# Machine Learning Techniques for Detecting BGP Anomalies

by

**Hardeep Kaur Takhar**

Thesis Submitted in Partial Fulfillment of the
Requirements for the Degree of
Bachelor of Applied Science (Honours)

in the
School of Engineering Science
Faculty of Applied Sciences

# APPROVAL

| | |
|---|---|
| **Name:** | Hardeep Kaur Takhar |
| **Degree:** | Bachelor of Applied Science (Honours) |
| **Title of Thesis:** | Machine Learning Techniques for Detecting BGP Anomalies |

Glenn Chapman

Dr. Glenn Chapman, P.Eng
Director, School of Engineering Science

**Examining Committee:**

[illegible]

Dr. Ljiljana Trajkovic, P.Eng. (Supervisor)
Professor, School of Engineering Science

[illegible]

Dr. Ivan Bajic, P.Eng.
Professor, School of Engineering Science

[illegible]

Dr. Faisal Beg, P.Eng.
Professor, School of Engineering Science

[illegible]

Dr. Andrew Rawicz, P.Eng. (Chair)
Professor, School of Engineering Science

**Date Approved:** December 21, 2020

# Abstract

Cyber attacks are unauthorized actions of network scammers and intruders with the motive to destroy, steal, or manipulate sensitive information. This unusual behavior deviating from the usual network activity is known as an anomaly. Breach of network data has severe economic and social repercussions for both individuals and corporations. Detection of network anomalies in order to improve cybersecurity remains a critical and evolving topic. Machine learning techniques are widely advocated for the detection of anomalies due to their strong computational abilities. Border Gateway Protocol (BGP) is a routing protocol that contains information about traffic flows and, hence, extracted BGP update messages may help understand network activities. Machine learning techniques have been effectively employed to detect BGP anomalies. In this Thesis, we employ Broad Learning System (BLS), a recently proposed supervised machine learning algorithm, to detect BGP anomalies during a power blackout event. Its performance is compared with long short-term memory (LSTM), a widely used supervised machine learning algorithm. Metrics such as confusion matrix, precision, sensitivity, accuracy, and F-Score, which rely on the selection or combination of features, are evaluated and used for performance comparison.

**Keywords:** network anomalies, intrusion detection, denial of service attacks, Border Gateway Protocol (BGP), feature selection, machine learning, recurrent neural networks, broad learning system (BLS)

# Dedication

This Thesis is dedicated to my family for always supporting me and showering me with their unconditional love.

# Acknowledgements

I would like to thank my Senior Supervisor Prof. Ljiljana Trajković for her guidance and support. She has always encouraged me in my work. Thank you for motivating me to push harder in my career. I am in debt for all her support and guidance. I want to thank her for all her efforts and time spent guiding me to complete this Thesis.

I would like to thank my Committee Members Prof. Ivan V. Bajić and Prof. Mirza Faisal Beg for providing valuable comments and suggestions. I would also like to thank Prof. Andrew Rawicz for chairing the Thesis defense. Special thanks to Zhida Li and Ana Laura Gonzalez Rios for their guidance and support during this research project.

# Table of Contents

# List of Tables

# List of Figures

# List of Acronyms

|   |   |
|---|---|
| **ANNs** | Artificial Neural Networks |
| **ARP** | Address Resolution Protocol |
| **ASes** | Autonomous Systems |
| **AUC** | Area Under ROC Curve |
| **BGP** | Border Gateway Protocol |
| **BLS** | Broad Learning System |
| **CNNs** | Convolutional Neural Networks |
| **DARPA** | Defense Advanced Research Projects Agency |
| **DDoS** | Distributed Denial of Service |
| **EE** | Elliptic Envelope |
| **EGP** | Exterior Gateway Protocol |
| **FN** | False Negative |
| **FNN** | Feedforward Neural Network |
| **FP** | False Positive |
| **FPR** | False Positive Rate |
| **GRU** | Gated Recurrent Unit |
| **ICF** | Insurance Claim Fraud |
| **IDSs** | Intrusion Detection Systems |
| **IETF** | Internet Engineering Task Force |

|  |  |
|---|---|
| **IF** | Isolation Forest |
| **IGP** | Interior Gateway Protocol |
| **IoT** | Internet of Things |
| **IP** | Internet Protocol |
| **ISP** | Internet Service Provider |
| **IX** | Internet Exchange |
| **KPI** | Key Provider Indicator |
| **LOF** | Local Outlier Factor |
| **LSTM** | Long Short-Term Memory |
| **MRT** | Multi-threaded Routing Toolkit |
| **MSK** | Moscow |
| **NB** | Naïve Bayes |
| **NCC** | Network Coordination Centre |
| **NIDSs** | Network Intrusion Detection Systems |
| **NLRI** | Network Layer Reachability Information |
| **OCSVM** | One-Class Support Vector Machine |
| **RDA** | Regional Dispatch Administration |
| **RIB** | Routing Information Base |
| **RIPE** | Réseaux IP Européens |
| **RIS** | Routing Information Service |
| **RNNs** | Recurrent Neural Networks |
| **ROC** | Receiver Operating Characteristic |
| **SML** | Supervised Machine Learning |
| **SOM** | Self Organizing Maps |

| | |
|---|---|
| **SSAD** | Semi-supervised Statistical approach for Anomaly Detection |
| **SSL** | Semi-Supervised Learning |
| **SVM** | Support Vector Machine |
| **TCP** | Transport Control Protocol |
| **TN** | True Negative |
| **TP** | True Positive |
| **TPR** | True Positive Rate |
| **TSS** | Time Series Split |
| **UES** | Unified Energy System |
| **USL** | Unsupervised Machine Learning |

# Chapter 1

# Introduction

Anomalies are defined as deviations of data from expected behavior. Factors such as the presence of an intruder trying to pollute or leak sensitive information, faulty device or sensor, or detection of medical conditions in medical data introduce anomalies in a regular data. Non-adhering data or patterns are usually referred to as anomalies, outliers, discordant observations, exceptions, aberrations, surprises, peculiarities, or contaminants, depending on the application domain. Anomaly detection is the process of finding such patterns in a dataset. Application of anomaly detection includes credit card fraud detection, insurance, health care, intrusion detection for cyber-security, fault detection in safety-critical systems, or military surveillance for enemy activities [1]. With the expansion of the Internet, digital presence and connectivity are increasing daily, which has led to a rise in cyber threats and crimes. Therefore, the detection of network anomalies has become one of the core tasks to maintain network security and stability in enterprise and Internet Service Provider (ISP) networks. To maintain a secure cloud computing infrastructure (a data-centric network), the designers need to carefully monitor network traffic for unusual or unexpected behavior [2].

Flagging malicious behaviors helps obtain critical information such as fraudulent activity in a bank account, detection of cancer or disease in medical data, or an attack in a communication network. By analyzing maleficent behavior, analysts work towards securing the network service. Hence, the topic of anomaly detection is highly relevant. A two-dimensional dataset is illustrated in Fig. 1.1. A majority of the data are grouped in regions A, B, C, and D, wherein region E has fewer data points. Datapoints x and y have type groups A and B but do not belong to the clusters of majority data and, hence, may be identified as outliers. Anomalies may be categorized as contextual, point, or collective. If a data instance is considered anomalous in a given context but regular otherwise, then the instance is a contextual anomaly. For example, suppose an instrument records a low temperature during a summer month on a hot day. In that case, it is defined as a contextual anomaly because a low temperature would otherwise be considered regular in a winter month. Point anomaly is a data instance that is considered anomalous with respect to the entire dataset.

In Fig. 1.1, data points x and y represent point anomalies. Collective anomalies are related data points whose collective occurrence represents an anomaly while the individual occurrence of each data point is regular. Collective anomalies are usually found in data whose samples are related. The collective occurrence of a low value for a longer duration in a human electrocardiogram will indicate a disease while the low value standalone does not indicate an anomaly [1]. Factors such as a masked action of an attacker, the evolution of otherwise normal network behavior, or distinguishing between noise and anomaly make the task challenging. Furthermore, no one rule applies to the detection of anomalies. Several methods have been developed depending on the area of application, type of labels, and available data [2].

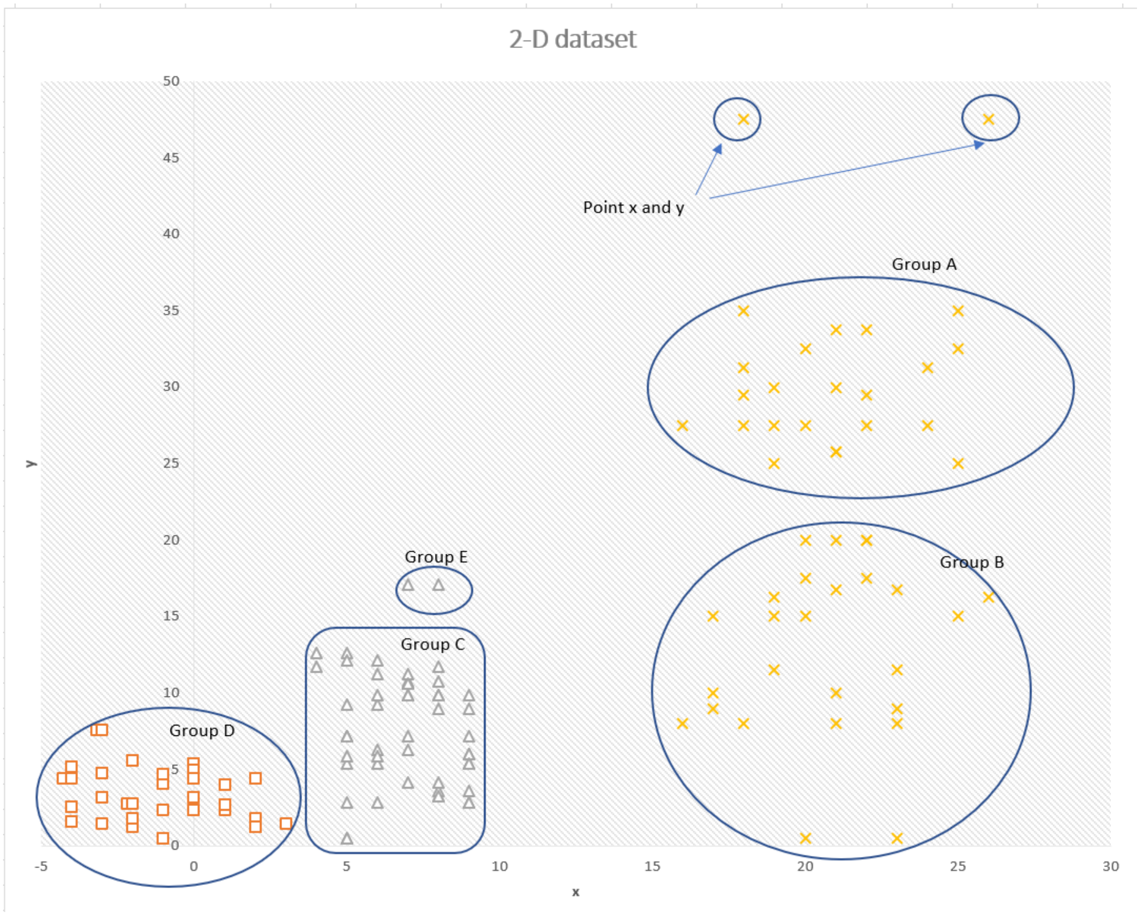


Figure 1.1: Example of point anomalies in a 2-D dataset [1].

## 1.1 Border Gateway Protocol

In this project, the Border Gateway Protocol (BGP) data has been used for analyzing anomalies. Internet Protocol (IP) traffic is routed from source to destination via Autonomous Systems (ASes) using BGP. An AS is a group of BGP routers (peers) managed by a single administrative domain. It is composed of one or more networks that follow common routing policies controlled by an administrator [3]. Devices that are connected to the Internet are also connected to an AS [4]. ASes perform data packet delivery and enable connectivity. BGP determines the best path that a data packet needs to transverse to reach its destination.

BGP relies on the Transport Control Protocol (TCP) to establish a connection (port 179) between the routers. A gateway BGP router establishes a TCP connection with its peers that reside in different ASes. Peering may be defined as a process where different ASes connect and exchange routing information [4]. BGP routing tables are exchanged between the peering routers. BGP allows ASes to exchange reachability information with peering ASes to exchange information about the availability of routes within an AS. Based on the exchanged information and routing policies, BGP routers determine the most effective path a packet should take to reach its destination.

