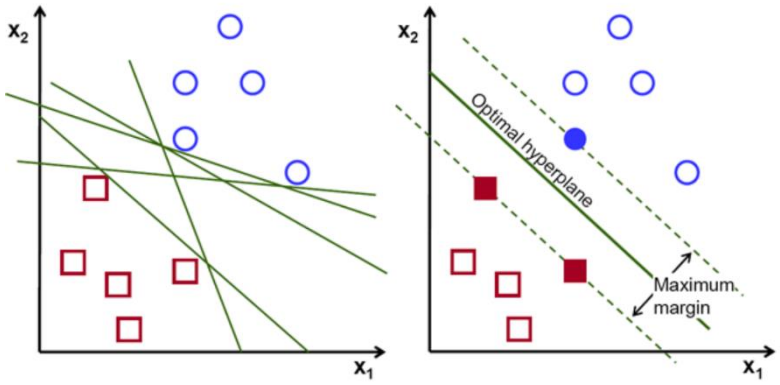


Figure 6: SVM finding the optimal hyperplane and extending the margins to the closest support vectors (Bellazzi & Zupan, 2008)

SVM has shown to be a great performer which requires low computational power to run. It has been increasingly used in medicine and bioinformatics. SVM shows to be a great option when focused on predictive performance compared to artificial neural networks, which usually require more computational power to run (Bellazzi & Zupan, 2008).

Multilayer Perceptron

Multilayer Perceptron is the first form of neural networks. They are a feedforward network consisting of one input layer, one or multiple hidden layers, and one output layer. These layers each have neurons that have a weighted link with each other. This web of weighted links is optimized during the training stage of this model. Each neuron takes the input from the feature vector or previous neuron and performs an activation function such as a sigmoid or tanh function, making all inputs summed and scaled for the next layer. The output layer would have a sigmoid function in classification problems. This way, an output can be assigned to a class just as in the LR.

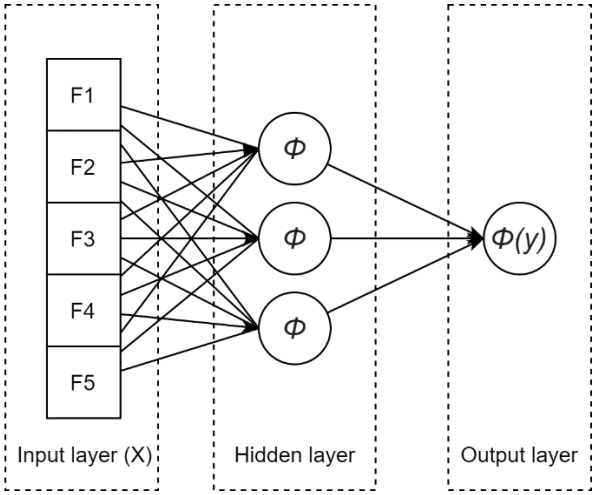


Figure 7: Multilayer perceptron where (x) is the feature vector, and (Φ) the activation function.

The main advantage of MLP compared to traditional methods is that the MLP can model highly complex and non-linear relations of features. It also adapts to these features automatically instead of the developer needing to adapt the model for specific relationships. It can learn the relationships from the training data and can apply this knowledge to future predictions.

Although the flexibility of MLP is a great feature, it does come at the price of computation power. MLP takes more processing power to run and can grow to an extensive network when multiple hidden layers are used. A trade-off is made between computational performance and predictive performance. The more hidden layers used usually increase the predictive performance in the cost of higher needed computational power. Another problem is that the model is prone to overfitting. This happens when the model has high predictive power for the trained data set but fails to predict on new data correctly. This overfitting problem is combatted by different methods like early stopping (Montavon et al., 2012), regularization (Demyanov, 2015), and dropout (Srivastava et al., 2014).

Recently the use of MLP in healthcare has been growing in (Shahid et al., 2019). With the help of recent studies combatting the disadvantages, it is a useful model to use. The specific use of readmission prediction has shown its promising application (Awan et al., 2019; Goudjerkan & Jayabalan, 2019).

2.3.2 Sequence input

The models mentioned above are not able to model the temporal characteristic of EHR data. However, as mentioned in chapter 2.1, this temporal characteristic holds valuable information. This chapter describes algorithms used in healthcare that includes the temporal dimension into their calculations.

Hidden Markov Model

The Hidden Markov Model (HMM) is a first and relatively simple method to model sequential data. HMM, are probabilistic models used to predict a sequence of hidden variables based on a set of observed variables. This prediction is only possible by assuming that the sequence is a Markov process, which means that the probability of an event happening is based on the state of a previous event. A classic example is predicting the weather (hidden variable) when looking at a person's clothes (observed variable). Observing a warm jacket predicts that it is probably a cold day. The links between the observed and hidden variables show the influential probability as is shown in Figure 8, where there are four observed variables to predict the three hidden variables.

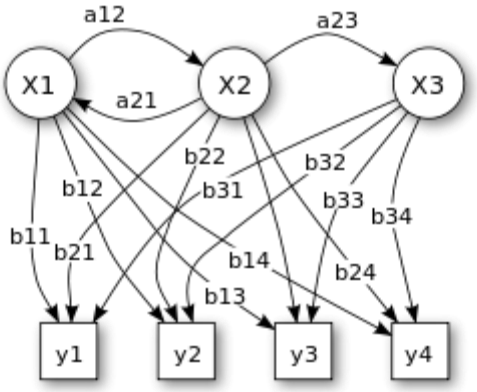


Figure 8: Hidden Markov Model (Wikipedia)

HMM has been used in healthcare for multiple different studies, but the model also shows flaws. The first being the assumption necessary that the sequence is a Markov process. Not all historical data in healthcare complies with this assumption, which makes the model obsolete. The second problem is that the complexity of the model rises considerably when more hidden states are introduced. However, HMM shows promising results in different medical studies. No previous studies surrounding readmission prediction are found using the HMM because the prediction of readmission is not based on a Markov process.

Convolutional Neural Network

Convolutional Neural Networks (CNN) are primarily used in image classification. The use of CNN for different domains has grown since the victory of AlexNet in the ImageNet competition (Krizhevsky et al., 2012). Its use of CNN spread to image recognition (Szegedy et al., 2014) and natural language processing tasks (Bahdanau et al., 2015), showing promising results and increasing the use of CNN for time series analysis.

CNN is primarily used as a feature extraction and dimensional reduction method. When taking a 2D image, it looks at all the pixels and groups individual sections into a smaller two-dimensional convolutional feature. A pooling technique is used to locate characteristics of the image and extract these from the convolutional layer, which results in a large image being shrunk to the essential characteristics, which can be used as input for another model. This compression reduces the computational effort for the final model.

The same principle applies to time series classification, but the convolution can act as a sliding filter of the time series. That would result in a one-dimensional convolution layer of time. So, when convolution is done with the length of 3 and the filter set to [1/3, 1/3, 1/3] and will apply a moving average of the time window with a length of 3. The formula for applying a convolution for a set timestamp is given in Figure 9.

$$C_t = f(\omega * X_{t-l/2:t+l/2} + b) \mid \forall\ t \in [1, T]$$

Figure 9: Result of a convolution (C) applied on a univariate time series (X) of length (T) with a filter (ω) of length (l), bias parameter (b) non-linear function (f) (Ismail Fawaz et al., 2019)

A pooling layer then filters the convolutions. The final discriminative layer takes the input from the pooling layers and creates a probability distribution of the dataset classes. This step is comparable to the activation functions of the MLP and results in the final prediction outcome. An overview of a CNN for time series analysis is given in Figure 10.

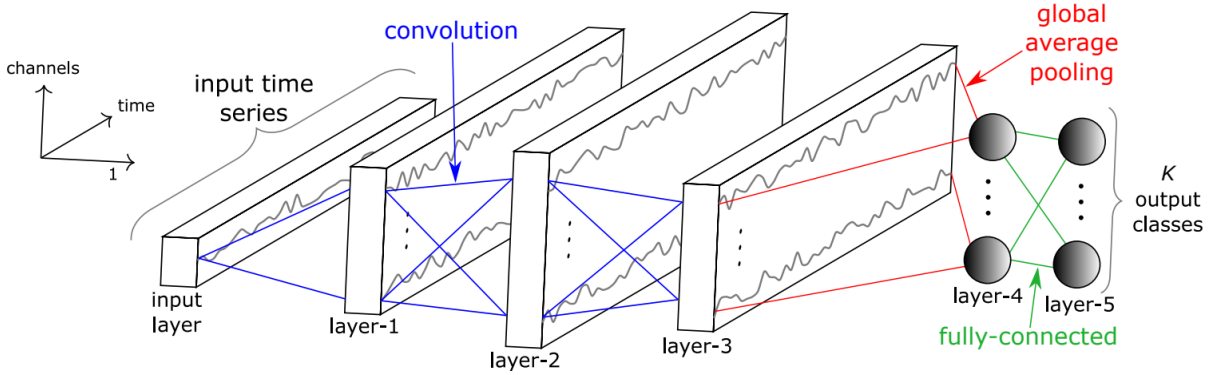


Figure 10: Convolutional neural network with 3 hidden layers (Ismail Fawaz et al., 2019)

CNN is used in healthcare for various reasons, such as activity recognition (Uddin & Hassan, 2019) and image classification (Qayyum et al., 2017). It has also shown to be effective at readmission prediction as multiple studies from Table 2 use CNN as a method. These studies use CNN image analysis (Cheng et al., 2016) or text mining (Liu et al., 2019) to predict readmission. So, CNN can lower high-dimensional data sets to create lighter computational models, including the temporal dimension. All these benefits make the CNN a reliable performing algorithm for readmission prediction.

Long Short Term Memory

Long Short Term Memory (LSTM) is an adaption of a Recurrent Neural Network (RNN). An RNN is a feed-forward neural network like MLP, but it has an internal memory to store previous predictions' output. The storage makes all input related to each other and handled with the same weight. So, the final output depends on the input of the previous steps. One problem with RNN's is the vanishing gradient problem, which occurs when the models are fed too much data and start to "forget" the first inputs it received. An adaption is made to combat forgetfulness, which is an LSTM model (Gers et al., 1999).

LSTM models include a forget gate which makes it possible to learn or forget efficiently. It also features a cell state which is the memory of the LSTM. The forget gate is connected to the cell state and calculates how much of the previous cell state it needs to forget. This way, the first input of an LSTM can survive throughout all the hidden layers, as shown in Figure 11.

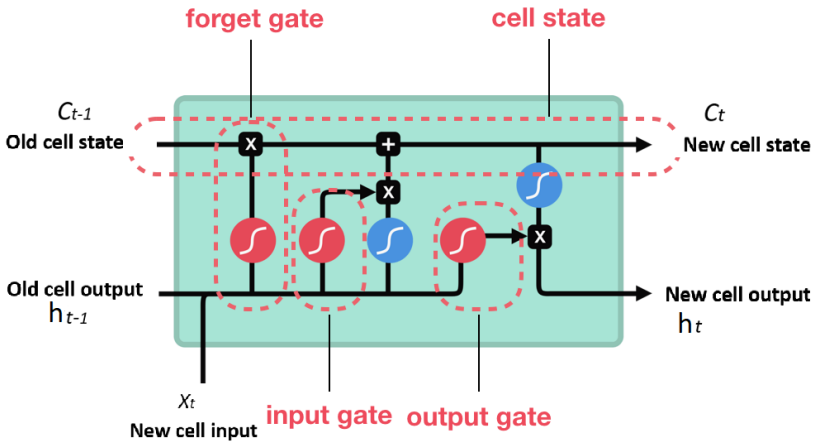


Figure 11: LSTM model where X is the new input and H the output for the specific timestamp. The blue circles are tahn functions, and the red are sigmoid functions.

LSTM in healthcare has also shown multiple applications (Ontology & Travel, 2019; Yu et al., 2019). These applications show LSTM to perform better than standard analysis using aggregated features. The applications of LSTM methods for readmission prediction are text mining and time series analysis, which is also used for the LSTM entry in Table 2.

Although LSTM shows excellent performance, it also comes with a high computational cost. The hidden layers of an LSTM are fully connected and need to memorize previous computations for each time step. Making it more computationally expensive, the more extensive the time sequences are, which makes it hard for an LSTM model to scale when more data is added to the model.

2.3.3 Evaluation

The mentioned models can give a classification prediction, but to evaluate these models' performance, a test is done to calculate specific metrics. These metrics are precision, recall, NPV, F1 and AUC score. Calculating these metrics is done based on the confusion matrix filled by the number of True/False Positives (TP/FP) and True/False Negatives (TN/FN). The accuracy is predominantly used to indicate the predictive performance, which is not sufficient as a comparable metric between algorithms. In Figure 12, a confusion matrix is shown. The predicted values are compared to the actual values of the test data set. When a model predicts a 1 and the test set's actual value is a 1, then it is a TP. The same applies when a 0 is predicted, and the actual value is also a 0. When the model predicts a 1, but the actual value is 0, then it is called a False positive, and an FN is predicted when the model predicts a 0 when the actual value is a 1.

		Actual values: 1	Actual values: 0
Predicted values	1	True Positive	False Positive
	0	False Negative	True Negative

Figure 12: Confusion matrix

As shown in section 3.2, all readmission prediction methods are compared using the Area Under the Curve (AUC) metric, also known as the C-statistic. The AUC proceeds from the confusion matrix and is calculated from the TP ratio against the FP ratio. These ratios are called sensibility and specificity. Combining the ratios results in a Receiver

Operating Characteristic (ROC) curve. The AUC is the area underneath the ROC and shows a score between 0.5 and 1. The closer the AUC is to the 1, the better it distinguishes the classes. The AUC is also favorable over accuracy when dealing with imbalanced data. The AUC is favorable because high accuracy can be achieved by only predicting the majority class. Only predicting the majority class would results in a low AUC score because the model did not distinguish the classes correctly. The complete list of evaluation metrics for classification is given in Figure 14.

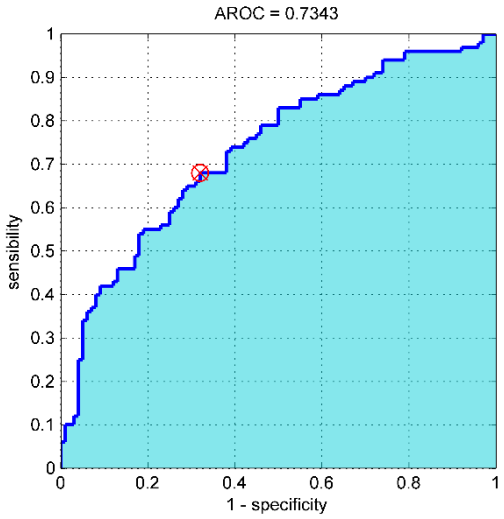


Figure 13: ROC and AUC score output

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \qquad Recall/Sensitivity = \frac{TP}{TP + FN}$$

$$Specificity = \frac{TN}{TN + FP} \qquad Precision = \frac{TP}{TP + FP}$$

$$f1\ score = 2 * \frac{Precision * Recall}{Precision + Recall} \qquad NPV = \frac{TN}{TN + FN}$$

Figure 14: Evaluation metrics

Problem statement

This thesis's goal is adapting the LSTM model to efficiently adjust to new high dimensional data and give a new prediction using the new data. The new data is needed because of the evolving state of a patient. When the patient goes through a new clinical test or gets new medication, it influences their health. The model needs these new investigations to update the prediction as the previous prediction becomes irrelevant due to the patient's care. This thesis proposes an adapted model by extending the LSTM with a Convolutional Neural Network, extracting prominent features, and reducing the EHR data dimensions into low dimensional sequences. The method ensures the LSTM can scale efficiently with new data and increase its predictive performance.

