Learning-based Vehicle Diagnostic and Prognostic System Utilizing Natural Language Processing

By

ALI KHODADI
DISSERTATION

Submitted in partial satisfaction of the requirements for the degree of

DOCTOR OF PHILOSOPHY

in

Electrical and Computer Engineering

in the

OFFICE OF GRADUATE STUDIES

of the

UNIVERSITY OF CALIFORNIA

DAVIS

Approved:

Chen-Nee Chuah, Chair

Soheil Ghiasi

Michael Zhang

Committee in Charge

2022

To my wife and daughter:

Your confidence in me and warm words of encouragement kept me going through uncertain moments.

Contents

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

Initial fault detection and diagnostics are essential elements to improve the efficiency, safety, and stability of vehicle operation. Diagnostics can make direct and indirect financial impacts on service and support entities in place for the vehicle. Remote diagnostics can reduce vehicle downtime in service centers and increase customer satisfaction, primarily when conducting over-the-air updates or telephone lines. In order to troubleshoot a vehicle, specific tools can be used to look up failure codes stored in vehicle controllers or manually gather failure symptoms through customer service hotlines and remote service technicians. The overall gathering of data, deciphering, and execution of any repairs still consumes precious time and may suffer from potential human errors. Recently, numerous studies have investigated data-driven approaches to improve vehicle diagnostics using available vehicle data.

This study investigates a machine learning pipeline to improve automated vehicle diagnostics and prognostics. Using Natural Language Processing (NLP), we demonstrate a comprehensive model to extract the customer and agent interactions from repair-service call transcriptions. This dissertation applies Machine Learning (ML) algorithms to identify accurate failure reports and claims. Also, it classifies the service requests to the proper service department and utilizes the historical service information along with current customer claims to identify possible failed vehicle parts.

First, NLP techniques are used to automate the task of crucial information extraction from free-text failure reports (generated within customers' calls to the service department). We have introduced an NLP taxonomy in the automotive domain since known NLP techniques had a weak performance on such texts. We have shown that domain-based NLP processing and feature extraction can help to extract meaningful information from the reports.

Deep learning algorithms are employed to validate service requests and filter vague or misleading claims. Various classification algorithms are implemented to classify service requests so that valid service requests can be directed to the relevant service department. We proposed to employ Bidirectional Long Short-term Memory (BiLSTM), along with Convolution Neural Network (CNN) model, which shows more than 18% performance improvement in validating service requests compared to average technicians' capabilities. Furthermore, using domain-based NLP techniques at preprocessing and feature extraction stages along with CNN-BiLSTM-based request validation enhanced the performance of the Gradient Tree Boosting (GTB) service classification model. The performance parameter of the Receiver Operating Characteristic Area Under the Curve (ROC-AUC) reached 0.82.

Next, we performed automated failure classification on extracted data to route the claims to the proper service departments. By introducing optimized feature extraction and classification methods, requests can be forwarded to the correct departments with 80% accuracy. This method exceeds the 60% baseline accuracy for an average customer service technician. NLP analysis can also generate technical information from the text report for vehicle and component prognostics that have not been previously studied.

Finally, we proposed a novel network structure that employs a multi-variant high-dimensional Markov chain to predict the possible failed component of the next service interval to enhance the CNN-LSTM model performance. The Markov model takes advantage of historical records to identify the most efficient CNN kernels in the network structure. The proposed model significantly improved data classification efficiency in correlated historical records such as vehicle service reports. Compared to conventional CNN-LSTM models, the introduced model demonstrated

significant performance enhancement of 8% accuracy, 9% sensitivity, 11% specificity, 10% precision, and 12% f-score by reducing the false positive cases in customer claim classification.

ACKNOWLEDGMENT

I am genuinely grateful to all the individuals whose encouragement and support made completing this thesis possible.

First, I would like to express my deep gratitude to Prof. Chen-Nee Chuah, whose guidance, support, and encouragement made three of the projects presented in this thesis possible. When I joined the University of California Davis Ph.D. program in 2016 and expressed interest in this research area, Prof. Chuah accepted me as a RUBINET lab member. Since then, I have received an enormous amount of advice and knowledge from her that will be forever valued in my professional and personal life.

I would like to thank my committee members, Prof. Soheil Ghiasi and Prof. Michael Zhang, for their valuable time and input.

I would also like to thank my colleagues in the RUBINET lab for their advice and unwavering support. Their willingness to support and collaborate with my work, provide peer reviews, and offer helpful suggestions improved my work immensely.

Last but not least, my family has been there for me from the very beginning: my wife, Ellie, and my daughter Melina. Their love and support keep me going. I dedicate this thesis to them, who encouraged me to pursue my dreams and continue to be my most significant source of strength.

1 Introduction

Efficient customer service is an essential aspect of most businesses. In 2017, U.S. companies lost $75 billion through poor customer service, where customers encountered unhelpful staff or spent too much time on unresolved issues. Customer management software companies analyze customer-agent transcriptions using sentiment analysis and topic modeling methods to improve their clients' customer service. However, these approaches are not optimized to account for the sequential nature of these customer-agent interactions. For example, while knowing how many customers cancel service is essential, businesses also need to understand how agents respond to a cancellation request and how specific dialogue or agent actions may lead to a positive or negative outcome [1] as a Markov decision process (MDP).

Early diagnostics and predictive maintenance have become more critical in many industries, including automotive vehicles. It is typically hard to diagnose a failure in advance in the vehicle industry because of the limited availability of sensors and some design exertions. Moreover, more comprehensive components are integrated into the next generation of vehicles, such as autonomous vehicles. However, with the significant development in the automotive industry, it looks challenging to have an optimal diagnostics and prognostic system using the current solution that the automotive industry uses. Reference [2] illustrate the necessities of more profound vehicle diagnostic needs.

In this study, we perform comprehensive text analysis on vehicular customer service reports. We develop a comprehensive NLP pipeline and provide an automated diagnostic and prognostic solution using text-based customer and service team reports. We detail methods using NLP and ML on the reports to evaluate customer claims, address truth requests into correct service

departments, and estimate the lifetime of specific system components. The outcome of the report can be used for the following purposes:

- Recognizing invalid and vague claims.
- Helping service departments address customer requests faster and more accurately.
- Achieving improved vehicle prognostics by calculating lifetime expectations of specific components.
- Helping a customer service representative by NLP-based classification of the vehicular failure reasons, which can guide the representative towards more effective troubleshooting, better estimations of service time, or better routing of the customers to the proper department based on a predictive model trained using historical service data.

The main structure of this dissertation is as follows. We detailed and compared similar studies referenced or used in this dissertation in Chapter 2 (Background and Related Work).

Chapter 3 (Dataset) details the structure of the dataset. This section explains how a customer service request is initiated and shows the detailed formatting of each section of the data..

Chapter 4 (Natural Language Processing Taxonomy) outlines techniques and methods employed in this research to extract meaningful information from free and unstructured data. In this chapter, we detailed how we have used created domain-related techniques and modified known preprocessing and feature extraction tools to enhance the content extraction process. We also detail reasons and examples of how using generic NLP tools is not as efficient.

Chapter 5 (Validating Customer Claims) formulates and presents different statistical and deep-leaning models to separate valid customer service requests from vague or nonrelated claims. The main goal of this section is to identify true services that customers need to get addressed

appropriately. We detail each model and different evaluation metrics used to compare the models. Finally, we compare model performance using the proper evaluation metrics.

Chapter 6 introduces automated customer request routing classification. Valid customer requests identified in the previous chapter are classified into different troubleshooting groups. Each group represents a specific department that is trained and experienced provide technical support for the issue. Suited classification models are compared and evaluated for this application and detailed on how some specific models outperform others in this field.

In Chapter 7, we have introduced a pipeline on utilizing such historical relations to predict the failure rate of individual components and use such info to improve the text mining and classification models. Finally, we summarize the overall research and compare what has been achieved in detail. We also provide some insights into how the proposed pipeline can be used in other applications. Figure 1.1 summarizes the overall structure of the NLP-based vehicular diagnostic system proposed in this dissertation.

Part of the results of this study has been published and submitted to different conferences and journals as shown below:

1. Ali Khodadadi, Sang Hoo Woo, Ashish Dalal, Chen-Nee Chuah "Mining Vehicle Failure Consumer Reports for Enhanced Service Efficiency" IEEE Vehicular Technology Conference 2019.
2. Ali Khodadadi, Soroush Ghandiparsi, Chen-Nee Chuah "A Natural Language Processing and Deep Learning based Model for Automated Vehicle Diagnostics using Free-Text Customer Service Reports" Machine Learning with an Application Journal, 2022.

3. Ali Khodadadi, Soroush Ghandiparsi, Chen-Nee Chuah "An Attention-based Markov Chain CNN-LSTM Model for Vehicle Diagnostics" submitted to Machine Learning with Application October 2022.

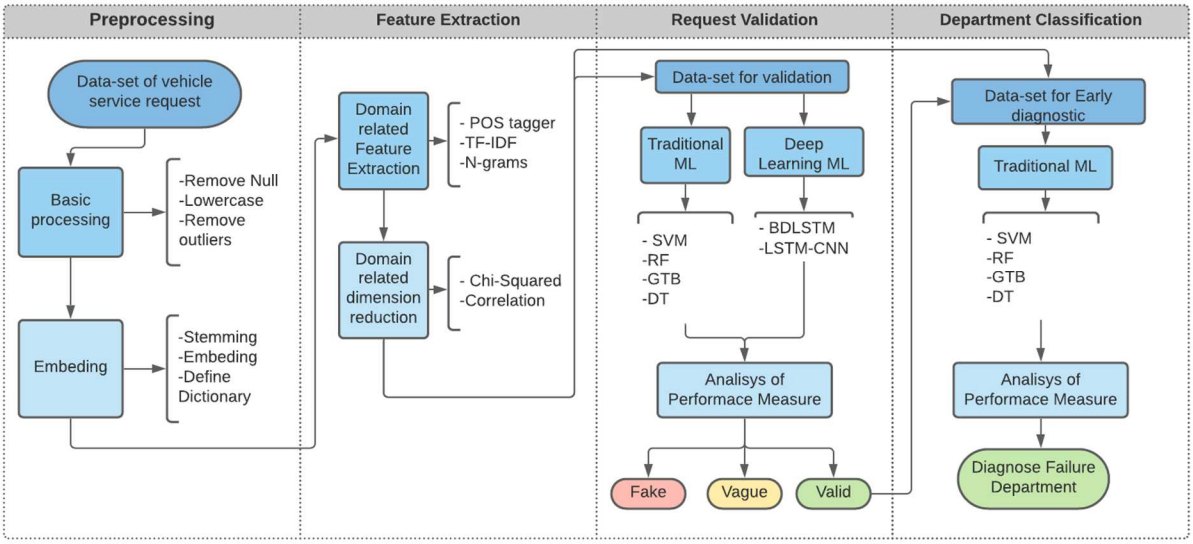

Figure 1.1:Summary of NLP and deeplearning based customer support system

2 Background and Related Work

2.1 Background

Machine learning (ML) and data science techniques have led to remarkable improvements in the automotive industry. By the end of this decade, the automotive sector will upgrade its products with partial or complete autonomous taxi (also known as robotaxi) solutions. Vehicle-to-vehicle and vehicle–to-network communication will be integrated into Intelligent Transportation Systems (ITS) [3]. Therefore, up-to-date customer service is critical for future transportation systems. Early-stage diagnostics and prognostics of failed vehicle components are vital for research and development sections to regulate product improvement strategies. Moreover, efficient diagnostics and prognostics lead to the more successful sale of both products and services.

U.S. businesses suffer billions of dollars in loss from customer dissatisfaction due to the lack of efficient customer service measures. For instance, from Microsoft reports, 22% of clients contacting the customer service section had to request their service requests multiple times [4]. Inefficient customer service wastes customers' time, decreases a company's reputation, and causes higher company costs. Moreover, leaving customers on hold or passing the call to an unrelated service department would lead to lower customer satisfaction and more unresolved issues. Just one bad experience with a company may lead customers to leave the brand.

Vehicle road service plays a crucial role in automotive industry customer support since such failures often occur while the customer is on the road and has limited access to troubleshooting resources. The current vehicle service industry addresses vehicle failures using two main approaches [5]:

- On-Board Diagnostic (OBD) systems using unique toolsets: The operator or technician must communicate via a scanner tool with the Vehicle Controller Module (VCU). The VCU usually provides predesigned fault codes, historical operations statistics, and system and component maintenance dues. During the diagnosis operation, the operator can sometimes record communication signals from the onboard vehicle sensors, actuators, and controllers.
- Placing a service call to customer support: The client places a service request to the customer support department for the product, describes the issue to the best of his/her knowledge, primarily in free text or voice format, and pursues "on-the-road" services support if further assistance is required.

There are known limitations to use the OBD troubleshooting systems. They are typically exclusively designed for each vehicle manufacturer. Thus, they cannot be used across multiple vehicles. The most well-known standards, OBD-II or J1939, are designed to address generic vehicle issues (mainly emission related) and are incapable of comprehensive vehicle diagnostics and prognostics. In addition, due to vehicle hardware computation limitations, diagnostic logic is statics and not designed to be user-friendly. As a result, customers typically prefer to contact the company's service section to resolve the issue. On the contrary, extending the "human-based customer service" is not desirable for auto manufacturers and fleet industries as each case imposes extra expenses, including labor, office, and on-the-road support expenses.

Improving vehicle customer service is crucial for the next generation of intelligent transportation systems. Researchers have recently focused on data-driven fault detection algorithms to enhance diagnostic methods. However, most data-driven algorithms require a large amount of data to train an efficient model. Obtaining sufficient labeled data for such models is a challenging task in the

automotive industry. Therefore, most researchers have relied on short-term data logging or generating simulated and syntactical data [6] to create a practical dataset from customer service reports to feed data-driven diagnostic and prognostic algorithms.

To the best of our knowledge, comprehensive approaches to applying the standard NLP methods in vehicle diagnostics have not been explored in depth. Green et al. [7] introduced a pipeline for driver-vehicle voice interactions by concentrating on vehicle operator voice commands. Likewise, Zheng [8] developed an ML algorithm to enhance GPS routing commands for vehicle drivers. In a similar study, Jalayer et al. [9] classified hydroplaning crashes from free text police accident reports, including numerous domain-related expressions and abbreviations. Therefore, introducing field-related feature extraction and preprocessing for unique NLP vehicle diagnostic applications is essential.

We discuss the related works in three sub-sections in relation to the three main goals of this dissertation: identifying valid service requests, customer request classification, and part failure prediction or prognostics.

2.2 Related Works

2.2.1 Validation

How to efficiently mine a selected text to extract pertinent information has been an important research topic of NLP for decades. The recent years' widespread use of online and publicly available tools has led to the accumulation of large volumes of textual content ready to be analyzed for numerous practical purposes. These tools include news portals, user forums, blogs, publishing platforms, and social media sites like Twitter, Facebook, and Instagram. Some of the leading research on automatic analysis of such content include sentiment analysis (opinion mining), emotion recognition, argument mining (reason identification), veracity, sarcasm/irony detection,

rumor detection, and fake news detection. Automated and high-performance automated methods offer solutions to facilitate essential tasks ranging from trend and market analysis, predictions for elections and referendums to targeted advertising, obtaining user reviews for products, polling, automatic media monitoring, opinion surveys, filtering out unconfirmed content for better user experience, to and online public health surveillance.

Stance detection (also known as stance classification [10], stance identification, stance prediction, and debate stance classification [11]) is a considerably recent subclass of the afore-mentioned research problems. It is typically considered a subsection of sentiment analysis and intends to classify the stance of the text author toward a goal (an entity, opinion, idea, topic, concept, event, or claim) either explicitly mentioned or obscure within the text. Although they evolve around this essential purpose and are semantically close, there are three mainstream definitions regarding the stance detection problem (some in distinct problem settings) as reported in the literature, namely, generic stance detection, rumor stance classification [12], and fake news stance detection. Based on the number of targets and the existence of the stance target in the training and testing datasets of the experimental settings, two other subclasses of the initial generic stance detection problem can be defined: multi-target stance detection and cross-target stance detection. Before presenting these definitions, it will be helpful to define the stance itself from the point of view of linguistics. The core meaning of the stance and further details on a linguistics-based unified stance framework is elaborated in [13].

Saroj et al. [14] used a two-phase LSTM model with an attention model to stance detect Twitter news. This problem is similar to our service validation. Their model obtains the best-case macro F-score of 68.84% and a best-case accuracy of 60.2%, outperforming the existing deep learning-based solutions. Tim [15] used the same dataset but introduced a bidirectional conditional

encoding method to increase Twitter's performance. They have improved the result by changing the LSTM layout.

Deep learning methods have been widely utilized in content extraction in various languages. Alyaba et al. [16] have used it to extract information from the text in the Arabic language and detail domain-specific limitations caused by to use of this method vs. other traditional approaches.

We have posed the “Customer Request Validation” problem as filtering out fake and vague customer service requests, which shares some similarities to fake news stance detection in NLP literature. The body is a classification problem for input in the form of the Service Call Log. This type of the body's stance towards the headline's claim is sought in the form of a category. There have been many approaches to identifying fake news or stance analysis [17]. Our goal is to employ similar techniques to facilitate the task of identifying vague or non-relevant service requests which impose financial costs on companies.

2.2.2 Classification

Unstructured text is a challenge in almost any area that regularly uses text databases or communication. Research institutions, universities, businesses, government funding agencies, and technology-intensive companies are areas that create unstructured test content on day-to-day operations [18]. Eighty percent of entity data (place, person, or thing) are available only in an unstructured format. These texts are in the form of news, reports, email, and views. NLP is the yet hidden relationships between entities in a dataset to derive meaningful patterns which reflect the knowledge contained in the dataset. This knowledge is utilized in decision-making [19]. Clustering, classification, and categorization are NLP's fundamental techniques. It is the process of assigning, for example, a document to a particular class label (in our application, different

troubleshooting departments). Thus, text classification is a mandatory part of knowledge discovery [20]. This section briefly describes various text classification techniques related to our research.

Classification algorithms are an essential part of text mining techniques [21]. Similar to text validation, classification techniques could be divided into statistical and ML approaches. The algorithms are broadly divided into supervised, unsupervised, and semi-supervised categories according to the learning criteria followed. Among the supervised classification algorithms, there are two categories, parametric and non-parametric, based on the supremacy of parameters in the data. Logistic regression and Naïve Bayes are the parametric classification algorithms most widely used [22]. Support Vector Machine (SVM), Decision Tree, Rule Induction, KNN, and Neural Networks are their non-parametric counterparts. Fuzzy c-means, k-means clustering, and Hierarchical clustering are unsupervised learning approaches, and co-training, self-training, transductive SVM, and graph-based methods form the constituents of semi-supervised learning methods.

There have been relatively few studies on classifications of customer reports in automotive applications. However, numerous research studies exist on the classification of text in the context of medical records or social media.

Medical applications benefited from this technique in a variety of sectors, such as medical imaging reports [23], [24] or cancer reports [25], [26]. News deep learning classifications also are detailed in [27].

Botsis et al. [28] have developed methods to extract features for medical applications; they developed a pipeline in the US Vaccine Adverse Event Reporting System (VAERS) that collects spontaneous reports of adverse events following vaccination. Medical officers review the reports

and often apply standardized case definitions, such as those developed by the Brighton Collaboration.

Bac [29] used Deep Neural Network (DNN) and LSTM in parallel and exported the result to NN for stance analysis for the Vietnamese language. We have performed our unique architecture due to the same reason. We will detail them more in **Error! Reference source not found.**.

2.2.3 Prognostics

The term "prognostic" means foretelling a future outcome or event in dictionaries. In prognostics and health management (PHM), prognostics means a process to predict future degradation and the system's Remaining Useful Life (RUL) based on the available degradation data. There are several definitions of interest for prognostics.

1) Prognostics is the process of estimating the remaining life of the component [30].

2) In International Organization for Standardization document (ISO 13381-1, Section 5.1), prognostics aim to provide the user with the capability to predict RUL with a satisfactory confidence level [31].

3) Prognostics is the process of predicting the future reliability of a component or a system by assessing the extent of derivation or degradation of the product from its expected normal operating conditions. It predicts the future state of health based on current and historical health conditions [32].

4) Prognostics predicts future damage/degradation and the RUL of in-service systems based on the measurement damage data [33].

5) Prognostics addresses the use of automated methods to detect, diagnose, and analyze the degradation of physical system performance, calculating the remaining life in an acceptable operating state before failure or unacceptable performance degradation occurs [34].

A degradation process involves many uncertainties, which can all cause the inaccurate prediction of RUL. From the standpoint of statistical uncertainty, the degradation process varies across different applications manufactured under the same process and condition. In a specific operation environment, units would have different degradation trajectories. For example, batteries tested under the same profile and environment would form different capacity degradation processes. In the meantime, uncertainties from the operating and environmental conditions, such as future loads and circumstance changes, exist in the degradation modeling. Modern prognostic approaches aim to compute these uncertainties to obtain an accurate RUL prediction effectively. Advanced techniques, such as machine learning (ML) algorithms, are used to model the degradation process. However, model errors still exist, which might be caused by missing failure modes, misspecified models, and unmodeled phenomena. It can be considered a biased understanding of the degradation process of interest. Comprehensive prognostics models intend to incorporate physical understandings of the degradation process to reduce model errors.

For example, the incorporation of electric vehicles in the degradation modeling of State of Health (SOH) of electrical battery pack models has been investigated. However, physical understandings of the degradation mechanism are usually limited and incomplete for complex systems. Therefore, uncertainties from model errors can rarely be eliminated, even with improved methods and investigations. Recently, uncertainties caused by measurement devices have attracted attention since the measurement data are taken as the input of the prognostics algorithm. Biased

measurement/performance data might lead to total invalid prognostics. These uncertainties can be classified into four categories [35].

1) Input uncertainties are inherent in any process, such as geometric characteristics, initial state estimation, geometric characteristics, material property, and manufacturing variability. These uncertainties are tied to the degradation process and cannot be eradicated. The design of multiple experiment runs, such as repeated measurement design, is the most commonly used method to characterize these uncertainties.

2) Model uncertainties are related to errors, such as unexplained features, misspecified methods, and unmodeled phenomena. Advanced methods, including data science and testing techniques, have been developed to reduce these uncertainties. With the development of advanced sensing techniques, large volumes of measurement data are available. The "big data" challenge might cause more model uncertainties.

3) Operational uncertainties are related to within operation and involve environmental conditions. These uncertainties are similar to model uncertainties, which stem from the limited understanding of the degradation process. A deeper investigation can reduce these uncertainties, especially under various operating and environmental conditions.

4) Measurement of uncertainties is related to uncertainties in data observation methods, such as sensor noise and filter error. Improved sensing techniques and advanced data collection and processing methods can reduce these uncertainties. Adequate prognostic methods are expected to handle all of these uncertainties. Specifically, prognostic models should be capable of presenting variety and veracity in data and a comprehensive physical basis so that model uncertainties can be reduced to improve the robustness of data-driven computational models.

5 Many types of research have been done to review prognostic approaches and their applications, such as [36]. Based on such review works, this article summarizes the most recent prognostic works considering data availability and the physical mechanisms of the applications.

Condition-Based Maintenance (CBM) for widespread application was identified during a 2002 workshop organized by the National Institute of Standards and Technology (USA) [37].

Prognostics and Health Management (PHM) is an emerging engineering discipline that links studies of failure mechanisms (corrosion, fatigue, overload) and life cycle management [38]. It aims to extend an engineering asset's service cycle while reducing exploitation and maintenance costs. Mainly, the acronym PHM consists of two elements [39].

