DEGREE PROJECT IN COMPUTER SCIENCE AND ENGINEERING, SECOND CYCLE, 30 CREDITS

STOCKHOLM, SWEDEN 2018

Change Point Detection in Sequential Sensor Data using Recurrent Neural Networks

VEERESH ELANGO

KTH ROYAL INSTITUTE OF TECHNOLOGY
SCHOOL OF ELECTRICAL ENGINEERING AND COMPUTER SCIENCE

Change Point Detection in Sequential Sensor Data using Recurrent Neural Networks

VEERESH ELANGO

Master in Data Science
Date: September 20, 2018
Supervisor: Sarunas Girdzijauskas
Examiner: Henrik Boström
Industrial Supervisor: Kuo-Yun Liang

Abstract

Change-point detection is the problem of recognizing the abrupt variations in sequential data. This covers a wide range of real world problems within medical, meteorology and automotive industry, and has been actively addressed in the community of statistics and data mining. In the automotive industry, sequential data is collected from various components of the vehicles. The changes in the underlying distribution of the sequential data might indicate component failure, sensor degradation or different activity of the vehicle, which explains the need for detecting these deviations in this industry. The research question of this thesis focuses on how different architectures of the recurrent neural network (RNN) perform in detecting the change points of sequential sensor data. In this thesis, the sliding window method was utilised to represent the variable sequence length into fixed length. Then this fixed length sequences were provided to many input single output (MISO) and many input many output (MIMO) architectures of RNN to perform two different tasks such as sequence detection, where the position of the change point in the sequence is recognized and sequence classification, where the sequence is checked for the presence of a change point. The stacking ensemble technique was employed to combine results of sequence classification with the sequence detection to further enhance the performance. The result of the thesis shows that the MIMO architecture has higher precision than recall whereas MISO architecture has higher recall than precision but both having almost similar f1-score. The ensemble technique exhibit a boost in the performance of both the architectures.

Sammanfattning

Ändringspunktdetektering är problemet med att upptäcka den plötsliga förändringen av egenskaperna hos sekventiell data. Detta täcker ett brett spektrum av problem inom t.ex. medicin, meteorologi och fordonsindustrin, och har diskuteras aktivt i statistik- och datavinnnings området. I bilindustrin samlas sekventiella data från olika delar av fordonet. Förändringen i egenskapen hos sekventiella data kan indikera komponentfel, sensornedbrytning eller förändring av fordonets användning, vilket förklarar behovet av att detektera dessa avvikelser i denna bransch. I denna uppsats undersöker vi olika arkitekturer av återkopplade neurala nätverk (engelska recurrent neural networks, RNN), såsom många insignaler och en utsignal (MISO) och många in- och utsignaler (MIMO) arkitekturer, för att detektera detektera förändringspunkter över sekventiella data från fordonsensorer. I denna uppsats användes ett glidande fönster för att omvandla de variabla sekvenslängderna till sekvenser av fasta längder. Dessa sekvenser tillhandahölls till MISO- och MIMO-arkitekturerna för RNN för att utföra två olika uppgifter: sekvensdetektering för att hitta positionen av ändringspunkten i sekvensen och sekvensklassifiering för att upptäcka om sekvensen innehåller en ändringspunkt. Stapling av klassificerare har använts för att kombinera resultaten av sekvensklassificering med sekvensdetektering för att ytterligare förbättra prestanda. Resultatet av uppsats visar att MIMO-arkitekturen har högre precision än känslighet medan MISO-arkitekturen har högre känslighet än precision men båda har liknande f1-poäng. Stackningssamlingstekniken förbättrar resultaten i båda arkitekturerna.

Contents

Chapter 1

Introduction

In this thesis project, we investigate how different architectures of recurrent neural networks (RNN) can be used to detect change points in sequential data. The project was carried out within the research and development department of Scania CV AB, Södertalje, Sweden.

1.1 Background

The common kind of data observed in a wide range of industries is sequential measurements of data which describes the behaviour of systems over time. The change in the behaviour might be the vital indication of failure or a complete change in activity of systems. This problem of detecting the abrupt changes in the sequential data is called Change Point Detection. This problem covers a broad range of real-world problems such as climate change detection[47, 31, 16], speech recognition [10, 49], medical condition monitoring [59, 38], human activity analysis [57] are few example instances. In order to solve this problem, machine learning based methods of data science are chosen in this thesis.

The change point detection methods can be classified into two categories based on the delay of detection.

1. Online detection: These algorithms run concurrently with the process they are monitoring, processing each data point as it becomes available, with the real-time constraint that processing should complete before the next data point arrives.

2. Offline detection: These algorithms consider the entire data set at once, and there is no real-time constraint on the runtime.

The change points can be discovered only when there is a temporal dependence in sequential data. The essential property of sequential data is the order of the information, which is why algorithms designed to handle this kind of data must be able to process each element in the order that it appears. Sequential data are commonly observed in several industries such as medical, chemical, meteorological and automotive. In case of automotive industries, sequential measurements of various components are collected during manufacturing and also during the usage of the vehicle to study the behavioural changes of the component. The change in the behaviour might be due to different reasons, such as component failure, sensor degradation and different activity of vehicle. This detection of behavioural or abrupt changes in sequential data is called a change point detection problem. The first published article concerning change points was in 1954 [44]. This considered testing for a potential single change point for data from a common parametric distribution and was motivated by a quality control setting in manufacturing. Over the decades, changepoint analysis has developed rapidly with multiple change points, different types of data and other assumptions being considered.

In recent years, deep learning has emerged as one of the most popular machine learning techniques, yielding state-of-the-art results for a range of supervised and unsupervised tasks. In case of sequences, RNN outperforms other methods because of its "memory" which captures information about what has been calculated so far and their ability to learn long-range patterns [41, 52, 1]. Also, RNN, has a key feature that they can be applied to sequences of varying length without making any assumptions on how many previous points are needed for the predictions. Secondly, RNN has been shown to perform better than other methods in similar problems of change point detection, such as intrusion detection in [36, 39, 51].

1.2 Problem

There has been an extensive work on offline change-point detection methods, for example, the cumulative sum [3], the generalised likelihood-ratio method [25], and the change finder [53]. Such a strategy has also

been employed in novelty detection [24] and outlier detection [28].

In the supervised approach of change point detection, the machine learning algorithms can be trained as binary or multi-class classifiers. When the multiple states are defined, the algorithms can be trained to detect change point between the multiple states. This multi-class classifier approach requires sufficient amount of training to represent all of the classes. There are variety of algorithms applied to solve this problem, example decision trees [46][60][61], naive bayes [46], bayesian net [60], support vector machines [46][60], nearest neighbour [46][57], [11][27] and Gaussian mixture model [11][27]. In case of binary classifier approach, the algorithms such as support vector machines [18][13], naive Bayes[18] and logistic regression [18] has been tested. This type of problem suffers from extreme class imbalance as there are typically many more within-state sequences than change point sequences. This skewness in the amount of change point sequences makes it harder for the algorithms to detect as they are not completely unique value over the complete sequence but just a change in that context. Thus by approaching the data in a sequential manner where each observation has dependency over the previous observation might improve the detection but as for the best of our knowledge this has not attempted.

On the other side, Recurrent Neural Networks (RNN), and specifically a variant with Long Short-Term Memory (LSTM) [30], has recently emerged as an effective model in a wide variety of applications that involve sequential data. These include language modelling [41], handwriting recognition and generation [20], machine translation [52, 1], speech recognition [21], video analysis [15] and image captioning [56, 32].The LSTMs are also employed in anomaly detection which is the parent group of change point detection. There are works with supervised approach such as [9][37] and unsupervised approach such as [36, 8, 5].

Despite the success of RNN with sequential data in various fields and problems, its application for offline change point detection problem remains unclear so far.

1.3 Purpose

This thesis attempts to fill the knowledge gap in the application of different architectures of RNN for detecting the change points in the

sequential form of sensor data and their limitations as well. This thesis will investigate the following research question: *How do the different architectures of the recurrent neural network perform in detecting the change points of sequential sensor data?*

1.4 Ethics and Sustainability

The benefit of this project is that the industries from various sectors which has the similar problem will get to know whether RNN will be suitable to solve their problem or not.

The direct and indirect impacts of change point detection are discussed.

- Direct impact: CPD is used to understand the abrupt changes in the behaviour of objects and to take action during the change points or used to enhance the performance of the objects e.g. sleep pattern segmentation, failure of the vehicle component. Thus it does not cause any direct impact on environment or concerns regarding production, waste, harmful by-products, or pollution. Even though it will require computer hardware resources to be implemented, the energy consumption and other effects of such systems are negligible.

- Indirect impact: This project is mainly focused on the offline change point detection. This helps in labelling the possible replacement of a component happened over the time in the vehicles. The results of this project will help in the further analysis of the performance of the specific components and do not directly affect the real world user. Saying that the results of this project will indirectly help in further analysis to predict the optimal time for component replacement or service. Based on this prediction, a component replacement or service could be extended. If the prediction went wrong, it will lead to on-road failure which in turn leads to loss of the customer. Thus it demands high attention while making use of the results of change point detection

Since we used artificial datasets, publicly available datasets and the authorised data collected from vehicles of Scania's customers there were no concerns regarding the collection of data from human subjects. Use of public datasets also allays privacy concerns. We do not

claim superiority over other works, do not falsify any results or data, and take appropriate caution to avoid plagiarism and give proper citations wherever needed.

1.5 Methodology

The research question focuses on the effectiveness aspect of the models created using MISO and MIMO architectures of RNN. The effectiveness of the model could have been analysed theoretically using the statistical implications in [30]. Instead, as this was an industrial thesis project, the effectiveness aspect of the thesis will be approached using the *empirical* and *experimental* research methods. The main dataset used in thesis was collected from several hundreds of Scania trucks during their visit to workshops. Hence, *experiments* data collection method will be employed rather than other methods such as *questionnaire* and *case study*. The experiments on the models of two different architectures of RNN has used the same dataset and varying other parameters such as learning rate, number of hidden layers, additional features and data scaling techniques. The result of these experiments are the numerical values which are collected again using *experiments* data collection method. These results are then processed quantitatively using *descriptive statistics* data analysis method.

1.6 Delimitations

The dataset used in this thesis has at most only one change point in the sequence, hence only single change point detection problem is focused. This thesis focuses only on offline change point detection.

1.7 Outline

The rest of the report is organised as follows: In chapter 2 we give an overview of the theory and concepts that are essential to understanding the work done in rest of the project. Next, we introduce the model and methods used in this project in chapter 3. Then we present our results and evaluation in chapter 4. A discussion on the method used,

experiments, results and some ideas for future work follows in chapter 5. Finally, we conclude the report in chapter 6.

Chapter 2

Extended Background

2.1 Artificial Neural Networks

Artificial neural networks (ANNs) were originally developed as mathematical models of the information processing capabilities of biological brains [40, 48]. It is a computational model that is inspired by the way biological neural networks in the human brain process information. The basic unit of computation in a neural network is the neuron, often called a node or unit. The node combines the input from external source or from some other nodes with set of weights. These weighted signals from multiple nodes are then added together and sent into non-linear activation function which then creates the output. The functioning of single neuron is shown in the Figure 2.1.

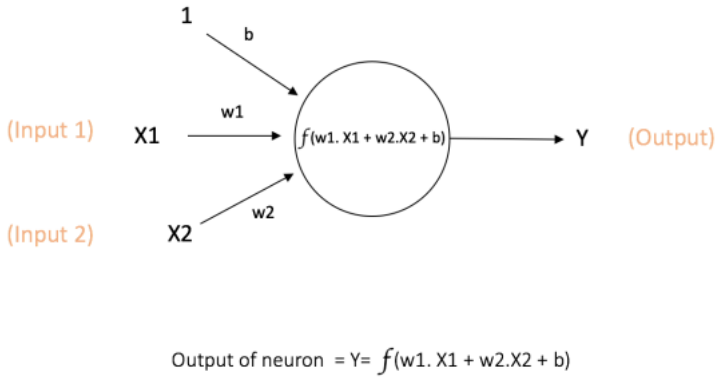


Figure 2.1: **Functioning of a neuron**. Image adapted from [55]

In the past years, many varieties of ANNs have appeared, with

widely varying properties. The important distinction is between ANNs whose connections form cycles, and those whose connections are acyclic. ANNs without cycles are known as feed forward neural networks. The most widely used form of feed forward neural network is *multi layer perceptron* (MLP) [48, 4]. ANNs with cycles are referred to as feedback, recursive, or recurrent, neural networks, and are dealt with in Section 2.2

2.1.1 Multilayer perceptrons

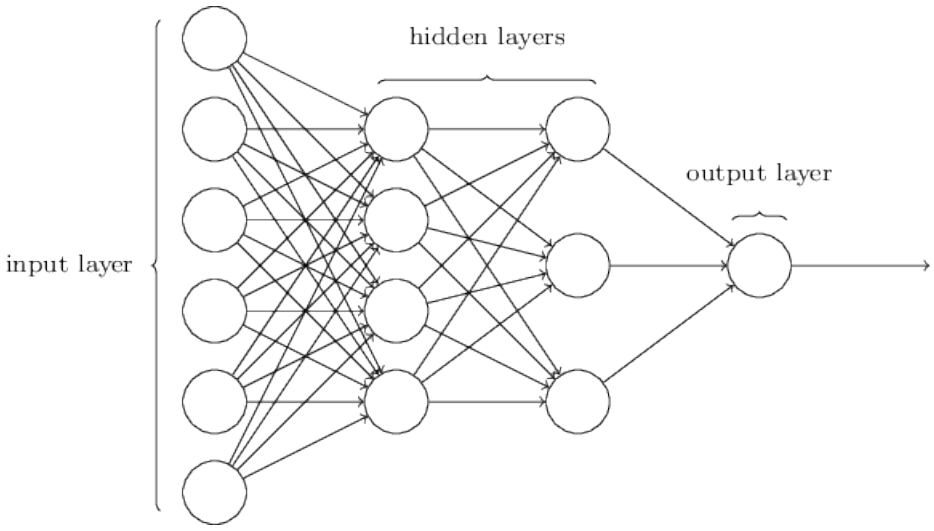


Figure 2.2: Multilayer perceptron. Image adapted from [42]

A multilayer perceptron consists of a system of simple interconnected neurons or nodes, as shown in the Figure 2.2, which is a model representing a nonlinear mapping between an input vector and an output vector. The nodes are a function of the sum of the inputs to the node connected by weights and output signals which are modified by a simple nonlinear transfer, or activation, function. The superposition of many simple non-linear transfer functions that enables the multilayer perceptron to approximate extremely non-linear functions. The output of a node is scaled by the connecting weight and fed forward to be an input to the nodes in the next layer of the network. This implies a direction of information processing, hence the multilayer perceptron is known as a feed forward neural network. The architecture of a multilayer perceptron is variable but in general will consist of sev-

eral layers of neurons. A multilayer perceptron may have one or more hidden layers and finally an output layer. There exist no interconnections within a layer while all neurons in a layer are fully connected to neurons in adjacent layers.

