Analysis of Temporal Network Architectures for Phase Detection

Dissertation submission for Project module while studying Computer Science for second semester of academic year 2020–2021.

Written by:
Mr Borys Kubisty

In association with:
Dr Massoud Zolgharni, Project Supervisor
Dr Elahe Kani-Zabihi, Module Leader

University of West London
London, United Kingdom

Abstract

Computer-assisted identification of end-diastolic (ED) and end-systolic (ES) heart phases within echocardiography (echo) scan enables a level of automation, a process previously only achievable by qualified medical staff members. Identifying ED and ES heart phases using machine learning (ML) is a part of an automated heart assessment. The objective is to streamline and shorten the acquisition, analysis and interpretation process through automation. Within a heart assessment, locating heart phases will lead to additional analytical and possibly machine learning assisted tasks, where a goal might be to establish the heart's ejection fraction value. Previous research in this field used established open-source machine learning tools and techniques that allowed phase detection to match medical experts in accuracy. ML model architectures established in heart-phase detection typically consist of spatial and temporal feature extraction segments.

This research focuses on the possible improvements using variations of neural network architectures, primarily focusing on the temporal aspect. Convolutional neural networks (CNNs) can take a data instance, and output easily interpreted data by subsequent model layers, a process known as spatial feature extraction. The subsequent model layer takes the spatially feature-extracted sample data and uses recurrent neural networks (RNNs) to enable predictions on series data. A CNN and RNN model architecture usability applies to heart-phase detection, where expertly annotated ED and ES echo scans are the ground truth for model training.

Results show that variates tested have the ability to outperform some existing solutions, falling behind architectures with more comprehensive spatial capturing characteristics. Additionally, performance analysis between variates suggests networks with bidirectional architectures increase computation time significantly compared to their counterparts.

Table of Contents

Abstract 2
Motivation 6
Literature 6
  Survey 6
  Evaluation 8
Methodology 9
  Dataset Considerations 9
  Sample Considerations 11
  Classification Type 11
  Sample Formation 12
  Capturing Spatial Characteristics 14
  Capturing Temporal Characteristics 16
    Sequence Prediction Type 16
    Multilayer Perceptron vs Recurrent Neural Network 18
    Network Training 18
    Backpropagation Through Time 20
    Recurrent Neural Network Variations 21
    Loss and Metrics 22
    Optimiser 23
  Performance Metric 24
  Converting Prediction Output to Labels 24
Results 25
  Accuracy 25
  Efficiency 30
Conclusion 31
  Model Performance 31
  Future Work 33
  Availability 33
Bibliography 34

Table of Figures

Figure 1 – Stages in a self-driven heart assessment 6
Figure 2 – Detailed stages within a self-driven heart assessment 6
Figure 3 – Holistic model view featuring dataset preprocessing, spatiotemporal feature extraction with devitations of recurrent neural network architectures, and label generation from regression output 10
Figure 4 – Framing echo scan labelling as regression with core value of 0.5 having interpolation between ED and ES with 0 and 1, respectively. 12
Figure 5 – Echo scan labelling represented as a raw table showing relation to a regression line format 12
Figure 6 – Segmentation of a full-length sample to capture ED and ES phases in a regression view 13
Figure 7 – Regression view of heart phases with initial timestep referencing an ED phase 13
Figure 8 – Deconstruction of an image showing pixels 14
Figure 9 – Deconstruction of an image showing pixels with three colour channels: red, green, and blue 15
Figure 10 – Two implementations of using raw data to produce a result where processes differ in the responsibility of feature extraction 15
Figure 11 – Diagram containting an non-order dependant sample set with order depandent frames for each samples 16
Figure 12 – Network structure consisting of multiple inputs and hidden states to produce an output vector 17
Figure 13 – Showing relationship between the network structure consisting of multiple inputs, hidden states, and an output vector, with the expected output from a phase detection regression problem 17
Figure 14 – Folded view of a network 19
Figure 15 – Unfolded view of a network 20
Figure 16 – Model structure showing network layers per variation 22
Figure 17 – Naive regression implementation of frame encoding for echo scan .. 23

Figure 18 – Loss function mapping of two for two different network instances (source: blog.paperspace.com [23]) 24
Figure 19 – Data points highlighted which reference the ED and ES phase phases 25
Figure 20 – Before/after of Savizky–Golay digital filter on a staggered dataset . 25
Figure 21 – Final error values (mse and mae) per variate after training over 50 epochs 26
Figure 22 – Error value (mse) for validation data throughout the training process per variate 27
Figure 23 – Error value (mse) for training and validation data throughout the training process for variate E 27
Figure 24 – Error value (mse) for validation data throughout the training process with notes of lowest error per variate 28
Figure 25 – ED error value (mae) frequency (histogram) for testing data 29
Figure 26 – ES error value (mae) frequency (histogram) for testing data 29
Figure 27 – Time to complete training of 50 epoch in seconds per variate 30
Figure 28 – Time to complete training of 50 epochs in seconds against number of trainable parameters per variate 31
Figure 29 – Alternative regression sample formation with more sudden phase interpolation (source: Dezaki et al. [7]) 33

Table of Tables

Table 1 – Hot-ones encoding of an echo scan 11
Table 2 – Model structure showing network layers with configured parameters per variation 21
Table 3 – Final error values (mse and mae) per variate after training over 50 epochs 26
Table 4 – Lowest error value and related epoch per variate 28
Table 5 – Error values (mse and mae) for variate E at lowest error epoch per heart phase 29
Table 6 – Comparison of accuracy by reviewing aaFD for various network architectures from various studies 32

Motivation

A key driver for this research is the investigation and potentially proposition of alternative neural network architectures to aid the accuracy and performance of self-driven heart assessments. A self-driven heart assessment includes three stages: acquisition, analysis, and interpretation [1].

Figure 1 – Stages in a self-driven heart assessment

The acquisition stage relates to echocardiography (echo) scans that provide information for analysis. Echo scans output images of the heart and vessels using sound waves, typically used for heart assessments [2]. The analysis is the focal point of an automated self-driven heart assessment, with machine learning aiding in various steps in the analysis stage. The analysis completion will trigger a possible patient diagnosis. See Figure 2 for a visualised breakdown of the process.

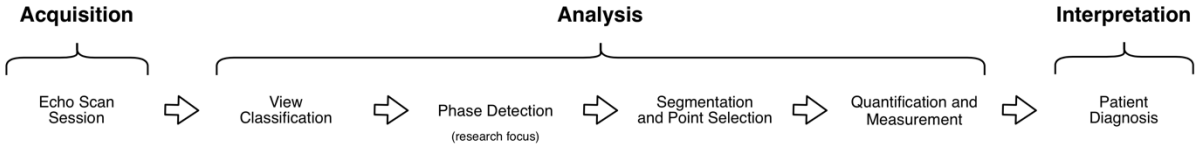

Figure 2 – Detailed stages within a self-driven heart assessment

A standalone echocardiogram (echo) can help identify damage from a heart attack, heart failure, congenital heart disease, problems with the heart valves, and cardiomyopathy [3]. The previously mentioned automated heart assessment focuses on heart phase detection in an echo scan. The subject of this research is the find the time step of end-diastolic (ED) and end-systolic (ES) heart phases within an echo scan. The analysis of ED and ES heart phases could provide information on ejection fraction (EF). EF references the amount of blood pumped from the left ventricle per heart contraction [4] and is one of the various aspects that identifying ED and ES heart phases can help with heart assessment.

Literature

Survey

In 2014 conference proceedings, a study documents the application of Artificial Neural Networks (ANN) in echocardiography imaging [5]. Sundaramurthy et al. detailed their efforts to create a neural network that recognises the two main stages of the heart (diastole and systole) and consisted of three parts: pre-processing, feature extraction, and classification. The pre-processing handled the image input, denoising, binarisation and edge detection. Feature extraction focused on extraction of useful information within the data by using a wavelet decomposition. The last section comprises of a neural network that

predicts the heart stage in a binary classification, either diastole or systole. Their neural network has a single hidden layer and holistically managed to achieve 97% prediction accuracy. The study provides insight that providing information for future studies, the concepts used focus on more general views, predicting the general cardiac cycle phases. While there are applications for the technology, the focus on my research centres in specific end-diastole and end-systole heart phases. Additionally, publicly available convolutional neural networks like ResNet and InceptionV3 consist of many filters, a portion of which may be edge-detecting, whereas this study includes edge detection as the main characteristic. A problem could occur regarding the sensitivity of the model when predicting on echo scans from, e.g., slightly offset camera angles, where the specific edges may differ. CNNs like InceptionV3 include filter for detecting general shapes which is potentially important for more generalised detection. Another concern regards the dataset origins. The study's model could potentially have issues transferring that accuracy on cine scans recorded from different equipment as the entire dataset concentrates from Apollo Hospital Madurai in India which may use the same machines.

A study published in October 2016 focused on the specific end-diastole and end-systole heart phases. The subject matter was to automate the entire process where other implementations may include partial user involvement. The study features the TempReg-Net framework combining convolutional and recurrent model structures to form a completely automatic ED and ES detection [6]. Interestingly, the study uses a Long Short-Term (LSTM) recurrent neural network which is used in research to complete sequential problems. State-of-the-art results are still achieved with the LSTM methodology that attempts to control the problem of the vanishing gradient; a key advantage compared to more primitive recurrent neural networks. Spatial feature encoding (CNN) was used for decoding temporally (LSTM). The CNN uses several layers to convolve the image which concludes in a fully connected layer. The use of the LSTM provides the ability to recognise short- and long-term patterns in the sequence problem, which applies to phase detection in cine scan. The ability of order dependency in LSTMs, in addition to feature extraction, enables an average frame difference of 0.38 and 0.44 for ED and ES respectively. Segmentation-based approaches compared in the study perform worse. Nevertheless, the model is trained on MRI images which has a larger footprint than an echocardiography machine. The detail provided from MRI is more significant but requires a large setup for when conducting the imaging for each person. Various requirements are expected for the user to abide by. NHS states "in some cases, you may be asked not to eat or drink anything for up to 4 hours before the scan, and sometimes you may be asked to drink a fairly large amount of water beforehand" [3]. Additionally, "as the MRI scanner produces strong magnetic fields, it's important to remove any metal objects from your body" [3], meaning objects like jewellery and dentures need to be accounted for. The MRI process causes significant difficulty when considering mass and frequent testing when observing the goal of automated heart assessments.

In a 2017 study by Dezaki et al. [7], an already established network named ResNet, which was combined with two stacked LSTM blocks, was used to handle phase detection on echocardiogram imaging. The ability to transfer a networks capability into another domain is called transfer learning. ResNet can be used as feature extraction by utilising already established weights. The methodology

provides a standardised and easy-to-implement system for building up model components in various problem domains. ResNet, along with others, have an implementation built into frameworks like Keras/TensorFlow. During testing, the model managed aaFD accuracy of 3.7 and 4.1 for ED and ES frames, respectively. The research managed to mitigate the issues with the study in the conference proceedings in 2014 by Sundaramurthy et al [5]. The previously discussed study may have trained on a single machine which may present some prediction abnormalities when presented testing data from different machines. The dataset by Dezaki et al. was responsibly sourced, having clearance by the Clinical Medical Research Ethics Board of the Vancouver Coastal Health (VCH), and consisted of samples obtained by different probes from various manufacturers.

In 2018, a research project evaluated three different machine learning models. The project used CNN, ResNet, and 3D CNN image recognition models [8]. An LSTM network is utilised for considering the temporal aspect. There were comprehensive techniques used to deter the model from overfitting. One strategy detailed in the article was the subtle but purposeful rotation and crop of the input image, allowing the model to learn the general aspects instead of the specific characteristics of the training dataset. The results of all models are compared on a validation dataset. The mean absolute error (mae) in the frames mark the average absolute frame difference (aaFD) for the two main heart stages (end-diastole and end-systole) when comparing the labelled and predicted ES and ED frames. The 3D CNN and LSTM model performed notably better than the other models.