1. Prognostics is a prediction/forecasting/extrapolation process by modeling fault progression based on current state assessment and future operating conditions.

2. Health management refers to a decision-making capability to intelligently perform maintenance and logistics activities based on diagnostics/prognostics information.

Prognostic approaches:

The core process of prognostics is to estimate the RUL of a component or system by predicting the future evolution at an early stage of degradation. An accurate RUL estimation enables running the equipment safely as long as it is healthy, which provides additional time to opportunely plan and prepare maintenance interventions for the most convenient and inexpensive times [40]. Due to the significance of such aspects, study on PHM has overgrown in recent years, where different prognostic approaches are being developed. Several review papers on the classification of prognostic approaches have been published [41], [42], [43].

Most of the prognostic approaches are divided into three categories: data-driven, physics-based, and hybrid, based on the type of physical knowledge used to model the degradation process(referring to Figure 2.1).

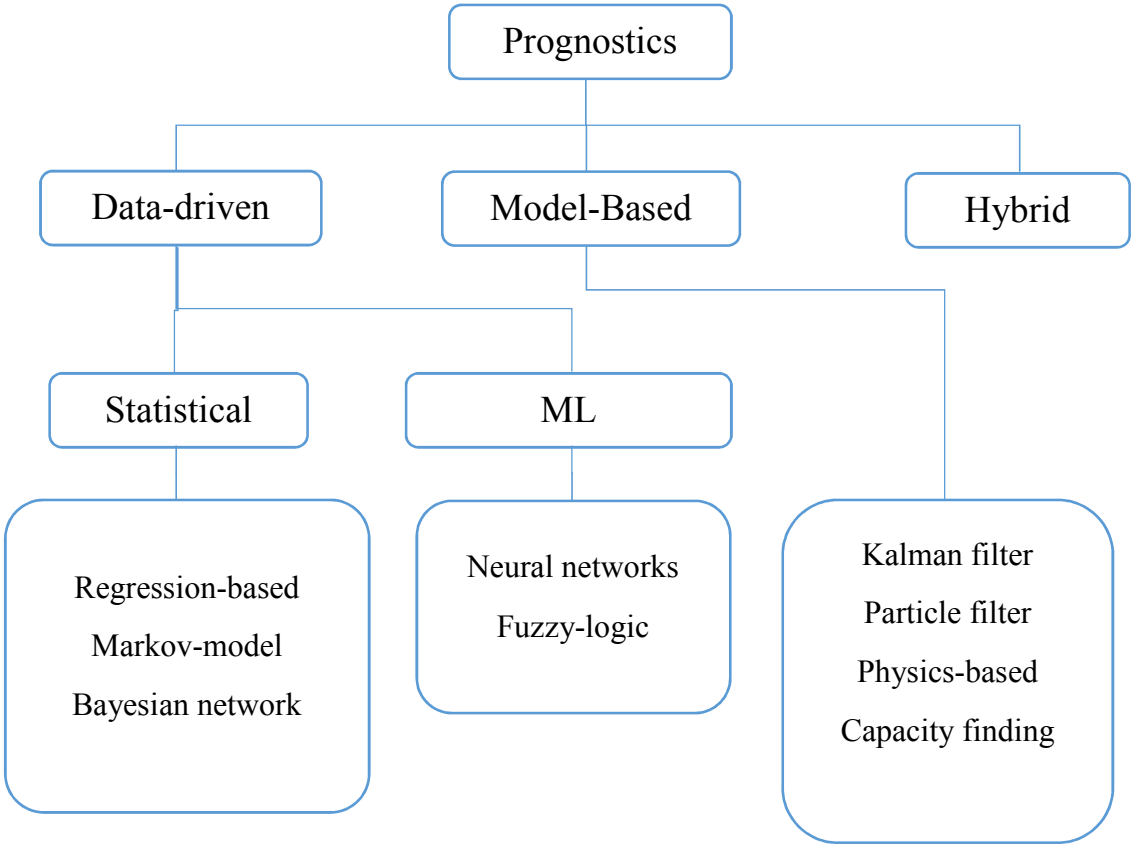

Figure 2.1:Summary of prognostic models

Model-based approach:

Numerous model-based approaches have been developed depending on the model's complexity and capability to model the system.

The physics-based approaches use explicit mathematical representation (or white-box model) to formalize the physical understanding of a critical system [44]. RUL estimates with such approaches are based on gathered knowledge of the process that affects regular machine operation and can cause failure. They are based on the principle that failure occurs from fundamental

processes: electrical, thermal, chemical, mechanical, and radiation [45]. Standard physics-based modeling approaches include material-level models like spall progression, crack-growth, and gas path [46]. These models utilize knowledge such as a system's loading conditions, geometry, and material properties to identify potential failure mechanisms [47].

Moreover, such methods require detailed knowledge and understanding of the process and mechanisms that cause failure to predict the system's behavior. In other words, failure criteria are created using physics of failure (POF) analysis and historical data information about failed equipment. Implementation of a physics-based approach has to go through several steps, including Design Failure Modes and Effects Analysis (DFMEA), feature extraction, and RUL estimation.

Zhang et al. [48] modeled a permanent magnet electric motor using structural analysis to define a specific diagnostic system in electric vehicles. Meinguet et al. [49] feed similar modeling methods with obtained data from an electrical traction Hardware in the Loop (HIL) system to introduce a DC-DC inverter diagnostic system. However, their research addresses issues at a component level.

Bayesian-based models are the primary approach for systems that involves some uncertainty modeling and estimation of system status based on observations. Different versions of filters, such as Kalman filter (KF), Extended Kalman Filter (EKF), and Particle Filter (PF), are employed depending on the model's situation and complexity [50]. These models can be used to model multivariate, dynamic processes. Basic KF is computationally efficient, particularly for systems with many states. It can accommodate incomplete and noisy measurements. EKF variants are available for non-linear processes.

On the other hand, the process and measurement noise must be Gaussian. Also, some variants diverge quickly, and variants for non-linear systems are more computationally intensive than basic Kalman filters. Measurement data is essential for these models.

Data-driven prognostics

Data-driven (DD) prognostic approaches can be seen as black box models that learn systems behavior directly from collected condition monitoring (CM) data (e.g., force, pressure, temperature, current, vibration, acoustic signal, voltage). They rely on the assumption that the statistical characteristics of system data are relatively unchanged unless a malfunction occurs. Such methods transform raw monitoring data into relevant information and behavioral models (including the degradation) of the system.

Data-driven methods can be affordable models with better applicability, as they only require data instead of prior knowledge or human experts [51]. According to the literature, several studies have been performed to categorize data-driven prognostic approaches. [52] and [53] classified data-driven methods into machine learning ML and statistical approaches. A survey on ML approaches for prognostics was presented by [54], where data-driven approaches were categorized as conventional numerical and machine learning methods. Reference [55] classified data-driven prognostic methods as evolutionary, machine learning, and state estimation techniques. We classify data-driven approaches for prognostics into two categories: ML and statistical approaches.

Machine learning approaches

Machine learning approaches attempt to learn by examples and are capable of capturing complex relationships among collected data that are hard to describe. Such methods are suitable for situations where the model-based approaches are not favorable for replicating behavior models

[56]. The learning process can be performed in various methods depending on the available data type. Supervised learning can be applied to labeled data, i.e., data are composed of input, and the desired output is known. Unsupervised learning is applied to unlabeled data, i.e., learning data are only composed of input, and the desired output is unknown. A semi-supervised learning model involves both labeled (few data points) and unlabeled data. Partially supervised learning is performed when data have imprecise or uncertain soft labels (i.e., learning data are composed of input, and desired outputs are known with soft labels or belief mass [57]). ML is a rapidly growing field in the PHM domain, and many algorithms are being developed.

Cheng et al. [58] proposed an ML fault detection method, named the "Deep Slow Feature Analysis" method (DSFA), for the running gears of high-speed locomotives. The designed system used statistical algorithms to develop fault detection on the multi-dimensional running gear data in the test bench. The train was equipped with six extra sensors to collect the data via timely and expensive test procedures. Prytz et al. [59] applied supervised ML techniques to detect the failure of commercial trucks' air compressors. Logged onboard data collected over three years from a large number of trucks. Wolf et al. [60] proposed an unsupervised learning data-driven diagnostics approach to detect faults by transferring the concept of deeply embedded clustering for static data to multivariate in-vehicle time series. However, the collected data in mentioned studies could not reach a sufficient resolution to apply to accurate data mining algorithms. They all have generated necessary data either from testbench or used simulation data. In general, ML models help incorporate with physics of failure models. The confidence limits are available from the underlying model (for which parameters are estimated). In contrast, fewer data are required for estimating parameters as models tend to be failure specific, and determining the most appropriate model is essentially trial and error and can be time-consuming

Statistical approach

Statistical-based models estimate the RUL by fitting the probabilistic model to the collected data and extrapolating the fitted curve to failure criteria. They are simple to conduct. Like ML approaches, statistical methods also require sufficient condition monitoring (CM) data to learn the behavior of degrading machinery. However, they can have significant errors if data is not incomplete. Therefore, the nature of data is essential in this category. The research team in [61] presented a state-of-the-art review of statistical approaches, where the taxonomy was mainly based on the nature of CM data.

From the systematic review, some commonly known prognostic approaches can be regression-based methods, stochastic filterings, or state estimation methods like Kalman Filters, Particle Filters and variants, Hidden Markov models, and variants. Further details about this taxonomy are described in [40]. It should be noted that the Bayesian techniques cited above can also be addressed as machine learning approaches. Other methods in this category can be classical time series prediction methods like Auto-regressive moving averages and variants or proportional hazards models.

Lam et al. have used the Markov chain to evaluate customer agent performance [1]. Tsui et al. [62] developed a PHM framework that offers comprehensive yet individualized solutions for managing system health. It is a well-established approach and can model numerous system designs and failure scenarios. It also can readily manage incomplete datasets. Conversely, a reasonably large volume of data is required for training. It assumes a single monotonic, nontemporal failure degradation pattern (i.e., different stages of failure cannot be accounted for). Moreover, it cannot model previously unanticipated faults and root causes. As a result, more

complex semi-Markov models are required if failures or failure times are not exponentially distributed.

3 Dataset

3.1 Description

We used a dataset of twelve years of customer service calls to a utility fleet truck company in this research. The dataset was obtained from the private sector; additional details are omitted due to its proprietary nature. In addition, some of the names and specific labels are modified for privacy reasons. The sample data consists of 120,000 recorded service records. Each service record starts with a service call. Then Customer shares the finding of the issue with the service representative. The service representative summarizes the finding in the "Service Call Log." The request is transferred to the proper service department to address the issue. Once troubleshooting is finished, the mobile or shop service technician summarizes the detailed task of all service procedures in "Service Detail." They also log the replaced parts into the system database. Finally, the quality department evaluates service request through the Service Call Logs and Service Details reports and create the "Service Department Relation." Figure 3.1 demonstrates the service request and how each data section is created on each call.

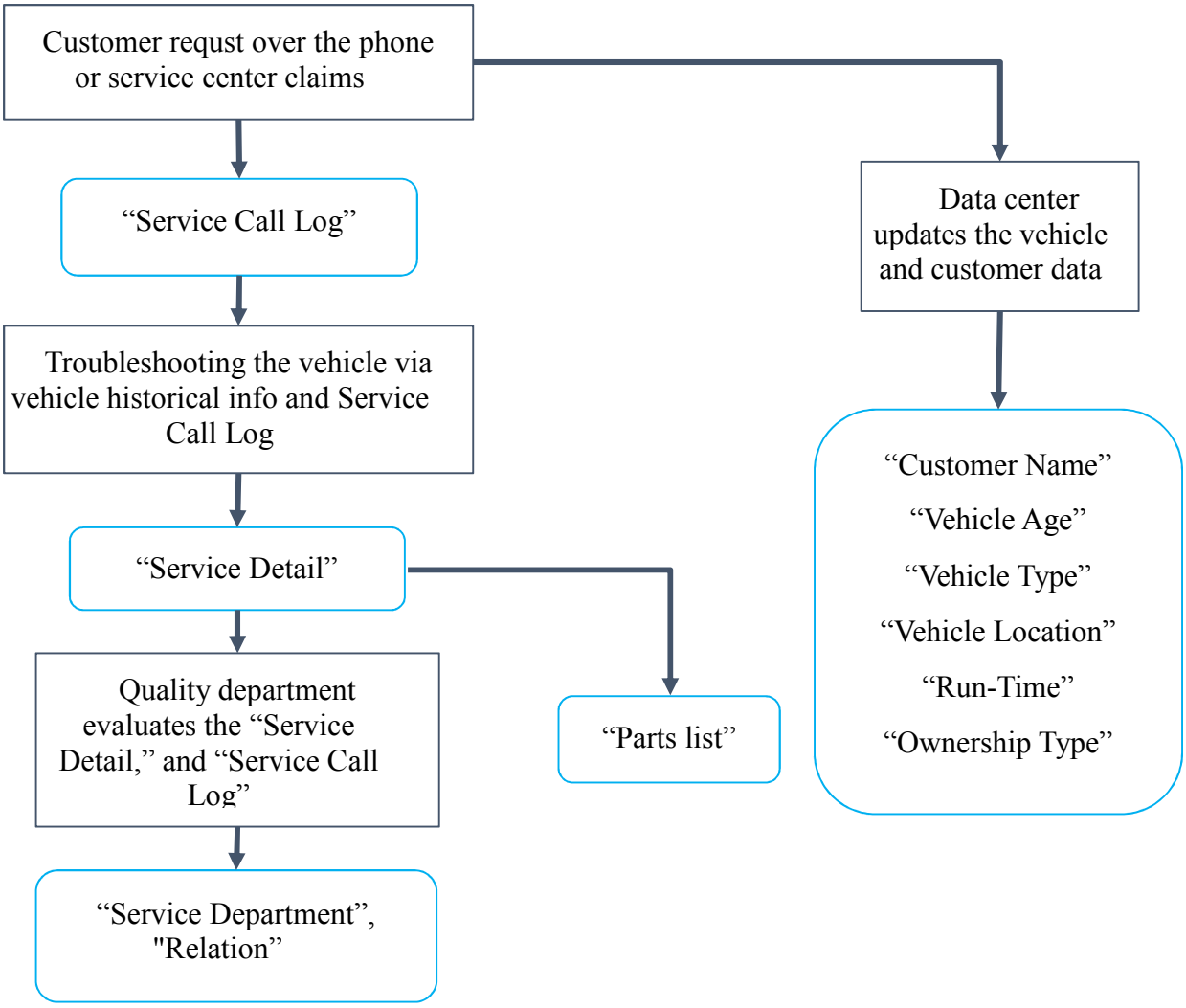

Figure 3.1: Structure of vehicle service request

3.2 Service Call

Each service call includes 12 main sections, as follows:

- " **Service Department**" specifies one of the 16 different departments. Each focuses on troubleshooting a specific vehicular subsystem. The departments are shown in Table 3.1. One department is titled" Vague" to represent rare or unique service requests.

Table 3.1: Service Departments

1	Controls	9	Vague
2	Harness	10	Hydraulics
3	PTO	11	Boom
4	Maintenance	12	Test
5	Rotation	13	Auger
6	Outrigger	14	Digger
7	Body	15	Chassis
8	Electronics	16	Resale

- " **Service Call Log**" contains a few words summary entered by the service representative. The information concludes the customer's complaints. That can be a combination of words, information about the vehicle, date and time of the service. It can also be without any comment if the representative cannot summarize the service call into a sentence. Sometimes, it can be general comments such as" service needed."

Table 3.2 Shows several samples of Service Call Logs.

SPACE007-Repairs from inspection **24506-replace winch rope unit is down Inoperable** **3336-stuck in the air** **Unit is down** **---**

Table 3.2: Samples of Service Call Logs

- " **Service Detail**" Typically consists of a paragraph to the one-page description of the service performed to date on the failed vehicle. Table 3.3 shows examples of Service Details. Usually, a dash or a new line separates different service tasks. Connection words, dictation mistakes, and numbers are included in the report.

Sentences in this section have better formatting than Service Call Log section. This section does not have any structure. It includes the service date, vehicle model or identification number, replaced parts, and a list of actions performed to fix the issue. It is, by nature, full of technical abbreviations and expressions.

cut off and replaced damaged area or repair
installed new hinges and pained area
replaced and adjusted transfer pin-- perform test
replaced gasket PN9700007824--had to cut off
inpected and cleaned area--cleaned up shaft
970000271 replaced, adjusted system pressure

Table 3.3: Samples of Service Details

- " **Relation**" summarizes whether the service Call Log and the Service Detail have the same claim. This section is manually annotated after service is done with one of the following classes:

    1. False claim: The Service Detail information differs from the Service Call Log. It could result from inaccurate technician diagnostic, lack of meaningful information, or wrong direction in providing information from the customer.
    2. Valid claim: Service Call Log reflects the detailed information properly. The report includes valuable information and can get addressed to the proper service department
    3. Vague claim: The service Call Log does not provide any information to route to the proper department." blank" Service Call logs or general terms like" unit failed" are included in this section.

- " **Age of Unit**" Reflects the vehicle age at the time of the incident. The age is used for the helpful time expectancy of parts and components, which would be beneficial information for system prognosis.

- " **Vehicle Location**" is the postal code of the vehicle service shop. This information reflects which geographical location was vehicle typically used during its lifetime.
- " **Customer Company**" The vehicles in this study are mainly used for commercial applications, and this section contains the company name associated with the vehicle.
- " **Date of Failure**" include the date the vehicle failure was reported to the service center. It defines which day of the week and year it failed.
- " **Run-time**" reflects how long the vehicle has been in service since these service vehicles are also utilized stationary and idle.
- " **Ownership Type**" defines whether the vehicle was rented, leased, or purchased.
- " **Time Last Service**" Shows the duration the vehicle was used after its last service to determine if it was due for service at the time of failure.
- " **List of Parts** " summarizes parts that have been replaced during the service. This list is created after the service is finished.

4 Natural Language Processing Taxonomy for Vehicle Industries

Natural Language Processing (NLP) explores how math, algorithms, and linguistics can be used to understand and manipulate natural language text or speech for different applications. NLP researchers investigate and gather knowledge to formulate how human beings understand and use language to develop the appropriate tools and techniques. This knowledge enables computer systems to understand and manipulate natural languages to perform desired tasks. NLP applications include many fields of study, such as machine language translation, natural language text interpretation and summarization, user interfaces, multilingual and cross-language information retrieval, speech recognition, artificial intelligence, and expert systems. Following is a description of some of the mentioned applications:

- **Machine translation**: The science of converting source text to the target language is known as machine translation.
- **Automatic summarization:** Information overload can be challenging when extracting a specific and significant detail from an extensive knowledge base is required. An example of this application is gathering information from social media or summarizing news reports.
- **Sentiment analysis** usually refers to extracting reactions to the text related to specific topics. Many companies use this method to classify users' feedback on their products or services automatically.
- **Text classification:** routing text to different buckets through specific filters is known as classification. Spam filtering in emails is a typical example of this.

- **Question answering:** Any pipeline to identify the human request, intention, and response properly fit this category. It is a challenging task, and there has been numerous research and development in this section.

4.1 Domain-related Preprocessing

Text preprocessing is a combination of various methods to clean the text data from unnecessary part of the text and prepare it to feed data to the model. Text data contains noise in various forms like emotions, punctuation, and grammatical and dication mistakes. Humans can express their intention using different word combinations, so depending on the application and writer's intention, other words might interpret the same meaning. Computing machines and algorithms work with numbers, not words, so it is necessary to convert text to numbers efficiently. This process is called embedding and is the first step in NLP models. There are standard NLP steps to preprocess the text, and standard toolsets are created to perform these tasks. We initially investigated such toolsets. However, the initial results were not satisfactory.

Technical abbreviations and expressions are standard in the Service Call Log and Service Detail sections. It is also evident that grammatical variances are present. Consequently, data preprocessing includes domain-related and intuitive approaches detailed as follows.

4.1.1 Stop Word

The words filtered out before processing a natural language are called "stop words." Articles, prepositions, pronouns, and conjunctions are the most common stop words in most texts. They do not add much information. Examples of a few stop words in English are "the," "a," "an," "so," and "what." Despite most applications, emotions must be removed from technical reports' content. As a result following action has been added to the stop word step of the NLP model.

1. Removed punctuations interpreted feelings such as "!" and"?".

2. Removed external links

3. Since connection words such as "a" or "an" could represent domain-related expressions and can represent failure correlation, we need to keep them in this application. For example, "*an 57 is failed*" in the studied database represents a vehicle model. In this case, "an" has meaning if the following word is a number. However, if "an" followed by a word does not have any value and needs to get removed for such, we have added "an" to the "stop word" list, only it will not follow by any number. In another example, "*ring and gasket replaced*" represents the failure correlation between these two parts and the need to keep it within the model.

4. Remove none relevant words: We have introduced an extra step to remove pertinent none words from the text. For example, the NLP algorithm does not need to search for combustion engine-related parts like "*Fuel tank*" within electric car service reports. This extra step would improve the calculation speed and performance of the model since there would reduce the dimension of inputs to the model.

5. Remove rare words: unlike most applications, we must keep all rare words within the text. Therefore, in most texts, the existence of rare would not imply any meaningful information. However, in this application, the presence of a world in the text is usually necessary. As an example, it may indicate the failure of specific parts.

6. Furthermore, some combinations of words, such as "needs service," are considered vague or irrelevant. They are removed from the dataset.

4.1.2 Lower Casing

The lower casing is another standard step in NLP preprocessing to ensure the model works the same if letters are upper or lower in the text. However, like stop word, this would cause an issue in technical text processing in-vehicle applications since having an upper case letter sometimes has a different meaning than a lower case. For instance, the term "AM" in this application expresses a specific vehicle model and needs to get identified differently than the verb "am." As a result, the introduced algorithm does not allow lowercase if a word in the text with all uppercase letters.

4.1.3 NER

Named Entity Recognition (NER) are usually elements in the sentence into categories such as person or location. Identifying NER in a text is a challenging task that traditionally requires large amounts of knowledge in the form of feature engineering to achieve the desired performance. Nicholas et al. [63] have used deep learning LSTM to identify names approach to identify NER in a text.

Since NLP in the automotive field has been less explored than in other areas, we had to create a NER dictionary for this application. So we have developed a comprehensive taxonomy in the automotive field to identify automotive-related expressions such as parts names, specific failures, and troubleshooting methods. Unfortunately, most available dictionaries are not an appropriate fit for this application. For example, "*Timing Belt*" in automotive is related to a specific part.

In another step, we performed numerous spell corrections. I.e., "*HYD*," "*Hydraulic*," "*hydraulic*" all changed to "*hydraulic.*"

4.1.4 Tokenizer

A tokenizer is a tool that categorizes types of words in the text from a pool of words. Tokenizers are not suited for identifying the type of service call. An example includes the keyword" Hyd," which represents a Hydraulic system. A normal tokenizer separates this example as an unknown word when it is a noun in this application. We modified the tokenizer and picked acceptable expressions to increase the precision to the point of utilization.

4.1.5 Stemming and Lemmatization

Tests and reports use different forms of a word for various grammatical reasons. As an example, multiple forms of expression, such as "*run*," "*running*," and "*ran,*" have the same meaning at different times. Additionally, there is various type of a word with similar purposes, such as "*democracy*," "*democratic*," and "*democratization*."

Both stemming and lemmatization aim to reduce inflectional forms of related word forms to a common base form. For instance, "am," "are," and "is" change to the root word of the "be" sign. They reflect the same meaning.

The result of this mapping of text is shown in Figure 4.1.

