Master's thesis

NTNU
Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Dept. of Information Security and Communication Technology

Benjamin Grjotheim Dybvik
Johanne Kaatorp

A Dynamic Graph, Context, and Content Analysis Approach to Detect Cybergrooming

Master's thesis in Information Security
Supervisor: Patrick Bours
June 2024

NTNU
Norwegian University of
Science and Technology

Benjamin Grjotheim Dybvik
Johanne Kaatorp

A Dynamic Graph, Context, and Content Analysis Approach to Detect Cybergrooming

Master's thesis in Information Security
Supervisor: Patrick Bours
June 2024

Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Dept. of Information Security and Communication Technology

NTNU
Norwegian University of
Science and Technology

Problem description

Title: A Dynamic Graph, Context, and Content Analysis Approach to Detect Cybergrooming
Students: Banjamin Dybvik & Johanne Kaatorp

During the evolution of the internet, how we communicate with each other has changed from in-person communication to more extent use of online communication. This applies to people of all ages, but especially the younger generation who are growing up in a more digital world than ever. Today, it is common that children meet online through social media platforms, rather than at the playground. These online activities ease the grooming process for predators who no longer need to put themselves at risk and hide when at public places. Instead, they can move from one online platform to another looking for their next victim anonymously, even pretending to be someone else. This makes it easier than ever for predators to contact children and establish trust with children to later be able to sexually abuse them. There have been conducted several academic studies that aim to find ways of detecting predators. Most of these studies are of a forensic nature and focus on detecting predators rather than preventing them in real-time.

This master's thesis aims to contribute to the research field of preventing cybergrooming by studying different approaches to detect potential predators. By dynamically looking at messages sent on a game platform geared towards children, the goal is to attribute each user a risk score. The score will be updated dynamically as the users keep sending messages. The accumulated scores will be used to detect suspicious users.

Approved on: 2024-02-20
Supervisor: Patrick Bours, NTNU, IIK

Abstract

The internet, and especially social media, has become a fundamental part of our life, no matter the age. Many social media platforms are targeted towards children enabling them to contact new friends without the need for physical meetings. Despite the clear benefits of social media, it also raises concerns about threats facing children online. These platforms do not only give access to children, but also to people with bad intentions, such as predators. Predators can create fake online profiles, pose as a child, and contact vulnerable children with a minimal risk of disclosure. Online assaults can result in psychological, physical, emotional, behavioral, and psycho-social issues affecting the child for the rest of its life. To avoid such life altering consequences it is crucial to detect and prevent sexual abuse online.

During this thesis we have investigated whether a combined graph, context and content analysis approach could be used to dynamically detect predators online. This was accomplished by studying the behavior of individual users in game chats. We implemented a supervised machine learning algorithm which classified the messages sent by the users based on several behavioral features. Further, a detection mechanism was created to detect predators as early as possible whilst achieving high recall and precision.

Based on the results achieved we concluded that dynamic detection of predators in chats is possible. In addition, we concluded that early detection of predators was possible when monitoring the user's behavior in ongoing chats. To continue the research into improving detection, the use of other classification algorithms, inclusion of other features and approaches to calculate them, and other detection mechanisms should be studied.

Sammendrag

Internett, og spesielt sosiale medier, har blitt en fundamental del av livene våre, uansett alder. Mange sosiale medieplattformer er rettet mot barn og gir dem mulighet til å kontakte nye venner uten å behøve og møtes fysisk. Til tross for de tydelige fordelene ved sosiale medier åpner det også for bekymringer knyttet til truslene som møter barna på nett. Disse plattformene gir ikke bare barn enkel tilgang, men også personer med fiendtlige intensjoner, slik som overgripere. Overgripere kan opprette falske nettprofiler, utgi seg som barn og kontakte sårbare barn, med minimal risiko for å bli avslørt. Nettovergrep kan resultere i psykologiske, fysiske, emosjonelle, adferdsmessige og psykososiale problemer som kan påvirke barnet livet ut. For å unngå slike livsendrende konsekvenser er det avgjørende å detektere og forhindre seksuelle nettovergrep.

I løpet av denne masteravhandlingen har vi undersøkt hvorvidt en kombinert tilnærming av graf-, kontekst- og innholdsanalyse kan benyttes for å dynamisk detektere overgripere på nett. Dette oppnådd vi ved å studere oppførselen til individuelle brukere i spill chatter. Vi implementerte en veiledet maskinlæringsalgoritme som klassifiserte meldingene sendt av brukere basert på flere atferdsegenskaper. Videre ble det implementert en deteksjonsmekanisme for å detektere overgripere så tidlig som mulig samtidig som høy dekning og presisjon kunne oppnås.

Basert på de oppnådde resultatene konkluderte vi med at dynamisk deteksjon av overgripere i chatter er mulig. I tillegg konkluderte vi med at tidlig deteksjon av overgripere var mulig ved å overvåke brukernes atferd i pågående chatter. For å fortsette forskning for å forbedre deteksjon, bør bruken av andre klassifiseringsalgoritmer, inkludering av andre adferds-egenskaper og tilnærminger for å beregne dem, samt andre deteksjonsmekanismer studeres.

Preface

This thesis is written as the completion of the experienced based MSc in Information Security at Norwegian University of Science and Technology (NTNU). The supervisor for the thesis has been Professor Patrick Bours at the Department of Information Security and Communication Technology at NTNU. The research for this thesis was performed in cooperation with the company Aiba. A preliminary study for this master's thesis was conducted in the spring of 2023.

Acknowledgements

We would like to thank our supervisor Patrick Bours for his advice and support. His insights have been crucial in the making of this thesis. Thank you for putting up with us for the past 1.5 years and showing genuine interest for this thesis. We would also like to thank our family and friends for their support throughout the entire course of study.

Contents

Figures

Tables

Acronyms

ACC Accuracy.

Avg Msg Average Message.

BoW Bag of Words.

CTDG Continuous-Time Dynamic Graph.

DEC Decision.

DT Decision Tree.

DTDG Discrete-Time Dynamic Graph.

FN False Negative.

FNR False Negative Rate.

FP False Positive.

FPR False Positive Rate.

GAD Graph Anomaly Detection.

GAN Generative Adversarial Net.

GAT Graph Attention Network.

GC Graph Convolutions.

GCN Graph Convolutional Networks.

GNN Graph Neural Network.

LSTM Long Short-Term Memory.

MLP Multilayer Perceptron.

Msg Message.

NBC Naïve Bayesian Classifier.

OCAN One-Class Adversarial Nets.

PR Precision.

RC Recall.

RF Random Forrest.

SBB Social Behavior Biometric.

SCI Suspicious Conversation Identification.

STAB Stabilization.

SVM Support Vector Machine.

TADGNN Time Augmented Dynamic Graph Neural Network.

TN True Negative.

TP True Positive.

TPR True Positive Rate.

VFP Victim From Predator.

Win Size Window Size.

Chapter 1

Introduction

This master's thesis aims to determine if it is possible to use a graph theoretical approach with context and content analysis to detect predators in online game chats created for children. In this chapter, the thesis will be introduced, including the motivation for studying this topic, the research questions, the thesis's outline, and finally a disclaimer. Some sections of this chapter are based on an earlier preparation for this thesis [1], although additional information has been supplied afterwards.

1.1 Motivation

There are countless ways to get in touch online today, and many social media platforms are targeted towards children. For example, game platforms which in addition to hosting a game, can provide chatting opportunities where players can chat with each other during the game. This form of chatting is popular amongst children. Most of these platforms also allow the players to find friends and it is common to create new friendships or relationships online. Social media is thus very important for children's social life.

Even though social media clearly has several benefits it also raises some concerns regarding threats that children might face whilst online. The ease of getting in touch with new friends is not limited to children, also people with bad intentions have access. Predators can create fake online profiles and pose as children to enable them to contact children with a goal of, for example, sexual abuse. All children on social media platforms can be exposed to predators. According to the project deShame from 2017, 23% of teens in the age of 13-17 received unwanted sexual messages and images online [2]. In addition, during the Covid-19 lockdown several countries reported an increase of online sexual exploitation of children [3][4]. Some reported an increase of as much as 17% only during the first six months [3]. Online assaults against children may result in the child having psychological, physical, emotional, behavioral, and psycho-social issues for the rest of their lives [5]. To avoid these consequences it is crucial to detect and prevent sexual abuse online.

Several academic studies have been conducted to find ways of detecting predators. Most studies are of forensic nature and do not try to prevent predators in real-time, but rather detect them after the fact. Of the studies that focuses on detection, most of them are based on content analysis in forms of analyzing the chat messages aiming to determine different known vocabulary represented in the chats and hence attribute the message as predatory [6] [7] [8]. Some studies use a static graph theoretical approach to detect anomalies based on how they act in a network [9] [10] [11] [12].

Both content analysis and a graph theoretical approach on its own have shown us to achieve satisfactory results. But both also have some disadvantages, for example, content analysis alone can work poorly if chats consist of different languages since there are no one-to-one relation between words of different languages. A graph theoretical approach requires a lot of data to accurately detect predators which can prolong the detection time of predators. By combining these two detection methods some of the disadvantages may be eliminated. With the graph theoretical approach, we will analyze the chat behavior between users on a game platform and try to recognize patterns that deviate from normal users. With content analysis we will analyze the chat content and try to establish features associated with predatory choice of words. During the literature review, no previous studies were found which uses this method for predator detection, hence will this thesis provide new insights to the field of predator detection.

1.2 Research Questions

This thesis has one main research question and two sub-questions which will help answer the research question. Research questions and sub questions were created during the pre-project proceeding this master thesis [1]. As more insights into the project were acquired, the research questions were adapted. The research question for this thesis is as follows:

> **RQ1**: Can graph theory, context, and content analysis be utilized to dynamically detect predators in chats?

The main goal of this thesis is to investigate whether a combined graph, context and content analysis approach can be used to dynamically detect predators online. The intent of the research question is to study different approaches to detect abnormal users in chats. Abnormal users, in the context of this thesis are users who exhibit predatory behavior, but it can also include spammers, users who are sexting or others who display behavior not categorized as normal. In earlier and similar studies, such as the dynamic graph theoretical approach by Eng [12], the method only considered graph theoretical, and context approaches to dynamically detect predators in chats. In this thesis, the goal was to examine whether a content analysis approach together with previously studied approaches could enhance the

results of dynamically detecting predators in chats.

Two sub-questions have been defined to help answer the overall research question. The sub-questions are as follows:

> **SubQ1:** What features are suitable for detecting predators in chats?

The first sub-question aims to explore if there are any distinct behavioral characteristics which can be used to differentiate between normal and abnormal users and whether they can be used by a classification model. Exploring the performance of different feature sets can be used to gain insight into how well the classification and detection of abnormal users can perform. This insight will help answer the overall research question.

> **SubQ2:** What combinations of graph theory, context, and content analysis can optimize detection of predators in chats?

The second research question aims to explore different combinations of graph, context, and content analysis which can help optimize the detection of abnormal users. The question will aid in exploring combinations of detection mechanism variables which will be used to increase the accuracy of detecting abnormal users, and at the same time decrease the detection time. This trade-off will help determine how dynamically the proposed method can detect abnormal users. Exploring the performance of different combinations of the variables in the detection mechanism can help answer the overall question.

1.3 Outline

Chapter 2 presents the background knowledge needed to understand the research problem in this thesis. Chapter 3 covers the state of the art and related work is reviewed and presented. In chapter 4 the methodology utilized is presented. Chapter 5 includes the results followed by a discussion of these results in chapter 6. Lastly in chapter 7, the conclusion and future work are presented.

1.4 Disclaimer

The thesis aims to discover new methods to dynamically detect predators online. The dataset made available was not labeled, implying we could only look for predatory behavior without confirming whether the user was a predator or not. In this thesis we claim to look for predators, but in reality a predator might have normal conversations with most of the chat partners, but have predatory conversations with one or a few. The actual predator classification will only be based on the chat conversations in the end. The goal is therefor to perform behavioral analysis

utilizing graph theory, context and content analysis to narrow down the number of potential predators. During the classification process the term abnormal user will be used to avoid misunderstandings.

Chapter 2

Background

The background knowledge needed to understand the research problem for this thesis is presented in this chapter. First, an introduction of the term cybergrooming and its characteristics will be presented. Secondly, the most relevant concepts of graph theory will be covered. Third, fundamental of machine learning necessary for this thesis are introduced.

2.1 Cybergrooming

In this section a defenition of cybergrooming will be given. Then characteristics of cybergrooming will be introduced. Last, the different stages of cybergrooming will be presented.

2.1.1 Definition of Cybergrooming

The term cybergrooming is used when talking about sexual grooming performed in cyberspace. The term online sexual grooming is also commonly used, but in this thesis the term cybergrooming will be used. Defining the term online sexual grooming has been attempted numerous times and generally the literature on online sexual grooming relies upon the same definitions as those proposed for in-person sexual grooming. Winters and Jeglic [13] proposed a definition on sexual grooming based on an extensive research of prior definitions as follows: *"Sexual grooming is the deceptive process used by sexual abusers to facilitate sexual contact with a minor while simultaneously avoiding detection. Prior to the commission of the sexual abuse, the would-be sexual abuser may select a victim; gain access to and isolate the minor; develop trust with the minor and often their guardians, community, and youthserving institutions; and desensitize the minor to sexual content and physical contact. Postabuse, the offender may use maintenance strategies on the victim to facilitate future sexual abuse and/or to prevent disclosure."*

The main differences between in-person sexual grooming and cybergrooming is that for cybergrooming the emotional connection developed with the victim happens in an online setting, the abuse could also be performed online and the

groomer can gain access to a much larger pool of victims. Some groomers might seek contact with minors online to later progress to in-person sexual contact. Others might only seek to engage in online activities such as sexual chats. Independent of the groomers intention as Martellozzo [14] wrote, that introducing the internet as a platform for groomers does not create new stages of the abusesycle, but the internet helps the abusecycle to go quicker.

2.1.2 Characteristics

There are several characteristics of cybergrooming and it is important to explore the types of individuals who commit those types of offenses to better understand cybergrooming.

Cybergroomers use the internet to locate and communicate with adolescents with the intention of engaging in online or offline sexual abuse. The term *online sexual solicitation* is generally used in such scenarios and involves the use of internet for interaction with minors for sexual purposes. The interactions can include different cybersexual activities such as, sending and receiving sexualized images or videos for the purpose of sexual gratification of the groomer [15].

In general, evidence has shown that cybergroomers, in comparison to in-person sexual offenders, tend to be white, younger, unemployed men who have less offense supportive beliefs and limited emotional identification with adolescents. Even though they have limited emotional identification with the victim, they are able to make the victim believe that they have higher levels of empathy [16]. In addition to the general characteristics of cybergroomers we have more defined types. The most common types of cybergroomers were presented by Winters and Jeglic [13] and a summarization of the types are shown in table 2.1.

What type a cybergroomer is categorized as depend on what their ultimate goal is. *Contact-driven offenders* aim to chat with adolescents online with the goal of moving on to engaging in sexual activities in an offline setting. *Fantasy-driven offenders* on the other hand, solely seek to engage in cybersexual activities, not moving to an offline setting [17].

Intimacy-seeking groomers use significant amounts of time talking to the minors before engaging in sexual behaviors or in-person meetings. This group of groomers view their offense as a “consenting” relationship and thus present as themselves online. The *adaptable style* groomers shape their identity and grooming strategies based on the presentation and reaction of the minor. This group of groomers tend to have prior sexual convictions, and could possess some indecent images of children. The *hyper-sexualized* groomers are highly sexual and move quickly when interacting with minors. This group of groomers tend to misrepresent their identity and contact other people who also are engaged in improper sexual activity [18].

Hyperconfident groomers are those who openly share their interest and material such as nude pictures of themselves. *Hypercautious* groomers spend time establishing relationships with the minors they contact and do not promptly share

sexual content. This group of groomers can be said to be more dangerous in the sence that they are not as easily identifiable, and they tend to maintain a passive role [14].

Cybersex only groomers tend to chat with minors for longer periods of time, such as months, asking questions about appearance, coached the minor in masturbation, and hinted on meeting in the future. Cybersex only and fantasy-driven groomers have some of the same approaches, but while cybersex only groomers wants to create an online relationship with the victim, the fantasy-driven groomers only wants to engage in online sexual activities. *Cybersex/schedulers* have the same approach as the cybersex only groomers, but they schedule an in-person meeting. *Schedulers* will try to schedule an in-person meeting in a relative short time, such as within a week. *Buyers* schedules meetings through third parties and focus on negotiating terms, such as cost and the form of sexual activity [19]

Table 2.1: Sexual grooming Typologies

Author	Typology	Description
Briggs et al. [17]	Contact-driven	chat with adolescents online, moving on to engaging in sexual activities in an offline setting
	Fantasy-driven	seek to engage in cybersexual activities, not moving to an offline setting
Webster et al. [18]	Intimacy-seeking	significant amounts of time talking to the minors before engaging in sexual behaviors or in-person meetings
	Adaptable style	shape their identity and grooming strategies based on the presentation and reaction of the minor
	Hyper-sexualized	highly sexual and move quickly when interacting with minors
Martellozzo [14]	Hyperconfident	openly share their interest and material such as nude pictures of themselves
	Hypercautious	spend time establishing relationships with the minors they contact and do not promptly share sexual content
DeHart et al. [19]	Cybersex only	chat with minors for longer periods of time
	Cybersex/schedulers	schedule an in-person meeting
	Schedulers	schedule an in-person meeting in a relative short time
	Buyers	schedules meetings through third parties and focus on negotiating terms

2.1.3 Stages of Cybergrooming

Efforts have been made to identify the different stages a predator goes through during a cybergrooming process. There exist several models showing the stages of cybergrooming, ranging from 3 to 7 stages. One model that is commonly used was described by O'Connell [20]. O'Connell wrote a grooming report where behavioral patterns were categorized in an attempt to understand how grooming advances. In the report six stages of cybersexploitation or cybergrooming is presented. For each stage there are identifiable differences between the behavioral patterns of the groomers, which is closely related to their type and hence not all groomers will progress through all the stages or sequentially through them. For example, some groomers will stay in one stage for a longer duration of time, whilst some might skip the stage entirely. But also, these variations can be used to identify groomers with bad intentions. The six stages of cybergrooming is presented below.

1. **Friendship forming stage** In the friendship forming stage the groomers intention is to get to know the child. The amount of time a groomer uses in this stage varies from one groomer to another. For example, an intimacy-seeking groomer would spend significant longer time in this stage than a hyper-sexualized groomer. Typical questions to be asked by the groomer in this stage is whether the child can share pictures of themselves, and if they do, if they can send them. The pictures do not necessarily need to be of a sexual nature at this stage of the conversation.
2. **Relationship forming stage** In the relationship forming stage the friendship forming stage is extended. As mentioned, some groomers spend lesser time in the friendship forming stage and those might not engage in the relationship forming stage at all. What characterizes this stage is that the groomer might discuss for example home and school life in more detail. The goal in this stage is to maintain contact with the child and act as the child's best friend.
3. **Risk assessment stage** In the risk assessment stage, the groomer will typically ask about, for example, if the computer the child is using is used by others or where the computer is located. By collecting this information, the groomer can assess the likelihood of the conversation getting detected by bystanders, such as the child's parents or siblings.
4. **Exclusivity stage** In the exclusivity stage the groomer will change the pace of the conversations so that the idea of the groomer being the child's best friend comes from the child itself. The interactions change in a way that are based on the characteristics of a sense of mutuality, that the groomer and child and only them have mutual respect and that they should remain as a secret from others. Typically, the groomer will question how much the child trust him and usually the child will respond by acknowledging that they trust the groomer completely. By gaining this trust the groomer can more easily move on to the sexual stage.
5. **Sexual stage** In the sexual stage the groomer can start by asking sexualized

questions such as *Have you ever kissed someone?* or *Have you ever touched yourself?*. From the child's perspective, the conversation going in this direction might seem innocuous because the groomer has arranged the conversation so that a sense of mutual trust has been established. In addition, if the child has not encountered conversations like this before it can be more difficult to navigate. On the other hand, if the child has been sexually abused before the groomer will adjust their approach to give them the most leverage with the child.

During this stage the differences between the different types of groomers are most present. For the groomers who plan to maintain the relationship with the child and keep up the appearance of a mutual trust, such as the intimacy-seeking groomers, they will more gently enter this stage. The child's boundaries are more gently pressured than with the groomers that want to meet the child physically as soon as possible. If the child, at any point feels pressured the groomer typically express deep regrets towards the child prompting forgiveness by the child which might establishes an even deeper mutuality between the groomer and the child.

As part of the relationship forming stage the groomer might create a rationale of the relationship including forming a friendship with the goal of ultimately facilitate for future activities such as meeting the child in-person. Characteristics of the sexual stage might include requests for the making, exchange, and distribution of erotic and pornographic material. Next, web cams may be used to stream videos of the child in real time.

6. **Concluding stage** There are mainly three methods a groomer can use to conclude the grooming encounter with a child and maintain control. The first tactic is the damage limitation tactic which involves positive encouragement for a child with the goal of reducing the risk of the child sharing the events with anybody else. The groomer might repeat phrases such as *This is our secret*. This is common characteristics of the concluding stage and seem to be a necessary part of the conversation.

   The second tactic is the hit and run tactic. This tactic is mostly seen used by aggressive groomers such as the hyper-sexualized groomers. The groomers using this tactic rarely were interested in damage limitation nor extending the contact with the child. They even did not schedule in an in-person meeting.

   The third tactic is adjusting for age. It is often very difficult for a child to detect that they are not talking with another child or to discover what the intention of the groomer is. The conversation will vary based on the age of the child, but when a baseline of understanding is established, the conversation continues as described above.

2.2 Graph Theory

Graph Theory has many applications in science and is ideal as a method to describe the relationship between objects. In this thesis graph theory will mainly be used to describe the relationship between users in chats, sending and receiving messages between each other.

Wilson [21] described a graph as *"A simple graph G consists of a non-empty finite set $V(G)$ of elements called vertices, and a finite set $E(G)$ of distinct unordered pairs of distinct elements of $V(G)$ called edges. We call $V(G)$ the vertex set and $E(G)$ the edge set of G."* Vertices v, also called nodes, are the fundamental entities of a graph and the edges e are the connections between two vertices. The edges are used to describe the relationships between nodes and is called an incident to two nodes [22].

There are multiple characteristics of graphs and one important is directed graphs ability to maintain the historical terminology [23]. Directed graphs, as illustrated in figure 2.1a, have edges which are directed towards one of the nodes. A directed edge is also called an arc, and is illustrated in the figure with an arrow pointing from node A to node C. In our previously example with chats, a directed graph can be utilized to represent which direction the communication is going. Where a message sent from person A to person C is represented with a directed edge pointing from node A to node C. Even though a directed graph goes from node A to node C, does not exclude that another opposite directed edge can go from node C to node A, as seen in the example between node A and B. Graphs can also be characterized as undirected graphs, as figure 2.1b, if edges are not directed. Undirected graphs indicate a mutual relationship since the edge does not indicate a direction.