Multilayer perceptrons have the ability to learn through training in a supervised manner. During training the multilayer perceptron is repeatedly presented with the training data and the weights in the network are adjusted until the desired input–output mapping occurs. An error signal is defined as the difference between the desired and actual output. Training uses the magnitude of this error signal to determine to what degree the weights in the network should be adjusted so that the overall error of the multilayer perceptron is reduced. There are many algorithms that can be used to train a multilayer perceptron. The multilayer perecptron can generalise to new unseen input data when it is trained with suitable representative training data.

2.1.2 Backpropagation

The popular training algorithm used in multilayer perceptron is backpropagation. The backpropagation training algorithm[48] uses the procedure of gradient descent to attempt to locate the absolute (or global) minimum of the error surface. Backpropagation has been shown to perform adequately in many applications; the majority of the applications discussed in [48] used backpropagation to train the multilayer perceptrons. Backpropagation only refers to the training algorithm and is not another term for the multilayer perceptron or feed-forward neural networks, as is commonly reported.

The online training based backpropagation algorithm is summarised below based on the implementation in the book [4]

1. initialise network weights
2. present an input vector from training data into the network
3. propagate the input vector through the network to obtain an output
4. calculate an error signal by comparing the actual output to the desired output
5. error signal is propagated back through the network

6. alter weights such that overall error gets minimised
7. until overall error is satisfactory, repeat steps 2-7 with next input vector

In this online training based backpropagation algorithm, the network weights are adapted after each pattern has been presented. The alternative is known as batch training, where the summed error for all patterns is used to update the weights.

2.1.3 Activation Functions

The commonly used activation functions are *sigmoid, tanh,* and *rectified linear units*[35]. The sigmoid function has s-shaped graph and squashes the input into range between 0 and 1. In particular, large negative numbers become 0 and large positive numbers become 1. This function is differentiable and monotonic but its derivative is not monotonic. It has mathematical form is defined by

$$\sigma(x) = \frac{1}{1 + \exp(-x)} \tag{2.1}$$

. The hyperbolic tangent activation function is symmetric bipolar activation function and it squashes the input into the range between -1 and 1. This function is differentiable and monotonic but its derivative is not monotonic. Its mathematical form is defined by

$$tanh(x) = 2\sigma(2x) - 1 \tag{2.2}$$

The rectified linear unit is simply a threshold at zero and it is famous in recent years because it greatly accelerate the convergence of stochastic gradient descent when compared to sigmoid and tanh. Both the function and its derivative are monotonic. Its mathematical form is defined by

$$f(x) = max(0, x) \tag{2.3}$$

2.2 Recurrent Neural Networks

Recurrent neural networks (RNN) are a type of neural network used to process the sequential data where outputs are dependent on the previous computations. To process this sequence of inputs, RNN uses

internal state (memory) which is not possible with the feedforward neural networks as it consider each input independently. This memory is called the state, and is denoted at the time t as h_t. The state at time t can be described by the following equation:

$$h_t = f(h_t - 1, x_t) \tag{2.4}$$

where $h_t - 1$ is the previous state and x_t is the current input. Unlike a traditional deep neural network, which uses different parameters at each layer, an RNN shares the same parameters across all steps. This reflects the fact that we are performing the same task at each step, just with different inputs. This greatly reduces the total number of parameters we need to learn.

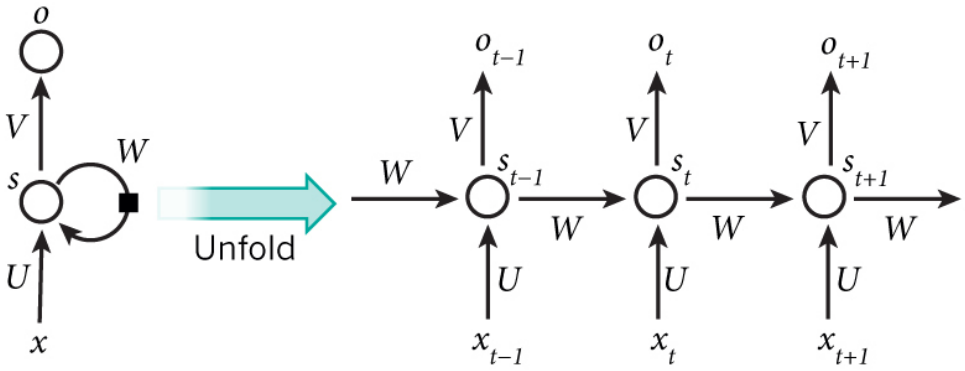


Figure 2.3: **Structure of RNN**. The left hand side figure shows the folded RNN whereas the right hand side figure shows the unfolded RNN. Image adapted from [7]

As shown in the Figure 2.3, the learning at each step is based upon the previous states. This nature of influence from the previous input to the current is not observed in the artificial neural networks explained in the section 2.1.

The core reason that RNNs are more exciting is that they allow us to operate over sequences of vectors: Sequences in the input, the output, or in the most general case both.

The RNNs work upon the fact that the result of an information is dependent on its previous state or previous n time steps. Regular RNNs might have a difficulty in learning long range dependencies. The problem was explored in depth at [45], which found fundamental reasons why it might be difficult. The backpropagation of error

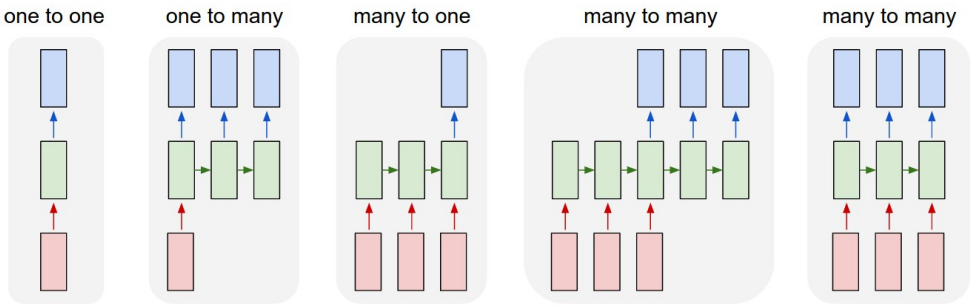


Figure 2.4: **Different operations on RNN**. a) Fixed size input and output b) Sequence output c) Sequence Input d) Sequence Input and Sequence Output e) Synced sequential input and output . Image adapted from [54]

works like chain rule. In chain rule, if any of the gradients approached zero, all the gradients would rush to zero exponentially fast due to the multiplication. Such states would no longer help the network to learn anything. This is known as the *vanishing gradient problem*.

2.2.1 Different Architectures

The main advantage of the recurrent nets are that they allow us to operate over sequences of vectors. As shown in the Figure 2.4, the RNN could be designed in several methods based upon the input and output. The different architectures in the Figure 2.4 a) it takes single input and expects single output which is usual neural network architecture without RNN used in cases such as image classification, b) it takes single input and provides many outputs with RNN used in cases such as image captioning where multiple words are produced as output for single input, c) it takes many inputs and provide single output (MISO) with RNN used in cases such as sentimental analysis where sequence of words are given as input and the sentiment class is obtained as output, d) it takes many input and provides many output where not all the cells of RNN have corresponding output, used in cases such as machine translation, and e) it takes many input and provides many output (MIMO) where each cell of RNN has been synchronised with output, used in cases such as video classification where we wish to label each frame of the video. In this thesis, the MISO and MIMO architectures are chosen for the investigation.

2.2.2 Bidirectional RNN

The RNNs have a *casual* structure, meaning that the state at time t captures information only from the past $x_1, ..., x_{t-1}$ and the present input x_t. However, in many application, the output may depend on *whole input sequence*. For example, in speech recognition, because of co-articulation the correct interpretation of the current sound as a phoneme may depend on the next few phonemes and may even depend on the next few words because of the linguistic dependencies between nearby words.

Bidirectional RNN(BRNN) [50] was invented to address these kind of problems and extremely successful in applications such as handwriting recognition [23], speech recognition [22] and bioinformatics [2].

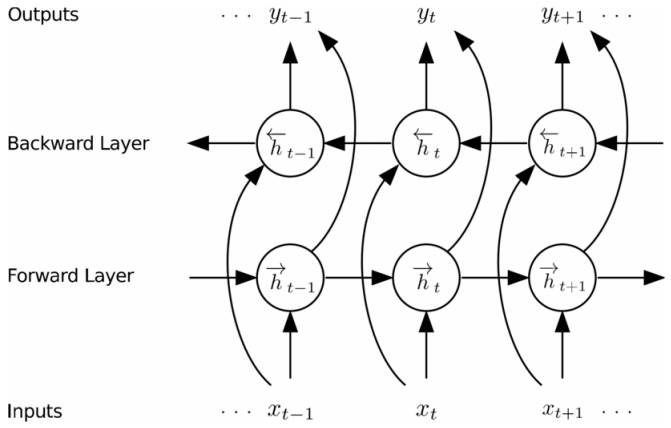


Figure 2.5: BRNN architecture. Image adapted from [21]

BRNN combines an RNN that moves forward through time, beginning from start of the sequence, with another RNN that moves backward through time, beginning from the end of the sequence, as shown in the Figure 2.5. The output layer is not updated until both hidden layers have processed the entire input sequence.

2.2.3 Long short-term memory network

Long short-term memory networks (LSTM) is a special kind of RNN introduced in [29] which was capable of learning long-term dependencies through recurrently connected subnets known as memory blocks. Each block contains one or more self-connected memory cells and three

multiplicative units such as input, output and forget gates. This provides the ability for the LSTM to remove and add information to the cell state.

2.3 Need for RNN in Change Point Detection

Change points also appear under a variety of synonyms across a variety of scientific fields. This includes *segmentation, structural breaks, break points, regime switching* and *detecting disorder*. The point is to have an algorithm that can automatically detect changes in the properties of the sequences for us to make the appropriate decisions.

The change point in a sequence will be very few like anomalies which makes it harder for the classification problem to detect it, as the dataset will be biased. Also, the points are considered to be changes only at that temporal context and not as independent point. Hence, this problem cannot be solved using general classification algorithms.

As explained in the above section, RNNs have memory within their structure, which makes them capable to caputre the patterns inside the sequences. The behavioural change in the sequence at any temporal context will also have pattern among them. Thus, RNN makes a reasonable option to solve the change point detection problem as they have the capability to learn the pattern in the sequences.

2.4 Ensembling

Ensembling is the process combining predictions of set of individually trained classifiers to obtain better performance [43, 62]. It has been applied on problems such as cancer prediction, sentiment analysis[17], images etc., The ensembling provides better performance when the learning algorithm is searching for a space of hypotheses that is large for the amount of available learning data and also, when a learning algorithm suffers a high variance[14].

The ensembling methods are broadly classified into

1. *Bagging* [6] (stands for Bootstrap Aggregating) is an ensemble method where the same learning algorithm is used for different training subsets to obtain different classifiers. The training subsets are obtained by sampling with replacement on the train-

ing data. The individual classifiers' predictions (having equal weights) are then combined by taking majority voting.

2. *Boosting* [19] uses a family of machine learning algorithms to convert weak learners to strong ones. The first classifier is constructed from the original dataset where every sample has an equal weight. In the succeeding training dataset, the weight is reduced if the sample has been correctly classified, otherwise it is increased if the samples are misclassified. In the committee decision, a weighted voting method is used so that a more accurate classifier is given greater weights than a less accurate classifier.

3. *Stacking* [58], performs the tasks in two phases i) in phase-1, the classifiers are trained on the same data and ii) in phase-2, the output of the classifiers are fed to meta-level classifier function. The purpose of this method is to correct the incorrect learning of a classifier through other classifiers, as shown in the Figure 2.6. The phase-2 layer of this method can be replaced with different rules.