## 1.2 Data Processing

Open, update, keep-alive, and notification messages are exchanged among BGP peers. Upon establishing the BGP session, update messages containing information of all possible routes are shared by routers to update their routing tables. BGP provides update messages only if there is a change within the network's reachability or topological paths. Later, update messages of only withdrawals of existing prefixes are exchanged. BGP update messages contain critical information about the protocol status and configurations. Therefore, data may be generated by extracting fields of the collected BGP update messages during well-known anomalies. The existence of an established connection is ensured by sharing keep-alive messages among peers. If there is a disagreement in the configuration parameters, a notification message is used to close a peering connection. An example of a BGP update message is shown in Table 1.1. It contains useful information such as time, origin, AS-path, next-hop, and the Network Layer Reachability Information (NLRI) announcement prefix. We may extract values of the listed fields in a BGP update message to capture details about the route of a packet [3]. In Table 1.1, the BGP update message was sent from the router with IP address 192.65.184.3 to the router with IP address 193.0.4.28. The next hop of the packet towards its destination is the router with IP address 192.65.182.3. NLRI field consists of a network address of a subnet (prefix) and a length. Prefix listed in the NLRI field is the advertised subnet that may be reached via the BGP neighboring router [49], [50]. AS-path field contains a list of routers available to best route a packet from its source to its destination.

The exchange of routing information among BGP peers is prone to malicious activities such as misconfiguration of BGP routers, power outages, blackouts, and worms. They are known as BGP anomalies and are manifested by anomalous traffic behavior that may have a severe impact on Internet servers and hosts. In communication networks, consequences of BGP anomalies include traffic behavior deviating from its usual profile, the spread of false information by dropping packets or redirecting traffic through unauthorized ASes, and eavesdropping. Power outages may lead to an instrument failure in the network, isolating the impacted networks, and disrupting network services. Events of a power outage on a large-scale impact ISPs due to lack of reliable power backup. Configuration errors such as prefix hijack and routing table leaks usually cause routing errors and disconnections in the Internet [3].

Table 1.1: Example of a BGP update message [3]. IGP: Interior Gateway Protocol, NLRI: Network Layer Reachability Information.

| **Field** | **Value** |
|---|---|
| Time | 2003 1 24 00:39:53 |
| Type | BGP4MP/BGP4MP_MESSAGE AFI_IP |
| From | 192.65.184.3 |
| To | 193.0.4.28 |
| BGP packet type | Update |
| Origin | IGP |
| AS-path | 513 3320 7176 15570 7246 7246 7246 7246 7246 7246 7246 7246 7246 |
| Next-hop | 192.65.184.3 |
| Announced NLRI prefix | 198.155.189.0/24 |
| Announced NLRI prefix | 198.155.241.0/24 |

## 1.3 Detection of Anomalies

An increase in the Internet services has led to a high probability of cyber attacks and crimes, which result in severe economic and social consequences. Analyzing anomalous routing events and their causes helps to prevent future data losses. Statistical models and machine learning methods have been employed for the detection of BGP anomalies such as intrusion attacks, worms, and distributed denial of service attacks (DDoS) [3]. BGP data may be used to develop models using machine learning algorithms to detect anomalies. The process of learning a model (classifier) from known data samples and then categorizing an unseen test data sample as one of the classes by comparing it against the learned model is called classification. Classification results of the generated models are compared by calculating various performance metrics. Classification of the anomalous events aids in designing tools to halt their impact on routing performed by BGP.

The main focus in the research literature has been to develop models for classifying anomalies. In the past, the emphasis was on building statistical models [5]. However, due to high data dimensions, there has been a trend to use computationally viable methods such as machine learning. Patterns in collected data are modeled using machine learning algorithms

[6]. The accuracy of the classifier depends on extracted features, a combination of selected features, and the underlying models. Several rule-based models have also been developed and implemented for anomaly detection. However, these models lack capabilities for adaptable learning. The major drawback of rule-based models is their need to know prior network conditions, slower performance, and higher computational complexity. Anomaly detection, as a classification problem, includes using labels "anomaly" or "regular" for datapoints [5]. Redundancies in the collected data may degrade the performance of the classifying machine learning models. Classification accuracy may be improved by selectively choosing and extracting features from the given data to reduce feature redundancy. Without jeopardizing the accuracy of the algorithm, a subset of desired features may be selected to reduce the training time. Attributes of BGP update messages may be used as guidance for the feature extraction. Projecting features into a lower-dimensional space loses the physical meaning of the features [5], these procedures may be implemented to improve the classification results. Violation of system integrity or a resource is known as an intrusion. Several intrusion detection systems (IDSs) have been implemented to monitor suspicious activities exploiting computer or network resources. IDSs are categorized as misuse-based and anomaly-based. Misuse-based IDSs require a database that contains a description of known attacks for classification. However, they lack the ability to detect unfamiliar attacks. Anomaly-based IDSs rely on regular data to model the classifier. However, anomaly-based IDSs flag unknown attacks and suffer from high false alarms [17]. Several network intrusion detection systems (NIDSs) are based on deep learning algorithms such as convolutional neural networks (CNNs) [45], recurrent neural networks (RNNs) [42], [43], and autoencoders [46] have been used for anomaly detection [44]. Performance comparisons between supervised RNNs (long short-term memory (LSTM) and gated recurrent unit (GRU)) and broad learning system (BLS) used for network traffic anomaly detection have been reported [7], [39]. In this Thesis, the supervised machine learning method BLS is used as an anomaly classifier, and its performance is compared to LSTM.

## 1.4 Organization of the Thesis

In Chapter 1, network anomalies were introduced, followed by a brief description of BGP anomalies along with the importance of anomaly detection. The three machine learning approaches (supervised, semi-supervised, unsupervised) for BGP anomaly detection are discussed in Chapter 2. In the following Chapter, a description of the collected data is provided alongside a pre-processing procedure and suggestions for improving the classification performance. In Chapter 4, two supervised machine learning algorithms (LSTM and BLS) are described followed by the classification procedure and performance evaluation of the model. The importance of $k$-fold cross-validation in machine learning and suggestions to improve the performance of the classifier is also discussed in Chapter 4. The experimental procedure and results are summarized in Chapter 5. Lastly, we conclude with Chapter 6 followed by references.

# Chapter 2

# Related Work

With the advancement of technologies such as the Internet of Things (IoT) and e-commerce, the number of devices connecting to the Internet on an hourly basis has immensely increased. Digital transformation in businesses, learning, entertainment, and commercial online operations have advanced the digitization of all aspects of our daily lives. With such massive operations being conducted online, there is an urgent need to maintain and provide network reliability and sustainability to users. Machine learning is the science of analyzing data by imitating human cognitive abilities (using algorithms) to form decisions (obtaining results using algorithms) and by learning from experiences. We live in a society that thrives on information and computers for success. Large volumes of operational data have been collected by computers in diverse fields that may be used to generate machine learning models to form essential conclusions. The performance of machine learning methods makes them a widely used solution for the detection of anomalous data. Machine learning techniques modify/adjust their parameters to increase accuracy through experience and align their results close to the expected results. Characteristics such as a learning task, performance measurement, and experience gained from the task should be identified to constitute a well-defined learning problem.

The human brain learns a task from experience by remembering, adapting, and generalizing. Hence, these three components are essential to solve a machine learning problem. For example, when encountering an unfamiliar situation, the human brain tries to find similarities from past experiences to formulate a conclusion. A similar phenomenon is executed using machine learning algorithms. Certain software systems emphasize identifying the system's input and output rather than how the output is achieved; a similar approach is followed for machine learning methods, a learning task may define the input to a software system [8].

Classification, clustering, regression, and associative are the four types of learning tasks. Depending on the type of available data, these learning tasks are applied as a solution to a given problem. Classification learning relies on using data that are already classified to build a model. In clustering, groups of similar data are clustered together while in a regression task, numerical approaches are used to fit the results to a curve. In associative learning, a relationship between the data is required. Individual or combination of these learning tasks may be used in machine learning methods [8]. Machine learning is broadly categorized into four categories: supervised, semi-supervised, unsupervised, and reinforcement learning. Three of these approaches are discussed in the following Sections.

## 2.1 Supervised Machine Learning

Various proposed solutions for BGP anomaly detection rely on using machine learning techniques [4]. Supervised Machine Learning (SML) methods are used for anomaly detection when the anomalous data are categorically labeled [9]. Based on the available labeled data samples, they may be considered as a classification or regression task [8]. Predictive models may be built using the sample data for determining the classification of the test data. One of the significant drawbacks of supervised learning methods is the unbalanced distribution of class labels in the training dataset, where the majority of training data samples are regular. Several techniques have been proposed in the literature to inject artificial anomalies into a dataset. It is challenging to accurately distinguish anomaly class labels [1].

Supervised learning algorithms include Support Vector Machine (SVM), Hidden Markov Models, Naïve Bayes (NB), and Long Short-Term Memory (LSTM) [9]. SVM performs best in terms of accuracy and F-Score compared to other machine learning algorithms. However, it is often not selected for implementation due to its computational complexity [9]. SVM effectively detects unknown anomalies [11] but suffers from high false-positive rates [10]. The importance of eliminating irrelevant features in a dataset has been emphasized in the literature to improve the performance metrics of a classifier. [12]. Defense Advanced Research Projects Agency (DARPA) intrusion dataset was used to select the important features from the given 41 features for each of the five classes (normal, probe, denial of service, user to root, remote to local) in order to enhance SVM performance. The subset of important features was selected from the given dataset to improve the performance of SVM but no enhancement was observed in performance (accuracy and F-Score) for the selected important features. Similar SVM classification performance was noticed using all the features and the selected important features. Hence, using SVM to build an intrusion detection system for classification [10], [12] did not take into consideration the interrelationships of the features. In contrast, LSTM has an architecture that employs a gradient-based deep learning algorithm [9], [13]. LSTM learns from past events with longer intervals between events, which makes it more suitable for learning a time sequence. LSTM is an effective time sequence learning classifier and may be suitable for analyzing sequential BGP events for detecting anomalies [9].

BLS is a computationally powerful solution due to its flat architecture and simplicity. However, its application is limited to supervised learning solutions [14]. Supervised machine learning approaches have also been applied to analyze the cellular network traffic using a two-step approach (SVM, LSTM) [15]. Periodic key performance indicators (KPIs) of a cellular network were analyzed using supervised machine learning (SML) techniques in combination with SVM to detect the anomalies at the first stage. During the second stage, LSTM combined with SML is used to analyze the long-term progression of anomalies. This approach was used to automate the labeling process and has outperformed unsupervised labeling procedures [15]. Selection of 60% of data anomalies for training and the remaining 40% for testing has generated the best performance for BGP datasets (Code Red I, Nimda, and Slammer) using supervised learning techniques [7], [16]. When supervised RNNs (GRU and LSTM) algorithms were tested on BGP datasets, LSTM with four hidden layers generated the best F-Score. BLS and its extensions showed comparable performance and required shorter training time due to their broad learning architecture [16].

## 2.2 Semi-Supervised Machine Learning

Data labeling is an expensive task because it requires significant human labor and time [12]. Unlike labeled data, in most applications, unlabelled data are available in abundance and, hence, it is easier to collect or simulate data for the regular class. Therefore, Semi-Supervised Learning (SSL), an approach where only regular labeled data are used for training the model [1], is a preferred method for detecting anomalies when regular class data is available in abundance. It may be considered a combination of supervised and unsupervised learning [14]. It is quite challenging to collect data for every possible anomalous event. Hence, SSL methods are used to build models using regular class data. In cases where anomaly data are challenging to obtain, SSL models are used to identify anomalies. Models are built using training data that contains labels for regular class only and test data are fed to the trained model for predicting labels [1]. SSL methods are categorized as generation-based, difference-based, discriminant-based, and graph-based [14]. Graph-based methods are predominately used in SSL.