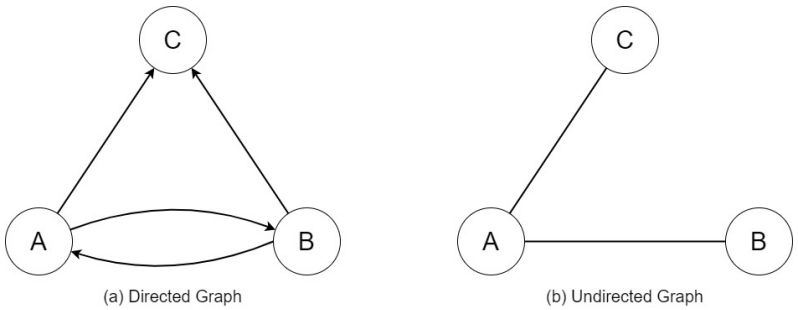


Figure 2.1: Examples of directed and undirected graphs

In some cases graphs can be weighted. Figure 2.2 is an example of a weighted graph where the edge between node A and C is weighted with the numerical value of 2. The value can represent the cost of the edge, or in our case, how many messages has been sent from node A to node C. Graph theory is suitable to describe a chat where each user is represented by a node and the edges represent messages sent between the users.

The node degree of an node can be calculated for both nodes in undirected

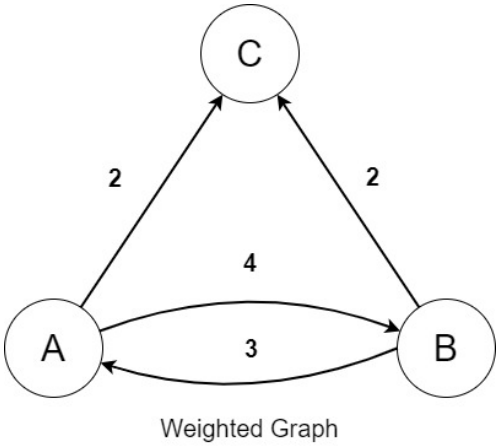


Figure 2.2: Examples of a weighted graph

graphs and nodes in directed graphs. The node degree is the number of edges incident to a node [22]. Node degree can be annotated as $deg(v)$. Two other degree measurements which are commonly used are in-degree and out-degree. They only apply for directed graphs as they rely on the direction the edge is going. In-degree is the number of edges pointing towards the node, or terminating at node v. The out-degree is the number of directed edges pointing away from the node v. Therefore, the node degree of a node in a directed graph will be the sum of the in-degree and the out-degree of the node.

There are many concepts of measurements which are much used within graph theory. Degree centrality is a type of centrality measures which like other centrality measures describe the influence of a node in a network, this is further elaborated in section 3.2. Other centrality measures are Betweenness centrality, Closeness centrality and Eigen-Value [24]. When measuring graphs, it is important to differentiate which graphs are being measured. Often it is the egocentric graph we want to study, and the egocentric graph is seen from the perspective of the node we are interested in studying. This graph will also be called an ego-graph and is a subset of the original graph, later referred to as a subgraph. Any nodes which are directly connected with an edge to the node of interest, is called a neighbor. It could be necessary to remove nodes from the ego-graph which are a given number of hops away from the main node, but the ego-graph will always be a subgraph of the original graph [22].

Cluster Coefficient is a measurement taken of egocentric graphs where the density of the graph is calculated [25]. In other words, *Clustering Coefficient* measures how connected the network of a particular node is, to understand the connectivity between the neighbors of the node. In particular, this is called the *local clustering coefficient*, *global clustering coefficient* will not be covered. *Clustering Coefficient* for node v can be annotated as $CC(v)$ and is calculated using the total number of neighbors k, the total number of possible connections between neighbors $\frac{k(k-1)}{2}$, and the actual number of connections between neighbors e. $CC(v)$

is expressed in equation 2.1 [26]. In figure 2.3 is an example of the calculations of the *Cluster Coefficient* for three different scenarios. The red dotted lines are potential edges which are not established.

$$CC(v) = \frac{2e}{k(k-1)} \tag{2.1}$$

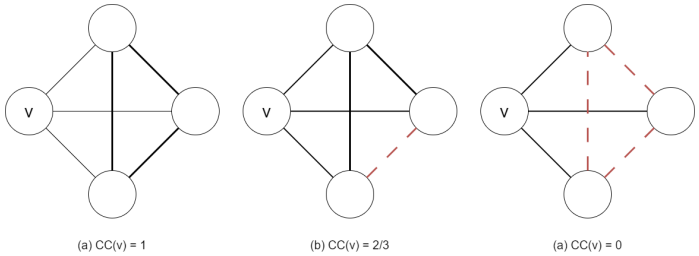


Figure 2.3: Clustering Coefficient of three graphs with different density

2.3 Machine Learning

In this section, we will cover the fundamentals of Machine Learning relevant for this thesis. Machine learning is a subset of Artificial Intelligence which could be defined as the study of how to make computers mimic human intelligence and even perform better than humans [27]. Machine Learning has many applications such as text and speech recognition, natural language processing, computer vision, robot control and many others [28]. Zhou [29] defined Machine Learning as a computational method where the system learns from experience and therefore will increase its performance over time. This technique has similarities with how humans learn from our experiences. We collect data through our observations which we base on our predictions. For example, in the morning we look outside the window to see dark heavy clouds and by our experience we can predict that there is a high probability of rain and therefore decide to bring our umbrella. With Machine Learning, learning algorithms are fed with training data, which is our experience, and the learning algorithms then create models. These models are used to make predictions or decisions based on the training data.

The data set which is used to train the model could describe an event or object, these descriptions are called features of the instances. The model's prediction can be discrete or continuous and will determine if it is a classification or regression problem. Every specific instance of events or object will have an output, these are called a label and is the prediction of the model [29].

Verifying users in a chat environment can be seen as a classification problem since a user in a chat will be classified as either a normal user or an abnormal user. The input data used in this project are text, graphs, and context. For these to be used as input in machine learning, features will be extracted and encoded as num-

bers. Next, supervised learning will be described followed by a brief introduction to the machine learning algorithms used in this thesis.

2.3.1 Supervised Learning

In supervised learning the machine learning algorithm is fed training data with observations and corresponding known output values [30]. The goal is to train a model, which maps inputs to outputs so that it is possible to predict the data which comes from unpredicted data where the input values are observed but not their corresponding output values. Supervised learning has two main categories, 1) classification - output values are categorical, 2) regression – output values are numerical. As described in section 2.3 classification will be used in this thesis.

The best practice when selecting a supervised model is to separate the data into three subsets. One subset for training, one for validation, and one for test. The training set is used to create different models. After the different models are created the validation set is used to choose an algorithm. The model performing best on the validation set is selected and the test set will be used to assess the generalization error. As labeling data to be used in supervised learning is time-consuming there will not be produced a validation set in this thesis. Since the dataset will be too small to extract a decent validation set, a cross-validation technique will be implemented.

Cross-validation is performed by dividing a training set into k subsets where subsets k-1 is used for training and the last subset is used to assess the performance. The process is repeated k times, and every subset is used once for validation. The performance score is calculated by the average for all the subsets. An illustration of the cross-validation technique is shown in figure 2.4.

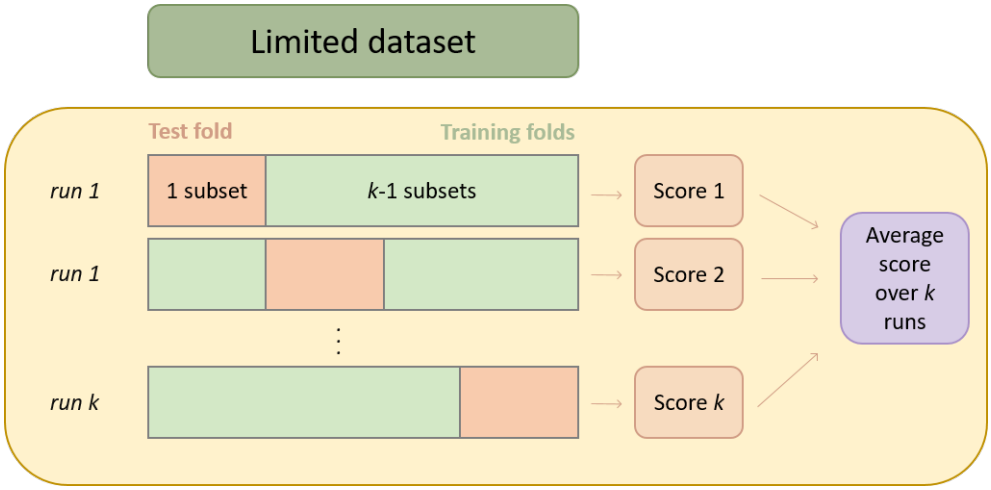


Figure 2.4: Cross-validation adapted from [30]

Performance measures is used to assess the performance of the model. For two-class classification, performance measures are usually extracted from a confusion matrix as depicted in figure 2.5. The different measures are as follows:

- *Precision (PR)*: The ratio between the True Positive (TP) values and the total number of predicted positive values (True Positive (TP) and False Positive (FP)). The equation is shown in 2.2.

$$PR = \frac{TP}{TP + FP} \tag{2.2}$$

- *False Negative Rate (FNR)*: The ratio between the False Negative (FN) values and the total number of predicted negative values (True Positive (TP) and False Positive (FP)). The equation is shown in 2.3.

$$FNR = \frac{FN}{TN + FN} \tag{2.3}$$

- *Recall (RC) / True Positive Rate (TPR)*: The ratio between the TP values and the total number of positive values in the dataset (True Positive (TP) and False Negative (FN)). The equation is shown in 2.4.

$$RC = \frac{TP}{TP + FN} \tag{2.4}$$

- *False Positive Rate (FPR)*: The ratio between the False Positive (FP) values and the total number of negative values in the dataset (False Positive (FP) and True Negative (TN)). The equation is shown in 2.5.

$$FPR = \frac{FP}{FP + TN} \tag{2.5}$$

- *Accuracy (ACC)*: The ratio between the correctly predicted values (TP and TN) divided by the total number of values in the dataset. The equation is shown in 2.6.

$$ACC = \frac{TP + TN}{TP + TN + FP + FN} \tag{2.6}$$

- *F-Score*: Combining the Precision and Recall allows us to weight Precision or Recall more highly if it is important for the use case. F_1 is used if the Recall and Precision are equally weighted. F_2 is used when the Recall is twice as important as the Precision. $F_{0.5}$ is used when the Precision is twice as important as the Recall. The equation is shown in 2.7.

$$F_\beta = (1 + \beta^2) \times \frac{PR \times RC}{(\beta^2 \times PR) + RC} \tag{2.7}$$

		Predicted labels: 1	Predicted labels: 0	
Actual labels (observations)	1	True Positive (TP)	False Negative (FN)	Recall=TPR (True Positive Rate) $TPR = \frac{TP}{TP+FN}$
	0	False Positive (FP)	True Negative (TN)	Specificity = $\frac{TN}{TN+FP}$ False Positive Rate: $FPR = \frac{FP}{FP+TN}$
		Precision $\frac{TP}{TP+FP}$	False Negative Rate $\frac{FN}{TN+FN}$	Accuracy $\frac{TP+TN}{TP+TN+FP+FN}$

Figure 2.5: Confusion matrix for performance measures [30]

2.3.2 Support Vector Machine

SVM is a two-class classification model. It can perform classification and regression tasks and handle both continuous and categorical features. It is known for its ability to determine optimal decision vectors for classification problems. This can result in high Accuracy when new unlabeled data is introduced [31]. In this thesis SVM is used for a classification problem by finding the optimal hyperplane that maximizes the margin between the opposite classes' data points. In figure 2.6 the principals of SVM are depicted. The hyperplane is a n-dimensional space where n is the number of features fed to the SVM algorithm. By maximizing the margin between data points, SVM can find the most accurate decision boundary between classes. The lines next to the hyperplane are the support vectors which run along the data points and hence represent the maximum margin. When the optimal hyperplane and the maximized margin is generated, new data can more accurately be predicted [32].

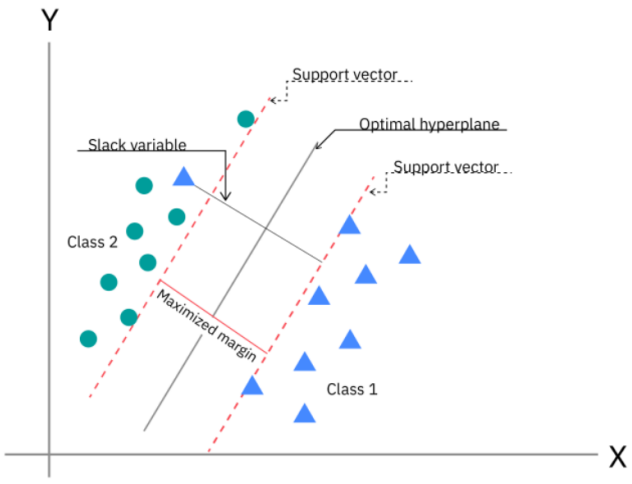


Figure 2.6: Support Vector Machine [32]

Chapter 3

Related Work

This chapter covers the state of the art in relation to the topic of this thesis. Related work is reviewed, and the relevant information is presented. First, work regarding predator detection with different techniques is covered. Next, network analysis using graph theory is presented where the objective is to cover techniques for anomaly detection. Then literature specifically focusing on predator detection using graph theory is presented. Literature has been reviewed as the state of the art, and contributed and shaped this thesis. However, this chapter covers the essential literature for the reader. The related work chapter is based on an earlier preparation for this thesis [1], although additional literature has been supplied afterwards.

3.1 Predator Detection

In the literature there are described multiple methods to detect cybergrooming in chats where the intention is to identify which user could be a predator. The majority of published literature is focusing on the content of the messages exchanged between the users. Then, for example, linguistic analysis can be used to extract features from the text which creates the foundation of the given approach to identify a predator or victim. In the following paragraphs, we will summaries different approaches, what their methods are and how they have performed.

Villatoro-Tello et al. [6] proposed a two-step approach where they were able to first identify suspicious conversations, with Suspicious Conversation Identification (SCI) and second disclose the predator from the victim, with Victim From Predator (VFP). They confirm the hypothesis which says that terms used in a chat where cybergrooming is happening is different from terms used in a normal conversation, both in a categorical and psychological manner. They also proved that predators have a unique pattern when exploiting children, which a classifier can label. Both the SCI and VFP process was solved with text classification where a Bag of Words (BoW) model was used and both classifiers used supervised learning with a labeled dataset. For VFP the messages within the chat were divided into

two inversions, one for each user, and then classified individually. The methodology performed well with a $F_{0.5}$ score of 0.93. The author emphasizes that the two-step approach, where the problem is divided into two stages, is of great importance for the methodology to perform well. The methodology was presented for PAN2012 [33], where it performed best out of all proposed methods.

A significant part of the research of predator detection has utilized a methodology where the vocabulary is analyzed. A BoW dictionary consists of words, terms or phrases frequently used by the main two groups, predators, and the victim. Other approaches in the literature are analyzing the sentiment and emotional behavior during cybergrooming. Black et al. [34] performed a study to compare face-to-face grooming and computer-mediated communication. They found that timing and order of the grooming process is somehow different and evolving when communicating in a chat. However, they found that the strategies used by predators face-to-face are present in digital communication. Although predators behave differently it is documented that predators can be emotionally unstable. For example, initially the predator will use positive words to create a relation with the victim and feel excited. Upon denial or if the predator are being impenitent, they could express anger or negativity. Lastly, most predators are aware of them performing illegal acts when cybergrooming which could result in them showing fear of getting caught.

Bogdanova et al. [7] approached the issue of detecting predators by using emotional markers as their feature set. By analyzing text messages they grouped words into categories based on the emotional expression each word represented. Sadness as an emotion could be represented by words such as "lonely" or "cry". A word list for each of the six different emotions; anger, disgust, fear, joy, sadness and surprise was then used as markers to count the occurrences of each word in the messages. In addition to the six features several others were added, such as approach words which could be "come" and "meet". In addition, other feature set which the author expected to be used by predators was included, for example family words and relationship words.

Wani et al. [8] also proposed a method to detect predators with the use of emotional behavior features. It was based on Social Behavior Biometric (SBB), and both vocabulary and emotional behavior features were used. First, a vocabulary was created with words preferred by predators and victims. Then, the MoodBook lexicon by Wani et al. [35] was used to extract the emotional-based features. The lexicon consists of 8 emotions but the method is the same as the one used by Bogdanova et al. They concluded that their emotional behavior approach by only using features describing user's emotional behavior, was not sufficient. However, combining both features based on emotional behavior and vocabulary was significant to detect predators. It was tested with three machine learning models, Support Vector Machine (SVM), Decision Tree (DT) and Random Forrest (RF), where it outperformed earlier PAN2012 solutions on detection rate and accuracy. The approached solution with RF performed best with an $F_{0.5}$ value of 0.96 and an accuracy of 0.99.

Rahman et al. [36] combined three different machine learning methods creating a model called DT-SVMNB to detect abnormal users in social networks. DT-SVMNB classified social network users as suicidal or depressed by using Decision Tree, Support Vector Machine and Naïve Bayesian Classifier (NBC). The first process of DT-SVMNB was to select necessary features which could differentiate normal and abnormal users, as seen in figure 3.1. They analyzed two sets of features, user's profile-based features and content-based features. Second, a supervised Decision Tree classifier was used to filter anomalous users and pass them to the SVM classifier. The SVM classifier distinguished between happy and disappointed users and passed all disappointed users to NBC which lastly detected if they were suicidal or not. Two datasets were used for testing and comparison, one synthetic dataset and one real. Based on both behavioral feature and user's content, DT-SVMNB outperformed compared classifiers and reached an accuracy of 0.97.

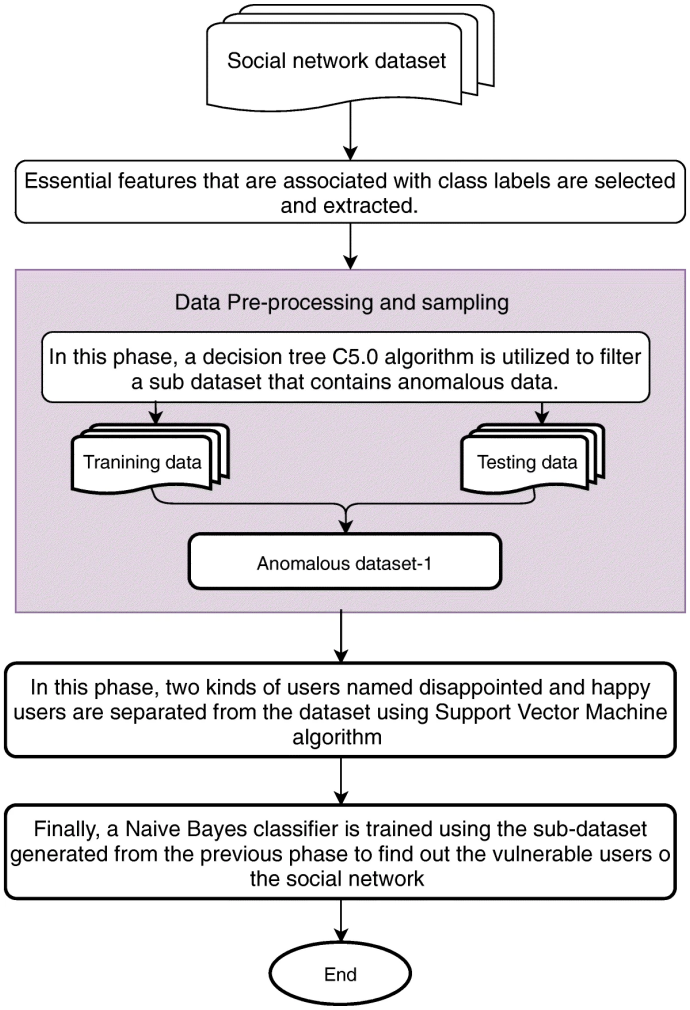


Figure 3.1: The flow of proposed method, DT-SVMNB [36]

Tang [37] proposed a risk-based system to address the problem of early detection of cybergrooming in online conversations. The main objective was to create a mechanism which could produce a risk value for each conversation. The risk value is to be updated for each message sent and received, and then update the risk value for the given conversation. If the risk value reaches a given threshold, an alarm should be trigged to the moderator. The risk score was calculated using a language model called Zero-shot, created by Aiba, which gave a score between 0 and 1 based on considering if the message was normal or sexual loaded. The risk update model also consists of a fine tuning mechanism which impacted the parameters used to calculate the risk value. Tang found the proposed risk-based system to perform well and was able to detect predators in an early phase of the conversation.

Cheong et. al [38] proposed a method to detect predatory behavior in MovieStarPlanet chats utilizing different machine learning methods for text classification. MovieStarPlanet provided two datasets, one with predator (labeled) data and another with non-predator (unlabeled) data. The unlabeled dataset spanned over 15 minutes and contained approximately 65.000 lines, whilst the labeled dataset contained the full chatlogs for the predator which could span over as much as three months. As part of the preprocessing, they manually removed lines that did not show predatory behavior. Further they created three data subsets, the first included the last 15 minutes of the predator chats (L15), the second included the whole predator chats divided into 15 minutes chunks (W15). The third dataset included one 15-minute chunk of handpicked predator chats (HP15). The labeled dataset was concatenated into 15 minutes chunks to match the unlabeled dataset. Then 14 features plus BoW features were extracted. The features were used to train and test 6 classification algorithms with 5-fold cross validation. The best results were achieved when combining all features and using SVM and Multilayer Perceptron (MLP) on the HP15 and W15 datasets. The highest accuracy was 0.93 with a F_1 score of 0.78 and a $F_{0.5}$ score of 0.86.

More recently, a graph theoretical approach to predator detection has been tested. This is covered in the next section after we review some overall literature on graph theory and network analysis.

3.2 Network Analysis with Graph Theory

When performing social network analysis, graph theory can be applied to describe the characteristics of the network structure. Majeed and Rauf [39] mention five questions to some network characteristics which could be of importance to understand graph structure properties and each users' behavior in a social network.