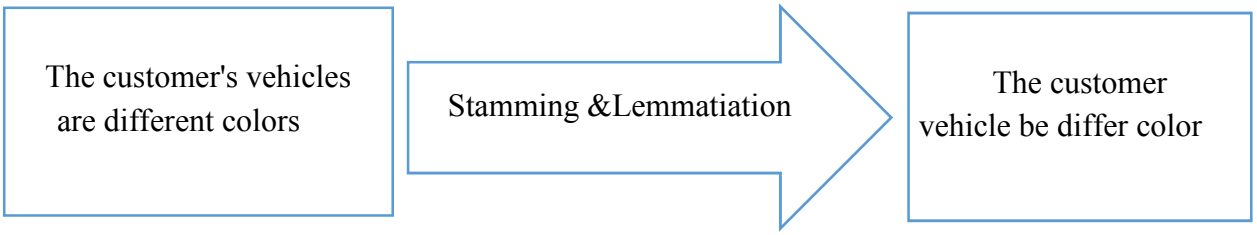

Figure 4.1: Stemming example

The stemming goal refers to a process that chops off the ends of words hoping to achieve the root of the word.

Lemmatization usually refers to using a vocabulary and morphological analysis of words, typically aiming to remove inflectional endings only and to return the base or dictionary form of a word, known as the lemma.

The difference between Stemming and Lemmatization can be understood with the example provided in Table 4.1.

Table 4.1: Stemming Vs. Lemmatization

Original Word	After Stemming	After Lemmatization
goose	goos	goose
geese	gees	goose

4.1.6 Padding

Padding is an operation that adds value to the edges of a vector or matrix to increase its size to reach a specific desired size. This process is done here because the sentences in a text do not have the same length or number of words. The value "0" is padded to the end of the sentences until they all have the same size. The padding value "0" does not stand for any word or symbol, and the networks learn that it does not contain any information.

Table 4.2 summarizes an example of the developed preprocessing approach in this project vs. other known solutions. As is shown in the table, the outcome of processing models is not similar.

Table 4.2: Comparing the output of introduced processing vs. typical processing models

Original text	AM 55 unit was down. Oilring washer in HHD line is replaced
Typical preprocessing model	55 unit be down. na washer na line be replace

Developed preprocessing model	AM 55 unit be down PN 24024562 in PN 24201903 be replace

4.2 Feature Extraction

The feature extraction procedure extracts a predefined set of features from the text. Since algorithms cannot directly work on the raw text, set vectors of weighted features are given to predictive models. The process of converting a raw set of selected words into a matrix (or vector) is called "word embedding." Various feature extraction has been approached in consumer reviews in different industries. Mars et al. [65] presented a new method to extract product features' customer opinions from social networks using text analysis techniques. This method identifies customers' opinions regarding product features.

Similar to preprocessing, we took some additional steps. The unique feature extractions will be described in this chapter. For instance, some Service Call Logs lacked any proper technical values. Instead of containing imprecise information, the operator summarises issues such as "blank" or "machine does not work." These ambiguous data points could result from either a lack of knowledge on the part of the customer service representative or a lack of precise information provided by the customer. The presence of keywords such as "inspection," " PM," and "Annual checkup" would help with a more straightforward classification. Conversely, some stop words such as "service needed" or" failed unit" were considered irrelevant and excluded from the data. The following sections of this chapter summarise the feature extraction technique used in this research.

4.2.1 POS Tagger

A Part-of-Speech (PoS) tagger is a feature extraction tool that classifies words into several grammatical categories, such as nouns, verbs, adjectives, prepositions, pronouns, and adverbs. [64]. Similar to other NLP preprocessing and feature extraction techniques, authors have noticed that typical POS taggers may misclassify a word, leading to the unintentional removal of essential terms.

We modified the POS tagger to increase the precision to the point of utilization. Further issues arise to utilize a tokenizer to parse the data.

These issues can be challenging when a combination of words represents one expression in the system. For example, "left valve" refers to a specific component in the vehicle. A standard tokenizer, however, would split this phrase into two different words. This action would lead to "upper" being classified as an adjective and "valve" being classified as a noun, resulting in the poor performance of the models.

As another example, "vehicle down" in the report means the total failure of the vehicle. A standard tokenizer would split this phrase into two words leading to "vehicle" being classified as a noun and "down" as an adjective. However, in this application, this phrase represents the total failure of the vehicle. This challenge illustrates the need to manually modify the tokenizer by incorporating domain-specific terminologies [64].

Initially, the most common nouns, adverbs, adjectives, and bigrams were utilized to extract features from the report. N-grams are the sequence of n-words in a sentence, considering it as a single unit. Combinations of two words represent lots of components and services. As a result, bigram counts constitute an essential feature of this approach. For example, "IS inspection"

denotes a separate service department, and "Rotation Oilring" represents a unique component in the vehicle.

The Service Call Log is typically a sentence long. We removed verbs and adverbs from this section. For example, " Electrical controller does not work" would fit into the electrical category. The word" electrical" would be a vital feature of the sentence, while the verbs, adverbs, and adjectives are not as essential and are therefore removed. One of the challenges observed is that the customer reports may contain technical abbreviations that are uncommon or unknown within traditional NLP tools.

Figure 4.2 represents how a customized POS tagger performs in this research.

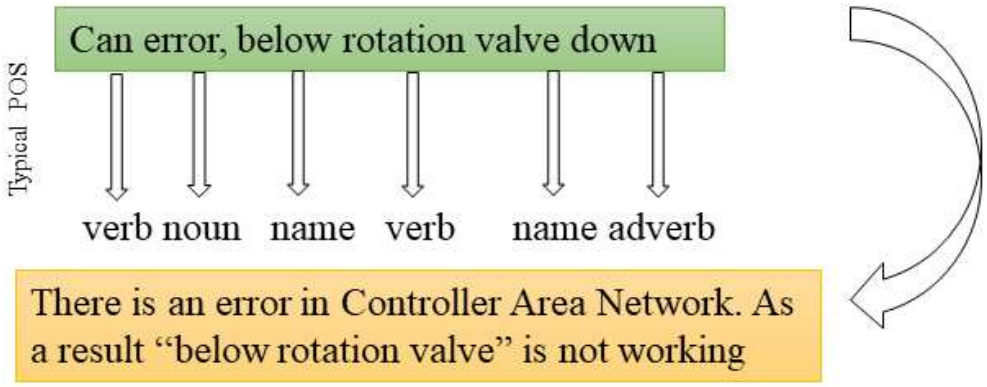

Figure 4.2: POS tagger example: typical vs. introduced

4.2.2 Bag-Of-Word

Bag of Words is a simple NLP tool that counts how many times a word appears in a document. Bag-of-Words is commonly used in clustering, classification, and topic modeling by weighing particular words and relevant terminologies. Below is a flow of Bag-of-Words transformation.

Figure 4.3 represents a symbolic view of the bag of words works.

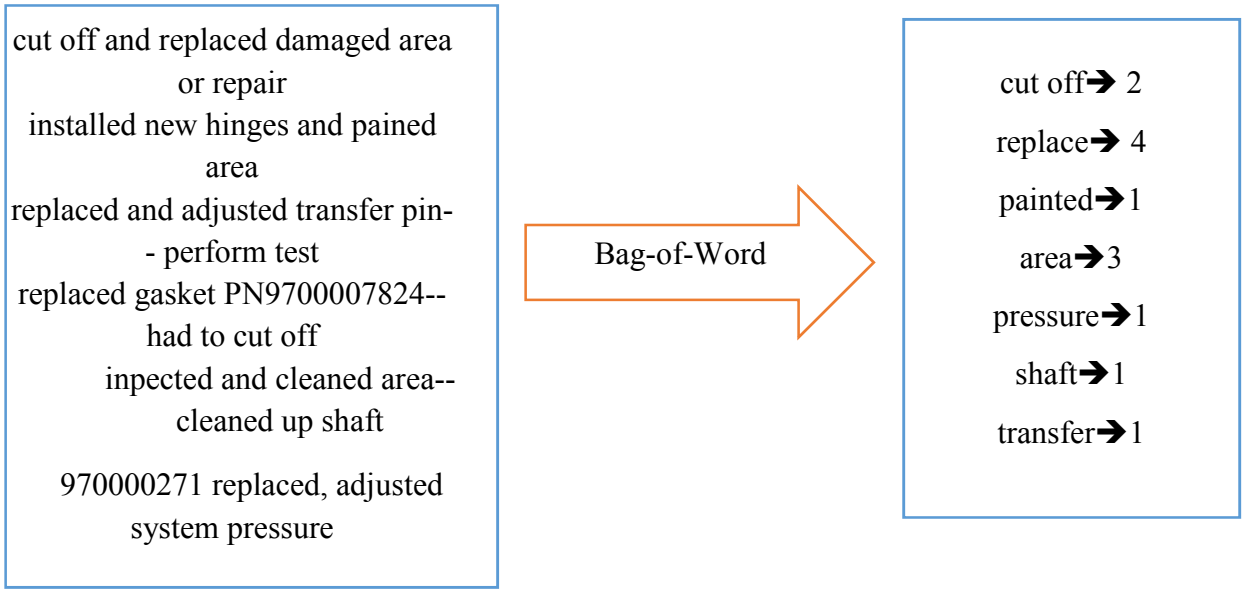

Figure 4.3:Bag-of-Word Representation

4.2.3 TF-IDF

TF-IDF is the abbreviation for Term Frequency-Inverse Document Frequency. The main goal of this tool is to scale how important a word is to a document in a collection or corpus.

The TF-IDF value increases proportionally when a specific word appears in the document. Also, it is offset by the number of documents in the corpus that contain the word, which helps to adjust that some words appear more frequently in general. $tf_{i,j}$is the number of occurrences of "i"in "j." df_i The number of documents containing "j" and "N" is the total number of records.

$$TFIDF\ W_{i,j} = tf_{i,j} * log\frac{N}{df_i} \qquad (4.1)$$

We have not observed relative strength in the frequency of the word used in a specific report compared to the entire dataset. As a result, we did not pursue this approach in our study. Customer

reports can contain technical abbreviations that are uncommon or unknown in traditional NLP tools.

4.2.4 Embedding

Embedding or Word2Vec takes a large corpus of text as input. It produces a vector space, typically of several hundred dimensions, with each unique word being assigned a corresponding vector in the space.

Word embedding is one of the known representations of document vocabulary. It can capture a word's context in a document, semantic and syntactic similarity, and relation with other terms. It was developed by Tomas Mikolov [65]. It is an essential step in feature extraction in this research.

Consider these similar sentences: "Wiring is in bad condition" and "wiring is in an unstable condition." They have similar meanings. So it is essential to create a comprehensive vocabulary of V = {Wiring, is, unstable, bad, condition}.

Generating a one-hot encoded vector for each of these words in V. Length of our one-hot encoded vector would equal the size of V (=5). There would be a vector of zeros except for the element at the index representing the corresponding word in the vocabulary. That particular element would be one. The encoding would formulate as follows:

$$wiring = \begin{bmatrix}1\\0\\0\\0\\0\end{bmatrix}, is = \begin{bmatrix}0\\1\\0\\0\\0\end{bmatrix}, bad = \begin{bmatrix}0\\0\\1\\0\\0\end{bmatrix}, unstable = \begin{bmatrix}0\\0\\0\\1\\0\end{bmatrix}, condition = \begin{bmatrix}0\\0\\0\\0\\1\end{bmatrix},$$

Suppose we utilize this encoding approach where each word occupies one of the dimensions. The model recognizes "bad" and "unstable" as different from "condition" and "wiring," which is not valid. The objective of this research was to have words with similar contexts occupy close spatial

positions. The cosine of the angle between such vectors should be close to one, i.e., an angle close to zero.

Word2Vec is a tool to construct such an embedding. Mainly uses two methods Skip Gram and Common Bag Of Words (CBOW)

- **CBOW Model**: This method takes each word's context as the input and tries to predict the corresponding phrase. In the mentioned example, we used the word "bad" CBOW tries to predict a target word "condition." More specifically, we use the one-hot encoding of the input word and measure the output error compared to the one-hot encoding of the target word (condition).
- **Skip-Gram model:** is based on CBOW. However, the multiple-context model just got flipped.

Both methods have their pros and cons. According to Mikolov [65], Skip Gram works well with a small size of data and is found to include rare words well. However, CBOW is faster and has better representations for more frequent terms. So we have used CBOW in our model.

4.2.5 Dimension Reduction

In the last feature extraction step, we investigated different dimension reduction techniques for this application.

As it is self-explanatory from its name, dimension reduction is focused on reducing the dimension of data. As a result, the output data representation retains meaningful properties of the original data. In a nutshell, this technique would create the following benefit to the model:

1. Improves visualization and exploration of the dataset.
2. Reduce memory usage of the dataset.

3. Simplifies the model algorithm.
4. Decreases overfitting ratio.
5. Improving the performance of the model by choosing the right features

Several dimension reduction methods can be used with different data types. Figure 4.4 represents the main dimension reduction methods.

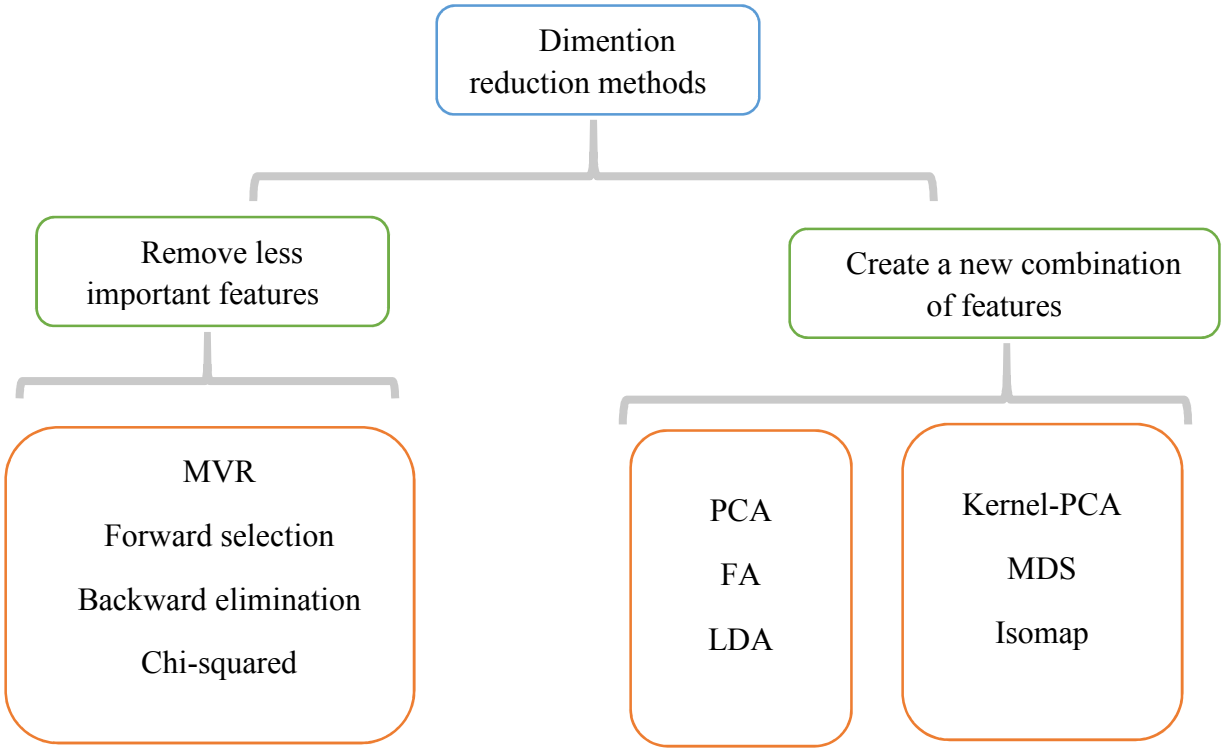

Figure 4.4: Dimension reduction methods

Dimension reduction methods are in two main categories. The more straightforward group methods focus on keeping the essential features in the dataset. No transformation function is applied to the set of features in this method. Backward elimination, Forward selection, and Missing Value Ratio (MVR) are examples of this group.

- **Backward Elimination:** This method eliminates features from a dataset using a recursive feature elimination process. The algorithm first attempts to train the model on

the initial set of features in the dataset and calculates the model's performance. Then, the algorithm eliminates one feature (variable) at each step, trains the model on the remaining features, and recalculates the performance scores. The algorithm repeats eliminating features until it detects a slight change in the performance score of the model and stops there!

For instance, the word "unit" or "vehicle" does not add any meaningful value to the algorithms, and as a result, they are removed from the feature list. So, counting these words does not represent their importance in the text.

- **Forward Selection:** In simple terms, this method is the opposite of the backward elimination process. Instead of eliminating features recursively, the algorithm attempts to train the model on a single feature in the dataset and calculates the model's performance. Then, the algorithm adds features sequentially, trains the model on those features, and calculates the performance scores. The algorithm repeats adding features until it detects a small (or no) change in the performance score of the model.
- **Missing Value Ratio (MVR):** This is a primary dimension reduction method in data mining. MVR attributes the data or features that have a high ratio missing value. We have observed that using this technique would cause losing meaningful information from the data despite standard text. For instance, "Cylinder block" has been used much less than other features in the dataset. The failure ratio of this part is rare compared. However, the representation of this word in the text has a higher weight in classification.

- **Chi-Squared:** The Chi-squared method measures the lack of independence between a feature and a class. The output is a contingency table that identifies the relationship between words in each service report and the output class.

Chi-Square formulates as follows. Where "O" is the observed frequency of the word, "E" is the expected frequency in each class.

$$\chi^2 = \frac{\sum(O - E)^2}{E} \qquad 4.2)$$

Figure 4.5 presents the ten most significant feature distributions using the chi-square method.

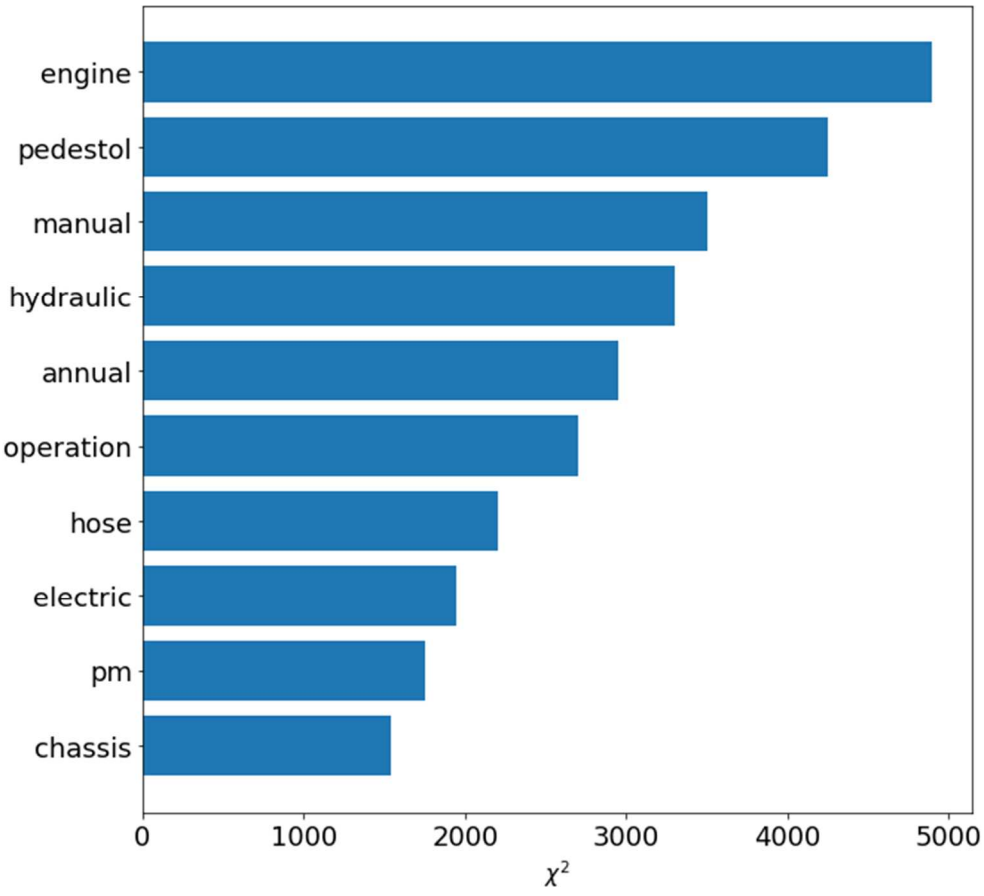

Figure 4.5: Most ten active features using the Chi-squared method

The second group of dimension reductions finds a combination of new features. These methods apply transformation functions to the features. The algorithm's outcome creates a new set of features that contains different values instead of the original ones. These methods are also into two main categories of linear and non-linear methods. Non-linear methods are well known as manifold learning. Principal Component Analysis (PCA), Linear Discriminant Analysis (LDA), and Factor Analysis (FA) are the main types of linear dimensionality reduction methods. Kernel PCA, Isometric mapping (Isomap), and Multidimensional Scaling (MDS) are the main non-linear dimension reduction methods.

- **Principal Component Analysis (PCA):** PCA transforms a set of correlated variables (p) into a smaller n ($n < P$), the number of uncorrelated variables, which are known as principal components. At the same time, it retains as much of the variation in the original dataset as possible.
- **Factor Analysis (FA):** Factor Analysis reduces the data's dimension and is a practical approach to finding latent variables that generally are not directly measured in a single variable dimension reduction method.
- **Linear Discriminant Analysis (LDA):** LDA is generally used for multi-class classification. It is also known as a dimension reduction technique. LDA defines a linear combination of input features that optimizes class separability. However, PCA identifies a set of uncorrelated components of maximum variance in a dataset. Moreover, PCA is an unsupervised algorithm, whereas LDA is a supervised algorithm.

Applying all Preprocessing and feature extraction algorithms, the dataset is ready to be used on different models in the following sections.

We have also used a heatmap to highlight the correlation coefficient between the independent components. It represents the strength of any relationship between two features. In the ML context, the col-linearity between features can reduce the quality of a learning model. Usually, feature selection methods are deployed to decrease high dimensional feature sets to a smaller set for computational efficiency and to reduce noise from redundant features. Figure 4.6 shows the correlation between the features shown in Figure 4.5.

A summary of the frequency of the variety of types of features in the dataset is presented in Table 4.3. Table 4.4 shows the ten most frequent nouns and bigram features in the Service Detail section [64]. Furthermore, " Unit" was the most commonly used noun in Service Detail but did not weight feature extraction and logic detection. The word "unit" was removed from algorithms to reduce the calculation load. This table also presents the distribution of the top ten bigrams in our Service Call Log.