A 2019 research focused on the comparison between various spatial and temporal network structures. The most performant network, which also included a newly proposed loss global extrema function, achieved aaFD of 0.20±0.67 and 1.34±1.17 for ED and ES phases, respectively [9]. The network consisted of CNN and RNN segments, i.e., DenseNet and 2-GRU. It was apparent that the error in ES frame localisation was regularly higher than the ED counterpart. The study includes a clinical explanation for this observation. The hypothesis states that there's a more visible characteristic regarding the closure of the mitral valve contributing to a more accurate ED prediction.

All discussed research bears an issue that a study published in 2021 attempts to circumvent through the use of sliding window methodology, enabling multi-beat phase detection [10]. All studies until now framed the problem in a fashion where each sample is expected to contain one ED and ES annotation. A sliding window approach permits the sample to include variation in the output. In the case of ES Lane et al. implementation, the sliding window covers the amount of timesteps to include either none, or an ED or ES frame. Overlapping individual predictions are patched together and averaged to result in a continuous varied length annotated cine scan. The study achieves an aaFD of 2.3 and 3.49 with a training dataset of 1000 samples with varying cine scan dimensions and frame rates (PACS-dataset). The diversity in the video properties of the training dataset is an advantage. However, only one machine is used to take the echo scan. The study has an advance on this concern by including a testing samples spanning three datasets: PACS, MultiBeat, and EchoNet, which use mixed equipment.

Evaluation

The aspect of no standardised dataset prompts the issue in the comparison of performance. While studies focus to provide metric for comparison (e.g., aaFD), differing datasets for each of the tested models trigger the issue of unfair testing as more than one variables differ. In the case of phase detection, various models are subject to comparison, yet they are tested on varying datasets. The variation of datasets could skew results as some samples may be more predictable than others. Even if studies use the same dataset, there is no guarantee that training, validation and testing subset include the same samples.

This study attempts to highlight the differences in temporal network variation. To ensure a fair test for equal comparison, all variates use the same data for training, validation, and testing after feature extraction.

Methodology

To ready the network for real-world usage, a subsection of the dataset will be used for training. The model will determine what features are important in the cine scan by supplying cine scans with annotations. After the model training, an unlabelled cine scan can be run through the model to predict ED and ES frames for unseen data. The structure of the model is demonstrated in Figure 3.

Dataset Considerations

The EchoNet dataset consists of more than 10,000 samples from patient visiting the Stanford University Hospital in California. ED and ES markers are available per echo sample which are annotated by qualified medical staff. The dataset is sufficiently large and consists of TTE imaging, a relatively common and easier-to-perform type of echo scan.

The echocardiogram types include the transthoracic echocardiogram (TTE), transoesophageal echocardiogram (TOE), and several others [3]. The predominantly used echo scan type for heart phase detection is TTE. TOE provides a more comprehensive view of the heart [11] but requires additional equipment. Fewer available TOE scan datasets stem from the initial drawback but impact the usage of TOE scans in data science. When training ML models, more available data results in the scientist having higher flexibility over the design and training process. Datasets containing a bundled recording of an echo scan and electrocardiogram (ECG) are available. While more information per sample would be preferred, there are two aspects to consider. Firstly, fewer datasets include both recordings, impacting the model design and training flexibility, as discussed previously. Secondly, for real-world application, there is a benefit when considering fewer pieces of equipment. Comparing a model trained on only echo data with a model that used echo and ECG recordings, the former is preferred due to less required setup, primarily if the results have equal accuracy.

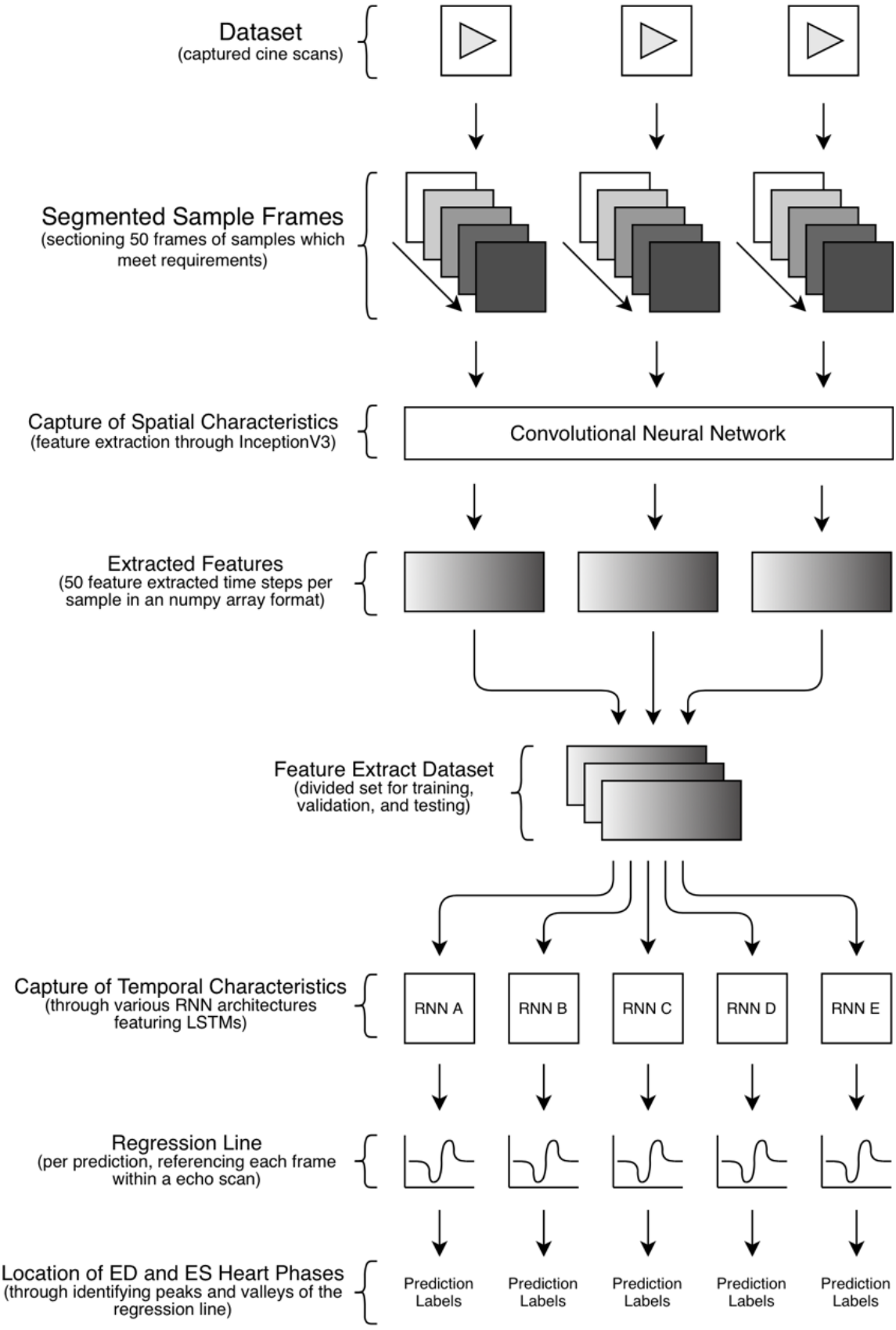

Figure 3 – Holistic model view featuring dataset preprocessing, spatiotemporal feature extraction with devitations of recurrent neural network architectures, and label generation from regression output

Sample Considerations

The EchoNet dataset contained samples of various lengths, and only one ED and ES heartbeat phase is annotated per sample, even if containing more than one beat. Additionally, the dataset has some inconsistencies, where some annotations were spread throughput multiple heartbeats. When filtering for more consistent samples, more refined but less diverse dataset is established. The filtering performed in this study selected samples that met a set of requirements:

- the ES follows the ED
- no more than 25 timesteps between the ED and ES heart phases
- at least 25 timesteps must be available before and after the ED and ES heart phases, respectively

The requirements cut the dataset, removing 6,025 samples which results in a new working dataset size of 4,005 from 10,030.

Classification Type

Initially, the problem was framed as a classification problem, labelling the heart phases with 0, 1, and 2 for regular, ED, and ES phases, respectively. Hot-one encoding allocates a value for each frame and is compatible with frame classification in phase detection. Table 1 shows a demonstration of a hot-ones encoding applied to a phase detection problem.

Table 1 – Hot-ones encoding of an echo scan

Frame	None	ED	ES
1	1	0	0
2	0	1	0
3	1	0	0
…	…	…	…
n	0	0	1

However, the hot-ones encoding style may have questionable learning ability when considering that few frames within an echo scan have an ED and ES frame, resulting in all but two frames classified as 0. The learning process may be skewed when judging its accuracy for model weight updates. Suppose a model learns to classify all frames as a singular value. In that case, if only a couple of values are incorrectly classified, the model's accuracy is $n-2$, where n is the frames in a series. Let us explore this situation for an echo scan. A model statically classifying all frames as standard, i.e., not ED or ES, in a cine scan of 50 frames will result in 96% accuracy. Significant model tuning would be required to adjust the learning function to update the weight adequately.

Alternatively, regression can establish a gradual relation between the ED and ES phases, recognising the structure of the data, i.e., an ES follows the ED phase. The model benefits from gradual changes in heart phase labels, potentially recognising the dependencies before, during, or after a phase. Regression allows classification of frames/segments in a continuous linear or non-linear approach. The methodology used in this research maps the ED phase to 0 and the ES phase to 1, interpolating between each phase. The beginning and the end of the sample arise to a value of 0.5. Figure 4 shows how a regression problem may be framed.

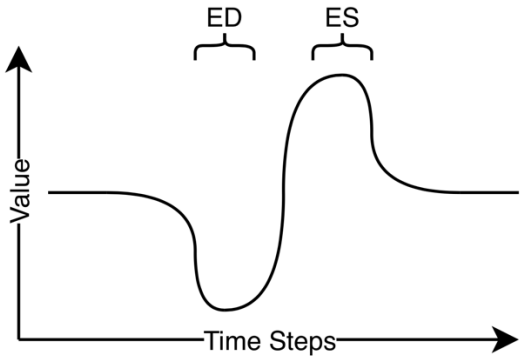

Figure 4 – Framing echo scan labelling as regression with core value of 0.5 having interpolation between ED and ES with 0 and 1, respectively.

The model's output provides an array of values which are mapped to a regression line, as seen in Figure 5.

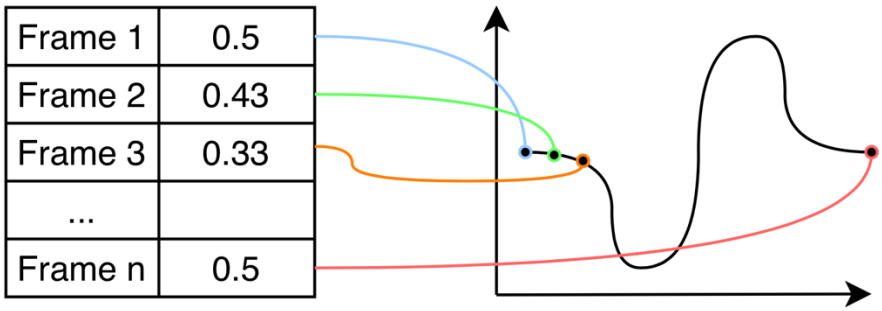

Figure 5 – Echo scan labelling represented as a raw table showing relation to a regression line format

Sample Formation

The construction of each sample provides a predictable structure, i.e., ED followed by ES, in addition to all previously mentioned requirements, e.g., minimum length, maximum ED and ES difference, and more. The model will take samples of length 50, and with EchoNet providing only one ED and ES beat annotation per sample, the section of the sample used for training the model will be the beat with the ED and ES annotation. The duration capturing the ED to ES transition must be up to 25 timesteps, leaving at least 25 timesteps to match the fixed length of 50 (Figure 6).