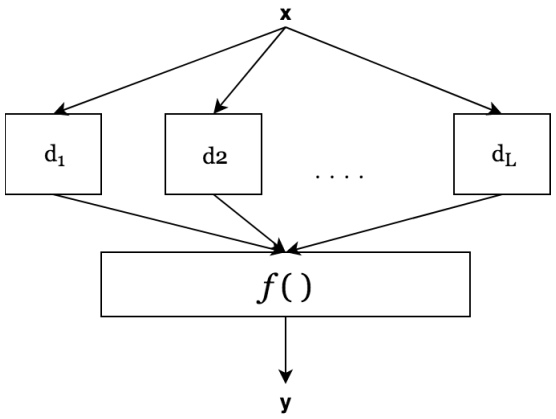


Figure 2.6: Stacking method of ensembling

2.5 Evaluation Metrics

A first step at evaluating the performance of a supervised approach is to generate a confusion matrix which summarises the actual and predicted classes. The confusion matrix is a specific table visualisation of the performance of an algorithm. The number of correct and incorrect

predictions are summarized with count values and broken down by each class. Figure 2.7 illustrates a confusion matrix for a binary classifier.

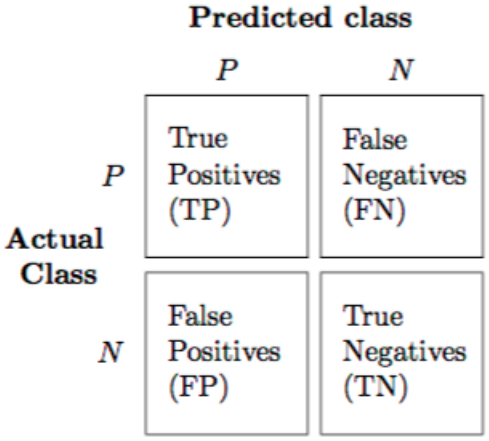


Figure 2.7: **Confusion matrix**. Image adapted from [12]

1. Accuracy is calculated as the ratio of correctly-classified data points to total data points. This measure provides a high-level idea about the algorithm's performance. Error Rate is the companion to accuracy, which is computed as *Accuracy*. Accuracy and Error Rate do not provide insights on the source of the error or the distribution of error among the different classes. In addition, they are ineffective for evaluating performance in a class-imbalanced dataset, which is typical for change point detection, because they consider different types of classification errors as equally important.
$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN} \tag{2.5}$$
2. Recall is the true positive rate (TP Rate). This refers to the portion of positive class that was classified correctly.
$$Recall = TPRate = \frac{TP}{TP + FN} \tag{2.6}$$
3. Precision is calculated as the ratio of true positive data points to total points classified as particular class.
$$Precision = \frac{TP}{TP + FP} \tag{2.7}$$

4. F-measure is the measure that combines precision and recall is the harmonic mean of precision and recall.

$$F_\beta = (1 + \beta^2) \cdot \frac{precision \cdot recall}{\beta^2 \cdot precision + recall} \tag{2.8}$$

5. Receiver Operating Characteristics (ROC) curve plots the TP-rate vs. the FP-rate as a threshold on the confidence of an instance being positive is varied

6. Precision-Recall Curve (PR Curve) plots the precision vs. recall (TP-rate) as a threshold on the confidence of an instance being positive is varied

2.6 Sequence Representation

In the automotive industry, generally, the interval between observations varies based on the type of the vehicle and also the number of observations for all the vehicles at any given time *t* will not be the same. In case of, vehicle sensor observations, the abrupt change in the behaviour is not frequent as the changes might be observed due to rare actions such as component failure, degradation of the sensor, replacement of the component or change in activity of the vehicle. Thus, the possibility of change points in the entire sequential observations of a vehicle will be at most one or two in most of the cases. Also, the change points are more clearly distinguishable in its specific occurrence context rather than the entire observations. Thus it is preferable to slice the entire sequential observation of every vehicle into multiple chunks as it makes the change points more distinguishable.

The sliding window technique was used in this thesis to divide the entire sequence of every vehicle into multiple chunks with fixed window size `n`. For a given sequence of `k` observations, the sliding window technique with fixed window size of `n` creates `(k-n)+1` subsequences as shown in the Figure 2.8. This technique captures possible previous and following observations corresponding to the change point. This technique was also used in data types such as images, text, audio.

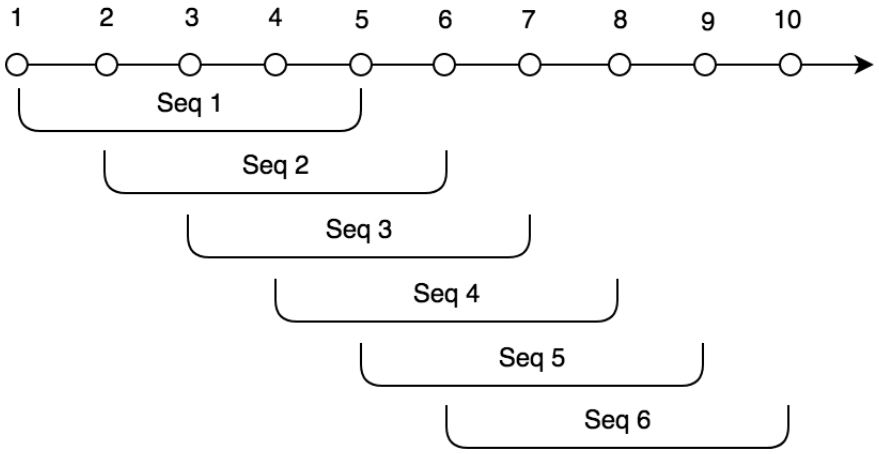


Figure 2.8: Example of sliding window mechanism applied on a sequence of length 10 with window size of 5

2.7 Models

This thesis approached the change point detection in two ways. The first approach was to create a *sequence detection* model with the motive to detect at which point in sequence the change has happened. The second approach was to create *sequence classification* model to classify the sequence has change point or not. The models of these approaches were trained independently with similar input data. Then the results of these models were combined using the ensemble technique mentioned in the section 2.7.3 to investigate the performance.

2.7.1 Sequence Detection

Sequence Detection is a task of identifying events or observations within input sequence, such as anomalies or change points.

The objective of this approach was to investigate the capability of RNNs to detect which observation in the sequence was a change point. The change point in the sequence of `n` observations can be present at any observation of the entire sequence or there can be no change point in the sequence at all. This leads to approach this task in two different ways. The first is to consider this as *multi-class classification* problem as any sequence can be classified into any one of `n+1` target classes. The second approach was to consider this as *multi-label classification* problem as each observation in a sequence can be a change point or

not.

Multi-class classification with MISO architecture (MC-MISO)

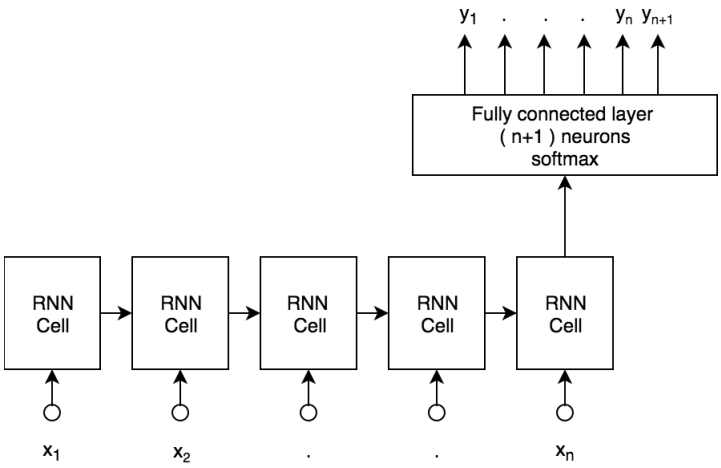
In Multi-class classification approach, the sequential data with `n` time steps will be fed as input and the output will be categorised into any one of the target class with maximum probability. For this requirement of input and output structure, the *many to one* architecture of RNN is chosen, as shown in Figure 2.4 (c). The models created with this approach have the first layer as RNN (or any of its variants) and the output layer as a fully connected layer. Since the output has to be any one of the `n+1` target classes, *softmax* function is chosen as the activation function for the output layer as this function will calculate the probabilities of each target class over all possible target classes, as shown in the Figure 2.9a.

Multi-label classification with MIMO architecture (ML-MIMO)

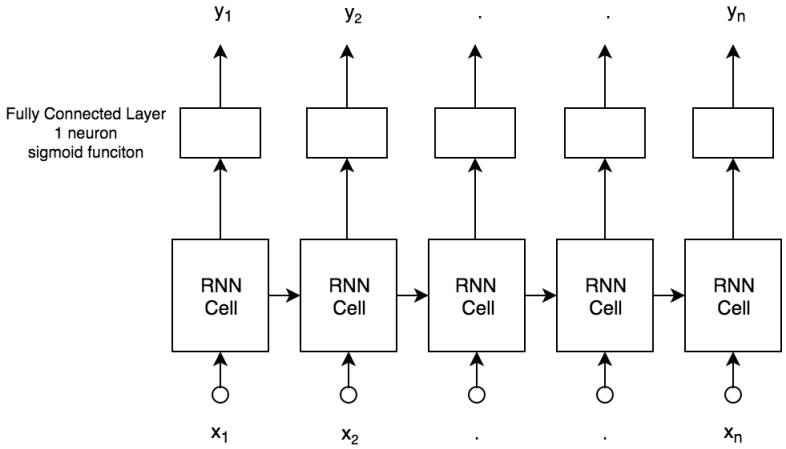
In Multi-label classification approach, the sequential data with `n` time steps will be fed as input and each time step will be assigned with a label at the output. For this requirement of input and output structure, the *many to many* architecture of RNN is chosen, as shown in the Figure 2.4 (e). The models created with this approach also have the first layer as RNN (or any of its variants) and the output layer as a fully connected layer. The output of the models of this approach has to classify each time step of the sequence as change point or not. Thus *sigmoid* function is chosen as the activation function for the output layer as it could diminish any real numbers to the range between 0 and 1, as shown in the Figure 2.9b

2.7.2 Sequence Classification

Sequence classification is where each input sequence is assigned a single class. It has many real-world applications like in health-informatics, classifying ECG signals to tell if it is from a healthy person or from a patient with heart disease[57] and in anomaly detection/intrusion detection, the sequence of a user's system access activities on Unix is monitored to detect abnormal behaviours[34].



(a) MISO



(b) MIMO

Figure 2.9: Sequence Detection Model Architectures

Binary classification with MISO architecture

(BC-MISO) The objective of this approach was to investigate the capability of RNNs to identify the pattern among the sequences which has change points and classify them correctly. This was a comparatively less challenging task for RNN as it has to detect whether the sequence has change point or not rather than detecting where the change has happened in the sequence. In this approach, the sequential data with `n` observations have to be fed as input and output will be one target class, thus *many to one* architecture was chosen for this. The models

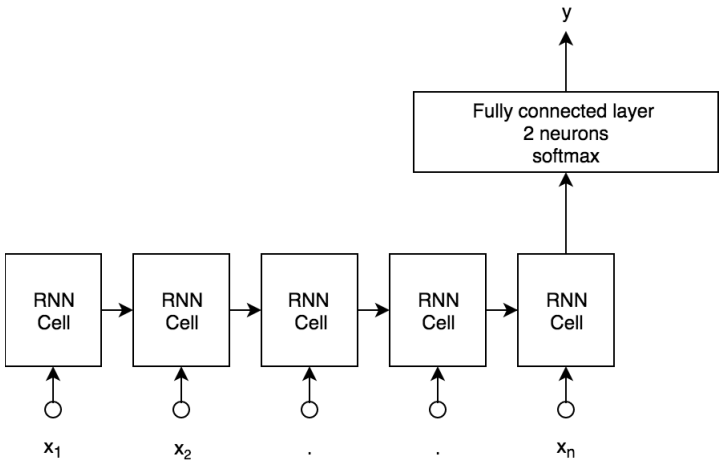


Figure 2.10: Sequence Binary Classification model

created with this approach have RNN as the input layer and the output layer is a fully connected layer with activation function as *softmax*, as shown in the Figure 2.10

2.7.3 Ensembling

Ensembling provides better performance as it combines the predictions of set of individually trained classifiers [43, 62, 17, 14].

The ensemble technique was chosen for this thesis primarily because the amount of training dataset is less and to investigate the effect of this technique in change point detection problem. The *stacking*[58] method focused in this thesis with the phase-1 layer having sequence detection and sequence classification model trained on the same data and the phase-2 layer is defined with a function which considers the output of the sequence detection model only when the sequence classification model predicts change point in the sequence, as shown in the Figure 2.11.