Table 4.3: Feature Extraction summary

	Service Call Log	Service Detail
Name	75938	540930
Verb	3536	327446
Adjective	3434	53670
Adverb	245	110095
Unigram	24	228009
Bigram	31435	125075

Table 4.4:Most frequent nouns and bigrams in Service Detail

Noun	count	Bigram	count
Unit	248470	PM Inspection	14850
Boom	87380	Dielectric Test	13201
Service	43760	Unit Function	11890
Checked	41326	Hydraulic Leak	10459
Pole	39375	Boom Function	8900
Winch	38541	Upper Controls	7927
Inspection	37044	Annual PM	7900
Install	35744	Hyd Leak	6566
Hose	34909	Pole Guide	6100
Cylinder	32700	Rotation Gearbox	4300

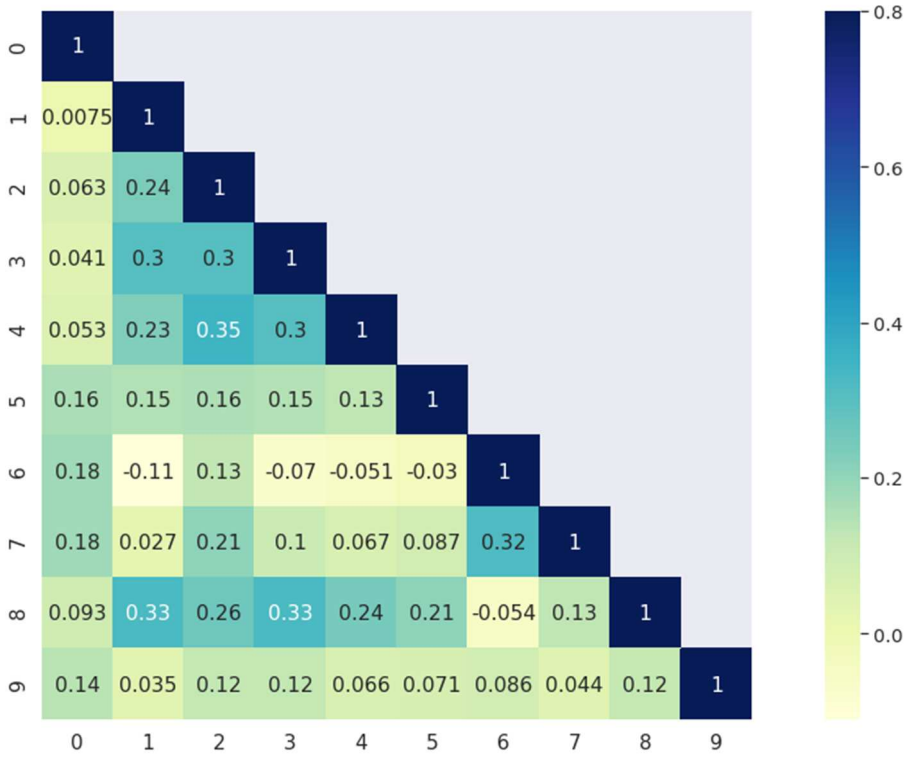

Figure 4.6: Heatmap of correlation between example features

5 Validating Customer Claims

5.1 Introduction

The first step on every customer request is to validate the request. Our model can identify whether the requested service was valid, vague, or invalid. For instance, the service calls that the Service Call Log is not related to Service Detail cannot get used for developing the classification pipeline and eventually get used for part failure analysis. This chapter elaborates on different statistical and deep learning classification models used to investigate the correlation between "Service Call Log" and "Service Detail." Compare and analyze the result.

We have preprocessed data and extracted valuable features in chapter four. In this chapter, we used "Service Call Log" and "Service Detail" and manually labeled the connection between these two in "Relation."

We use statistical and deep learning models to identify valid claims, analysis each model, and identify the best fit for this application.

5.2 Statistical Approach

5.2.1 Support Vector Machine

"Support Vector Machine" (SVM) is a well-known basic supervised machine learning algorithm for classification and regression problems. It was introduced by Vapnik [66] for pattern recognition in image processing. First, SVM plots each data point in an n-dimensional space (where n is the number of features in the dataset). Each feature represents the value of a particular coordinate. Then, the model identifies the best hyper-plane that separates the classes (Figure 5.1).

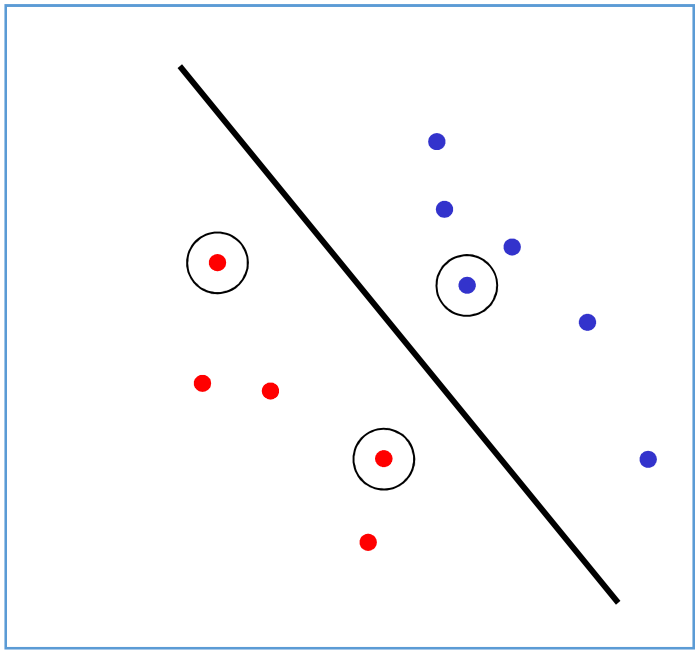

Figure 5.1: SVM: The support vectors are indicated by a circle around them

In this model, vectors are the coordinates of individual observation. The SVM classifier is a frontier that best segregates the two classes (hyper-planes /lines). Given a set of labeled training vectors (positive and negative input examples), each object x_i attached with a label $y_i(y_i \epsilon \{-1,1\})$, SVM builds a linear decision boundary to discriminate between the two classes [67] [68].

Numerous research has been done on the techniques and theories coupled with extensions to regression and density estimation of SVM. The geometrical interpretation of SVM is that the algorithm searches for the two optimal parallel separating surfaces. The SVM was initially introduced on the linearly separable datasets. Later, the kernel approach is added to construct non-linear decision surfaces for the non-linearly separable case. SVM first selects the hyper-plane that classifies the classes accurately. Then it maximizes the margin.

Mars et al. [69] have used SVM to perform sentiment analysis for tweets. Since this application's dataset study has many mixed-up samples, it does not seem to be the best model. We have used

some hyper planning to distinguish the sample; however, the result detailed in 5.4.11 was not satisfying.

5.2.2 Decision Tree

A Decision Tree (DT) is a flowchart structure model where each node represents an explicit feature of the dataset, each branch is used to represent a decision, and each leaf is used to show the outcome. The first node in a DT is known as the root node. It learns to partition based on the feature value. Then, it recursively partitions the tree, also called recursive partitioning. This flowchart structure helps in the decision-making process. DT performs similarly to the human way of thinking, making them easy to follow. However, the major disgrace with decision trees is overfitting, so they usually perform well on the validation dataset but poorly on the test dataset (we detail it in section 5.4.10). Ensemble learning was introduced later on to overcome the overfitting issue.

Ensemble learning is a model that makes predictions based on several different models. Ensemble learning can be more flexible (less biased) and less sensitive to data change by fusing several different models. The two most common ensemble learning approaches are bagging and boosting.

- **Bagging:** Training a couple of the same models in parallel. Each one learns from a subset of the data. Random Forests (RF) is the most known bagging model form of DT. Each tree is trained on an arbitrary subset of the same data. The result from all trees is averaged to find the classification. The RF used in this approach is detailed in section 5.2.4
- **Boosting:** Training two or more models sequentially so that each model learns from the mistakes of the previous model. We have used information gain to pick the correct variable in the decision tree. It calculates the accuracy based on each parameter and picks the best parameter first. In other words, it uses information gain. The loss function for multiple

classification algorithms is cross-entropy. Gradient Boosting is the main application of boosting method in DT. It is detailed in section 5.2.3.

5.2.3 Gradient Boosting Tree

Boosting, introduced in 5.2.2, is a procedure in simple terms that means improving the model's performance in a sequential pattern. In boosting, weak learners are used, which perform only slightly better than a random chance. It focuses on sequentially adding up these weak learners and removing the observations that a learner gets correct at each sequential step.

The gradient Boosting Tree (GBT) model is a boosting method incorporated into the DT. The weak learning models are the DTs. In boosting mechanism, all the trees got connected in series, and each tree attempts to reduce the error of the previous one. Since GBT has a sequential topology, it is a slow learner and should have higher accuracy than DT.

The final model combines the result of each step, and finally, a strong learner is developed. The loss function detects the residuals. In this research, we have used logarithmic loss (log loss). The efficiency of the GTB algorithm improves using the sequential boosting method (Figure 5.2).

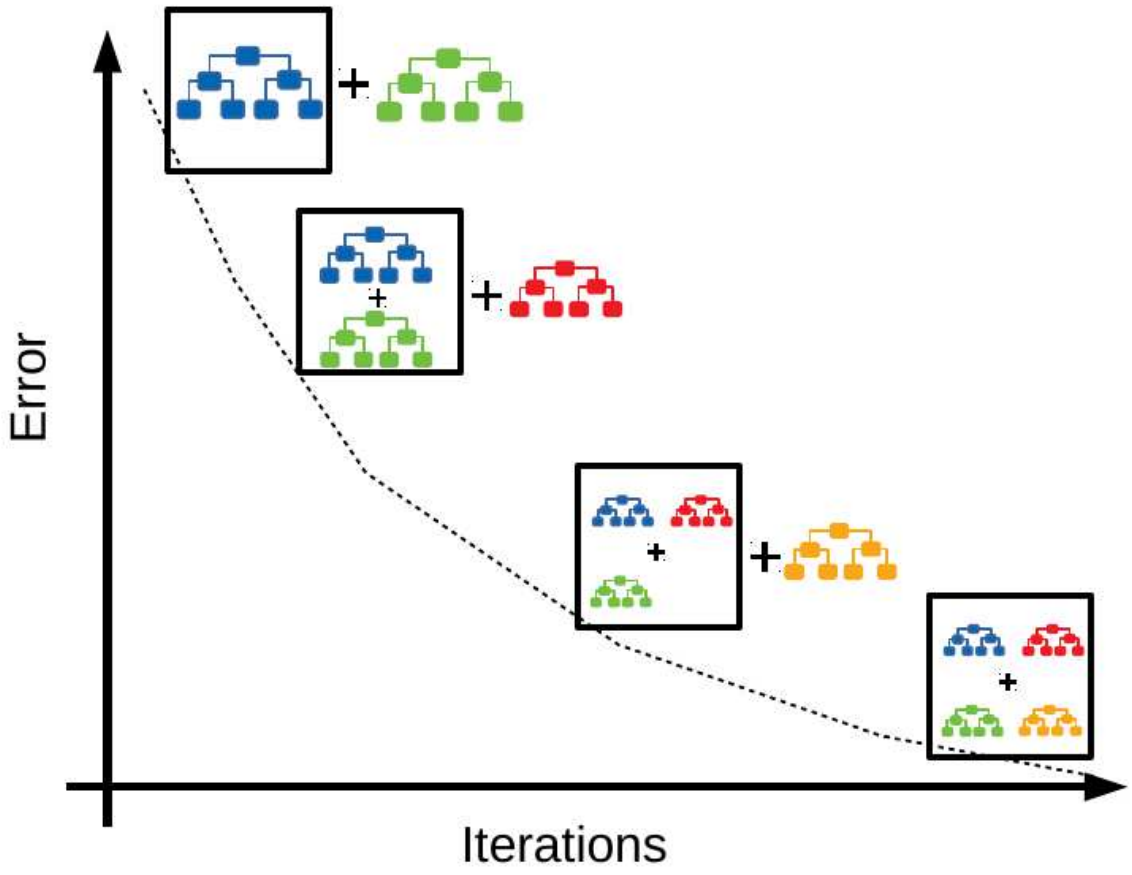

Figure 5.2:GTB efficiency vs. iterations

5.2.4 Random Forest

As its name implies, Random Forest (RF) model consists of many individual DTs. Each tree in the RF dribbles out a class, and the class with the best performance becomes the model's prediction (see Figure 5.3). A simple and powerful concept creates RF architecture.

Dataset is divided into different sections to train each train individually. The low correlation between models is the critical point in this model. The main reason for this behavior is that the trees protect each other from individual errors (given that they do not constantly get all errors in the same direction). While some trees may make false classifications, others make the correct

decision. As a group, the trees tend to move in the correct direction. We have pursued the following conditions to increase RF performance:

1. There needs to be some actual signal in the feature set so that models built using those features do better than random guessing.
2. The individual trees' predictions need to have minimum correlations.

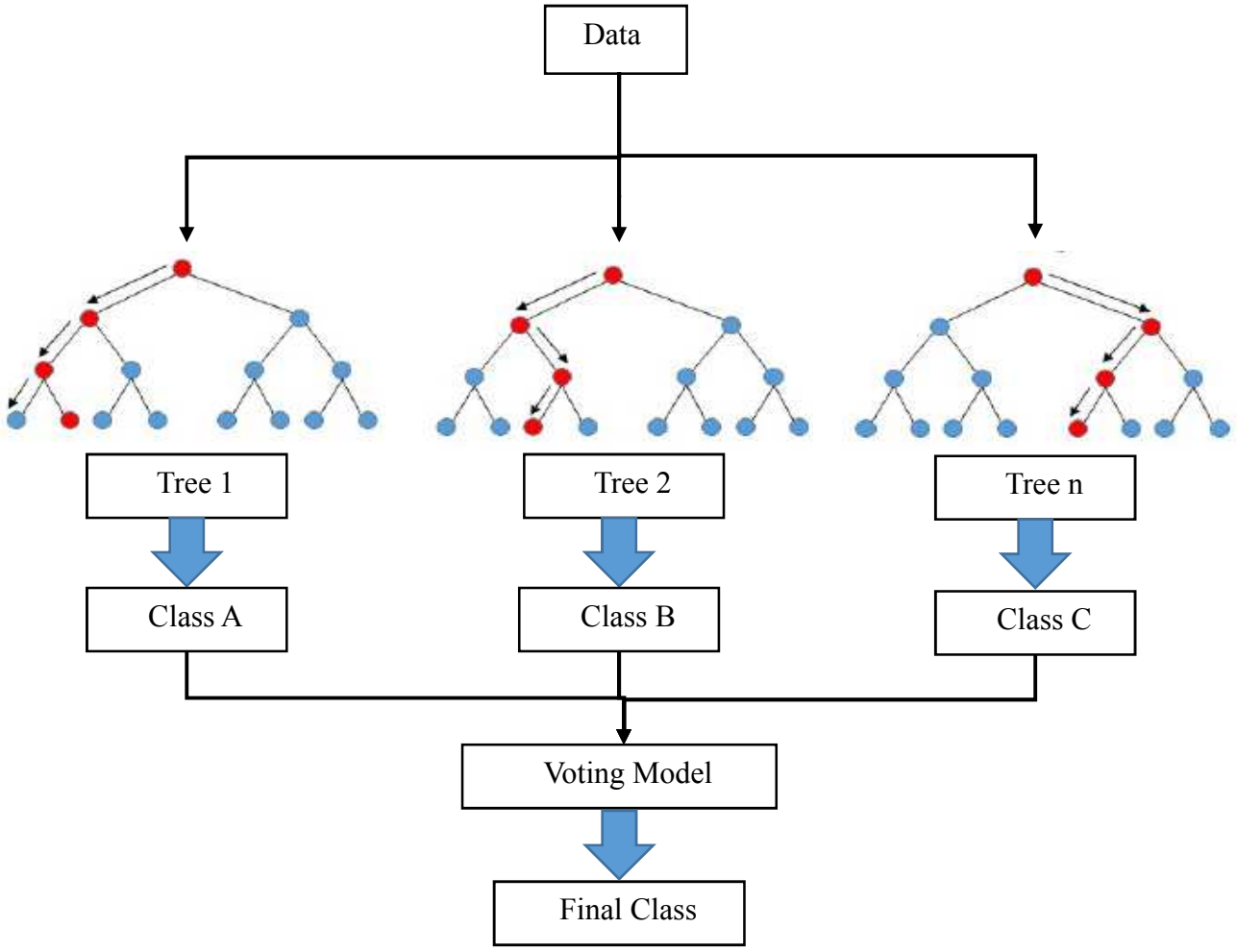

Figure 5.3:Random forest layout

This model has been utilized in similar applications before. Demircan et al. [70] have developed RD for sentiment analysis in the specialty languages like Turkish. It also has been used for sentiment analysis in Twitter comments [71].

5.3 Machine Learning Approach

This section elaborates on how deep learning models are used for request validation. Deep learning is the preferred research approach for complicated NLP tasks. Wee et al. [72] have used Convolutional Neural Network along with Long Short-Term Memory (CNN-LSTM) model to identify metaphors within a text. The LSTM algorithm has also gotten used in question-answering systems and has shown promising results. IBM team has used this method in their products and demonstrated the detail in [73]. Yin et al. [74] have established a comprehensive study on CNN and RMM deep learning networks in NLP applications. The three most widely used type of deep learning models in a representative sample of NLP tasks is CNN, GRU (Gated Recurrent Unit), and LSTM. We found that Recurrent Neural Networks (RNN) perform well and are robust in a broad range of problems except when the task is essentially a keyphrase recognition model, as in some sentiment detection and question-answer matching learnings.

Zhou et al. [75] have utilized two-dimensional max polling CNN algorithm for NLP application. Since we got most of our info from the historical sequence of the word in the text, it was not necessary to use this approach.

5.3.1 CNN-based Model

Convolutional Neural Network (CNN) has been widely used as a successful method of image processing and machine vision. They have been the focus of NLP domain research in recent years. For example, it can be a successful content extraction modeler for domain-based text [76]. Lopez et al. [77] investigated using CNN in NLP applications. Compared to regular, fully connected layers, they have extra layers in the model that enhance their recognition of the pattern (image or text). This layer is detailed in the following section.

Convolution layer: This layer is structured as sliding single or multiple filter windows over the input data. Then, the model performs the convolution operation to find the correlation between the filter and the input sample. This section is a hyperparameter. It is also called a receptive field and matches the filter size. The sliding step is based on the filter size and shifting step stride. As in feed-forward networks, each dot product operation is followed by a non-linear activation function.

For example, After embedding the text to matrix $x.y$ and each word represents a vector. The CNN sweeps a filter of size $x_f . y_f$, where $x_f < x$, $y_f < y$, (We picked filters of sizes two, three, and four) on each spatial position (x', y') to create an output of size of $(x_o . y_o)$. The filter specifies a feature of the combination of words regardless of the location in the input text.

Figure 5.4 represents the overall structure of the CNN model used in this research. A total of five with the size of three sweeps over the input text.

The convolutional layer consists of multiple filters. The output of each filter $(x_o . y_o)$, (5.1, is stacked with other filters' output to form feature maps$(x_o . y_o . n)$, where n is the number of filters (Where x is input sample, w is the weight matrix and b is the bias vector).

$$f(x; w, b) = ReLU(Conv(x; w, b)) \tag{5.1}$$

Every CNN network layer has hyperparameters we need to calibrate to improve performance. Table 5.1 shows a summary of the CNN hyperparameters. The spatial size of the output follows Equation (5.2), where x, x_f, p, s_x represent an input sample, filter size, padding, and stride step, respectively.

It is a standard procedure to utilize pooling layers within CNN models to reduce the dimensions of an input size, which leads to decreasing the number of parameters and computation load of the

network. It functions by applying a reduction operation to a small spatial neighborhood. In addition, the parameters reduction procedure helps to overcome or control overfitting [78].

$$x ==> \frac{x - x_f + 2p_x}{s_x} + 1 \qquad (5.2)$$

$$y ==> \frac{y - y_f + 2p_y}{s_y} + 1$$

Table 5.1:Overview of image-based CNN layers. Input size: x.y.d.

	Conventional	Pooling	Fully Connected
Hyperparameters	Filter size x_f, y_f Stride s_x, s_y *Padding* p_x, p_y Number of filters n	Filter size x_f, y_f Stride s_x, s_y	Number of filters n
Number of train parameters	$(x_f . y_f . d + 1).n$	none	$(x . y . d + 1).n$
Output size	x➔ $\frac{x - x_f + 2p_x}{s_x} + 1$ y➔ $\frac{y - y_f + 2p_y}{s_y} + 1$ d➔ n	x➔ $\frac{x - x_f}{s_x} + 1$ y➔ $\frac{y - y_f}{s_y} + 1$ d➔ n	x➔1 y➔ 1 d➔ n

Pooling layer: The pooling layer functions individually on every input element of convolution matrix outputs and resizes it using the pooling operation. The max-pooling and average-pooling are the most known pooling methods that have been practiced in other research.

The most common form of pooling layer with filters of size two applied with a stride of two. These pooling operations sample every depth slice in the input by two along width and height. The pooling operation can compute a composite value, such as the average from the window's values, or select the maximum values of the selected matrix. As pooling is applied separately on each depth slice, the output volume has the same depth as the input. The max process discards 75% of the entries of the pooling region. It takes a max of over four numbers [79].

As shown in Figure 5.5, the network consists of a single-dimension convolution layer with 300 filters of sizes two, three, and four. The convolution process is applied with padding and stride one on a five-by-five input volume. Each filter (in orange, green, yellow, and red) operates across the two-dimensional matrix to yield four feature maps, one of each filter.

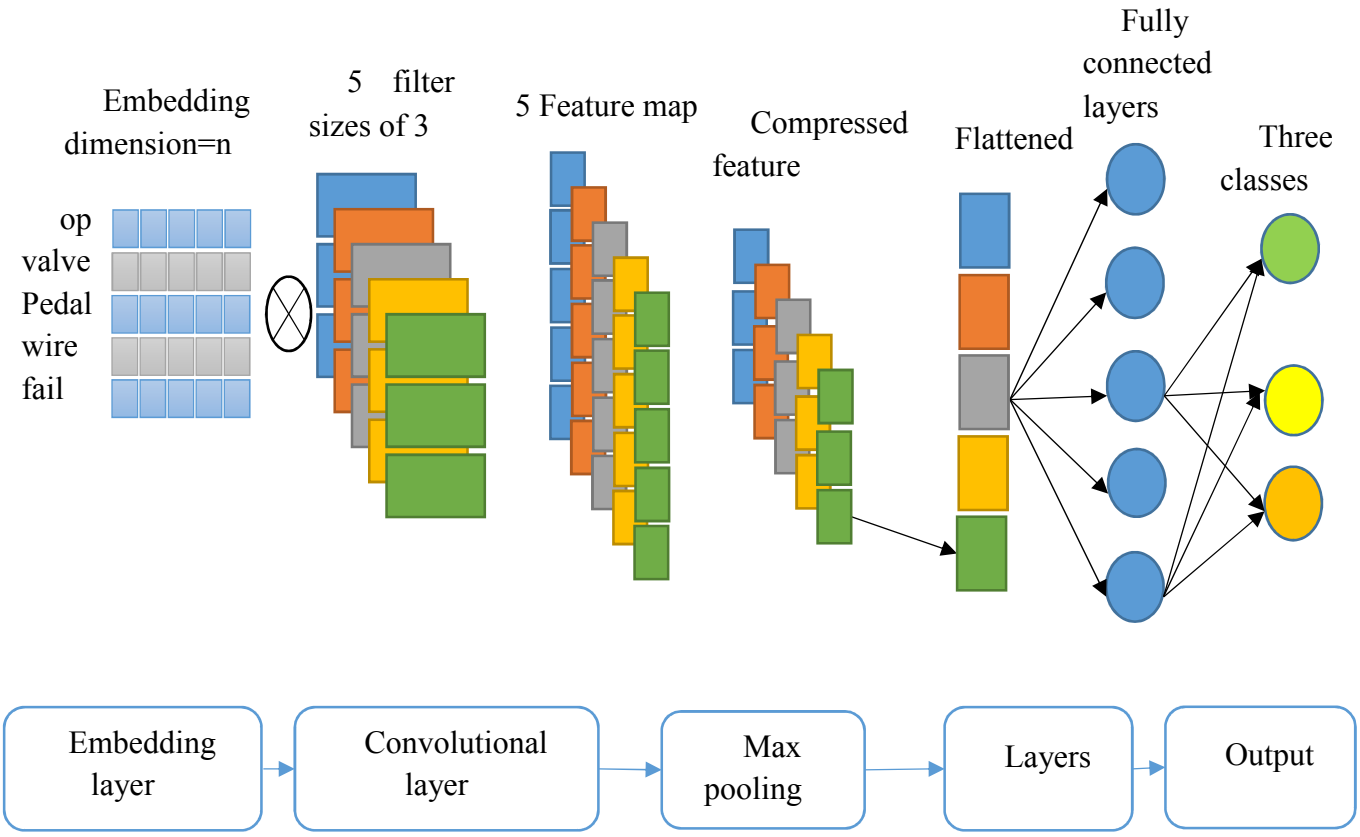

Figure 5.4:NLP CNN architecture

5.3.2 LSTM-based Model

A Recurrent Neural Network (RNN) is one of the most popular architectures used in different NLP tasks. The recurrent structure is very suitable for processing variable-length text. RNN can utilize distributed representations of words. The RNN model first converts the tokens comprising each text into vectors, which form a matrix. The created matrix includes the time step and the feature vector dimension. Then most existing models usually utilize one-dimensional (1D) max pooling or attention-based operation only on the time-step dimension to create a fixed-length vector.