1. *How highly connected is an entity (Social Network user) within a network?*
2. *What is an entity's overall importance in a network?*
3. *How does information flow within a network?*
4. *How central is an entity within a network?*

5. *Can a particular entity be an influential spreader in a graph?*

Mathematical formulas are used to calculate properties of the graph structure and to describe the social network and/or its nodes within the network. These centrality measures are much used in network analysis, and four important of them are Degree Centrality, Closeness Centrality, Betweenness Centrality and Eigenvalue Centrality [39]. The calculation can be seen as a measure of how important a node is [40]. Degree Centrality is the numbers of edges a node has in the network. Closeness Centrality is the measure of how close a node is to all the other nodes. Betweenness Centrality is the value of how important a node is for connecting the network, for example connecting two communities. Eigenvector centrality is a complex centrality measure and describes the influence a particular node has in its network. We will see that centrality measures have been important for most of the related work to network analysis.

Fire et al. [9] proposed a method to detect malicious profiles like fake profiles, social-bots and sexual predators using the features of the social network topology. First, they used the Louvain methodology which detects communities in the network by optimizing the modularity. Then they extracted four features: "*(1)The user degree, (2)User's connected communities number, (3)The number of connections between the user's friends, (4)The average number of friends inside connected communities*" [9]. Features were selected based on their theory that victims were randomly selected by the attacker and therefore they assume that the attacker is reaching out to users from different communities, where a community is a group of acquaintances. Second, feature number 4 regarding the average number of friends inside connected communities, were added because the attacker is less likely to contact victims who know each other. Two supervised learning algorithms, Decision Tree and Naïve Bayesian Classifier, were used to create two classifiers. Both were able to detect anomalies in the network topology. This study is more related to anomalies in a network which have proven to be efficient towards fake profiles, spammers and other users targeting random people, which in some cases could be predators but not exclusively.

Wang [10] implemented a spam detection system towards Twitter which detected suspicious users. A graph model was used to map the relation between users and followers, they used a directed graph because Twitter users are followers and not necessary friends, and hence, being a follower is not the same as being in a mutual relation. The proposed methodology used both graph based and content based features. Examples of graph based features were the in-degree and out-degree of a specific user. Content based features were the degree of duplication where two messages were compared to check how similar they were since spammers often publish the same content or an edited version. Several classification algorithms were tested and the Naïve Bayesian Classifier performed best. The author highlighted that the Naïve Bayesian Classifier is more noise robust due to the fact that the relationship between the spam messages and the feature set is non-deterministic.

Muchnik et al. [41] studied social network to understand the main character-

istics. Three general properties are highlighted; high clustering, short distances and power-law degree distribution. The high clustering and short distances is a phenomenon mentioned by Watts and Strogatz [42], and is also known as the *small-word* characteristic. The power-law distribution has also been seen in other studies where the majority of nodes have a limited number of connections, and fewer nodes have many connections and are causing the high clustering in the network. Muchnik et al. [41] also studied the degree distribution by analyzing the contribution on Wikipedia. They observed a heavy-tailed degree distribution and that a minority of the users were responsible for the majority of the activities.

Aarekol [11] studied a graph theoretical approach to predator detection. An unlabeled dataset from a game chat was represented as a graph and analysed. Then a feature set was extracted. Aarekol's intention was to detect anomalies only by analyzing the chat behavior. A hypothesis was that a predator will behave different than normal users, and that the pattern will be detected. Before Aarekol tested her hypothesis with different clustering algorithms she extracted 22 features based on an analysis of the features in the dataset. The features represented important characteristics and gave insight for the classifier to detect abnormal behavior. Among all 22 features were node's degree, number of conversations, neighbors, in-degree, out-degree and different features measuring the length of the conversation.

Aarekol's dataset consisted of non-continuous data with different time intervals, the longest for five months. Six different clustering algorithms were tested; Agglomerative clustering, BIRCH, k-means, mean shift, DBSCAN, and Gaussian Mixture Model. Due to the complexity of agglomerative clustering algorithm it was not applied on larger datasets. Aarekol was not able to confirm the results because the data was not labeled with information regarding which users that were predators, instead she had to manually check the messages for features indicating the presence of predators. An user she suspected was a predator was confirmed with a person with knowledge of the dataset.

Analyzing the result, Aarekol did conclude that it is possible to detect predators in chats with clustering algorithms. In her case the BIRCH algorithm disclosed most predators or other users with illegal activity. As her method disclosed users with abnormal behavior it was not exclusively predators, abnormal users were found in smaller clusters. The timespan of the dataset did affect the methodology and results a bit different. The dataset containing data collected over five months was often too large to efficiently examine in a manual manner. Smaller datasets, for one day, did capture many false positives as normal users were classified with abnormal behavior, most likely due to little data. Aarekol found monthly datasets the most valuable as they had enough data to classify user's behavior, it was a continuous dataset, and clusters were not too large to examine.

For future work, Aarekol mentioned dynamic graphs to examine how users behave over a period of time which can enlighten features which are not possible to measure in a static graph. Another perspective is the timeframe for detection as it should be long enough to be accurate and precise, but not too late to detect

illegal activity. Another aspect was to include centrality features which were too computationally exhaustive to calculate.

3.3 Dynamic Graph

The research represented which includes graph theory and predator detection focuses mainly on the graph as a static element which is normally not the case. Harary and Gupta [43] studied graph theory to propose models which could be used in applications of dynamic graph theory. Most use cases for graph theory are dynamic, like a social network where participants are chatting together. Users are represented as nodes and their communication with other users are represented as edges. As users are joining and leaving, nodes are created and eliminated. Harary and Gupta proposed two approaches for studying dynamic graphs. The first approach is using logical programming paradigm modelling a dynamic graph. Logical programming consists of statements which accomplish facts. The second approach is to study the dynamic graph as a sequence of static graphs. To examine the features of the graph they features have to be studied for each sequence of the graph.

Further, studying dynamic graphs is not a well-studied topic. What is special about dynamic graphs is that the structure of the graph, the attributes of the nodes and node labels are temporal and will evolve over time. Study of temporal graph embedding has led to two categories, or methods, which are used to analyse dynamic graphs; Continuous-Time Dynamic Graph (CTDG) and Discrete-Time Dynamic Graph (DTDG) [44][45]. It is easy to look at DTDG as a series of static graphs sampled at a specific time frame. These are snapshots which represents the dynamic graph over a period of time. On the other hand, a CTDG is a continuous graph which also has a static representation. However, since it is a continuous-time process the events are updating the graph instead of a given time interval. An event could be a node or edge deletion/addition or updating of edge/node features. Figure 3.2 is a representation of the evolution of CTDG. The figure shows that for each time interval an new event has occurred. Zhong and C. Huang [45] mention both advantages and disadvantages with CTDG and DTDG. DTDG is easier to use as a method for analyzing static graphs because it can be applied for each sub-graph of DTDG, and then updating the feature set and continue onto the next sub-graph. However, DTDG handles less temporal data then CTDG and is resource demanding on larger graphs as it holds a static graph for each timestamp. DTDG is therefore often used on smaller datasets. CTDG stores more temporal information about the graph since it is event/observation oriented. CTDG do have an issue in temporal graph transformation since the time requirement often leads to only aggregating one-hop neighbors' information.

The applications of dynamic graphs are broad as they describe an evolving environment such as social networks, computational finance (customer, supply chain, relations), biology (protein interaction), or recommendation systems for products, movie categories or media content[47]. Feature extraction is necessary

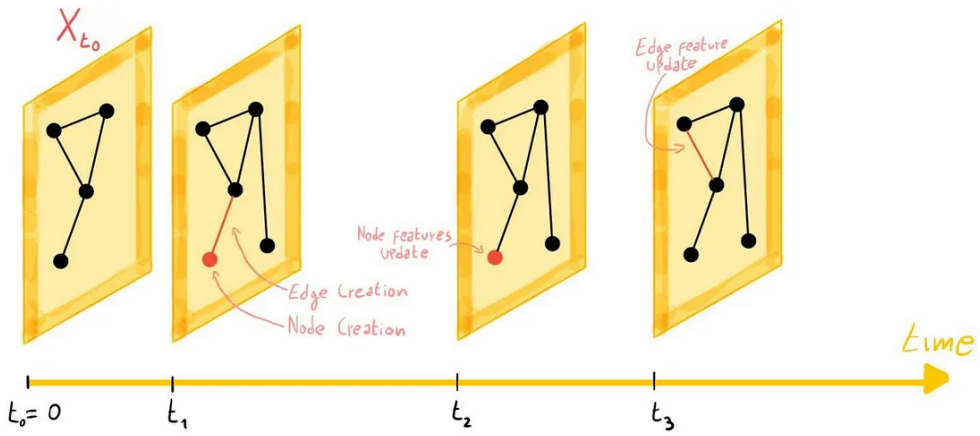


Figure 3.2: Evolution of CTDG [46]

to capture properties which are valuable in a temporal dataset. The goal could be to predict link formations, events, node attributes, node clustering, community detection or node classification which, all could be relevant for predator detection. Kazemi et al. [47] presented a survey on representation learning for dynamic graphs, both discrete and continuous-time events, to review techniques that produces time-dependent embedding. It was observed that classification in a dynamic environment was difficult due to changes in the distribution over time. They concluded that there is a lack of research on CTDG and techniques often implies sampling a dynamic graph into a series of static graphs, like DTDG.

Manessi et al. [48] mention that most applications of the graph domain can be categorized within two main categories. These are vertex-focused and graph-focused. Vertex-focused is the process of classifying vertices, or nodes, within the graph. Examples of these classifications could be image and object detection, or classification of a web page to determine the theme or content. Graph-focused is classifications performed on the whole graph and they are using a biological example to determine the likelihood that chemical bindings will occur. Then an atom is represented by a node/vertex and the bindings between the atoms are represented by edges.

Manessi et al. [48] approached the issue of working with dynamic graphs by exploiting the architecture of already existing neural networks. Their intuition were to propose a model which would maintain as much temporal information as is possible when using dynamic graphs, by using existing architecture of neural networks applied on structured graphs. The approach was combining Graph Convolutional Networks (GCN) and Long Short-Term Memory (LSTM). GCN has good performance for applications where graph structured data is used, but cannot deal with the time-dimension of dynamic graphs. LSTM on the other hand, is an algorithm which handles dependencies both in short and long range in the time-dimension. Manessi et al. proposed two solutions with two different; Graph Convolutions (GC), *Waterfall Synamic-GC* and *Concatenated Dynamic-GC*. For classi-

fication both could do semi-supervised and supervised classification. Both applications performed better than the baseline they were compared with. Although, *Waterfall Synamic-GC* could handle more complex graph structures, whilst *Concatenated Dynamic-GC* handled noise in the dataset better, which is good when graph structures are limited.

Zheng et al. [49] proposed a model called One-Class Adversarial Nets (OCAN) for fraud detection in social networks. OCAN consists of two steps where the first is learning user representation where LSTM-autoencoder is used. For the former methods the dataset included malicious and benign users for training, OCAN only needs a dataset with benign users to learn the LSTM-Autoencoder, so called one-class classification. The second step was to learn a Generative Adversarial Net (GAN) which could distinguish benign users from malicious users. GAN generates a representation of a fake user and trains a discriminator to distinguish the two user groups. OCAN only looks at node attributes by extracting user's content features for classification. Among other datasets, OCAN was tested on a Wikipedia dataset to detect vandals in a dynamic manner. OCAN outperformed all the state-of-the-art classifiers it was compared against; OCNN, OCGP and OCSVM.

Yu et al. [50] did also approach a novel method, NetWalk, with the intention to detect anomalies in dynamic graphs with a flexible deep embedding approach. This approach is focused on the network structure and how the structure is updated. NetWalk aims to detect anomalous nodes, edges and communities in real-time. The dynamic network is encoded to a network representation which is a feature matrix where a row is a vector representation of a node in the network. NetWalk uses a proposed network representation which is a network embedding algorithm they called *clique embedding*, which is defined as a network walk. In addition, a model for updating and maintaining the network representation was created to capture the dynamic of the graph structure. Clustering algorithms were applied on the low-dimensional node representation to detect anomalies. NetWalk was tested on multiple dynamic networks, like a social network where users exchanged messages. When NetWalk was compared with other techniques, it was shown that NetWalk was both computationally efficient and outperformed the others.

Ma et al. [51] performed a comprehensive survey on Graph Anomaly Detection (GAD) with deep learning. Four overall methods of GAD are covered; edge detection, node detection, graph detection and sub-graph detection. They reviewed the state-of-the-art when it comes to GAD looking both at static and dynamic graphs. They found that GAD for node detection to be more challenging for dynamic graphs because of the large volume of data and the temporal information following the sequence of graphs. On the other hand, they found dynamic graphs to be more detailed and could potentially find anomalies which is not possible in static graphs. *"In fact, some anomalies might appear to be normal in the graph snapshot at each time stamp, and, only when the changes in a graph's structure are considered, do they become noticeable."* [51]. Their survey concluded that most deep learning models for GAD are not designed for dynamic graphs, but for static

graphs. Another aspect is that few of the reviewed papers does account for complex dynamics in graphs. Most studies only address the change in either network structure or node attributes. Ma et al. concludes that both components have to be studied together to reflect the real word.

Sun et al. [52] performed node classification on a dynamic graph via time augmentation. Graph Neural Network (GNN) has mainly been applied to static graphs, however, they propose a new framework based on GNN called Time Augmented Dynamic Graph Neural Network (TADGNN). TADGNN consists of a two step method. First, a time augmentation module is applied. The intention of the time augmentation module is to create a time-augmented spatio-temporal graph, meaning capturing the temporal evolution of every node across the time frame using DTDG. Secondly, an information propagation module is used to understand both the temporal and structural properties of the graph. For testing, four real world datasets were used, where two was from social network sites; Wiki and Reddit. The two remaining networks were from two rating sites. TADGNN was compared to four other classification algorithms and outperformed all algorithms on all datasets except Graph Attention Network (GAT) which performed better on one dataset, but GAT had larger variance on remaining datasets. They concluded that TADGNN did perform better and was more efficient in time and space complexity.

Eng [12] explored the possibility to utilize dynamic graph theory in early detection of abnormal users in chat conversations. The aim was to use chat patterns to disclose abnormal users in an ongoing conversation. After studying different behavior features of users in chats, Eng chose 25 different features to further examine. The different features were for example different user's characteristics which where numerical values calculated by, numbers of messages which were sent between already connected neighbors, messages sent per minute, number of messages between specific users, etc. To simulate a real time conversation, messages were analyzed message by message continuously updating the features. SVM was used as classifier and utilized the features to create a probability score. The probability score was continuously calculated and resulted in an individual timeline of probabilities for each user. Then a threshold was used to distinguish between normal and abnormal user based on their probability score.

During Eng's testing of the probability timeline she discovered that in the beginning of a conversation the probability score varied a lot, and often oscillated between the threshold of abnormal users. Therefore, Eng waited to after 200 messages was sent, as the probability timeline had stabilized, and then could classify if a user had normal or abnormal behavior. She was not concluding that early detection was not suitable before 200 messages were sent, but the trade-off between false positive and true positive was acceptable [12].

Chapter 4

Methodology

In this chapter the methodology will be presented. The methodology used in this thesis is based on our previous work as part of the preparation for the thesis [1]. The methodology has been adapted based on new insight and experience. Literature review was first chosen as a method to obtain relevant knowledge on the topic. It enabled us to try already developed techniques which we evolved further, tested in new combinations, and then evaluate it and again iterate over method and improved it. As our methodologies main purpose was to answer the research question, our initial method was to study and analyze the dataset to get valuable information. The properties of the dataset is first presented in section 4.1 followed by the analysis of the data in section in section 4.2. The analysis of the data was the foundation of the feature selection 4.3 and lead to transformation of data which was used for testing and training a classification and detection model.

An important part of the methodology was to iterate over different methods to improve each sequence. After features were tested, additional features could be created, evaluated and compared to the other results. Although, the following chapter might seem like a sequential list of methods which is followed, this is not the case. Each iteration is not described in detail, although the implementation will describe the different combinations of features which has been iterated over. For instance, the data analysis methods were also iterated over to capture new characteristics of the dataset which was further used in new iterations. Iterating over different methods, which will be presented in all the following sections of this chapter, resulted in continues improvement of the result as well as a better understanding of how to answer the problem description.

4.1 Data Collection

In this section the fundamentals of the dataset will be presented and described, as well as initial preprocessing performed. The dataset used in this thesis was gathered by Aiba from a game platform. The games on this platform is created for children between the ages of 8 and 15, but with no upper or lower age requirements. In the game, the player creates an avatar which can meet other avatars

and chat either privately or publicly. A big part of the game is to crate new relationships or friendships. Because of this, many chats revolve around topics such as romance. The platform has several terms and conditions that the players need to obey. It is, for example, prohibited to write things that are sexually suggestive, abusive or racist. To further ensure such language is not used, a filter is used to prohibit the use of sexually suggestive, abusive, or racist words. Even when using a filter, they are not able to remove such words if they are intentionally circumvented by for example adding some special characters into the word or intentionally misspelling them. Examples of such attempts to circumvent the rules will be presented in section 4.6.

To ensure the privacy of the users attending the platform, the dataset was preprocessed by Aiba. Originally the dataset contained information such as usernames of the users playing. To ensure that the privacy of the users was obtained, all users were given a random ID. Only personnel from Aiba could link the IDs back to the original usernames. In addition to sorting the whole dataset by the time and date, and removing any duplicated messages, this was the only preprocessing performed on the dataset. Further, to ensure that no data were disclosed we signed an NDA and were given access to a workspace in the Azure Machine Learning Studio platform. The dataset was uploaded to Azure and all processing of the data was done using that platform. This ensured that the data was isolated and never were downloaded locally or distributed outside the platform.

4.1.1 Dataset Properties

The dataset uploaded to the Azure platform spanned from 2022-08-01 to 2022-10-31. It contained 80 parquet files with an average of 355'000 data entries, all unlabeled. Every entry consisted of 6 objects and represented one message from one player to another in a private chat. Below is a list of the 6 objects with an example from one message in the dataset. More examples from the dataset is listed in table 4.1

- **dateUtc:** The dateUtc object contains the date and time for the current message with format yyyy-mm-dd hh:mm:ss.sss
  Example from dataset: *2022-10-07 07:22:44.580*
- **messageId:** The messageId object is a 32-character random string representation of each message in a conversation.
  Example from dataset: *C2EC7D9DF79F90AC0A7B865CE5DF8870*
- **Context:** The Context object is a 32-character random string representation of the conversation between an initiator and a receiver.
  Example from dataset: *59F80B5B138AFB2B65157BD01C207191*
- **Initiator:** The initiator object is a 32-character random string representation of the user sending the current message.
  Example from dataset: *4702C8FA837F04F46AD533CF1535AEAF*
- **Receiver:** The receiver object is a 32-character random string representation of the user receiving the current message.

Example from dataset: *682940073C0485F5A918E04192492189*

- **Content:** The content object contains the actual text that was sent from the initiator to the receiver.
  Example from dataset: *your beautiful*

Table 4.1: Example from dataset

dateUtc	messageId	context	initiator	receiver	content
2022-08-01 00:00:00	1EE2C...	B3BFE...	A625A...	8D8FC...	(random I know but what time is it for you? a...
2022-08-01 00:00:00	CB5A2...	D6316...	6127D...	E2161...	[]lck is pointing up*
2022-08-01 00:00:00	71932...	2BC4B...	57BB4...	3D652...	check your trade offers please
2022-08-01 00:00:00	DC52A...	E2C6B...	350E6...	1E7F9...	is xxx ur friend?
2022-08-01 00:00:00	219F8...	3B9D0...	92249...	88B03...	ew
2022-08-01 00:00:00	5F97E...	4F92C...	3EF1B....	24B3B...	whats trad ?
2022-08-01 00:00:00	827E3...	88D04...	436E6...	A8572...	BRO HUH
2022-08-01 00:00:00	86187...	F12B4...	F5B07...	680D1...	Hi
2022-08-01 00:00:00	8F5D2...	62310...	0B317...	981B4...	Hey
2022-08-01 00:00:00	84C44...	02CD1...	30E3B...	5BA36...	wenn du vernünftig...
2022-08-01 00:00:01	B4D9C...	15830...	62E55...	33AC3...	I have ski jacket
2022-08-01 00:00:01	CCA08...	35464...	8C68C...	AEA84...	hi
2022-08-01 00:00:01	7E39D...	99205...	D77C1...	8ED13...	its actually pretty active compared to other p...
2022-08-01 00:00:01	55563...	EF45E...	F9125...	7C420...	TY(:

4.2 Data Analysis

In the data analysis section, we have further examined the dataset to understand the structure on a deeper level. The main motivation was to map potential features which could be further utilized, in addition to get statistical and characteristic knowledge. To obtain the goal, statistical methods were applied to examine the dataset. An analysis of the language used in messages sent between users was also performed. A part of the analysis was to select and label a subset of the dataset to be used as training data for the classification model, which is described in section 4.4. To standardise the labeling, a method was utilized to increase the likelihood of labeling the same data independently of the person performing the labeling. The knowledge obtained during the data analysis was used as a product later when performing feature selection in section 4.3.

4.2.1 Statistics

To analyze the data statistically the Python library created for data analysis, Pandas [53], was used. To be able to use the Pandas library the original 80 .parquet-files were imported as data frames. Then, using Pandas, the data frames were sorted by date and combined into one dataset in a feather-file. The feather-file was then used to perform statistical computations.

The Pandas library has several predefined functions that can be used to gather statistical information from a dataset. By utilizing some of these functions it was discovered that the dataset consisted of a total of 28'428'425 messages. The total number of conversations, meaning the number of messages found between two users, were 1'386'761. There are 221'683 unique users participating in the game. 169'066 users initiated a message, and 208'037 users received a message. This indicates that approximately 1 in 4 users never sent a message.

When looking at the initiator values and applying the mean, median and max Pandas functions, it was found that the average user sent 168 messages. The median was 11 messages and the user initiating the most messages sent a total of 56'415 messages. Next, looking at the received values and applying the same functions, it was found that the average user received 137 messages. The median was 7 messages and the user receiving the most messages received a total of 56'825 messages. Looking at messages within the same context, hereby referred to as the same conversation between an initiator and a receiver, it was found that the average conversation length contained 20 messages. The median was 4 messages and the conversation containing the most messages consisted of a total of 109'847 messages.

Looking at the content of each message and the content for a conversation, it was found that the average message contained 4 words. The median was 3 words and the message containing most words contained a total of 500 words. Looking at the whole conversation it was found that the average conversation contained 91 words. The median was 15 words and the conversation containing the most words contained a total of 554'611 words. Figure 4.1 displays the histogram of words per message with bins of 10, whilst figure 4.2 displays the histogram of words per conversation with bins of 10.000.