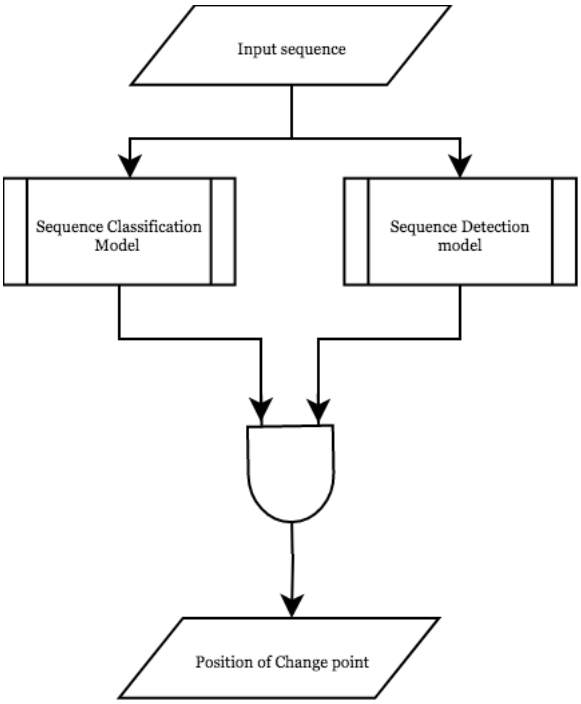


Figure 2.11: Ensemble Learning - Combining Function

Chapter 3

Methodology

This chapter describes the detailed strategy to answer the research question raised in Section 1.3

How do the different architectures of the recurrent neural network perform in detecting the change points of sequential sensor data?

The above research question is concerned with effectiveness aspect as the performance in detection of the change points is compared between the different models of two different architectures of recurrent neural networks. We note that only many input many output (MIMO) and many input single output (MISO) architectures (as described in Section 2.2.1) were chosen for this research analysis as they were best suitable for the dataset used in this thesis.

3.1 Research Methods

The taxonomy of the most common research methods referred in this thesis were described in [26] to choose the appropriate research method in this work. The choice of the methods is focused around the research question.

The research question focuses on the effectiveness aspect of the models created using MISO and MIMO architectures of RNN. The statistical implications of the RNN is explained in [30] which could be used to analyse the effectiveness of models of different architectures. Instead, as this was an industrial thesis project, the effectiveness aspect of the thesis will be based on the *empirical* research method. In

this research, knowledge will be gained quantitatively by conducting sequence of experiments and the performance of the different models with the same set of data sets.

3.1.1 Data collection

In order to execute the before mentioned experiments, the data collection methods has to be specified. The data collection indicates the data collected for the experiments and also the results of the experiments.

The research question of this thesis focuses on the effectiveness of different models created with MISO and MIMO architectures of RNN. As these models will be developed using deep learning based algorithms, they need large data set to produce effective results. The dataset required to solve the research question should have large observation of sensor readings with labels on the change points. Any dataset which has this characterstics can be used. Thus best data collection method to solve the research question will be *experiments* method. The other data collection methods are inapplicable to solve this research question.

In this thesis, a large dataset of sensor readings collected from several hundreds of Scania trucks running all over the world will be used along with another synthesised dataset with almost similar characterstics and the creation is explained in Section 3.3.2.

Similarly, for the data collection of experiment results, the *experiments* data collection method was employed. For the experiment of evaluating the effectiveness of the models of the MISO and MIMO architectures, the results included the precision, recall and f1-score on two different data sets.

3.1.2 Data Analysis

After the results of the experiments are collected, appropriate data analysis method has to be employed to support decision-making and drawing conclusions.

The results of the experiments will be measured using the evaluation metric such as precision, recall and f1-score. These measurements has to be compared with each other in order to calculate the effectiveness of the different models. The sense of the *comparable performance* is subjective hence the comparable performance in this research is de-

fined with respect to the f1-score metric of the models. The model with greater f1-score is accepted as the better model than the others. The choice of f1-score has been explained in later Section 3.6. The *descriptive statistics* data analysis method will be employed in order to draw conclusions. Since, the results of the experiments need to compared statistically other possible options would be to apply statistical hypothesis testing based data analysis method.

3.2 Method Outline

As mentioned in the Section 1.3, the ability of MISO and MIMO architectures of the RNN to recognise the change points were studied in this thesis.

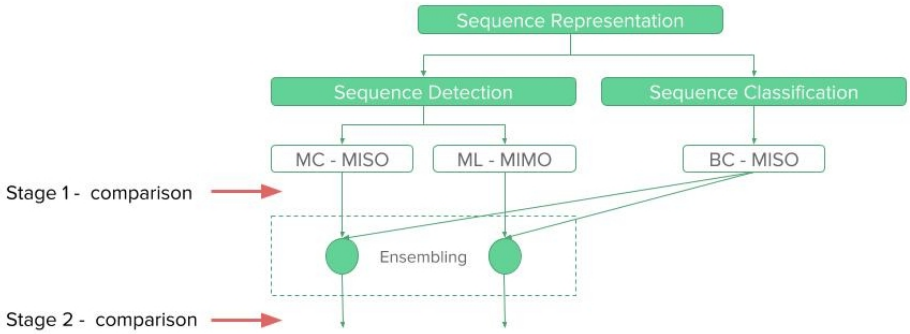


Figure 3.1: The high level explanation of the methodology applied in this thesis. (MC- Multi Class, ML-Multi level, BC- Binary Class)

As shown in the Figure 3.1, the sequence representation was the first step in this thesis. The total number of observations of every truck varies. In order to have constant sequence length, the sliding window method (explained in Section 2.6) was applied. The choice of this method when compared to other naive methods like choosing fixed number of observations was to prevent the loss of data. The dataset used in this thesis were provided by Scania and a similar synthetic dataset was created in order to evaluate the conclusions. The description of the dataset collection and processing was explained later in the Section 3.3. After processing the dataset, it was split into 70 - 30 for train and test. The experiments were first conducted to create sequence classification model using MISO architecture. Since the sequence clas-

sification task has only two classes, either yes or no, only MISO architecture was used. The experiments were conducted with different RNN variants such as simple RNN, LSTM, Bidirectional RNN and Bidirectional LSTM. The experiments started with single hidden layer and five cells in it, as each sequence had five data points after applying sliding window. The number of epochs and learning rate was started with 100 and 0.001 respectively. After the identifying the best model, the experiments for sequence detection model was started. For the experiments of sequence detection, models were created using both MISO and MIMO architecture. The experiments included varying data processing methods such as with and without normalisation, with and without scaling, and with and without adding features such as milage and truck details. Also the experiments were conducted with different RNN variants such as simple RNN, LSTM, Bidirectional RNN and Bidirectional LSTM. The experiments started with single hidden layer and five cells in it, as each sequence had five data points after applying sliding window. The number of epochs and learning rate was started with 100 and 0.001 respectively. The evaluation metrics used in this thesis were precision, recall and f1-score (as discussed in Section 2.5). The reason for the choice this evaluation metric was due to the skewness in the dataset and recommendation from Scania.

3.3 Dataset

The dataset used for this project was provided by Scania. The data was the sequential pressure sensor measurements of *Diesel Particulate Filter* component.

3.3.1 From Scania

Description

The diesel particulate filter (DPF) is a device designed to remove diesel particulate matter or soot from the exhaust gas of a diesel engine. The incomplete combustion of diesel fuel produces soot (particulate matter) particles resulting in diesel particulate matter. These particles are tiny nanoparticles - smaller than a thousandth of a millimetre (one micron). Soot and other particles from diesel engines worsen the particulate matter pollution in the air and are harmful to health. The DPF

gets accumulated with this particles over the time which is constantly converted to CO_2 by the coated catalyst under the passive heat of the exhaust system. This conversion process is believed to be smooth in case of long-haulage vehicles and uneven in case of distribution vehicles. Thus the DPF has another mechanism called *regeneration*, when the soot accumulation reaches 80%, the temperature inside the DPF will be automatically increased to induce the conversion of soot to ash. Over the period of time, most of the DPF gets filled with ash and partly soot, which indicates the replacement of either new filter or other cleaned DPF.

Data Collection

The DPF load was measured using the differential pressure sensors between the start and end of the DPF. This differential pressure measurement was then converted to the percentage of the maximum allowed pressure drop of the DPF using the proprietary algorithm of the company in the engine control unit of the vehicle. There is a limitation in the data storage capacity inside a vehicle as several signals are measured continuously or at regular intervals. Hence, the data is aggregated in the form of histograms and then stored. In case of DPF load, the percentage of the maximum allowed pressure drop is divided into 8 range of values from zero to maximum and the total amount of time (in seconds) spent in each range is stored. Thus each observation of the DPF load will have eight features where each feature representing a particular range of the capacity. The values of those features are the total time (in seconds) spent in that range from the last reset of the values.

The dataset consists of measurements taken sequentially for 187,112 trucks. Among these trucks, the experts have labelled the filter replacements in 115 trucks. These 115 trucks, consists of 2588 observations in total were each observation is of 8 features.

Data Pre-Processing

There are three major data pre-processing steps applied after discussing with the experts related to this data. The first step is the removal of aggregation as the DPF load observations were aggregated values which are not always reset after every retrieval. In order to maintain the consistency, the aggregation is removed by the algorithm generally used

in the company for this purpose. The second step is the removal of unlabelled data as the expert has labelled at most one filter change per vehicle's sequence of observations which could have more than that. Those labels denote the first filter change of the vehicles, so observations after the label are removed as there could be a possible observation with filter change which we do not know where it is present over the sequence. The third step is to assign the first observations of all the vehicles as filter changes as it has more similar characteristics of the filter change. A processed observation of a sample vehicle is shown in the Figure 3.2

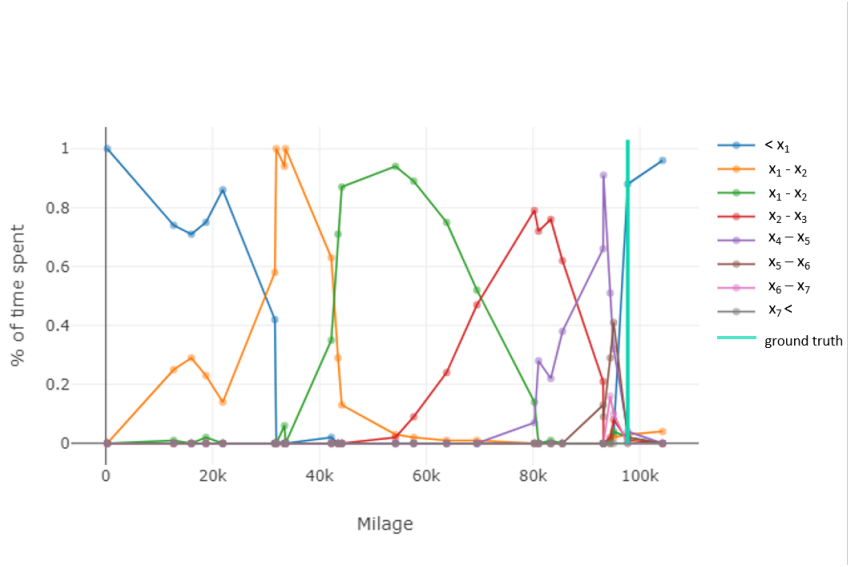


Figure 3.2: A sample vehicle's processed sequence of DPF sensor observation. The green line indicates the filter change

During the service visits, the DPF is replaced based on the fuel and oil consumption. The mechanical model utilising the fuel and oil consumption may lead to the early or late replacement of the DPF which is a cost for the customers. Thus the company is seeking to find the optimal milage to replace the DPF using data-driven methods. In order to solve this bigger challenge, the company is first looking to find when the filters were replaced in the vehicles as data about the filter changes in chassis are entered manually and not useful. Thus the business objective of the company in this thesis was to find when the DPF has been replaced in the vehicles with the help of the sequential observation of differential pressure sensor readings taken during the service

visits of the vehicle.

Dataset Preparation

The machine learning models modelled in this thesis expects the input data in the form of a sequence. The sequence of observations of every vehicle was divided into sub-sequences using *sliding window technique*, as shown in the Figure 2.8. The windows size chosen for this technique was 5, as it was the minimum length of the sequence of a vehicle in the dataset. The application of this technique resulted in the creation of 8720 sequences of length five in total. This dataset is then divided into train and test based on the distribution of the type of vehicles as the characteristics of the soot load accumulation is correlated to the type of vehicle. By random, 102 vehicles summing up to 7360 sequences were chosen for training and 13 vehicles summing up to 1360 sequences were chosen for testing.

According to the model architectures, three different kinds of labels were created for this dataset. For the multi-class classification model, as mentioned in the sub-section 2.7.1, every sequence has one hot encoded vector of length one greater than the input sequence length. For the multi-label classification model, as mentioned in the sub-section 2.7.2, every sequence has a vector of length equal to the input sequence length. These vectors are filled with 1's and 0's were 1 represents the filter change of the corresponding observation in the input sequence and vice versa. For the binary classification model, as mentioned in sub-section 2.7.2, every sequence has one hot encoded vector of length two with 1 representing change point in the sequence and vice versa.

3.3.2 Synthesised

In order to compare the results, another dataset has been synthesised with the help of *butter* function in *scipy*. In the real dataset, every vehicle has sequences of observations representing the values taken from the same sensor. For synthesised dataset, 200 sequences, each of 50 observations and one change point per sequence has been generated. A sample sequence is shown in the Figure 3.3.