RNNs are a series of feedforward networks. In typical RNNs, each section (labeled as “f” in Figure 5.5) has a simple structure, such as a single $tanh$ layer [80].

Long Short-Term Memory Network (LSTM) is a specific type of RNN that can learn long-term dependencies between input data parts. LSTMs were developed by Hochreiter & Schmidhuber [81] to avoid the long-term dependency loss problem. Therefore, remembering information for long periods is their default function.

LSTM also has a sequential structure similar to RNNs. However, instead of one neural network,

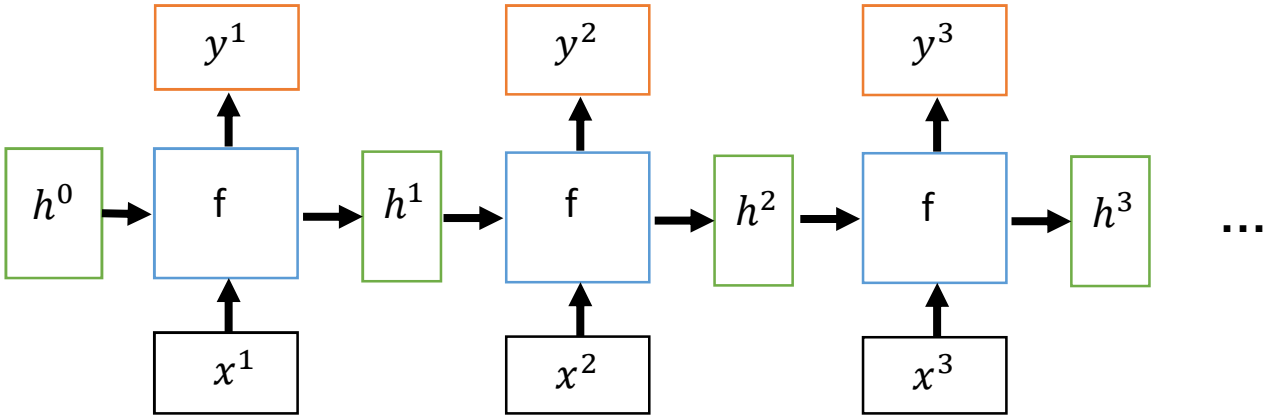

Figure 5.5:Recurrent Neural Network layout

LSTM has the following sections:

1. **Cell gate**: The cell gate transfers information from the previous to the next state. It transfers signals through the entire network chain.
2. **Forget gate**: This section decides which part of the information to throughput from the cell state. A sigmoid function makes this decision. This layer processes h_{t-1} and x_t and outputs a number between zero and one for each number in the cell state C_{t-1}. The higher gain represents keeping the information, while lower values represent neglecting the information

$$f_t = \sigma(W_f.[h_{t-1}, x_t] + b_f) \qquad (5.3)$$

3. **Input gate:** The input section decides where the new information needs to store in the cell state, referring to Figure 5.5. The sigmoid function in the input gate determines which values to update. Then, a $tanh$ layer generates a vector of new candidate values. The last step is to combine the results of the sigmoid and $tanh$ functions to produce an update to the cell state. To filter unwanted information, f_t is multiplied by the old cell state. Then, a piece of new candidate information, it ú C≥t, is added to yield new candidate values, consisting of two main steps described.

$$i_t = \sigma(W_i.[h_{t-1}, x_t] + b_i) \qquad 5.5.4)$$

$$C_t^{\sim} = tanh(W_c.[h_{t-1}, x_c] + b_i) \qquad (5.5)$$

$$C_t = f_t.C_{t-1} + i_t.C_t^{\sim}$$

First, a sigmoid layer is applied to decide the output part of the cell state. Second, a $tanh$ function is used with the cell state to set the values between -1 and 1 and multiply them by the output of the sigmoid gate. This step would control the output as desired.

$$O_t = \sigma(W_o.[h_{t-1}, x_1] + b_o) \qquad (5.6)$$

$$h_t = o_t.\tanh(C_t) \qquad (5.7)$$

Figure 5.6 shows the overall structure of an LSTM model.

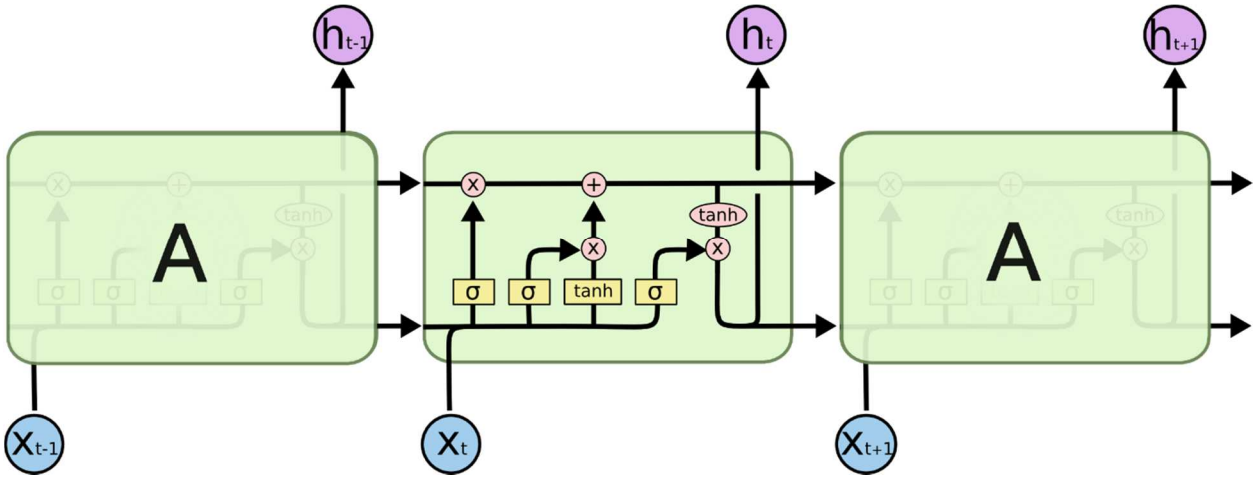

Figure 5.6: Layout of an LSTM cell

5.3.3 BiLSTM-based Model

Bidirectional LSTM (BiLSTM) is the combination of two LSTMs. One section takes the input in a forward direction and the other LSTM in a backward direction. BiLSTMs effectively increase the size of information available to the model, improving the content available to the algorithm (e.g., knowing what words immediately follow and precede a word in a sentence).

Figure 5.7 shows the overall framework of the BiLSTM model used in this research. For a given sentence, the system's input is a one-dimensional matrix composed of the word vectors of all words. The sentence matrix is transformed into a new sentence matrix by the Bi-directional LSTM model. The new sentence matrix is then sequentially passed through convolutional and max-pooling layers for feature extraction. The extracted features are then passed through a dense layer to build a sentence vector for emotion intensity prediction.

Next step, we replaced the LSTM layer with a bi-directional LSTM layer consisting of two LSTMs running in parallel: the first layer on the input sequence and the other layer on the reverse of the input sequence. At each sequential step, the hidden state of the bi-directional LSTM is the

concatenation of the forward and backward hidden states. The hidden state can thus capture both past and future information.

BiLSTM also has increased performance in context extraction. Bahad et al. [82] have used such an algorithm to improve the performance of fake detection Tweets. Also, it is proposed for medical named entity recognition [83].

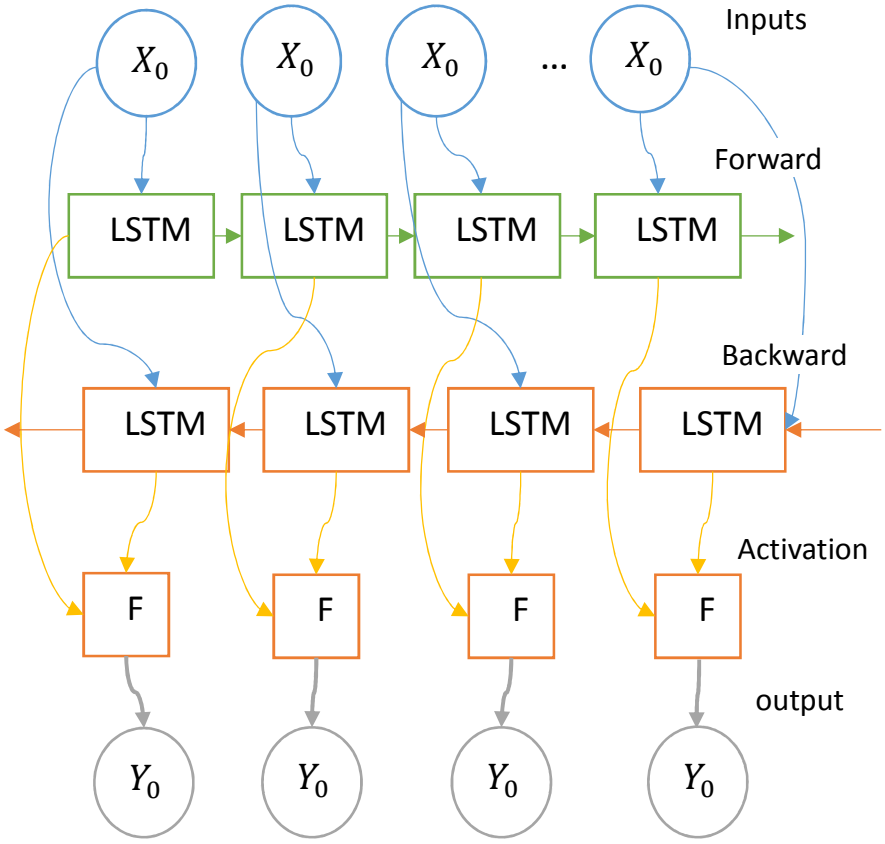

Figure 5.7:Bidirectional LSTM structure

5.3.4 CNN-BiLSTM-based Model

Saaudi et al. [84] employed CNN-LSTM architecture to extract meaningful content from free text. We propose using a granularity level to represent users' log data and textual session-based data samples. The user's behaviors are modeled using character embeddings and a deep learning model that consists of CNN and LSTM. Character embeddings are used to represent the input samples.

Then, a convolution layer is used to capture local trigram features from the input samples, followed by an LSTM layer to consider the order of these given features (tri-grams). We conduct experiments using several variations of model architectures with no handcrafted features. The final CNN-BiLSTM model used in this research to identify valid service requests is shown in Figure 5.8.

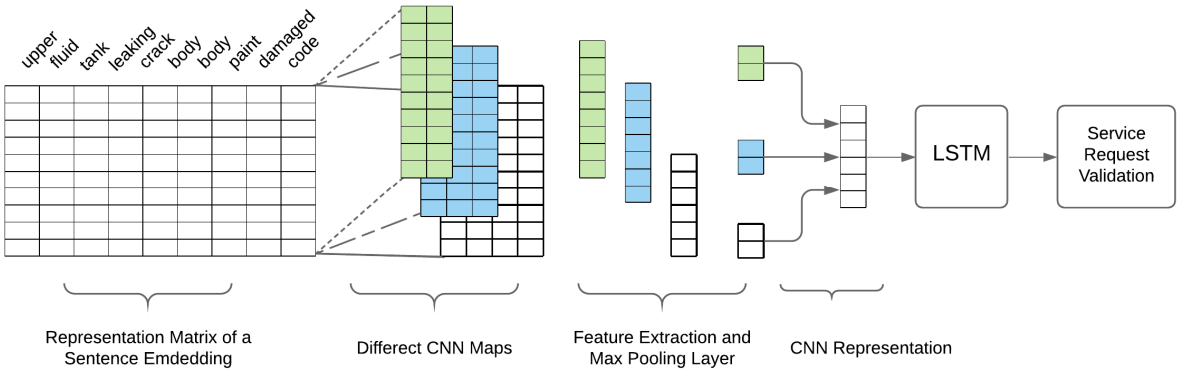

Figure 5.8: Overall BiLSTM-CNN Structure

5.4 Performance Evaluation

In machine learning, the general goal is to predict an outcome using the available data. The prediction model is called the "classification model" when the outcome represents different classes. It is known as a "regression problem" when the outcome is a numeric and continuous function. Many classification models involve only two classes, although issues may need to route the input instance to more than two categories of output. They are known as "multi-class classification."Service request validation is a multiclass classification problem,

The prediction task is addressed using different mathematical techniques in ML models. However, they all share a common factor. They use available information (Xvariables) to obtain the best prediction $\hat{Y}$of the variable Y as output.

In Multi-class models, the variable Y and the prediction $\hat{Y}$ as two discrete random variables and they assume values in $\{1, \cdots, N\}$, and each number represents a different class. The algorithm assumes that a specific data point belongs to one class; then, a classification rule is applied to assign a single class to each individual. The rule is generally straightforward; the most known rule assigns a data point to the class with the highest achieved probability.

A classification model defines the possibility of each data point belonging to a specific class. Starting from the probability assigned by the model, a calibration value is usually implemented to decide which class has to be predicted for each unit in the two-class classification problem. While in the multi-class case, the highest probability value and the softmax are the most employed techniques.

Performance indicators are essential when evaluating and comparing different classification models or machine learning techniques. In this chapter, we cover the evaluation methods practiced in this research.

5.4.1 Evaluation Matrics

Various evaluation methods are developed for the classifier, mainly used for two reasons.

- To investigate the performance of two different models
- To analyze the behavior of the same model by tuning different parameters.

Many methods are based on the confusion matrix since it concludes all the relevant information about the algorithm and classification rule performance.

Parmar et al. [85] elaborate and compare various methods for classifying the multiclass section in an unbalanced dataset.

5.4.2 Confusion Matrix

The confusion matrix is a comprehensive table that records the number of occurrences between two or more classes. The confusion matrix of customer validation is summarized in Table 5.2. For consistency throughout this research, the columns stand for model prediction, whereas the rows display the actual classification.

Therefore, the correctly classified parts are located on the main diagonal from the top left to the bottom right, corresponding to the number of times the two raters agree.

Table 5.2: Truth table example for service request validation

		PREDICTED CLASS			
	Classes	Related	Non-related	Vague	Total
ACTUAL CLASS	Related	**8172**	0	1350	9522
	Non-related	4086	**12285**	1374	17718
	Vague	1362	0	**13620**	14982
	Total	13620	12258	16344	42222

5.4.3 Precision & Recall

In a two-class confusion matrix, the precision is the fraction of True Positive (TP) elements divided by the number of positively predicted units (column sum of the predicted positives). In particular, TPs are the elements labeled positive by the model and are positive in the dataset, while False Positives (FP) are the elements labeled as positive by the model. However, they are not positive in the dataset.

$$Precision = \frac{TP}{TP + FT} \tag{5.8}$$

The precision determines how much we can trust the model when it predicts an individual as Positive or belonging to a specific class.

The recall is the fraction of TP elements divided by the total number of positively classified samples (row sum of the actual positives). In particular False Negative (FN) are the elements identified as negative by the model, but they are positive samples in the dataset.

This function measures the model's predictive accuracy for the positive class. In a simple form, it measures the model's capability to find all positive units (Or belong to a specific class in our case) in the dataset. Hereafter, we have investigated different metrics for the multi-class problems, outlining the pros and cons and detailing which metric would fit this application.

$$Precision = \frac{TP}{TP + FN} \tag{5.9}$$

5.4.4 Accuracy

Accuracy is one of the primary metrics in multi-class classification, and it is directly computed from the confusion matrix referring to Table 5.2.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{5.10}$$

The accuracy is the sum of TP and TN elements divided by the sum of all the confusion matrix entries. TP and TN are the elements correctly classified by the model and are on the confusion matrix's main diagonal. At the same time, the denominator is all the dataset elements. In simple terms, it considers choosing a random unit and predicting its class; accuracy is the probability that the model prediction is correct.

Accuracy returns an overall parameter of how much the model correctly predicts on the entire dataset. The metric's fundamental part is the dataset's single individuals: each unit has the same weight, contributing equally to the accuracy value.

The ideal dataset includes a nearly equal sample of data in each class. In other words, classes are balanced. However, samples belonging to one class might be higher than the other classes in many cases. Hence, this is an imbalanced dataset. In this situation, highly populated classes have a higher weight than smaller ones.

Accuracy is an efficient method to predict the highest number of individuals in the correct class and is more important than class distribution. A practical example is imbalanced datasets (when most units are assigned to a single class). Accuracy tends to hide vital classification errors for classes with fewer sample rates since those classes are less relevant than the larger classes.

Accuracy is a metric that is intuitive and easy to understand. Both in binary cases and multi-class cases, It assumes values between zero and one, while the quantity missing to reach one is called the misclassification rate [86]

5.4.5 Balanced Accuracy

Balanced Accuracy (BA) is a specific type of accuracy used in binary and multi-class classification and is computed starting from the confusion matrix.

$$Balanced\ Accuray = \frac{\frac{TP}{Total\ P} + \frac{TN}{Total\ N}}{2} \qquad 5.11)$$

In simple terms, BA is an average of recalls. Hence, it provides equal attention across the different classes. Therefore, if the dataset is relatively balanced, i.e., the classes are similar in size, accuracy and balanced accuracy converge to the same value. The main difference between balanced and regular accuracy appears if the dataset shows an unbalanced class distribution.

In equation 5.11), smaller classes have more than a relational influence on the accuracy, although their size is reduced in the number of units. This tendency also means that balanced accuracy is

insensitive to imbalanced class distribution and assigns greater attention to the instances coming from minority classes. Versus, accuracy treats all instances alike and usually favors the majority class.

5.4.6 Balanced Weighted Accuracy

The Balanced Weighted Accuracy (BWA) takes advantage of the balanced accuracy formula by multiplying each recall by the weight of its class, namely the frequency of the class on the entire dataset. Then, the sum of the weights is added to the denominator.

$$Balanced\ Weighted\ Accuray = \frac{\sum_{k=1}^{k} \frac{TP_k}{Total_k . w_k}}{k.w} \qquad (5.12)$$

Once recalls have been weighted based on the frequency of each class (w_k), low-frequency classes no longer manipulate the average of recall. As a result, each class has a proportional weight to its size compared with the balanced accuracy.

BWA is an efficient performance indicator when the goal is to train a classification algorithm on a dataset with many classes since every recall is weighted by the class frequency of the initial dataset. This metric allows separate algorithm performances on the different classes to track down which class causes poor performance. In the meanwhile, it keeps track of the importance of each class.

This property ensures a reliable value of the overall performance of the dataset. This metric has been used as the probability of correctly predicting a given unit.

5.4.7 F1-Score

F1-Score assesses the classification model's performance starting from the confusion matrix. It uses precision and recall measures under the concept of harmonic mean.

$$F1 - Score = 2.\frac{Precision.Recall}{Precision + Recall} \quad (5.13)$$

The F1-Score is a weighted average between Precision and Recall. F1-Score variates its value between 1 and the worst score at 0. Precision and Recall's relative contribution is equal to the F1-Score, and the harmonic mean helps identify the best balancing between the two quantities [11]. The added precision and recall could refer both to binary and multi-class classification. In the models, matric focuses on the positive class, vs. in the multi-class case, it considers all the classes one by one and, consequently, all the confusion matrix entries.

F1-Score tends to assign considerable weight to smaller classes and higher scores to models with similar precision and recall values. For example, Model A with precision equal to recall (75%), and Model B, whose precision is 60% and recall is 90%. Using the mean formula, precision and recall are equal in both models, but using (5.13, F1-Score, model A obtains a score of 75%, while Model B has only a score of 72%. In addition, precision and recall take values in the range of *[0;1],* and when one assumes values close to 0, the F1-Score suffers a considerable drop. The harmonic mean pays greater attention to lower values.

In multi-class models, F1-Score needs to count on all the classes. So it requires a multi-class measure of precision and recalls to be reflected in the harmonic mean. It can get calculated using two methods: Micro F1-Score and Macro F1-Score.

5.4.7.1 Macro F1-Score

Macro-Precision and Macro-Recall calculations are essential to obtain Macro F1-Score. They are calculated using the average precision for each class and the average recall for each actual class. Later, the Macro approach considers all the classes as a fundamental part of the calculation. All

classes get the same weight on average. As a result, there is no distinction between classes with different populations.

Our approach uses the confusion matrix to focus on individual classes and label the tiles accordingly. In particular, we consider TP the only correctly classified data point for desired class, whereas FP and FN are the classified elements in the wrong way on the column and the row of the class, respectively. TN is all the other tiles, where class "b" is considered reference focus. Precision and recall for each class are calculated using a similar binary setting and labeling method. The formulas below represent the two quantities for a generic class k.

$$Precision_k = \frac{TP_k}{TP_k + FP_k} \tag{5.14}$$

$$Recall_k = \frac{TP_k}{TP_k + FN_k} \tag{5.15}$$

Macro Average Precision and Recall are calculated as the arithmetic means of the metrics for single classes.

$$Macro\ Average\ Precision = \frac{\sum_{k=1}^{k} Precision_k}{K} \tag{5.16}$$

$$Macro\ Average\ Recall = \frac{\sum_{k=1}^{k} Recall_k}{K} \tag{5.17}$$

Finally, Macro F1-Score is defined as the harmonic mean of Macro-Precision and Macro-Recall:

$$Macro\ F1 - Score = 2.\frac{Macro\ Ave\ Precision.Macro\ AveRecall}{Macro\ Ave\ Precision + Macro\ Ave\ Recall} \tag{5.18}$$

5.4.7.2 Micro F1-Score

Micro-Precision and Micro-Recall need to be calculated to obtain Micro F1-Score. Micro-averaging utilizes all the units together without considering possible class differences to calculate its parameters. So, the Micro-Average Precision is computed as follows:

$$Micro\ Average\ Precision = \frac{\sum_{k=1}^{K} TP_k}{\sum_{k=1}^{k} Total\ Column} = \frac{\sum_{k=1}^{K} TP_k}{Grand\ Total} \quad 5.19)$$

$$Micro\ Average\ Recall = \frac{\sum_{k=1}^{K} TP_k}{\sum_{k=1}^{k} Total\ Row} = \frac{\sum_{k=1}^{K} TP_k}{Grand\ Total} \quad (5.20)$$

Micro-Average precision and recall seem to have the same values based on equations. Therefore, the MicroAverage F1-Score is just the same. (The harmonic means of two equal values is the same value).

$$MicroAverage\ F1 - score = \frac{\sum_{k=1}^{K} TP_k}{Grand\ Total} \quad 5.21)$$

Micro-Average F1-Score is just equal to Accuracy based on the above equation. Later, there are pros and cons between the two functions. Both methods give more attention to larger classes than smaller classes because they consider all the datasets together.