The technical problem can be stated as: with a combination of $X_1 \dots X_n$ investigations with a timestamp $t \in [0, T]$ is given with each investigation being data from medical appliances, lab results, or clinical measurements by staff, containing $i_1 \dots i_j$ results. Resulting in the data points of the sequence to be stated as: $X_{i,j}^t$ These data points are chronologically ordered into a sequence S and used to predict the patient's readmission risk within 30 days. So, given a sequence S with the length of T and height of X investigations where each investigation contains i results, can we predict $Y \in \{0,1\}$ where Y is the risk of readmission within 30 days after sequence S.

3. Literature review

The next chapter gives a comprehensive description of scalability in ML and shows its importance, and aims to answer sub-question 1 and 2. The chapter is set up by first focusing on the definition and characteristics of scalable ML. Afterwards, a semi-systematic analysis is done on the current state of readmission prediction.

3.1 Scalable machine learning

Research into large scale ML has grown recently, making it possible to solve more complex problems, such as text mining or voice recognition (Parker, 2012). Problems of that nature require higher complexity models or a high amount of data input to create an optimal result. Creating issues for current models that are used for empirical learning (static learning). This learning method involves a model to learn on a set of examples, a training set, and give a prediction based on this training set. When the training set's data increases or the models become more complex, it results in an increased difficulty making the computational requirements for training higher (Chilimbi et al., 2014).

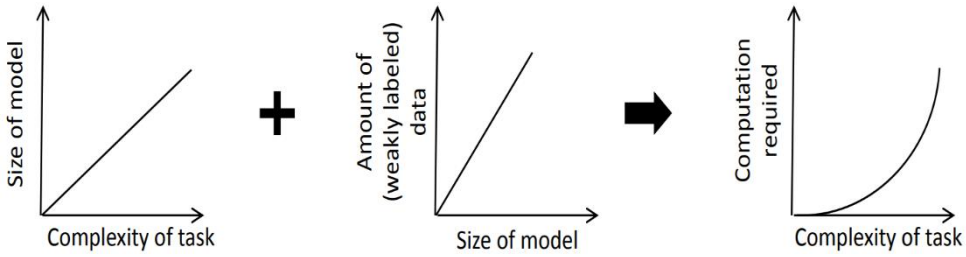


Figure 15: Model complexity formula (Chilimbi et al., 2014)

The complexity challenges are more prominent in deep learning than baseline models, where highly complex models need much computational power. The challenges of scalability are strongly connected with the characteristics and challenges of big data. The 4-V's model of big data describes the characteristics to be Volume (the growing data scale), Variety (various types of data), Velocity (rapid generation and timelines of data), and Value (extract significant value from dense data) (Gupta et al., 2016). Scalable ML focuses on using scalable models to discovering meaningful value out of fast-growing and changing data (Gupta et al., 2016). The following definition of scalable machine learning is given based on these characteristics**.** *"A scalable model is a model able to handle a large amount of data without consuming ever-growing amounts of computational resources within a practical time frame (Bekkerman et al., 2012)."* Achieving a scalable model is done by focusing on one or all the following parts of the pipeline: machine scaling, algorithm optimization, and data management, which are explained below.

Machine scaling

A solution to combat processing time is to add more processing power, as is mentioned in (Gomes et al., 2019; Mayer & Jacobsen, 2020). An increase in power is chosen (over other machine scaling methods) when a single machine is not capable of executing the computations. The addition of processing power is created by adding CPU or GPU clusters, creating a network of processing machines. Not all algorithms are optimized for distributed work; therefore, it is not always a solution to add more computing power. The main challenge is to split the model into partitions for the workers, creating model parallelism. Reinforcement learning is a common approach to split across multiple workers. The advantage of model parallelism is reducing the memory used per workers, but it comes with the disadvantage of heavy communication between the workers. Communication between workers makes it hard to effectively split the CNN-LSTM models as processes stall due to synchronization delays and communications overhead. Making it so that more machines does not necessarily mean a decrease in training time (Mayer & Jacobsen, 2020).

Algorithm optimization

The goal of algorithm optimization is to reduce the neural network's loss function in the fastest way possible by adapting the model's hyperparameters (Faghri et al., 2017). This way, the model can tweak its performance automatically during training. The optimization algorithm helps by solving specific optimization issues. These issues can be divided into three steps. The first step is starting the neural network to converge to the lowest loss. The second step is making the converging happen as fast as possible, and the third step is to ensure that the lowest loss is achieved (global minima) (Sun, 2019). Defining the amount of loss is done by measuring the difference between the predicted value against the actual value. There are multiple loss calculations such as Mean Square Error and Cross-Entropy Loss. Improving any of the three steps decreases the learning time, making it faster and adaptable to larger data sets.

Data management

The final part of scalability is data management. Increasing the model with growing amounts of data needs to be managed in a regulated process, creating a data stream. For a data-stream, the assumption is made that the data evolves and may be infinite, but processing the data needs to be done with finite resources. Literature shows two main approaches for data-stream classification problems: batch learning or instance learning (Read et al., 2012).

Batch learning or dynamic learning occurs when a model is fed new data. It uses its current train set and the new batch data to update the model, resulting in a static learning model retrained regularly using additional data. Batch learning handles the data characteristic of variety and velocity by increasing the training examples over time. The main disadvantages of batch learning are that a specific batch size needs to be specified. That old batches of data need to be deleted due to computational restraints, making room for new batches. The model can only retrain when a new batch is full of new instances, delaying the model's update.

Instance learning or online learning works by building an initial pre-trained model and retrain only on the new data. The model will update whenever a new instance of data is available. Instance learning is primarily used in systems that need to update their prediction in (near) real-time. Instance learning also handles the variety and velocity, but it also efficiently manages the volume of data by only training on the new data. Its real-time prediction application makes it also arguably better in extracting value when an instant prediction is necessary. The disadvantages of incremental learning are that there are far fewer algorithms available for classification. The model needs a far greater number of examples to learn a concept correctly.

One common problem of working with any of the two approaches is that data-streams are susceptible to concept drift. Concept drift occurs when the underlying data distribution changes over time. Concept drift reduces the accuracy of models as the data streams change (Tsymbal, 2004). This reduction happens when a hidden context influences the target concept, which means that a factor not included in the predictor variables influences the outcome. An example can be seasonality in weather or changing living conditions that influence patients' health (Tsymbal, 2004). A model needs to be able to detect these changes and adapt accordingly.

Concept drift is expected to happen in most data streams with batch and instance learning but detecting concept drift is important with instance learning since the model only learns on new data. With instance learning, the model is susceptible to drift because a new instance can hold a hidden context without linking it to previous or coming data. Concept drift does not occur in batch learning because the model can see the new context shift over a group of instances and adapt to it gradually (Read et al., 2012).

There are three methods for handling concept drift: instance weighting, instance selection, and ensemble learning. Instance weighting uses an algorithm to link new instances to the current concept with a corresponding weight, influencing that instance's importance. However, instance weighting has been seen to be prone to overfitting by (Klinkenberg, 2004), making it perform worse than the other methods.

Instance selection is shown to perform better by selecting instances relevant to the current concept. The most common technique for this is window selection, with a window moving along the timeline selecting new instances when available, forgetting the old instances. Originally these window sizes were fixed, but more research has been done on adaptive window sizes to increase detection of concept drift (Klinkenberg, 2004). With this method batch learning becomes less susceptible to drift because a new batch is also a new window selection.

Ensemble learning uses multiple models to weight or select new instances. The first system handles drift by creating a base model of concept descriptions, usually consisting of original features and combining these outcomes with new models training on new data. (Nick Street & Kim, 2001) Furthermore, (Haixun Wang et al., 2003) show a technique that divides the total data set into sequential chunks of the same fixed size, training the model on those chucks and ensemble the outcome to combat concept drift.

With this thesis focusing on solving the challenge of temporality, most focus is given to data management and not machine scaling or optimization. The EHR data will be analyzed using batch learning, where this thesis's experiment uses all admission available, without a window selection. Future weekly windows would be optimal to add because the system of the UMCU updates weekly, making each week a suitable batch of new learning data. The weekly batches would also combat concept drift.

Scalable LSTM

Reviewing the LSTM model for scalability shows a significant flaw regarding data management**.** LSTM's value of solving the vanishing gradient problem requires holding many previous states in memory. Holding previous states in memory does lead to a linear increase in memory as more time steps are passed. Studies are focusing on making the LSTM scalable by hardware acceleration (Yazdani et al., 2019), adapting the model to more efficiently talk to hardware components (Li et al., 2019), or by better optimization (Al-Shedivat et al., 2017). This thesis focusses on making the LSTM scalable by data management and controlling the increasing input.

So, reducing the input while keeping the necessary information is crucial for making the LSTM scalable. (Alonso et al., 2019) proposes to predict electrical power consumption with big data in a scalable fashion by training a shallow LSTM on a subset of the data and applying the model to other similar power consumption samples. Producing promising results is limited to only having univariate forecasting, and it shows little generalizability as the samples need to be like the train data. It also does not show the efficiency gain in training times, besides claiming the model to be X times faster related to the subsets created, as the model must only train on one subset.

The medical study (Maragatham & Devi, 2019) uses big data and the LSTM model to predict HF. They compare one hot encoding technique to their novel medical concept vectors. They conclude that one hot encoding creates too large of a dataset, significantly increasing the memory necessity of the LSTM model. Their new medical concept vectors use NLP to create one vector based on diagnoses to reduce the dataset dimensions significantly. Instead of taking 5500 seconds, it took 700 seconds of training time to complete classifying 35000 patients, making the model 7,8 times faster. This increase in speed indicates that the lower the memory usage of the LSTM, the more efficient it can predict, making it more scalable for big data.

Another method for reducing the dataset while still containing the relevant information are CNN's. These models have been primarily used in image time series problems and outperformed LSTM models because CNN can compress the 2d image into a smaller convolutional layer (Chen et al., 2019; Guo et al., 2018). CNN has also been shown to predict time series data. The CNN's are more efficient than LSTM's, but research shows that they cannot always outperform LSTM in predictive performance (Chen et al., 2019; Guo et al., 2018; Livieris et al., 2020; J. Wang et al., 2016). Combining the two methods makes the CNN reduces the input of the LSTM, making it more efficient and able to scale on big data. Adding the CNN would increase the time an epoch needs to go through the model, but this does not increase the overall training time as the total amount of epoch necessary is reduced, resulting in

the model (Tsironi et al., 2017) to be 2,7 times faster. Therefore, the combination of a CNN and LSTM is most favorable.

CNN is also known for learning the internal representation of time sequences, as is shown in (Ismail Fawaz et al., 2019; Kim & Cho, 2019; Livieris et al., 2020), dealing with the challenges of high dimensionality, sparse, and irregular data. The study (Livieris et al., 2020) uses a CNN to extract meaningful features from a univariate time series. It combines this with an LSTM model to identify long and short-term dependencies, predicting gold prices, making the combination perform better than using the LSTM alone. The study (Kim & Cho, 2019) used a CNN-LSTM structure to forecast household power consumption with multivariate time series data. The CNN made sure that the noise is removed, reducing the data for the LSTM, and creating an efficient and robust model able to train quickly on large amounts of data compared to the vanilla LSTM. The difference between this thesis and the previous two studies is that their data is more continuous than EHR data. So, an adaptation of the model is needed for EHR data.

Creating a hybrid model that leverages both the CNN and LSTM strengths would benefit efficiency and scalability. Reducing the input of the LSTM while keeping the most relevant features and making it possible to use the temporal relationship between features. Placing the CNN-LSTM model in a dynamic setting also deals with the final challenge of temporality, profiting from the continuous updates from new data.

3.2 The current state of readmission prediction

Many different methods are used to make a predictive model, with most of them showing average results. Most of the methods use baseline statistical concepts such as logistic regression or survival analysis. More studies have been done using DL models for readmission prediction, showing better results regarding their predictive performance, as shown in Table 2.

Multiple review studies by (Ouwerkerk et al., 2014; Ross et al., 2008; Tripoliti et al., 2017) created a detailed overview of the current state of readmission prediction. Although the studies provide a comprehensive review, few mentions of models use a neural network or deep learning. Presumably, because research using deep learning in health care has primarily been conducted in recent years. An overview of current studies is given in Table 2. Here, multiple studies are compared to their used method, prediction window, and performance. All performance is measured using the Area Under the Curve (AUC) score. An explanation of this measure is given in chapter 2.3.3.

Table 2: Overview of current literature of readmission prediction. Full model names can be found in the appendix.

#	Study	Model	Prediction window	Prediction outcome	Performance (AUC)	Data set
1	(Kang et al., 2016)	DT	60 days	HF	0.59	Questionnaire
2	(Philbin & DiSalvo, 1999)	LR	365 days	HF	0.60	Questionnaire
3	(Keenan et al., 2008)	LR	30 days	All-cause	0.60	High-level EHR
4	(Zolfaghar et al., 2013)	RF	30 days	HF	0.63	High-level EHR
5	(Allam et al., 2019)	RNN	30 days	All-cause	0.65	High-level EHR
6	(Felker et al., 2004)	LR	60 days	All-cause	0.69	Low-level EHR
7	(Roy et al., 2015)	HDC	30 days	HF	0.69	Low-level EHR
8	(Shah et al., 2015)	SVM	180 days	HF	0.70	Low-level EHR
9	(Yang et al., 2018)	DSRF	60 days	HF	0.71	High/low-level EHR
10	(Haishuai Wang et al., 2018)	CNN	60 days	HF	0.71	Low-level EHR
11	(Liu et al., 2019)	CNN	30 days	All-cause	0.72	Clinical notes text
12	(Hameed & Bukhari, 2020)	ANN	30 days	All-cause	0.75	High-level EHR
13	(Cheng et al., 2016)	CNN	180 days	HF	0.76	Images of admission sequence
14	(Ashfaq et al., 2019)	LSTM	30 days	HF	0.77	High/low level EHR
15	(Koulaouzidis et al., 2016)	NB	1 day	HF	0.82	Home sensors

Table 2 shows that the average AUC of baseline methods is generally lower than the AUC of deep learning methods. This difference can be explained by the simplification that occurs when using baseline methods instead of deep learning methods. Also, specific deep learning methods are specialized for performing on time series data. Making deep learning methods specialized for time series data the best method to use. The datasets used can be categorized as high-level EHR data, low-level EHR data, or secondary data. Where high-level EHR data uses aggregated features such as total count of lab tests, and low-level EHR are clinical values from a lab test. Secondary data are all datasets not using EHR data, such as question airs, clinical notes, image data, and home sensors.