Every sequence was synthesised by combining two butter signals of different frequency at random places. This synthesised dataset was processed using sliding window (window length =5) similar to the real

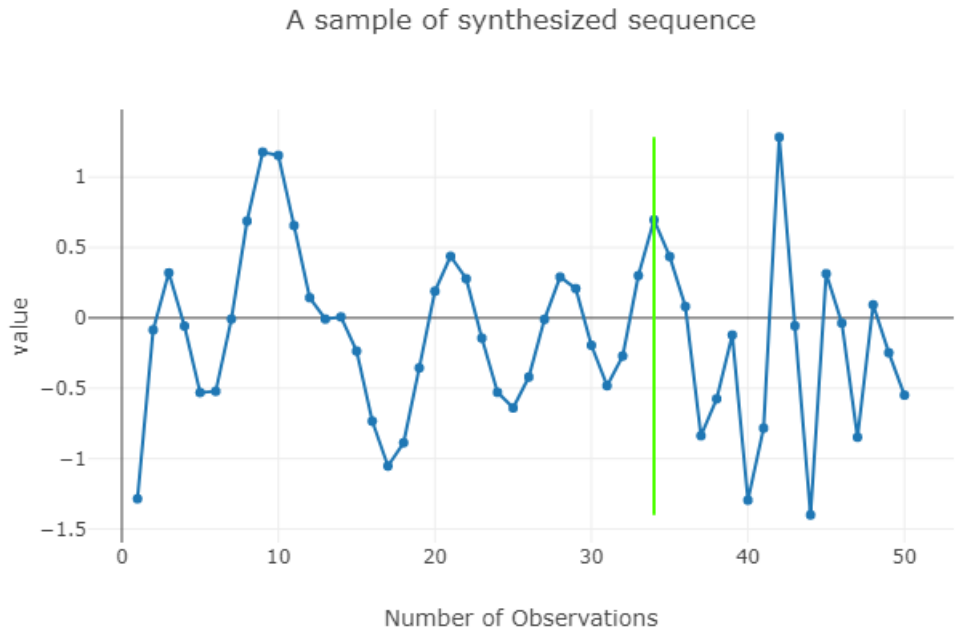


Figure 3.3: A sample sequence from the synthesised dataset. The green line indicates the change point.

dataset. The training dataset has randomly chosen 150 sequences, after applying sliding window it became 6900 sub-sequences. The test dataset has the remaining 50 sequences, after applying sliding window it became 2300 sub-sequences. The main difference between the real and synthesised dataset was that the synthesised was uni-variate whereas the real was multi-variate.

3.4 Experimental setup

The libraries used during the thesis are mentioned in the Table 3.1 .

With *python* as the choice of programming language, *scipy* & *pandas* was used for data exploration and preparation, *keras* library with *tensorflow* as backend was used to create deep learning models and *plotly* was used for visualisation.

Libraries	Versions
python	3.4.5
scipy	0.18.1
pandas	0.20.3
keras	2.1.5
plotly	2.0.15

Table 3.1: Details about the libraries used in this thesis

3.5 Baseline models

In order to compare the performance of the sequence detection and sequence classification models, baseline models were create as explained below.

- Sequence detection: The business case of this thesis was to detect the change point in the complete sequence of the vehicle. The exploratory analysis of the dataset helped in the observation of a common behaviour of sudden increment in a particular feature whenever there was a filter change. We applied *gradient* to capture the sudden increment and applied if-else condition until it does not overfit with the data. This rule-based model was used to compare the performance of sequence detection models.

- Sequence Classification: The objective of the sequence classification model was to detect whether the given sequence has change point or not. The business requirement was not close to the objective of these models but this was another important kind of use case in change point detection problems. Thus, the traditional approach of comparing the machine learning model with the random performance was chosen for these models. For the random model, the final value is the average of the random prediction of zero and one for 10 simulations.

3.6 Evaluation

The evaluation metric chosen for this thesis is *f1 score* as explained in the Section 2.5. The f1-score is a harmonic mean of precision and recall. The motivation to choose this evaluation metric was because of

the biased dataset. Due to the skewness of non-change points, other metrics such as accuracy will always produce better results. This evaluation metric focuses only on the change point detection rather than the non-change point detection.

There are two types of evaluations carried out in this thesis, model evaluation and business case evaluation.

- Model evaluation: The sequence of observation of all the vehicles are divided into sub-sequences using the sliding window method. In this evaluation, the model performance is evaluated based on its ability to detect the change point in the sub-sequence irrespective of the order of that sub-sequence in the vehicle.
- Business case evaluation: In this evaluation, the model performance is evaluated based on its ability to detect the change point in complete sequence of the vehicle. Hence, the detections on the sub-sequences of every vehicle were grouped together to evaluate the performance.

3.7 Experiments

For all the experiments, stratified cross-validation with three folds has been applied and the results considered were the average of all the folds.

3.7.1 Sequence Classification

In this approach, the input is a sequence and output is 0 or 1. Thus *many input single output* architecture of RNN was chosen for this. The input layer was tested with different variants of RNN such as simple RNN, BRNN, LSTM and BLSTM. The output layer was a fully connected layer with *softmax* activation function. The hyper parameter used for the above mentioned variants were number of layers n=1, number of neurons N = 10, learning rate = 0.001, epochs = 500, batch size = 200.

In case of the synthesised dataset, different variants of RNN such as simple RNN, BRNN, LSTM and BLSTM with *many input single output* architecture of RNN were tested, as similar to the real dataset. The hyperparameters used for this dataset were number of layers n=1, num-

ber of neurons N = 5, learning rate = 0.001, epochs = 500, batch size = 200.

3.7.2 Sequence Detection

As explained in the section 2.7.1, there are two approaches in this task, multi-label classification and multi-class classification. The configurations of the model were kept the same for both the approaches, in order to compare the results.

The input layer was tested with different variants of RNN such as simple RNN, BRNN, LSTM and BLSTM. In case of multi-class classification, the *many input single ouput* architecture was chosen with the output layer was a fully connected layer with *softmax* activation function and 6 neurons, where each neuron representing each class. In case of multi-label classification, the *many input single ouput* architecture with the output layer was a fully connected layer with *sigmoid* activation function and 1 neuron per observation of the sequence. The hyperparameter used for the above mentioned variants were number of layers n=1, number of neurons N = 10, learning rate = 0.001, epochs = 500, batch size = 200. The BRNN and BLSTM variants are also tested with another layer of a same number of neurons.

In case of synthesised dataset, similar to real dataset, multi-class classification with the *many input single ouput* architecture and multi-label classification with *many input single ouput* architecture was tested. The hyper parameters used for this dataset were number of layers n=1, number of neurons N = 10, learning rate = 0.001, epochs = 500, batch size = 200.

Chapter 4

Results

In this chapter, the performance of different architectures of the RNN on synthesised and real datasets are shown. For the real dataset, the results of the evaluation of model performance and business case performance, as discussed in the section 3.6, are shown. In this thesis, there are two classes such as change points and non-change points. The evaluation metric used in thesis to evaluate the performance of the models in detecting the change points class were precision, recall and f1-score.

4.1 Synthesised Dataset

As discussed in section 2.7, the sequence detection and classification models were evaluated on this dataset and results are shown below.

4.1.1 Model Evaluation

Sequence Detection

In the sequence detection task, there were two different approaches, multi-class classification with many input single output architecture (MC-MISO) and multi-label classification with many input many outputs architecture (ML-MIMO) (section 2.7.1). The best performance in both these approaches was given by the BLSTM with one layer of 5 neurons and the results of them are shown in the Figure 4.1. As shown in the Figure 4.1, the MC-MISO has shown 5% better f1-score than ML-MIMO. Even though both the architectures having 100% recall value,

the precision of ML-MIMO architecture was 10% lesser than the MC-MISO.

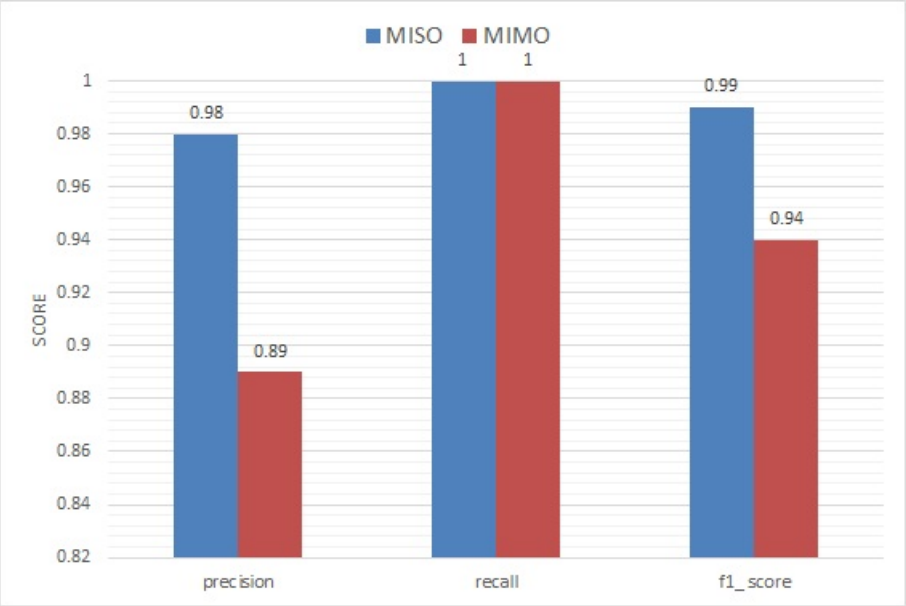


Figure 4.1: Comparison of best performing models in MISO and MIMO architectures on the synthesised dataset for sequence detection task

The performance of these models at each position of the change point is shown in the Figure 4.2. As shown in this figure, MC-MISO architecture was constant result over all the positions whereas ML-MIMO has lower precision in the middle positions such as 2, 3 & 4.

Sequence Classification

In the sequence classification task, the binary classification model with many input single output architecture (BC-MISO), as discussed section 2.7.2, performance was investigated along with the random model. The results are shown in the Figure 4.3. As we can notice in the figure, the BC-MISO has captured all the change points with 98% precision which is leading to 99% of f1-score.

4.1.2 Business Evaluation

In order to have generalised observation of techniques, the synthesised dataset was also evaluated under business conditions,as discussed in

Figure 4.2: Comparison of best performing models in MISO and MIMO architectures at different positions on the synthesised dataset for sequence detecion task

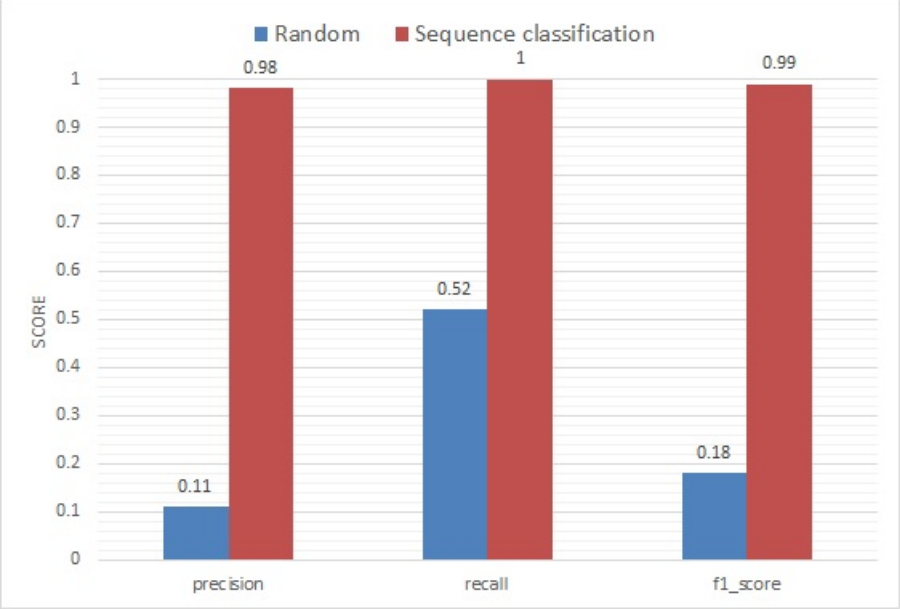


Figure 4.3: Comparison of best performing models in MISO architecture on the synthesised dataset for sequence classification task

section 3.6.

Sequence Detection

The best model in the sequence detection task was chosen for the business evaluation and the results are shown in the Figure 4.4.

Figure 4.4: Under business scenario comparison of best performing models in MISO and MIMO architectures on the synthesised dataset for sequence detection task

4.1.3 Ensemble

For the synthesised dataset, the results after ensembling is shown in the Figure 4.5. The ensemble technique has improved the precision of MISO architecture to 1.6% and MIMO architecture to 4.2% when compared to results without ensembling.