5.4.8 Receiver Operating Characteristics/ Area under the Curve

The most commonly used performance measure for classification tasks is accuracy. It simply measures the percentage of correctly classified samples. However, the accuracy is too rough to adequately describe the performance of the classier, a more fine-tuned and sophisticated measure is needed. This issue is due to the nature of the classes, which the input data can unevenly represent.

The frequency of the classes ranges from 7% down to 4% in the studied dataset. It is necessary to look at the true-positive rate (TPR) and the false-positive rate (FPR). The TP rate is also known as sensitivity, while the FP rate is computed as (1 - specificity).

The model only outputs probabilities for the input to be in a particular category. Thus, the experimenter must decide on a threshold probability from where the categorization is considered positive. While the most natural choice is 0.5, other threshold values can also be reasonable, for example, if an FP damage is much more severe than an FN or vice versa. However, the choice of the threshold value is an additional parameter and uses the estimation of performance. Therefore, one compares the True Positive Ration (TPR) and False positive Ratio (FPR) by plotting them against each other in a graph for many different choices of thresholds. The result is known as the Reciever Operating Characteristic (ROC) curve. An independent threshold measure of performance is the Area Under the Curve (AUC).

5.4.9 K-fold Cross-validation

The entire data body is used in two separate sections for training the network and estimating its performance. For this reason, the dataset is split into two parts: training and validation data. The downside of splitting the data is that a part of the data, the validation data, cannot get used for the improvement of the network, and since data in many cases is expensive and hard to obtain, we need available data in an optimum way to train a model.

The usual approach to resolve this problem is K-fold cross-validation. This method ensures the data is used by training several networks with different data splits and averaging over it. The exact procedure is explained in the next paragraph.

The dataset is split into K parts or "folds in K-fold cross-validation." The number of folds, K, is an unfixed parameter that can be chosen taking into account the characteristics of the specific data set. A commonly used value is K = 10. One fold is then taken as the test set for estimating the network's general performance in the end, while all the other folds are combined in the so-called construction set. This approach is made for each section of the K folds. As a result, there are K variations of Test-Construction splits in the end. The construction data is split into N parts for every variation of this split. As before, N variations of splits are made, each consisting of one validation fold and the other N-1 folds combined in the training set. This action results in one inner and outer loop with N and K iterations, respectively. In the inner loop, M networks are trained per value of the investigated hyperparameter. The results are averaged, and the optimal value for the hyper-parameter is found. Then K networks are tested in the outer loop with this hyper-parameter choice, and the results are averaged again.

5.4.10 Bias Vs. Variance

In ML, “Bias” is the difference between the model's average prediction and the correct value. The classification model with high bias is not modeled well to the training data and oversimplifies the model. So, high training and test data errors result from models with high bias.

On the other hand, “Variance” is the prediction variability for a given dataset or a value that details data spread. Higher variance rates indicate that a highly trained model on the dataset does not generalize on the data it has not seen before. Accordingly, such models perform very well on training data but have high test data error rates.

Assuming the function a model needs to predict models as (5-18), and “e” represents the estimation error and normally distributed:

$$Y = f(X) + e \qquad (5.22)$$

The simple format of the squared expected error of such a model is represented by:

$$Error(x) = E[(Y - \hat{f}(x))^2] \qquad (5.23)$$

The $Error(x)$ is further developed as:

$$Error(x) = (E[\hat{f}(x)] - f(x))^2 + E[(\hat{f}(x) - E[\hat{f}(x)]]^2) + e^2 \qquad (5.24)$$

$$Error = Bias^2 + Variance + Unpreventable\ error \qquad (5.25)$$

Underfitting occurs when a model cannot capture the data pattern in supervised learning. As a result, these models usually have higher bias and lower variance. An underfitting situation results from fewer data points to build an accurate model.

The overfitted models capture the noise and the underlying data pattern in supervised learning. This issue usually occurs on training a model with a noisy dataset. These models tend to have low bias and high variance. These models are relatively complicated and likely to be overfitting.

If the trained model is simple and has few parameters to calibrate, it tends to have high bias and low variance. On the other hand, if the trained model has a relatively large number of parameters, the model would have high variance and low bias. As a result, there is an equilibrium point to find the correct balance in tuning parameters to prevent overfitting and underfitting.

A decent classification model has an acceptable balance between bias and variance to minimize the total error. The total model error can get calculated as follows:

$$Total\ Eror = Bias^2 + Variance + Irreducible\ Error \qquad (5.26)$$

An optimal balance of bias and variance would never overfit or underfit the model.

Therefore understanding bias and variance is critical for understanding the behavior of prediction models. Figure 5.9 represents the total error for a typical model vs. the complexity of the model, models with high bias or high variance tend to have high errors.

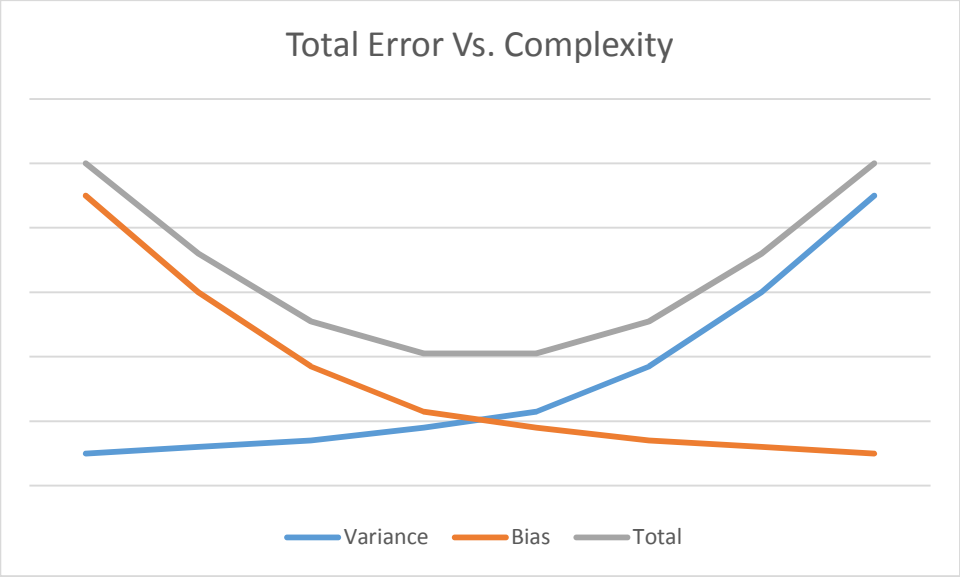

Figure 5.9: Model Error Vs. Complexity

5.4.11 Results

The main goal of Chapter 5 is to determine whether customer-requested service is valid. We used the data in Service Call Log, the Service Detail, and the Relation. This task is known as stance analysis in other domains. We removed service requests that the summary did not match with the reports. As a result, only service requests that the summary matched the reports used in the next section to classify the type of service request. The final results using classification methods represented in this section and the evaluation metrics provided are in Table 5.3.

We have conducted experiments with SVM, DT, RF, and GTB classifiers detailed in 5.2. We used the ten-fold cross-validation technique detailed in 5.4.9 to evaluate the trained models. The initial results of this classical machine learning approach were not satisfying. To improve performance, we attempted to produce a state-of-the-art sentiment classifier using Convolutional Neural Networks (CNN) and Long Short Term Memory (LSTMs) networks detailed in 5.3.

Method	Accuracy	Sensitivity	Specificity	Precision	F-score
DT	0.4971	0.5105	0.4791	0.554	0.5314
RF	0.5360	0.5688	0.5002	0.5510	0.5592
GTB	0.5732	0.6318	0.5207	0.5465	0.5861
LSTM	0.5991	0.6519	0.5406	0.5449	0.5936
BiLSTM	0.6770	0.6976	0.6561	0.5196	0.5956
CNN-BiLSTM	**0.7695**	**0.7843**	**0.7530**	**0.5225**	**0.6272**

Table 5.3:Service Validation Results

Service reports are the input to the model, which are extracted and tokenized in words. Each word is mapped to a vector representation, i.e., a word embedding, such that an entire report can be assigned to an $(s.d)$-sized matrix, where s is the dimension of the embedding space and d is the number of words in the report (d = 500 is picked in the model). The results of the deep learning models are presented in the same table. As shown, we achieved the highest performance using the CNN-BiLSTM model.

The best-performing result of the CNN-BiLSTM Model was utilized using a batch size of 200 after 25 epochs. Figure 5.10 shows that the loss function of the basic BiLSTM architecture for each epoch is improving. Also, loss consistently decreases with the number of training epochs. Figure 5.11 also compares the accuracy of the two methods in training and validation. Similar to

the loss function, the accuracy of the combined CNN-BiLSTM is approximately 80% improved compared to the BiLSTM model. Also, accuracy improves with the number of training epochs.

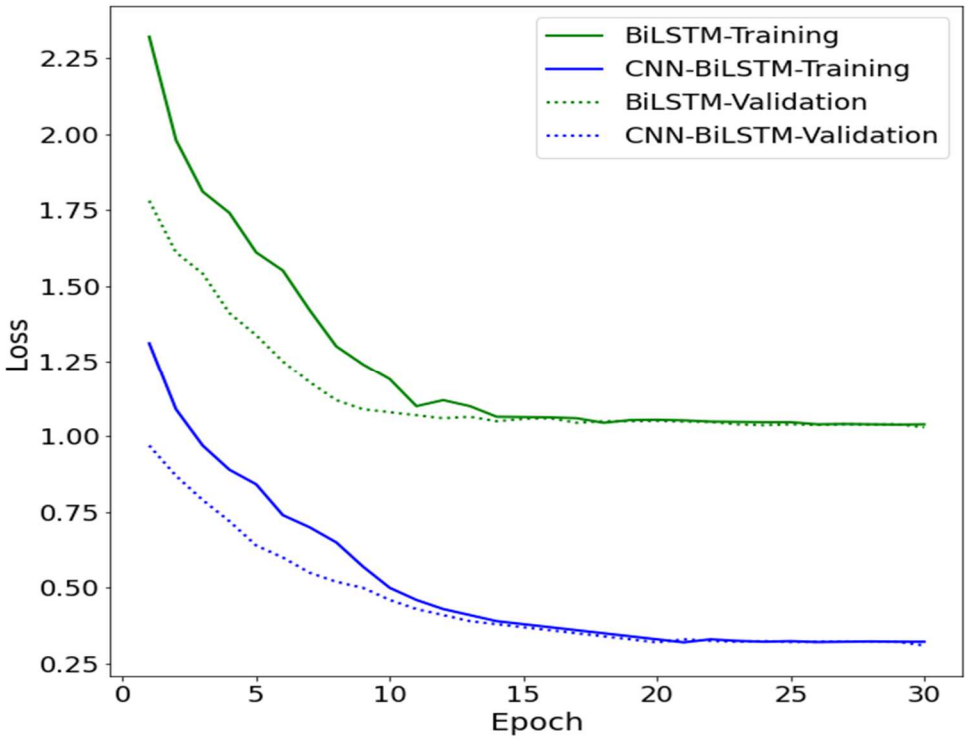

Figure 5.10:Loss in CNN-BiLSTM is compared to BiLSTM machine

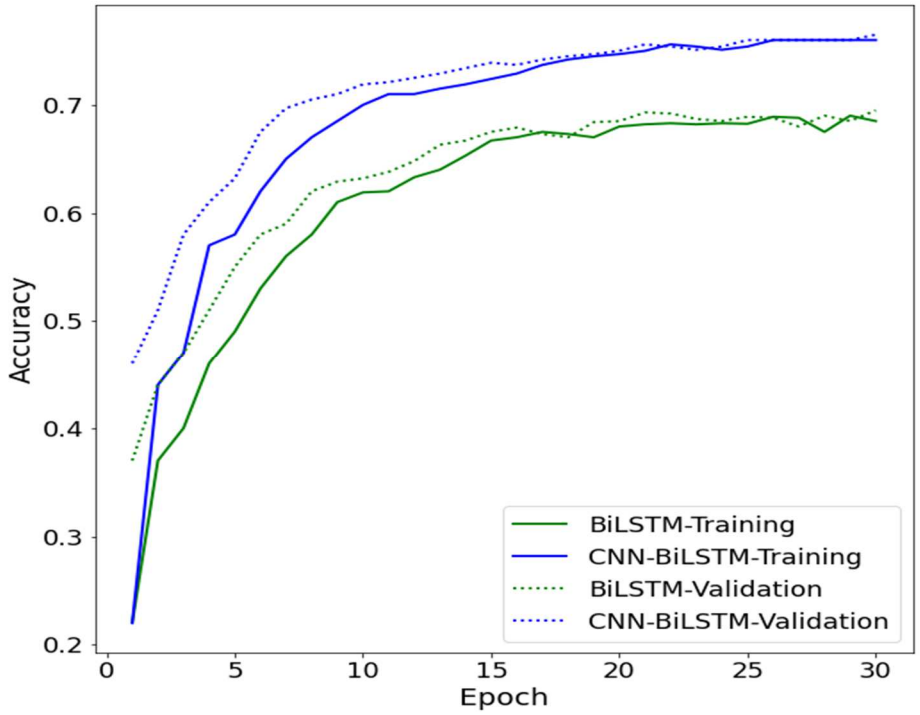

Figure 5.11:Accuracy comparison in CNN-BiLSTM vs. BiLSTM machine

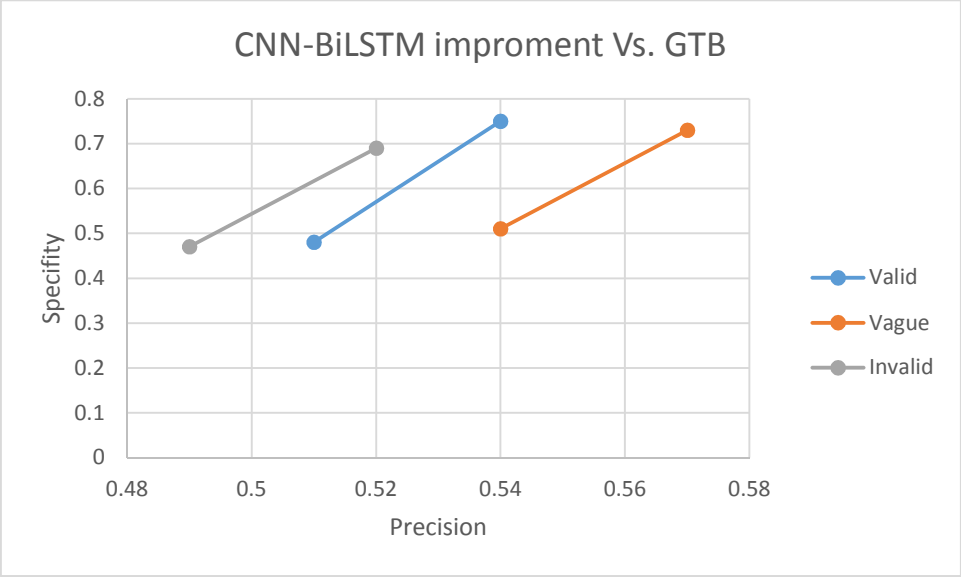

Figure 5.12: Sevice Validation improvement, CNN-BiLSTM Vs. GTB

Deep learning models have improved classification performance across all classes. Figure 5.12 compares the performance of deep learning models vs. the statistical approach. On average, 25% improvement has been achieved using the deep learning model in all performance metrics. (Accuracy, Sensitivity, Specificity, Precision, F-Score)

6 Automated Customer Request Routing

6.1 Introduction

The next step of NLP-based customer service diagnostic involves training classification models to route valid requests to the appropriate department. The process to identify a valid request is detailed in chapter 5.

This section investigates models used in the classification and studies the results using two datasets. First, processed data that validation is not performed and dataset with only valid data.

We used the features extracted from the Service Detail and Service Call Log that is detailed For classification. Also, each request is labeled with the routing department that the service technicians have annotated. So the models in this section are supervised classification.

6.2 Models

Simple and complex classification methods have been implemented on different types of texts. Since previous attempts have not been made to review customer service reports, we compare a set of well-known classifiers. We trained five classifiers, including SVM, Decision Tree, Gradient Tree Boosting, and Random Forest using the preprocessed data from various feature lists. To evaluate the trained models, we used the ten-fold cross-validation technique.

6.3 Performance Evaluation

Based on recorded data, service operators have a 70% accuracy of predicting and routing the customer claim to the proper service department. Also, on average, skilled operators have a 90% chance of predicting and routing the customer's claim to the related service department. It is around 60% for the newer operator.

In our initial experiment, the 2000 most frequent nouns, adjectives, verbs, and bigrams were chosen as input features. Typical classifiers achieved an average accuracy of around 70% utilizing the most common feature.

Our results reveal that names and bigrams have a higher impact on models' performance in this application; comparing the accuracy results of different classifiers with various features, such as nouns, adverbs, and adjectives, exposed that words reflecting sentiments generally impact classification performance. Therefore, fewer adverbs and adjectives were chosen as features.

We have observed that the RF performance is slightly better than the DT since it eliminates the variance in error observed with DT; however, it is more time-consuming to train the model. Traning computation power and time were not an issue in this study.

The performance we observed in the application is slightly different from that of classification methods in other domains, such as customer reviews or medical applications. This change is due to the nature of the technical text.

The results show that by introducing the domain-specific preprocessing and feature extraction methods (explained in 4) in the first step, the GTB classifier performance improved (25% accuracy, 39% sensitivity, 26% f-score, 11% precision, and 11% specificity). The performance of different classifications using domain-based NLP techniques at preprocessing and feature extraction stages, along with CNN-BiLSTM-based deep learning request validation models, are compared in Table 6.1. We have used the performance evaluation methods detailed in 5.4.

The performance is measured in terms of the most available metrics in this study. The Area Under the Curve (AUC) between all 16 departments to are also investigated. Usually, companies pay diverse attention to different types of failure based on none technical factors such as the cost to

address the issue and the severity of downtime of the product. In our study, other mathematical calculations need to get considered to compare the classification methods. Figure 6.1 shows the AUC of ten service department classifications using RF. The model has better success in some departments than others. The AUC difference achieved across different departments is due to vehicle failure. I.e., There is a higher chance of actual failure probability of the engine if this word is identified in the text. On the other hand, the "HV battery" does not have a good AUC. Generally, most electrical component failures would cascade into this component failure in a vehicle.

Finally, the combined two-stage model of Request Validation and customer routing using CNN-BiLSTM/GTB shows ~8% improvement in terms of accuracy compared to simple classification and not using the deep learning request validation section

To provide statistical quantification as to whether a difference in model performance is conclusive enough to state the difference is significant or if the observed difference is by random chance, the statistical significance test for performances of the F1-score is shown in Table 6.1.

Table 6.1:Average improvement of primary vs. deep learning classifier

F1-Score	AVG (%) Improvement	T-test	P one-tail	P two-tail
AVG-IMP comparing GTB	13.5	6.441	1.39E−6	2.78E−6

The most relevant research to our work by Jalayer successfully applied RF to free-text police accident reports to route hydroplaning crashes with a precision of 0.8136, a sensitivity of 0.6234, and an accuracy of 0.6429. We did not have access to their model or dataset to make direct comparisons, but our model achieves a comparable accuracy level.

Finally, Table 6.2 summarizes the classification model's performance, with and without the deep learning request validation model.

Table 6.2: Classification performance results

Service Classification Mode	CNN-BiLSTM Based Request Validation	Accuracy	Accuracy	Specificity	Specificity	F-score	Kappa
GTB	no	0.6018	0.5114	0.7201	0.7168	0.5997	0.22413
	yes	**0.8585**	**0.9080**	**0.8375**	**0.8137**	**0.8583**	**0.6659**
RF	no	0.6351	0.6270	0.6433	0.6395	0.6327	0.2703
	yes	0.8010	0.8548	0.7541	0.7659	0.8079	0.6069
DT	no	0.5261	0.5290	0.5227	0.5441	0.6407	0.0517
	yes	0.6293	0.6672	0.5921	0.6162	0.7208	0.2591
SVM	no	0.4301	0.3882	0.4913	0.5289	0.4478	0.0132
	yes	0.6121	0.5939	0.6327	0.6461	0.6189	0.2549

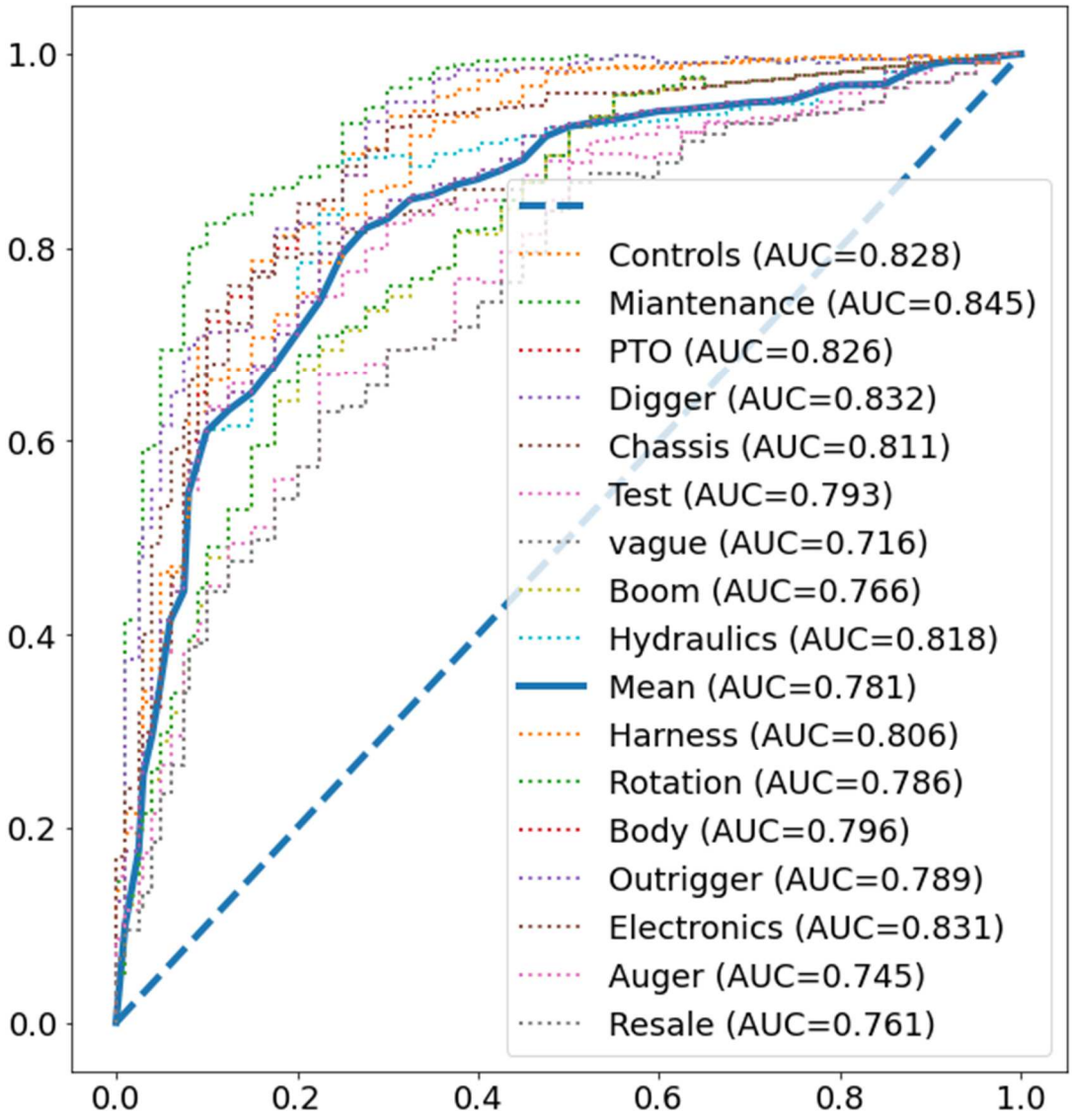

Figure 6.1: AUC-ROC Curve of classification

7 Part Failure Prediction

7.1 Introduction

Electromechanical systems, their components, and individual objects are subject to gradual tear and wear. This issue will ultimately interrupt their proper operation, and they either work out of desired operating condition or stop to operate. The deterioration procedure varies across components and systems and depends on specific operating conditions such as mechanical and electrical stress, load, and environment. Periodic maintenance is vital in assuring the safe and proper operation of the existing systems.