Figure 4.1: Histogram of words per message

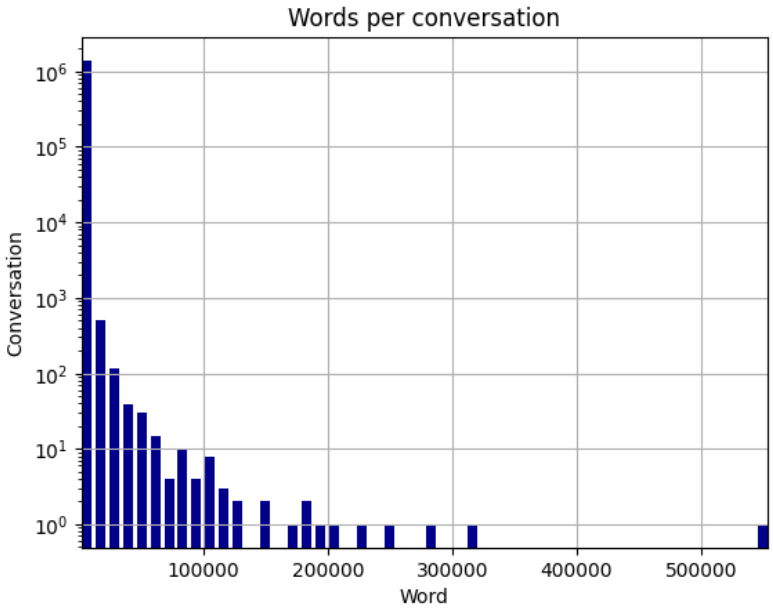


Figure 4.2: Histogram of words per conversation

By utilizing the *langdetect* [54] python library the most commonly used language were found to be English, with approximately 75% of the messages detected as English. Several other languages were also detected, but due to the number of misspellings, slang, to short messages etc. messages were wrongly detected as another language even when the message was written in English. When manually checking the messages it was found that even though many English messages were wrongly classified, it was rarely any non-English messages that were detected as anything else than English. Table 4.2 shows some messages, the detected language and the actual language of the message. By looking at some of the results it is evident that the accuracy of the *langdetect* library is dependent on the message length, the longer the message the more accurate is the language detection.

Table 4.2: Language detection examples

Detected Language	**Message**	**Actual Language**
English	i will stop immediately if you delete it	English
German	OMG BYE	English
Croatian	im ok i think	English
Tagalog	np o3o	English
Welch	oh wait nvm	English
English	If it does, you can always just cut it back to it's original length	English
Turkish	ingilizce okuyorum bölümü	Turkish
German	wenn du vernünftig bist dann klären wir das wenn nicht dann nicht	German

4.2.2 Labeling the Dataset

To test and train a classification model, it was necessary to create a labeled subset of the dataset. The original dataset only contained unlabeled data therefore a method to label users was created. Based on the method, it was decided that 100 users should be labeled, where 50 should be labeled as normal and 50 as abnormal. The requirements for a user to be labeled as normal or abnormal will be described shortly. To ensure the labeled user would be suited for training and testing the classification model, the user had to have sent at least 500 messages. The labeling method consisted of a manual check of the message content in addition to looking at the ego-graph simulations elaborated in section 4.2.3.

In the case of labeling users, several characteristics were used to determine if the user should be labeled as normal or abnormal. The text string *How old* was used to search for users who had sent that message to multiple users. The text string was chosen since it is likely that predators will ask for the age dur-

ing a grooming conversation. The content for a handful of users sending *How old* was manually studied to look for behavioral features, as describe in section 4.3.3. These features could be used to determine if the users were candidates to be labeled as abnormal. The ego-graphs for the abnormal candidates were then plotted. The ego-graph simulations showed similar characteristics, such as numbers of close neighbors, and the candidates were labeled as abnormal. The ego-graph characteristics, further described in section 4.2.3, were used in combination with manually checking for behavioral content features when labeling the rest of the abnormal users. Labeling normal users were done by manually choosing conversations and looking at the message content, the ego-graph simulations and the distribution of conversation length, which will be elaborated with ego-graph simulation. If the user interacted as most other users, clearly following the rules of the game etc. they were labeled as normal.

4.2.3 Ego-graph Simulation

While studying the dataset and calculating statistics, the extent of the information in the dataset was revealed. In addition, we needed to find a way to represent the data graphically. The ego-graph representation was appropriate for visualizing individual user's behavior. We based the ego-graph simulations on the work done by Aarekol [11] and Eng [12]. The ego-graph simulation utilizes *NetworkX* [55] which is a python library suited for creating, manipulating and studying network graphs. The ego-graph simulation was also necessary to calculate the node characteristics which will be elaborated in section 4.3.1.

The motivation behind the ego-graph simulation was to study how a user behaved, seen from a local perspective where all the neighbors are centred around the node of interest. Ego-graph simulation made it possible to compare how different users' behaved with its neighbors. The original dataset was first filtered to only contain messages between the ego-node and the one-hop neighbors, in addition to all messages sent between any of the one-hop neighbors of the ego-node. A directed ego-graph was then created based on the filtered messages, and the edges were weighted based on the numbers of messages sent between the initiator and the receiver. To observe how the graph evolved as messages were sent, a loop was created to process the first 5'000 messages. By running this simulation we were able to observe how the graph for each user were represented after 5'000 messages sent. Counting messages was an easier comparison than based on a given time period, which we tried initially. We found it difficult to compare users after a given time had evolved, instead of after a given numbers of messages sent, because it was not conclusive why there was a difference. By counting sent messages it was easier to determine the distribution of messages within a conversations. By using time as a parameter, it could indicate how the user chooses to behave with its neighbors, closer and fewer connections, or multiple connections with fewer messages.

In figure 4.3 the ego-graph for a normal user is plotted with the first 5'000

messages the user is participating in, meaning sent or received by the ego-node, or between any of the one-hop neighbors. The center node marked in red is the ego-node which in this example represent a normal user. All other nodes around the ego-node are one-hop neighbors. Two arrows between two nodes represents both sent and received messages, the arrow is pointing in the direction of the edge from the initiator to the receiver, and the color coding is mapped with the weight of each edge. The color mapping is; green, blue and purple, which is mapped to three bins based on numbers of messages in this given order: 1-3, 4-15, 16+ messages. Nodes which are closer located to the red ego-node has edges with higher weight value than the nodes further away from the ego-node. Note that this is not an exact calculation, but a tendency which *NetworkX* used when plotting.

The particular user in figure 4.3 was categorized as normal based on both the content of the messages and the graph-representation. The figure only shows the first 5'000 messages within the ego-nodes graph. The majority of messages are sent between few of the closest neighbors, as well as between some of the one-hop neighbors. First, we inspected the messages sent between the user and some random neighbors to get an impression of the user. As it appeared to be a normal user we plotted the ego-graph and compared it visually to other users who had also been classified manually as normal users.

Some of the similarities we found in the group of normal users was the observation that they often had the same amount of neighbors which shared edges with high weight score as users they shared edges with lower weight score. In figure 4.3 we can see that there is an even distribution of neighbors both close to the ego-node and in the outer circle where all edges are green which means only 1-3 messages has been exchanged per edge. It can also be observed that most of the neighbors have responded to the ego-node, or at least there is a two-way communication. There are also multiple one-hop neighbors which interact with other one-hop neighbors.

In figure 4.4 we can see the same normal user after a period of three months. The normal user has a total of 266 neighbors and within the graph 74 403 messages have been sent, where 7'670 are sent only between the ego-node and neighbours. Meaning 66'793 of the messages are only sent between one-hop neighbors of the ego-node. However, the tendency is much the same, many of the shared edges between the neighbors and the ego-node has a high weight score, since many of the neighbors are closely located to the ego-node.

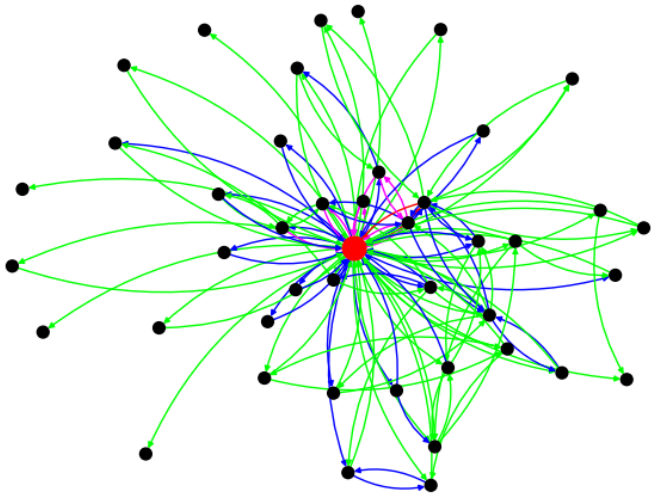


Figure 4.3: Normal user after 5000 messages

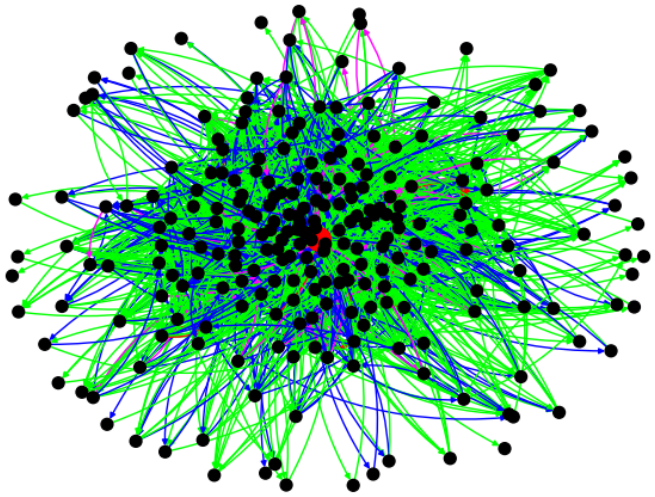


Figure 4.4: Normal user - All messages

Figure 4.5 is the ego-graph simulation of a user who has been manually classified as an abnormal user after a message inspection and ego-graph simulation. This graph contains the first 5'000 messages which the ego-node has participated in. When comparing the normal and abnormal user it is obvious that after the first 5'000 messages, the abnormal user has established more connections than the normal user. This implies that an abnormal user's conversation on average has fewer messages. It is difficult to interpret by the figure, but it is also possible to see that multiple nodes have not responded to the abnormal user. Whether this is because the receiver does not respond at all, or if the abnormal user is simply reaching out to several users simultaneously and the receiver has not yet responded, needs to be verified in the overall communication. The abnormal user also has neighbors which are interconnected with other one-hop neighbors. However, there are many more nodes located further away from the ego-node, than for the normal user where nodes tend to cluster closer to the ego-node.

In figure 4.6 the abnormal user graph after three months are visualized. The user has a total of 597 neighbors and within the graph 25'068 messages are sent. 7'729 of the total messages are sent between the ego-node and a neighbor. 17'339 of the messages are sent only between one-hope neighbors of the ego-node. When we zoomed in on the figure we could still see that there are multiple nodes which have received messages without sending any. The distribution of neighbors located farthest away from the ego-node is still high, meaning there are fewer messages sent between the one-hop neighbors and the ego-node. An observation from this and other graphs of abnormal users shows the difference between the amount of communication between the one-hop neighbor. In figure 4.6 we can see that multiple one-hop neighbors are communicating.

Comparing these normal and abnormal users, the graphs might seem skewed when it comes to the total messages sent or received. Both users sent or received approximately 7'000 messages in total. It might seem like the abnormal user has sent and received more messages than the normal user. The abnormal user has many neighbors, but most conversations consist of few messages. The normal user, on the other hand, has fewer neighbors, but most conversations consist of many messages.

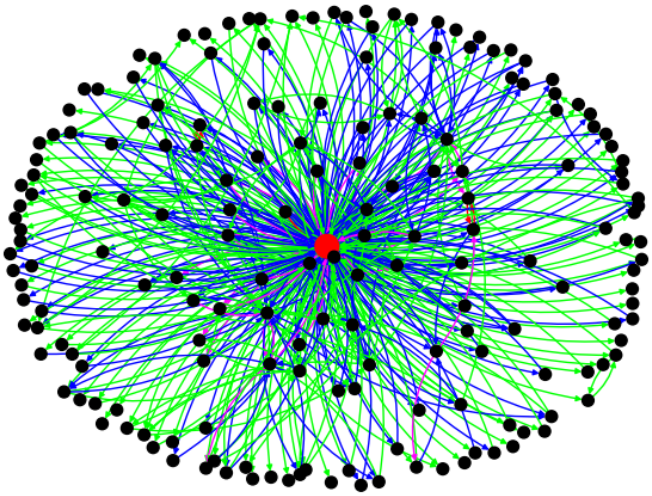


Figure 4.5: Abnormal user after 5000 messages

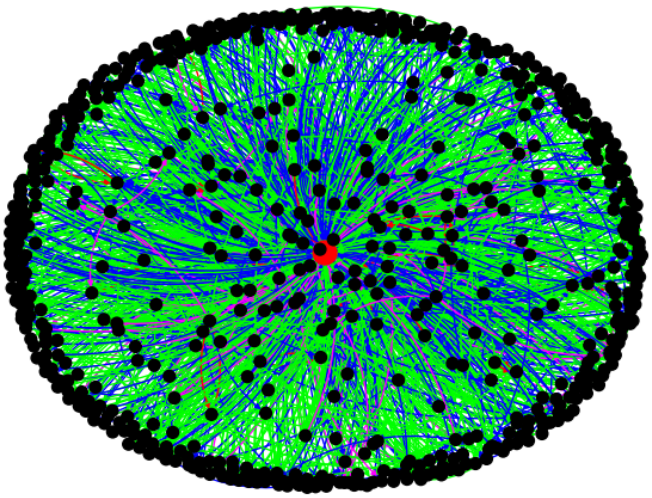


Figure 4.6: Abnormal user - All messages

While studying the ego-graph simulation it was necessary to look at the conversation length distribution for normal and abnormal users. During the ego-graph simulation it was graphically observed that normal users tend to have longer conversations than abnormal users. In multiple cases abnormal users had many short conversations with one to three messages. The distribution of conversation length was therefore plotted. Figure 4.7 is the conversation length distribution for a user which was labeled as a normal user. The normal user has over 25 conversations where the message length is less than 25 messages. However, there are multiple conversations with lengths greater than 50 messages. Looking at an abnormal user, in figure 4.8, the user has an even higher overweight of shorter conversations under 50 messages, compared to longer conversation with more than 50 messages. The conversation count for the two graphs are not even, and the abnormal user also has conversations with message lengths over 50. These were reoccurring characteristics when investigating multiple conversation length distribution graphs for normal and abnormal users.

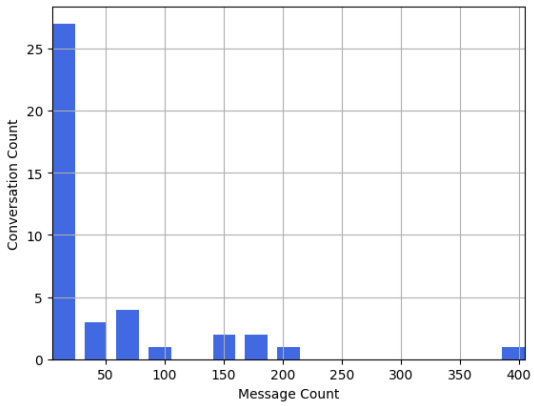


Figure 4.7: Conversation length distribution - A normal users

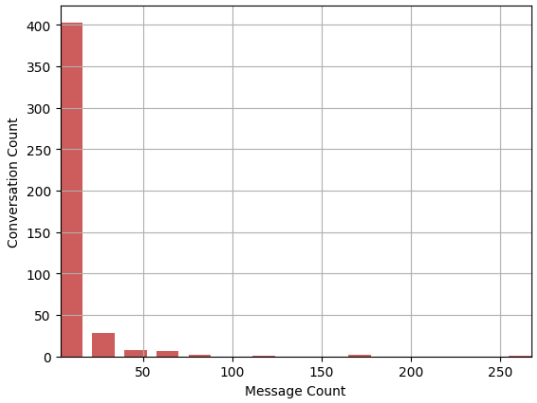


Figure 4.8: Conversation length distribution - An abnormal user

The conversation length distribution was also plotted for all 50 normal and 50 abnormal users. In figure 4.9 the total conversation distribution for all normal users are illustrated. Notice that the y-axis is a logarithmic scale. Figure 4.10 illustrates the total conversation distribution for all abnormal users. For the normal users there also are outliers with message counts up to 14‘000 which are removed from the graph. When comparing the two graphs there is a significant difference in conversation lengths, keeping in mind that the scale for conversations are logarithmic. Normal users tend to have longer conversations than abnormal users.

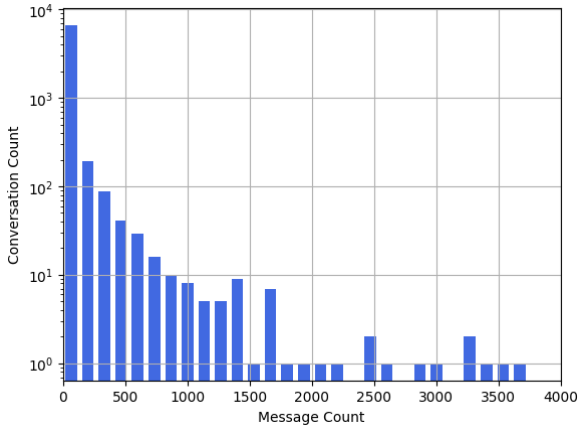


Figure 4.9: Conversation length distribution - All normal users

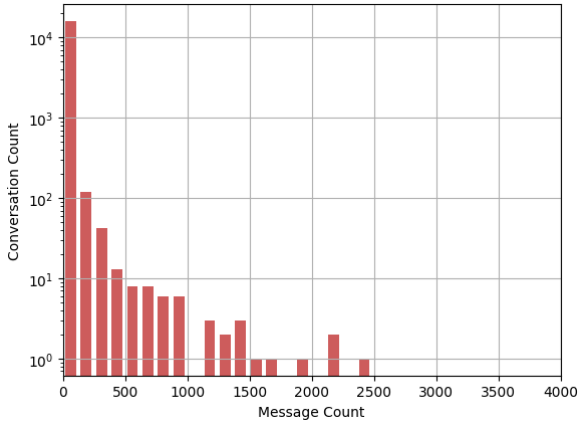


Figure 4.10: Conversation length distribution - All abnormal users

4.3 Feature Selection

Feature selection was the process of transforming the data and knowledge from the data analysis phase and turn it into a product for later classification. There were several characteristics discovered in the data analysis section, and it was therefore necessary to categorize what had been observed into three main categories. The first category was characteristics which described the node, when studying individual users. This was mainly utilized through graph theory and based on the ego-graph simulation. The second category contained characteristics about the context, which is how the user behaves in a conversation. Through the ego-graph simulation and reading multiple conversations manually, it was observed different patterns when a user interacted with its neighbors. The last category deals with all the content specific behavior. This includes how the user express itself through text messages in a conversation, from language and grammar, to obfuscating the language and still being able to express their opinions. In the following sections all the features will be elaborated through node, context and content specific characteristics. All the features are presented together in section 4.3.4.

4.3.1 Node Specific Characteristics

During the feature selection features were categorized into subsets. Features based on node specific characteristics will be elaborated in this section. A total of 21 features were created based on the social graph network accumulated by all the chat messages in the dataset. Node specific features were limited to studying the characteristics of a node in a graph network. It might be interesting to look at text characteristics within messages sent by the node, this will be covered in section 4.3.3. These features are based on both observations in the ego-graph simulation, and by Aarekol's [11] and Eng's [12] work. It seems obvious that the interaction between the users is crucial to capture, which will been further investigated in this section. Using the graph utilities which comes with *NetworkX* [55], as well as the ego-graph simulation, we were able to both discover and create node specific features.

For the first features, we looked at how many conversations and messages the user started and exchanged with its neighbors. From the ego-graph simulation we observed differences between an abnormal and a normal user regarding the number of established conversations and the number of messages within the conversations. The first feature is the degree of the node which is annotated as $Deg(v)$ where v is the node. The node degree is the total number of conversations the node v is a part of, where at least one message is sent or received. In addition, in-degree and out-degree are two other features based on degree which highlights the user's activity from another perspective. In-degree is annotated as $inDeg(v)$ and is the number of conversations where at least one message is sent from a neighbor to node v. The out-degree on the other hand is the number of conversations where at least one message is sent from the node v to a neighbor.

Out-degree is annotated as $outDeg(v)$.

Another interesting feature is the duration of a conversation. Users will probably have longer conversations with users they find interesting or want to have a connection with. This was addressed looking at the weighted degree of the node, as the weight in this scenario is the number of messages sent within a conversation. Weighted degree is annotated as $Deg_w(v)$ and is the total number of messages both received and sent to the neighbors. Weighted in-degree, $inDeg_w(v)$, is the total number of messages which node v has received. Weighted out-degree, $outDeg_w(v)$ is the total number of messages node v has sent to its neighbors. The two features *Ratio of outgoing conversation* and *Ratio of sent messages* was introduced by Eng [12] in her thesis to capture the change in distribution over time. *Ratio of outgoing conversation* is expressed in equation 4.1 and *Ratio of sent messages* in equation 4.2.

$$\frac{outDeg(v)}{deg(v)} \tag{4.1}$$

$$\frac{outDeg_w(v)}{deg_w(v)} \tag{4.2}$$

The feature *Clustering Coefficient* is annotated as $CC(v)$ and is calculated with the equation found in 2.1 which can be presented with the new annotations seen in equation 4.3. Since this is a directed graph, $N(v)$ is the total numbers of directed triangles, which is a set of three nodes connected by directed edges through node v. The *Clustering Coefficient* can possibly tell in what degree the users are interconnected. Interconnections are more likely to happen when a group of users are talking with friends rather than strangers. Hence, it is less likely that strangers have common friends. *Numbers of initiated conversations* is a feature measuring how many conversations the node has initiated, annotated as $conDeg_{in}(v)$. Another feature which could indicate the activity of the user is *Initiated messages last 10 min*, annotated $Deg_{w,10min}(v)$. Counting numbers of messages initiated by the users for the last 10 minutes will highlight a user initiating multiple messages with the same or multiple users.

$$CC(v) = \frac{N(v)}{Deg(v)(Deg(v)-1)} \tag{4.3}$$

The last set of features based on node specific characteristics, are looking at the distribution length of conversations. Where one set of features includes all the messages in a conversation, and the other set only counts messages which are sent by the node. Both sets of features are divided into bins of messages. As most of the node specific features in this thesis is based on Aarekol's [11] and Eng's [12] features, we decided to use the same bins to easier compare the results. The first set is counting the numbers of conversations where all the weighted messages are within the given bin, $Deg_{con,w}(v)$ where w is the bin. The second set of features also sum the numbers of conversations per message $outDeg_{con,w}(v)$, but only for

messages sent from the node. There are 5 bins: 1) Only one message sent, 2) Between two and five messages sent, 3) Between 6 and 15 messages sent, 4) Between 16 and 32 messages sent, 5) More than 32 messages sent. These are the last ten out of the 21 node specific features. In table 4.4, all node specific features are listed, including the equations and a brief description.