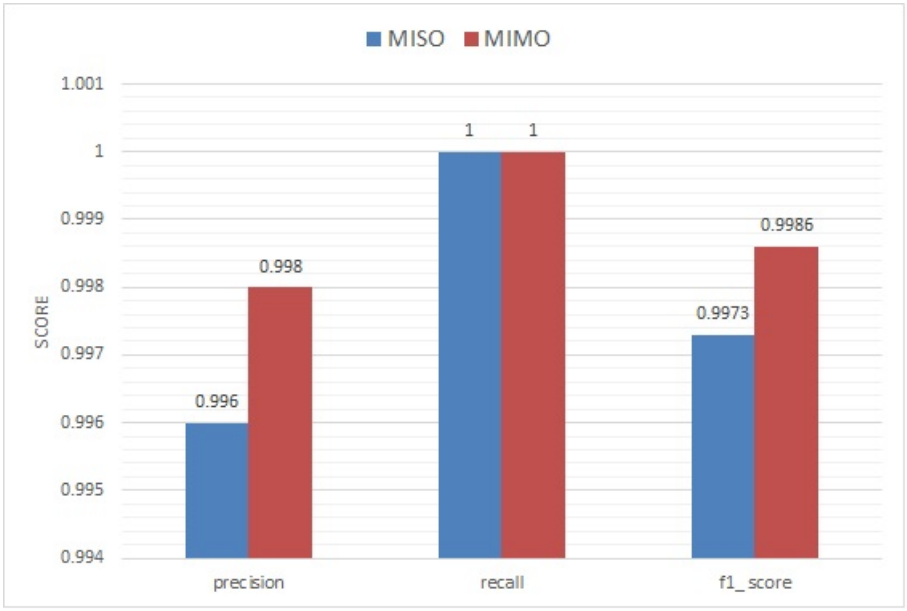


Figure 4.5: After ensembling, under business scenario comparison of best performing models in MISO and MIMO architectures on the synthesised dataset for sequence detection task

4.2 Scania Dataset

4.2.1 Model Evaluation

Sequence detection

In the sequence detection task, there were two different approaches, MC-MISO (section 2.7.1) and ML-MIMO (section 2.7.2). The best performance in both these approaches were given by the BRNN with 2 layers of 10 neurons each and the results of them are shown in the Figure 4.6.

The MC-MISO having the same score of precision and recall, whereas the ML-MIMO having higher precision than the recall but providing the same f1-score.

As the change point position could be present at any position of the sequence, the performance of these approaches at different positions were analysed as shown in the Figure 4.7. The performance of both the models was poor at the last position. The precision of the ML-MIMO architecture is better than MC-MISO at all the positions but in case of the recall, MC-MISO has performed better than ML-MIMO.

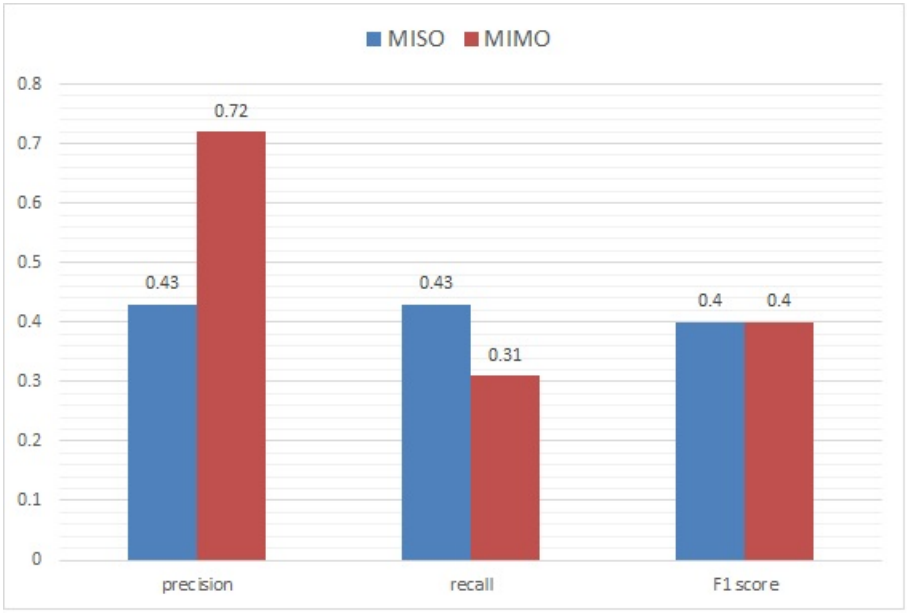


Figure 4.6: Comparison of best performing models in MISO and MIMO architectures on the real dataset for sequence detection task

Figure 4.7: Comparison of best performing models in MISO and MIMO architectures at different positions on the real dataset for sequence detection task

Sequence classification

For the sequence classification approach, the BC-MISO approach was used and the best results were obtained with BLSTM variant with a single layer of 10 neurons. This approach's performance is evaluated only from the model perspective and not from the business perspective as it does not directly suffice the business requirements. The

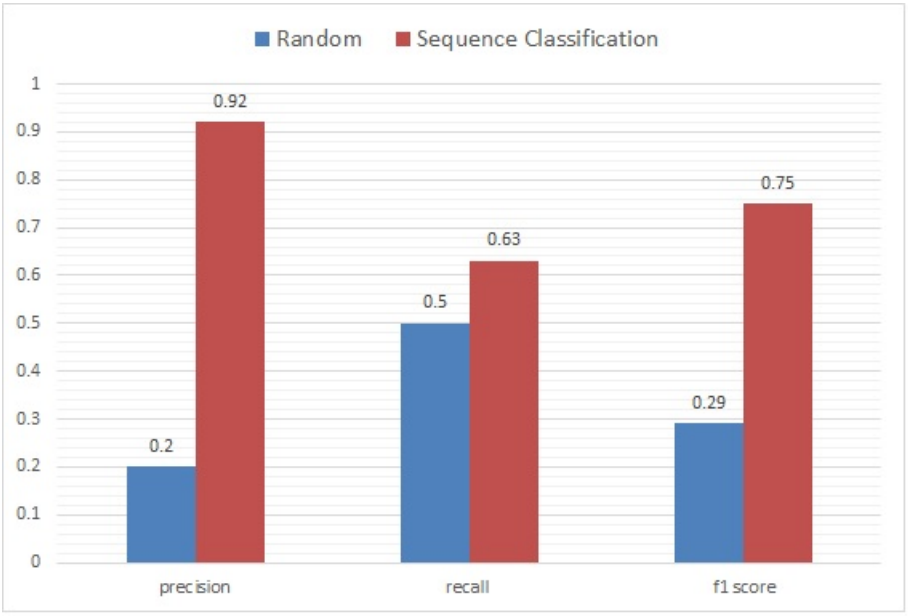


Figure 4.8: Comparison of the best model in MISO architecture with the random model on the real dataset for sequence classification task

sequence classification approach has 158.6% increase in the f1-score when compared to the random model. This model has higher precision compared to the recall.

4.2.2 Business Case Evaluation

The best models from the model performance have also shown the best performance in the business case evaluation, as discussed in section 3.6.

Sequence Detection

The performance of the best models were compared with the rule-based models, which is discussed in the section 3.5, are shown in the

Figure 4.9. As shown in the figure, the ML-MIMO approach has better performance when compared to the MC-MISO approach because of very high precision than the other.

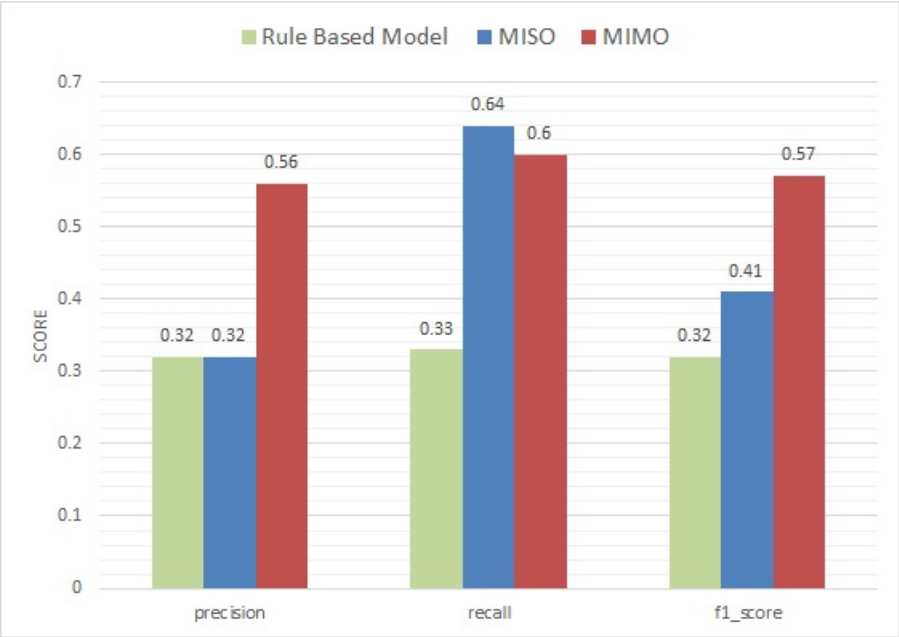


Figure 4.9: Business case comparison of best performing models in MISO and MIMO architectures with rule based model on the real dataset for sequence detection task

4.2.3 Ensemble

As explained in the section 2.7.3, the stacking ensemble method was used to combine the results of the best models of MISO and MIMO from sequence detection and sequence classification.

For the real dataset, the results of the sequence classification were ensembled with the best model of MC-MISO and ML-MIMO approaches, as discussed in section 2.7.3. The results after ensembling is as shown in the Figure 4.10. After applying ensemble technique, there is a 93.5% increase in the precision in the MC-MISO and also minor increase in the recall in both the approaches.

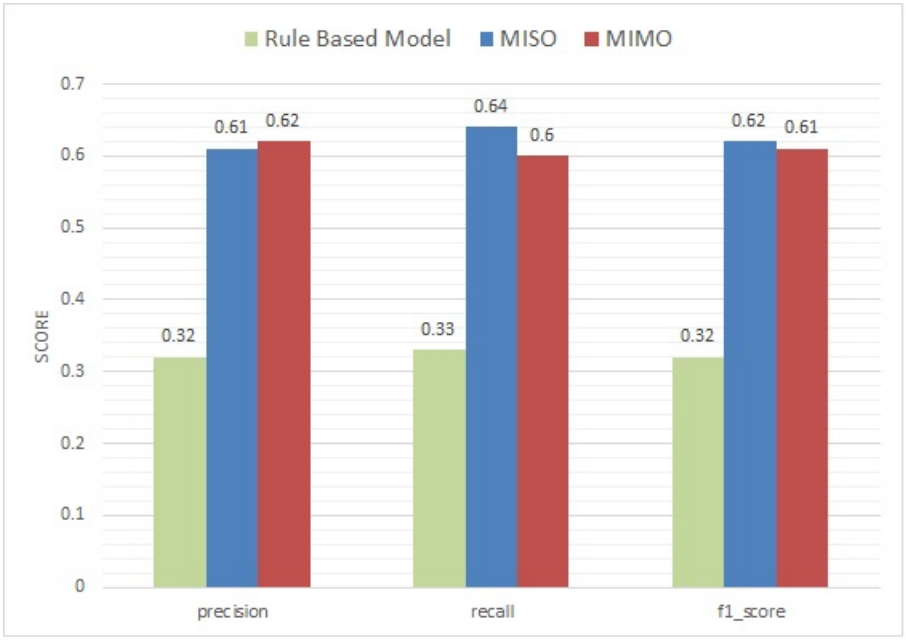


Figure 4.10: After applying ensembling technique, business case comparison of best performing models of MISO and MIMO architectures with rule based model on the real dataset for the sequence detection task

Visualisation of detection

The main business objective was to detect the DPF change in the vehicle. Thus the performance of the best sequence detection model was visualised over the sequence of observations of every vehicle. The MC-MISO based model was chosen as it obtained the best performance as shown in the Figure 4.10. The result on a vehicle before applying ensemble technique is shown in the Figure 4.11. As we can notice, the model has detected the actual change point along with two other false positives.

The result on a vehicle after applying ensemble technique is shown in the Figure 4.12. After applying the ensemble technique, the false positives have been removed but still, the correct prediction was maintained. The best model failed to detect the change point at certain vehicles and one of them is shown in the Figure 4.13.