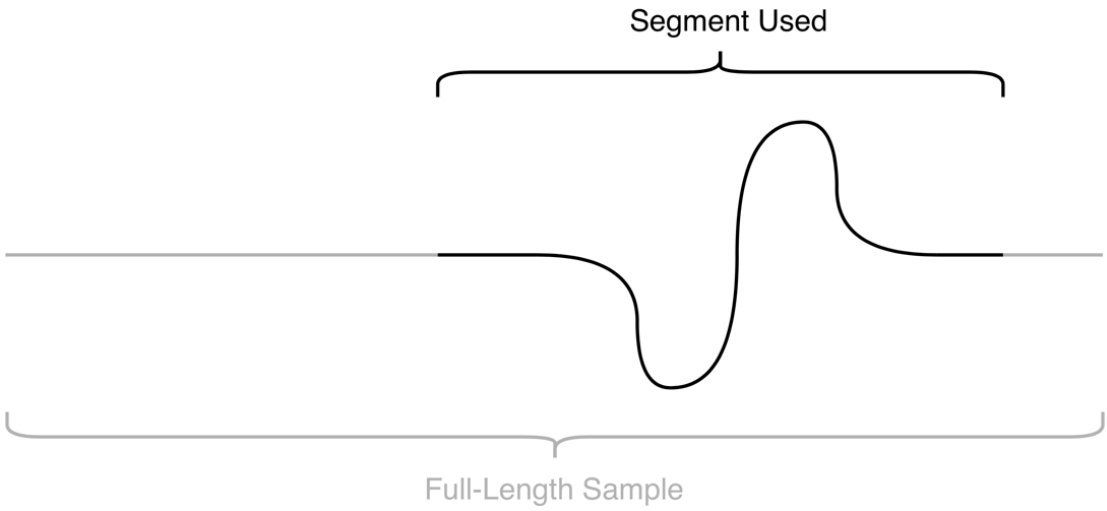

Figure 6 – Segmentation of a full-length sample to capture ED and ES phases in a regression view

To form a sample, one method may include initially start with the ED heart phase, letting the remaining 50 timesteps account for the transition from ED to ES and extra after the fact. While feasible, the model would primarily be learning intricacies of the ES heart phase, not considering what leads to the ED phase. See Figure 7 for visualisation of the problem.

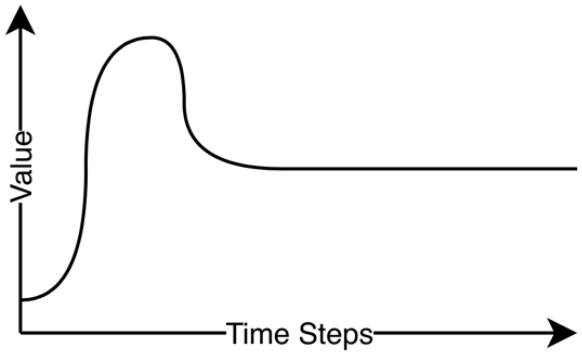

Figure 7 – Regression view of heart phases with initial timestep referencing an ED phase

There is a fundamental disadvantage to above mentioned method. For instance, by inputting echo scans for prediction, the model would only provide theoretically beneficial results if the very start of the sequence was an ED phase, essentially only predicting where the ES heart phase is within the sample.

Alternatively, randomising the spacing before the ED and after the ES heart phases creates a model more suitable for real-world applications, where inputting the echo scan for prediction does not require the ED heart phase to be found. If entirely random, the value pre-ED, and post-ES may be equal to 0. Therefore, a minimum timestep buffer is required before and after the timesteps correlating to ED and ES phases. The implementation controlled the stochastic nature by fixing the ED and ES buffer per sample for each model architecture variation: spatially feature extracted data are reused per model variation.

Capturing Spatial Characteristics

A convolutional neural network (CNN) provides the ability to extract spatial feature. CNNs in the machine learning space are widespread, which can be manually established and trained for specific problems. Manually training CNNs requires significant data to be available during the process, which may not be the same as the data used for the network holistically. For phase detection, manually forming a CNN would require a significant amount of time and resources, including designing layers and gathering labelled data. Alternatively, using an developed CNN like InceptionV3, the network is established, and available weights enable data scientists to use an 'off-the-shelf' solution. Using an already established network and weights for another problem domain is referred to as transfer learning [12].

"The convolutional neural network, or CNN for short, is a specialised type of neural network model designed for working with two-dimensional image data, although they can be used with one-dimensional and three-dimensional data." [13]. As previously discussed, the input of the model consists of echo scans processed on a frame-by-frame basis. Like most digital frames, the two dimensions reference the width and height, as seen in Figure 8.

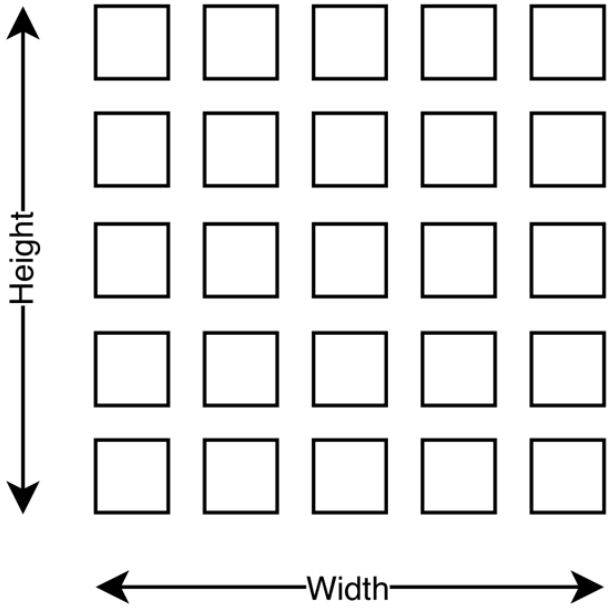

Figure 8 – Deconstruction of an image showing pixels

Within the digital image, pixels consist of various values that determine the colour and shade. Like typical digital images, pixels in the echo scan detail three values: red, green, blue. Combining the width and height with the colour work will form a three-dimensional object. Figure 9 exhibits a holistic view of a single frame that includes the width, height, and colour.

When comparing a CNN with less complex artificial neural networks (ANN), the most significant difference involves the lack of feature extraction done by a user. The work previously needed to 'ready' the data for processing is now

handled by the neural network. See Figure 10 for a representation of a fundamental ANN compared to a CNN.

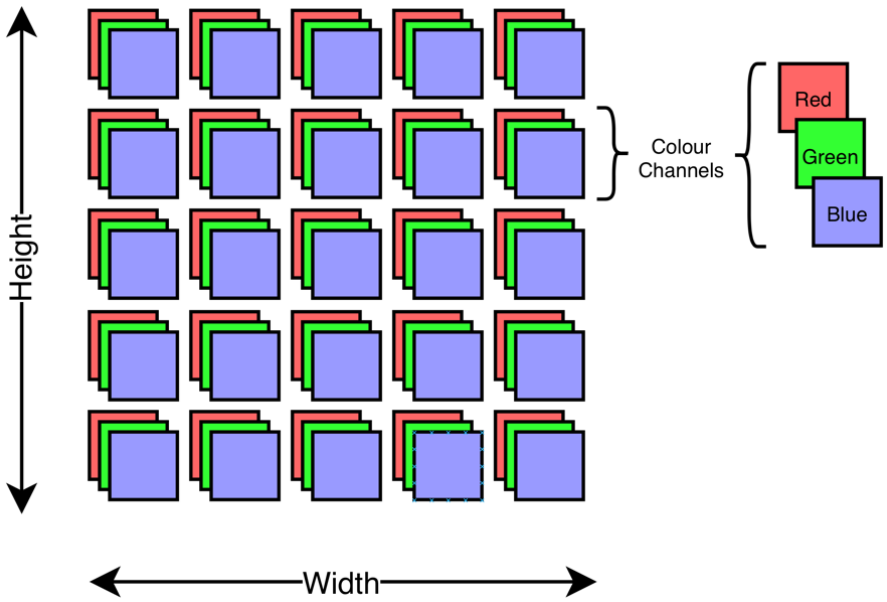

Figure 9 – Deconstruction of an image showing pixels with three colour channels: red, green, and blue

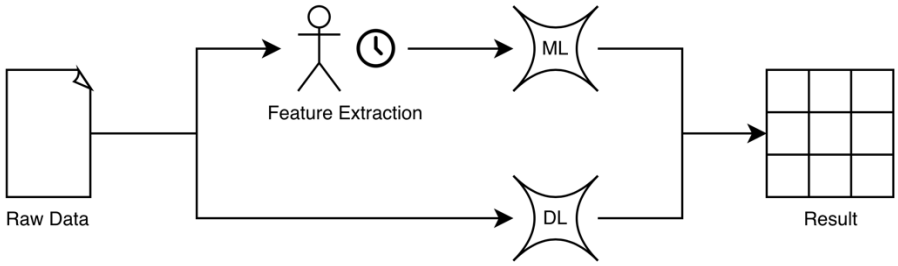

Figure 10 – Two implementations of using raw data to produce a result where processes differ in the responsibility of feature extraction

While the ML and DL terms are used flexibly in the community and may differ between people, it is important to recognise that the method used in this study doesn't require any user handling, resulting in a prediction process where a video is fed and does not need any extra processing by the user. Alternatively, feature extraction done by the user could comprise of annotating heart valves, and their sizes throughout the echo scan, without explicitly providing heart phase annotations.

"Central to the convolutional neural network is the convolutional layer that gives the network its name. This layer performs an operation called a convolution" [13]. The convolution process convolves together the filter and image, i.e., "slide over the image spatially, computing dot products." [14]. "A dot product is the element-wise multiplication between the filter-sized patch of the input and filter, which is then summed, always resulting in a single value." [13].

After completing a run of convolution, in addition to possible deconvolutions, networks will specialise in detecting patterns through the use of filtering. A collection of filters will allow the network to detect an edge or shape, for example. By using the filters for identifying if edge or shape are present, the model can predict if those edges and shapes match with its 'previous understanding' of the combination of features (e.g., edges, shapes, and more). The model's previous understanding is founded by training the model.

Using an established CNN like InceptionV3 with already trained weights, the data scientist can use the CNN for classification predictions. Alternatively, using the CNN to extract feature only (not classifying the image) enables data scientists to use the very complex and already establish CNN filters for another use, a process commonly referred to as *transfer learning*.

This study will utilise the concept of transfer learning with the InceptionV3 network for feature extraction. The CNN used in this research is the InceptionV3 but is one of the many CNN available. A more direct comparison is achievable by focusing on the differences when adjusting the temporal aspect alone instead of the tempo-spatial model characteristic.

Capturing Temporal Characteristics

The model is exposed to three sample sets in this research: training, validation, and testing. Within a training, validation, or testing set, the model should not observe the order. In other words, from the available dataset, the order of samples picked does not matter (Figure 11). However, each sample will contain frames whose order is of value to the model. A heartbeat is not randomly interpolating through various phases. Instead, a particular order is observable, i.e., the end-diastolic happens before end-systolic, classifying the problem of phase detection as sequential in nature.