4.3.2 Context Specific Characteristics

The second subset of features are the context specific features. Node specific features originates from a graph theoretical aspect of a node, its graphs and its neighbors. Context specific features, on the other hand, capture the characteristics of the conversation. Although a conversation is all about the dynamic between two users, the features will always be calculated from one users point of view. Many of the context specific features are inspired by Eng [12], although the implementation is only based on the concepts as the exact implementation is unknown. There is a total of five context specific features, the implementation of them is described in section 4.4.1.

Response time is the first context specific feature and measures the time, starting with a neighbor sending a message to a user, and stopping after the user has responded to the message. Response time is an interesting measure as there can be different reasons why the response time can vary. For example, the user can be highly involved in the conversation and eager to answer, or the user can wait due to other priorities. As it is possible for a user to send multiple consecutive messages, the response time will span from the first message sent from the neighbor to the user until the user responds, regardless of how many consecutive messages the neighbor sends.

There are two other features which are also relying on consecutive messages. Eng [12] grouped consecutive messages sent by the same user and checked the number of times there was a switch between the sender and the receiver, this feature is called *Number of context switches*. After the messages are grouped in blocks based on consecutive messages it was also possible to count the number of messages within a block. *Variance in consecutive messages* is a feature calculated by the variance between the last two blocks sent by the two users participating in the conversation. Eng's idea was that the variance could show how the messages in a conversation are unevenly distributed between the users. *Activety Time* is a measure of the time between the last message and the previous one, regardless of who the sender is. This feature indicates how rapidly messages are sent in the conversation. The last feature is *Message Length* and is a measure on how long the messages are. The feature is measured in numbers of characters and can indicate how much effort the sender puts into the conversation in form of text. In table 4.5, all context specific features are listed with a brief description.

4.3.3 Content Specific Characteristics

The content in the messages exchanged in the game have several characteristics that can be used to extract features which can be used by machine learning algorithms to distinguish between normal and abnormal users. Below are some characteristics found when looking through the dataset.

Language: Based on the findings in section 4.2.1 most of the users communicate in English (>75%). But there are also other languages present such as German, Swedish, Finnish, French etc.

Obfuscation: Due to the strict guidelines, users tend to obfuscate words or phrases so that they are able to send illegal messages without getting caught or banned from the platform. This obfuscation can include changing some letters with special characters, writing one letter per line, changing the arrangement of letters etc.

Misspellings: The way the users chat in the game is different from most forms of written text resulting in the extremely high rate of misspellings, grammatical errors, and slang. Although, this is common to chat data, it may be even more prevalent in this game due to the user's age being mainly between 8-15 years old.

Role-play: Due to the nature of the game the users might talk about being single, dating and some users are also in relationships either with their in-life boyfriend/girlfriend, online boyfriend/girlfriend or with a predator. Either way, the users seem to frequently engage in role-play where they, for example, pretend to be dad, mom, brother, or sister.

Rule breaking: The game has an extensive list of alert words and blacklist words which is used to filter out inappropriate behavior. If the initiator uses any of those words they will not be sent to the receiver, but it is still stored in the dataset as a regular part of the conversation.

Content Features

Based on the characteristics described above several behavioural features were extracted. The behavioral features are based on several characteristics regarding the users behavior such as how the users attempt to obfuscate the use of rule breaking words or expressions.

The background for choosing the following behavioral features were the observations made of the conversations in the dataset about how the users tried to avoid being caught and banished from the game by breaking the rules. Most of the behavior involved trying to avoid using words found in the alert- and blacklist. Due to not having access to this alert- and blacklist an effort was made to get an understanding of what was allowed to write and what was not and how the users circumvented those rules by manually looking at the messages. The following features were extracted.

Profanity: A user writes a bad word such as "fuck", "sex", "hell" etc. Most of those words are an alert or blacklist word and are only visible in the dataset, not to the users. All words found in a profanity dataset [56] are counted.

One letter lines: A user writes an alert or blacklist word by typing and sending one letter at a time until the partial or full alert or blacklist word has been sent, seen in table 4.3. All single letter messages are counted.

Spaces: A user writes an alert or blacklist word by adding blank spaces in-between the letters in the word, such as “F u c k”, “s e x”, “h e l l” etc. All single letters divided by blank spaces and repeated at least 3 times are counted.

Special characters: A user writes an alert or blacklist word by replacing letters with special characters or numbers, such as “F#ck”, “s*x”, “h3||” etc. All words containing letters and special characters or numbers are counted.

Consecutive letters: A user writes an alert or blacklist word with consecutive letters within a word, such as “seeeex”, fuuuuck”, “heeeell”. All words containing more than two consecutive letters are counted.

Misspellings: A user writes an alert or blacklist word by misspelling it. All misspelled words are counted.

Table 4.3: One letter per line in chat message

dateUtc	messageId	context	initiator	receiver	content
2022-08-10 00:17:34.252	86187...	B3BFE...	62310...	5BA36...	s
2022-08-10 00:17:34.746	28674...	B3BFE...	62310...	5BA36...	n
2022-08-10 00:17:35.191	39625...	B3BFE...	62310...	5BA36...	a
2022-08-10 00:17:35.608	72612...	B3BFE...	62310...	5BA36...	p

In table 4.6, all content specific features are listed with a brief description.

4.3.4 Accumulated Features

In this section all features selected for node, context and content are accumulated. The node specific features are located in table 4.4, the context specific features are located in table 4.5, and content specific features are located in table 4.6.

Table 4.4: Node Specific Features

Feature	Expression	Description
Degree	$Deg(v)$	Total number of conversations node v is a part of
Weighted Degree	$Deg_w(v)$	Total number of messages sent to or from node v
In-Degree	$inDeg(v)$	Number of conversations where node v has at least received one message
Weighted In-Degree	$inDeg_w(v)$	Number of messages sent to node v
Out-Degree	$outDeg(v)$	Number of conversations where node v has at least sent one message
Continued on next page		

Table 4.4: Node Specific Features - Continued

Feature	Expression	Description
Weighted Out-Degree	$outDeg_w(v)$	Number of messages node v has sent
Ratio of outgoing conversation	$\frac{outDeg(v)}{deg(v)}$	Ratio between the number of outgoing conversations and the total number of conversation
Ratio of sent messages	$\frac{outDeg_w(v)}{deg_w(v)}$	Ratio between the number of outgoing messages and the total number of messages received or sent
Clustering Coefficient	$\frac{2N(v)Deg(v)}{Deg(v)-1)}$	Ratio of interconnection between neighbors
Numbers of initiated conversations	$conDeg_{in}(v)$	Numbers of conversations initiated by node v
Messages last 10 minutes	$Deg_{w,10min}(v)$	Numbers of initiated messages by node v the last 10 minutes
Number of conversations with 1 message	$Deg_{con,1}(v)$	Conversations with only 1 message
Number of conversations with 2 to 5 messages	$Deg_{con,2-5}(v)$	Number of conversations with 2 to 5 messages
Number of conversations with 6 to 15 messages	$Deg_{con,6-15}(v)$	Number of conversations with 6 to 15 messages
Number of conversations with 16 to 32 messages	$Deg_{con,16-32}(v)$	Number of conversations with 16 to 32 messages
Number of conversations with more than 32 messages	$Deg_{con,32>}(v)$	Number of conversations with more than 32 messages
Out conversations with 1 message	$outDeg_{con,1}(v)$	Conversations where node v only has sent 1 message
Out conversations with 2 to 5 messages	$outDeg_{con,2-5}(v)$	Number of conversations where node v has sent 2 to 5 messages
Out conversations with 6 to 15 messages	$outDeg_{con,6-15}(v)$	Number of conversations where node v has sent 6 to 15 messages
Out conversations with 16 to 32 messages	$outDeg_{con,16-32}(v)$	Number of conversations where node v has sent 16 to 32 messages
Out conversations with more than 32 messages	$outDeg_{con,32>}(v)$	Number of conversations where node v has sent more than 32 messages

Table 4.5: Context Specific Features

Feature	Description
Response time	The time between initiated message and response
Number of context switches	Number of times the two user has changed the role as sender and receiver
Variance in consecutive messages	How the last two blocks of messages are distributed over the two users
Last activety time	Time between last two messages in the conversation
Message length	The message length by counting characters

Table 4.6: Content Specific Features

Feature	Description
Profanity	The number of times a word found in a profanity dictionary is used
One letter lines	The number of times a single letter message is sent
Spaces	The number of times a space is added within a word
Special characters	The number of times special characters or numbers are added within a word
Consecutive letters	The number of times a word contains more than two consecutive letters
Misspellings	The number of times a word is misspelled

4.4 Implementation

This section covers the implementation of the methods applied in this thesis. The implementation was coded in Azure Machine Learning Studio within a notebook based on the computing platform Jupyter Notebook. First, the methods utilized in the data analysis phase will be covered, explaining the technical implementation. Second, the data transformation process which was initiated after selecting the first set of features will be described. The classification model will also be explained in detail before the training and testing implementation will be elaborated. A dynamic detection mechanism used to classify normal and abnormal users will be described.

4.4.1 Data Transformation

To perform initial data analysis, an ego-graph simulation was created to observe individual users within the closest circle of neighbors. The dataset was sorted by date and was filtered to only contain messages sent to and from the ego-node, and between any one-hop neighbors. The dataset was stored in a dataframe. An empty directed graph was created using *NetworkX* [55]. By iterating over all the filtered messages, the graph was updated for each sent message. It was crucial that the filtered dataset contained messages between the one-hop neighbors, to be able to calculate all desired features. A for-loop was created to iterate over all

messages, and an if-statement checked if the current message was sent by the ego-node. If the ego-node sent the message, a function was called to calculate all node specific features for that message. For each iteration, the node specific features were calculated by applying functions from the library *NetworkX* on the directed graph. Calculating all features for each iteration was resource demanding, but was necessary to capture how the node evolved after each sent message. Features were then stored to a separate file for each user. The features are listed in table 4.4. The labeled users will be addressed as users hereby on, as the remaining users will be called neighbors.

For context features it was necessary to modify the dataset. It was filtered to only contain whole conversations where the users were participating. For each user, a for-loop was created to iterate over all sent and received messages. An if-statement checked if the user was the initiator which then would trigger a calculation of the context features. It was important to calculate the features in the same way as the node features to be able to merge the feature sets later. Both feature sets could have been calculated over the same loop, however, separating them led to less data overhead when calculating context features. The labeled users had accumulated a total of 330'066 messages together which they had sent, an average of 3'300 messages each. In addition, messages received also had to be iterated over as the context features relied on them. The large dataset was therefore time and resource demanding to process. An additional dataframe was created to store iterated messages to optimize the calculation. The other alternative would be to filter the dataset by date which is a less efficient method, due to the search mechanisms for dataframes. To calculate context features, functions and packages from *Panda* [53] and *Numpy* [57] were utilized. Features for each user was stored in a separate file. The context specific features are listed in table 4.5.

When transforming content features, an additional modification to the original dataset was necessary. The content features were limited to only analyzing the messages sent by each user. Therefor, the dataset was first filtered to only contain the 330'066 messages sent. The filtered dataset was then grouped by user so it was possible to iterate over each user and calculate all features for one user at a time. Three methods were applied to calculate features, 1) *Alt-Profanity-checker* [56], 2) *PySpellChecker* [58], 3) regular expressions. The *Alt-Profanity-checker* was implemented and used to give each message a score in the interval 0 to 1. Higher scores indicates a higher probability and more extreme use of profanity. *PySpellChecker* is a python function used to detect misspelled words. All misspelled words in a message were counted and used as a feature. The remaining method utilized regular expressions to match or count occurrences of the expressions. After all features were generated for all messages in the dataset, they were stored in a separate file per user. All content features are listed in table 4.6.

As all messages for content features were calculated individually, it was required to calculate the cumulative value for each message. A dataframe was created to store iterated messages for each user, starting with the first message. For

each iteration, messages were grouped by conversation id and the mean of all iterated messages were calculated and stored in the final feature set. The mean value was chosen as aggregation method since it has proven to give promising results in previous studies [12]. Creating multiple feature sets with different aggregation methods would increase the number of test cases dramatically. It is important to clarify that the final features for all the three different feature sets were cumulatively calculated. Meaning that for each individual message, the feature vector was calculated in relation to all the previous messages.

Next, normalization of the features was performed to prepare each feature set for the SVM model. The normalization process transformed the data points into numerical values between 0 and 1. For all three feature sets, every feature vector for each user was added to a single dataframe. All the data points within a feature had to be normalized in relation to each other. In 4.4, the equation for ordinary min-max normalization is presented. x' is the normalized data point and x is the original value. $min(x)$ and $max(x)$ is the minimum and maximum value occurring in the dataset for the specific feature. All normalized features were then stored in a new feather-file for each feature category.

$$x' = \frac{x - min(x)}{max(x) - min(x)} \tag{4.4}$$

The final process of data transformation was to combine the different categories of features. The process included importing the different feature sets into new dataframes. Since the dataframe contained *None* or *Nan* values, a function was applied to replace these values with zero. The label for each user was also added to the dataframe, so the dataset later could be split into a training and testing set. It was decided to prepare the data to test all combinations of the three categories. The three feature sets were tested using 7 combinations. All combinations are listed in table 4.7. In addition to the extracted features, the dataset included the following columns; *initiator*, *messageId* and *label*.

Table 4.7: SVM - Test Combinations

Test Case	**Feature Category Combination**
1	Node features
2	Content Features
3	Context Features
4	Node and Content Features
5	Node and Context Features
6	Content and Context Features
7	Node, Content and Context Features

4.4.2 Classification Model

In this thesis we will try to distinguish between abnormal and normal users which is a classifiacation problem. SVM was selected as the classification model for this thesis. Through the literature review, elaborated in chapter 3, SVM was identified as a promising machine learning model for binary classification. SVM was also chosen due to its efficiency on binary classification problems and works well on large datasets with multiple features [59]. Due to these properties, SVM was implemented with a linear kernel using a *Scikit Learn module* [60] to solve the classification problem.

The regularization parameter, annotated as C, is used to tune both the margin and the classification error. A larger margin is preferable to better separate the two classes, but a large margin can lead to overfitting. A trade-off between the two was chosen where the regularization parameter was set to $C = 1$. The intention of the SVM model was to give each message, which the user sends, a probability score so it could be monitored over time. Probability scores were given by SVM by setting the $probability$ parameter to true. *x_train* and *y_train* are the feature vectors and the corresponding label. *predict_proba*() predicts the class for testing vectors *x_test* and return a float value between 0 and 1.

For 5-fold cross validation, the SVM model was trained and tested over 5 different iterations. All 5 folds had the same amount of normal and abnormal users. For each iteration, one fold was selected as a test fold, and the remaining folds were the subset used to train the machine learning model. For the second iteration the next fold was used for testing, and this was repeated until all folds were iterated over. This is illustrated in figure 2.4. The SVM model was applied for each fold, meaning a new SVM model was created for each iteration. Probability scores were calculated for each message to later dynamically detect abnormal and normal users. A confusion matrix was created for each fold, in addition to evaluation metrics such as Accuracy, Recall, Precision and F-scores. After all iterations, a new confusion matrix was created for the whole feature set by calculating the average of all 5 folds. Evaluation metrics were calculated again, this time for the whole feature set. Results from the SVM model and the 5-fold cross validation can be found in section 5.1.

4.4.3 Dynamic Detection Mechanism

All 330'066 messages were processed through the SVM model in 7 different combinations. The three combinations which achieved the highest F_1-scores were used with the dynamic detection mechanism. These three combinations were; 1) node features, 2) node and context features and 3) node, context and content features. The results using these combinations are found in section 5.1. The datasets for the three combinations included the probability score for each message calculated by the SVM model. The intention behind a dynamic detection mechanism was to create a function which dynamically could determine if the user was normal or abnormal. A dynamic threshold implies that there is no hard limit, which enabled

us to make individual classifications, and hence detect normal and abnormal users faster. This approach was inspired by Eng [12], and contains four different parameters; 1) *window size*, 2) *decision threshold*, 3) *stabilization threshold* and 4) *start message*.

A *window size* was set including all probability scores for the messages inside the chosen window. The standard deviation was calculated for all the messages inside the window, and a *stabilization threshold* was used to decide if the messages were stable enough to determine a class. A *stabilization threshold* ensures that the probability scores within the window have low variations, meaning the users probability scores had stabilized and a more accurate classification could be made. The probability score would be compared to the *decision threshold*. If the probability score was below the *decision threshold* it meant the user was classified as a normal user, and values above the *decision threshold* meant it was classified as an abnormal user. The fourth parameter was *start message* and indicated how many messages that should be skipped before the detecting mechanism started. The *start message* parameter was introduced because it was assumed that the probability score could oscillate more in the beginning.

When implementing the dynamic detection mechanism all labeled users were iterated over one by one. If a *start message* value was set, the first x messages would be skipped for all users. For each user, a window was created and contained a sequence of messages which were the same size as the *window size*. The standard deviation of all messages within the window were calculated and the result was compared with the *stabilization threshold*. If the standard deviation was above the *stabilization threshold* the loop would break and the function would continue with the next window. However, if the standard deviation was below the *stabilization threshold*, the current probability score was compared to the *decision threshold*. Then the the user was classified based on the *decision threshold*. When a user was classified by the function it would store the user id, its label, number of messages sent before the classification, and the actual class of the user. After all users were classified, a confusion matrix was created and evaluation metrics were calculated. The results are found in section 5.2.

As mentioned, the dynamic detection mechanism was tested on three different datasets. These were the three combinations of feature sets with the most promising results from the SVM model, when comparing F_1-scores. In addition, it was necessary to evaluate which values should be used for *window size*, *decision threshold*, *stabilization threshold* and *start message*. Numerous iterations using different combinations of the four parameters were first tested on the dataset with node features. The initial evaluation was used to tune parameter values to obtain a baseline for all the three datasets. The *window size* was initially tested over 10, 20, 30, 40 and 50 messages. The *decision threshold* was tested with probability scores of 0.25, 0.5 and 0.75. The *stabilization threshold* was tested with standard deviations of 0.01, 0.02 and 0.04. The *start message* was tested with 0, 50, 100, 150, 200 and 250 messages.

Chapter 5

Results

In this chapter the results obtained after implementing the methodology described in chapter 4 are presented. First, the results achieved when utilizing the SVM model described in section 4.4.2 will be covered in section 5.1. The model was tested with 7 different feature set combinations as elaborated in section 4.4.1. Next, the results obtained when implementing the detection mechanism will be presented in section 5.2. The dynamic detection mechanism was utilized for the three feature set combinations which achieved the best results after the SVM classification. In addition, all the combinations of variables used within the detection mechanism, described in section 4.4.3, which were calculated for the three feature set combinations will be described. Reflections regarding the results described in this chapter will be discussed in chapter 6.

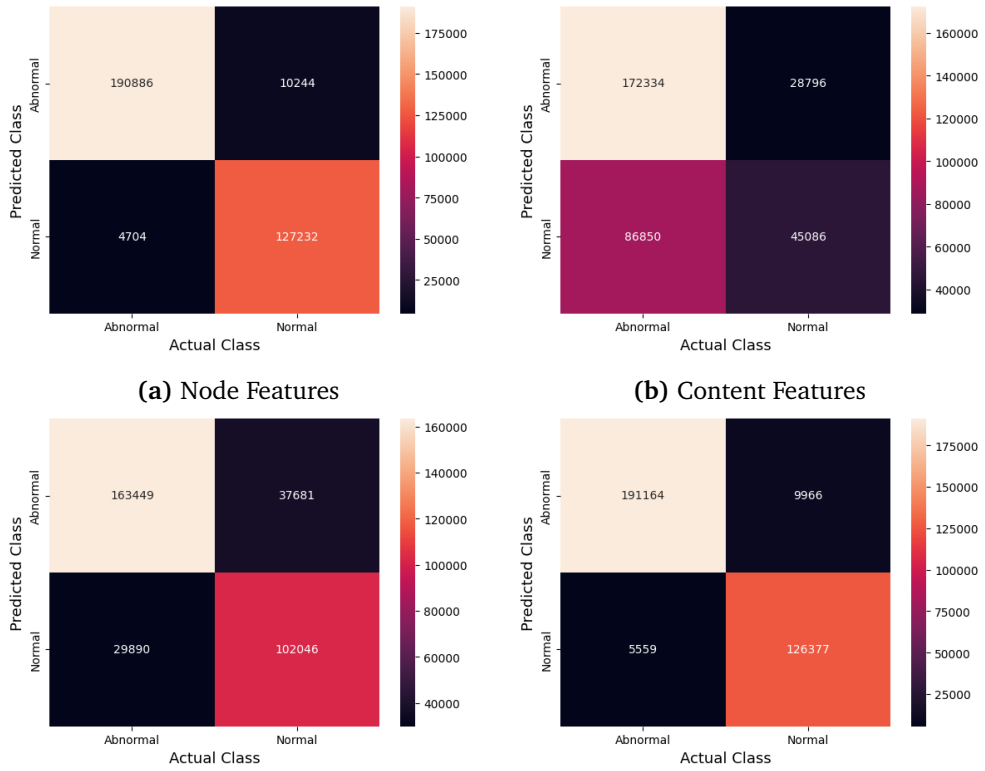
5.1 SVM Classification Model

This section presents the results obtained when analyzing the feature set combinations using the SVM model with linear kernel and default C parameter ($C = 1$) as elaborated in section 4.4.2. As described in section 4.4.1, 7 combinations of feature sets were used by the SVM model. The combinations are listed in chapter 4, in table 4.7.

For each of the 7 feature set combinations the average was calculated across the five folds as described in section 4.4.1. These average scores were placed in a table to be able to compare their performance. The results are found in table 5.1, all the best individual scores are written in bold text. For better comparison to related and future work, the $F_{0.5}$- and F_2-scores are included in the table. Details regarding the results for all folds are found in appendix A. The average confusion matrix for all individual feature set combinations is depicted in figures 5.1 and 5.2.