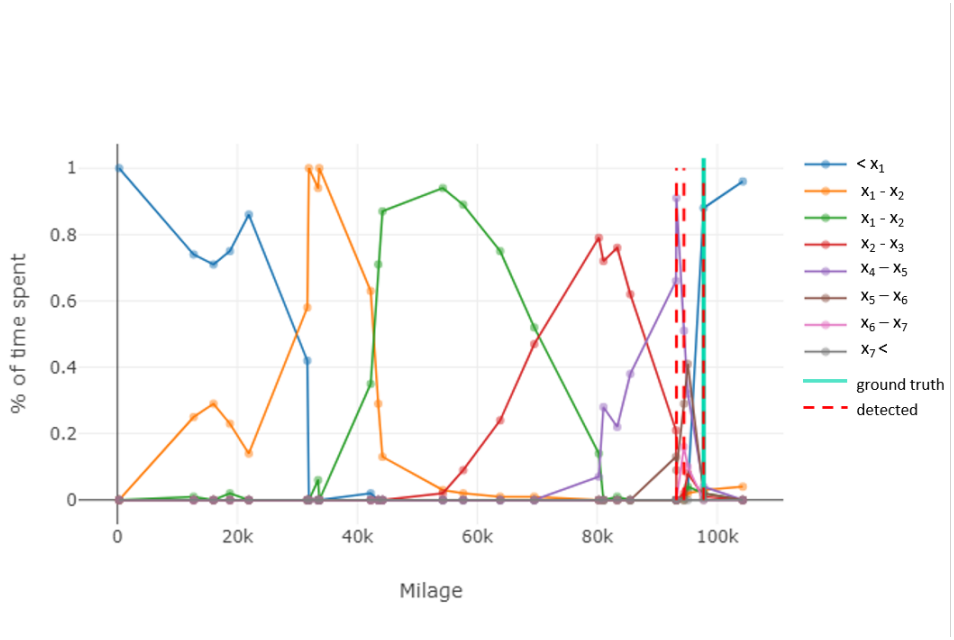


Figure 4.11: Detection of best model on a vehicle before applying ensemble technique

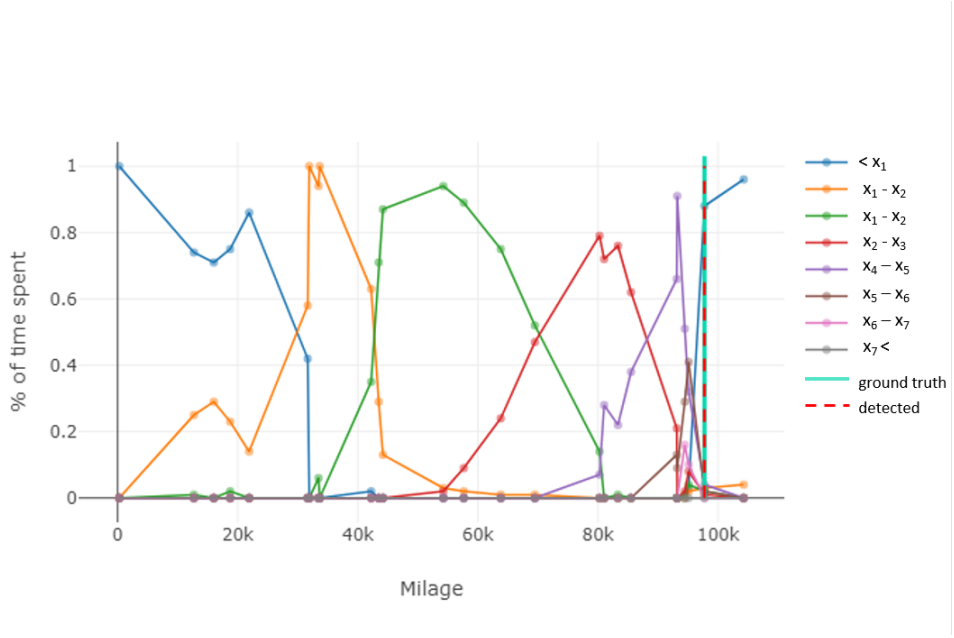


Figure 4.12: Detection of best model on a vehicle after applying ensemble technique

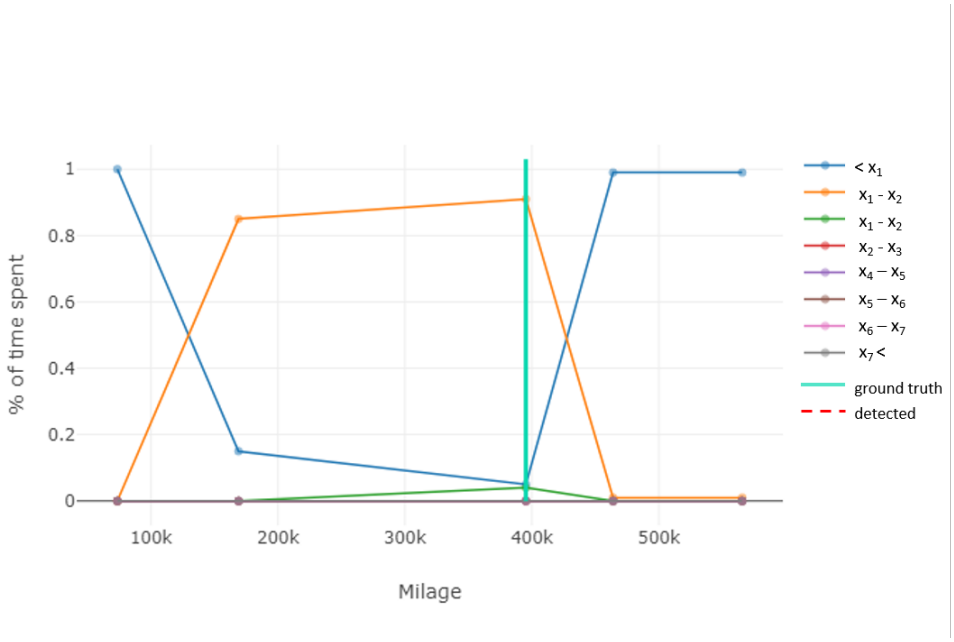


Figure 4.13: Poor performance of best model on a vehicle after applying ensemble technique

Chapter 5

Discussion

In this chapter, the results of the experiments were analysed. The comparison of performance of different architectures in sequence detection and classification approaches were discussed, followed by the effect of ensemble technique and bidirectional layers.

5.1 Performance of different architectures in sequence detection

The MC-MISO architecture and ML-MIMO architecture has almost similar performance on both the real and synthesised dataset, as shown in the Figure 4.6 and Figure 4.1 respectively.

5.1.1 Synthesised Dataset

In the Figure 4.1, both the architectures were able to detect all the change points correctly. As it was synthesised dataset, the reason might be because the sequence with change point observations was standing out from other sequences which do not have the change points in them. Thus, both the architectures were able to clearly detect them. The precision of the ML-MIMO architecture was 10% lesser than other architecture. This might be due to the choice of the threshold. The chosen threshold detected few more change points which did not actually change points.

5.1.2 Scania Dataset

In the Figure 4.6, the ML-MIMO architecture displays a higher value of precision and lower value of recall. This behaviour was due to the detection of few correct change points without many false positives. In the same Figure 4.6, the MC-MISO architecture has 67% decrease in precision and 39% increase in recall with a comparison to the ML-MIMO architecture which indicates that it detected much more actual change points but with more false positives. This behaviour was also supported in the Figure 4.7, as the precision of the ML-MIMO, was higher in all the positions but the recall was lower.

The reason for the ML-MIMO architecture having less false positives and low recall might be due to the high global threshold usage. The high global threshold leads to detection of only the points which shows higher difference among its neighbouring observations in the sequence. Thus, it also leads to the detection of only a few points because not all the change points in this real dataset show higher difference among its neighbouring observations.

The reason for the MC-MISO architecture having low precision but better recall than the other architecture might be due to the ability to sense all the minor change values as change points, as we did not declare any threshold value which we did for the other architecture.

In the Figure 4.7, both the architectures have shown low performance at the first and last position when compared to the middle positions 2, 3 & 4. The possible reason could be that the last position performance has influence from past states and current state but not future state and for the first position the performance was influenced from current and future states but not past states. This indicates that the change points in this data are hard to differentiate without a future point.

5.2 Performance of different architectures in sequence classification

The sequence classification approach was investigated in this thesis as it is another common task in change point detection problems. This requirement of this approach was to classify whether a sequence has change point or not. Since the output of this approach is a binary

value, only many input single output (BC-MISO) architecture was tested. The results of this model were compared with the random model, as explained in the section 3.5.

The best sequence classification models have more than 100% better performance than the random model in both the real and synthesised dataset. This indicates that the RNN has the huge ability in detecting the presence of change point in the sequence. The f1 score of the best sequence detection model on real dataset was 0.75 wheres in case of the synthesised dataset it was 0.99. The high precision and low recall of the model on the real dataset means that there were few points which were hard to distinguish from the sequences that do not have change point in them.

The results could be improvised by increasing more neurons or stacking another layer because it would take the points to higher dimension which may lead to clear separation of the sequences with change points and without change points.

5.3 Performance in business scenario

The business requirement was to detect when the DPF has been changed in the sequence of observations of the vehicle. The best performing models from both the architectures were evaluated as per the business conditions, as explained in the section 3.5. The ML-MIMO model has shown 39% better f1 score than the MC-MISO model, as shown in the Figure 4.9. Both the models have the almost similar recall, but MC-MISO model has precision just equal to the rule-based model. This low precision behaviour of the MC-MISO model was also observed in model performance evaluation, as shown in the Figure 4.6. The reason for the poor precision would be because of its ability to detect all minor changes in the sequence. The example of the MC-MISO behaviour was shown in the Figure 4.11 which shows the detection of two false positive detections (red dashed lines) along with one correct detection.

The ensemble technique to combine the output of BC-MISO type sequence classification model with MC-MISO and MC-MIMO type sequence detection models has improved the performance of both the models, as shown in the Figure 4.10. The precision of the MC-MISO has shown a 93.5% increase and the recall of both the models has remained same. This clearly indicates that the ensembling helped in re-

moving the false positives has sequence classification model has efficiently learnt the sequences with filter change and non-filter change. Thus the same MC-MISO model after applying the ensemble technique has shown improved performance on the same vehicle as shown in the Figure 4.12, as it indicates the removal of false positives but maintaining the correct prediction.

This similar behaviour was observed when synthesised dataset was analysed under business scenario, as shown in the Figure 4.5. As shown in this figure, there was a imminent increase in the precision without affecting the recall because of the removal of false positives only.

5.4 Effect of ensemble technique

Based on the business requirements, there are two main tasks in change point detection problem, i) Does this sequence has change point? and ii) Where is the change point in the sequence?. Even though the earlier task is not related to the business requirement of the thesis, it has been investigated since it was a common task in various applications.

As like the saying, it is always better to ask opinion from more than one expert before making any decision, ensemble technique provides better performance than the single model. The main reason was that each model could draw the decision boundaries at different dimension space which mean each model something different from other. Thus each model could not be perfect by themselves but when they are combined with suitable logic, they could become one. In this thesis, we took AND gate function to combine the results of sequence detection and sequence classification model.

We observed that the sequence classification has comparatively better performance than the sequence detection model even though both were trained on same data, on the other hand, best model in sequence detection were correctly detecting the change points but with few false positives. Thus we thought to combine both the models together to test whether the performance could be improved and resulted in the same, as shown in the Figure 4.10 and in the Figure 4.5.

This could be improvised by testing with different combining functions based on the requirement and understanding of what each model have learnt well.

5.4.1 Other ensemble approach

In this thesis, the sequence detection model and classification model was trained individually and then ensembled to improve the performance of sequence detection. The other possible approach could be to train the sequence detection model only on the samples which are classified as sequence with change points by sequence classification model. This approach reduces the ensembling computation.

5.5 Performance of bidirectional layers

The bidirectional layers provided the best performance in both sequence detection and classification approach. The ability of the bidirectional layers to get the information from the past and future states was the key reason for this performance. The behaviour of the DPF and signal synthesised dataset were having a temporal dependency between the observations which made the change points not be random. Hence, the future observations have bigger influence detection of the current state and this was leveraged by using the bidirectional layers.

5.6 Analysis of the metrics

The metrics used in this thesis was *f1 score*. This metric is a harmonic mean of precision and recall. The models with same f1 score will not have the same precision and recall. There are three kinds of possibilities to get a same f1 score high precision & low recall, low precision & high recall, and same precision & recall. Thus, based on the requirement we have to choose the model.

This kind of metric works better in case of sequence classification tasks as the model needs to answer the presence of change point, as yes or no. In case of, sequence detection as the model needs to answer where is the change point in the sequence, another possible metric to evaluate would be *root mean square error*(RMSE). The RMSE metric calculates how far is the detected change point from the actual change point. If the detected change point is farther from the actual change point then the RMSE will be larger and it will be smaller if it closer to the actual change point. This RMSE metric is more preferable if the

sequence has only one change point and the requirement is to detect change point close enough to the actual point. Also, the model has to be designed in such a way that output will be the position of the change point observation.

5.7 Future Work

The original sequences in both the real and synthesised dataset were sliced into sub-sequences of length five using sliding window technique. Hence, it would be interesting to test how varying the window length could affect the performance of the RNN.

The models designed in this thesis were tested with sequences of short length even though the original sequences were of variable length. The RNN has shown prominent results with varying and long length sequences in case of natural language processing problems[52, 1]. It would be interesting to test the RNN by feeding the sequences of varying length directly to detect the change point in the sequence.

In the MIMO architecture based model of this thesis sigmoid activation function was used at the output layer. This lead us to choose a threshold value, to decided an observation was change point or not. Another possible approach would be to use softmax as the activation layer at the output, which does not require us to choose threshold value. Hence it could be considered in the future work.

Chapter 6

Concluding Remarks

6.1 Conclusions

In this thesis, the performance of the MISO and MIMO architectures of RNN in recognising the change points was investigated. The experiments included addition of more features, scaling of data and with different variants of RNN such as simple RNN, LSTM, bidirectional RNN and bidirectional LSTM. The metric used to evaluate the architectures were precision, recall and f1-score.

The real dataset was collected from hundreds of trucks which has different ages, sensor manufacturers and under different environmental conditions. The data was extracted from the trucks during the workshop visits. On average, a truck visits workshop two or three times per year which varies based on the type of customer, vehicle and contract. In between these workshop visits, the data gets accumulated which would be reset after the extraction. The reset operation was manual hence this operation might be skipped sometimes. This accumulated data was cancelled out during the data processing with other signals stored in the database. The average data points in a sequence for a truck was around twenty one. All the sequences were converted into fixed length of five. This dataset showcases the exponential increase and approximately sudden decrease behaviour. The results of the thesis gives the conclusions as follows

- MISO and MIMO architecture has nearly same performance with respect to f1-score metric. The reason for this performance was because of the experiments conducted and compared with similar configuration for both the architectures. The bidirectional

models had better performance when compared to simple RNN and LSTM models. This was because the bidirectional models considers all the points in the future and past to predict whether a data point is a change point or not. This ability of the bidirectional models outperformed the other models.