Method analysis

Analyzing the methods of Table 2 show that the lowest-performing studies use a questionnaire or build a model on high-level EHR data. (Kang et al., 2016; Philbin & DiSalvo, 1999) created a questionnaire to predict readmission risk based on second-hand variables such as living conditions, insurance, and education level. Both studies created an own scoring method on the questionnaire, where the more point a patient score, the higher the readmission risk. Both studies also select patients who use Medicare, which means that more low-income patients are selected for the study, making it not generalizable to all patients. The use of second-hand variables should not be used to predict readmission. The study (Keenan et al., 2008) also focusses on Medicare patients. However, it uses a method to categorize patients on individual EHR values, such as stating if a patient has a high or low heart rate. This method improves on questionnaires as it comes closer to the actual clinical values of patients. It does state the model used to have a computational limitation as it cannot account for the within-patient correlation of multiple index heart failure hospitalizations per patient.

Other studies using high-level EHR data such as (Zolfaghar et al., 2013) and (Allam et al., 2019) improve on Keenan's study by using more intricate models to predict readmission risk. Another exciting part of this study is the integration of big data. As Zolfaghar notes, this has shown exciting features such as hospital rating to enrich the patient dataset. Although these data points' publicity is only available at the state or county-level and many data points are missing. To solve the missingness of the data, an average is taken to impute the data. Zolfaghar also notes that an RF can train high amounts of data efficiently and in parallel, but the method only shows that an RF can run parallel on multiple Mahout nodes and does not relate to improving predictive performance. The study's limitation is that only the current values of patients' EHR data are taken and not historical data points of patients. (Allam et al., 2019) use of an RNN mitigates this limitation and includes previous data points from patients, increasing the predictive results. Improving on the limitation of the Zolfaghar but still uses high-level administrative data instead of rich EHR data.

An increase in predictive performance is observed when the rich low-level EHR data is used as in the studies of (Felker et al., 2004; Roy et al., 2015; Shah et al., 2015). These studies share the same limitations as Zolfaghar as they do not include patients' historical data points. Felker's computational limitation is also shown as they only use five features for their simple LR model. Meanwhile, Shah is the study with the most features showing promising low-level EHR results. It only trains the model on a small dataset of 379 patients, this combined with the long prediction window, makes the result less generalizable. A more generalizable study is (Roy et al., 2015) this is the only study using a clustering technique to classify readmission risk in stages, differentiating between 3 stages of readmission risk. It also does this on 6378 patients making it one of the larger datasets in the analysis.

Almost none of the studies handle the temporality challenge that was introduced by (Cheng et al., 2016). The only study to use a form of dynamic predicting is (Yang et al., 2018). In this study, they use a moving window technique that updates its predictions over time. It shows that new data provide as well or better data points to predict readmission, resulting in a model saving computational memory and power and making dynamic prediction more efficient and effective than traditional methods using empirical learning. Using the sliding window technique also combats concept drift by using monthly batches to update the model. Essentially, creating an optimal instance selection every month. The limitation of the study is the use of the random forest as an analyzing method. Simplifying the data and losing temporal linkage between features. So, changing out the random forest for a more advanced model could improve predictive performance. Making it possible for this study to use their method combined with the proposed CNN-LSTM model of this thesis.

The final studies using DL are (Ashfaq et al., 2019; Hameed & Bukhari, 2020; Haishuai Wang et al., 2018). These studies all include historical datapoints from patients on a large scale and improve on the previous studies by combining high and low-level EHR data points. With the focus (Haishuai Wang et al., 2018), using the CNN to extract EHR data features automatically creates an end-to-end prediction model. This study provides promising results compared to manual feature extraction. The study (Hameed & Bukhari, 2020) focuses on the admission and

discharge locations as the most influencing features to predict readmission, showing new results because of the inclusion of alternative care locations. Taken from this study, it might be interesting to include discharge location as a variable combined with the low-level EHR data. The final study of (Ashfaq et al., 2019) shows the best method to predict readmission based on EHR data. This study combines high & low-level EHR data and can use historical data points from patients. The selection of features is made by doctors and machines, making a unique combination of features compared to the other studies. This study's limitation is that it only contains patients from a single region in Sweden, making it a homogeneous population. Furthermore, it does not use the clinical data points from lab results but states the lab as a categorical feature.

The studies (Cheng et al., 2016; Koulaouzidis et al., 2016; Liu et al., 2019) are not relevant to this thesis. They show exciting results, but completely different methods (Liu et al., 2019) use text mining on clinical notes for its predictions. The study does confirm that getting low-level clinical results are essential for prediction, as there are used as features but in the form of words instead of numbers. (Cheng et al., 2016) uses image classification by marking all the investigations of patient timelines with "1". Creating a matrix showing when and how many times an investigation occurred. This matrix is converted to an image and then used to predict the readmission. This method only uses meta-features and only performs well when 90%+ of the data is used for training, resulting in a low generalizable and low performing model. (Koulaouzidis et al., 2016)'s study is the highest performing one and shows a promising future research method as they use IOT health sensors on patients at home. Using the IoT sensors makes it possible to be closer to the patient and measure outside the hospital. Although the study shows promising results, it does so by only having a small dataset of 309 patients and only predicting one day in the feature.

All these methods are to use an LSTM method that handles historical data points and can combine high and low-level EHR data. The model should not look at second-hand influencing variables such as income or living location as this does not improve the predictability. It also shows the necessity for the model to include historical data from patients. Finally, the model should include the dynamic setting of (Yang et al., 2018) to dynamically predict over time, as this would deal with the challenge of temporality. An adaption of these methods is needed for this thesis because of the intricacy of our dataset. This thesis needs a model able to take in large amounts of historical EHR data.

For this reason, a focusing is given to the models using a recurrent structure for EHR data. The RNN structure of (Allam et al., 2019) is not a viable option because of the vanishing gradient problem. The CNN of (Haishuai Wang et al., 2018) shows promising results but lacks the performance of using the temporal links between features such as the LSTM model by (Ashfaq et al., 2019). The LSTM model's problem is that it will not scale when it is continuously fed new data, as the memory will increase because a lot of previous time steps are kept in memory.

Data analysis

Another factor influencing the predictive performance is the data used as input for the model. The studies show multiple different data sources such as simplified questionnaire answers, clinical notes, or clinical history of patients. Studies such as (Kang et al., 2016; Philbin & DiSalvo, 1999) using questionnaires mostly use demographic data of the patient to predict their change op readmission, showing the influence of age, living location, and social-economic class. These studies base the readmission risk on second-hand influences instead of direct clinical data, showing worse predictive power.

A notable observation is that relatively few studies use clinical investigations, such as ECG scans or echoes. More common is creating a meta-feature counting the number of scans, still resulting in high AUC scores. An explanation for this method can be that sicker patients need more investigation, mitigating the need for results from each investigation (Allam et al., 2019). A second reason can be that the used model cannot handle all clinical results from each test as studies using such features tend to use a linear regression using a relatively low number of features. The study of (Shah et al., 2015) is the study that uses the most clinical data points, creating a comprehensive clinical

pattern of each patient and showing promising results. This clinical pattern is also shown in the study using clinical notes combined with text mining (Liu et al., 2019), revealing direct clinical information from patients in a different format. Using the popular meta-features in combination with detailed clinical results show the best results and input for a model, as this gives the best representation of the patient's history (Ashfaq et al., 2019; Hameed & Bukhari, 2020; Yang et al., 2018). Using data taken from home sensors (Koulaouzidis et al., 2016) or image data (Cheng et al., 2016) is outside this thesis's scope because these data sources are not available.

Scalability

The only study focusing on scalable readmission data is (Zolfaghar et al., 2013), showing the need for a big data analytics platform such as Mahout when the dataset increases with new admissions over time. With Mahout, they use machine scaling with multiple nodes to predict readmission in parallel, making it possible to handle 1.5 million records of patient data. The other studies do not use large enough data sets to call it big data and do not show training time in their studies to compare.

Using an LSTM to predict readmission shows the best performance in the literature review, which is not surprising. LSTM is one of the state-of-the-art time series classification models (Smirnov & Nguifo, 2018). The problems surrounding the expensive computational need for LSTM make it hard to scale for big data. Using the techniques mentioned in 4.1, it is possible to efficiently adapt the LSTM to include a growing patient's history over time. Using this novel model with promising features such as the popular meta-features (count of admission, investigations) and low-level clinical results (lab, ECG, echo results), creates the optimal solution for readmission prediction that updates over time. No distributed analytics platforms, such as Mahout, are used because of the focus on data management and not machine scaling.

4. Solution process

The following chapter describes the complete process in developing and evaluating the CNN-LSTM model against the other models. The chapter follows the structure of the CRISP-DM cycle, starting from domain understanding to evaluation.

The experiment resources are all delivered from the UMCU hospital, consisting of a research pc with an Intel i5-6500 processor and 32GB ram. All code is executed using Python 3.7.5 combined with the anaconda environment, a full list of packages and versions are listed in Appendix 3.

4.1 Domain understanding

The experiment is conducted at the Department of Cardiology at the University Medical Center Utrecht in the Netherlands. The dataset used for the experiment comes from the UNRAVEL research data platform. The experiment follows the steps of CRISP-DM and is explained in chapter 1.4. Multiple medical, statistical, and technical experts from the UNRAVEL team have been consulted throughout the process. The UNRAVEL platform contains a unique dataset of tertiary patient data from the UMCU hospital. All patients have proven or suspected cardiac disease and have permitted to use their medical history data. The data is updated weekly and consists of a mix in EMR and SD data, all stored in temporal sequences.

The experiment consists of the first goal being to predict the readmission risk based on admission history. The unique addition is that a full admission window with daily timesteps is used as input for the LSTM model instead of aggregated results from previous admissions. Using the data windows would give the LSTM lower-level data and insight to make its prediction compared to baseline models. By using the full admissions window, the total data significantly increases. The second goal is to combat high training times by creating an addition to the LSTM with a CNN making testing to see if this addition would scale up time series analysis while keeping the same predictive performance. A comparison of the AUC is made to evaluate the models' predictive performance, where a higher AUC results in better predictive performance. Their training times and the number of epochs is compared to evaluate the models' scalability. Lower training times and a lower number of epochs means the model is more scalable.

With the medical supervisor's help from the UMCU, a comprehensive overview is created to describe the objective and current solutions for the problem entirely. The overview is shown in chapter 2 in this document, where an explanation is given for this experiment's need and to compare existing methods and results. With the technical supervisor's help from Utrecht University and the medical supervisor, a selection of the most promising methods and data selection is created; the selection is shown in Table 2. It explains which methods and data are optimal for the experiment. This experiment tries to predict readmission risk within 30 days because hospitals do not get reimbursement for readmitted patients within 30 days.

4.2 Data understanding

The data collected from the UNRAVEL database is selected with the help of the medical supervisor. A selection of heart failure patients is made by selecting patient id's based on medical diagnoses. These patients are then cross-referenced with medical data relevant to predicting the readmission risk of heart failure patients. All data is collected and exported into CSV files using the SAS enterprise platform, which is the front-end system connected to the UNRAVEL database.

The patients' diagnoses are stored in text, and to select the relevant patients, a filter is applied. Patients that have the following key-words in their diagnoses: "hartfal", "hartdecomp"," decompensatio cordis" are diagnosed with heart failure and are selected. Within this selection, an exclusion rule is created based on the following text: "Cardiaal

probleem, muv hartinfarct, hartritmestoornis, hartfalen", which excludes the diagnose of not having heart failure. All patient ids are selected based on these conditions and results in a list of 3632 patients.

Next to the patient list, a selection is made on admissions that adhere to the right characteristics. Not all patient admissions are selected as a viable admission. The characteristics of a viable admission are that the admission needs to be at the cardiology department. The admission was unplanned, meaning that the origin of admission was not from the waiting list. The admission needs to be clinical; this excludes checkup admissions or medicine pick up meetings. Finally, the admission needs to be longer than one day, including the critical admission requiring recovery time. Combining the patient list with their viable admissions creates a list of 7783 admissions.

These admissions serve as the time windows where all medical tests occur. The relevant medical tests and results are divided into the following categories: ECG tests, echo tests, patients' demographics, including length and weight, blood pressure tests, blood lab tests, medication history and related diagnoses. The results from these categories will fill the time window of the admissions for each patient. All data sources are connected based on the pseudo patient id and date; with these data points, it is possible to create a full admission window for each patient. The data characteristic of each data source is shown in Table 4.

Table 3: Total data taken per source from UNRAVEL platform.

Data source	Variable type	No. records	No. features
Patient demographics	Categorical / continuous	3632	3
Admission	Categorical / continuous	7783	7
ECG	Continuous	153859	19
Echo	Continuous	15008	31
Length	Continuous	60354	3
Weight	Continuous	60354	3
Blood pressure	Continuous	672409	5
Blood lab	Continuous	116102	11
Medication	Categorical	717092	3
Diagnoses	Categorical	3860	4

Exploring these data sources shows a large difference in the amount of test being consulted per patient. Blood pressure is tested multiple times per day and shows the most data, while echoes are tested the least. This difference shows the first challenge of sparsity when creating the final time windows, as it would create empty values when a test is not executed on that day. This also shows why it is hard to include low-level test results because it introduces a significant increase in missing values. The exploration also shows that the prediction target is imbalanced as 89,9% of patients are not readmitted within 30 days. The data quality shows to be mostly clean, as most of the data is generated by machines. The length and weight data source's quality shows the most problems as these are inserted by doctors and show a wrong comma placement or wrong characters. All manually entered data sources are analyses, and outliers are discussed with the medical supervisor to create cut-off limits to combat these errors.

Table 4: Patient and admission descriptive

Patient	
n	3632
Gender	Male: 2356 / Female: 1276
Age at discharge	Mean: 65 / Std: 15
Admission	
n	7783
Average admission duration	Mean: 12.6 days / Std: 16.7
Readmission within 30 days (%)	793 (10.1%)

4.3 Data preparation

The data preparation is essential since this thesis uniquely combines actual test results with temporal admission windows. Extending the meta-features used in other studies with low-level test results on day by day basis. The additional features come with the challenge of error-prone data due to manually entering test results and increased empty values because of the irregularity and sparsity found in HER records. Another challenge is converting all records into admissions windows compatible with the CNN-LSTM model. The addition of test features and temporal time windows are expected to increase predictive performance because of a more extensive representation of the patient's admission, compared to aggregating the features into a single row. The predictive performance is measured by the AUC score, where the temporal dataset is expected to result in a higher AUC score.