SSL is used for anomaly detection in medical and public health records, intrusion detection systems (IDSs), insurance claim fraud (ICF), and fault detection in mechanical units. In medical institutes, human and instrumentation errors may cause critical consequences that could have been otherwise prevented. In IDSs, the challenge of handling a large volume of streaming data that requires computationally effective solutions may be addressed by SSL methods. The large size of the input data results in high false alarm rates in IDSs. Financial losses in ICF may be prevented while wear and tear detection in mechanical parts due to unexpected circumstances may be performed using SSL techniques by generating predictive models using regular class data. Algorithms based on nearest neighbors and clustering such as Self Organizing Maps (SOM) perform better using SSL techniques. The probability of missing anomalies using SSL nearest-neighbor techniques is lower than using unsupervised techniques. However, the likelihood of obtaining high false positives is higher due to a lack of similarly-behaving regular training data points. SSL clustering techniques may be used to improve clusters in the multi-class classification problem. Hence, a better anomaly score may be achieved. SSL methods outperform other machine learning methods for classifying anomalies and detecting variation of anomalies in bulk [1]. Methods based on the matching patterns, strong computational abilities, and statistical discrimination have been suggested to detect anomalies. A two-staged statistical anomaly detection SSL technique called the Semi-supervised Statistical approach for Anomaly Detection (SSAD) [17] has been implemented to build a classification model with a lower false-positive rate. In the first SSAD stage, a probabilistic model (classifier) is built using labeled instances to classify anomalies that exceed the determined threshold. During the second stage, an iterative approach reclassifies anomaly clusters from the first stage outputs by using similarity distances and dispersion rate to reduce the false alarm rate. SSAD was implemented using the NSL-KDD [18] (a derivative of KDD'99) and Kyoto 2006+ [19] (traffic data from Kyoto University) datasets. Its performance was observed to be superior to Naïve Bayes in terms of True Positive Rate (TPR) and False Positive Rate (FPR) [17].

## 2.3 Unsupervised Machine Learning

Supervised learning has a shortcoming of giving a high false-positive rate for unbalanced datasets [10], [20]. Training models without the assignment of labels are classified as Unsupervised Machine Learning (USL). When the sample data labels are unknown, the problem is solved using clustering and association learning tasks [8]. USL algorithms include (One-Class Support Vector Machine (OCSVM), Local Outlier Factor (LOF), Isolation Forest (IF), and Elliptic Envelope (EE) [20]. Unsupervised models use an available dataset to generate a model without requiring class labels. Performing unsupervised learning involves pre-processing the data and segregating the anomalous from the regular datapoints. The regular data are then used for training the model. Finally, the model is tested with unknown data. IF performs the best among the unsupervised techniques [20]. USL techniques analyze the data by using algorithms that rely on numerical methods such as regression and clustering [21]. Unsupervised algorithms are used for data where regular instances outweigh anomalous data points. Otherwise, the algorithm's FPR is higher. Unsupervised learning methods are used for intrusion detection systems, rule-based anomaly detection, and clustering (nearest neighbor) based algorithms. USL methods do not perform well in cases where data do not have instances grouped into clusters for a regular class hence making anomaly categorization difficult. Clustering techniques are dominant USL approaches along with statistical techniques. Information-theoretic techniques such as Kolmogorov complexity, entropy, and relative entropy may be performed using USL methods to detect anomalies.

USL methods are not suitable for detecting variation of anomalies in bulk and for generic spectral techniques for higher-dimensional datasets [1].

Artificial Neural Networks (ANNs) [22] are mathematical models developed to mimic information processing capabilities of the human brain. The output of such networks is determined by propagating the input signal through the hidden layers connected to each other by computed weights. An ANN in which the signal propagates in the forward direction to compute an output is called Feedforward Neural Network (FNN). Unsupervised learning techniques have also been widely explored to detect anomalies in input data using FFNs [22]. Most of these methods rely on non-temporal information for analysis, unlike sequential data that contain redundancies and extended temporal features. The performance of LSTM, although a widely popular supervised learning model, has also been evaluated for unsupervised learning.

The combinatorial algorithm for $k$-means clustering and ID3 (decision tree) methods has been proposed [23] to classify anomalies of Address Resolution Protocol (ARP) traffic. It has shown a higher F-Score than each of these algorithms when implemented standalone. Receiver Operating Characteristic (ROC) curve is generated by plotting TPR against FPR at variable threshold settings. The unsupervised learning algorithm (isolation forest) outperforms the supervised learning algorithm (random forest) for bankruptcy prediction by giving a higher value for area under the ROC curve (AUC) [24]. A high value of AUC is an indicator of high precision and sensitivity (recall) [24]. IF, when implemented on synthetic and real datasets, has demonstrated higher accuracy and outperformed other unsupervised learning methods [25]. When tested on larger datasets for scalability, IF demonstrated acceptable memory usage for datasets with up to one million instances [23]. IF, when employed using shuttle and satellite datasets, outperformed other unsupervised and supervised learning algorithms [20]. Although computationally inexpensive, OCSVM did not perform very well compared to IF when tested on larger datasets [23].

# Chapter 3

# Description of Datasets

Classification and learning process in machine learning techniques depends on the quality of a dataset. Reliable and high-quality datasets are crucial for obtaining dependable results. Production of high-quality data is costly and involves an extensive human effort to clean and accurately label the data. The risk of misclassification also increases if the data were not reliable, which might cause severe consequences depending on the application. In the case of BGP datasets, the cost of misclassifying an anomaly is higher than classifying a regular point as an anomaly.

## 3.1 Data Collections

Features such as volume and AS-path are extracted from data collected in 1 min intervals over the duration of a five-day period for an anomalous Internet event (blackout). To minimize storage and computational requirements, the analysis is limited to a five-day period only: data collected two days before and two days after the event and hours during which the anomaly event lasted. Details of remote route collectors (rrcs) that obtained data using Routing Information Service (RIS), AS Peer information, and date of the Moscow blackout event are shown in Table 3.1 [3]. The routing data used for the detection of anomalies in this Thesis are obtained from the Route Views [53] project at the University of Oregon, USA and the RIS project initiated by the Réseaux IP Européens (RIPE) Network Coordination Centre (NCC) [52]. Both organizations collect and store data chronologically and have made BGP update messages publicly available to the research community. In this experiment, Moscow blackout BGP data collected by RIPE (rrc04, rrc05) and Route Views (route-views2) collection sites are analyzed for classifying anomalies. The collectors rrc04 and rrc05 are located in Vienna and Geneva, respectively. Details of the Moscow blackout are discussed in the following section.

Table 3.1: Moscow power outage dataset [3].

| **Event** | **Date** | **RIS** | **Peers** |
|---|---|---|---|
| Moscow power blackout | May 25, 2005 | rrc05 | AS 1853, AS 12793, AS 13237 |

## 3.2 Moscow Blackout Dataset

On May 24, 2005 at 20:57 (MSK), the Chagino substation (part of the Moscow Energy Ring) experienced a transformer failure that led to complete shutdown of the substation and failure in power transmission to the attached high voltage transmission lines [26], [27] halting numerous trains and elevators and resulting in a complete blackout in the southeast part of the city [28]. The Moscow regional dispatch administration (RDA) technicians executed redistribution of the power grid capacity to prevent overloads due to the blackout. On May 25, 2005, high voltage lines suffered from outages during the morning peak load, which led to the cascaded failure of the transmission lines in the Moscow energy system. The cascading failure effect of the blackout was stopped within 2hr and 20 min by establishing the emergency response center by Unified Energy System (UES) of Russia at 11:00 (MSK) on May 25, 2005 [26]. With preventative measures, the power was restored to the socially relevant and essential infrastructures of the city. Power was fully restored by 16:00 (MSK) on May 26, 2005 [26]. The event of the power outage at the Moscow Internet Exchange (MSK-IX) [8], the main traffic exchange point through which 80% of the Russian Internet flows on an average day, took down all Moscow websites. As an alternative route, the traffic was routed through foreign channels. However, due to capacity limitations, even foreign channels were affected [28]. Instabilities in the network traffic routing were observed due to the loss of connectivity of some ISPs peering at the MSK-IX. Overflow of announcement messages incoming from AS 12793 at the RIS remote routing collector in Vienna (rrc05) showed the impact of the power outage [29].

Update messages provided by RIPE and Route Views collection sites have been used to extract Volume and AS-path features [54]. Features such as NRLI prefixes and the number of announcements show a spike in both datasets. Announcement messages from AS peer 12793 indicating routing failures on May 25, 2005, are shown in Fig. 3.1. The data segregation details for construction and verification of the Moscow blackout data classification model are shown in Table 3.2. The peak power outage that occurred during the peak morning load lasted approximately 4 hrs [30]. Therefore, anomalies between 7:00 and 10:59 (MSK) [31] have been considered in the analysis of the datasets.

### 3.2.1 Processing of Collected Data

Several BGP Cisco routers and Zebra servers have been set up at diverse geographic locations by the Route Views project. Every two hours, BGP routing tables are collected. RIPE NCC has been collecting data since 2001 via the RIS project. After July 2013, the interval

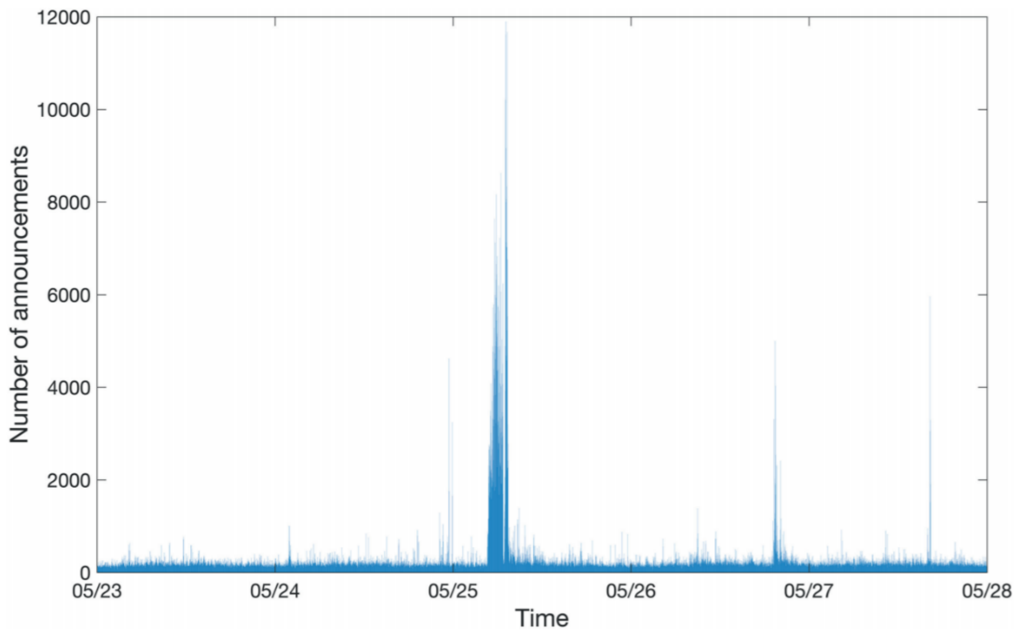


Figure 3.1: Announcement messages from May 23 to May 28, 2005 at peer AS 12793 in Moscow [3].

Table 3.2: Moscow blackout training and test datasets [31].

| | **Regular** (min) | **Anomaly** (min) | **Regular** (training) | **Anomaly** (training) | **Regular** (test) | **Anomaly** (test) | **Collection date** **Start** | **End** |
|---|---|---|---|---|---|---|---|---|
| RIPE | 6,960 | 240 | 3,120 | 180 | 3,840 | 60 | 23.05.2005 00:00:00 | 27.05.2005 23:59:59 |
| Route Views | 6,865 | 130 | 3,075 | 85 | 3,790 | 45 | 23.05.2005 00:00:00 | 27.05.2005 23:59:59 |

between consecutively exported data reports was reduced from every 15 mins to every 5 mins. A multi-threaded routing toolkit (MRT) was introduced by the Internet Engineering Task Force (IETF) for the export of protocol messages, state changes, and content of the routing information base (RIB). Collected BGP update messages are stored in MRT binary format. MRT files are converted to ASCII format using the *zebra-dump-parser* tool written in Perl [3]. The following RIS rrcs collect BGP update messages: rrc01 (LINX, London), rrc03 (AMSIX, Amsterdam), rrc04 (CIXP, Geneva), and rrc05 (VIX, Vienna). Data collected during the period of Internet anomaly by RIPE (rrc04, rrc05) and Route Views projects are used for this experiment. Multiple datasets may be concatenated in order to vary the size of the training datasets.

## 3.3 Extraction of Features

In the classification process, we first need to select and extract features. A tool written in C# is used to parse ASCII files to extract statistics of desired features [3]. Depending on the values of each feature, they may be categorized as continuous, categorical, or binary. Binary features have two numeric values (0, 1) wherein categorical features may have non-numeric values. Data used in this project have AS-path and volume features. AS-path features are derived from the AS-path field. Other extracted features besides AS-path features are known as volume features. A subset of extracted features was used to reduce the dimensionality of the dataset matrix. Table 3.3 lists 37 features used in this study, categorized as a volume or an AS-path feature. NLRI prefixes (volume feature) either have announcement or withdrawal BGP update messages. NLRI prefixes with identical BGP attributes are encapsulated to be sent in one BGP packet. A BGP packet may contain more than one announced or withdrawn NLRI prefix. AS-path may be calculated using the known average and maximum number of AS-peers. BGP update packets with identical NLRI prefixes and AS-paths are labeled as duplicate announcements while BGP announcements with different AS-paths whose NLRI prefixes have already been announced are known as implicit withdrawals. An incomplete BGP update message indicates that the source of the announced NLRI prefixes is unknown [3].