- MISO architecture has higher recall than precision. The reason for this behaviour was because of the usage of softmax function at the last layer to determine which position (class) has change point. This architecture picks up even minimal change in the normal behaviour which might be because of noise.

- MIMO architecture has higher precision than recall. The reason for this behaviour was because of the usage of sigmoid function at the last layer to determine which position has change point. In this case, threshold has to specified in order to classify a data point as change point. The high value of threshold made the model to detect the data points with higher deviation. If the threshold is lowered, then this architecture would also exhibit higher recall than precision.

- The stacking ensemble technique of combining the results of sequence classification with sequence detection enhances the performance by removing most of the false positives. The reason for this performance was because the sequence classification learnt the pattern in sequences having real change points and excluded the noise. When this results were combined with sequence detection, it removed the false positives and enhanced the performance.

A similar previous work [33] has also exhibited the boost in performance of detection by drop in false positives because of ensemble methods. For the reliability of the conclusions, similar experiments were conducted with another synthetic dataset. The results of that dataset also showcased similar conclusions except the recall was one(100%) for both the architectures as the change points were artificially induced.

6.2 Future Work

The original sequences in both the real and synthesised dataset were sliced into sub-sequences of length five using sliding window technique. Hence, it would be interesting to test how varying the window length could affect the performance of the RNN.

The models designed in this thesis were tested with sequences of short length even though the original sequences were of variable length. The RNN has shown prominent results with varying and long length sequences in case of natural language processing problems[52, 1]. It would be interesting to test the RNN by feeding the sequences of varying length directly to detect the change point in the sequence.

In the MIMO architecture based model of this thesis sigmoid activation function was used at the output layer. This lead us to choose a threshold value, to decided an observation was change point or not. Another possible approach would be to use softmax as the activation layer at the output, which does not require us to choose threshold value. Hence it could be considered in the future work.

Bibliography

[1] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. "Neural machine translation by jointly learning to align and translate". In: *arXiv preprint arXiv:1409.0473* (2014).

[2] Pierre Baldi et al. "Exploiting the past and the future in protein secondary structure prediction". In: *Bioinformatics* 15.11 (1999), pp. 937–946.

[3] Michèle Basseville, Igor V Nikiforov, et al. *Detection of abrupt changes: theory and application*. Vol. 104. Prentice Hall Englewood Cliffs, 1993.

[4] Chris Bishop, Christopher M Bishop, et al. *Neural networks for pattern recognition*. Oxford university press, 1995.

[5] Loıc Bontemps, James McDermott, Nhien-An Le-Khac, et al. "Collective anomaly detection based on long short-term memory recurrent neural networks". In: *International Conference on Future Data and Security Engineering*. Springer. 2016, pp. 141–152.

[6] Leo Breiman. "Bias, variance, and arcing classifiers". In: (1996).

[7] Denny Britz. *Recurrent Neural Networks Tutorial, Part 1 – Introduction to RNNs*. July 2016. URL: http://www.wildml.com/2015/09/recurrent-neural-networks-tutorial-part-1-introduction-to-rnns/.

[8] Sucheta Chauhan and Lovekesh Vig. "Anomaly detection in ECG time signals via deep long short-term memory networks". In: *Data Science and Advanced Analytics (DSAA), 2015. 36678 2015. IEEE International Conference on*. IEEE. 2015, pp. 1–7.

[9] Min Cheng et al. "MS-LSTM: A multi-scale LSTM model for BGP anomaly detection". In: *Network Protocols (ICNP), 2016 IEEE 24th International Conference on*. IEEE. 2016, pp. 1–6.

[10] Md Foezur Rahman Chowdhury, S-A Selouani, and D O'Shaughnessy. "Bayesian on-line spectral change point detection: a soft computing approach for on-line ASR". In: *International Journal of Speech Technology* 15.1 (2012), pp. 5–23.

[11] Ian Cleland et al. "Evaluation of prompted annotation of activity data recorded from a smart phone". In: *Sensors* 14.9 (2014), pp. 15861–15879.

[12] *Confusion Matrix*. URL: `https://rasbt.github.io/mlxtend/user_guide/evaluate/confusion_matrix/`.

[13] Frédéric Desobry, Manuel Davy, and Christian Doncarli. "An online kernel change detection algorithm". In: *IEEE Transactions on Signal Processing* 53.8 (2005), pp. 2961–2974.

[14] Thomas G Dietterich. "Ensemble learning". In: *The handbook of brain theory and neural networks* 2 (2002), pp. 110–125.

[15] Jeffrey Donahue et al. "Long-term recurrent convolutional networks for visual recognition and description". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015, pp. 2625–2634.

[16] Jean-François Ducré-Robitaille, Lucie A Vincent, and Gilles Boulet. "Comparison of techniques for detection of discontinuities in temperature series". In: *International Journal of Climatology* 23.9 (2003), pp. 1087–1101.

[17] Elisabetta Fersini, Enza Messina, and Federico Alberto Pozzi. "Sentiment analysis: Bayesian ensemble learning". In: *Decision support systems* 68 (2014), pp. 26–38.

[18] Kyle D Feuz et al. "Automated detection of activity transitions for prompting". In: *IEEE transactions on human-machine systems* 45.5 (2015), pp. 575–585.

[19] Yoav Freund, Robert E Schapire, et al. "Experiments with a new boosting algorithm". In: *Icml*. Vol. 96. Bari, Italy. 1996, pp. 148–156.

[20] Alex Graves. "Generating sequences with recurrent neural networks". In: *arXiv preprint arXiv:1308.0850* (2013).

[21] Alex Graves, Abdel-rahman Mohamed, and Geoffrey Hinton. "Speech recognition with deep recurrent neural networks". In: *Acoustics, speech and signal processing (icassp), 2013 ieee international conference on*. IEEE. 2013, pp. 6645–6649.

[22] Alex Graves and Jürgen Schmidhuber. "Framewise phoneme classification with bidirectional LSTM and other neural network architectures". In: *Neural Networks* 18.5-6 (2005), pp. 602–610.

[23] Alex Graves et al. "Unconstrained on-line handwriting recognition with recurrent neural networks". In: *Advances in neural information processing systems*. 2008, pp. 577–584.

[24] Valery Guralnik and Jaideep Srivastava. "Event detection from time series data". In: *Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM. 1999, pp. 33–42.

[25] Fredrik Gustafsson. "The marginalized likelihood ratio test for detecting abrupt changes". In: *IEEE Transactions on automatic control* 41.1 (1996), pp. 66–78.

[26] Anne Håkansson. "Portal of research methods and methodologies for research projects and degree projects". In: *The 2013 World Congress in Computer Science, Computer Engineering, and Applied Computing WORLDCOMP 2013; Las Vegas, Nevada, USA, 22-25 July*. CSREA Press USA. 2013, pp. 67–73.

[27] Manhyung Han, Young-Koo Lee, Sungyoung Lee, et al. "Comprehensive context recognizer based on multimodal sensors in a smartphone". In: *Sensors* 12.9 (2012), pp. 12588–12605.

[28] Shohei Hido et al. "Statistical outlier detection using direct density ratio estimation". In: *Knowledge and information systems* 26.2 (2011), pp. 309–336.

[29] Sepp Hochreiter and Jürgen Schmidhuber. "Long short-term memory". In: *Neural computation* 9.8 (1997), pp. 1735–1780.

[30] Sepp Hochreiter and Jürgen Schmidhuber. "LSTM can solve hard long time lag problems". In: *Advances in neural information processing systems*. 1997, pp. 473–479.

[31] Naoki Itoh and Jürgen Kurths. "Change-point detection of climate time series by nonparametric method". In: *Proceedings of the world congress on engineering and computer science*. Vol. 1. Citeseer. 2010, pp. 20–22.

[32] Andrej Karpathy and Li Fei-Fei. "Deep visual-semantic alignments for generating image descriptions". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015, pp. 3128–3137.

[33] Gyuwan Kim et al. "LSTM-based system-call language modeling and robust ensemble method for designing host-based intrusion detection systems". In: *arXiv preprint arXiv:1611.01726* (2016).

[34] Terran Lane and Carla E Brodley. "Temporal sequence learning and data reduction for anomaly detection". In: *ACM Transactions on Information and System Security (TISSEC)* 2.3 (1999), pp. 295–331.

[35] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. "Deep learning". In: *nature* 521.7553 (2015), p. 436.

[36] Pankaj Malhotra et al. "Long short term memory networks for anomaly detection in time series". In: *Proceedings*. Presses universitaires de Louvain. 2015, p. 89.

[37] Pankaj Malhotra et al. "LSTM-based encoder-decoder for multi-sensor anomaly detection". In: *arXiv preprint arXiv:1607.00148* (2016).

[38] Rakesh Malladi, Giridhar P Kalamangalam, and Behnaam Aazhang. "Online Bayesian change point detection algorithms for segmentation of epileptic activity". In: *Signals, Systems and Computers, 2013 Asilomar Conference on*. IEEE. 2013, pp. 1833–1837.

[39] Erik Marchi et al. "Non-linear prediction with LSTM recurrent neural networks for acoustic novelty detection". In: *Neural Networks (IJCNN), 2015 International Joint Conference on*. IEEE. 2015, pp. 1–7.

[40] Warren S McCulloch and Walter Pitts. "A logical calculus of the ideas immanent in nervous activity". In: *The bulletin of mathematical biophysics* 5.4 (1943), pp. 115–133.

[41] Tomáš Mikolov et al. “Recurrent neural network based language model”. In: *Eleventh Annual Conference of the International Speech Communication Association*. 2010.

[42] Michael A. Nielsen. *Neural Networks and Deep Learning*. misc. 2018. URL: `http://neuralnetworksanddeeplearning.com/`.

[43] David Opitz and Richard Maclin. “Popular ensemble methods: An empirical study”. In: *Journal of artificial intelligence research* 11 (1999), pp. 169–198.

[44] Ewan S Page. “Continuous inspection schemes”. In: *Biometrika* 41.1/2 (1954), pp. 100–115.

[45] Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. “On the difficulty of training recurrent neural networks”. In: *International Conference on Machine Learning*. 2013, pp. 1310–1318.

[46] Sasank Reddy et al. “Using mobile phones to determine transportation modes”. In: *ACM Transactions on Sensor Networks (TOSN)* 6.2 (2010), p. 13.

[47] Jaxk Reeves et al. “A review and comparison of changepoint detection techniques for climate data”. In: *Journal of Applied Meteorology and Climatology* 46.6 (2007), pp. 900–915.

[48] David E Rumelhart. “Learning internal represectations by error propagation”. In: *Parallel distributed processing* 1.8 (1986).

[49] David Rybach et al. “Audio segmentation for speech recognition using segment features”. In: *Acoustics, Speech and Signal Processing, 2009. ICASSP 2009. IEEE International Conference on*. IEEE. 2009, pp. 4197–4200.

[50] Mike Schuster and Kuldip K Paliwal. “Bidirectional recurrent neural networks”. In: *IEEE Transactions on Signal Processing* 45.11 (1997), pp. 2673–2681.

[51] Ralf C Staudemeyer and Christian W Omlin. “Evaluating performance of long short-term memory recurrent neural networks on intrusion detection data”. In: *Proceedings of the South African Institute for Computer Scientists and Information Technologists Conference*. ACM. 2013, pp. 218–224.

[52] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. "Sequence to sequence learning with neural networks". In: *Advances in neural information processing systems*. 2014, pp. 3104–3112.

[53] Jun-ichi Takeuchi and Kenji Yamanishi. "A unifying framework for detecting outliers and change points from time series". In: *IEEE transactions on Knowledge and Data Engineering* 18.4 (2006), pp. 482–492.

[54] *the unreasonable effectiveness of recurrent neural networks*. URL: http://karpathy.github.io/2015/05/21/rnn-effectiveness/.

[55] Ujjwalkarn. *A Quick Introduction to Neural Networks*. Aug. 2016. URL: https://ujjwalkarn.me/2016/08/09/quick-intro-neural-networks/.

[56] Oriol Vinyals et al. "Show and tell: A neural image caption generator". In: *Computer Vision and Pattern Recognition (CVPR), 2015 IEEE Conference on*. IEEE. 2015, pp. 3156–3164.

[57] Li Wei and Eamonn Keogh. "Semi-supervised time series classification". In: *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM. 2006, pp. 748–753.

[58] David H Wolpert. "Stacked generalization". In: *Neural networks* 5.2 (1992), pp. 241–259.

[59] Ping Yang, Guy Dumont, and John Mark Ansermino. "Adaptive change detection in heart rate trend monitoring in anesthetized children". In: *IEEE transactions on biomedical engineering* 53.11 (2006), pp. 2211–2219.

[60] Yu Zheng et al. "Learning transportation mode from raw gps data for geographic applications on the web". In: *Proceedings of the 17th international conference on World Wide Web*. ACM. 2008, pp. 247–256.

[61] Yu Zheng et al. "Understanding mobility based on GPS data". In: *Proceedings of the 10th international conference on Ubiquitous computing*. ACM. 2008, pp. 312–321.

[62] Zhi-Hua Zhou. "Ensemble learning". In: *Encyclopedia of biometrics* (2015), pp. 411–416.

TRITA EECS-EX-2018:584