A selection of relevant features is created from the data sources based on literature and the medical supervisor's input. The medical supervisor selects the medical features from each test. Besides these features, the list is enriched by multiple meta-features showing promising results in the relevant literature. These features are counts of each test and admissions, each admission duration, the origin location, and discharge location.

To prepare the features for the models, a mix of preparation methods are performed. All data preparation steps are done in Python. Each data sources undergoes the following steps:

1. Feature conversion to the appropriate data type, such as astype(int) for patient id's, astype(float) for test values and as to_datetime() for all dates. All dates are also represented in YYYY-MM-DD.
2. Converted binary categorical features to 0 or 1 and used one-hot encoding for multivariable categorical features.
3. Renamed features to a common name so that the different datasets can be merged.
4. Removed outliers based on statistics or cut-off limits created by the medical supervisor.

With these steps, only test results are removed or converted. No patients or admissions are removed for the final dataset. To create an overview of the features per patient, a dataset is created where each row represents a patient with the columns representing the features. The demographic features are measured at first admission, combined with all the patients' test results. When a test has been performed multiple times on a patient, an average value is taken from those tests, resulting in a descriptive dataset used for Table 6. Based on this dataset is each feature analyzed using descriptive statistics to check for inconsistencies and missing values. Validating the steps of the data preparation to make sure no unwanted records are created.

To analyze the difference between the target classes, a test is done to compare each feature's values. Determining which test is necessary is decided based on the distribution of the feature. Checking each feature's distribution is done by visually inspecting each feature's histogram and applying a Shapiro-Wilk (SW) test. All missing values are removed before an SW test is performed. Based on the SW results, it is possible to determine the right test to compare the differences between classes for each feature. The SW test is measured with an alpha of 0.05, meaning that all features with a value >0.05 are normally distributed and <0.05 are not normally distributed.

The SW shows that all features are not normally distributed, resulting in a Kruskal-Wallis test to determine the difference between the target classes. All categorical features are compared using a Chi-square test. The overview based on all data, descriptive statistics, missingness and the differences between the prediction labels is shown in Table 6. An adaption of Table 6 with the values of the patient at baseline is given in Appendix 4.

Analyzing Table 6 shows that not all patients undergo the same test, with all patients at least having one ECG test. The features with the most missing values are the ones related to the echo test. When an echo is performed, measurements are done for specific features, so not all features are filled with every echo. Another reason for the missing values is that the missing values actual represents valuable information. When analyzing the ECG features, an empty PR interval can show the possibility of heart rhythm disorders. So, it is not possible to fill or remove all

empty test results. Another observation is that the most significantly different features are mostly the meta-features, with only some of the test results showing a significant difference between the classes. These features also show that patients with multimorbidity are more likely to be readmitted within 30 days.

Table 5: Feature overview of target classes. Continuous variables: Mean [Q1, Q3]. Categorical variables: n (%)

Feature		Missing	Overall	Readmitted > 30	Readmitted < 30	P-Value
Patients			3632	3074	558	
Origin location	clinical	0	3154 (86.8)	2664 (86.7)	490 (87.8)	0.002
	nonclinical		311 (8.6)	280 (9.1)	31 (5.6)	
	other		167 (4.6)	130 (4.2)	37 (6.6)	
Discharge location	clinical	0	302 (8.3)	273 (8.9)	29 (5.2)	<0.001
	nonclinical		3192 (87.9)	2665 (86.7)	527 (94.4)	
	other		138 (3.8)	136 (4.4)	2 (0.4)	
Gender	F	0	1276 (35.1)	1099 (35.8)	177 (31.7)	0.074
	M		2356 (64.9)	1975 (64.2)	381 (68.3)	
Age at discharge (years)		0	65.0 [55.0,75.0]	65.0 [55.0,75.0]	65.0 [55.2,73.0]	0.528
VentricularRate (b/min)		0	80.4 [72.5,89.4]	80.2 [72.2,89.4]	81.2 [74.1,89.5]	0.103
AtrialRate (b/min)		0	88.1 [76.7,104.8]	87.9 [76.3,104.9]	89.2 [79.3,104.3]	0.089
P RInterval (ms)		115	166.4 [148.8,186.9]	166.4 [148.9,187.0]	166.1 [148.8,186.1]	0.629
QRS Duration (ms)		0	112.7 [96.2,137.8]	111.7 [95.6,137.2]	117.2 [100.8,142.0]	<0.001
Q TInterval (ms)		0	403.9 [377.4,432.4]	403.3 [376.7,431.4]	408.7 [382.7,435.4]	0.010
PAxis (°)		108	54.7 [42.8,65.7]	55.1 [42.8,65.9]	53.0 [42.7,64.4]	0.058
RAxis (°)		0	23.1 [-12.1,59.5]	21.9 [-12.4,58.8]	28.4 [-9.4,62.6]	0.032
TAxis (°)		0	73.7 [46.5,103.8]	73.9 [46.8,104.1]	73.1 [46.3,102.5]	0.889
QOnset (ms)		0	209.3 [193.0,215.9]	209.8 [193.9,216.3]	206.2 [187.5,214.1]	<0.001
QOffset (ms)		0	265.2 [256.0,272.6]	265.2 [256.4,272.7]	265.2 [250.7,271.9]	0.192
POffset (ms)		119	173.3 [149.4,187.5]	174.0 [150.8,188.0]	168.6 [144.4,183.4]	<0.001
T Onset (ms)		0	310.7 [296.9,321.2]	310.9 [297.5,321.6]	310.2 [292.6,319.4]	0.024
T Offset (ms)		0	405.1 [384.0,419.9]	405.3 [384.8,420.0]	404.1 [378.2,418.6]	0.073
QRS Onset (ms)		0	208.6 [192.1,215.5]	209.1 [192.9,215.8]	205.1 [187.0,213.7]	<0.001
QRS Offset (ms)		0	264.6 [254.3,272.0]	264.6 [255.2,272.0]	264.6 [248.0,271.4]	0.164
QTcFredericia (ms)		0	436.3 [417.6,463.8]	435.4 [416.5,462.8]	442.9 [423.1,469.5]	<0.001
Ao V2 max (cm/s)		1046	125.2 [103.2,164.2]	125.7 [103.3,164.5]	122.3 [103.0,160.6]	0.423
Ao max PG (mm[Hg])		1046	6.4 [4.3,10.9]	6.4 [4.3,10.9]	6.2 [4.3,10.3]	0.528
EDV (mL)		1118	166.3 [115.9,237.5]	165.1 [114.6,237.4]	170.0 [120.6,238.7]	0.398
IVSd (cm)		1291	1.0 [0.9,1.2]	1.0 [0.9,1.2]	1.0 [0.9,1.2]	0.270
LV mass(C)d (g)		1308	213.7 [171.7,265.2]	212.8 [171.7,265.6]	217.5 [171.4,263.5]	0.304
LVIDd (cm)		1168	5.5 [4.8,6.2]	5.5 [4.8,6.2]	5.5 [4.9,6.1]	0.578
LVPWd (cm)		1253	1.0 [0.9,1.1]	1.0 [0.9,1.1]	1.0 [0.9,1.1]	0.822
TAPSE (cm)		1449	1.8 [1.5,2.1]	1.8 [1.5,2.1]	1.7 [1.4,2.0]	0.004
TR Max vel (cm/s)		1424	256.5 [229.0,288.9]	256.9 [229.5,289.1]	255.1 [227.1,287.9]	0.398
EF (%)		1322	43.0 [29.5,58.4]	43.2 [29.4,58.9]	41.7 [30.2,56.0]	0.579
ESV (mL)		1318	85.0 [47.8,135.6]	84.9 [47.3,135.4]	86.9 [51.6,140.2]	0.286
LA dimension (cm)		1767	4.6 [4.1,5.2]	4.6 [4.1,5.2]	4.7 [4.1,5.3]	0.120

LV mass(C)dI (g/m2)		1550	109.0 [89.9,132.7]	108.8 [89.3,132.3]	110.8 [93.0,135.2]	0.129
Lat Peak E' Vel (cm/s)		1545	8.1 [6.3,10.4]	8.1 [6.3,10.4]	8.5 [6.6,10.6]	0.090
LVIDs (cm)		1498	4.2 [3.3,5.3]	4.2 [3.3,5.3]	4.3 [3.4,5.3]	0.515
MV E/A		1509	1.2 [0.8,1.9]	1.2 [0.8,1.9]	1.5 [1.0,2.3]	<0.001
Med Peak E' Vel (cm/s)		1539	5.7 [4.6,7.2]	5.7 [4.6,7.2]	5.8 [4.6,7.1]	0.446
TDI E/e'		1616	13.1 [9.9,18.6]	12.9 [9.8,18.2]	13.9 [10.8,20.1]	0.004
TDI E/e' Lateraal		1621	9.3 [7.1,13.0]	9.3 [7.1,12.9]	9.7 [7.2,13.4]	0.264
BSA(Haycock) (m2)		1604	2.0 [1.8,2.1]	2.0 [1.8,2.1]	1.9 [1.8,2.1]	0.392
LA Volume (biplane) (mL)		2385	82.7 [64.3,105.7]	81.5 [62.4,103.9]	89.8 [71.2,112.5]	<0.001
LA Volume / BSA (mL/m2)		2425	42.3 [33.5,53.7]	41.5 [33.1,53.3]	45.9 [37.5,55.8]	<0.001
Length (meter)		705	1.7 [1.7,1.8]	1.7 [1.7,1.8]	1.7 [1.7,1.8]	0.135
Weight (kg)		705	77.0 [67.3,88.1]	77.0 [67.3,88.3]	76.8 [67.9,87.5]	0.623
BMI (kg/m2)		672	25.5 [22.9,28.9]	25.5 [22.9,28.9]	25.4 [22.7,28.5]	0.230
ALAT (U/L)		1113	26.7 [19.5,41.6]	26.0 [19.0,40.7]	29.5 [22.1,45.8]	<0.001
ASAT (U/L)		1134	32.0 [24.8,46.9]	31.7 [24.4,45.7]	34.8 [26.5,53.6]	<0.001
BNP (pmol/L)		1719	106.0 [49.0,237.8]	103.8 [47.0,237.7]	126.7 [56.7,239.0]	0.036
Creatinine (µmol/L)		1056	103.6 [81.7,138.9]	101.8 [80.6,136.7]	114.4 [88.5,147.0]	<0.001
eGFR (ml)		1261	55.8 [42.5,70.5]	56.2 [43.0,71.0]	52.8 [40.7,67.6]	0.005
gamma-GT (U/L)		1132	57.0 [33.0,110.1]	55.7 [32.0,107.7]	65.2 [35.8,124.3]	0.002
Kalium (mmol/L)		1062	4.2 [4.1,4.4]	4.2 [4.1,4.4]	4.3 [4.1,4.4]	0.101
Natrium (mmol/L)		1066	137.5 [136.0,139.0]	137.6 [136.0,139.0]	137.2 [135.6,138.6]	0.001
Systolic blood pressure (mmHg)		1093	118.8 [106.6,131.2]	119.6 [107.5,131.9]	115.0 [102.6,128.0]	<0.001
Diastolic blood pressure (mmHg)		1093	70.0 [64.1,76.3]	70.0 [64.1,76.2]	69.9 [63.9,76.5]	0.685
Ecg count		0	31.0 [15.0,58.0]	28.0 [14.0,50.0]	61.0 [31.0,95.0]	<0.001
Echo count		856	4.0 [2.0,7.0]	4.0 [2.0,7.0]	5.0 [3.0,12.0]	<0.001
Blood count		1093	148.0 [40.0,345.0]	127.0 [34.0,290.0]	334.0 [133.0,589.0]	<0.001
Lab count		1054	26.0 [11.0,57.0]	24.0 [10.0,51.0]	46.0 [26.0,100.0]	<0.001
Medication count		766	125.0 [46.0,289.8]	111.0 [42.2,260.0]	237.5 [95.5,482.5]	<0.001
Diagnoses count		0	32.0 [17.0,57.0]	30.0 [15.0,53.0]	47.0 [28.0,76.0]	<0.001
Ischemic	0.0	0	2462 (67.8)	2092 (68.1)	370 (66.3)	0.445
	1.0		1170 (32.2)	982 (31.9)	188 (33.7)	
Idiopathic	0.0	0	2962 (81.6)	2536 (82.5)	426 (76.3)	0.001
	1.0		670 (18.4)	538 (17.5)	132 (23.7)	
Avg adm duration		0	9.0 [5.0,16.0]	9.0 [5.0,16.0]	10.0 [6.4,15.1]	0.014
Adm count		0	1.0 [1.0,3.0]	1.0 [1.0,2.0]	4.0 [3.0,6.0]	<0.001
Adm duration		0	15.0 [8.0,34.0]	13.0 [7.0,27.0]	43.0 [23.0,76.0]	<0.001

Data structuring

To include the temporal dimension for the CNN-LSTM, a restructure is necessary. By converting all admissions into separate time series, the temporal dimension is added. The admission and discharge date of each admission per patient is taken and imputed with days, creating a window with the first row being the admission day and the final row the discharge day. All rows in between are days the patient was admitted to the hospital, creating a time window with days as time steps. A visual representation is given in Figure 16.

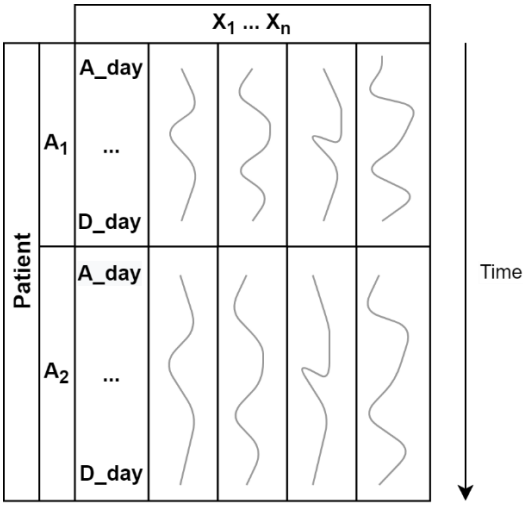


Figure 16: Timeseries structure of patient admissions (A1 & A2) with features (X) between admission day (A_day) and discharge day (D_day)

All tests results are saved in combination with the patient id and test date. Each test result is one row that can be merged with the patient's admission window (A_n) by patient id and date (E_1). Patients can receive a test outside of the selected admissions windows. Because of scheduled checkups or other planned events, these results are also used to fill the admission window and have the most relevant patient results. Filling the time windows is done by merging all test to the time window's closest date while only looking forward to finding the closest date. So, the tests done between admission windows can only influence the next admission window (E_2 and E_3). When multiple tests are done on the same day or between admission, an average is used for all test results while still adding each test to the count features. An overview is given in Figure 17.