Table 5.1: The average SVM results for all 7 feature set combinations

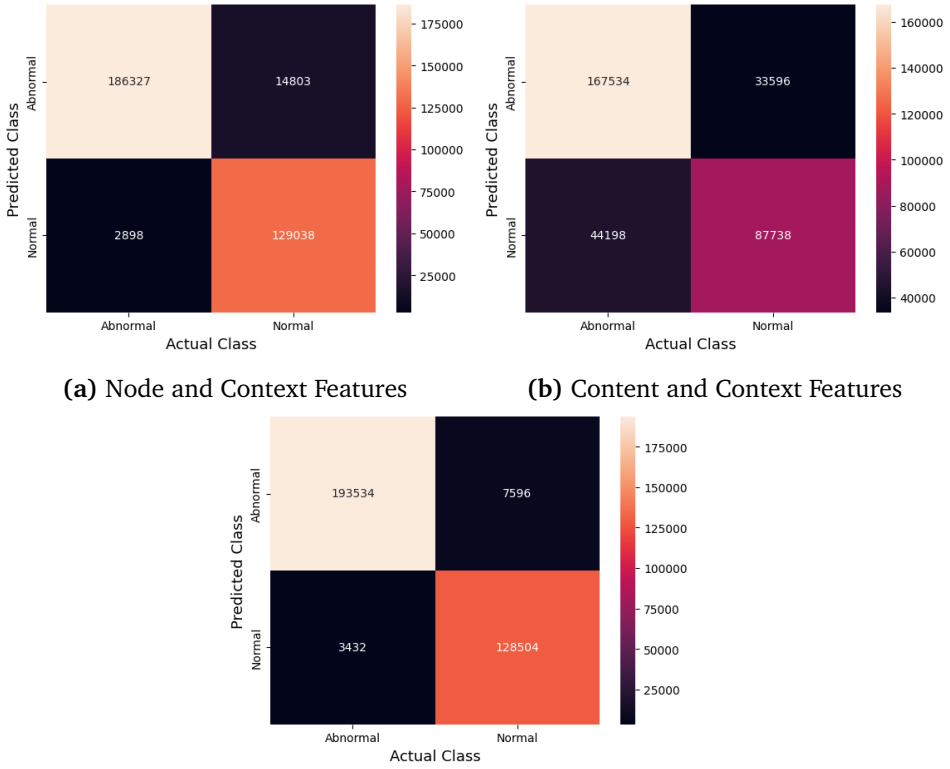
Feature set Combination	PR	RC	ACC	$F_{0.5}$	F_1	F_2
Node	0.914	0.962	0.944	0.921	0.935	0.950
Content	0.684	0.366	0.648	0.548	0.442	0.387
Context	0.728	0.813	0.788	0.739	0.761	0.789
Node and Content	0.913	0.955	0.941	0.919	0.931	0.944
Node and Context	0.908	**0.977**	0.946	0.920	0.940	0.961
Content and Context	0.731	0.721	0.756	0.711	0.700	0.707
Node, Content and Context	**0.934**	0.969	**0.959**	**0.941**	**0.951**	**0.962**



(a) Node Features **(b)** Content Features

(c) Context Features **(d)** Node and Content Features

Figure 5.1: Confusion matrix of aggregated SVM results for each feature set combination



(a) Node and Context Features

(b) Content and Context Features

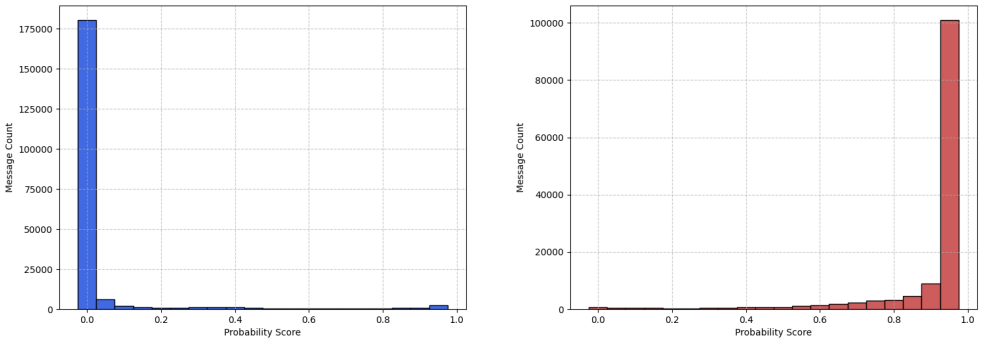
(c) Node, Content, and Context Features

Figure 5.2: Confusion matrix of aggregated SVM results for each feature set combination

After running all feature set combinations through the SVM model, the three combinations that performed best, based on their F_1-score, in ascending order, were;

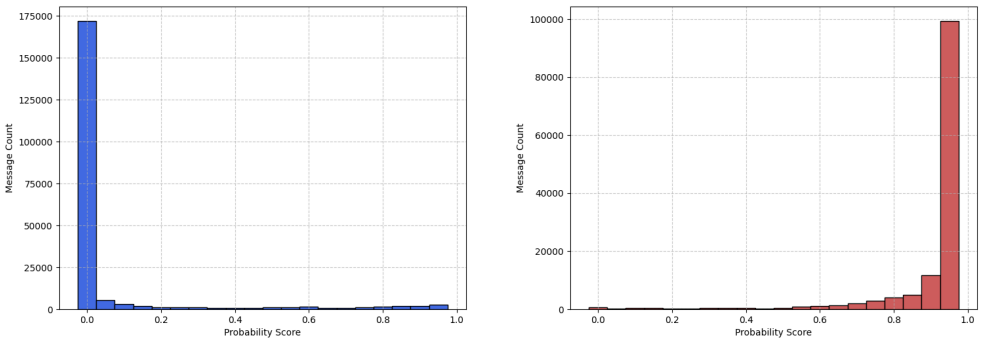
1. Node, Content, and Context
2. Node and Context
3. Node

The combination including all three feature sets, Node, Content and Context features, achieved the best overall results. This feature set achieved an F_1-score of 0.951, a Precision of 0.934, and an Accuracy of 0.959. The Recall-score was second best with a score of 0.969, the Node and Context feature set combination achieved the best Recall-score of 0.977. The probability distributions for normal and abnormal users for the best performing feature set combinations are depicted in figures 5.3, 5.4, and 5.5 respectively. The probability scores for all feature set combinations are listed in appendix A.



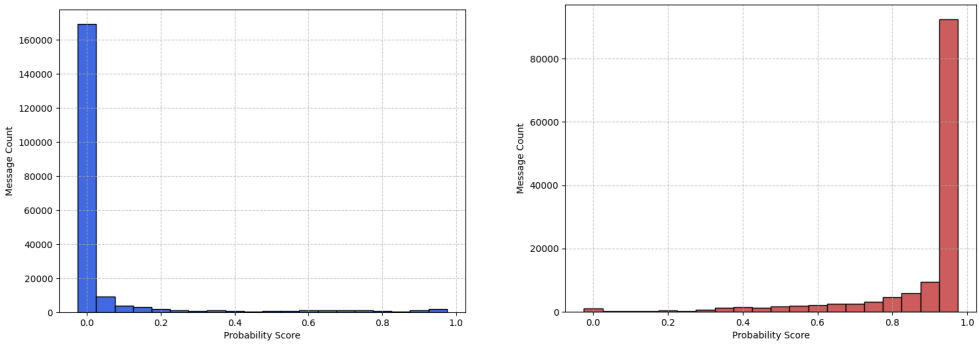
(a) Probability Score for Normal users **(b)** Probability Score for Abnormal users

Figure 5.3: Probability Score for Normal and Abnormal users - Node, Content, and Context



(a) Probability Score for Normal users **(b)** Probability Score for Abnormal users

Figure 5.4: Probability Score for Normal and Abnormal users - Node and Context



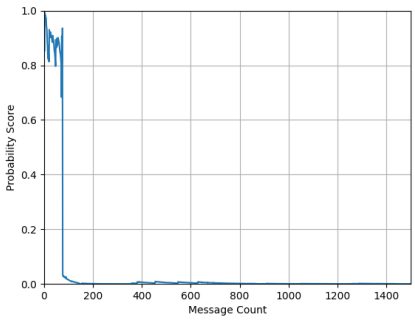
(a) Probability Score for Normal users **(b)** Probability Score for Abnormal users

Figure 5.5: Probability Score for Normal and Abnormal users - Node

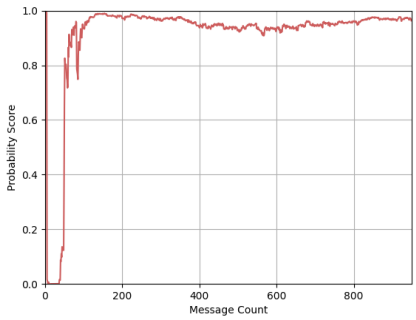
Probability timelines

A timeline for each user showing the probability score for every sent message were examined for the three feature set combinations that performed best with SVM. This was done to gain insight into the performance scores as the users sent more messages. These examinations were performed to identify patterns which could be utilized by the detection mechanism. A probability timeline for a user was created by plotting the probability calculated for all the messages sent by that user. A few examples of the same normal and abnormal probability timelines, for the top three feature set combinations, are shown in figure 5.6 and 5.7.

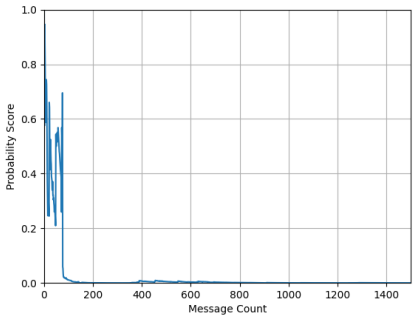
The probability timelines show that, for both normal and abnormal users, the probability scores oscillate at the beginning of their timeline. After approximately 150 messages, the majority of probability timelines stabilized around a probability score. Most of the labeled users showed these tendencies with some exceptions. All individual timelines for the three best performing feature set combinations can be found in appendix B.



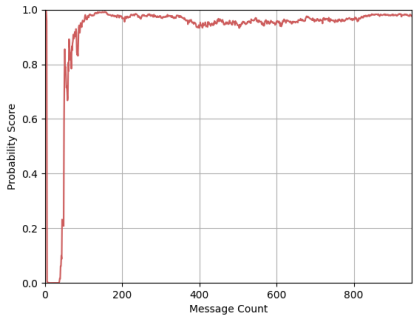
(a) Normal user - Node and Context



(b) Abnormal user - Node and Context

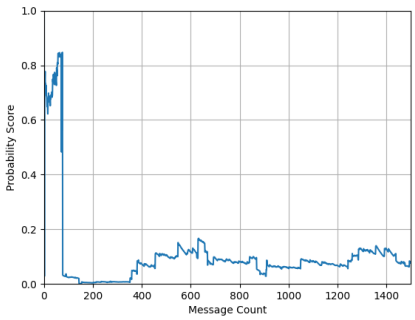


(c) Normal user - Node, Content, and Context

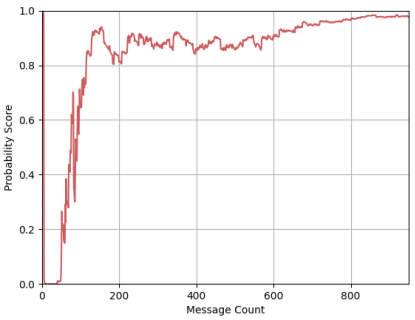


(d) Abnormal user - Node, Content, and Context

Figure 5.6: Probability timelines for normal and abnormal users



(e) Normal user - Node



(f) Abnormal user - Node

Figure 5.7: Probability timelines for normal and abnormal users

5.2 Dynamic Detection Mechanism

To be able to develop a detection mechanism, the probability distributions and individual user timelines for the three feature set combinations that performed best, were analyzed. Based on the probability timeline for each user, the detection mechanism was tested with all combinations of the decision parameters and their chosen values described in section 4.4.3. In total, 270 decision parameters combinations were tested on the three best feature set combinations. The same metrics, PR, RC, ACC, and $F_{\beta=\{0.5,1,2\}}$, that were tested for SVM were included for better comparison. In table 5.2, 5.3, and 5.4, the top 5 results for the detection mechanism sorted on F_1-scores for the top three feature set combinations are shown. The complete table including all decision parameter combinations are found in appendix C. The confusion matrix for the best 5 F_1-scores across all three feature set combinations are depicted in figure 5.8. The F_1-score for these 5 confusion matrix's are written in bold in table 5.4.

The best F_1-score of 0.970 was achieved with the node and context feature set combination. This score was accomplished after the users had sent an average of 290.54 messages. To obtain these results the detection mechanism started detection after the user had sent 150 messages. The *window size* was set to 40, the *stabilization threshold* was 0.01, and the *decision threshold* were 0.50. In general, the best results were achieved using large *window sizes* (40 or 50), a *decision threshold* of 0.50, a *stabilization threshold* of 0.01, and starting the calculation after at least 150 messages.

Table 5.2: Node, Content, and Context - Top 5 F_1-scores achieved by the detection mechanism

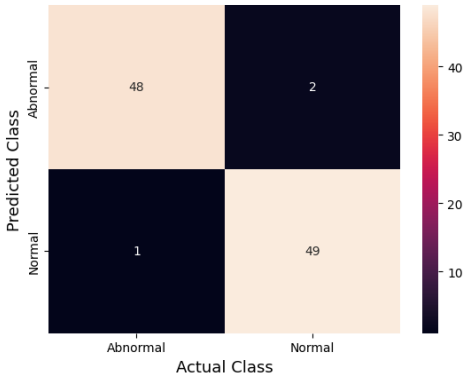
Top 5	Start Msg	Win Size	STAB	DEC	Avg Msg	PR	RC	ACC	$F_{0.5}$	F_1	F_2
1	200	40	0.01	0.50	335.39	0.923	0.960	0.940	0.930	0.941	0.952
2	200	50	0.01	0.25	351.88	0.923	0.960	0.940	0.930	0.941	0.952
3	200	50	0.01	0.50	351.88	0.923	0.960	0.940	0.930	0.941	0.952
4	250	40	0.01	0.50	369.17	0.923	0.960	0.940	0.930	0.941	0.952
5	250	50	0.01	0.25	386.25	0.923	0.960	0.940	0.930	0.941	0.952

Table 5.3: Node and Context - Top 5 F_1-scores achieved by the detection mechanism

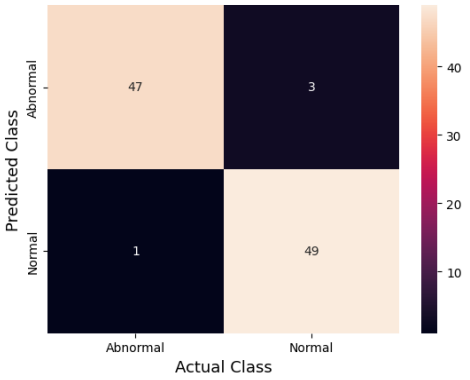
Top 5	Start Msg	Win Size	STAB	DEC	Avg Msg	PR	RC	ACC	$F_{0.5}$	F_1	F_2
1	150	40	0.01	0.50	290.54	0.961	0.980	0.970	0.965	**0.970**	0.976
2	200	50	0.01	0.25	368.52	0.942	0.980	0.960	0.950	**0.961**	0.972
3	250	50	0.01	0.50	402.15	0.942	0.980	0.960	0.950	**0.961**	0.972
4	200	30	0.01	0.50	280.55	0.960	0.960	0.960	0.960	**0.960**	0.960
5	150	40	0.01	0.75	290.54	0.960	0.960	0.960	0.960	**0.960**	0.960

Table 5.4: Node - Top 5 F_1-scores achieved by the detection mechanism

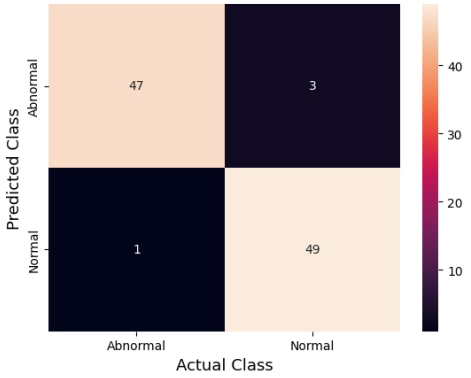
Top 5	Start Msg	Win Size	STAB	DEC	Avg Msg	PR	RC	ACC	$F_{0.5}$	F_1	F_2
1	250	50	0.02	0.25	322.61	0.891	0.980	0.930	0.907	0.933	0.961
2	250	40	0.01	0.25	338.39	0.891	0.980	0.930	0.907	0.933	0.961
3	250	50	0.01	0.25	369.52	0.891	0.980	0.930	0.907	0.933	0.961
4	250	40	0.02	0.25	306.38	0.875	0.980	0.920	0.920	0.925	0.957
5	250	30	0.01	0.25	315.95	0.875	0.980	0.920	0.920	0.925	0.957



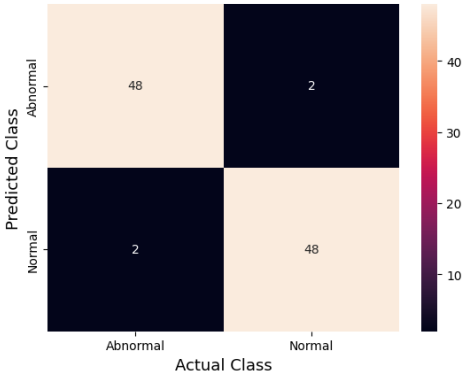
(a) Nr. 1 - F_1-score, Avg Msg sent (Node and Context)



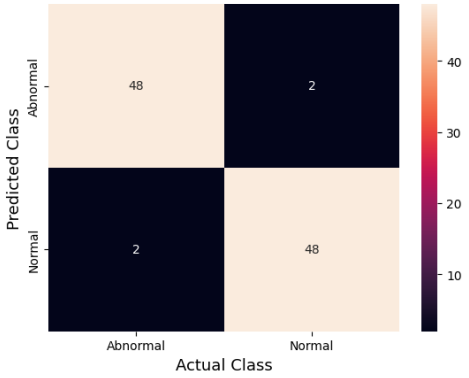
(b) Nr. 2 - F_1-score, Avg Msg sent (Node and Context)



(c) Nr. 3 - F_1-score, Avg Msg sent (Node and Context)



(d) Nr. 4 - F_1-score, Avg Msg sent (Node and Context)



(e) Nr. 5 - F_1-score, Avg Msg sent (Node and Context)

Figure 5.8: Top 5 F_1-Scores across all three feature set combinations when weighing F_1-Score

The main objective with the detection mechanism was to be able to dynamically detect abnormal users after sending as few messages as possible. The results were therefor sorted on number of average messages sent, and then F_1-scores between 0.850 and 0.970 were extracted. The first F_1-score extracted was greater than or equal to 0.850, with the least sent messages. The second F_1-score would then be the first which were greater than or equal to 0.860. For each extracted score the minimum F_1-score was increased with 0.010, until the maximum F_1-score achieved for the feature set combination were extracted. By only looking at the results that had a F_1-score above 0.850, a lot of the earliest detections were deprecated. This trade-off was made due to observations that the performance increased drastically after an average of approximately 65 messages were sent. The results are found in tables 5.5, 5.6, and 5.7. The confusion matrix for the best three F_1-scores, weighted on number of average messages sent, across all three feature set combinations, are illustrated in figure 5.10. These results were chosen based on breakpoints shown in figure 5.9. The F_1-score for the three confusion matrix's are written in bold in tables 5.5 and 5.6.

Tables 5.5, 5.6, and 5.7, presents the best performance results when sorting by the average messages sent value, and then the F_1-score. The node, context and content feature set combination achieved the earliest detection up to a F_1-score of 0.897 and an average of 112.04 messages sent. The node and context feature set achieved the best results after an average of 113.61 messages was sent. The node feature set achieved worse results than the two other feature set combinations. These results will be further discussed in chapter 6.

Table 5.5: Node, Content, and Context - F_1-scores when prioritizing number of average messages sent

Start Msg	Win Size	STAB	DEC	Avg Msg	PR	RC	ACC	$F_{0.5}$	F_1	F_2
50	10	0.04	0.50	63.99	0.783	0.940	0.840	0.810	0.855	0.904
50	10	0.02	0.50	66.60	0.797	0.940	0.850	0.822	**0.862**	0.907
0	50	0.04	0.50	104.28	0.849	0.900	0.870	0.859	0.874	0.889
50	40	0.04	0.25	112.04	0.803	0.980	0.870	0.833	0.883	0.939
50	40	0.04	0.50	112.04	0.842	0.960	0.890	0.863	0.897	0.934
50	50	0.04	0.50	134.95	0.902	0.920	0.910	0.906	0.911	0.916
100	30	0.04	0.25	138.43	0.875	0.980	0.920	0.804	0.925	0.957
50	40	0.02	0.50	150.68	0.922	0.940	0.930	0.925	0.931	0.936
200	40	0.01	0.50	335.39	0.923	0.960	0.940	0.930	0.941	0.952

Table 5.6: Node and Context - Best F_1-scores when prioritizing number of average messages sent

Start Msg	Win Size	STAB	DEC	Avg Msg	PR	RC	ACC	$F_{0.5}$	F_1	F_2
50	10	0.02	0.50	67.11	0.774	0.960	0.840	0.805	0.857	0.916
50	10	0.01	0.50	77.86	0.797	0.940	0.850	0.822	0.862	0.907
50	20	0.02	0.50	87.76	0.810	0.940	0.860	0.833	0.870	0.911
50	40	0.04	0.50	110.20	0.839	0.940	0.880	0.858	0.887	0.918
50	30	0.02	0.75	113.61	0.882	0.900	0.890	0.886	0.891	0.896
50	30	0.02	0.25	113.61	0.870	0.940	0.900	0.883	0.904	0.925
50	30	0.02	0.50	113.61	0.887	0.940	0.910	0.897	0.913	0.929
50	20	0.01	0.50	117.00	0.904	0.940	0.920	0.911	0.922	0.933
50	20	0.01	0.75	117.00	0.922	0.940	0.930	0.925	**0.931**	0.936
50	40	0.02	0.50	143.80	0.923	0.960	0.940	0.930	0.941	0.952
100	20	0.01	0.75	160.42	0.941	0.960	0.950	0.945	**0.950**	0.956
200	30	0.01	0.50	280.55	0.960	0.960	0.960	0.960	0.960	0.960
150	40	0.01	0.50	290.54	0.961	0.980	0.970	0.965	0.970	0.976

Table 5.7: Node - Best F_1-scores when prioritizing number of average messages sent

Start Msg	Win Size	STAB	DEC	Avg Msg	PR	RC	ACC	$F_{0.5}$	F_1	F_2
50	40	0.04	0.50	110.32	0.818	0.900	0.850	0.833	0.857	0.882
50	30	0.02	0.50	122.67	0.821	0.920	0.860	0.839	0.868	0.898
50	50	0.04	0.50	132.05	0.852	0.920	0.880	0.865	0.885	0.906
50	50	0.02	0.50	172.13	0.868	0.920	0.890	0.878	0.893	0.909
200	40	0.02	0.25	267.67	0.857	0.960	0.900	0.876	0.906	0.938
200	30	0.01	0.25	281.18	0.860	0.980	0.910	0.881	0.916	0.953
250	40	0.02	0.25	306.38	0.875	0.980	0.920	0.894	0.925	0.957
250	40	0.01	0.25	338.39	0.891	0.980	0.930	0.907	0.933	0.961

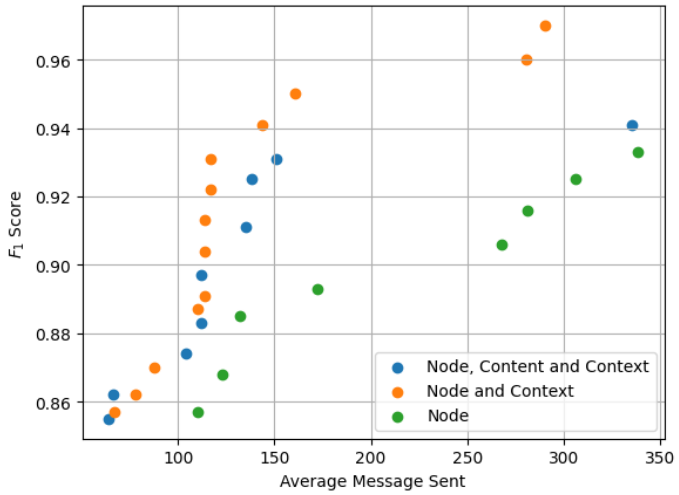
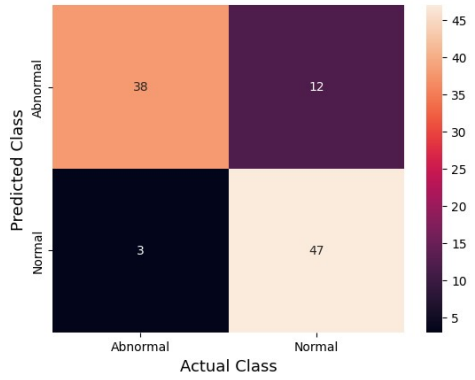
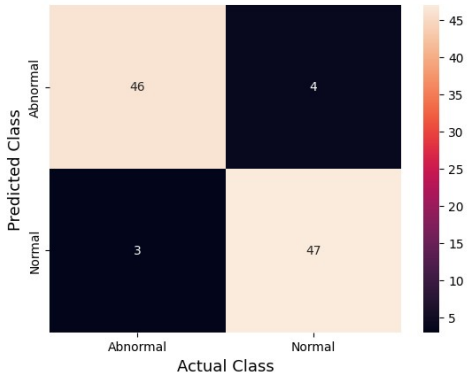


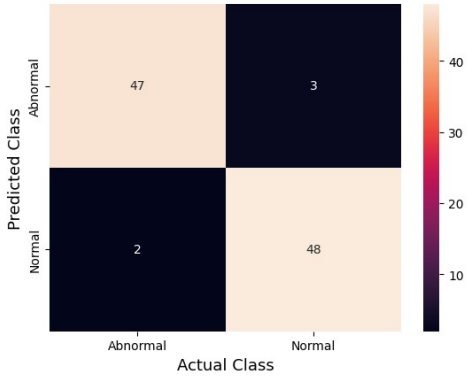
Figure 5.9: Best F_1-scores when prioritizing number of average messages sent for all three feature set combinations



(a) Top 1-3 - Avg Msg sent, F_1-score (Node, Content, and Context, $F_1 = 0.862$)



(b) Top 1-3 - Avg Msg sent, F_1-score (Node, and Context, $F_1 = 0.931$)



(c) Top 1-3 - Avg Msg sent, F_1-score (Node and Context, $F_1 = 0.950$)

Figure 5.10: Top 3 F_1-Scores across all three feature set combinations when weighing Average Message sent before F_1-score

Chapter 6

Discussion

In this chapter the methodology and results are discussed in relation to the overall research question and the sub-questions. Additionally, limitations of the thesis will be discussed.