Table 3.3: BGP dataset: Extracted features and their categorization [3].

| **Feature type** | **Name** |
|---|---|
| Volume | Packet size (B), Number of incomplete packets, Number of Exterior Gateway Protocol (EGP) packets, Number of Interior Gateway Protocol (IGP) packets, Inter-arrival time, Number of announcements, Number of withdrawals, Number of announced NLRI prefixes, Number of withdrawn NLRI prefixes, Number of duplicate announcements, Number of duplicate withdrawals, Number of implicit withdrawals |
| AS-Path | Maximum AS-path length = n: where n = (7, ..., 15), Maximum edit distance = n: where n = (7, ..., 17), Maximum edit distance, Average edit distance, Average AS-path length, Maximum AS-path length, Average unique AS-path length |

Performance of BGP protocol heavily relies on the trust relationship and the assumption that the exchanged information is accurate and reliable, thus making the protocol prone to anomalies. For example, an intruder may manipulate the routing information and cause the traffic to be forwarded to an unknown destination and hackers may also pollute the information resulting in the drop/loss of packets. Since an AS is a combination of networks, false information may quickly and widely propagate leading to a massive spread of inaccurate information (damage and losses) and resulting in a surge of BGP announcements. The flooding of BGP announcements impacts volume features. It has been observed that the most influential features are volume features and, hence, are more important for identifying anomalies. Edit distance is defined as the minimum number of deletions, insertions, or substitutions needed to match two AS-path attributes. During the anomalies, the variance of edit distances and the AS-paths are large. Hence, AS-path features may be the distribution outliers during an anomaly attack. For example, 58% of AS-path features have values larger than the distribution mean. A larger AS-path length indicates routing delays while a shorter AS-path length indicates hijack or a component failure [3].

Chapter 4

# Machine Learning: Algorithms and Performance Metrics

Securing a network from intruders attack or maintaining network connectivity is a challenging task that requires robust tools. Machine learning algorithms, due to their high computational capabilities, are used to detect anomalies. Several techniques have been devised to mimic human learning capabilities. Supervised machine learning approaches are popularly applied to solve classification problems. Supervised learning techniques use labeled data to build models. In this Chapter, we describe two supervised machine learning algorithms that may be employed for detecting BGP anomalies and discuss methods for their performance evaluation.

## 4.1 Long Short-Term Memory

The development of RNNs, a class of artificial neural networks (ANNs) [35], was inspired by the circular connections of neurons in a human brain. ANNs were initially designed as mathematical models to mimic the ability of the human brain to process information. They consist of small processing units interconnected via weighted connections. Activations in the network are provided by either all or selective inputs that spread throughout the network along the weighted connections. ANNs that propagate information in the forward direction are known as feedforward neural networks (FNNs). Unlike FNNs, RNNs relate inputs to the outputs by using circular connections in a network. RNNs are capable of mapping all the given past inputs to outputs and have proved to perform better for learning sequential data. They may provide sequence to sequence mapping for any inputs given that have a sufficient number of hidden units required for approximation [35]. To determine output, RNNs rely on the memory of previous input in the internal state of the network. The architecture of RNNs has its limitation, the influence of input on the hidden units impacts the output of the network. The phenomenon of output either exponentially decaying or increasing while propagating circularly in the network is known as a vanishing gradient problem. Long Short-

Term Memory (LSTM) algorithm, introduced by Hochreiter and Schmidhuber [35] in 1997, has been widely used to handle the vanishing gradient problem illustrated in Fig. 4.1. Darker shade represents the highest sensitivity of the node at the input in a given network. As time progresses, the sensitivity of the nodes vanishes due to being overwritten by newer inputs.

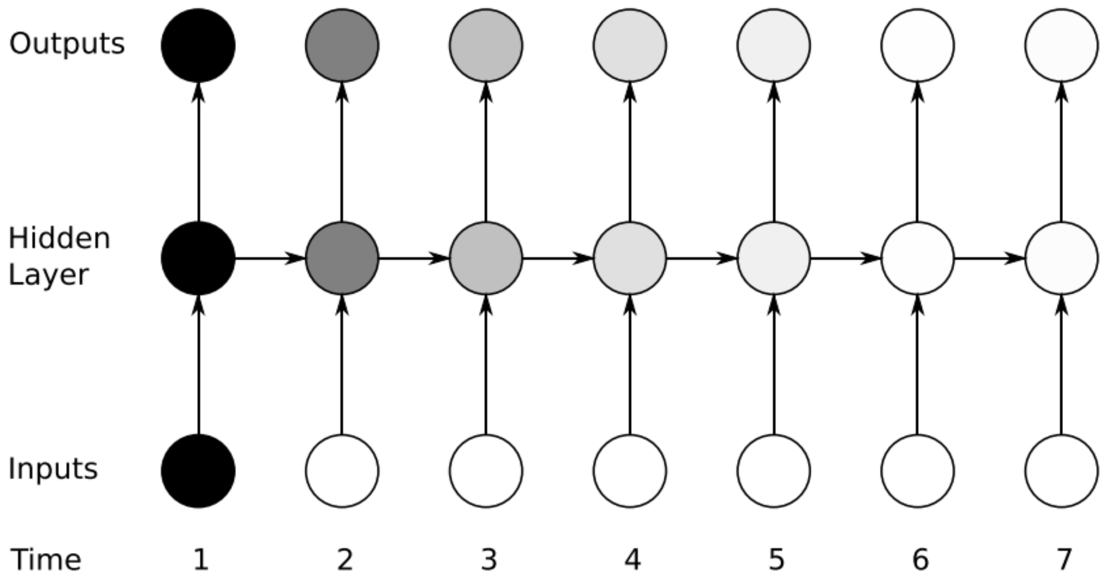


Figure 4.1: Vanishing gradient problem for RNNs over time. Sensitivity of nodes for an unfolded RNN is illustrated with shading [35].

In the 1990s, training approaches such as simulated annealing and discrete error propagation were used to handle the vanishing gradient problems. However, they led to the introduction of delays, time constants, and hierarchical sequence compression. LSTM architecture consists of subnets called memory blocks that connect recurrently to each other. Single LSTM memory block is composed of either one or more connected memory cells and three multiplying units for the input, output, and forget gates. The LSTM block architecture with one cell is illustrated in Fig. 4.2. LSTM differs from other RNNs because a memory block replaces the hidden RNN state. LSTM handles gradient vanishing problem by storing information over extended periods in memory cells. Inflow of new information is controlled by monitoring the input gate. If the activation value of the input gate is near zero (i.e input gate is closed), newer information cannot be written in the cell. LSTM has proved to be one of the best algorithms for precisely evaluating sequential temporal data [35].

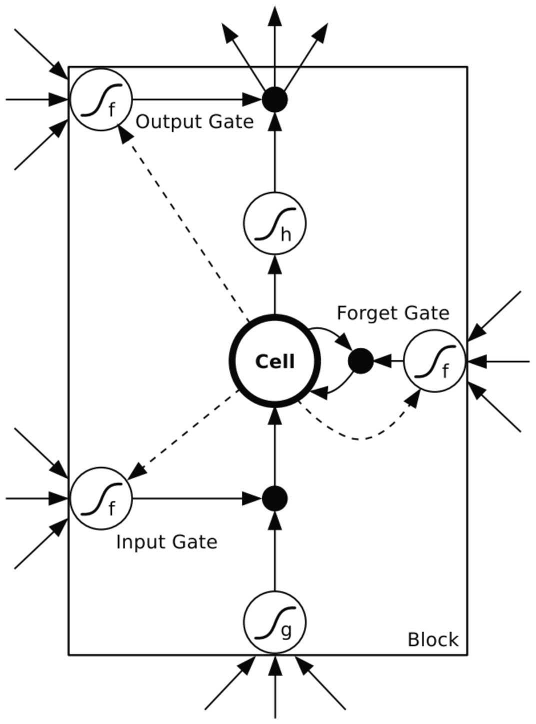


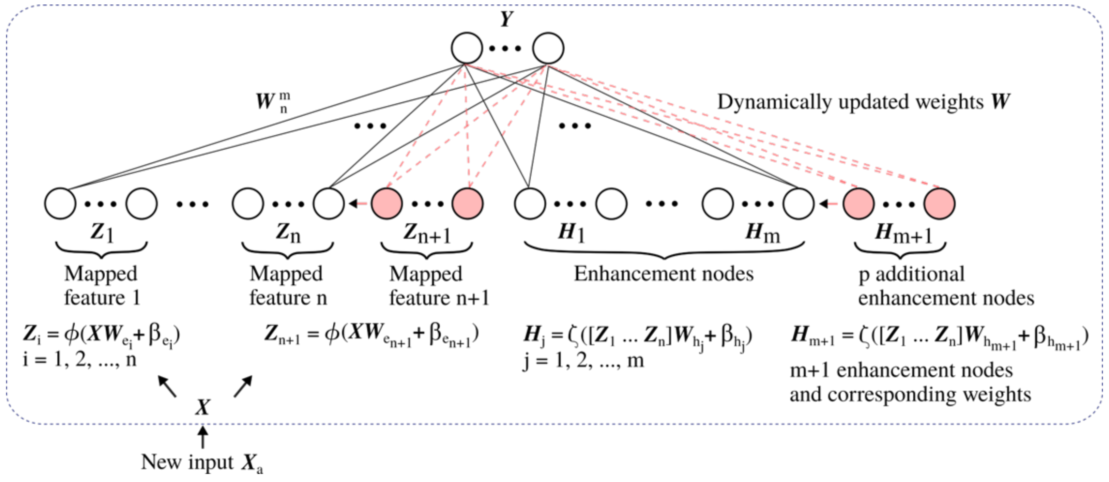
Figure 4.2: LSTM memory block with one cell and three multiplying units for input, output, and forget gates [35].

## 4.2 Broad Learning System

Algorithms with deep architectures have many hyperparameters and complex connections that require a time-consuming training process [36]. A large number of connecting parameters in layers make the training processing time consuming. Besides the increase in the size of datasets, data dimension has also increased, and handling higher dimensional data has become a challenge for neural networks. The rate at which the dimension and volume of data are increasing is inevitable. Hence, feature extraction and dimension reduction have been suggested [37]. BLS is an alternative to deep learning algorithms that suffer from a complete retraining process if the built model is insufficient. The BLS architecture is based on Random Vector Functional-Link Neural Network (RVFLNN) and is designed to overcome the shortcomings (longer training time, retraining) of deep learning [9]. We use BLS to perform anomaly detection [9], [14]. The single layer structure of BLS contributes to its superior performance speed [9], [36]. BLS architecture is quite flexible. By exploiting its properties, a number of algorithms have been developed: incremental learning BLS, radial basis function (RBF) network, BLS (RBF-BLS), BLS with cascades of mapped features (CFBLS), and BLS with cascades of enhancement nodes (CEBLS) [7]. Incremental learning BLS architecture shown in Fig. 4.3 is composed of the input, mapped features, enhancement nodes, and output [36].

The advantage of using BLS over other algorithms is its strong feature extraction and highly efficient computational capabilities [14]. Compared to other machine learning algorithms, BLS offers a shorter training time because it does not have hidden layers [7]. One of the BLS architectures used for detecting network traffic anomalies is illustrated in Fig. 4.3. Mapped features are first extracted from the input data by using a linear function. Then, mapped features are used to generate the enhancement nodes by using either a linear or a nonlinear function [6], [36].

BLS architecture may be dynamically and incrementally updated and the model may be expanded by adding mapped features and enhancement nodes [37] hence making the training process of the algorithm efficient. The first step is to create mapped features from input data. Enhancement nodes are then generated from mapped features. Lastly, the concatenation of mapped features and enhancement nodes is used to obtain the output. In this experiment, the input data X multiplied with randomly generated weights. The value of the bias parameters is set to zero. The enhancement nodes are generated by using a hyperbolic tangent function of the concatenated mapped features. The output weights are generated by calculating pseudoinverse of the concatenation of mapped features and enhancement nodes [37].