Traditionally, most maintenance activities have taken two approaches: preventive and corrective. The time (or duty-based) preventative maintenance, also known as scheduled maintenance, defines a periodic time interval (or a specific duty), usually based on experience (or tests), to replace the component irrespective of its actual health status. For instance, automotive engineering is the most common application of such a strategy. Time and mileage interval maintenance for vehicle components such as oil, filter, or engine components are simple examples of such approaches. These maintenance items are scheduled to perform after driving for a specific time (or miles).

Preventive maintenance has been an inefficient time and expensive strategy. In addition, this approach does not provide any information about the health status of an element, component, or system, which is a significant defect for safety-critical systems. On the other hand, the corrective maintenance strategy pursues replacing a part once it is partially or no longer operational and cannot perform its assigned task. This maintenance strategy, the most undesirable maintenance, has significant downsides.

It is more time and cost-intensive, does not reduce the risk of catastrophic failures, and causes unnecessary maintenance. Moreover, costs are associated with maintenance labor, downtime, safety concerns, and customer satisfaction. Since a passenger vehicle, the impact on customer satisfaction is a major driving factor because the component might fail miles away from any repair shop. For other safety-critical applications (e.g., aerospace engineering), corrective maintenance is evaded by adopting substitutes in which redundant components are considered since failure is not tolerated. Preventative maintenance expenses constitute a significant portion of the costs of many industrial companies.

The contestation related to these approaches led the researcher to introduce condition-based maintenance (CBM), wherein maintenance actions are accomplished as needed based on the condition of the equipment or component (see Figure 7.1).

CBM reduces maintenance costs by identifying valid maintenance actions based on the observation of abnormal behaviors of a component. It also reduces maintenance costs resulting from unsupervised system failures compared to corrective maintenance.

Also, it lowers the system's downtime, directly translating into significant amounts of money in an expensive fleet like the aerospace industry. CBM maintenance can directly affect the following aspects of a system:

1) Improves the ability to detect faults
2) Enhances the system safety
3) Create better maintenance plans which lead to more efficient operation
4) Reduce inspection time and associated costs
5) Increase the system uptime

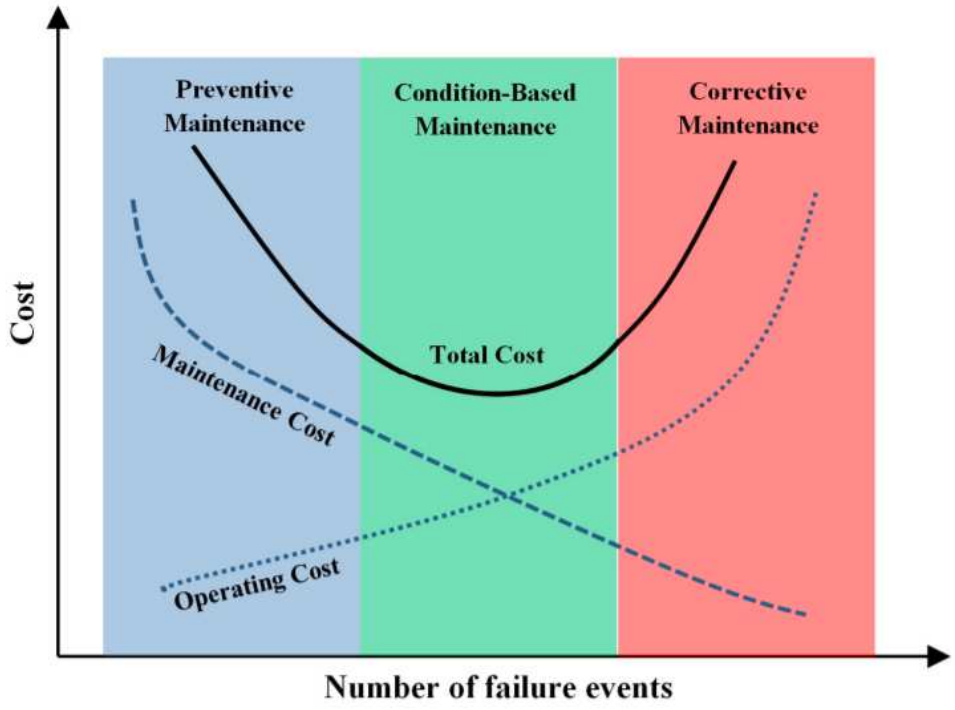

Figure 7.1:Maintenance cost across different maintenance types

The CBM systems usually estimate a system's remaining useful life (RUL) and its constituents or components. In other words, accurate RUL estimation can enable failure prevention in a more controllable manner in that effective maintenance can be executed appropriately to correct impending faults. In this chapter, we first investigate current methods and research on CBM in the "Background and Related Work" and then introduce different Markov models. Next, we elaborate on the Markov model we implemented in this research and is implemented in this research to identify failed components and finally evaluate the overall pipeline introduced in this research.

A vehicle's part failure probability depends on many parameters, such as historical service information, failure of other parts in a vehicle's location and time of use, and historical operating conditions. With a closer look at the dataset we study in this research, we have noticed a high correlation between specific replaced parts. i.e., the hydraulic oil filter and oil will most likely be

replaced on the same service. The deep learning model can benefit from such correlation to improve the model's performance. Figure 7.2 shows the dataset's heat map of the correlation matrix of 10 example parts. The correlation values are between -1 to 1. The higher values identified a higher chance of replacing the part simultaneously.

Furthermore, we have noticed a specific part replacement auto-correlation based on the historical service intervals. I.e., once a vehicle tire gets replaced in a particular service, there is less probability of replacing it or having a flat tire in the next service. Another example, some parts get replaced regularly, such as an air filter. So, the possibility of looking for air filter-related keywords in a specific text is higher once the air filter is on replacement due. Figure 7.4 shows the auto-correlation of an example of hydraulic oil ring replacement on each service. It reveals that the old filer needs to get replaced once every three times of service on average across the vehicles we have studied on this dataset.

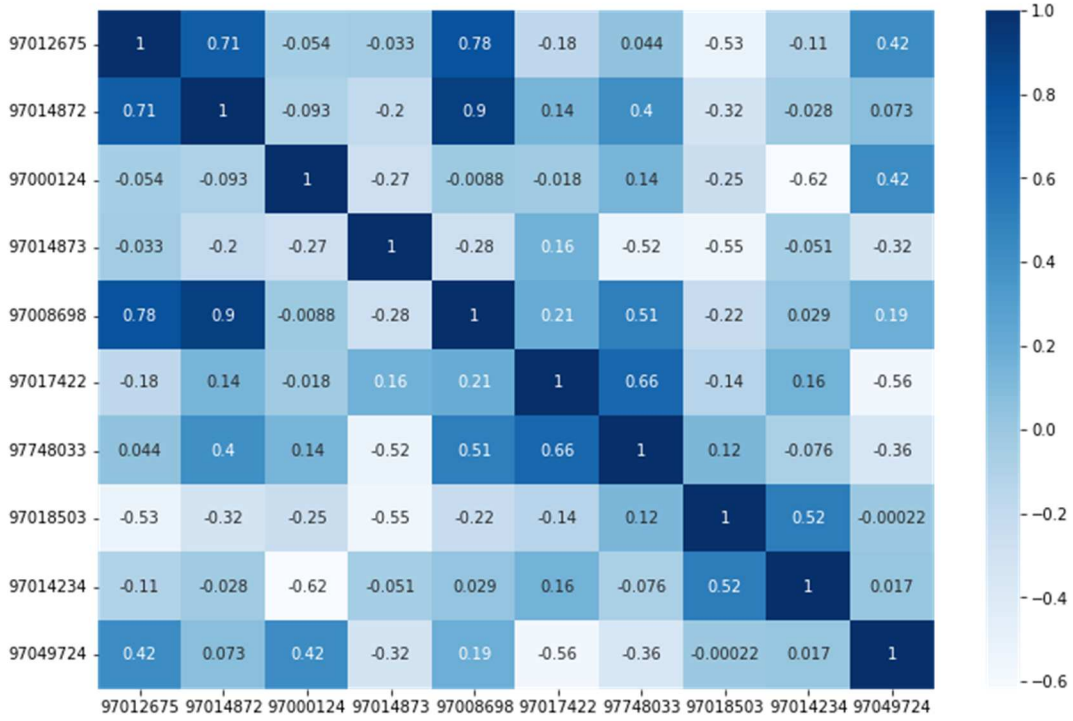

Figure 7.2: Correlation matrix between ten replacement parts

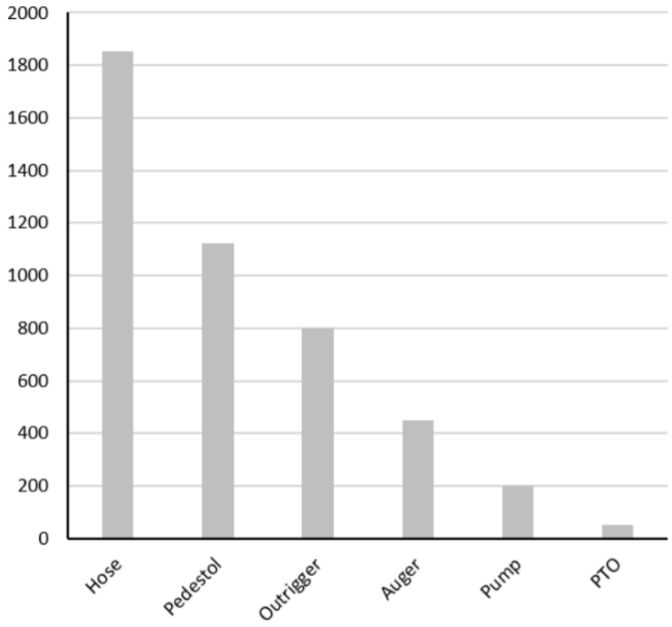

Figure 7.3: Six nouns that most frequently appear in conjunction with" leak" or" leakage"

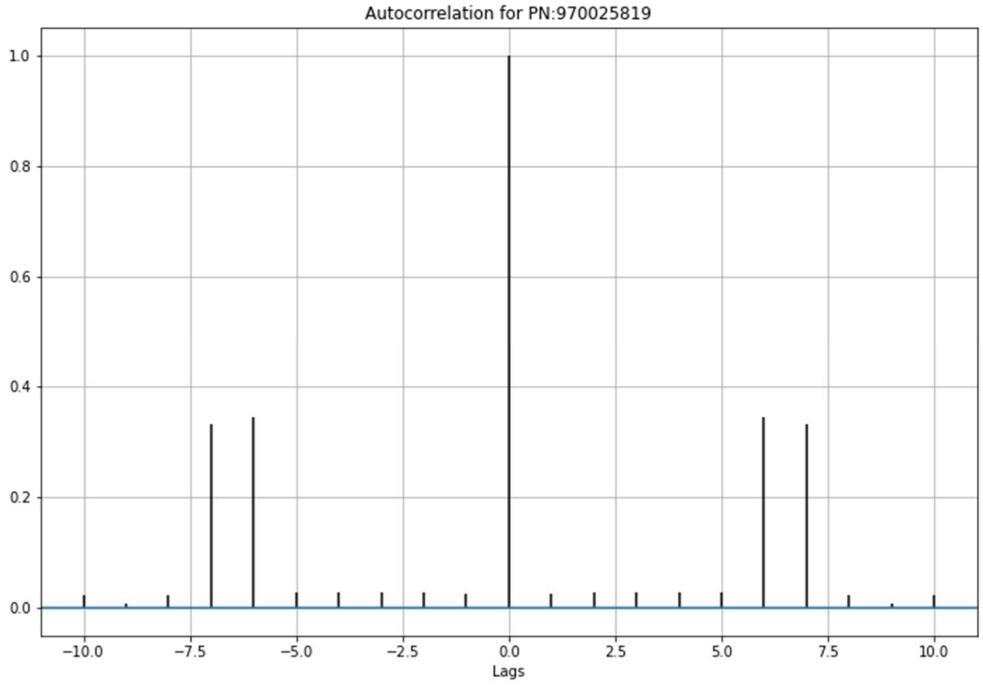

Figure 7.4:Auto correlation of part replacement in different service intervals

7.2 Background and Related Work

Current diagnostics systems can be categorized into three classes, as explained in section 2.2.3. Markov model is a statistical modeling technique with sequential and time-series data. Different Markov models have been successfully applied in various domains, such as pattern recognition and speech recognition [87]. They also have gained increasing attention in diagnosis and prognosis problems. They can depict the system or component's health condition with several meaningful states, such as "healthy" or "failure." Thereby, it can give concise and straightforward explanations for maintenance [88]. Le et al. [89] proposed a Hidden Markov model (HMM) framework for the Remaining Useful Life (RUL) estimation of systems under multiple deterioration modes. Ghasemi et al. [90] developed a method based on HMMs to calculate the reliability function and the mean residual life of a piece of equipment. Jianbo [91] proposed an adaptive-learning-based method for

faulty machine detection and health degradation monitoring with an adaptive HMM. Cholette and Djurdjanovic [92] described a novel data-driven approach based on characterizing the degradation process via a set of operation-specific HMMs to the monitoring systems operating under variable operating conditions. In the following section, we elaborate on the mathematical approach model of the Markov model used in this research.

7.3 Markov Chain

Markov Chain (MC) is the simplest type of Markov model that is widely used in modeling many practical systems such as queuing systems [93], manufacturing systems [94], and inventory systems [95], [96]. Applications of MC in modeling categorical data sequences can also be found in [97]. Definite data sequences (or time series) frequently occur in many real-world applications. A perfectly tuned definite model can make sound predictions and optimal planning in a decision process. A first-order multivariate MC has been proposed and studied by Ching et al. in [98] for multiple categorical data sequences.

Saaudi et al. [99] have used Hidden Markov Model (HMM) to model A user's typical behavior. The model is used to detect any deviation from the expected behavior. We also propose a sliding window technique to identify malicious activity effectively by considering the comparative history of user activity. A typical MC is modeled as follows [100]: Consider a random variable of $X = \{X_t : t \epsilon N\}$. In this model, the distribution of X_{t+1} depends on the past only through the immediate predecessor X_t Where:

$$P(X_{t+1} = x | X_0 = x_0, X_1 = x_1, \dots X_{t-1} = x_{t-1}, X_t = y) = P(X_{t+1} = x | X_t = y) \qquad (7.1)$$

In equation (7.1), x,y, and all x_1 are elements of a given state space, and P represents the probability of each component. For more information on the Markov chain mathematical concept, refer to [101].

7.4 High-dimensional Markov Chain

The high-dimensional or Multivariant MC model shows the behavior of multivariate categorical data sequences produced from a similar source. These models are demonstrated in detail in [102], [103]. They have been practiced in various applications. Zhang et al. [104] have modeled the construction and control of gene regulatory networks in this approach. X_{t+1} in these models, instead of been a single element, is a vector of elements depends on the dimension of the model. Each element's probability is related to all past elements' values in this model.

7.5 High-order Multidimensional Markov Chain

The Probability of each element on the state matrix in a typical MC depends only on the previous state. However, in many issues, this probability can be related to multiple previous sets of data at the same time. For example, Blasis et al. [104] used this model to formulate the wind farm production energy based on a specific location and its associated income depending on multi-variant such as wind physical characteristics. I.e., direction and speed and the dynamics of the electricity cost. Because of the evidence of cross-correlations between wind speed, direction, and price series and their lagged series, they have assessed the income of a wind farm by applying a high-order multivariate Markov model, which includes dependencies from each time series and a certain level of past values.

The overview of the MC implemented in this research is summarized in **Error! Reference ource not found.**

X_i^j represents possibility of failure of X at service interval j and component i. Each service interval represents the time step of the chain. Since vehicles can have multiple services high-order model is necessary, and the dimension of the states represents the parts.

We have identified the 500 most replaced parts in the model, so the Markov matrix dimension length is 500. Each element of the model is formulated as follows:

$$x_{r+1}^j = \sum_{k=1}^{s} \sum_{h=1}^{n} \alpha_{jk}^h \, P_h^{jk} x_{r-h+1}^k, r = n-1, n, \ldots \tag{7.2}$$

The parameters of the equations are described as follows:

- P_h^{jk}: Is the transition matrix of the time step h_t
- $\alpha_{j,k} > 0$ and j,k=1,2,3,…,s are weighting zero, and each element of the transition matrix
- Each element of the transition matrix are defined as follows:

$$\begin{bmatrix} \alpha_{ii}^1 p_1^{ii} & \alpha_{ii}^2 p_2^{ii} & \alpha_{ii}^n p_n^{ii} \\ I & 0 & 0 \\ 0 & 0 & I \end{bmatrix} \tag{7.3}$$

- $P(j,k)$ is the transition parameter which is the probability of failure of a specific part.
- Entries $P(j,k)$ be calculated directly from the categorical data sequences, and $\alpha(j,k)$ be calculated using linear programming.
- We have used maximum likelihood estimation to identify each transition matrix element.

$$P_{ij}^{MLE} \frac{n_{ij}}{\sum_{u=1}^{k} n_{uj}} ver \tag{7.4}$$

7.6 Final Model

Figure 7.5 represents the overall pipeline of this research to improve CNN-LSTM classification performance using the MC attention mechanism. The Attention-based Markov chain (ATT-MC) uses historical replacement part info to identify the most efficient kernels used in the CNN-LSTM model on each service record. By dynamically identifying the proper kernels, deep-learning algorithm calculation time decrease and efficiency increase since CNN will not use kernels that would increase the false positive in the model. The final model uses the following steps to predict future service interval failures and improve diagnostic and classification performance.

1. Perform covariant between replacement parts to identify most replacing rate parts (dimension reduction of Markov chain).
2. Calculate the maximum likelihood for each candidate part.
3. Calculate Markov chain parameters.
4. Estimate the probability of failure of each part.
5. RUN ROC curve for each part
6. Filter kernel pools based on each most probable failed part.
7. RUN CNN-BiLSTM using obtained kernels.

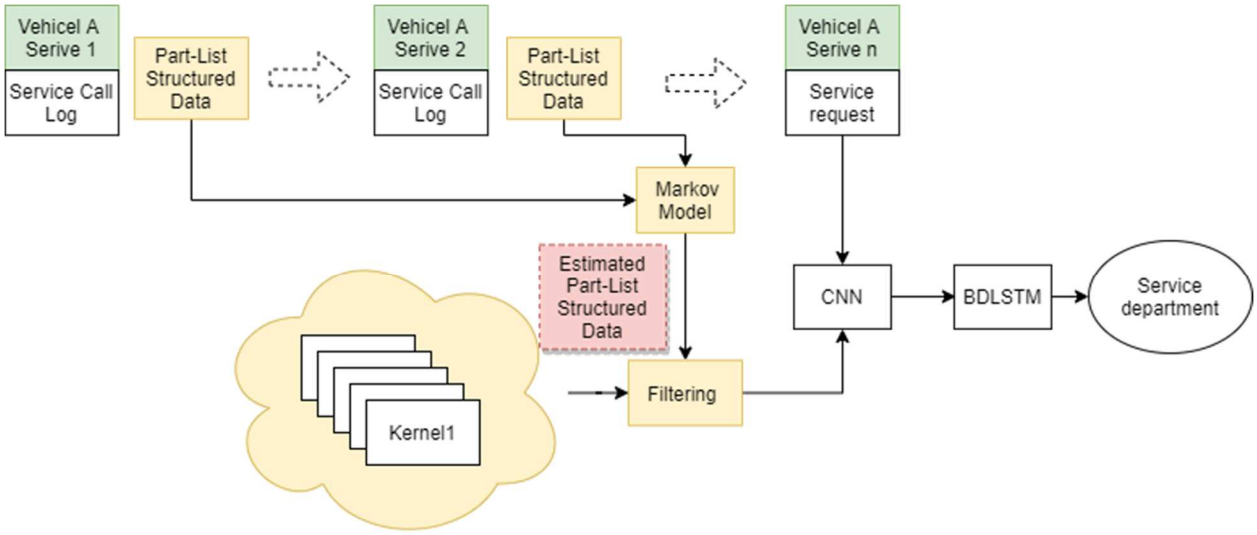

Figure 7.5: The attention-based Markov chain (ATT-MC)

7.7 Performance Evaluation

The evaluation metrics used to identify the performance of the ATT-MC algorithm are detailed in 5.4. We compare the final model attention-based CNN-LSTM performance on the same dataset to identify the valid customer request in chapter 5.

The MC model parameters get estimated in the following steps:

1. $P(j,k)$ is the transition parameter which is the probability of failure of the specific part.

2. Entries $P(j,k)$ can get calculated directly from the categorical data sequences, and $\lambda j,k$ can be obtained by linear programming.

3. Maximum likelihood is used for parameter estimation in this research.

$$P_{ij}^{MLE} = \frac{n_{ij}}{\sum_{u=1}^{k} n_{ju}} \quad 7.5)$$

GTB and CNN-BiLSTM outperformed other validation models in statistical and deep learning methods. We integrated the ATT-MC model into both validation models and compared performance results in Table 7.1.

Table 7.1:ATT-MC performance result on both statistic and deep learning Request Validation

Method	Accuracy	Sensitivity	Specificity	Precision	F-score
BiLSTM	0.5732	0.6318	0.5207	0.5465	0.5861
ATT-MC-BiLSTM	0.6670	0.6599	0.6752	0.5424	0.5844
CNN-BiLSTM	0.7695	0.7843	0.7530	0.5225	0.6272
ATT-MC CNN-LSTM	**0.8417**	**0.8636**	**0.8189**	**0.5254**	**0.6562**

ATT-MC has increased the BiLSTM performance by around 7% in all sectors and 8% in CNN-BiLSTM, respectively. The improvement in some sectors, such as Sensitivity, is more significant than accuracy. The ATT-MC has higher efficiency in removing FP rates in classification. Since it eliminates kernels that would increase the risk of false diagnostics, I.e., since the possibility of a flat tire is lower after a tire change, Having the flat tire Curnel in CNN would increase the chance of false information picked in the text.

The ATT-MC CNN-LSTM has achieved the best performance in identifying valid customer claims. As a result, it is highlighted in the table Table 7.1.

8 Conclusion and Future Work

8.1 Summary

This dissertation introduced a comprehensive vehicle diagnostic and prognostic pipeline from free text and structured recorded data. The main body of our work can be divided into four main steps.

First, we created a domain-specific taxonomy and deployed specific preprocessing and feature extraction NLP techniques to extract meaningful information from free-text reports. We compared feature extraction and dimension reduction techniques and detailed how the modified techniques enhanced the NLP efficiency to extract meaningful information from domain-specific text like vehicle service reports.