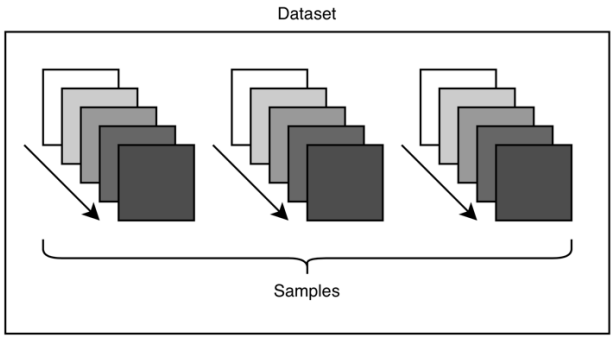

Figure 11 – Diagram containting an non-order dependant sample set with order depandent frames for each samples

Sequence Prediction Type

Sequence prediction problems have four variations. Within sequential problems, the amount in the types ("one" or "many") references the timesteps, e.g., frames of the echo scan. For instance, a many-to-many problem will take multiple inputs and produce multiple outputs.

1. One-to-One

2. One-to-Many
3. Many-to-One
4. Many-to-Many

Before classifying the phase detection problem relevant in this research, let us review how should the output be constructed to be helpful. A previously discussed arrangement states that the ED and ES frames will be 0 and 1, with interpolation present between, after and before each heart phase. See Figure 12 to view a visualised concept of input mapping within the phase detection scope which results in a singular output.

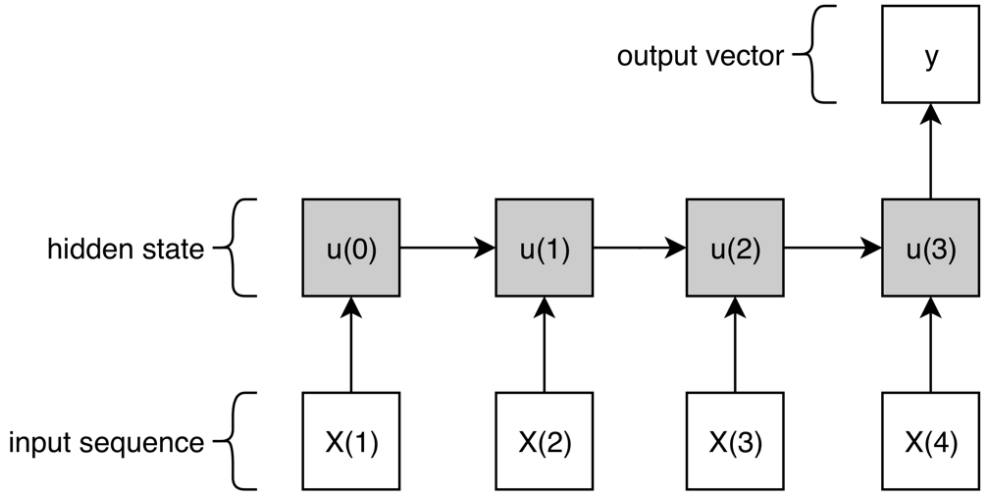

Figure 12 – Network structure consisting of multiple inputs and hidden states to produce an output vector

On brief inspection, many-to-many may satisfy the requirement of mapping a frame to a value from 0 to 1; given a range of inputs, another range will be output. Another approach is one-to-one mapping, where singular input values form an output value. Nevertheless, the strategy used in this research uses the many-to-one where multiple inputs are rated to provide one output. However, that output is a vector in which each element will have relevance to each time step. Figure 13 depicts how the output vector represents labelling for an echo scan sequence.

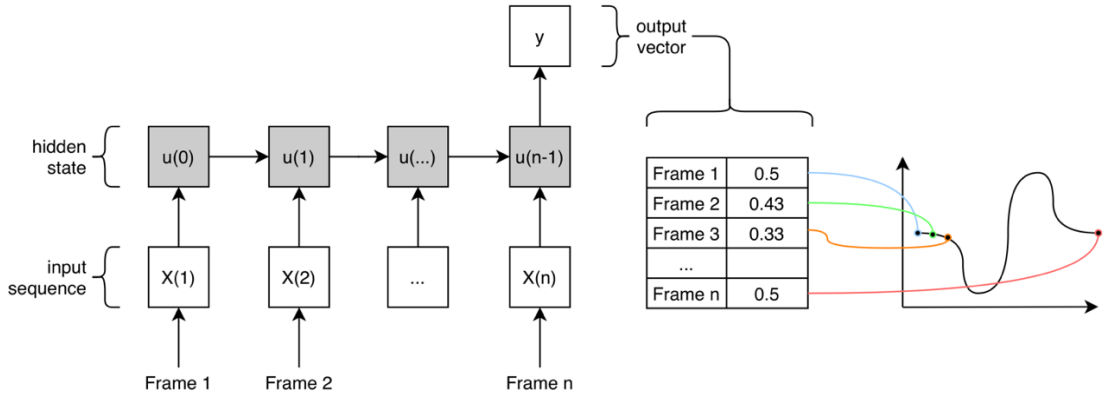

Figure 13 – Showing relationship between the network structure consisting of multiple inputs, hidden states, and an output vector, with the expected output from a phase detection regression problem

Multilayer Perceptron vs Recurrent Neural Network

There are two routes that a data scientist can orient and design a model to suit the problem, either using multilayer perceptrons (MLPs) or recurrent neural networks (RNNs).

MLPs define the input to output at a high level through a mapping function, enabling linear or non-linear approximations [15]. When compared to RNNs, there is a significant difference when considering the stateful capability of the state of cells. MLPs and RNNs are both capable of handling phase detection. Though, the nature of the problem suggests a stateful approach would yield better results. To better compare the potential effectiveness of MLPs and RNNs, let us consider how an MLP would handle a series prediction problem like phase detection.

Within the echo scan context, each frame is an associated time step within a sequential structure. By modelling the frames together as input features, the model will run through the echo scan, individually looking at each frame, primitively providing structure information next stage. The individual process cannot infer any concrete temporal structure, compared to an RNN that holds state per time steps, explicitly changing the state of other memory cells that handle other time steps.

The Long Short-Term Memory (LSTM) is the type of RNN used in this study to capture temporal information. Vanilla RNN has the potential to be used in this research but have the vanishing gradient problem [16]. The problem referencing the "influence of a given input on a hidden layer, and therefore on the network output, either decaying or blowing up exponentially as it cycles around the networks' recurrent connections" [17]. When defining the model, the input and output of the model define the input and output layers, and the hidden layer contains the connections responsible for managing the state. Recurrent connections refer to the cell network, and the user implicitly establishes the hidden layer during model definition.

The memory cells include input and output weights that govern the weight for the current time step (input), extracted information from the previous (output). Additionally, by processing the sample, the model will update the internal state.

The constant error carrousel (CEC) handles the time lags of significant duration, granting help for the vanishing gradient descent problem [18]. "Two gate units learn to open and close access to error flow within each memory's CEC. The multiplicative input gate affords protection of the CEC from perturbation by irrelevant inputs. Likewise, the multiplicative output gate protects other units from perturbation by currently irrelevant memory constants" [18].

Network Training

Due to the weights of a model influencing the prediction process, a set of weights will determine how performant the model is. Updating the model weights enables change in the prediction process, potentially improving the performance, which is iteratively possible per sample. Iteratively updating the model weights by processing multiple samples is called model training.

There are multiple processes performed during training, and the processes used depend on the type of network. Two general concepts form the basis of

relevant training processes: forward passing and backward passing. Fundamentally, during an iteration of training, a prediction is made, and the output can be used as a reference to change the weights appropriately. The initial prediction to get the values is a forward pass, where the network actively predicts a value. Next, the backwards pass can improve model prediction performance, which is possible with the information from the forward pass. The gradient descent algorithm is a type of backward pass.

The term backpropagation encapsulates the forward and backward passes and describes the process of trying to minimise the error to provide a better prediction using gradient descent. Mathematical derivatives establish the basis of gradient descent. The derivatives enable a point of reference within a weight update to help locate what parameters to change to match the ground truth more closely, edging towards a lower error. The ability to reference the actual values during training categorises this as supervised learning.

Figure 14 shows a raw recurrent neural network architecture often viewed within applied machine learning.

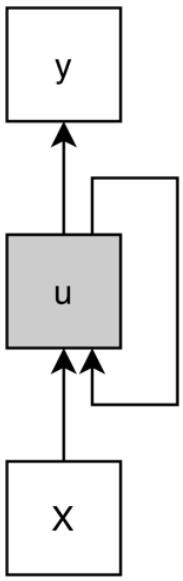

Figure 14 – Folded view of a network

The visualisation displays a form of recursion, but only to convey the concept to the reader. In reality, the network is "unfolded". See Figure 15 for an unfolded network.

In Figure 15, X and y represent the input and output of the RNN. Within the unfolded view, u denote the state that is available to the next hidden layer, and the output becomes available to the input of the RNN handling the next time step. When considering the unfolded network, the input weights remain the same during the prediction process, but the internal state and weights of the output change throughout the sample processing, i.e., attempting to understand the temporal structure. During model training, the input weights updated to minimise the prediction error.

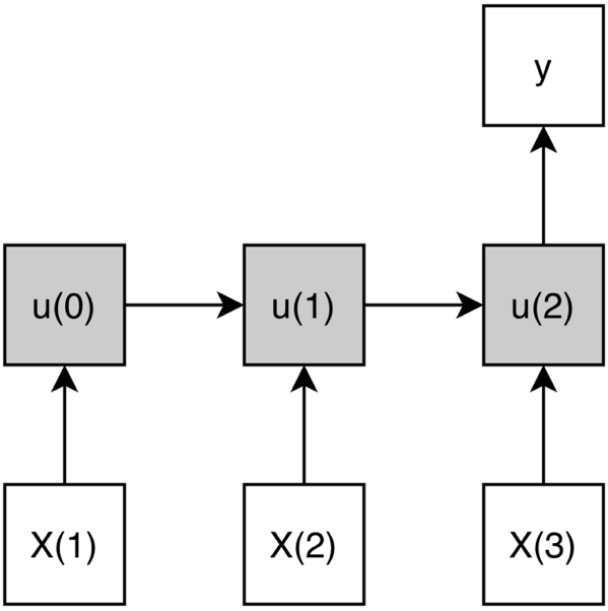

Figure 15 – Unfolded view of a network

Backpropagation Through Time

Backpropagation through time (BPTT) is a training algorithm for recurrent neural networks. BPTT is a backpropagation algorithm, mainly focuses on the order sensitivity rooted in sequential problems.

Finding the error value of a time step requires backpropagation of the previous timestep. In an RNN network, the BPTT applies the chain rule for error calculation similar to the regular backpropagation. Nevertheless, "[RNN BPTTs] loss function depends on the activation of the hidden layer not only through its influence on the output layer but also through its influence on the hidden layer at the next timestep" [19].

With BPTT, each sequence timestep has a copy of the RNN network, potentially problematic for larger sequence sizes. In long sequence problems, changing a parameter can be expensive, yet it serves as the essential aspect of network training. There are also efficiency concerns with parallel computing, i.e., scaling the problem onto multiple processing units [20]. Truncated backpropagation through time (TBPTT) aims to solve the immediate issues with BPTT. The TBPTT usage of Keras (the framework used in this study for LSTMs) forces the data scientist to be prepare data in a specific way to take advantage of the benefits seen in TBPTT, i.e., additional configuration to when the cell state is reset.

For this study, an echo scan of 50 frames will be input into the model to predict the locations of ED and ES frames. Due to the relatively small sequence size, the RNN will utilise untruncated backpropagation through time by using the TBPTT consisting of long short-term memory (LSTM) blocks.

Recurrent Neural Network Variations

The core of this research is to provide insight into various temporal network architectures. Five network types had their performance evaluated through validation and prediction after training named variate A-E. The list below contains a brief definition of variations:

- Variate A includes the most fundamental LSTM structure in this research: one LSTM.
- Variate B adds a second LSTM.
- Variate C is similar to B but includes a bidirectional LSTM first.
- Variate D includes two Bidirectional LSTMs.
- Variate E contains three non-bidirectional LSTMs.