The label given to each admission is based on the days between the discharge date and the next admission date. An admission is labelled a 1 when the next admission date is less than 30 days after the discharge date. Otherwise, the admission is labelled with a 0.

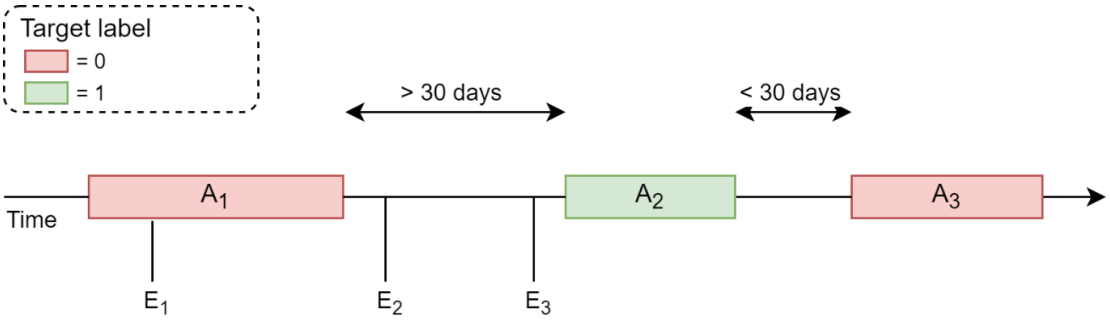


Figure 17: Admission window (A) labelling and exam (E) assigning overview.

Combining all 7783 admission windows results in a dataset of 98672 rows long and 70 features wide. All admissions are then min-max scaled and padded as this is required for the CNN-LSTM. The dataset is adjusted for

the baseline models used for comparison. These models are not able to utilize the temporal dimension. So, the data is reduced to one row per admission. The reduction is created by forward filling each admission and taking the final row of each admission as observation, simulating the model to know each feature's last possible result.

4.4 Modelling

The modelling technique proposed in this thesis is the CNN-LSTM model. The model would combine both the CNN and LSTM model strengths to create the best of both worlds' solution. The strengths of each model are explained in chapter 2.3. A comparison is made with four other models: DT, MLP, CNN and LSTM. These models are chosen to compare the difference between models that cannot use the temporal dimension (DT & MLP), and the CNN and LSTM are tested to see the improvement compared to their individual parts. The temporal models are expected to outperform on predictive performance compared to the baseline models because of the reasons given in section 4.3. Analyzing the models' scalability is done based on the training time and the total number of epochs. The LSTM is expected to provide the highest AUC score but the longest training time. The CNN-LSTM is expected to significantly reduce the training time while matching or improving the AUC score of the LSTM.

All models are built-in Python using the Scikit-learn package for the DT and TensorFlow 2.3.0 for the other models. Many experiments have been done to find the optimal model structure. The first model started small with just and LSTM layer and is evolved to the final CNN-LSTM model described below. The optimal hyperparameters are found using the method described in section training and test design.

CNN-LSTM

The proposed CNN-LSTM model consists of multiple layers and results in a many to one classification solution. The first layer is the convolution layer which takes in the input file and analyses the features using one-dimensional convolution combined with temporal padding. This step is repeated twice, after which each convolution is pooled using a max pool layer, reducing the rows of the admission window, and keeping the most relevant data points. The pooling layer's result is masked and used as input for the LSTM layer, which chronologically goes through all the pooling rows twice. The LSTM then outputs to a fully connected layer merging all variables to a final output cell using sigmoid activation to keep the output between 0 and 1. A visual representation is given in Figure 18.

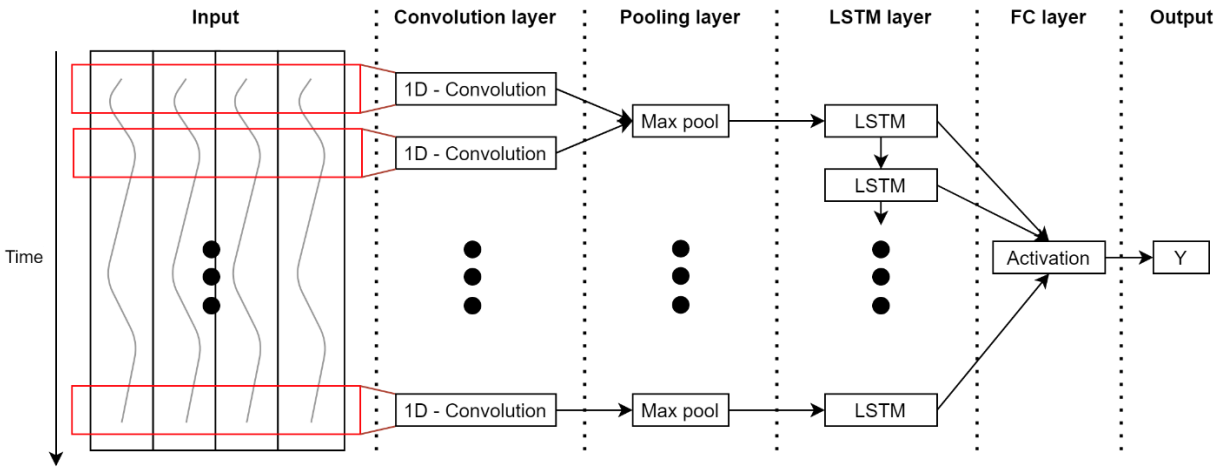


Figure 18: CNN-LSTM model

Other models

The CNN and LSTM model are also used as stand-alone models. The structure of these stand-alone models is the same as in their combined form. The only difference for the CNN is that the output after the pooling layer is flattened before it is connected to the fully connected layer. The difference for the LSTM is that the raw input is

masked and used as input, resulting in the full admission being used for the prediction. The MLP is built using three fully connected layers with a narrowing structure, and the DT is built using the Scikit package. The LR is replaced for the DT because the data does not meet the assumptions for the LR. All models have multiple parameters that are tuned to get the desired output. An overview of all model structures and tunable parameters for each model are shown in Appendix 5.

Training and test design

All admissions are divided into a train (80%) and test (20%) dataset using a stratified split. The stratification ensures that the class imbalance is the same in both the train and test set, dismissing a split's possibility of zero or low observations of the readmit <30 days class. To prevent overfitting, a combination of early stopping, dropout and regularization is applied during training. To handle the class imbalance of the dataset, a class weight is generated that balances out the observations. There is no separate trainset created at the start to use for final testing. The low amount of data and the low number of readmitted <30 days class make it unfavorable to split the data. CV is used instead of creating a separate test set to ensure the model is tested on multiple data splits, making the results more generalizable. Multiple data augmentation methods are tried, such as time flipping and event shuffling, but with minimal improvement for the final prediction.

The hyperparameters are set by selecting and evaluating a broad set of parameters on one-fold. After which, a subset of optimal parameters is evaluated using 5-fold cross-validation. Selecting the optimal hyper-parameters is done using the model's parameters, having the highest average AUC score of all 5-fold. The final model is then trained using 300 epochs with early stopping set to 10 epochs. The models are evaluated and compared by taking the average test AUC predicted over 5-folds.

5. Results

The following chapter evaluates the results from the experiment as well as discussing the interpretation of the results. Finally, the limitations and future works are discussed.

5.1 Evaluation

To answer SQ3 of the thesis, an evaluation is done on each model concerning their predictive performance using the AUC score and their ability to scale by analyzing the training time and the number of epochs. Table 7 shows the results of all models, ordered by the AUC score.

Predictive performance

An evaluation is done to check if the model predictions are better than random and statistically significant before comparing the results. The check is done by taking one prediction and measure the AUC score after shuffling the true labels 10000 times. Doing this test with an alpha of 0.05 shows that the AUC needs to be higher than 0.53 to be statistically significant.

Analyzing the models' predictive performance shows an improvement of the models using the temporal dimension compared to the baseline models. Looking at the baseline models shows that the MLP outperforms the DT on all metrics except the precision. The MLP is the best performing model using aggregated features. The best performing models benefit from the temporal dimension. Although all temporal models perform close to each other, the CNN shows moderate performance overall compared to all temporal models, tying on NPV with the CNN-LSTM and outperforming on the recall compared to the LSTM. The LSTM proves to be the best performing model overall, with only falling short of the CNN-LSTM regarding the NPV and recall. The CNN-LSTM model does not create the best of both worlds' situation regarding predictive performance, creating the lowest AUC score of all temporal models. However, it shows the highest recall score, which is impressive regarding the large imbalance in the data.

An ensemble method is also applied to improve predictive performance. The ensemble method uses the predictions of 2 models and takes the average of these predictions as the final output. The ensembles tried are an MLP & LSTM combination and a combination of the standalone CNN & LSTM. These two ensembles test if the combination of a non-temporal and temporal model improves the prediction power and the second ensemble tries to combine the strongest models. Both ensembles do not improve the predictive performance, as shown in the final two rows of Table 7.

Table 6: Experiment results

Model	Precision	NPV	Recall	F1	AUC	Duration (minutes)	Epochs
DT	0,12	0,90	0,20	0,15	0,52	0,00	-
MLP	0,12	0,91	0,60	0,19	0,55	0,47	201
CNN-LSTM	0,12	**0,93**	**0,75**	0,20	0,59	**5,93**	**16,2**
CNN	**0,13**	**0,93**	0,65	0,21	0,60	6,81	43,4
LSTM	**0,13**	0,92	0,61	**0,22**	**0,62**	16,96	28,2
MLP & LSTM	0,13	0,93	0,68	0,21	0,57	-	-
CNN & LSTM	0,13	0,93	0,64	0,22	0,58	-	-

Scalability

By evaluating the training time and training epochs, it is possible to determine the models' scalability. A model is scalable when the model can handle more data without significantly increasing training times. Analyzing Table 7 shows that the models with the lowest training times are the baseline models. The DT finishes near-instantaneous while the MLP needs about half a minute to finish training, resulting in the DT being more scalable than the MLP. Understandably, the baseline models training times are lower as the models need to process significantly less data than the temporal models.

Looking at the temporal models show a significant difference between the techniques used. The slowest temporal model is the LSTM which has the most complex internal structure and while the LSTM needs fewer epochs than the CNN. The CNN needs the most epochs out of the temporal models, but because the model is faster per epoch, the CNN takes less time than the LSTM making it the seconds fastest model, only falling short to the CNN-LSTM. The CNN-LSTM is the best model when evaluating scalability. While the model does show to take longer per epoch, the model needs the least epochs to converge, resulting in the quickest model overall. So, the CNN-LSTM does create the best of both worlds' scenario when it comes to scalability.

Explainability

As typical in healthcare and other domains, it is also preferred to know why the model gives a particular prediction. Current tools do not allow to analyze the temporal models extensively. An analysis of the feature importance is done to get an idea of which features influences the model prediction the most. The feature importance score is an average of the importance score over 5-folds made by the DT, with the top 25 features shown in Figure 19.

Analyzing the results show that the strongest feature is the patient's age at the time of discharge, which is also confirmed as a vital feature by the medical experts. The other features are a mix of high- and low-level features with the ECG tests values showing the most essential low-level features and age at discharge, combined with diagnose count and admission features being the most essential high-level features. A reason for the high number of ECG features can be that these features have the most variance because the ECG test is the most occurring. The test results also show high multicollinearity, explaining that multiple ECG test values are equally important. The reason for the high-level features to be important is grounded by the fact that patients that are more ill are generally older, have more test done and are longer in the hospital, explaining the high importance of the high-level features. While most of the features align with the literature's essential features, it does not align compared to the feature analysis. The model shows age at discharge to be the most crucial feature, while this feature is not significantly different between the groups. This selection of non-significant features does explain why the DT performs lower than the other models.

Combing all results from the experiments show the clear benefit of using the temporal data set when evaluating the AUC score, with all temporal models outperforming the baseline models. While this does significantly increase the data, combining the temporal data set with the proposed CNN-LSTM model shows quick training times, making the CNN-LSTM the most scalable option to improve readmission prediction compared to the baseline methods. Using the novel approach in this thesis of using the temporal data with the CNN-LSTM model shows an improvement in readmission prediction compared to baseline methods without creating long training times.

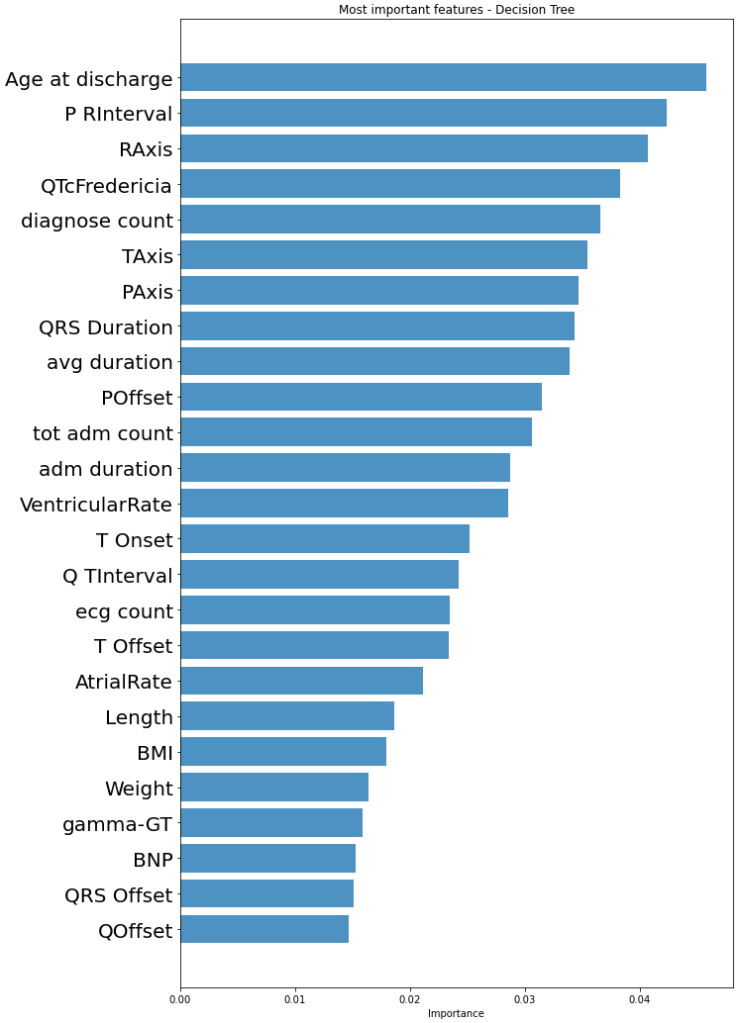


Figure 19: Top 25 Features from the decision tree

5.2 Discussion

With the ever-increasing data globally, the necessity for models that can handle more data is significant in many areas such as healthcare. The increase in data leads to more information for complex models to predict challenging topics more accurately, such as readmission prediction. Current research shows baseline models' limitations compared to RNN models when predicting time-series events because of temporal data exclusion. Including temporal data generally improves predictive power at the cost of significant longer training times. This thesis investigates a solution to create a scalable RNN for efficient time series analysis. This chapter discusses the main research question's results and corresponding sub-questions stated in chapter 1.3.