Accordingly, we detailed the necessity to filter all vague and non-valid service requests for an automated diagnostic system. To achieve this goal, utilized known classification models to validate the service request. The efficiency of deep learning and statistical models in validating customer service claims is evaluated. The most effective CNN-BiLSTM model reached 84% accuracy and 92% precision in validating customer vehicle service requests.

Once the valid customer request is identified, each request needs to get routed to the proper troubleshooting team. Notably, an automated diagnostic would enhance the processing time. Different classification methods are studied in this next step to route valid customer requests to the relevant service departments. We attained 85% efficiency in terms of accuracy, incorporating two-stage models of CNN-BiLSTM to screen fake/vague service claims and GTB to route the Valid Service claims to the admissible departments. Therefore, CNN-BiLSTM/GTB pipeline exceeds a 70% success rate compared to the average operator.

Finally, this study proposes a novel network structure that employs a multi-variant high-dimensional Markov chain to enhance the CNN-LSTM model performance. The Markov chain model takes advantage of historical records to identify the most efficient CNN kernels in the network structure. The proposed model significantly improved data classification efficiency in correlated historical records such as vehicle service reports. Compared to conventional CNN LSTM models, the introduced model demonstrated significant performance enhancement of 8% Accuracy, 9% Sensitivity, 11% Specificity, 10% Precision, and 12% F-score by reducing the false positive cases in customer claim classification.

8.2 Future Work

Each chapter of this dissertation addresses a specific problem in designing the NLP-based vehicle diagnostic and prognostics, and each solution has room for enhancement and improvement.

In Chapter 4, we presented the Domain-based NLP preprocessing and feature extraction. We have investigated and modified standard known tools at the time. More research and development would introduce more efficient methods and be valuable to investigate in the vehicle industry. Also, We have introduced the vehicle industry taxonomy based on the dataset studied in this dissertation. To the best of our knowledge, there is not a comprehensive NLP pool of data for this field yet.

Chapter 5 incorporates this application's most used statistical models for stance analysis. The poor result of using such models led us to use CNN_LSTM models for mentioned goal. Other models can get utilized to analyze the result. It would be valuable to expand the pipeline of other applications with similar nature text, such as the medical industry.

Chapter 7 is focused on predicting the lifetime of the parts from historical service information. The lifetime expectation of the components can get utilized in inventory management and

improves the efficiency of the CNN model. We have achieved the required efficiency Markov-chain in studies and simplified datasets. Many other Baysian-based sequential models can be investigated to predict historical and correlational relations in more complicated applications such as Kalman Filer or Particle Filter. In order to achieve better calculation time, we reduced the order of the Morkov Chain since the number of service intervals of a specific fleet unit is around 20 years and 95% of the vehicles in the dataset had less than 15 Service intervals. Matching the order of the Markov chain to the actual dataset might increase the model performance

A vehicle has around 20,000 parts in this study. We have simplified the model by creating a virtual Body of Materials (BoM) on parts with a higher correlated replacement rate. For example, Oil filers and rings are two different parts, but it is annotated as individual part. Assigning an element in calculations to each part might outperform the model introduced. This change would increase the calculation load of the model. Decreasing the service overhead cost is one of the goals of the introduced model in this destination. Due to privacy restrictions, we had no access to the financial consequence of each service item. For example, the company overload of replacing a simple item such as an air filter is not the same as replacing a part of the engine in our study. Incorporating the labor and part cost into any applications would help to define gains and weight across multiple parts or classification assets. Moreover, it would ultimately be more valuable to the vehicle industry.

Future work can be done to fine-tune the model and increase its accuracy and training time. The model can also be applied to similar datasets with relational information across each case, such as historical medical reports.

9 References

[1] S. Lam, K. Kristi, G. Wilson and C. j. Holt, "Optimizing Customer-Agent Interactions with Natural Language Processing and Machine Learning," in *IEEE Systems and Information Engineering Design Symposium*.

[2] U. Shafi, A. Safi, S. R. Shahid, S. Ziauddin and M. Saleem2, "Vehicle Remote Health Monitoring and Prognostic Maintenance Systems," *Journal of Advanced Transportation,* 2018.

[3] J. Greenblatt and S. Saxena, "Autonomous taxis could greatly reduce greenhouse-gas emissions of US light-duty vehicles," Nature Climate Change, July 2015.

[4] C. Reisenwitz, "16 Call Center Stats to Help You Stay On Top of the Trends," blog.capterra.com, 2018.

[5] E. Chowanietz, Automobile Electronics, Butterworth-Heinemann, 1995.

[6] R. Trips, "How chatbots can help reduce customer service costs by 30%," IBM blogs, October,2017.

[7] V. Lo, E. Wen and P. Green, "Development and evaluation of automotive speech interfaces: useful information from the human factors and the related literature," *International Journal of Vehicular Technology,* 2013.

[8] Y. Zheng, Y. Liu and J. Hansen, "Natural Language Processing in Dutch Free Text Radiology Reports: Challenges in a Small Language Area Staging Pulmonary Oncology," *IEEE Symposium on Intelligent Vehicle,* 2017.

[9] S. Das, A. Dutta, M. Jalayer and A. Mudgal, "Vehicle involvements in hydroplaning crashes: Applying interpretable," *Transportation Research Interdisciplinary Perspectives,* no. Elsevier, 2020.

[10] M. Walker, A. Anand and R. Abbott, "Stance classification using dialogic properties of persuasion," *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies,* pp. 592-6596.

[11] K. S. Hasan and V. Ng, "Stance classification of ideological debates: Data, models, features, and constraints," in *International Joint Conference on Natural Language Processing*, 2013.

[12] G. Zarrella and A. Marsh, "MITRE at SemEval-2016 task 6: Transfer learning for stance detection," in *10th International Workshop on Semantic Evaluation*, 2016.

[13] J. W. Du Bois, The stance triangle. Stancetaking in Discourse: Subjectivity, evaluation, interaction, 2007.

[14] D. Kuntal, S. Ritvik and S. Kaushik, "Topical Stance Detection for Twitter: A Two-Phase LSTM Model Using Attention," in *European Conference on Information Retrieval*, 2018.

[15] K. Bontcheva, T. Rocktäschel and A. Vlachos, "Stance Detection with Bidirectional Conditional Encoding," in *Empirical Methods in Natural Language Processing*, 2016.

[16] A. Alayba, V. Palade, M. England and R. Iqbal, "A Combined CNN and LSTM Model for Arabic," in *International Cross-Domain Conference for Machine Learning and Knowledge Extraction*, 2018.

[17] " Fake news challenge stage 1 (FNC-1): Stance detection," http://www.fakenewschallenge.org/, 2017.

[18] A. Khan, B. Baharudin and L. Lee, "A review of machine learning algorithms for text documents classification," *Journal of Advances in Information Technology,* pp. 4-20, 2010.

[19] S. Brindha, S. Sukumaran and K. Prabha, "A survey on classification techniques for text mining," in *Proceedings of the 3rd International Conference on Advanced Computing and Communication Systems*, 2016.

[20] K. Vasa, "Text classification through statistical and machine learning methods: A survey," *International Journal of Engineering Development and Research,* pp. 655-658, 2016.

[21] M. Allahyari, S. A. Pouriyeh, M. Assefi, S. Safaei and D. Gutierrez, "brief survey of text mining: classification, clustering and extraction techniques," . CoRR, abs/1707.02919, 2017.

[22] P. Tsangaratos and I. Ilia, "Comparison of a logistic regression and Naïve Bayes classifier in landslide susceptibility assessments: The influence of models complexity and training dataset size," *CATENA,* pp. 164-179, 2016.

[23] T. Kocatekin and D. Ünay, "Text mining in radiology reports," in *21st Signal Processing and Communications Applications Conference (SIU)*, 2013.

[24] A. Bodile and M. Kshirsagar, "Text mining in radiology reports by statistical machine translation approach," in *Global Conference on Communication Technologies (GCCT)*, 2015.

[25] H.-M. Lu, D. Zeng and H. Chen, "Medical Ontology-Enhanced Text Processing for Infectious Disease Informatics," in *IEEE Intelligence and Security Informatics*, 2007.

[26] I. McCowan, D. Moore and M.-J. Fry, "Classification of Cancer Stage from Free-text Histology Reports," in *International Conference of the IEEE Engineering in Medicine and Biology Society*, 2006.

[27] C. Li, G. Zhan and Z. Li, "News Text Classification Based on Improved Bi-LSTM-CNN," in *9th International Conference on Information Technology in Medicine and Education (ITME)*, 2018.

[28] T. Botsis, M. Nguyen, E. Woo and M. Markatou, "Text mining for the Vaccine Adverse Event Reporting System: medical text classification using informative feature selection," *National Library of Medicine,* pp. 631-638, 2011.

[29] H.-T. Nguyen and B. Le, "Multi-channel LSTM-CNN model for Vietnamese sentiment analysis," in *9th International Conference on Knowledge and Systems Engineering(KSE)*, 2017.

[30] N. Clements, "Introduction to prognostics," in *Annual Confference of Prognostics Health Manage*, 2011.

[31] I. 13381-1, "Condition Monitoring and Diagnostics of Machines Prognostics Part 1: General Guidelines," International Standards Organization, 2015.

[32] M. Pecht, Prognostics and Health Management of Electronics, Wiley, 2008.

[33] N. Kim and D. An, Prognostics and Health Management of Engineering System, Springer, 2017.

[34] Y. Peng, M. Dong and M. Zuo, "Current status of machine prognostics in condition-based maintenance: A review," *International journal of Advanced Technologies in Manufacturing,* pp. 297-313, 2010.

[35] S. Sankararaman and K. Goebel, "Uncertainty in prognostics and systems health management," *Int. J. Prognostics Health Management,* vol. 6, no. 10, pp. 1-14, 2015.

[36] H. Elatter, E. H. and A. Riad,, "Prognostics: A literature review," *Complex Intelligent Systems,* vol. 2, pp. 125-154, 2016.

[37] A. Hess, J. Stecki and R. Clark, "The Maintenance Aware Design environment:Development of an Aerospace PHM Software Tool," PHM technology, 2009.

[38] S. Uckun, K. Goebel and P. Lucas, "Standardizing research methods for prognostics," in *IEEE International Conference on Prognostics and Health Management*, 2008.

[39] E. Zio, Prognostics and Health Management of Industrial Equipment, IGI global, 2013.

[40] A. Jardine, D. Lin and D. Banjevic, "A review on machinery diagnostics and prognostics implementing condition-based maintenance," *Elsevier Mechanical Systems and Signal Processing,* vol. 20, no. 7, pp. 1483-1510, 2006.

[41] O. Dragomir, R. Gouriveau, F. Dragomir, E. Minca and N. Zerhouni, "Review of prognostic problem in condition-based maintenance," in *IEEE European Control Conference (ECC)*, 2009.

[42] D. Kiritsis, C. Emmanouilidis, A. Koronios and J. Mathew, "Engineering Asset Lifecycle Management," in *World Congress on Engineering Asset Management*, 2009.

[43] Y. Peng, M. Dong and M. Zuo, "Current status of machine prognostics in condition-based maintenance: a review," *The International Journal of Advanced Manufacturing Technology,* p. 297–313, 2010.

[44] R. Jaai and M. Pecht, "A prognostics and health management roadmap for information and electronics-rich systems," *Microelectronics Reliability,* vol. 50, no. 3, pp. 317-323, 2010.

[45] Y. Decai, "Physics-of-failure-based prognostics for electronic products," in *IEEE 15th International Conference on Electronic Packaging Technology*, 2014.

[46] J. Sikorska and M. Hodkiewicz, "Prognostic modelling options for remaining useful life estimation by industry," *Mechanical Systems and Signal Processing,* vol. 25, no. 5, pp. 1803-1836, 2011.

[47] M. Pecht and M. Kang, Prognostics and Health Management of Electronics: Fundamentals, Machine Learning, and the Internet of Things, Wiley Online Library, 2018.

[48] J. Jiyu, H. Yao and G. Rizzoni, "Fault Diagnosis for Electric Drive Systems of Electrified Vehicles Based on Structural Analysis," *Transactions on Vehicular Technology,* no. IEEE, 2015.

[49] M. Fabien, A. Sandulescu, X. Kestelyn and E. Semail, "A Method for Fault Detection and Isolation Based on the Processing of Multiple Diagnostic Indices: Application to Inverter Faults in AC Drives," *IEEE Transactions on Vehicular Technology,* 2013.

[50] A. Khodadadi, B. Moshiri and A. Mirabadi, "Assessment of particle filter and Kalman filter for estimating velocity using odometery system," *sensor review,* vol. 30, no. 3, 2010.

[51] K. Javed, R. Gouriveau, N. Zerhouni and P. Nectoux, "A feature extraction procedure based on trigonometric functions and cumulative descriptors to enhance prognostics modeling," in *IEEE Conference on Prognostics and Health Management (PHM)*, 2013.

[52] H. Zhang, R. Kang and M. Pecht, "A hybrid prognostics and health management approach for condition-based maintenance," in *IEEE International Conference on Industrial Engineering and Engineering Management*, 2009.

[53] Ö. Eker, F. Camci and I. Fatih, "Major Challenges in Prognostics: Study on Benchmarking Prognostics Datasets," in *1st European Conference of the Prognostics and Health Management (PHM)*, 2012.

[54] M. Schwabacher and K. Goebel, "A survey of artificial intelligence for prognostics," in *AAAI Fall Symposium*, 2007.

[55] K. Medjaher and N. Zerhouni, "Residual-based failure prognostic in dynamic systems," *IFAC Proceedings,* pp. 716-721, 2009.

[56] K. Javed, R. Gouriveau, N. Zerhouni and R. Zemouri, "Robust, reliable and applicable tool wear monitoring and prognostic: Approach based on an improved-extreme learning machine," in *IEEE Conference on Prognostics and Health Management*, 2012.

[57] S. Das, R. Hall, S. Herzog, G. Harrison and M. Gregory, "Essential steps in prognostic health management," in *IEEE Conference on Prognostics and Health Management*, 2011.

[58] C. Cheng, X. Qiao and C. Zhang, "Data-driven Incipient Fault Detection and Diagnosis for the Running Gear in High-Speed Trains," *Transactions on Vehicular Technology,* no. IEEE, pp. 9566-95763, 2020.

[59] R. Prytz, S. Nowaczyk and S. Byttner, "Predicting the need for vehicle compressor repairs using maintenance records and logged vehicle data," *Engineering Applications of Artificial Intelligence,* 2015.

[60] P. Wolf, A. Chin, B. Baker and J. Tian, "Bidirectionally coupled network and road traffic simulation for improved IVC analysis," in *IEEE,Vehicular Technology Conference*, 2019.

[61] X. ShengSi, W. Xiao, C.-H. Hu and D.-H. Zhou, "Remaining useful life estimation – A review on the statistical data driven approaches," *European Journal of Operational Research,* vol. 123, no. 1, pp. 1-14, 2011.

[62] K. Tsui, N. Chen, Q. Zhou and Y. Hai, "Prognostics and Health Management: A Review on Data Driven Approaches," *Mathematical Problems in Engineering,* no. Hindawi Publishing Corporation, 2015.

[63] E. Nichols and J. Chiu, "Named Entity Recognition with Bidirectional LSTM-CNNs," *Transactions of the Association for Computational Linguistics,* vol. 4, p. 357–370., 2016.

[64] N. Indurkhya and F. Fred, Handbook of Natural Language Processing, Chapman, 2010.

[65] T. Mikolov, I. Sutskever, K. Chen, G. Corrado and J. Dean, "Distributed Representations of Words and Phrases and their Compositionality," *Distributed Representations of Words and Phrases,* 2013.

[66] V. Vapnik., The Nature of Statistical Learning Theory, Springer, 1995.

[67] N. Cristianini and J. Shawe-Taylor, "An Introduction to Support Vector Machines," Cambridge University Press, 2000.

[68] E. Osuna, R. Freund and F. Girosi, "Support vector machines: training and applications," MIT, 1997.

[69] A. Mars, S. Hamem and M. Gouider, "New Ontological Approach for Opinion Polarity Extraction from Twitter," Springer International Publishing, 2017.

[70] M. Demircan, A. Seller and F. Abut, "Developing Turkish sentiment analysis models using machine learning and e-commerce dat," *International Journal of Cognitive Computing in Engineering,* vol. 2, pp. 202-207, 2021.

[71] K. Rajeev and J. Kaur, "Random Forest-Based Sarcastic Tweet Classification Using Multiple Feature Collection," *Multimedia Big Data Computing for IoT Applications,* pp. 131-160, 2019.

[72] C. Wu, F. Wu, Y. Chen, S. Wu, Z. Yuan and Y. Huang, "Neural Metaphor Detecting with CNN-LSTM Model," *Workshop on Figurative Language Processing,* p. 10–114, 2018.

[73] M. Tan, C. Santos, B. Xiang and B. Zhou, "LSTM-Based Deep Learning models for non-Factoid Answer Selection," in *International Conference on Learning Representations*, 2016.

[74] W. Yin, K. Kann, M. Yu and H. Schütze, "Comparative Study of CNN and RNN for Natural Language Processing," arXive.org, 2017.

[75] P. Zhou, Z. Qi, S. Zheng and B. Xu, "Text Classification Improved by Integrating Bidirectional LSTM with Two-dimensional Max Pooling," arxive.org, 2016.

[76] S. Poria, E. Cambria and A. Gelbukh, "Aspect extraction for opinion mining with a deep convolutional neural network," *Knowledge-Based Systems,* pp. 42-49, 2016.

[77] M. M. Lopez and J. Kalita, "Deep Learning applied to NLP," arxive.org, 2017.

[78] I. Goodfellow, Y. Bengio and A. Courville, "Deep learning," MIT press Cambridge, 2016.

[79] J. Johnson and A. Karpathy, "Course notes on cs231n: Convolutional neural networks for visual recognition, https://cs231n.github.io," 2019.

[80] C. Olah, "Understanding lstm networks," 2015.

[81] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation,* vol. 9, p. 1735–1780, 1997.

[82] P. Bahad, P. Saxena and R. Kamal, "Fake News Detection using Bi-directional LSTM-Recurrent Neural Network," in *International conference on recent trends in advanced computing*, 2019.

[83] K. Xu, Z. Zhou, T. Hao and W. Liu, "A Bidirectional LSTM and Conditional Random Fields Approach to Medical Named Entity Recognition," in *Proceedings of the International Conference on Advanced Intelligent Systems and Informatics*, 2017.

[84] A. Saaudi, Z. Al-ibadi and Y. Tong, "Insider Threats Detection using CNN-LSTM Model," in *International Conference on Computational Science and Computational Intelligence (CSCI)*, 2018.

[85] P. Parmar, P. Biju, M. Shankar and N. Kadiresan, "Multiclass Text Classification and Analytics for Improving Customer Support Response through different Classifiers," in *IEEE,International Conference on Advances in Computing, Communications and Informatics*, 2018.

[86] S. CHOI., "DISCRIMINATION AND CLASSIFICATION: OVERVIEW," *Advances in Theory and Applications,* pp. 173-177, 1986.

[87] B. Juang and L. Rabinaer, "Hidden markov models for speech recogni-," *Technometrics,* vol. 33, pp. 251-272, 1991.

[88] W. Liano, D. Li and S. Cui, " heuristic optimization algorithm for HMM based on SA and EM in machinery diagnosis," *Journal of,* vol. 19, pp. 1845-1857, 2018.

[89] T. Le, L. Manual and C. Berenguer, "Multi-branch hidden Markov models for remaining useful life estimation of systems under multiple deterioration modes," *Proceedings of the Institution of Mechanical Engineers,* 2016.

[90] A. Ghasemi, S. Yacout and M. Quoali, "egradation modeling and monitoring of machines using operation-specific hidden Markov models," vol. 59, pp. 45-54, 2010.

[91] Y. Jianbo, "Adaptive hidden Markov model-based online learning framework for bearing faulty detection and performance degradation monitoring," *Mechanical Systems and Signal Processing,* vol. 83, pp. 149-162, 2017.

[92] M. Cholette and D. Djurdjanovic, "Degradation modeling and monitor ing of machines using operation-specific hidden Markov models," *HSE transactions,* vol. 47, pp. 1107-1123, 2014.

[93] W.-K. Ching, Iterative Methods for Queuing and Manufacturing Systems, Springer, 2001.

[94] S. M. o. M. Systems, Buzacott, 1993.

[95] W.-K. Ching, W.-O. Yuen and A. W. Loh, "An inventory model with returns and lateral transshipments," *Journal of the Operational Research Society ,* vol. 54, no. 6, 2003.

[96] S. Nahmias, Production and Operation Analysis, McGraw Hill International, 1997.

[97] W. Ching and M. Ng, Advances in Data Mining and Modeling, World Scientific, 2003.

[98] W. Ching, E. Fung and M. Ng, "A multivariate Markov chain model for categorical data sequences and its applications in demand predictions," *IMA Journal of Management Mathematics ,* vol. 13, no. 3, 2002.

[99] A. Saaudi, Y. Tong and C. Farkas, "Probabilistic Graphical Model on Detecting Insiders:Modeling with SGD-HMM," in *International Conference on Information Systems Security and Privacy*, 2019.

[100] T. Rashid, I. Agrafiotis and J. Nurse, "A New Take on Detecting Insider Threats: Exploring the Use of Hidden Markov Models," in *ACN CCS international workshop on managing insider security threats*, 2016.

[101] C. Geyer, "Practical Markov Chain Monte Carlo," vol. 7, pp. 473-483, 1992.

[102] E. Fung, W.-K. Ching and S. Chu, "Multivariate Markov chain models," in *IEEE International Conference on Systems, Man and Cybernetics*, 2002.

[103] C. Wang, T.-Z. Huang and C. Jia, "A Simplified Higher-Order Markov Chain Model," *International Journal of Mathematical and Computational Sciences,* vol. 7, no. 1, 2013.

[104] S.-Q. Zhang and W.-K. Ching, "A Simplified Multivariate Markov Chain Model for the Construction and Control of Genetic Regulatory Networks," in *IEEE International Conference on Bioinformatics and Biomedical Engineering*, 2008.

[105] A. LaPlante, "6 Tips On Making Interactive Voice Response (IVR) Work For Your SMB.," Centurylink, 2019.

[106] IBM, "What is Natural Language Processing," organization={https://www.ibm.com/cloud/learn/natural-language-processing, 2017.

[107] E. Soysal, J. Wang, M. Jiang, Y. Wu, S. Pakhomov, H. Liu and H. Xu, "CLAMP–a toolkit for efficiently building customized clinical natural language processing pipelines," *Journal of the American Medical Informatics Association,* 2018.

[108] M. Nobe, S. Puts, F. Bakers and S. Robben, "Natural Language Processing in Dutch Free Text Radiology Reports: Challenges in a Small Language Area Staging Pulmonary Oncology," *Journal Digital Imaging,* 2020.

[109] A. Mars and M. S. Gouider, "Big data analysis to Features Opinions Extraction of customer," in *International Conference on Knowledge Based and Intelligent Information and Engineering Systems*, 2017.

[110] R. Blasis, G. Masala and F. Petroni, "A Multivariate High-Order Markov Model for the Income A Multivariate High-Order Markov Model for the Income," in *Energies*, 2021.

[111] M. GÜL and E. ÖZ, "A Simplified Multivariate Markov Chain Model for the Construction and Control of Genetic Regulatory Networks," *Journal of Economics,* vol. 2, pp. 75-88, 2018.

[112] T. Young, D. Hazarika, S. Poria and E. Cambria, "Recent Trends in Deep Learning Based Natural Language Processing," *IEEE Computational Intelligence Magazine,* vol. 13, no. 3, pp. 55-75, 2018.