Additional notes:

- For all instances of two or more LSTMs, a dense later is present as middleware.
- Each network type will include a flatten later and dense layer before output.
- All bidirectional LSTMs are in concatenation mode.

Figure 16 visualises the architecture. Table 2 describes the network layers for each variate in detail.

Table 2 – Model structure showing network layers with configured parameters per variation

<table>
<tr><th>Variate A</th><th>Variate B</th><th>Variate C</th><th>Variate D</th><th>Variate E</th></tr>
<tr><td>LSTM
units: 2048</td><td>LSTM
units: 2048</td><td>LSTM
(Bidirectional)
merge mode: concat
units: 2048</td><td>LSTM
(Bidirectional)
merge mode: concat
units: 2048</td><td>LSTM
units: 2048</td></tr>
<tr><td rowspan="4"></td><td>Dense
units: 512
activation: RELU</td><td>Dense
units: 512
activation: RELU</td><td>Dense
units: 512
activation: RELU</td><td>Dense
units: 512
activation: RELU</td></tr>
<tr><td>LSTM
units: 512</td><td>LSTM
units: 512</td><td>LSTM
(Bidirectional)
merge mode: concat
units: 512</td><td>LSTM
units: 512</td></tr>
<tr><td rowspan="2"></td><td rowspan="2"></td><td rowspan="2"></td><td>Dense
units: 256</td></tr>
<tr><td>LSTM
units: 256</td></tr>
<tr><td colspan="5">Flatten</td></tr>
<tr><td colspan="5">Dense
units: 50</td></tr>
</table>

Bidirectional LSTMs use concatenation ('concat') to merge the outputs of the LSTMs where the second LSTM units compute the input in the opposite direction, reversing the input sequence.

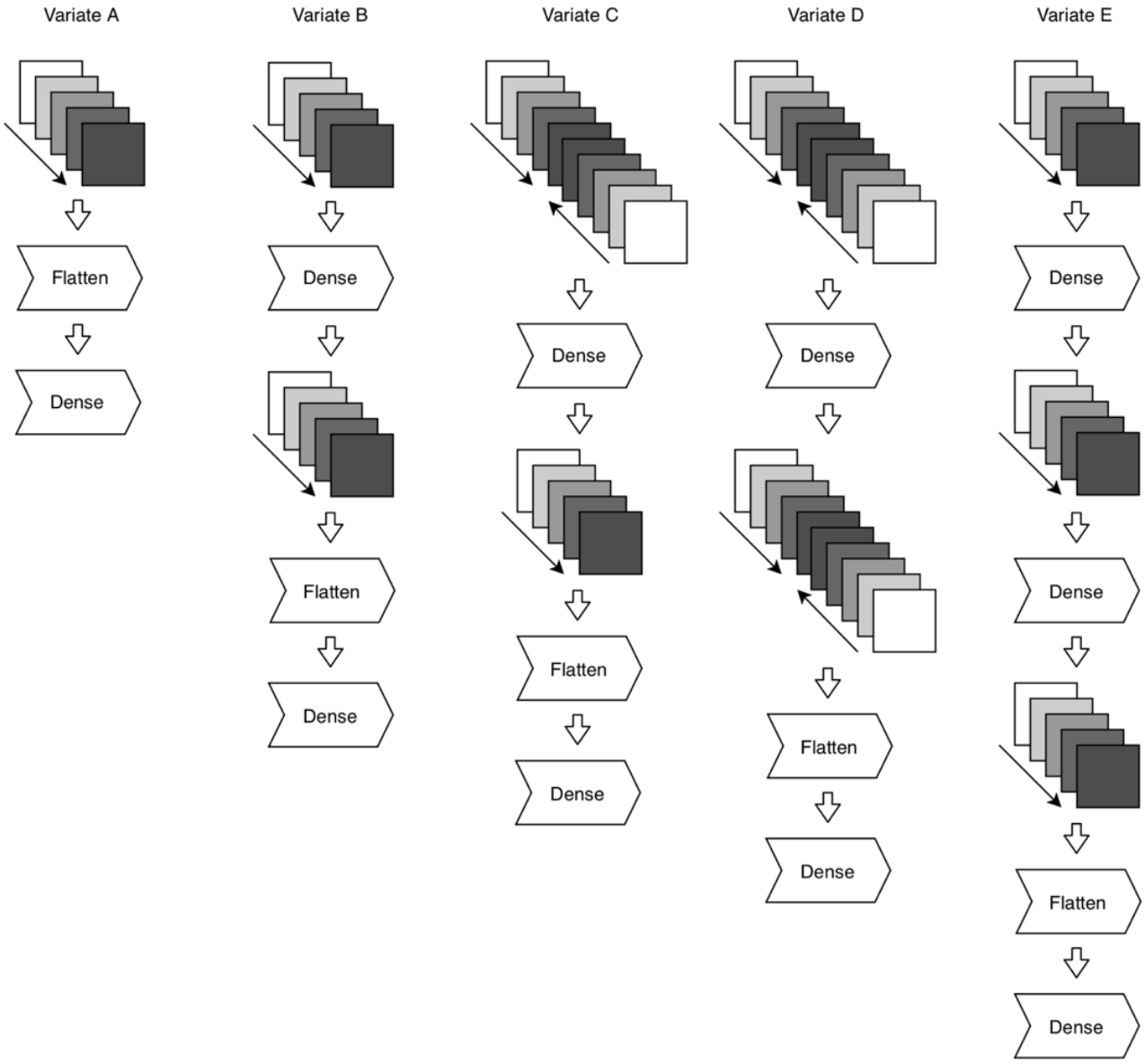

Figure 16 – Model structure showing network layers per variation

Loss and Metrics

To calculate the error of the model during training and otherwise, Keras provides various loss functions. The type of problem is the main factor to consider when deciding on the loss function. Binary or categorical cross-entropy functions may be used as the loss for probabilistic problems, whereas mean or absolute-squared errors could satisfy problems dealing with regression [21]. The framing of the phase detection problem for this study required a regression-based loss function. Mean-squared error is often referred to as mse and provides adequate functionality for the model used in this research. Mean-absolute error, mae for short, is used for the same purpose and can also be used as the loss function. The fundamental basis for both functions consists of pair values comparison and finding the difference, see the equation for a mathematical rendition.

$$mse = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

$$mae = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$$

Using mse as the loss function to find the error by comparing the predicted and ground truth values prompts the model to adjust the model's weights to decrease errors.

The deepest valley and highest peak represent the ED and ES heart phases, respectively. The non-linear representation of the heart phases is suitable for a mse function to compare values and provide the error. Mean-squared error is set as the metric to see the model performance throughout the training process, i.e., the mse display the model's performance per epoch. Notably, the error is not calculated on only the valley and peak location but also surrounding the entire prediction. More information is provided before, between, and after phases for the model to have more temporal references to the spatial data. Alternatively, let us consider a naive rendition of an input model as displayed in Figure 17. If the representation of heart phases consisted of an ED and ES phase referenced as 0 and 1, with only the value 0.5 representing other timesteps, the models learning ability would be similar to a hot-ones encoding.

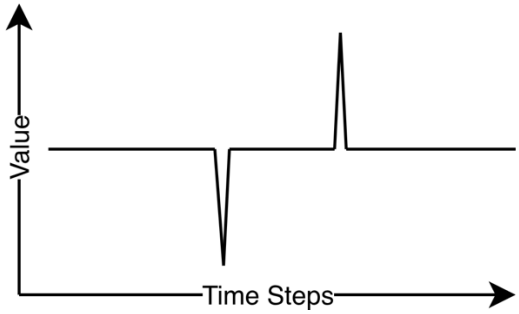

Figure 17 – Naive regression implementation of frame encoding for echo scan

Optimiser

The model uses the Adaptive Moment Estimation (Adam) optimiser, an "algorithm for first-order gradient-based optimisation of stochastic objective functions, based on adaptive estimates of lower-order moments." [21]. The optimiser effectively considers the sets of model weights for each weight update, as pooling through every weight combination will produce poor performance [22].

Gradient descent is when a network attempts to minimise error, sometimes referred to as locating a minima. Depending what trainable parameters are provided to the model, a loss function will determine, within all possible mapping of parameters, what the current cost is, and could provide insight how to adjust parameters to lower the error by computing partial derivatives. A normal gradient descent considers every parameter and datapoint. When handling large number

of parameters and datapoints, stochastic gradient descent may be more efficient while providing similar results.

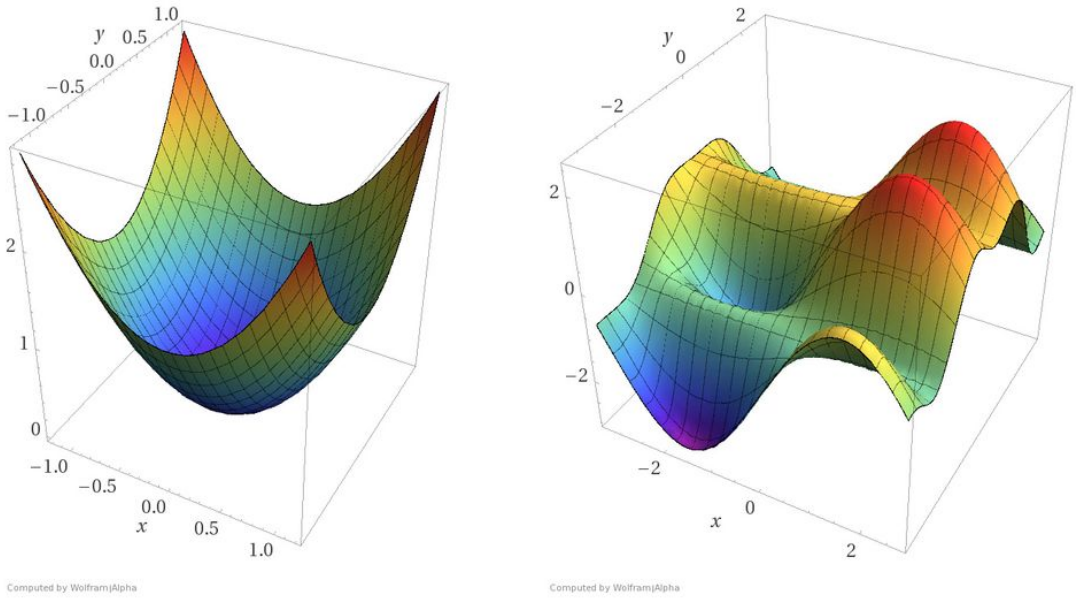

Figure 18 – Loss function mapping of two for two different network instances (source: blog.paperspace.com [23])

In Figure 18, the left mapping has a single point where the gradient descent can focus to reduce the error. However, when considering the mapping on the right, the complex nature of the loss function may result in the model approaching to a local minima, where the error is minimised per step in relation to a segmented area. Global minima describes the lowest error within the entire mapping. Stochastic gradient descent selects one or more samples for computation, instead of all. When considering compute time, the stochastic gradient enables higher efficiency as it is not considering all data points per step in a partial derivative calculation.

A version of the Adam optimiser is used as the network optimiser in this research. Adam is an extension of stochastic gradient descent which allows for faster convergence into a minima through the process of estimation. The Adam optimiser is used from the multi-beat phase detection study by ES Lane et al. [10].

Performance Metric

Comparing expertly annotated ED and ES heart phases against predicted values forms a performance metric used in phase detection research. The average absolute frame difference (aaFD) will evaluate the performance of the various models used in this study. A significant aaFD value suggests the prediction is relatively far off from the ground truth, whereas a small value will suggest a close match.