SQ. 1: What makes a machine learning model scalable?

The characteristics of a scalable model described in chapter 3.1 are closely related to the 4-V's of big data, but where big data focusses on storing data in machine learning, the focus is on how to discover meaningful value from the data. The following definition of scalable machine learning is given based on these characteristics. *"A scalable model is a model able to handle a large amount of data without consuming ever-growing amounts of computational resources within a practical time frame (Bekkerman et al., 2012)."* Achieving a scalable model can be done by focusing on one or more parts of the ML pipeline, such as machine scaling, algorithm optimization, and data management.

Machine scaling helps by adding more processing power with more compute units. This method's downside is that not all algorithms are optimized to work over multiple compute units, resulting in more overhead for the model to process the tasks in a distributed matter. Resulting in machine scaling not being a simple solution of just adding more compute units to improve the training times. Algorithm optimization is done by optimizing the process of finding the lowest loss by adapting hyperparameters. The first step is starting the neural network to converge to the lowest loss. The second step is making the converging happen as fast as possible, and the third step is to ensure that the lowest loss is achieved (global minima). Data management help regulate the continues incoming data streams. These streams update the model using batch (multiple instances) or dynamic learning (one instance).

Analyzing the LSTM internal structure shows a significant flaw. With the LSTM solving for the vanishing gradient problem, the model holds previous states in memory, leading to a linear increase in memory. To combat this, a reduction of input data is necessary. An efficient way of doing so is by convoluting the raw data before inserting it in the LSTM. The CNN-LSTM combination has shown to work in fields as images analysis and text mining. Creating a hybrid model that leverages both the CNN and LSTM strengths would benefit efficiency and scalability, reducing the input of the LSTM while keeping the most relevant features and making it possible to use the temporal relationship between features. Placing the CNN-LSTM model in a dynamic setting also deals with the final challenge of temporality, profiting from the continuous updates from new data.

SQ. 2: What are the current methods of predicting the readmission risk of patients?

Chapter 3.2 describes a review of recent readmission prediction papers with their results summarized in Table 3. A clear divide is shown in the average predictive performance of RNN compared to other methods while also making clear the right data features to use for readmission prediction. By analyzing the methods used in previous studies, a distinction is made between models that can use the temporal dimension or not. It is shown that the use of an LSTM outperforms other temporal models, and the model should be able to handle high- and low-levels features. The only method performing better is a method using sensors at home to analyze patients' health, resulting in a lower time to event, and giving more accurate data.

Analyzing the optimal features shows that it is best to use direct patient clinical data without influencing variables such as living location or social-economic class. Besides taking the medical tests' direct results, it is optimal to create meta-features of the test, such as the total count of tests taken. Combining these high-level meta-features with low-level medical test data shows the optimal feature set to predict readmission. Little attention is given to the scalability, with only one paper combining big data and machine scaling to test baseline model over large amounts of data. Creating a scalable version of the LSTM would create a better version of the best model. Using the novel CNN-LSTM model with promising features such as the popular meta-features (count of admission, investigations) and low-level clinical results (lab, ECG, echo results), creates the optimal solution for readmission prediction that updates over time.

SQ. 3: How does the adapted LSTM model perform compared to other (baseline) models concerning scalability and predictive performance?

The results described in chapter 4.5 show that based on the cardiology patient’s dataset, the temporal models' predictive performance outperforms the baseline models, showing the temporal dimension's impact. The best performing model is the LSTM model, although all temporal models do not show much difference in their predictive performance. The scalability results show that the baseline models perform significantly faster than the temporal models, which is understandable as the baseline models need to process far less data than the temporal models. Because of the reduction in data, it is not fair to compare the baseline modes' results directly with the temporal models. The most scalable temporal model is the novel approach of CNN-LSTM. The model needs the least epochs to converge, giving it the lowest overall training time of all temporal models. This fast convergence does come at a small predictive performance cost as the CNN-LSTM model scores slightly lower when comparing the AUC. From

this result, one can state that the CNN-LSTM combination creates the best of both world situation for scalable time series classification.

Predictive performance/scalability trade-off

Since our data structuring method increases the information available for the temporal models, one can expect the predictive power to increase. Analyzing the results shows this effect because when trying to scale the models, some form of aggregation is necessary, resulting in loss of information. Because of the aggregation, it is expected to have a trade-off between scalability and predictive power. Even though the baseline models are much quicker and easily scale to more data, it is not comparable to the temporal models as the difference in data processed is too large. The predictive power is also significantly less than the temporal models, making the models quick but ineffective. However, it is possible to compare the models within their group for the temporal models. The summary is that the temporal models show better predictive power, utilizing the temporal dimension to strengthen their predictions. The CNN-LSTM is the quickest model, making it the best scalable option out of all the models.

MRQ: How can the readmission risk of a patient with heart failure be predicted dynamically, using a scalable Long Short Term Memory model based on Electronic Health Records

This thesis proposed a new model for scalable time-series classifications by creating a CNN-LSTM combination, making it possible for a large amount of time-series data to be processed efficiently with minimal loss on predictive performance. The review of current research regarding readmission prediction and scalable time series analysis combined with the experiment results shows improved predictive performance of the temporal models. This thesis shows the clear benefit of the novel approach, creating detailed admission windows, including the low level EHR records. The temporal models outperform the baseline methods, showing the increase in data helping with the prediction. The proposed CNN-LSTM model show to be the most scalable model out of the temporal models. The CNN-LSTM trains on the data in 1/3 of the time compared to the LSTM model with only a slight trade-off in the predictive performance. With the CNN-LSTM model being the most scalable, it can quickly retrain over time and predict patients' readmission risk dynamically, creating a clear benefit over baseline models used in the experiment. That said, there is still work to be done as all models' predictive power is not strong compared to most of the methods in Table 2.

Clinical relevance

As the CNN-LSTM model is created to help as a decision support tool for doctors, an evaluation is done with the medical supervisor to analyze this thesis's clinical relevance. Discussing the results with the medical supervisor made it clear that this thesis is a first step towards the event's temporal analysis. However, the current predictive performance is not high enough for it to provide confident predictions. The current model gives calming predictions because of the high NPV and low precision values. Another reason for the model not being used in the clinical environment is the model's lack of interpretability. More work needs to be done to analyze why the CNN-LSTM predicts the given output per patient. While the results show improvement using the temporal data, an external validation is needed to validate the model’s predictions further. A randomized clinical trial is necessary to confirm the actual improvement to healthcare, resulting in the model’s actual clinical relevance.

5.3 Limitations & future work

The first limitation of the study is the patient cohort used for the analysis. The cohort is specialized because the UMCU is a tertiary hospital, and the study is done at the cardiology department. Combining this selection with the admission rules resulted in a large class imbalance, with only 10% of the readmissions being within 30 days. The literature study showed this to be 25 - 30% usually. This class imbalance is not optimal for machine learning models and making the results less generalizable for other hospital patients. The second limitation is related to the hyper-parameters used for the model comparison. The optimal hyper-parameters are chosen based on a pre-defined

value-list, it is possible that the best parameter was not included and not tested. The third limitation is the used features from test results as not all patients received the same test; it creates many empty values, which hurts prediction. It might be possible to combine features from the test results before applying the feature to the final timeline. Finally, it was impossible to test a sliding window's effect affecting the amount of necessary training data because of time restrictions. The sliding window's effect would be an exciting addition in future work as this addition would significantly help reduce training time when dealing with larger datasets. Another option for future research is to apply the CNN-LSTM model for a different application to see its impact. Combining this with additional model validation can create an improved version that is more generalizable for all many-to-on time series analysis. An external validation at another hospital is also interesting to research as well as a clinical trial to see the impact of the model when used by doctors.

6. Bibliography

Adler-Milstein, J., Everson, J., & Lee, S.-Y. D. (2015). *EHR Adoption and Hospital Performance: Time-Related Effects Health Services Research*. https://doi.org/10.1111/1475-6773.12406

Al-Shedivat, M., Wilson, A. G., Saatchi, Y., Hu, Z., & Xing, E. P. (2017). Learning scalable deep kernels with recurrent structure. *Journal of Machine Learning Research*, *18*, 1–37.

Allam, A., Nagy, M., Thoma, G., & Krauthammer, M. (2019). Neural networks versus Logistic regression for 30 days all-cause readmission prediction. *Scientific Reports*, *9*(1), 1–11. https://doi.org/10.1038/s41598-019-45685-z

Alonso, A. M., Nogales, F. J., & Ruiz, C. (2019). *A Single Scalable LSTM Model for Short-Term Forecasting of Disaggregated Electricity Loads*. http://arxiv.org/abs/1910.06640

Amarasingham, R., Moore, B. J., Tabak, Y. P., Drazner, M. H., Clark, C. A., Zhang, S., Reed, W. G., Swanson, T. S., Ma, Y., & Halm, E. A. (2010). An Automated Model to Identify Heart Failure Patients at Risk for 30-Day Readmission or Death Using Electronic Medical Record Data. In *Care* (Vol. 48, Issue 11).

Ashfaq, A., Sant'Anna, A., Lingman, M., & Nowaczyk, S. (2019). Readmission prediction using deep learning on electronic health records. *Journal of Biomedical Informatics*, *97*(July), 103256. https://doi.org/10.1016/j.jbi.2019.103256

Awan, S. E., Bennamoun, M., Sohel, F., Sanfilippo, F. M., & Dwivedi, G. (2019). Machine learning-based prediction of heart failure readmission or death: implications of choosing the right model and the right metrics. *ESC Heart Failure*, *6*(2), 428–435. https://doi.org/10.1002/ehf2.12419

Bahdanau, D., Cho, K., & Bengio, Y. (2015). *NEURAL MACHINE TRANSLATION*. 1–15.

Bekkerman, R., Bilenko, M., & Langford, J. (2012). *Scaling Up Machine Learning: Parallel and Distributed Approaches*. Cambridge University Press. https://books.google.nl/books?id=04V2nQAACAAJ

Bellazzi, R., & Zupan, B. (2008). Predictive data mining in clinical medicine: Current issues and guidelines. *International Journal of Medical Informatics*, *77*(2), 81–97. https://doi.org/10.1016/j.ijmedinf.2006.11.006

Brüngger, B., & Blozik, E. (2019). *Hospital readmission risk prediction based on claims data available at admission: a pilot study in Switzerland*. https://doi.org/10.1136/bmjopen-2018-028409

Chen, L., Wang, R., Yang, J., Xue, L., & Hu, M. (2019). Multi-label image classification with recurrently learning semantic dependencies. *The Visual Computer*, *35*, 1361–1371. https://doi.org/10.1007/s00371-018-01615-0

Cheng, Y., Wang, F., Zhang, P., & Hu, J. (2016). *Risk Prediction with Electronic Health Records: A Deep Learning Approach*.

Chilimbi, T., Suzue, Y., Apacible, J., & Kalyanaraman, K. (2014). Project ADAM: Building an efficient and scalable deep learning training system. *Proceedings of the 11th USENIX Symposium on Operating Systems Design and Implementation, OSDI 2014*, 571–582.

Chin, M. H., & Goldman, L. (1997). Correlates of early hospital readmission or death in patients with congestive heart failure. *American Journal of Cardiology*, *79*(12), 1640–1644. https://doi.org/10.1016/S0002-9149(97)00214-2

Choi, E., Taha Bahadori, M., Schuetz, A., & Stewart, W. F. (2016). *Doctor AI: Predicting Clinical Events via Recurrent Neural Networks*.

Demyanov, S. (2015). *Regularization methods for neural networks and related models*. http://minerva-access.unimelb.edu.au/handle/11343/57198

Dunlay, S. M., Redfield, M. M., Weston, S. A., Therneau, T. M., Long, K. H., Shah, N. D., & Roger, V. L. (2009). *Hospitalizations After Heart Failure Diagnosis: A Community Perspective*. https://doi.org/10.1016/j.jacc.2009.08.019

Faghri, F., Hashemi, S. H., Babaeizadeh, M., Nalls, M. A., Sinha, S., & Campbell, R. H. (2017). *Toward Scalable Machine Learning and Data Mining: the Bioinformatics Case*.

Farhat, N. H. (1992). Photonit neural networks and learning mathines the role of electron-trapping materials. *IEEE Expert-Intelligent Systems and Their Applications*, *7*(5), 63–72. https://doi.org/10.1109/64.163674

Felker, G. M., Leimberger, J. D., Califf, R. M., Cuffe, M. S., Massie, B. M., Adams, K. F., Gheorghiade, M., & O'Connor, C. M. (2004). Risk stratification after hospitalization for decompensated heart failure. *Journal of Cardiac Failure*, *10*(6), 460–466. https://doi.org/10.1016/j.cardfail.2004.02.011

Fox, J., & Weisberg, S. (2011). Cox Proportional-Hazards Regression for Survival Data in R. *Most*, *2008*(June), 1–18. https://doi.org/10.1016/j.carbon.2010.02.029

Gers, F. A., Urgen Schmidhuber, J. J., & Cummins, F. (1999). Learning to Forget: Continual Prediction with LSTM. In *IEE* (Vol. 2). http://www.idsia.ch/http://www.idsia.ch/

Gomes, H. M., Read, J., Bifet, A., Barddal, J. P., & Gama, J. (2019). Machine learning for streaming data. *ACM SIGKDD Explorations Newsletter*, *21*(2), 6–22. https://doi.org/10.1145/3373464.3373470

Goudjerkan, T., & Jayabalan, M. (2019). Predicting 30-day hospital readmission for diabetes patients using multilayer perceptron. *International Journal of Advanced Computer Science and Applications*, *10*(2), 268–275. https://doi.org/10.14569/ijacsa.2019.0100236

Guo, Y., Liu, Y., Bakker, E. M., Guo, Y., & Lew, M. S. (2018). CNN-RNN: a large-scale hierarchical image classification framework. *Multimed Tools Appl*, *77*, 10251–10271. https://doi.org/10.1007/s11042-017-5443-x

Gupta, P., Sharma, A., & Jindal, R. (2016). Scalable machine-learning algorithms for big data analytics: a comprehensive review. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, *6*(6), 194–214. https://doi.org/10.1002/widm.1194

Hameed, T., & Bukhari, S. A. C. (2020). Predicting 30-days all-cause hospital readmissions considering discharge-to-alternate-care-facilities. *HEALTHINF 2020 - 13th International Conference on Health Informatics, Proceedings; Part of 13th International Joint Conference on Biomedical Engineering Systems and Technologies, BIOSTEC 2020*, 864–873. https://doi.org/10.5220/0009385608640873

Ismail Fawaz, H., Forestier, G., Weber, J., Idoumghar, L., & Muller, P. A. (2019). Deep learning for time series classification: a review. *Data Mining and Knowledge Discovery*, *33*(4), 917–963. https://doi.org/10.1007/s10618-019-00619-1

Jin, B., Che, C., Liu, Z., Zhang, S., Yin, X., & Wei, X. (2018). Predicting the Risk of Heart Failure with EHR Sequential Data Modeling. *IEEE Access*, *6*, 9256–9261. https://doi.org/10.1109/ACCESS.2017.2789324

Kang, Y., McHugh, M. D., Chittams, J., & Bowles, K. H. (2016). Utilizing home healthcare electronic health records for telehomecare patients with Heart failure: A decision tree approach to detect associations with rehospitalizations. *CIN - Computers Informatics Nursing*, *34*(4), 175–182. https://doi.org/10.1097/CIN.0000000000000223

Keenan, P. S., Normand, S. L. T., Lin, Z., Drye, E. E., Bhat, K. R., Ross, J. S., Schuur, J. D., Stauffer, B. D., Bernheim, S. M., Epstein, A. J., Wang, Y., Herrin, J., Chen, J., Federer, J. J., Mattera, J. A., Wang, Y., & Krumholz, H. M. (2008). An administrative claims measure suitable for profiling hospital performance on the basis of 30-day all-cause readmission rates among patients with heart failure. *Circulation: Cardiovascular Quality and Outcomes*, *1*(1), 29–37. https://doi.org/10.1161/CIRCOUTCOMES.108.802686

Kim, T. Y., & Cho, S. B. (2019). Predicting residential energy consumption using CNN-LSTM neural networks. *Energy*, *182*, 72–81. https://doi.org/10.1016/j.energy.2019.05.230

Klinkenberg, R. (2004). Learning Drifting Concepts: Example Selection vs. Example Weighting. *Intell. Data Anal.*, *8*(3), 281–300.