Y
$W_n^m$
Dynamically updated weights W
$Z_1$
Mapped feature 1
$Z_n$
Mapped feature n
$Z_{n+1}$
Mapped feature n+1
$H_1$
$H_m$
Enhancement nodes
$H_{m+1}$
p additional enhancement nodes
$Z_i = \phi(XW_{e_i} + \beta_{e_i})$ i = 1, 2, ..., n
$Z_{n+1} = \phi(XW_{e_{n+1}} + \beta_{e_{n+1}})$
$H_j = \zeta([Z_1 \ldots Z_n]W_{h_j} + \beta_{h_j})$ j = 1, 2, ..., m
$H_{m+1} = \zeta([Z_1 \ldots Z_n]W_{h_{m+1}} + \beta_{h_{m+1}})$
m+1 enhancement nodes and corresponding weights
X
New input $X_a$

Figure 4.3: BLS architecture with increments of new input data, mapped features, and enhancement nodes [13].

## 4.3 Classification of Data Points

Machine learning methods use a feature matrix whose rows indicate data points while columns indicate features. Factors such as noise and redundancy may increase the test data points misclassification by machine learning algorithms that otherwise may have acceptable performance for anomaly classification. Using a sufficient number of relevant features may enable machine learning models to build a generalized model to classify data with a lower error rate. Therefore, feature selection algorithms heavily impact the classification results. Data need to be pre-processed by selecting important features before inputting them to machine learning algorithms for classification. For an effective classification, the training data set should be an accurate generalized representation of a problem. The combination of features is also critical for accurate classification [5]. Two scatter plots with Feature 9 (volume) vs. Feature 1 (volume) vs. Feature 6 (AS-path) [5] are shown in Fig. 4.4. The scatter plot shown on the left illustrates a better correlation of data than the scatter plot on the right.

Elements in each dataset may be classified into several classes. A class is defined as an element in a classification domain while classifier labels categorize data. In this study, we employ the supervised learning algorithm BLS for binary classification. Two labels (regular or anomaly) are used for classification.

The developed classifier model is evaluated using a test dataset. Metrics such as confusion matrix, precision, sensitivity, accuracy, and F-Score are calculated to measure the performance of the model. The cost of misclassification of data varies depending on the application domain.

Datasets used for training may be identified as balanced and unbalanced. An unbalanced dataset contains dominant samples of one class while under representing samples of the other class. Unbalanced datasets are known for misclassifying the minority class due to insufficient training samples to train the model, which results in a low accuracy [5]. The dataset used in this experiment is unbalanced: it has a higher number of regular instances than anomalies. To obtain higher accuracy levels, various approaches such as assigning weights to individual classes, or learning from one class (recognition-based) or two classes (discrimination-based) may be used when dealing with unbalanced datasets.

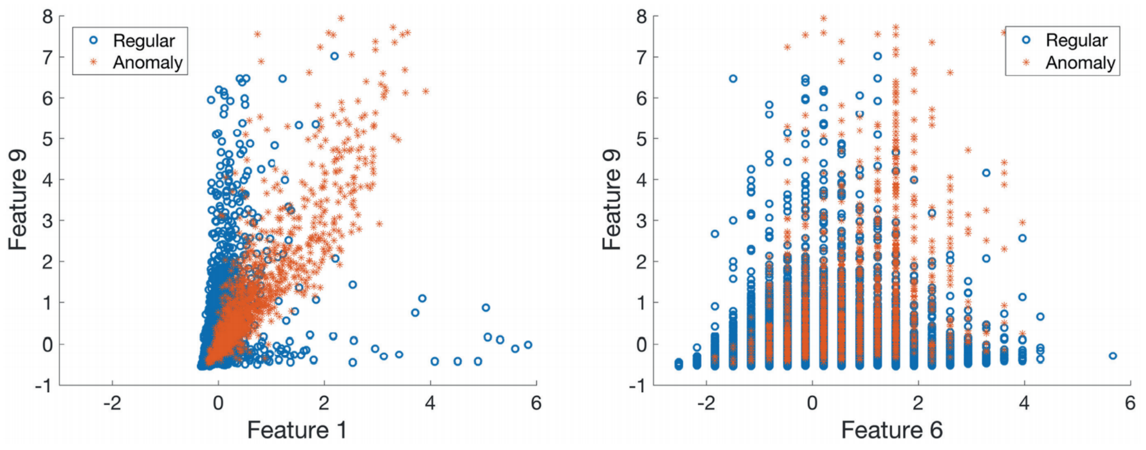


Figure 4.4: Effect of combining features for improving classification results of two classes (anomaly and regular). 2-D scatter plots of Feature 9 vs. Feature 1 (left) and Feature 9 vs. Feature 6 (right) extracted from BCNET traffic [3].

The detection of an anomaly is considered as a positive outcome while detection of a regular data instance is identified as a negative outcome. In order to generate a confusion matrix, we need to calculate: true positive (TP), true negative (TN), false positive (FP), and false negative (FN). TP is an event when an anomalous data instance is classified as an anomaly. If a regular data instance is classified as regular, it is counted as a TN. Similarly, if a regular data instance is misclassified as an anomaly, it is counted as a FP while the misclassification of an anomalous data instance as regular is counted as a FN.

Table 4.1: Confusion matrix.

| | | **Predicted Class** | |
|---|---|---|---|
| | | **Regular (negative)** | **Anomaly (positive)** |
| **Actual Class** | **Regular (negative)** | TN | FP |
| | **Anomaly (positive)** | FN | TP |

## 4.4 Performance Evaluation

Confusion matrix, precision, sensitivity, accuracy, and F-Score are performance metrics used to evaluate the classifier models. Precision is the measure of correctly identified positive cases from all predicted positive cases. Sensitivity (recall) is the measure of correctly identified positive cases from all actual positive cases. Accuracy is the most used parameter for balanced datasets to evaluate the performance of the classifier model as it considers anomalous and regular events equally important [5]. They are defined using the following expressions:

$$\begin{aligned}
\text{precision} &= \frac{TP}{TP+FP} &(4.1)\\
\text{sensitivity (recall)} &= \frac{TP}{TP+FN} &(4.2)\\
\text{accuracy} &= \frac{TP+TN}{TP+TN+FP+FN} &(4.3)\\
\text{F-Score} &= 2\times\frac{\text{precision}\times\text{sensitivity}}{\text{precision}+\text{sensitivity}}. &(4.4)
\end{aligned}$$

In other words, sensitivity measures the accurate classification of predicted anomalies while precision measures the ability of a model to accurately label known anomalies. Sensitivity is used when the cost of false negatives is high while precision is used when the cost of classifying false positives is high. Both precision and sensitivity measure the ability of a model to accurately classify or misclassify events. Accuracy is the measure of accurately classified true positives and true negatives. However, it does not provide a true measure for unbalanced datasets. F-Score is another important metric that is equivalent to the harmonic mean of sensitivity and precision. It reflects accurate detection of anomalies and is a better measure than the accuracy of incorrectly classified cases for unbalanced datasets [51]. TPR is the probability that a true positive will be classified as positive by the classifier during the testing phase. Similarly, TNR is the probability that a true negative will be classified as negative by the classifier. For unbalanced datasets, the mean of TPR and TNR measures the accuracy of the model precisely.

## 4.5 *K*-Fold Cross-Validation

In machine learning, the process of partitioning a single dataset S into $k$ subsets of approximately the same size, called folds, for generating pairs of validation and training sets to build a model classifier is known as cross-validation. From the generated $k$ subsets, each subset is labeled $S_i$, $i = 1, \ldots, k$. $S_{ith}$ dataset is used as the validating data and the remaining subsets are used for training. A learning algorithm is then applied to the dataset $k$ times [32]. The main aim is to achieve a cross-over between training and validating datasets so that data points are validated against each other during the successive cross-validation rounds. Cross-validation may also be defined as a statistical method for generating training and validation datasets to evaluate and compare the performance of the learning algorithms. The $k$-fold is the most general form of cross-validation. Other forms have been derived as variations of the $k$-fold method [33]. A case of $k$-fold cross-validation for $k = 4$ is illustrated in Fig. 4.5: blue color represents the training data subset and light orange color represents the validation data subset. In the initialization step, the dataset is divided into four subsets of approximately equal sizes. The first subset is reserved to validate the model while the remaining three subsets are used to train the model in the first iteration. In the second iteration, the second subset is reserved to validate the model while the first, third, and fourth subsets are used to train the model. Similarly, in the third iteration, the third subset is reserved for validating the model while the first, second, and fourth subsets are used to train the model.

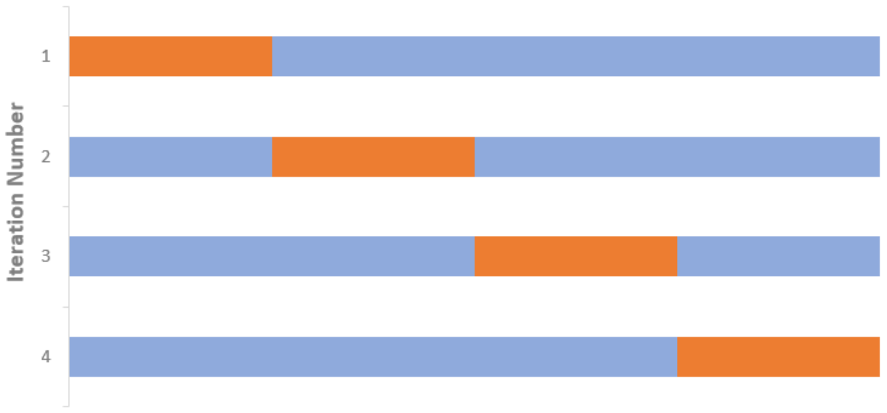


Figure 4.5: $K$-fold cross-validation ($k = 4$) [47].

The $k$-fold validation approach cannot be applied to time series data due to the time dependence of data samples [47]. The $k$-fold validation assumes that the input data instances are independent and equally distributed. Hence, when applied on time series data, the results will have a correlation between training and validation samples that should be avoided. A different approach, called Time Series Split (TSS), should be applied for cross-validation of time series data. TSS approach is a variation of $k$-fold validation that uses the first $k$ folds as training sets and the remaining $(k+1)^{th}$ dataset for validation [47]. An example of $k = 4$ is shown in Fig: 4.6 for TSS where five folds have been generated from the initial dataset in the initialization step. In the first iteration, the first subset is used to train the model while the second fold is used to validate the model. In the second iteration, the first two folds are concatenated to form the training dataset and the third fold is used for validation. A similar approach of concatenating folds to form the training datasets is followed until the fourth (final) iteration. Such an incremental procedure of concatenating the data subsets to train the model is implemented to preserve the sequence of the time series data. Classification results are calculated for each iteration. Accuracy and F-Score are compared to measure the performance of the learning algorithm in each iteration. Finally, the predicted results of these iterations are compared using statistical measures such as average or the hypothesis test to obtain the best performing combination. The most used value for data mining and machine learning applications is $k = 10$ [33].

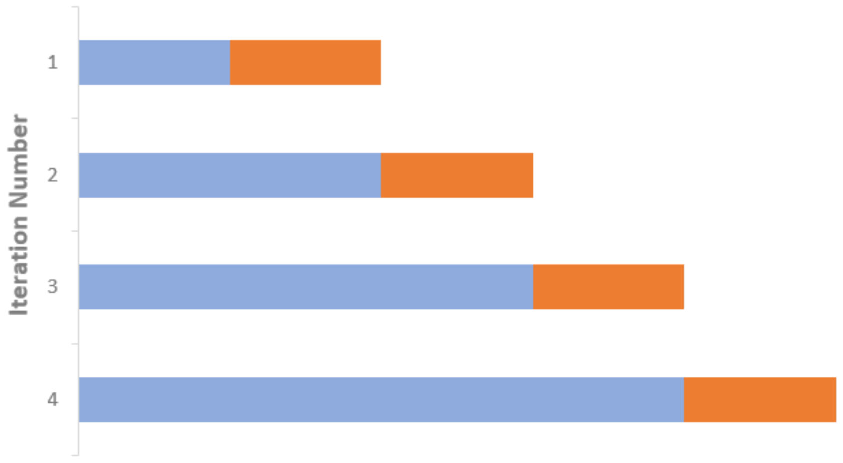


Figure 4.6: Time Series Split (TSS) cross-validation ($k = 4$) [47].