Converting Prediction Output to Labels

When performing a prediction on a sample, the comparison occurs between the predicted and expertly annotated ED and ES frames. The actual values, also known as ground-truth labels, are readily available from the EchoNet dataset. However, after training, the prediction is output is a non-linear regression. Locating the most significant valley and peak in the regression line forms the basis for labels from the prediction (Figure 19).

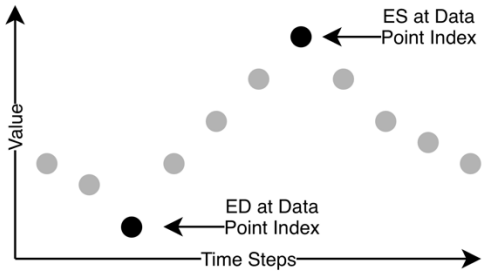

Figure 19 – Data points highlighted which reference the ED and ES phase phases

After predicting the result curves may confine multiple peaks and valleys. Therefore, the Savitzky–Golay digital filter will smooth the curve enough to warrant only one peak and valley per sample (Figure 20). Finding the smallest and largest data point value index after filter smoothing will provide the prediction labels.

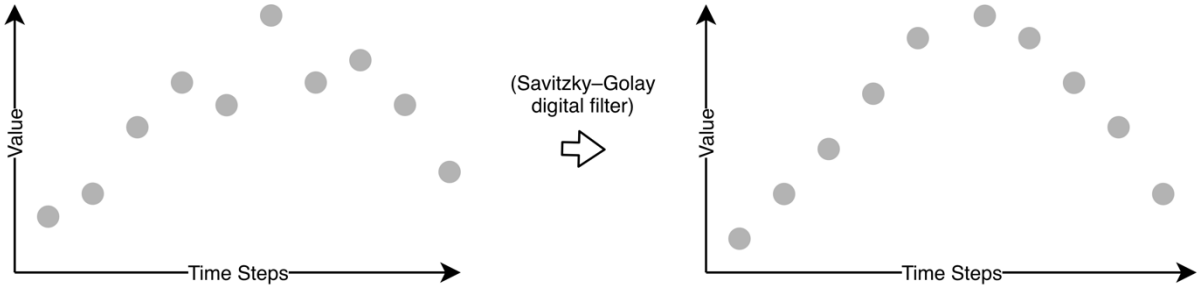

Figure 20 – Before/after of Savizky–Golay digital filter on a staggered dataset

Results

The results are split into two sections: reviewing the accuracy of the model, and the performance. The training process includes 3204 samples read by the model over 50 epoch with a batch size of 4. After each epoch, the model verifies its non-biased performance with a validation set of 600 samples that include not-seen-before model data.

Accuracy

After training, a test dataset of 202 samples provides insight in the form of mse and mae. Figure 21 and Table 3 detail the mse and mae values per variation per ED and ES prediction. In this mse and mae representation, the methodology

consists of comparing the predicted vs actual ED and ES data points, whereas more typical mse and mae are considering every data point along two regression lines. The implementation of mae regarding only ED and ES data points is equivalent to aaFD that is present in other phase detection problems.

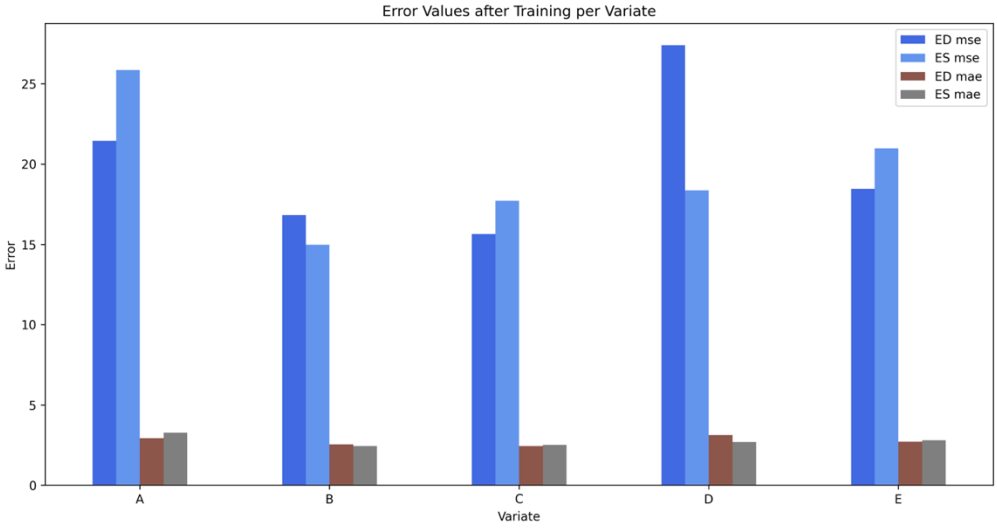

Figure 21 – Final error values (mse and mae) per variate after training over 50 epochs

Table 3 – Final error values (mse and mae) per variate after training over 50 epochs

Variate	Mean-Squared Error		Mean-Absolute Error (aaFD)	
	ED	ES	ED	ES
A	21.44554	25.86633	2.92079	3.26237
B	16.82178	**14.97029**	2.53465	**2.42574**
C	**15.63861**	17.71287	**2.44059**	2.51485
D	27.40099	18.36138	3.12376	2.68811
E	18.46534	20.98019	2.71287	2.79207

However, providing a comprehensive test at the end of the training (50 epochs) may not be adequate. Over-fitting is a critical problem plaguing machine learning models. The model begins to confine in the memorisation of the data instead of providing more general prediction with higher accuracy for unseen data. The implementation of the training process in this research includes collecting training mse and validation mse. When plotting the mse per variate, it is evident that there is a point where the model stopped learning the general characteristics of the data, memorising instead. See Figure 22 for plotted chart. It is important to note that this mean-squared error is of every data point across the regression line, using Keras' loss mse implementation with the default reduction parameter.

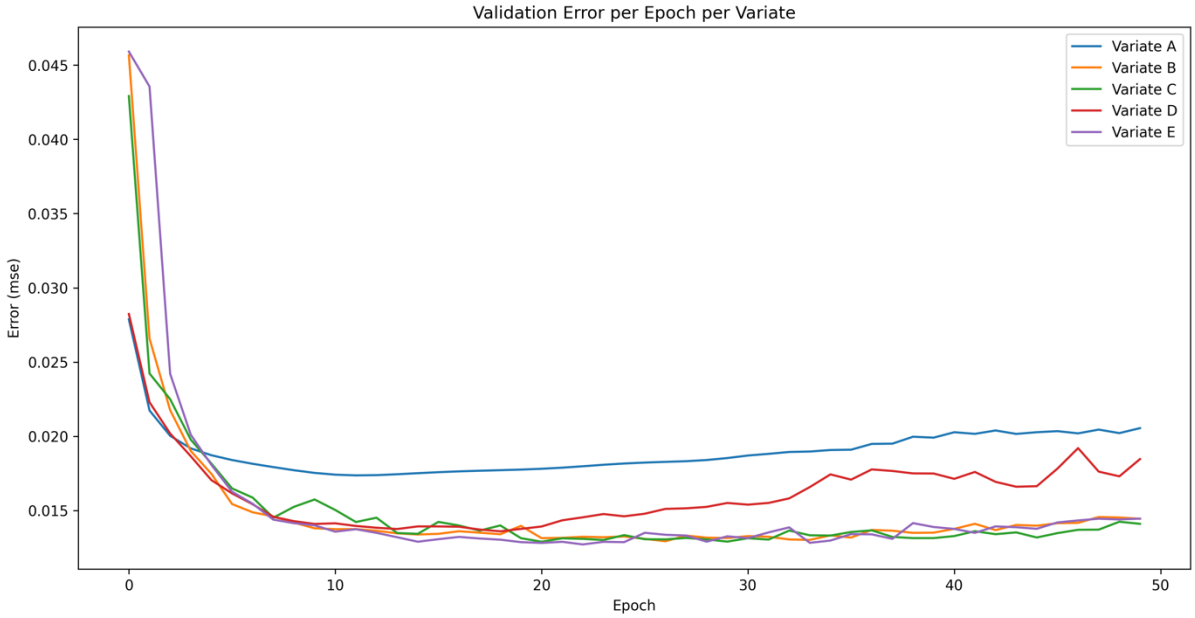

Figure 22 – Error value (mse) for validation data throughout the training process per variate

The overfitting relationship with the training data is an issue most prevalent in variate D (two bidirectional LSTMs). Around epoch 15, the training performance continues to improve, whereas validation error worsens—a sign of overfitting. See Figure 23 for a visualisation of the D variate overfitting with training and validation mse.

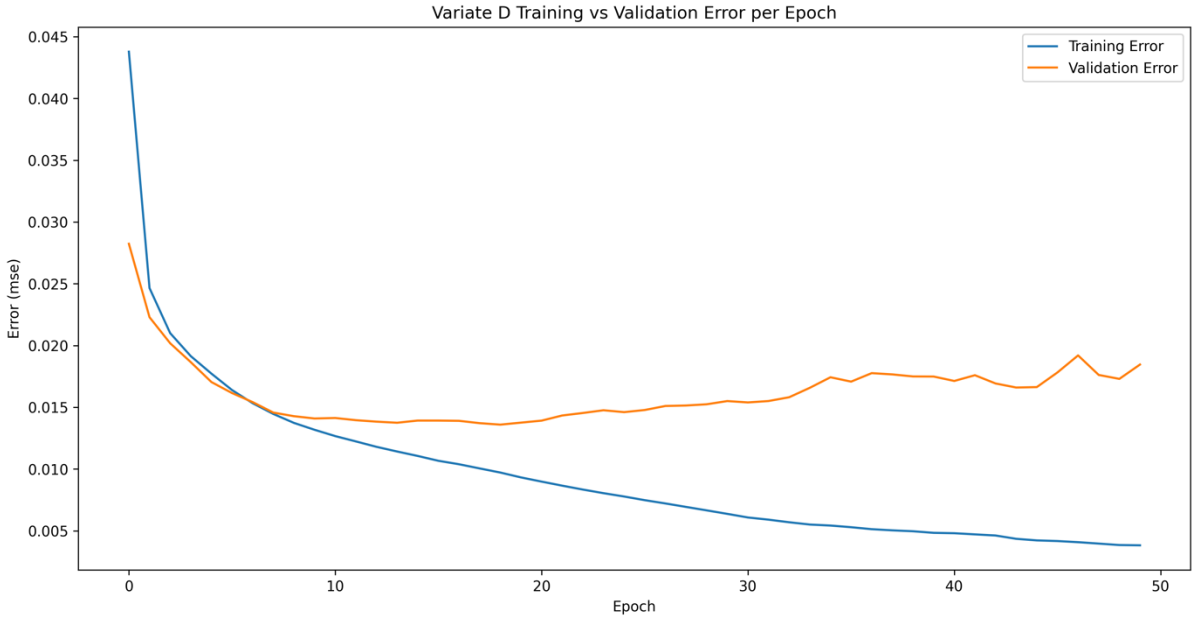

Figure 23 – Error value (mse) for training and validation data throughout the training process for variate E

To finalise the accuracy analysis, let us consider each model at the most accurate mse validation epoch. Figure 24 depicts the validation mse history of all variates with a highlight on each of the lowest points.