6.1 Research Question

6.1.1 SubQ1: What features are suitable for detecting predators in chats?

SubQ1 was addressed by performing an initial analysis of normal and abnormal users through an ego-graph simulation, studying conversation length distribution and manually inspecting conversations. With this insight, features were extracted and tested on an SVM model. Tables 4.4, 4.5, and 4.6 show the node, context, and content features that were chosen for this thesis. Different combinations of the three feature categories were tested. The features selected were used to detect abnormal users in the game chat dataset studied in this thesis. Results from the three feature sets are listed in table 5.1.

Node features - 21 node features were added into one feature vector and analyzed. When testing with SVM on this feature set alone it resulted in a F_1-score of 0.935 as shown in table 5.1. The individual node features were not tested, elaborated in section 6.2, this applies to all features.

The initial analysis, using ego-graph simulation, showed that abnormal users tended to have more neighbors, participate in more conversations, and initiate more conversations than normal users. The first two tendencies are captured by the *degree* features, whilst the latter by the *numbers of initiated conversations* features. This implies that those features might aid in the overall performance of the node feature set. In addition, abnormal users tend to have shorter conversations than normal users. This feature is captured by the *Number of conversations within bins of messages* features. As observations performed on the number of messages in a conversation showed a distinct difference between normal and abnormal users, it is reasonable to assume that these features would also be suitable for detecting

abnormal users.

Conversation length distribution features were sorted into fixed bins. The choice of bins was based on observations of a limited selection of users. Another approach might be to create bins based on min-max distribution of the conversation lengths and remove potential outliers.

Normal users tend to communicate with interconnected neighbors, whilst abnormal users to a greater extent communicate with seemingly random users. This behavior was represented by the *Clustering Coefficient*.

Context features - 5 context features were added into one feature vector and analyzed. When testing with SVM on this feature set alone it resulted in a F_1-score of 0.761 as shown in table 5.1.

The context features were calculated for each sent message regardless of which conversation they originated in and then the average of the feature values for the messages were aggregated. By calculating the features in this way some valuable information might disappear due to the aggregation. For example, for the context features *Message length* and *Response Time*, the average across all conversations is calculated. Users might have many conversations with short message lengths and a few conversations with very long message lengths. Utilizing our method, the long and short message lengths might be averaged out by each other. The same might occur when calculating the response time regardless of the conversation. To mitigate this, it could be interesting to investigate other approaches where the feature values are calculated for each conversation, rather than calculating the average across individual messages.

When manually investigating the dataset, it was difficult to distinguish between normal and abnormal users message lengths. It could be interesting to calculate the message length for all labeled users and compare the normal and abnormal users. This might give insight into whether a normal or an abnormal user tends to be most active. In addition, we only calculated the average for sent messages, but it could be interesting to calculate the ratio between sent and received messages within conversations. This could be added as a new context feature and be used to indicate if the normal or abnormal user tends to be most active during a conversation.

Content features - 6 content features were added into one feature vector and analyzed. When testing with SVM on this feature set alone it resulted in an F_1-score of 0.442 as shown in table 5.1.

To be able to dynamically detect abnormal users, we calculated the feature values for each sent message. Then, we calculated the average of the messages to update the feature values. Abnormal users will send some messages containing predatory language, but they will also send normal messages. When aggregating the feature values, predatory messages might drown in normal messages. Hence, the aggregated feature values might not be significant enough when training a classification model. It could be interesting to investigate other methods to calculate the feature values. One approach could be to accumulate the messages as they are sent, and then calculate the feature values based on these messages. This

might capture the development of the messages without losing important data. Labeling all conversations based on the user could be one of the reasons why content features performed poorly. Another approach could be to calculate content and context features for each conversation, instead of calculating for each user. From these features, each conversation can get a probability or risk score, which could be aggregated with node features. In this way, content and context features for conversations could be better captured. The reason for choosing our approach is that calculating new features for each sent message is more efficient than aggregating messages leading to more computationally heavy operations.

Comparing our results with those performed by Cheong et al. [38], shows that they achieved better results when implementing their content features on a similar, but much older dataset. This might be because they removed non-predatory messages from the abnormal users in their dataset, which led to more consistent classification across the messages than what we were able to achieve. They removed the non-predatory messages manually, which in our case, having a large dataset, was not possible as it would be time consuming. It could be interesting to test this approach by decreasing the number to, for example, a maximum of 500 messages per user.

When we labeled the dataset, we would in some cases distinguish between sexting and grooming. If it were sexting based on the message content, but the ego-graph simulation showed tendencies like a normal user, we would label the user as normal. Therefore, it is likely that sexual language can be found in messages from both abnormal and normal users. This might have affected the results, since classification could be more difficult, leading to a low F_1-score. On the other hand, it is reasonable to assume that children who are attending the game platform to start sexting conversations, might show the same tendencies as a predator looking at its ego-graph. This might make it difficult to distinguish between sexting and grooming.

As mentioned in section 4.3.3 we did not have access to the game platform's alert- and blacklist. Instead, we chose features that in some degree could detect attempts to circumvent the platform rules. Based on the content, we were able to find examples of obfuscation attempts which indicated what some of the contents of the blacklist were. If the alert- and blacklist were made available, a more extensive search for attempts to send alert- and blacklist messages could be performed. Adding this as a feature could be used to improve the overall results. By implementing this, the profanity checker feature might not be necessary as most profanity words likely are present in the alert- and blacklist.

When we manually looked through messages during the labeling process, we noticed that several users which we labeled as abnormal would start conversations with the same opening lines, such as *How old are you?*. Therefore, it could be interesting to include features which look for such opening lines which might indicate that the user is abnormal.

6.1.2 SubQ2: What combinations of graph theory, context, and content analysis can optimize detection of predators in chats?

The intention of SubQ2 was to investigate whether there are any combinations of graph theory, context, and content analysis that can optimize detection of abnormal users. To answer this question, we need to look at the detection mechanism, its variables, and how it performed on the different feature set combinations. We chose to test the detection mechanism for the three feature set combinations that achieved the best F_1-score with SVM. These feature set combinations were; 1) Node, Content, and Context ($F_1 = 0.951$), 2) Node and Context ($F_1 = 940$), and 3) Node ($F_1 = 0.935$). We considered two approaches to optimize detection of abnormal users; 1) Prioritizing the F_1-score, 2) Prioritizing number of average messages sent, then F_1-score. For the latter, we will discuss the trade-off between the number of messages sent and the achieved F_1-score.

Results when prioritizing the F_1-score - This detection approach focused on detecting abnormal users by combining the detection parameters with the intent to achieve the best F_1-score, even at the expense of early detection. The best results achieved when prioritizing the F_1-scores for the three feature set combinations are found in tables 5.2, 5.3, and 5.4. The best result achieved across all three feature set combinations were for the Node and Context feature set with a F_1-score of 0.970. This result was achieved using the following detection mechanism parameters; *start msg* $= 150$, *window size* $= 40$, *stabilization threshold* $= 0.01$, and *decision threshold* $= 0.50$. The Node feature set alone did achieve decent results when prioritizing the F_1-score, but performed worse than the two other feature set combinations. On the other hand, it is the smallest feature set making it easier and faster to calculate without a lot of performance degradation.

The overall best results were achieved when we used large *window size*, a low *stabilization threshold*, and started the calculations after at least 150 messages. In general, we discovered that using a *decision threshold* of 0.50 achieved the best results. An exception of this was for the node feature set which achieved the best results using a *decision threshold* of 0.25, but the F_1-scores for this feature set were lower than for the two other feature set combinations. This implies that a *decision threshold* of 0.50 is preferable over 0.25 or 0.75. Based on the top 5 results for all three feature set combinations, we found that a *window size* of either 40 or 50 achieved the best results together with a low *stabilization threshold* of 0.01. This might be because using a large *window size* with a strict *stabilization threshold*, enables the detection mechanism to require more stable results. At the same time, to acquire these results, we observe that the *start message*, where the mechanism starts the detection, must be after at least 150 messages, and in most cases after more than 200-250 messages. This leads to more accurate detection of abnormal users, but at the cost of early detection.

Results when prioritizing number of average messages sent - This detection approach focused on detecting abnormal users by combining the detection parameters prioritizing the average message sent before the F_1-score. Based on

investigations into the results achieved when implementing the detection mechanism, we manually picked out what we considered to be the best results prioritizing number of average messages sent, but which at the same time achieved F_1-scores higher than 0.85. The 0.85 cut off point was chosen due to observations that after approximately 65 messages in average were sent, the results increased drastically. In addition, To narrow down the list of results further, we extracted the best result within each percentile after 0.85, such as the best result within the 0.86 percentile with the lowest average of messages sent. These extractions were carried out until the best F_1-score for the feature set were extracted. Based on this selection, the best results achieved when prioritizing number of average messages sent before the F_1-scores for the three feature set combinations, are found in tables 5.5, 5.6, and 5.7. Note that some of these results achieved a precision lower than 0.80 which is the minimum precision needed in a practical system.

We plotted the F_1-scores extracted for all three feature set combinations in figure 5.9. This figure shows several breakpoints where the results stagnated. This figure was used to pick out results which could be used to optimize the detection giving the best F_1-score and keep the number of average messages sent as low as possible. Based on the figure, we found three datapoints that seemed to uphold these criteria. If the methodology were to be utilized by the game owners, they can choose their own desired operating points. But for this thesis we chose the first datapoint from the Node, Context, and Content feature set. The number of average messages sent was 66.60 and an F_1-score of 0.862. This implies a very early detection of abnormal users whilst still achieving a decent F_1-score. The two other datapoints were from the Node and Context feature set. The number of average messages sent was 117.00 and 160.42 and F_1-scores of 0.931 and 0.950, respectively. These performed better regarding the F_1-score, than the first result, but with longer detection time.

In general, all three showed the same tendencies when it came to the combination of detection parameters. The *window size* and *stabilization threshold* were low, the *decision threshold* was high, and the detection started after 50 or 100 messages. When comparing these results to the ones achieved when prioritizing F_1-scores, some significant differences were observed. The *window size*, when prioritizing the F_1-scores, were high (40 or 50), whilst for early detection it was low (10 or 20). This implies that the results achieved by early detection is based on more unstable detections. Despite this, the early detection results do achieve decent results, only 0.02 from the best result when prioritizing F_1-scores. The *start message* parameter also differed between the two, where the detection started early (50 – 100 messages) for early detection and later (200-250 messages) when prioritizing the F_1-score. This is natural as early detection requires the detection to start after as few messages as possible. An obvious consequence of using these detection parameters is that it can lead to more misclassifications, which is also shown by the lower F_1-scores achieved. The confusion matrixes shown in figures 5.8 and 5.10 also illustrate the difference in classification between the two.

Comparing our results with those achieved by Eng [12], shows that we are able

to achieve better results both when prioritizing the number of average messages sent and the F_1-score. For example, Engs detection mechanism achieved an F_1-score of 0.791 and a precision of 0.878 after an average of 200.40 messages sent when combining Node and Context features. The results achieved in this thesis were significantly better with an F_1-score of 0.931 and a precision of 0.922 after an average of 117.00 messages sent when combining Node and Context features. The Node and Context features used in this thesis are almost the same as those used by Eng which indicates that it is not the features that are the main reason to the increased results. One reason for the difference in the results can be that we calculate the features differently than Eng. In addition, we did not use the same labeled dataset as Eng which also can have caused the differences.

Detection mechanism - We chose to implement a dynamic detection mechanism to be able to detect abnormal users as early as possible, with as high F_1-scores feasible. The detection mechanism should be dynamic as the probability scores achieved after the SVM model, varied for each user. We tested all combinations of four different parameters and their values on the three best performing feature set combinations, as described in section 4.4.3. This testing was performed to find the best combinations of parameters which could be used to achieve early detection. As previously mentioned, the parameters used to achieve the best results when prioritizing the F_1-score, and those used when prioritizing the number of average messages sent, were different. But, in general we observed that some of the parameters influenced each other. If the detection started after at least 150 messages, the *window size* could be large and the *stabilization threshold* low, still obtaining decent results. It was also this combination of parameters that achieved the best results when prioritizing F_1-scores. On the other hand, we observed that if detection started early (0-50 messages), the *window size* had to be smaller, and the *stabilization threshold* higher to achieve decent results. This is probably because the probability scores varied more after few messages were sent.

The probability score for the users tends to oscillate during the first 50 messages sent, as illustrated in figure 5.6. This varied from user to user, but we observed that the detection mechanism was not able to classify the users during this period as the scores had not stabilized enough. However, as illustrated in figures 5.6 and 5.7, the timelines did not stabilize after 50 messages, but they crossed the *decision threshold* of 0.5. If the *stabilization threshold* had been greater than 0.04, accurate classifications might be possible. It could be interesting to look at a detection mechanism that monitors the average probability scores for users after a given number of messages sent and classify the users with an average over or under a *decision threshold*. This method could be implemented without a *stabilization threshold* or the use of *window size*.

6.1.3 RQ1: Can graph theory, context, and content analysis be utilized to dynamically detect predators in chats?

During this thesis we have investigated several combinations of feature sets from node, context and content, and results achieved when trying to detect abnormal users. After running all feature set combinations with the SVM model we found that Node, Context and Content features achieved the best results when classifying individual messages. In addition, the Node and Context feature set and Node feature set alone also achieved decent results. Context features and Content features alone seemed less appropriate for detection. We decided to use the three feature set combinations that performed best together with the detection mechanism and tried to find detection parameters which could be used to achieve early detection of abnormal users.

After testing all combinations of the detection parameters on the three feature sets, we found that all three achieved decent results and could be used for dynamic detection of abnormal users. Even though all three feature sets achieved decent results, we found that the combination of Node and Context features enabled the detection mechanism to perform the earliest detection of abnormal users. The Node feature set did achieve the worst results out of the three. The Node, Context and Content feature sets performed better than the Node feature set, but worse than the Node and Context feature set. This might imply that Content features lower the classification results.

Whether the results achieved were decent or not was based on how early the detection mechanism was able to detect abnormal users with the best F_1-score. An example of what we categorized as decent results, were the combination of detection parameters on the Node and Context feature set where the detection mechanism was able to classify abnormal users, after they had sent an average of 117.00 messages and still achieving a F_1-score of 0.931. The dynamic detection mechanism also proved to be appropriate for detecting abnormal users when the F_1-score was deemed most important. When prioritizing the F_1-score, the detection mechanism was able to detect abnormal users after an average of 290.54 messages and achieve a F_1-score of 0.970.

6.2 Limitations

When we trained the SVM model we used a balanced dataset with 50 normal and 50 abnormal users. In reality, the dataset is highly unbalanced which usually affects the performance of the SVM model negatively. Therefore, it would be interesting to investigate the use of unbalanced datasets together with SVM.

The dataset used for training was not labeled by the chat moderators. This means that we had to label it ourselves only by interpreting the available data. This might lead to a biased dataset as we used the features to determine if a user should be labeled as normal or abnormal. Previous studies have also performed the similar manual labeling [12], meaning that this limitation also extends to

other datasets.

When manually labeling the dataset, it was difficult to identify a predator and hence we decided to look for abnormal users which could include users who exhibited predatory behavior, but also spammers, users who are sexting or others who display behavior that is not categorized as normal.

The performance for each individual feature within the feature sets was not evaluated due to computational limitations. This made it difficult to accurately determine what individual features would be most suitable for detecting abnormal users.

For content analysis, we considered several features which we did not implement in the thesis. We tested the implementation of BoW, but due to the large dataset it resulted in a too large dictionary, and it was not computationally possible to use with SVM. If we had limited the number of messages to for example a maximum of 500 messages per user, we might have been able to calculate BoW features. In addition, we considered adding sentiment features. Sentiment features are known to work best on longer texts [7]. Due to our choice of methodology, only looking at one message at a time, we considered that these features would not work well with our short messages. If applying a methodology where the messages are accumulated as they are sent, and the feature values calculated based on these messages, it could be more relevant to implement sentiment features as this will accumulate into larger texts to test on.

We only looked at messages that were sent by the user and not the received messages. It could be interesting to look at both sent and received messages as the content of the received messages might give valuable information, such as how a victim reacts to a message sent from a predator.

During the thesis we only applied one machine learning algorithm, SVM. It could be interesting to use additional classification algorithms which might enhance the performance further. Also, when applying the SVM model, we only used the default parameters, these parameters might not be the ones that perform best. Testing with different SVM parameters, such as the regularization parameter, could be performed to detect the most efficient combinations. Different kernel parameters, such as the sigmoid or gaussian, could be tested to see if they better fit the data structure of the dataset.

Chapter 7

Conclusion and Future Work

This chapter concludes this thesis by presenting the achieved results and future work. We will present concrete answers to the research question elaborated in chapter 1. In addition, some areas regarding early detection of cybergrooming that should be studied or investigated further in the future, will be presented.

7.1 Conclusion

This master thesis has investigated whether combining different analysis approaches could be used to dynamically detect predators in chats. The results show that the methodology utilized can detect abnormal users both when prioritizing early detection and F_1-score. Detection of abnormal users, when prioritizing the F_1-score, was achieved after the users had sent an average of approximately 290 messages with an F_1-score of 0.970. When prioritizing early detection, several results could be considered reliable, we highlight the one that was able to detect abnormal users after an average of approximately 117 messages with an F_1-score of 0.931. Both results were achieved using Node and Context features in combination. Based on these results there are a trade-off between how early an abnormal users can be detected and the detection accuracy. For the chat moderators this means that, if choosing the early detection method, they will encounter more false positive results than if they chose the method where accuracy is prioritized.

The dataset received from Aiba, containing game chat data, was used to extract features to detect abnormal users. In addition, a subset was labeled and used to train the classification algorithm. A dynamic detection mechanism was implemented utilizing probability scores calculated by the classification algorithm. This methodology was implemented to answer the research question and sub-questions created for this thesis.

To answer the first sub-question, *What features are suitable for detecting predators in chats?*, we identified 32 features distributed across three behavioral categories: Node, Context, and Content. The combination of all features proved to enhance the performance of the SVM classification, compared to the other possible combinations of the features tested in this thesis. Node features alone contributed

most when classifying individual messages, but combining these with both Context and Content features improved the results further. Therefore, all three feature set combinations proved to be suitable for individual message classification.

The second sub-question, *What combinations of graph theory, context, and content analysis can optimize detection of predators in chats?*, was answered by utilizing a detection mechanism with four detection parameters. We calculated the results for all combinations of the detection parameters with the feature set combinations that proved most promising after the SVM classification. The Node and Context feature set proved to optimize the detection of abnormal users when prioritizing the F_1-score. The detection parameter configuration in this case were a large *window size*, a low *stabilization threshold*, a *decision threshold* at 0.50, and starting detection after the users had sent at least 150 messages. However, when prioritizing early detection, the parameter combination that proved to optimize the detection was a low *window size* and *stabilization threshold*, a *decision threshold* at 0.75, and starting detection after 50 messages. The overall results for the detection mechanism showed that, in contrast to the results achieved for the SVM classification, only combining Node and Context features proved to perform best regarding optimizing the detection mechanism. It was evident that Node features contributed most to the detection, but including Context features enhanced these results further. Including Content features with the two other, enhanced the detection compared to the Node features alone, but deprecated the results achieved with Node and Context features. The Content features chosen in this thesis are therefore proven less suitable for early detection of abnormal users.

Based on the results acquired when answering the sub-questions we can conclude that combining different analysis approaches can indeed be used to dynamically detect predators in chats.