Statistical and data mining methods are used to generate regression or classifier models from the given data. Such methods fail to evaluate the predictive abilities of models for unseen data accurately as their performance is tested using only the given data. Cross-validation helps eliminate the limitations and helps measure the generic performance of the model or an algorithm. Cross-validation is implemented as a standard method for selecting model parameters and estimating the performance of an algorithm and its generalizability [33]. Data stratification is a crucial step while forming folds for cross-validation. Stratification is the process of generating subsets (folds) in a dataset so that an individual fold is a fair representation of the dataset as a whole. For example: if a dataset used for binary classification contains x% of class one classifiers, then each generated fold should also contain x% of class one classifiers.

One of the main objectives of the cross-validation process for a given dataset is to validate the performance (accuracy) of a single algorithm for a given dataset. In the case of an algorithm A and N data points, the cross-validated accuracy of the algorithm measures predicted results accurately for unfamiliar data by a trained model generated by A using all N data points for training. If more than one algorithm is considered, cross-validation may be used to identify the best performing algorithm for a given dataset. To achieve these goals, various validation approaches have been proposed: resubstitution, hold-out, and cross-validation ($k$-fold, leave-one-out, repeated $k$-fold). Stratified 10-fold cross-validation, due to its least biased estimated accuracy, has been proven by Kohavi [32] as the best procedure for model selection.

While evaluating the performance of a model or comparing two different algorithms, an ideal experiment setup requires sufficiently independent performance metrics for each run. Because the selection of training and test datasets influences the performance of a model, overlap of training and test data should be avoided to eliminate overestimation. If a fold is used for testing more than once, the statistical comparison will not be valid as the results of those two runs will be dependent. Learning algorithms perform with higher accuracy when tested on familiar data. Hence, for evaluating the generic accuracy of an algorithm, unseen data should be supplied to the system. Complex models have numerous hidden layers, nodes, and tuning parameters. Models based on better-performing algorithms have proved to have high accuracy when trained with a larger dataset. The accuracy of the algorithms may be improved by using diverse data for training. Training and test datasets impact the performance metric indirectly because training data help generate the model while the test dataset is used to measure the model performance such as accuracy and F-Score. The training and test datasets are created by selecting the percentage of anomalies in a dataset. The most commonly chosen percentage split is 80% and 20% for the creation of training and test datasets, respectively. However, an approach of 60% and 40% percentage split has shown better performance results in some studies [7]. Overlapping of training subsets might be tolerated in machine learning methods while keeping the test dataset separate. In $k$-fold cross-validation, the subsets generated from the training dataset are used to train and validate the model. Selecting the value of $k$ helps control the size of the training data subsets and tune performance estimates. A large value of $k$ implies the smaller size of the validation subsets, less precise classification results, and higher overlap among training subsets. Choosing $k = 10$ is a reasonable estimate for obtaining generalized model performance metrics [33]. Cross-validation helps utilize the entire training data for obtaining the performance metric for most supervised learning applications. It may also be used for model selection or tuning of model parameters. When comparing two algorithms, an accurate two-sample hypothesis should be evaluated rather than comparing the average performance [33]. Comparison of several validation approaches is listed in Table 4.2.

Table 4.2: Comparison of validation methods [33].

| **Validation method** | **Pros** | **Cons** |
|---|---|---|
| Resubstitution validation | Simple | Over-fitting |
| Hold-out validation | Independent training and test | Reduced data for training and testing; large variance |
| $k$-fold cross-validation | Accurate performance estimation | Small samples of performance estimation; overlapped training data; error for comparison; underestimated performance variance |
| Leave-one-out cross-validation | Unbiased performance estimation | Very large variance |
| Repeated $k$-fold cross-validation | Large number of performance estimates | Overlapped training and test data between each round; underestimated performance variance or overestimated; |

# Chapter 5

# Experimental Results

We use Python as a coding language for anomaly detection. The developed Python code includes modules for BLS and its extensions [16]. The performance of models is measured using confusion matrix, accuracy, F-Score, and training time. The code is publicly available for download (folder: "BLS_SFU_CNL_V1.0.1.zip") [16]. The environment is configured by downloading desired libraries listed in the README.txt file and running either "BLS_demo_for_lower_memory.py" or "BLS_incremental_demo_lower_memory.py" file from the shell using Python3.

## 5.1 Implementation of the Broad Learning System

BLS was chosen to evaluate Moscow blackout data because of its single layer architecture, ability to expand its network (avoiding the retraining process), and shorter training time to generate a classifier. A blackout phenomenon is caused by an electrical components failure, unlike intrusion attacks that occur due to the presence of an intruder. During a blackout, the traffic may reroute via other ASes. Hence, it is challenging to identify the window of anomalies during a blackout. During the Moscow blackout, BGP link failures were experienced when impacted ASes redirected the traffic resulting in a narrower window of Internet anomaly [31]. We analyzed data provided by RIPE collection sites (rrc04 and rrc05) and Route Views. Pre-processed labeled data from RIPE and Route Views repositories were used to classify anomalies employing BLS [7]. The data have been labeled based on the time interval when the anomalous event occurred. The following percentage of anomalous data points were chosen for training and test datasets, respectively: 75% and 25% (RIPE) and 65% and 35% (Route Views) [31]. The experiments were conducted on a CPU, Dell Alienware Aurora with 32 GB memory, and Intel Core i7 7700K processor. Results were obtained using Python 3.6 running on Ubuntu 16.04.

## 5.2 Performance Evaluation

The 10-fold TSS cross-validation, a time consuming process, was performed for several values of BLS parameters. The best performance results achieved by BLS are shown in Table 5.1. After testing various combinations of numbers for mapped features, enhancement nodes, and groups of mapped features, the parameter values used to generate the best performing models are listed in Table 5.2. The data generated by RIPE (rrc05) have led to higher performance than the other datasets. A lower number of anomalies present in the dataset might have influenced the low F-Score achieved by the generated models. The best F-Score achieved with RIPE data collected by rrc05 is 24.034%. Similar F-Score and accuracy are achieved by BLS for both rrc04 and Route Views datasets. The very low precision and sensitivity indicate that other machine learning algorithms such as RNNs (LSTM and GRU) may be better suited for detecting Moscow blackout anomalies [31].

Table 5.1: Comparison of performance results for RIPE and Route Views Moscow blackout data.

| | **Precision (%)** | **Sensitivity (Recall) (%)** | **F-Score (%)** | **Accuracy (%)** |
|---|---|---|---|---|
| RIPE (rcc04) | 14.51 | 26.11 | 18.65 | 97.54 |
| RIPE (rr05) | 19.58 | 31.11 | 24.03 | 90.64 |
| Route Views (route-views2) | 17.95 | 19.58 | 18.73 | 95.79 |

Table 5.2: Number of feature, enhancement nodes, and groups of mapped features used to achieve the best F-Score.

| | **Mapped features** | **Groups of mapped features** | **Enhancement nodes** |
|---|---|---|---|
| RIPE (rrc04) | 300 | 30 | 600 |
| RIPE (rrc05) | 400 | 30 | 300 |
| Route Views (route-views2) | 300 | 30 | 600 |

A regularization parameter is used to avoid overfitting a model. It helps to reduce errors in the output weights and improves the computation of weights. Experiments with Moscow blackout data were also conducted by using various values of the regularization parameter for BLS. The values that achieved the best F-Score for each dataset are summarized in Table 5.3. The value of the regularization parameter for BLS is $2^n$, where $n$ is a negative value and may be modified. Results are illustrated in Fig. 5.1. The x-axis of the figure shows values of $n$ and the y-axis shows the F-Score for the three datasets. The value of F-Score for rrc04 data (blue) remains constant between $2^{-15}$ and $2^{-20}$ while a drop is observed at $2^{-10}$. The highest peaks occur at: $2^{-21}$ rrc04 data (blue), $2^{-16}$ rrc05 data (orange) and $2^{-13}$ route-views2 data (grey). Hence, a nonlinear relationship exists between the regularization parameter and F-Score for Moscow blackout data.

The high number of FPs and FNs compared to TPs for BLS models as listed in the Table 5.4 might have influenced the a low value of F-Scores.

Table 5.3: Regularization parameter values to achieve the best F-Score.

| | **Regularization parameter** |
|---|---|
| RIPE (rrc04) | $2^{-21}$ |
| RIPE (rrc05) | $2^{-16}$ |
| Route Views (route-views2) | $2^{-13}$ |

Table 5.4: Confusion matrix calculated for RIPE and Route Views data collected during the Moscow blackout in May 2005.

| **Dataset** | **True Positive (TP)** | **True Negative (TN)** | **False Positive (FP)** | **False Negative (FN)** |
|---|---|---|---|---|
| RIPE (rcc04) | 47 | 3,323 | 277 | 133 |
| RIPE (rcc05) | 56 | 3,370 | 230 | 124 |
| Route Views (route-views2) | 28 | 3,424 | 128 | 115 |

## 5.3 Improving Performance Metrics

A spatial separation of data features, a vital factor for higher classification accuracy, was better observed in RIPE datasets[31]. The RIPE dataset features help to better classify anomalies than Route Views due to a more complete dataset [31]. Misclassification of data might occur due to redundancies or noise in the data used to train the classifier. Selecting a sufficient number of essential features in the training data helps derive a generalized classifier model with fewer errors. Performance metrics of the classifier may be improved by pre-selecting relevant features and employing feature selection algorithms. Before applying machine learning algorithms, feature selection algorithms are employed to pre-process the data for classification. In our experiments, 37 BGP features [3] were selected to classify the data. Certain combinations of features provide better spatial separation and, hence, may improve the anomaly classification. Memory usage and computational complexity are dependant on feature selection that helps reduce the dimension of the data. The dimensionality of the data may be reduced by eliminating redundant, irrelevant, and noisy features. Factors such as training time, modeling accuracy, and elimination of redundancies are improved by selecting features, hence reducing misclassification.

Route Views consist of a larger number of ASes than RIPE and, hence, collects a higher number of data instances (Table 3.2). AS-path features (Average AS-path length and Maximum edit distance) and volume feature (Number of implicit withdrawals) graphs plotted for RIPE and Route Views data are shown in Fig. 5.2 for regular and anomaly classes. During the anomaly, the Average AS-path length feature shows no activity in the RIPE dataset as illustrated in Fig. 5.2 (top). Activity is observed for both datasets during the

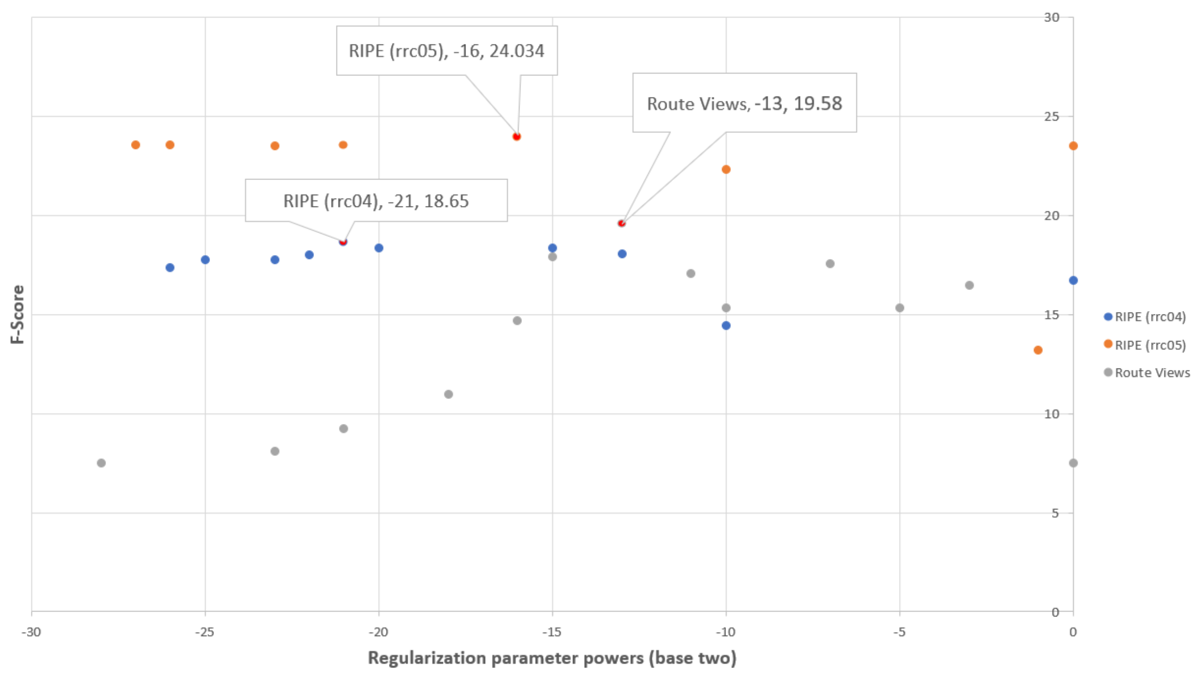


Figure 5.1: Scatter plot of regularization parameter vs. F-Score for Moscow blackout data.

period of anomaly for the Maximum edit distance as shown in Fig. 5.2 (middle). A number of implicit withdrawals shown in Fig. 5.2 (bottom) illustrates activity in the RIPE dataset different from Route Views data, which might be due to the geographical location of the RIPE ASes (Europe). Similarly, the volume features plotted in Fig. 5.3 and Fig. 5.4 illustrate more prominent activity in RIPE than Route Views during the anomaly event. The number of incomplete packets shown in Fig. 5.4 (top) sent during the blackout window is also higher for the RIPE dataset. Hence, the dataset collected by RIPE might be considered more reliable than data collected by Route Views.