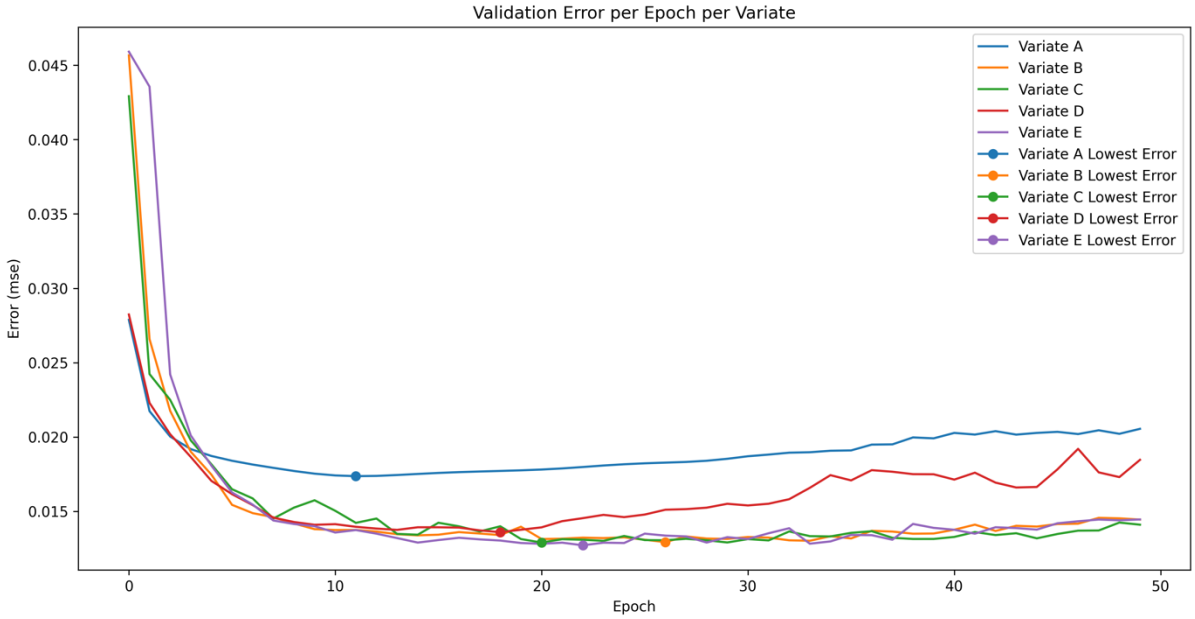

Figure 24 – Error value (mse) for validation data throughout the training process with notes of lowest error per variate

Table 4 documents the epoch and value at the high accuracy (to the fifth decimal point).

Table 4 – Lowest error value and related epoch per variate

Variate	Error Value	Epoch
A	0.01736	11
B	0.01292	26
C	0.01289	20
D	0.01359	18
E	**0.01270**	**22**

Variate E performs the best from all variates in 22 epochs and represents the absolute best possible heart phase detection model version of all tested instances. Table 5 shows the summation of best achieved the mse and mae (aaFD) for ED and ES data points only (up to the fifth decimal point).

Table 5 – Error values (mse and mae) for variate E at lowest error epoch per heart phase

Variate	Mean-Squared Error		Mean-Absolute Error (aaFD)	
	ED	ES	ED	ES
E (22nd epoch)	16.05940	14.86138	2.53465	2.45544

Let's consider the frequency of each errors at the 22nd epoch for variate E. To display the distribution of errors, view Figure 25 and Figure 26 which detail histograms of error per heart phase.

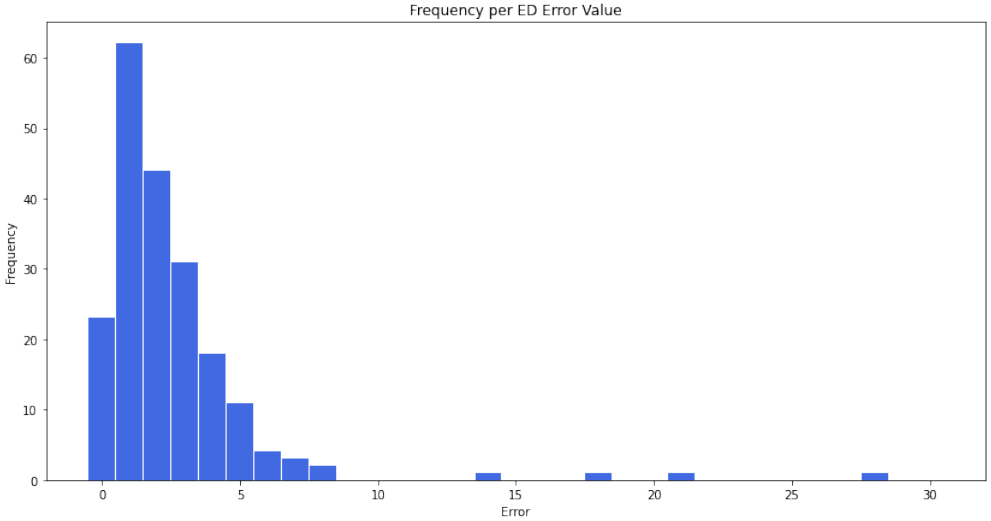

Figure 25 – ED error value (mae) frequency (histogram) for testing data

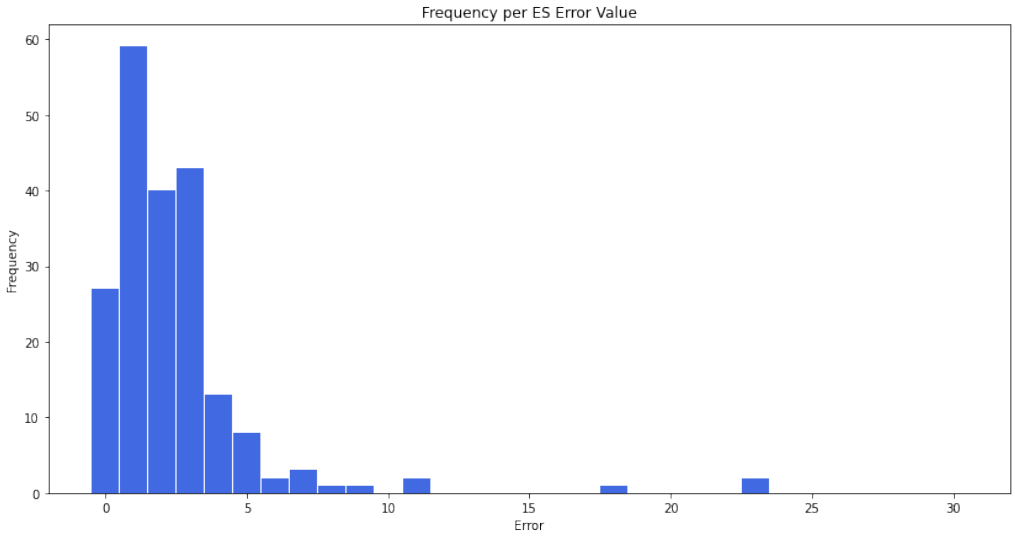

Figure 26 – ES error value (mae) frequency (histogram) for testing data

The frequency of ED and ES errors falls largely within 5, with very few entries for greater values. Both figures show information that is in line with 2.5

and 2.4 mae values for ED and ES, respectively. Interestingly, the model seems to be attempting to learn relevant patterns. An alternatively distribution within the histograms could include a large focus between two values with few entries between. For instance, 90% of all values are either in the 0 or 5, instead of being spread further apart.

Even though the large error values are infrequent, there is a hypothesis for their occurrence. Error values of 20 or high could be predicting the phases heart beats before or after the EchoNet annotation. The window of 50 frames could possibly include multiple heart beats, especially when considering a faster heartbeat. Therefore, it is not unreasonable to speculate that some of the errors of large values could correctly predict phases, just not the same annotation.

The current error rates demonstrate that there is untapped optimisation that could push the accuracy even higher as the core structure prompts the model to learn the data in the right direction, and for the intended objective.

Due to the differences in complexity, the time to complete training varied. If finding the most accurate model under a time restraint environment, the ideal architecture and selected epoch might deviate from the conclusion discussed in this study.

Efficiency

Regarding efficiency, there are a few artefacts to consider. Each training and testing process utilised the same computer with the same specification. From this observation, a direct comparison of the milliseconds per step, and time to complete training provide a type of complexity-per-architecture attribute.

The following discussion stems from 50 epochs of 3204 samples with a batch size of 4. The validation and testing sample sizes are 600 and 202, respectively. The graphics card used was a reference NVIDIA 980 Ti with 2816 CUDA cores (v5.2) which was connected through PCIe 3.0 16x slot on an Z170 motherboard with an Intel 6700k and 16GB of 2133MHz RAM. Upon reviewing the time to complete training and milliseconds per steps, some key factors are present.

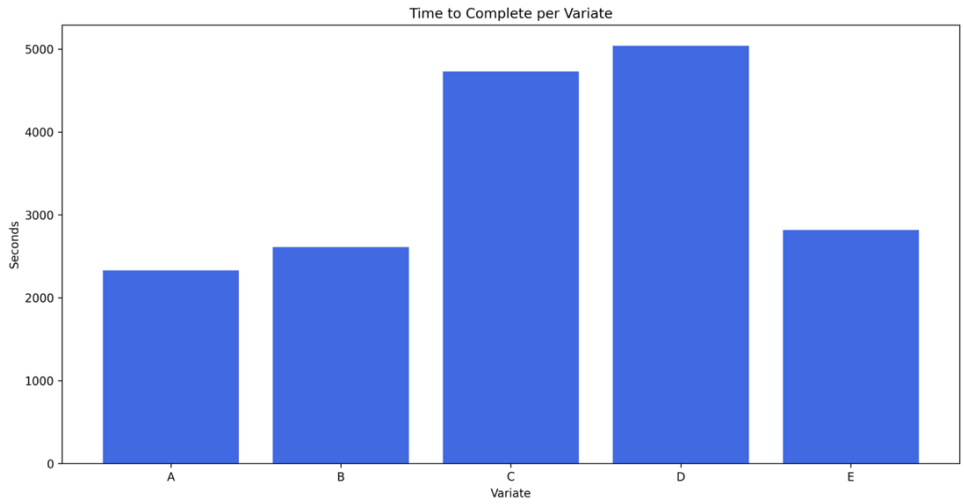

Figure 27 – Time to complete training of 50 epoch in seconds per variate

When viewing Figure 27, the variates A, B and E form a linear increase in the relative complexity as each adds a non-bidirectional LSTM. Thus, a minor increase in needed performance is required even when adding a long short-term memory network layer. Alternatively, when viewing networks with bidirectional layers, there is a significant jump in required computation. Notice the interesting characteristic that most complex non-bidirectional LSTM architecture (variate E) cuts computation by nearly half. However, when considering the addition of a second bidirectional LSTM, the compute time is barely noticeable.

Compiling the model reveals the number of trainable parameters. Let us plot this data against the time taken for training and the milliseconds per step, as seen in Figure 28.

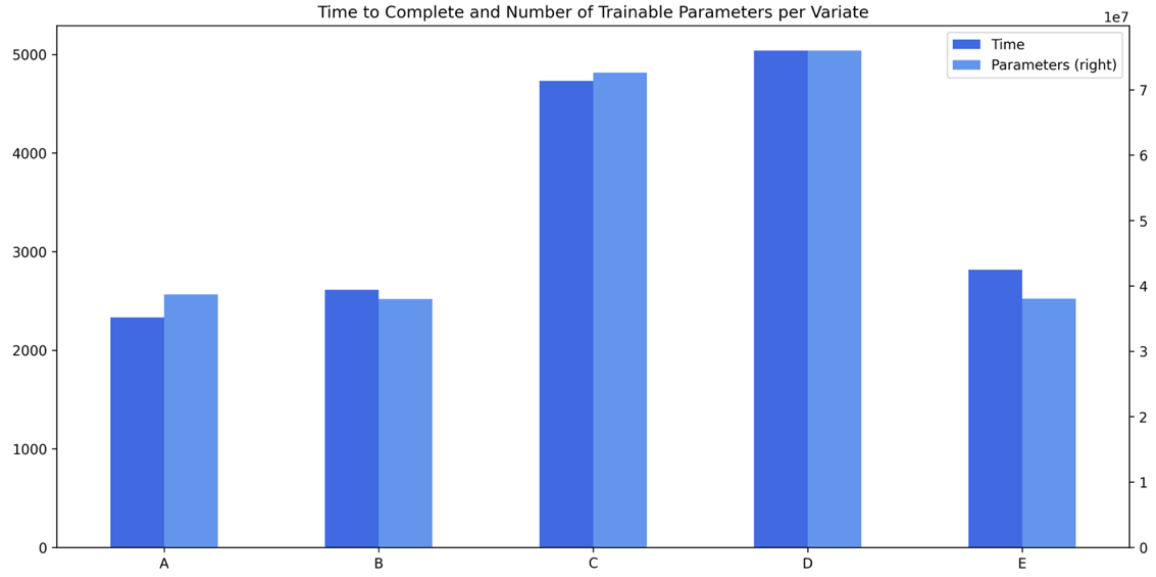

Figure 28 – Time to complete training of 50 epochs in seconds against number of trainable parameters per variate

The bar chart evidences a clear correlation between the time to complete training, and the number of trainable parameters, explaining the characteristic of increased computation required in bidirectional networks.