Koulaouzidis, G., Iakovidis, D. K., & Clark, A. L. (2016). Telemonitoring predicts in advance heart failure admissions. *International Journal of Cardiology*, *216*, 78–84. https://doi.org/10.1016/j.ijcard.2016.04.149

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). *ImageNet Classification with Deep Convolutional Neural Networks*. http://code.google.com/p/cuda-convnet/

Lee, C., Luo, Z., Ngiam, K. Y., Zhang, M., Zheng, K., Chen, G., Ooi, C., Luen, W., Yip, J., Luo, Z., Ngiam, · K Y, Zheng, · K, Ooi, · B C, Yip, · W L J, Ngiam, K. Y., Zheng, K., Ooi, B. C., Yip, W. L. J., Zhang, M., & Chen, G. (2017). *Big Healthcare Data Analytics: Challenges and Applications*. https://doi.org/10.1007/978-3-319-58280-1_2

Li, Z., Ding, C., Wang, S., Wen, W., Zhuo, Y., Liu, C., Qiu, Q., Xu, W., Lin, X., Qian, X., & Wang, Y. (2019). E-RNN: Design optimization for efficient recurrent neural networks in FPGAS. *Proceedings - 25th IEEE International Symposium on High Performance Computer Architecture, HPCA 2019*, 69–80. https://doi.org/10.1109/HPCA.2019.00028

Liu, X., Chen, Y., Bae, J., Li, H., Johnston, J., & Sanger, T. (2019). Predicting Heart Failure Readmission from Clinical Notes Using Deep Learning. *Proceedings - 2019 IEEE International Conference on Bioinformatics and Biomedicine, BIBM 2019*, 2642–2648. https://doi.org/10.1109/BIBM47256.2019.8983095

Livieris, I. E., Pintelas, E., & Pintelas, P. (2020). *A CNN–LSTM model for gold price time-series forecasting*. https://doi.org/10.1007/s00521-020-04867-x

Maragatham, G., & Devi, S. (2019). LSTM Model for Prediction of Heart Failure in Big Data. *Journal of Medical Systems*, *43*(5). https://doi.org/10.1007/s10916-019-1243-3

Mayer, R., & Jacobsen, H. A. (2020). Scalable deep learning on distributed infrastructures: Challenges, techniques, and tools. *ACM Computing Surveys*, *53*(1). https://doi.org/10.1145/3363554

Montavon, G., Orr, G. G. B., Müller, K.-R., LeCun, Y., Bottou, L., Orr, G. G. B., Muller, K., Montavon, G., Orr, G. G. B., & Müller, K.-R. (2012). Neural Networks: Tricks of the Trade. *Springer Lecture Notes in Computer Sciences*, *MAY 2000*, 432. https://doi.org/10.1007/3-540-49430-8

Nick Street, W., & Kim, Y. S. (2001). A streaming ensemble algorithm (SEA) for large-scale classification. *Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 377–382. https://doi.org/10.1145/502512.502568

Ontology, F., & Travel, A. (2019). *Fuzzy Ontology and LSTM-Based Text Mining: A Transportation Network Monitoring System for Assisting Travel*. https://doi.org/10.3390/s19020234

Organization, W. H. (1993). *The ICD-10 Classification of Mental and Behavioural Disorders Clinical descriptions and diagnostic guidelines World Health Organization*.

Ouwerkerk, W., Voors, A. A., & Zwinderman, A. H. (2014). Factors influencing the predictive powerofmodels for predicting mortality and/or heart failure hospitalization inpatients with heart failure. *JACC: Heart Failure*, *2*(5), 429–436. https://doi.org/10.1016/j.jchf.2014.04.006

Parker, C. (2012). Unexpected challenges in large scale machine learning. *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 1–6. https://doi.org/10.1145/2351316.2351317

Pete Chapman, Julian Clinton, Randy Kerber, Thomas Khabaza, Thomas Reinartz, Colin Shearer, & Rüdiger Wirth. (2000). *CRISP-DM 1.0*. DaimlerChrysler.

Pham, T., Tran, T., Phung, D., & Venkatesh, S. (2017). *DeepCare: A Deep Dynamic Memory Model for Predictive Medicine*.

Philbin, E. F., & DiSalvo, T. G. (1999). Prediction of hospital readmission for heart failure: Development of a simple risk score based on administrative data. *Journal of the American College of Cardiology*, *33*(6), 1560–1566. https://doi.org/10.1016/S0735-1097(99)00059-5

Ponikowski, P., Anker, S. D., Alhabib, K. F., Cowie, M. R., Force, T. L., Hu, S., Jaarsma, T., Krum, H., Rastogi, V., Rohde, L. E., Samal, U. C., Shimokawa, H., Siswanto, B. B., Sliwa, K., & Filippatos, G. (2014). *World Heart Failure Alliance Global Heart Failure Awareness Programme Heart failure Preventing disease and death worldwide*.

Ponikowski, P., Voors, A. A., Anker, S. D., Bueno, H., Cleland, J. G. F., Coats, A. J. S., Falk, V., González-Juanatey, J. R.,

Harjola, V. P., Jankowska, E. A., Jessup, M., Linde, C., Nihoyannopoulos, P., Parissis, J. T., Pieske, B., Riley, J. P., Rosano, G. M. C., Ruilope, L. M., Ruschitzka, F., … Davies, C. (2016). 2016 ESC Guidelines for the diagnosis and treatment of acute and chronic heart failure. *European Heart Journal*, *37*(27), 2129-2200m. https://doi.org/10.1093/eurheartj/ehw128

Qayyum, A., Anwar, S. M., Awais, M., & Majid, M. (2017). Medical image retrieval using deep convolutional neural network. *Neurocomputing*, *266*, 8–20. https://doi.org/10.1016/j.neucom.2017.05.025

Razavian, N., Marcus, J., & Sontag, D. (2016). *Multi-task Prediction of Disease Onsets from Longitudinal Lab Tests*. https://github.

Read, J., Bifet, A., Pfahringer, B., & Holmes, G. (2012). Batch-incremental versus instance-incremental learning in dynamic and evolving data. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *7619 LNCS*, 313–323. https://doi.org/10.1007/978-3-642-34156-4_29

Ross, J. S., Mulvey, G. K., Stauffer, B., Patlolla, V., Bernheim, S. M., Keenan, P. S., & Krumholz, H. M. (2008). Statistical models and patient predictors of readmission for heart failure: A systematic review. *Archives of Internal Medicine*, *168*(13), 1371–1386. https://doi.org/10.1001/archinte.168.13.1371

Roy, S. B., Teredesai, A., Zolfaghar, K., Liu, R., Hazel, D., Newman, S., & Marinez, A. (2015). Dynamic hierarchical classification for patient risk-of-readmission. *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, *2015-Augus*, 1691–1700. https://doi.org/10.1145/2783258.2788585

SAITO, H., SHOJI, M., TAKI, I., MURASE, R., KAMEI, D., SHINKE, T., & IWAI, S. (2020). Five Prognostic Factors for Readmission in Patients Over 75 Years Old with Worsening Heart Failure. *The Showa University Journal of Medical Sciences*, *32*(1), 33–42. https://doi.org/10.15369/sujms.32.33

Sammani, A., Jansen, · M, Linschoten, · M, Bagheri, · A, De Jonge, · N, Kirkels, · H, Van Laake, · L W, Vink, · A, Van Tintelen, J. P., Dooijes, · D, Te Riele, · A S J M, Harakalova, · M, Baas, · A F, Asselbergs, F. W., Jansen, M., Van Tintelen, · J P, Bagheri, A., & Vink, A. (2019). UNRAVEL: big data analytics research data platform to improve care of patients with cardiomyopathies using routine electronic health records and standardised biobanking. *Netherlands Heart Journal*, *27*, 426–434. https://doi.org/10.1007/s12471-019-1288-4

Savarese, G., & Lund, L. H. (2016). Pharmacological Therapy Nitrates as a Treatment of Acute Heart Failure Pharmacological Therapy. *The Journa of the Heart Team*, *3*(1), 51–55. https://doi.org/10.15420/cfr.2016

Shah, S. J., Katz, D. H., Selvaraj, S., Burke, M. A., Yancy, C. W., Gheorghiade, M., Bonow, R. O., Huang, C. C., & Deo, R. C. (2015). Phenomapping for novel classification of heart failure with preserved ejection fraction. *Circulation*, *131*(3), 269–279. https://doi.org/10.1161/CIRCULATIONAHA.114.010637

Shahid, N., Rappon, T., & Berta, W. (2019). Applications of artificial neural networks in health care organizational decision-making: A scoping review. *PLoS ONE*, *14*(2), 1–22. https://doi.org/10.1371/journal.pone.0212356

Smirnov, D., & Nguifo, E. M. (2018). Time Series Classification with Recurrent Neural Networks. *ECML/PKDD Workshop on Advanced Analytics and Learning on Temporal Data*, 1–8.

Srivastava, N., Hinton, G., Krizhevsky, A., & Salakhutdinov, R. (2014). Dropout: A Simple Way to Prevent Neural Networks from Overfitting. In *Journal of Machine Learning Research* (Vol. 15).

Sun, R. (2019). *Optimization for deep learning: theory and algorithms*. 1–60. http://arxiv.org/abs/1912.08957

Szegedy, C., Reed, S., Sermanet, P., Vanhoucke, V., & Rabinovich, A. (2014). *Going deeper with convolutions*. 1–12.

Tripoliti, E. E., Papadopoulos, T. G., Karanasiou, G. S., Naka, K. K., & Fotiadis, D. I. (2017). Heart Failure: Diagnosis, Severity Estimation and Prediction of Adverse Events Through Machine Learning Techniques. *Computational and Structural Biotechnology Journal*, *15*, 26–47. https://doi.org/10.1016/j.csbj.2016.11.001

Tsironi, E., Barros, P., Weber, C., & Wermter, S. (2017). An analysis of Convolutional Long Short-Term Memory Recurrent Neural Networks for gesture recognition. *Neurocomputing*, *268*, 76–86. https://doi.org/10.1016/j.neucom.2016.12.088

Tsymbal, A. (2004). The problem of concept drift: definitions and related work. *Computer Science Department, Trinity College Dublin*.

Uddin, M. Z., & Hassan, M. M. (2019). Activity Recognition for Cognitive Assistance Using Body Sensors Data and Deep Convolutional Neural Network. *IEEE Sensors Journal*, *19*(19), 8413–8419. https://doi.org/10.1109/JSEN.2018.2871203

Walters, S. J. (2009). What is a Cox model? *Survival*, *May*, 1–8.

Wang, F., Zhang, P., Qian, B., Wang, X., & Davidson, I. (2014). *Clinical Risk Prediction with Multilinear Sparse Logistic Regression*. https://doi.org/10.1145/2623330.2623754

Wang, Haishuai, Cui, Z., Chen, Y., Avidan, M., Abdallah, A. Ben, & Kronzer, A. (2018). Predicting Hospital Readmission via Cost-Sensitive Deep Learning. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, *15*(6), 1968–1978. https://doi.org/10.1109/TCBB.2018.2827029

Wang, Haixun, Fan, W., Yu, P. S., & Han, J. (2003). Mining concept-drifting data streams using ensemble classifiers. *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 226–235. https://doi.org/10.1145/956750.956778

Wang, J., Yang, Y., Mao, J., Huang, Z., Huang, C., & Xu, W. (2016). *CNN-RNN: A Unified Framework for Multi-label Image Classification*.

Wen, W. (2019). *Efficient and Scalable Deep Learning Efficient and Scalable Deep Learning*.