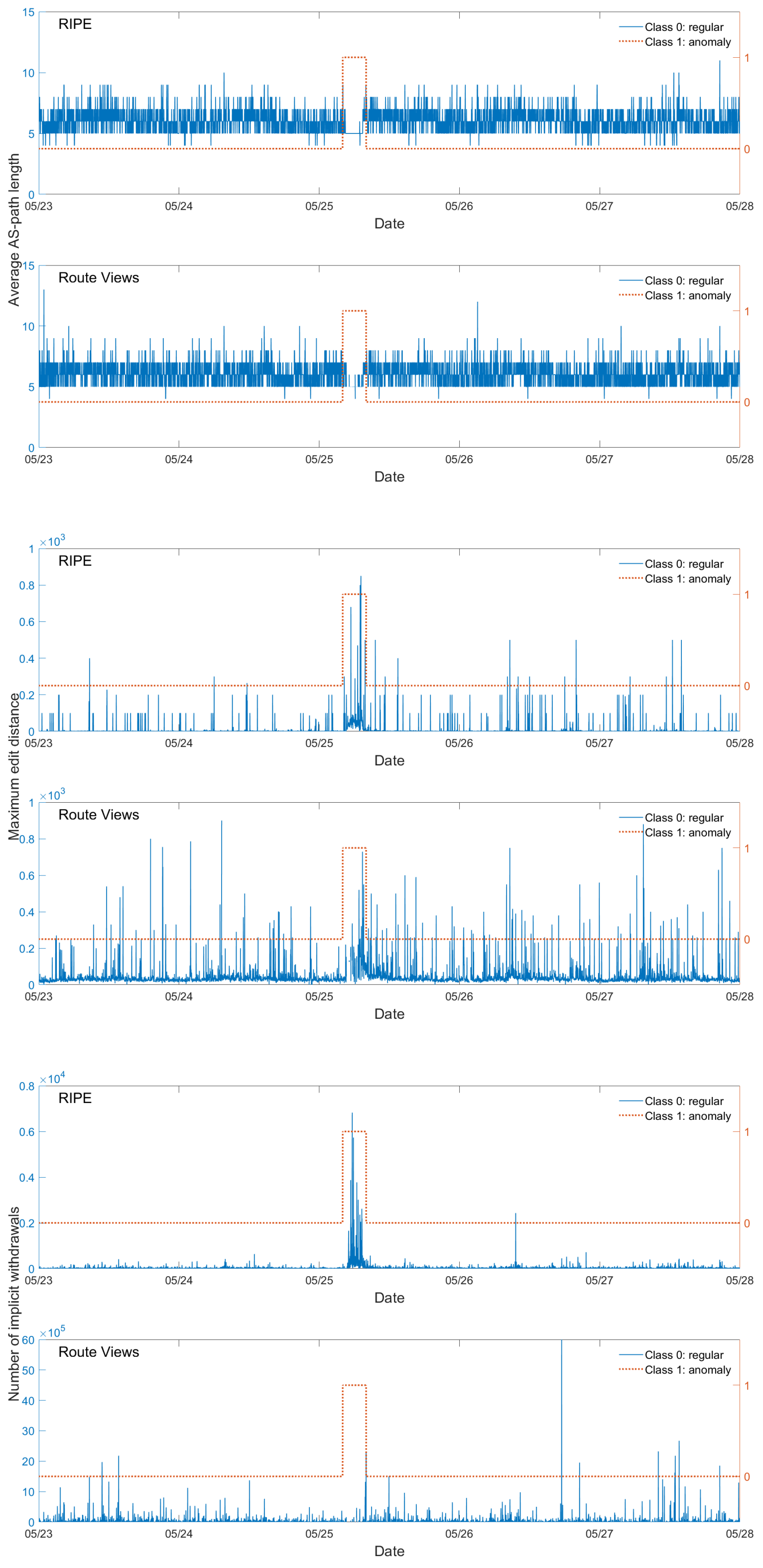


Figure 5.2: Moscow blackout (RIPE and Route Views): Average AS-path length (AS-path feature), Maximum edit distance (AS-path feature), and Number of implicit withdrawals (volume feature) [38].

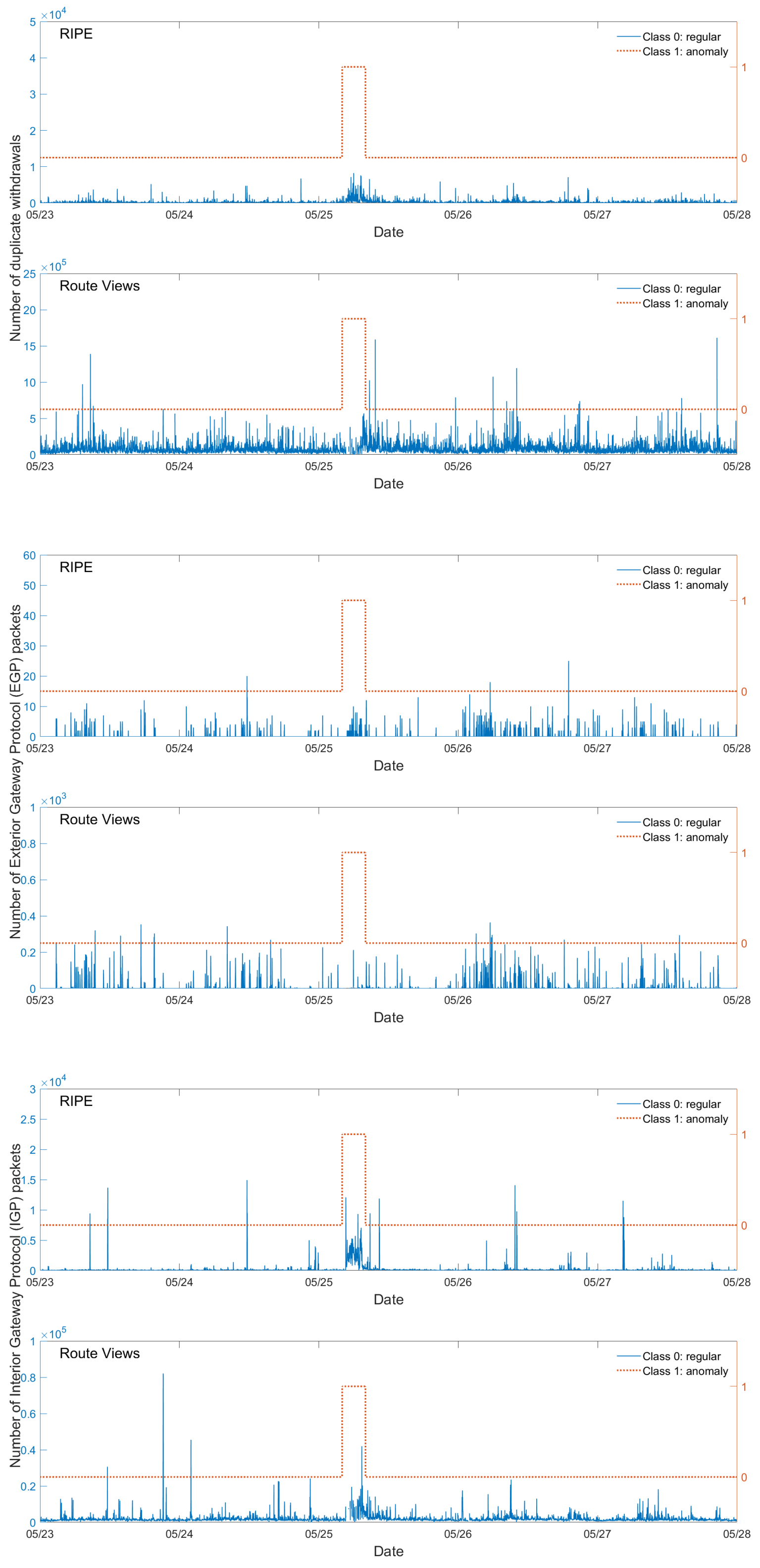


Figure 5.3: Moscow blackout (RIPE and Route Views): Number of EGP (volume feature), and IGP packets (volume feature), Number of duplicate withdrawals (volume feature) [38].

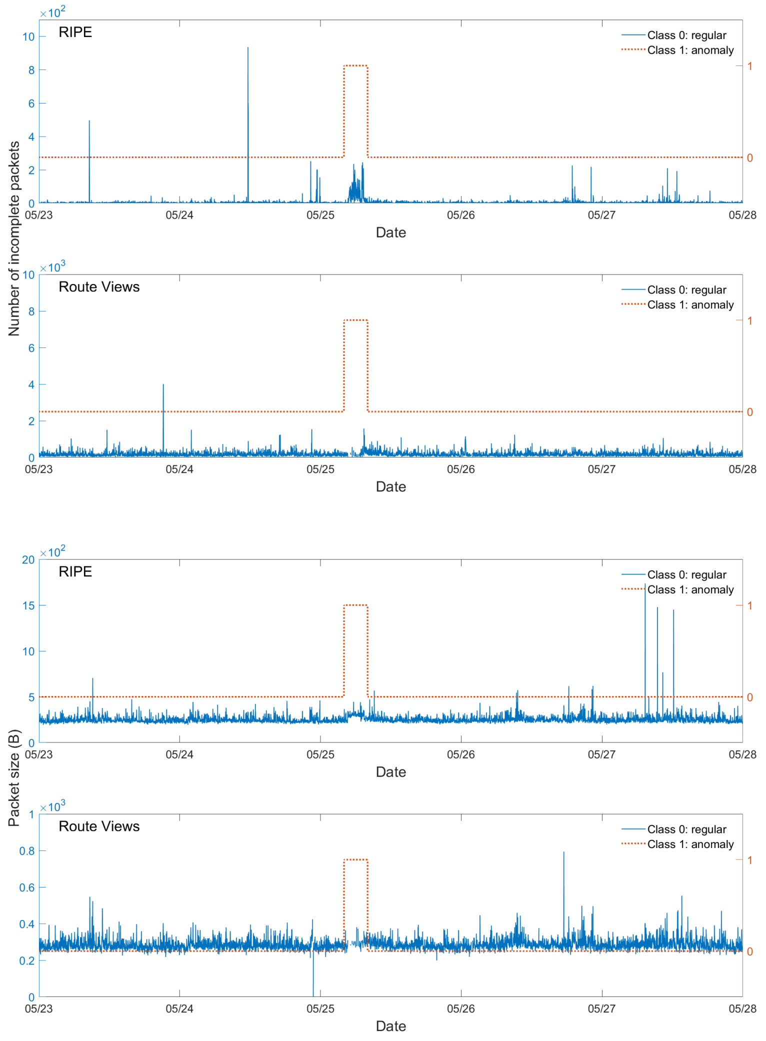


Figure 5.4: Moscow blackout (RIPE and Route Views): Number of incomplete packets (volume feature) and packet size (volume feature) [38].