Conclusion

Model Performance

Table 6 contains accuracy values for various fixed-length phase detection implementations. Variate E at the 22nd epoch outperforms some ResNet and CNN phase detection implementations. However, the more advanced feature extraction on a three-dimensional CNN network achieves lower aaFD. Variate E featuring three stacked LSTMs outperforms variate B with two stacked LSTMs. Higher accuracy is achieved through the addition of another LSTM. When considering the MultiBeat study by ES Lane et al. [10], there may be capacity for even higher accuracy when adding another LSTM, resulting in potentially higher accuracy

than aaFD of 0.66 and 0.81 found. Computation-wise, the performance would come with minimal added training cost.

Table 6 – Comparison of accuracy by reviewing aaFD for various network architectures from various studies

Method	Mean-Absolute Error (aaFD)	
	ED	ES
CNN + LSTM + L_2 loss [6]	6.3	7.3
CNN + LSTM + L_2 loss + structured loss [6]	6.4	7.3
ResNet + LSTM + L_2 loss [7]	4.4	4.7
ResNet + LSTM + L_2 loss + structured loss [7]	3.7	4.1
InceptionV3 + LSTM + mse loss + Adam optimiser (variate E at 22 epoch)	2.5	2.4
3D CNN + LSTM + cross-entropy loss [8]	**1.6**	**1.7**

The EchoNet dataset used iE33, Sonos, Acuson SC2000, Epiq 5G, or Epiq 7C ultrasound machines for the capturing of cine scans [24]. Various equipment used in capturing cine scans creates positive aspect when considering diversity. However, patient background statistics aren't available for all dataset used to train the methodologies in table [24]. Not being able to validate the dataset causes uncertainty in its application in patients from different ages and ethnic backgrounds, who may exhibit slight differences in the structure of their heart, and more. Researchers found that "the right ventricle is: smaller but pumps harder in older adults, larger in men than women, and smaller in African-Americans and larger in Hispanics, compared with Caucasians." [25]. Integrating researched models in regions where the study takes place poses lower risk when considering the alternative. Research done in phase detection applies in areas where modernised hospitals are elevating some work to machines that was previously done by medical staff. There is reason to believe that phase detection research, among other cardio studies, could be as useful for integration in developing nations, where they may be no medical staff available, for annotating cine scans or otherwise. Citizens of these areas may not be properly represented in the datasets, a complication that could stem into abnormal predictions potentially causing false positives or false negatives in diagnosis. Most dataset mentioned in this study originates from university hospitals in extensively modernised societies (EchoNet dataset origin: Stanford University Hospital, California, United States [24], and MultiBeat and PACS origin: St. Mary's Hospital and Imperial College Healthcare, United Kingdom [10]). When considering implementation of an accurate model, there should be, at least, considerations of the origin of training dataset, and the intended destination.

Preferably, all models should train on diverse enough datasets to account for those who immigrate from less represented countries.

Future Work

The CNN used is a shared variable in each model to ensure a fair comparison, and various other research utilised other CNNs. A possible step forward would be to investigate how CNNs like ResNet, ResNet50, Xception, and NASNet perform when the LSTM elements within those models are changed. A broader scope will increase the direct comparisons that data scientists can make when designing future phase detection models.

The requirements constitute as a substantial refinement of the dataset, cutting more than half of the samples. However, while reducing restrictions may result in a more extensive dataset, the foremost difficultly is the limited ED and ES annotation. Having ED and ES phases annotated in the dataset is essential for training a model for heart-phase detection. Due to the limited annotation on the EchoNet dataset, a different dataset may be more suitable for phase detection. A more appropriate dataset could include a more comprehensive ED and ES phase annotation, marking every phase per heart contraction.

Another next step could be to investigate what curves the regression line could be to maximise the learning ability. Current structure details the ED and ES heart phase time steps in one oscillation. Another instance is used in a study by Dezaki et al. in 2017, where interpolation between ED and ES sections is more sudden, as can be seen in Figure 29.

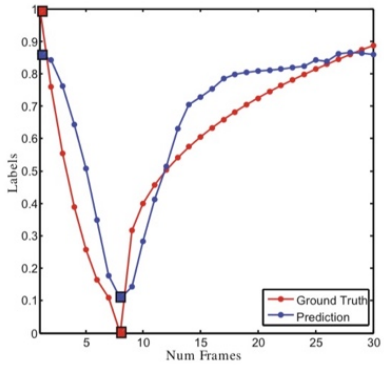

Figure 29 – Alternative regression sample formation with more sudden phase interpolation (source: Dezaki et al. [7])

Availability

The source code for the project takes of the form of Python notebooks which are available on the accompanying GitHub repository, in addition to network weights:
https://github.com/dropcmd/Temporal-Neural-Network-Analysis

Dataset used in this research, EchoNet, is publicly available [24]: https://echonet.github.io/dynamic/

Bibliography

[1] M. Zolgharni, *AI-assisted self-driving heart exam, are we there yet?!,* London: University of West London , 2021.

[2] Texas Heart Institute, "Echocardiography," [Online]. Available: https://www.texasheart.org/heart-health/heart-information-center/topics/echocardiography/. [Accessed 31 May 2021].

[3] NHS, "Echocardiogram," 23 July 2018. [Online]. Available: https://www.nhs.uk/conditions/echocardiogram/. [Accessed 31 May 2021].

[4] heart.org, "Ejection Fraction Heart Failure Measurement," 31 May 2017. [Online]. Available: https://www.heart.org/en/health-topics/heart-failure/diagnosing-heart-failure/ejection-fraction-heart-failure-measurement. [Accessed 31 May 2021].

[5] S. Sundaramurthy, A. Wahi, L. Priyanga Devi and S. Yamuna, "Cardiac Cycle Phase Detection in Echocardiography Images Using ANN," Coimbatore, 2014.

[6] B. Kong, Y. Zhan, M. Shin, T. Denny and S. Zhang, "Recognizing End-Diastole and End-Systole Frames via Deep Temporal Regression Network," *Lecture Notes in Computer Science,* vol. 9902, 2016.

[7] F. T. Dezaki, N. Dhungel, A. H. Abdi, C. Luong, T. Tsang, J. Jue, K. Gin, D. Hawley, R. Rohling and P. Abolmaesumi, "Deep Residual Recurrent Neural Networks for Characterisation of Cardiac Cycle Phase from Echocardiograms," *Lecture Notes in Computer Science,* vol. 10553, 2017.

[8] A. M. Fiorito, A. Østvik, E. Smistad, S. Leclerc, O. Bernard and L. Lovstakken, "Detection of Cardiac Events in Echocardiography Using 3D Convolutional Recurrent Neural Networks," *2018 IEEE International Ultrasonics Symposium (IUS),* 2018.

[9] F. T. Dezaki, Z. Liao, C. Luong, H. Girgis, N. Dhungel, A. H. Abdi, D. Behnami, K. Gin, R. Rohling, P. Abolmaesumi and T. Tsang, "Cardiac Phase Detection in Echocardiograms With Densely Gated Recurrent Neural Networks and Global Extrema Loss," *Transactions on Medical Imaging,* vol. 38, no. 8, pp. 1821-1832, 2019.

[10] E. S. Lane, N. Azarmehr, J. Jevsikov, J. P. Howard, M. J. Shun-shin, G. D. Cole, D. P. Francis and M. Zolgharni, "Multibeat echocardiographic phase detection using deep neural networks," *Computers in Biology and Medicine,* vol. 133, no. 104373, 2021.

[11] A. Ashraf and D. Carroll, "Transoesophageal echocardiography," [Online]. Available: https://radiopaedia.org/articles/transesophageal-echocardiography?lang=gb. [Accessed 31 May 2021].

[12] TensorFlow, "Transfer learning and fine-tuning," Google, 25 March 2021. [Online]. Available: https://www.tensorflow.org/guide/keras/transfer_learning. [Accessed 31 May 2021].

[13] J. Brownlee, "How Do Convolutional Layers Work in Deep Learning Neural Networks?," Machine Learning Mastery, 17 April 2020. [Online]. Available:

https://machinelearningmastery.com/convolutional-layers-for-deep-learning-neural-networks/. [Accessed 31 May 2021].

[14] M. Zolgharni, *CP6CS57E - Machine Learning, Convolutional Neural Network Lecture (Week 11),* London: University of West London, 2020.

[15] G. Dorffner, "Neural Networks for Time Series Processing," *Neural Network World,* vol. 6, pp. 447-468, 1996.

[16] J. Brownlee, "How to Fix the Vanishing Gradients Problem Using the ReLU," Machine Learning Mastery, 25 August 2020. [Online]. Available: https://machinelearningmastery.com/how-to-fix-vanishing-gradients-using-the-rectified-linear-activation-function/. [Accessed 31 May 2021].

[17] A. Graves, M. Liwicki, S. Fernández, R. Bertolami, H. Bunke and J. Schmidhuber, "A Novel Connectionist System for Unconstrained Handwriting Recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 31, no. 5, pp. 855-868, 2009.

[18] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," *Neural Computation,* vol. 9, pp. 1735-1780, 1997.

[19] A. Graves, Supervised Sequence Labelling with Recurrent Neural Networks, vol. 385, Springer, 2012.

[20] H. Kyuyeon and S. Wonyong, "Online Sequence Training of Recurrent Neural Networks with Connectionist Temporal Classification," in *International Conference on Machine Learning*, Seoul, 2016.

[21] D. P. Kingma and J. Ba , "Adam: A Method for Stochastic Optimization," in *International Conference for Learning Representations*, San Diego, 2014.

[22] J. Brownlee, "Loss and Loss Functions for Training Deep Learning Neural Networks," Machine Learning Mastery, 23 October 2019. [Online]. Available: https://machinelearningmastery.com/loss-and-loss-functions-for-training-deep-learning-neural-networks/. [Accessed 31 May 2021].

[23] A. Kathuria, "Intro to optimization in deep learning: Gradient Descent," Paperspace Blog, [Online]. Available: https://blog.paperspace.com/intro-to-optimization-in-deep-learning-gradient-descent/. [Accessed 31 May 2021].

[24] D. Ouyang, B. He, A. Ghorbani, N. Yuan, J. Ebinger, C. P. Langlotz, P. A. Heidenreich, R. A. Harrington, D. H. Liang, E. A. Ashley and J. Y. Zou, "Video-based AI for beat-to-beat assessment of cardiac function," *Nature,* no. 580, p. 252–256, 2020.

[25] American Heart Association, "Size, strength of heart's right side differs by age, gender, race/ethnicity," Science Daily, 9 June 2011. [Online]. Available: https://www.sciencedaily.com/releases/2011/06/110606161024.htm. [Accessed 31 May 2021].