Wieringa, R. J. (2014). Design science methodology: For information systems and software engineering. In *Design Science Methodology: For Information Systems and Software Engineering*. https://doi.org/10.1007/978-3-662-43839-8

Yang, T., Yang, Y., Jia, Y., & Li, X. (2018). *Dynamic prediction of hospital admission with medical claim data*. https://doi.org/10.1186/s12911-019-0734-y

Yazdani, R., Ruwase, O., Zhang, M., He, Y., Arnau, J.-M., & Gonzalez, A. (2019). *LSTM-Sharp: An Adaptable, Energy-Efficient Hardware Accelerator for Long Short-Term Memory*. http://arxiv.org/abs/1911.01258

Yu, Y., Si, X., Hu, C., & Zhang, J. (2019). *A Review of Recurrent Neural Networks : LSTM Cells and Network Architectures*. *1270*, 1235–1270. https://doi.org/10.1162/neco

Zheng, S., Hanchate, A., & Shwartz, M. (2019). One-year costs of medical admissions with and without a 30-day readmission and enhanced risk adjustment. *BMC Health Services Research*, *19*(1), 1–10. https://doi.org/10.1186/s12913-019-3983-7

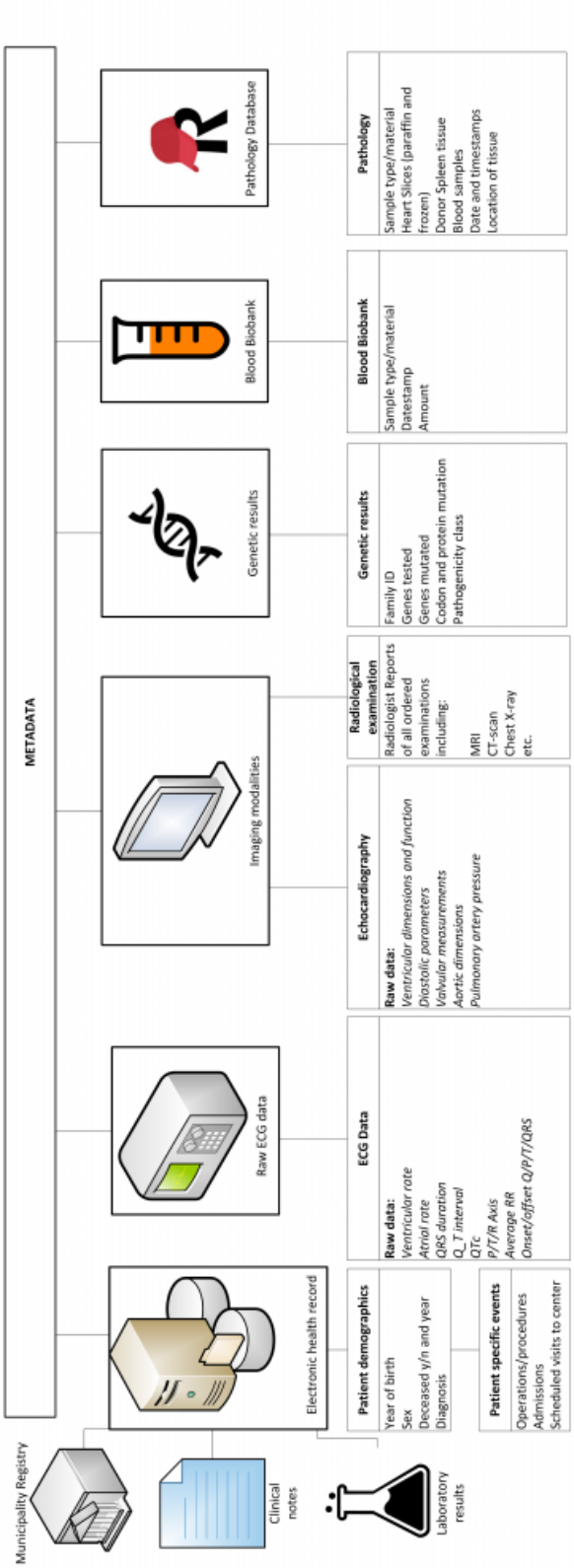
Zolfaghar, K., Meadem, N., Teredesai, A., Roy, S. B., Chin, S. C., & Muckian, B. (2013). Big data solutions for predicting risk-of-readmission for congestive heart failure patients. *Proceedings - 2013 IEEE International Conference on Big Data, Big Data 2013*, 64–71. https://doi.org/10.1109/BigData.2013.6691760

7. Appendix

1. Model abbreviations

DT: Decision Tree	Baseline
LR: Linear Regression	Baseline
RNN: Recurrent Neural Network	Deep learning
HDC: Hierarchical Dynamic Clustering	Baseline
DSRF: Dynamic Survival Random Forest	Deep learning
CNN: Convolutional Neural network	Deep learning
NN: Neural Network	Deep learning
LSTM: Long Short Term Memory	Deep learning
RF: Random Forest	Baseline
NB: Naive Bayes	Baseline

2. UNRAVEL framework



3. Python packages used

Name	version
Python	3.7.9
Numpy	1.18.5
Tensorflow	2.3.0
Pandas	1.1.3
Sklearn	0.23.2
matplotlib	3.3.2
pathlib	2.3.5

4. Table one of patient cohort at baseline

		Missing	Overall	0.0	1.0	P-Value
n			3632	3074	558	
Origin location	clinical	0	3154 (86.8)	2664 (86.7)	490 (87.8)	0.002
	nonclinical		311 (8.6)	280 (9.1)	31 (5.6)	
	other		167 (4.6)	130 (4.2)	37 (6.6)	
Discharge location	clinical	0	302 (8.3)	273 (8.9)	29 (5.2)	<0.001
	nonclinical		3192 (87.9)	2665 (86.7)	527 (94.4)	
	other		138 (3.8)	136 (4.4)	2 (0.4)	
Gender	F	0	1276 (35.1)	1099 (35.8)	177 (31.7)	0.074
	M		2356 (64.9)	1975 (64.2)	381 (68.3)	
Age at discharge		0	65.0 [55.0,75.0]	65.0 [55.0,75.0]	65.0 [55.2,73.0]	0.528
VentricularRate		237	81.3 [71.9,91.3]	81.3 [71.9,91.5]	81.5 [72.1,90.2]	0.985
AtrialRate		237	86.9 [74.3,103.9]	86.9 [74.4,103.9]	86.9 [74.2,103.4]	0.954
P RInterval		370	163.5 [146.3,184.5]	163.2 [146.5,184.7]	165.0 [145.5,183.6]	0.707
QRS Duration		237	108.0 [94.2,133.8]	107.2 [93.7,133.4]	111.7 [96.9,137.7]	0.003
Q TInterval		237	400.6 [373.6,429.3]	400.2 [373.1,428.8]	402.5 [376.6,433.1]	0.109
PAxis		364	53.7 [40.5,65.4]	54.0 [40.7,65.7]	52.0 [39.4,64.2]	0.043
RAxis		237	20.8 [-14.5,57.7]	20.0 [-15.1,57.2]	27.7 [-11.0,61.0]	0.033
TAxis		237	74.1 [42.9,105.0]	74.1 [42.5,105.0]	74.1 [44.3,106.0]	0.717
QOnset		237	210.6 [194.2,216.5]	211.0 [195.5,216.8]	207.5 [186.7,214.9]	<0.001
QOffset		237	265.0 [255.7,272.3]	265.0 [256.2,272.5]	264.6 [247.5,271.3]	0.076
POffset		372	174.8 [148.0,188.9]	175.6 [150.1,189.3]	170.4 [135.9,185.9]	<0.001
T Onset		237	309.9 [294.7,321.4]	310.2 [295.9,321.5]	308.4 [285.5,320.7]	0.010
T Offset		237	404.1 [381.0,420.2]	404.4 [382.3,420.3]	402.3 [366.1,418.9]	0.018
QRS Onset		237	210.0 [193.1,216.2]	210.5 [194.2,216.4]	206.4 [183.7,214.0]	<0.001
QRS Offset		237	264.4 [253.4,271.8]	264.5 [254.6,272.0]	263.9 [239.8,270.8]	0.025
QTcFredericia		237	434.1 [414.4,461.0]	433.6 [414.3,459.7]	438.3 [416.6,465.3]	0.028
Ao V2 max		1925	124.1 [99.5,162.2]	125.1 [100.5,162.1]	117.3 [96.6,162.4]	0.136
Ao max PG		1925	6.2 [4.0,10.7]	6.3 [4.1,10.7]	5.6 [3.8,11.0]	0.160
EDV		1976	176.6 [122.3,262.6]	175.3 [121.8,264.5]	186.3 [127.1,254.1]	0.710
IVSd		2139	1.0 [0.9,1.2]	1.0 [0.9,1.2]	1.0 [0.8,1.2]	0.940
LV mass(C)d		2156	222.5 [177.3,275.0]	222.4 [178.3,274.9]	223.0 [172.7,277.2]	0.858
LVIDd		2032	5.6 [5.0,6.4]	5.6 [4.9,6.4]	5.7 [5.0,6.3]	0.657
LVPWd		2109	1.0 [0.9,1.1]	1.0 [0.9,1.1]	1.0 [0.8,1.1]	0.816
TAPSE		2286	1.8 [1.4,2.1]	1.8 [1.4,2.1]	1.7 [1.3,2.1]	0.339
TR Max vel		2259	263.0 [232.3,294.8]	262.8 [231.8,294.2]	264.8 [234.0,298.0]	0.433
EF		2164	37.7 [25.6,53.9]	38.2 [25.7,54.8]	36.4 [25.6,49.1]	0.212
ESV		2160	97.4 [55.2,152.6]	97.0 [53.7,152.6]	103.1 [63.8,153.2]	0.257
LA dimension		2521	4.6 [4.1,5.2]	4.6 [4.1,5.2]	4.6 [4.1,5.3]	0.165
LV mass(C)dI		2395	115.7 [94.2,140.6]	115.3 [94.1,139.7]	118.8 [94.9,144.4]	0.352
Lat Peak E' Vel		2400	7.9 [6.0,9.9]	7.9 [6.0,9.9]	8.0 [6.3,10.0]	0.465
LVIDs		2331	4.6 [3.5,5.6]	4.5 [3.4,5.6]	4.6 [3.7,5.7]	0.192
MV E/A		2316	1.2 [0.8,1.9]	1.2 [0.8,1.9]	1.5 [0.9,2.4]	<0.001
Med Peak E' Vel		2393	5.6 [4.4,7.0]	5.5 [4.3,7.0]	5.7 [4.5,6.8]	0.629

TDI E/e'		2461	14.0 [10.2,19.1]	13.8 [10.2,18.9]	14.6 [10.9,21.2]	0.061
TDI E/e' Lateraal		2474	9.9 [7.5,13.7]	9.8 [7.5,13.6]	10.2 [7.1,14.5]	0.608
BSA(Haycock)		2459	2.0 [1.8,2.1]	2.0 [1.8,2.1]	1.9 [1.8,2.1]	0.158
LA Volume (biplane)		3010	86.6 [67.0,108.4]	84.6 [65.6,107.1]	93.0 [72.5,121.7]	0.015
LA Volume / BSA		3040	43.5 [34.5,55.8]	43.3 [33.9,55.1]	45.1 [37.1,63.8]	0.024
length		1615	1.7 [1.7,1.8]	1.7 [1.7,1.8]	1.7 [1.7,1.8]	0.134
Weight		1615	77.0 [67.1,89.2]	77.1 [67.1,89.2]	76.5 [66.6,89.1]	0.673
BMI		1596	25.5 [22.9,29.1]	25.6 [23.0,29.1]	25.1 [22.4,28.5]	0.128
ALAT		2060	29.3 [20.2,47.2]	28.3 [20.0,46.2]	33.6 [24.6,52.9]	0.001
ASAT		2073	34.0 [25.8,51.4]	33.6 [25.5,50.1]	36.4 [27.5,64.4]	0.047
BNP		2579	142.5 [65.0,291.5]	135.5 [64.0,290.0]	168.5 [73.6,313.1]	0.214
Creatinine		2033	96.3 [77.6,126.2]	95.6 [77.3,124.5]	99.9 [80.6,130.9]	0.105
eGFR		2248	57.5 [45.1,72.6]	57.6 [45.4,73.0]	56.0 [43.3,69.7]	0.098
gamma-GT		2078	61.2 [35.0,113.7]	60.0 [34.3,113.0]	67.0 [39.9,118.4]	0.066
Kalium		2034	4.2 [4.0,4.4]	4.2 [4.0,4.4]	4.2 [4.1,4.4]	0.587
Natrium		2035	137.5 [135.8,139.0]	137.5 [135.9,139.2]	137.0 [135.5,138.4]	0.002
Systolic blood pressure		2037	118.0 [105.1,130.3]	118.3 [105.9,130.8]	113.7 [100.9,127.0]	0.001
Diastolic blood pressure		2037	69.8 [64.3,75.2]	69.7 [64.3,75.1]	70.4 [64.1,75.6]	0.360
Ecg count		237	15.0 [8.0,27.0]	14.0 [8.0,25.0]	22.0 [13.0,36.0]	<0.001
Echo count		1710	2.0 [1.0,4.0]	2.0 [1.0,4.0]	3.0 [2.0,5.0]	<0.001
Blood count		2037	52.0 [22.0,100.0]	47.0 [21.0,92.0]	85.0 [43.0,132.0]	<0.001
Lab count		2031	18.0 [8.0,39.0]	16.0 [7.0,36.0]	31.0 [14.8,56.2]	<0.001
Medication count		1664	58.5 [21.0,121.0]	56.0 [21.0,115.0]	78.0 [14.2,172.5]	0.002
Diagnoses count		20	14.0 [8.0,27.0]	14.0 [8.0,27.0]	16.0 [8.0,28.0]	0.021
Ischemic	0.0	0	2462 (67.8)	2092 (68.1)	370 (66.3)	0.445
	1.0		1170 (32.2)	982 (31.9)	188 (33.7)	
Idiopathic	0.0	0	2962 (81.6)	2536 (82.5)	426 (76.3)	0.001
	1.0		670 (18.4)	538 (17.5)	132 (23.7)	
Avg adm duration		0	9.0 [5.0,16.0]	9.0 [5.0,16.0]	10.0 [6.4,15.1]	0.014
Adm count		0	1.0 [1.0,3.0]	1.0 [1.0,2.0]	4.0 [3.0,6.0]	<0.001
Adm duration		0	15.0 [8.0,34.0]	13.0 [7.0,27.0]	43.0 [23.0,76.0]	<0.001

5. Hyperparameters

Parameter	Value option
Pooling	[1, 2, 3, 4, 5]
Filter	[1, 3, 4, 5, 8, 10]
Dropout	[0, 0.01, 0.05, 0.1, 0.15, 0.2]
L1 regularizers	[0, 0.1, 0.01, 0.001, 0.0001, 0.00001]
L2 regularizers	[0, 0.1, 0.01, 0.001, 0.0001, 0.00001]
Learning rate	[0.0001, 0.0005, 0.00001, 0.00005]

CNN-LSTM	
convolution layer	
Hidden nodes	70
filter	4
convolution layer	
Hidden nodes	70
filter	4
Pooling	3
Masking layer	
LSTM layer	
Hidden nodes	64
LSTM layer	
Hidden nodes	32
Dense layer	
Hidden nodes	10
output layer	
Hidden nodes	1
Optimization	
Recurrent dropout	0,2
L1 & L2 regulizer	0,01
Batchnormalization	

CNN	
convolution layer	
Hidden nodes	70
filter	4
convolution layer	
Hidden nodes	70
filter	4
Pooling	3
Flatten layer	
Dense layer	
Hidden nodes	10
output layer	
Hidden nodes	1
Optimization	
Dropout	0,2
L1 & L2 regulizer	0,01
Batchnormalization	

MLP	
Dense layer	
Hidden nodes	70
Dense layer	
Hidden nodes	50
Dense layer	
Hidden nodes	10
output layer	
Hidden nodes	1
Optimization	
Dropout	0,1
L1 & L2 regulizer	0,001

LSTM	
Masking layer	
LSTM layer	
Hidden nodes	64
LSTM layer	
Hidden nodes	32
Dense layer	
Hidden nodes	10
output layer	
Hidden nodes	1
Optimization	
Recurrent dropout	0,2
L1 & L2 regulizer	0,001
Batchnormalization	