Chapter 6

# Conclusion

We considered BGP anomalies in communication networks by analyzing Moscow blackout data collected by RIPE (rrc04 and rrc05) and Route Views. We employed BLS with various values of mapped features, enhancement nodes, groups of mapped features, and regularization parameter. A combination of parameters that generated the best performing F-Score was chosen. Although it is a single layered (architecture) and computationally efficient (memory consumption and training time) algorithm, BLS did not perform well for the Moscow blackout data. BLS achieved 24.03% as the best F-Score for the RIPE rrc05 data collector. In the past, BLS has achieved high accuracy and F-Score for detecting worm attacks. Note that the Moscow blackout is a different phenomenon that occurred due to the failure of an electric component rather than being caused by the presence of an intruder or a hacker. Hence, the nature of data is different and may require another approach. Blackout data should be further analyzed to better understand its intrinsic features and to enhance the performance of the classifier, data features may also be selected for better spatial separation. Poor F-Score indicates lower precision and sensitivity of a classifier, factors such as a more precise window selection for labeling the data may improve the performance matrix of the classifier. The anomaly window chosen for labeling data in reported experiments was based on the collected evidence, which might have impacted BLS performance. A more precise window for labeling the anomalous data may be selected to enhance the classifier performance. Another approach of splitting the data with partitions such as 60% (80%) of data for training and 20% (40%) of data for testing may also enhance the performance of the classifier. Furthermore, semi-supervised learning approaches may prove more successful. They are known to perform better for the classification of anomalies due to their learning process.

# Bibliography

[1] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: a survey," *ACM Comput. Surv.*, vol. 41, pp. 15:1–15:58, July 2009.

[2] S. Baek, D. Kwon, S. C. Suh, H. Kim, I. Kim, and J. Kim, "Clustering-based label estimation for network anomaly detection," *Digit. Commun. Netw.*, June 2020.

[3] Q. Ding, Z. Li, S. Haeri, and L. Trajković, "Application of machine learning techniques to detecting anomalies in communication networks: datasets and feature selection algorithms," in *Cyber Threat Intelligence*, M. Conti, A. Dehghantanha and T. Dargahi, Eds., Berlin: Springer, 2018, pp. 47–70.

[4] (2021, Jan.) What is an autonomous system? | What are ASNs?. [Online]. Available: https://www.cloudflare.com/learning/network-layer/what-is-an-autonomous-system/ .

[5] Z. Li, Q. Ding, S. Haeri, and L. Trajković, "Application of machine learning techniques to detecting anomalies in communication networks: classification algorithms," in *Cyber Threat Intelligence*, M. Conti, A. Dehghantanha and T. Dargahi, Eds., Berlin: Springer, 2018, pp. 71–92.

[6] S. B. Wankhede, "Anomaly detection using machine learning techniques," in *Proc. IEEE Int. Conf. Convergence Technol.*, Bombay, India, March 2019, pp. 1–3.

[7] A. L. Gonzalez. Rios, Z. Li, K. Bekshentayeva, and L. Trajković, "Detection of denial of service attacks in communication networks," in *Proc. IEEE Int. Symp. Circuits Syst.*, Seville, Spain, Oct. 2020.

[8] M. Gopal, "Well-posed machine learning problems," in *Applied Machine Learning*, New York, NY, U.S: McGraw-Hill Education, 2019.

[9] Q. Ding, Z. Li, P. Batta, and L. Trajković, "Detecting BGP anomalies using machine learning techniques," in *Proc. IEEE Int. Conf. Syst., Man, Cyber.*, Budapest, Oct. 2016, pp. 3352–3355.

[10] A. Patcha and J.-M. Park, "An overview of anomaly detection techniques: existing solutions and latest technological trends," *Comput. Netw.*, vol. 51, no. 12, pp. 3448–3470, Aug. 2007.

[11] V. Hodge and J. Austin,"A survey of outlier detection methodologies," *Artif. Intell. Rev.*, vol. 22, no. 2, pp. 85–126, Oct. 2004.

[12] A. H. Sung and S. Mukkamala, "Identifying important features for intrusion detection using support vector machines and neural networks," in *Proc. Symp. Appl. Internet*, Orlando, FL, USA, Jan. 2003, pp. 209–216.

[13] C. L. P. Chen and Z. Liu, "Broad learning system: an effective and efficient incremental learning system without the need for deep architecture," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 1, pp. 10–24, Jan. 2018.

[14] H. Zhao, J. Zheng, W. Deng, and Y. Song, "Semi-supervised broad learning system based on manifold regularization and broad network," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 67, no. 3, pp. 983–994, Mar. 2020.

[15] S. M. A. A. Mamun and J. Valimaki, "Anomaly detection and classification in cellular networks using automatic labeling technique for applying supervised learning," *Procedia Comput. Sci.*, vol. 140, pp. 186–195, 2018.

[16] (2020, Dec.) Machine learning algorithms for classifying network anomalies and intrusions. [Online]. Available: http://www.sfu.ca/l̃jilja/cnl/projects/BLS_intrusion_detection/ .

[17] N. B. Aissa and M. Guerroumi, "Semi-supervised statistical approach for network anomaly detection," *Procedia Comput. Sci.*, vol. 83, pp. 1090–1095, 2016.

[18] (2021, Jan.) The NSL-KDD Data Set. [Online]. Available: https://www.unb.ca/cic/datasets/nsl.html .

[19] (2021, Jan.) Traffic data from Kyoto University's honeypots. [Online]. Available: http://www.takakura.com/Kyoto_data/data/ .

[20] S. Shriram and E. Sivasankar, "Anomaly detection on shuttle data using unsupervised learning techniques," in *Proc. Int. Conf. Comput. Intell. Knowl. Economy*, Dubai, United Arab Emirates, Dec. 2019, pp. 221–225.

[21] A. Graves, "Supervised sequence labelling with recurrent neural networks," in *Studies in Computational Intelligence*, vol. 385, J. Kacprzyk, Ed., Berlin; Heidelberg; Verlag, Springer, 2012.

[22] M. Klapper-Rybicka, N. N. Schraudolph, and J. Schmidhuber, “Unsupervised learning in LSTM recurrent neural networks,” in *Proc. Int. Conf. Artif. Neural Netw.*, Berlin, Aug. 2001, pp. 684–691.

[23] Y. Yasami and S. P. Mozaffari, “A novel unsupervised classification approach for network anomaly detection by k-means clustering and ID3 decision tree learning methods,” *J. Supercomput.*, vol. 53, no. 1, pp. 231–245, Oct. 2010.

[24] S. Fan, G. Liu, and Z. Chen, “Anomaly detection methods for bankruptcy prediction,” in *Proc. 4th Int. Conf. Syst. Informat.*, Hangzhou, Nov. 2017, pp. 1456–1460.

[25] R. Domingues, M. Filippone, P. Michiardi, and J. Zouaoui, “A comparative evaluation of outlier detection algorithms: experiments and analyses,” *Pattern Recognit.*, vol. 74, pp. 406–421, 2018.

[26] (2021, Jan.) Moskovskii elektroshok. [Online]. Available: https://www.comnews.ru/content/13693 .

[27] (2021, Jan.) RAO, “UES of Russia” Annual Report 2005. [Online]. Available: http://www.rustocks.com/put.phtml/EESR 2005 sec.pdf .

[28] (2021, Jan.) Runet nagljadno prodemonstriroval svoju bezzaŝitnost. [Online]. Available: http://www.webplanet.ru/news/internet/2005/5/25/shit_happens.html .

[29] K. G. Mehrotra, C. K. Mohan, and H. Huang, “Anomaly detection,” in *Anomaly Detection Principles and Algorithms*, Cham: Springer, 2017, pp. 21–32.

[30] (2021, Jan.) Protecting electricity networks from natural hazards. [Online]. Available: https://www.osce.org/secretariat/242651 .

[31] Z. Li, A. L. Gonzalez Rios, and L. Trajković, “Detecting Internet worms, ransomware, and blackouts using recurrent neural networks,” in *Proc. IEEE Syst., Man, Cybern.*, Toronto, Canada, Oct. 2020, pp. 2165-2172.

[32] “Cross-Validation,” in *Encyclopedia of Machine Learning*, C. Sammut, G. I. Webb. Eds., Boston, MA, USA: Springer, 2010, pp. 249–249.

[33] P. Refaeilzadeh, L. Tang, and H. Liu, “Cross-Validation,” in *Encyclopedia of Database Systems*, L. Ling and M. T. Ozsu, Eds., Boston, MA, USA: Springer, 2009, pp. 532–538.

[34] (2021, Jan.) Cross-validation: evaluating estimator performance. Scikit learn, [Online]. Available: https://scikit-learn.org/stable/modules/cross_validation.html .

[35] A. Graves, “Long short-term memory,” in *Supervised Sequence Labelling with Recurrent Neural Networks*, Berlin, Heidelberg: Springer, 2012, pp. 37–45.

[36] X. Peng, K. Ota, and M. Dong, “A broad learning-driven network traffic analysis system based on fog computing paradigm,” *China Commun.*, vol. 17, pp. 1–13, Feb. 2020.

[37] C. L. P. Chen and Z. Liu, “Broad learning system: A new learning paradigm and system without going deep,” in *Proc. 32nd Youth Academic Annual Conf. Chinese Assoc. Autom.*, Hefei, May 2017, pp. 1271-1276.

[38] Z. Li and A. L. Gonzalez Rios, Private communications, Dec. 2020.

[39] Z. Li, A. L. Gonzalez Rios, G. Xu, and L. Trajković, “Machine learning techniques for classifying network anomalies and intrusions,” in *Proc. IEEE Int. Symp. Circuits Syst.*, Sapporo, Japan, May 2019.

[40] Y. Sun, L. Guo, Y. Li, L. Xu, and Y. Wang, “Semi-supervised deep learning for network anomaly detection,” in *Algorithms and Architectures for Parallel Processing*, S. Wen, A. Zomaya, and L. T. Yang, Eds., Cham, Springer International Publishing, 2020, pp. 383–390.

[41] M. Ring, S. Wunderlich, D. Scheuring, D. Landes, and A. Hotho, “A survey of network-based intrusion detection data sets,” *Comput. Secur.*, vol. 86, pp. 147–167, 2019.

[42] K. Cho, B. van Merriënboer, C. Gülçehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, “Learning phrase representations using RNN encoder–decoder for statistical machine translations,” in *Proc. Conf. Empirical Methods Natural Lang. Process.*, Doha, Qatar, Oct. 2014, pp. 1724–1734.

[43] K. Greff, R. K. Srivastava, J. Koutnik, B. R. Steunebrink, and J. Schmidhuber, “LSTM: a search space odyssey,” *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 10, pp. 2222–2232, Oct. 2017.

[44] G. Karatas, O. Demir, and O. K. Sahingoz, “Deep learning in intrusion detection systems,” in *Proc. Int. Congr. Big Data, Deep Learning Fighting Cyber Terrorism*, Ankara, Turkey, Dec. 2018, pp. 113–116.

[45] T. Kim, S. C. Suh, H. Kim, J. Kim, and J. Kim, “An encoding technique for CNN-based network anomaly detection,” in *Proc. IEEE Int. Conf. Big Data*, Seattle, WA, USA, Dec. 2018, pp. 2960–2965.

[46] N. Shone, T. N. Ngoc, V. D. Phai, and Q. Shi, “A deep learning approach to network intrusion detection,” in *IEEE Trans. Emerg. Topics Comput. Intell.*, vol. 2, no. 1, pp. 41-50, Feb. 2018.

[47] (2021, Jan.) Cross-validation: evaluating estimator performance. [Online]. Available: https://scikit-learn.org/stable/modules/cross_validation.html .

[48] ((2021, Jan.) Google. (2020). Regularization for sparsity: $L_1$ regularization. [Online]. Available: https://developers.google.com/machine-learning/crash-course/regularization-for-sparsity/l1-regularization .

[49] (2021, Jan.) The Cisco learning network. [Online]. Available: https://learning network.cisco.com/s/question/0D53i00000Kt28w/nlri-what-is-this .

[50] (2021, Jan.) BGP network layer reachability information (NLRI). [Online]. Available: https://bit.ly/2Lwod2c .

[51] (2021, Jan.) Accuracy vs. F1-Score. [Online]. Available:https://medium.com/analytics-vidhya/accuracy-vs-f1-score-6258237beca2 .

[52] (2021, Jan.) RIPE NCC. [Online]. Available: https://www.ripe.net .

[53] (2021, Jan.) University of Oregon Route Views Project. [Online]. Available: http://www.routeviews.org .

[54] (2021, Jan.) Z. Li, A. L. Gonzalez Rios, and L. Trajković, Border Gateway Protocol routing records from Reseaux IP Europeens (RIPE) and BCNET. [Online]. Available: https://ieee-dataport.org/open-access/border-gateway-protocol-routing-records-reseaux-ip-europeens-ripe-and-bcnet .

[55] A. L. Gonzalez Rios, Z. Li, G. Xu, A. Diaz Alonso, and Lj. Trajković, "Detecting network anomalies and intrusions in communication networks," in *Proc. 23rd IEEE International Conference on Intelligent Engineering Systems 2019,* Gödöllő, Hungary, April 2019, pp. 29-34.