The results achieved in this thesis contribute to dynamically detect predators in chats. The methodology utilizes two approaches for detection, which can be used by chat moderators to narrow down the pool of abnormal users. One limitation with this thesis is the use of only one classification algorithm with default parameter values. Another limitation was the dataset labeling, which was based on our interpretation of the messages, rather than by chat moderators. Our labeling can have led to biased labeling which influenced the achieved results.

7.2 Future Work

Based on the investigations performed in this thesis we identified several subjects which would be interesting for future research. This includes the use of other classification algorithms, the inclusion of other features, and approaches to calculate them, and utilization of other detection mechanisms.

To be able to continue the research into improving detection of predators in chats, other classification algorithms than SVM should be tested. By exploring different classification algorithms, the most effective and appropriate algorithm could be identified.

Results show that the Content features chosen did not improve the dynamic detection. Further research, including Content feature calculations for aggregated messages, rather than individual messages, should be explored. Further feature calculations to find the optimal feature selection from the feature sets provided in this thesis should be explored. Inclusion of other features could also be investigated such as including sentiment features and BoW to enhance the detection. In addition, looking at both sent and received messages could be studied to determine if even more content features could be extracted and aid in the overall detection results. Another method that should be investigated is to calculate content and context features for each conversation, rather than for each user. This might mitigate some of the limitations discussed.

In this thesis we utilized a detection mechanism with several detection parameters. Different approaches for detection should be further investigated, such as a mechanism that monitors the average probability scores for users which circumvents the need for the *stabilization threshold* and *window size* parameters.

Bibliography

[1] B. Dybvik and J. Kaatorp, 'Research project planning: Dynamic graph theoretical and content analysis of cybergrooming detection in chatrooms,' Department of Information Security, Communication NTNU – Norwegian University of Science and Technology, Project report in IMT4205, Jul. 2023.

[2] P. deShame, *Young people's experiences of online sexual harassment*, eng, 2017. [Online]. Available: https://www.childnet.com/what-we-do/our-projects/project-deshame/research/.

[3] M. Harris, S. Allardyce and D. Findlater, 'Child sexual abuse and covid-19: Side effects of changed societies and positive lessons for prevention,' *Criminal Behaviour and Mental Health*, vol. 31, pp. 289–292, Oct. 2021. DOI: 10.1002/cbm.2214.

[4] D. F. Isabelle Anillo and T. Kennedy, 'A global outlook on child sexual abuse and sexually explicit material online during covid-19: Trends and interdisciplinary prevention methods,' *Journal of Child Sexual Abuse*, vol. 32, no. 8, pp. 921–939, 2023, PMID: 37994473. DOI: 10.1080/10538712.2023.2285960. eprint: https://doi.org/10.1080/10538712.2023.2285960. [Online]. Available: https://doi.org/10.1080/10538712.2023.2285960.

[5] Z. Zuo, J. Li, P. Anderson, L. Yang and N. Naik, 'Grooming detection using fuzzy-rough feature selection and text classification,' Jul. 2018. DOI: 10.1109/FUZZ-IEEE.2018.8491591.

[6] E. Villatoro-Tello, A. Juárez-González, H. J. Escalante, M. Montes-y-Gómez and L. Villaseñor-Pineda, 'A two-step approach for effective detection of misbehaving users in chats,' in *Conference and Labs of the Evaluation Forum*, 2012.

[7] D. Bogdanova, P. Rosso and T. Solorio, 'On the impact of sentiment and emotion based features in detecting online sexual predators,' in *Proceedings of the 3rd Workshop in Computational Approaches to Subjectivity and Sentiment Analysis*, Jeju, Korea: Association for Computational Linguistics, Jul. 2012, pp. 110–118. [Online]. Available: https://aclanthology.org/W12-3717.

[8] M. A. Wani, N. Agarwal and P. Bours, 'Sexual-predator detection system based on social behavior biometric (ssb) features,' eng, 2021, ISSN: 1877-0509. [Online]. Available: https://hdl.handle.net/11250/2982092.

[9] M. Fire, G. Katz and Y. Elovici, 'Strangers intrusion detection-detecting spammers and fake profiles in social networks based on topology anomalies,' *Human journal*, vol. 1, no. 1, pp. 26–39, 2012.

[10] A. H. Wang, 'Don't follow me: Spam detection in twitter,' in *2010 International Conference on Security and Cryptography (SECRYPT)*, 2010, pp. 1–10.

[11] A. F. Aarekol, *A graph theoretical approach to online predator detection*, eng, 2022. [Online]. Available: https://hdl.handle.net/11250/3014533.

[12] I. E. Eng, *Dynamic graph theoretical analysis of cybergrooming detection in chatrooms*, eng, 2023.

[13] G. M. Winters and E. L. Jeglic, 'Online sexual grooming,' in *Sexual Grooming: Integrating Research, Practice, Prevention, and Policy*. Cham: Springer International Publishing, 2022, pp. 65–86, ISBN: 978-3-031-07222-2. DOI: 10.1007/978-3-031-07222-2_5. [Online]. Available: https://doi.org/10.1007/978-3-031-07222-2_5.

[14] E. Martellozzo, 'Policing online child sexual abuse: The british experience,' *European Journal of Policing Studies*, vol. 3, pp. 32–52, Sep. 2015. DOI: 10.5553/EJPS/2034760X2015003001003.

[15] A. Schulz, E. Bergen, P. Schuhmann, J. Hoyer and P. Santtila, 'Online sexual solicitation of minors: How often and between whom does it occur?' *Journal of Research in Crime and Delinquency*, vol. ahead of print, Sep. 2015. DOI: 10.1177/0022427815599426.

[16] K. Babchishin, R. Hanson and C. Hermann, 'The characteristics of online sex offenders: A meta-analysis,' *Sexual Abuse A Journal of Research and Treatment*, vol. 23, pp. 92–123, Mar. 2011. DOI: 10.1177/1079063210370708.

[17] P. M. Briggs, W. T. Simon and S. Simonsen, 'An exploratory study of internet-initiated sexual offenses and the chat room sex offender: Has the internet enabled a new typology of sex offender?' *Sexual Abuse: A Journal of Research and Treatment*, vol. 23, pp. 72–91, 2011. [Online]. Available: https://api.semanticscholar.org/CorpusID:29502427.

[18] S. Webster, J. Davidson, A. Bifulco, P. Gottschalk, V. Caretti, T. Pham, J. Grove-Hills, C. Turley, C. Tompkins, S. Ciulla, V. Milazzo, A. Schimmenti and G. Craparo, 'European online grooming project - final report,' Mar. 2012.

[19] D. DeHart, G. Dwyer, M. Seto, R. Moran, E. Letourneau and D. Schwarz-Watts, 'Internet sexual solicitation of children: A proposed typology of offenders based on their chats, e-mails, and social network posts,' *Journal of Sexual Aggression*, pp. 1–13, Oct. 2016. DOI: 10.1080/13552600.2016.1241309.

[20] R. O. Cyberspace, 'A typology of child cybersexploitation and online grooming practices,' 2003. [Online]. Available: https://api.semanticscholar.org/CorpusID:25803044.

[21] R. J. Wilson, *Introduction to graph theory*, en, 4th ed. London, England: Prentice-Hall, Mar. 1996.

[22] S. Rahman, *Basic Graph Theory* (Undergraduate Topics in Computer Science), en, 1st ed. Cham, Switzerland: Springer International Publishing, May 2017.

[23] R. Gould, *Graph theory*. Courier Corporation, 2012.

[24] S. Leonardi, *Algorithms and Models for the Web-Graph: Third International Workshop, WAW 2004, Rome, Italy, October 16, 2004, Proceeedings* (Lecture Notes in Computer Science), eng. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004, vol. 3243, ISBN: 3662200384.

[25] D. L. Hansen, B. Shneiderman and M. A. Smith, 'Chapter 3 - social network analysis: Measuring, mapping, and modeling collections of connections,' in *Analyzing Social Media Networks with NodeXL*, D. L. Hansen, B. Shneiderman and M. A. Smith, Eds., Boston: Morgan Kaufmann, 2011, pp. 31–50, ISBN: 978-0-12-382229-1. DOI: https://doi.org/10.1016/B978-0-12-382229-1.00003-5. [Online]. Available: https://www.sciencedirect.com/science/article/pii/B9780123822291000035.

[26] S. Bhattacharya, S. Sinha, P. Dey, A. Saha, C. Chowdhury and S. Roy, 'Chapter 5 - online social-network sensing models,' in *Computational Intelligence Applications for Text and Sentiment Data Analysis*, ser. Hybrid Computational Intelligence for Pattern Analysis and Understanding, D. Das, A. K. Kolya, A. Basu and S. Sarkar, Eds., Academic Press, 2023, pp. 113–140, ISBN: 978-0-323-90535-0. DOI: https://doi.org/10.1016/B978-0-32-390535-0.00010-0. [Online]. Available: https://www.sciencedirect.com/science/article/pii/B9780323905350000100.

[27] W. Ertel, *Introduction to artificial intelligence*. Springer, 2018.

[28] M. I. Jordan and T. M. Mitchell, 'Machine learning: Trends, perspectives, and prospects,' *Science*, vol. 349, no. 6245, pp. 255–260, 2015.

[29] Z.-H. Zhou, *Machine learning*. Springer Nature, 2021.

[30] S. Badillo, B. Banfai, F. Birzele, I. Davydov, L. Hutchinson, T. Kam-Thong, J. Siebourg-Polster, B. Steiert and J. D. Zhang, 'An introduction to machine learning,' *Clinical Pharmacology & Therapeutics*, vol. 107, Mar. 2020. DOI: 10.1002/cpt.1796.

[31] H. Li, 'Support vector machine,' in *Machine Learning Methods*. Singapore: Springer Nature Singapore, 2024, pp. 127–177, ISBN: 978-981-99-3917-6. DOI: 10.1007/978-981-99-3917-6_7. [Online]. Available: https://doi.org/10.1007/978-981-99-3917-6_7.

[32] IBM, *What Is Support Vector Machine? | IBM — ibm.com*, https://www.ibm.com/topics/support-vector-machine, [Accessed 02-03-2024], 2023.

[33] G. Inches and F. Crestani, 'Overview of the international sexual predator identification competition at pan-2012,' Jan. 2012.

[34] P. J. Black, M. Wollis, M. Woodworth and J. T. Hancock, 'A linguistic analysis of grooming strategies of online child sex offenders: Implications for our understanding of predatory sexual behavior in an increasingly computer-mediated world,' *Child Abuse & Neglect*, vol. 44, pp. 140–149, 2015, ISSN: 0145-2134. DOI: https://doi.org/10.1016/j.chiabu.2014.12.004. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0145213414004360.

[35] M. A. Wani, N. Agarwal, S. Jabin and S. Z. Hussain, 'User emotion analysis in conflicting versus non-conflicting regions using online social networks,' *Telematics and Informatics*, vol. 35, no. 8, pp. 2326–2336, 2018, ISSN: 0736-5853. DOI: https://doi.org/10.1016/j.tele.2018.09.012. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0736585318308402.

[36] M. S. Rahman, S. Halder, M. A. Uddin and U. K. Acharjee, 'An efficient hybrid system for anomaly detection in social networks,' *Cybersecurity*, vol. 4, no. 1, p. 10, Mar. 2021, ISSN: 2523-3246. DOI: 10.1186/s42400-021-00074-w. [Online]. Available: https://doi.org/10.1186/s42400-021-00074-w.

[37] J. L. Tang, *Early detection of cybergrooming conversations*, eng, 2023.

[38] Y.-G. Cheong, A. K. Jensen, E. R. Guðnadóttir, B.-C. Bae and J. Togelius, 'Detecting predatory behavior in game chats,' *IEEE Transactions on Computational Intelligence and AI in Games*, vol. 7, no. 3, pp. 220–232, 2015. DOI: 10.1109/TCIAIG.2015.2424932.

[39] A. Majeed and I. Rauf, 'Graph theory: A comprehensive survey about graph theory applications in computer science and social networks,' *Inventions*, vol. 5, no. 1, 2020, ISSN: 2411-5134. DOI: 10.3390/inventions5010010. [Online]. Available: https://www.mdpi.com/2411-5134/5/1/10.

[40] J. Golbeck, 'Chapter 21 - analyzing networks,' in *Introduction to Social Media Investigation*, J. Golbeck, Ed., Boston: Syngress, 2015, pp. 221–235, ISBN: 978-0-12-801656-5. DOI: https://doi.org/10.1016/B978-0-12-801656-5.00021-4. [Online]. Available: https://www.sciencedirect.com/science/article/pii/B9780128016565000214.

[41] L. Muchnik, S. Pei, L. . Parra, S. D. Reis, J. S. A. Jr, S. Havlin and H. A. Makse, 'Origins of power-law degree distribution in the heterogeneity of human activity in social networks,' 2013. [Online]. Available: https://www.nature.com/articles/srep01783#citeas.

[42] D. J. Watts and S. H. Strogatz, 'Collective dynamics of 'small-world' networks,' *Nature*, vol. 393, no. 6684, pp. 440–442, 1998, ISSN: 1476-4687. DOI: 10.1038/30918. [Online]. Available: https://doi.org/10.1038/30918.

[43] F. Harary and G. Gupta, 'Dynamic graph models,' *Mathematical and Computer Modelling*, vol. 25, no. 7, pp. 79–87, 1997, ISSN: 0895-7177. DOI: https://doi.org/10.1016/S0895-7177(97)00050-2. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0895717797000502.

[44] J. Guo, Z. Han, Z. Su, J. Li, V. Tresp and Y. Wang, 'Continuous temporal graph networks for event-based graph data,' 2022. [Online]. Available: https://aclanthology.org/2022.dlg4nlp-1.3.pdf.

[45] Y. Zhong and C. Huang, 'A dynamic graph representation learning based on temporal graph transformer,' *Alexandria Engineering Journal*, vol. 63, pp. 359–369, 2023, ISSN: 1110-0168. DOI: https://doi.org/10.1016/j.aej.2022.08.010. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1110016822005336.

[46] S. Huang, 'Temporal graph learning in 2023,' 2016.

[47] S. M. Kazemi, R. Goel, K. Jain, I. Kobyzev, A. Sethi, P. Forsyth and P. Poupart, 'Representation learning for dynamic graphs: A survey,' 2020. [Online]. Available: https://www.jmlr.org/papers/volume21/19-447/19-447.pdf.

[48] F. Manessi, A. Rozza and M. Manzo, 'Dynamic graph convolutional networks,' *Pattern Recognition*, vol. 97, p. 107 000, 2020, ISSN: 0031-3203. DOI: https://doi.org/10.1016/j.patcog.2019.107000. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0031320319303036.

[49] P. Zheng, S. Yuan, X. Wu, J. Li and A. Lu, *One-class adversarial nets for fraud detection*, 2018. arXiv: 1803.01798 [cs.LG].

[50] W. Yu, W. Cheng, C. C. Aggarwal, K. Zhang, H. Chen and W. Wang, 'Netwalk: A flexible deep embedding approach for anomaly detection in dynamic networks,' in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, ser. KDD '18, London, United Kingdom: Association for Computing Machinery, 2018, pp. 2672–2681, ISBN: 9781450355520. DOI: 10.1145/3219819.3220024. [Online]. Available: https://doi.org/10.1145/3219819.3220024.

[51] X. Ma, J. Wu, S. Xue, J. Yang, C. Zhou, Q. Z. Sheng, H. Xiong and L. Akoglu, 'A comprehensive survey on graph anomaly detection with deep learning,' *IEEE Transactions on Knowledge and Data Engineering*, pp. 1–1, 2021. DOI: 10.1109/TKDE.2021.3118815.

[52] J. Sun, M. Gu, C.-C. M. Yeh, Y. Fan, G. Chowdhary and W. Zhang, 'Dynamic graph node classification via time augmentation,' 2022.

[53] pandas.pydata.org, *Pandas - python data analysis library — pandas.pydata.org*, 2024. [Online]. Available: https://pandas.pydata.org/.

[54] *Langdetect — pypi.org*, https://pypi.org/project/langdetect/, [Accessed 13-03-2024].

[55] P. S. Aric Hagberg Dan Schult. 'Networkx.' (), [Online]. Available: https://github.com/networkx/networkx. [Accessed 15-01-2024].

[56] M. Kotoglou, *Alt-profanity-check*, https://pypi.org/project/alt-profanity-check/, [Accessed 22-03-2024], 2024.

[57] *Numpy*, https://numpy.org/, [Accessed 15-01-2024].

[58] P. Nordvig, *Pyspellchecker*, 2018. [Online]. Available: https://pyspellchecker.readthedocs.io/en/latest/.

[59] C. J. C. Burges, 'A tutorial on support vector machines for pattern recognition,' *Data Mining and Knowledge Discovery*, vol. 2, pp. 121–167, 1998. [Online]. Available: https://api.semanticscholar.org/CorpusID:215966761.

[60] *Support vector machine*, https://scikit-learn.org/stable/modules/svm.html, [Accessed 28-03-2024].

Appendix A

Results from SVM model for all combinations of feature-set

A.1 Node

Table A.1: Node - SVM 5-fold cross validation results

Fold	PR	RC	ACC	$F_{0.5}$	F_1	F_2
1	0.981	0.904	0.929	0.965	0.941	0.918
2	0.945	0.988	0.979	0.953	0.966	0.979
3	0.939	0.995	0.977	0.950	0.966	0.983
4	0.973	0.986	0.983	0.975	0.980	0.984
5	0.731	0.936	0.851	0.764	0.821	0.886
Average	0.914	0.962	0.944	0.921	0.935	0.950

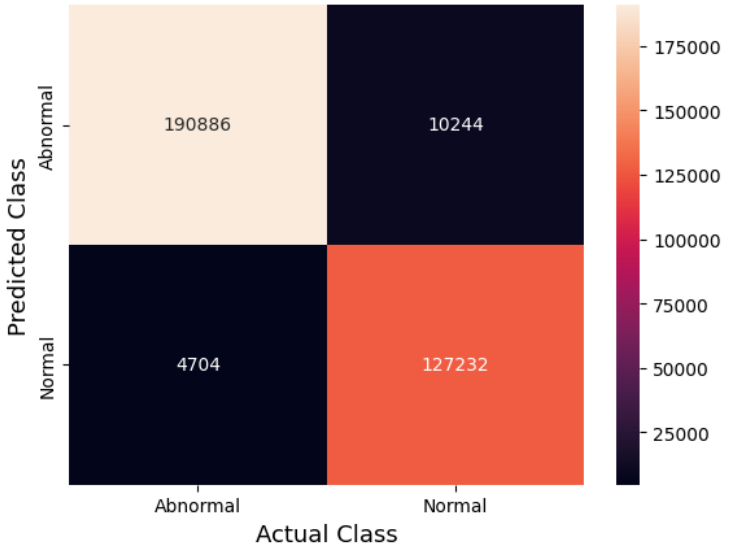
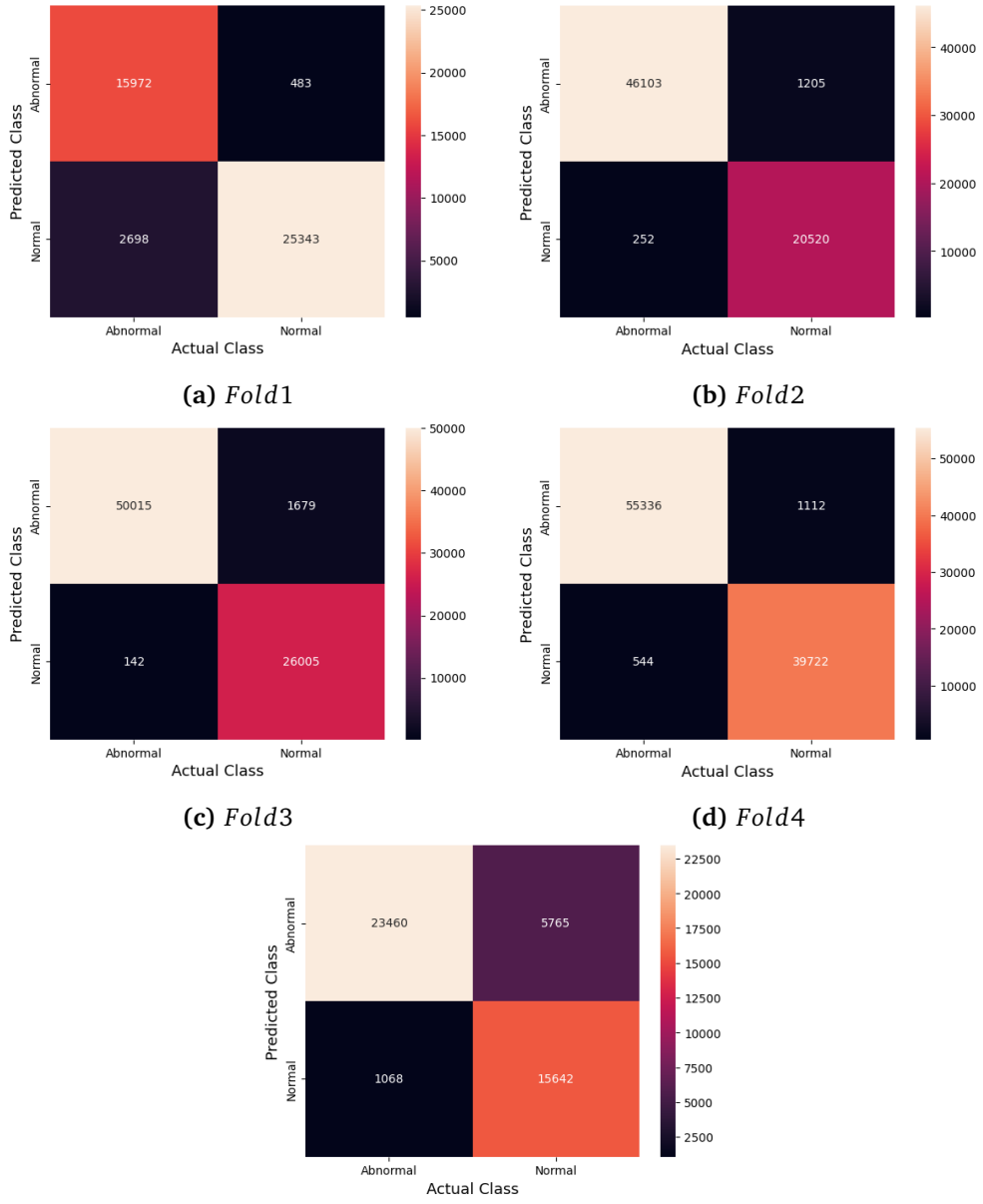


Figure A.1: Total Confusion Matrix for all 5 folds for Node Features



(a) $Fold1$ **(b)** $Fold2$

(c) $Fold3$ **(d)** $Fold4$

(e) $Fold5$

Figure A.2: Node features - All 5 folds

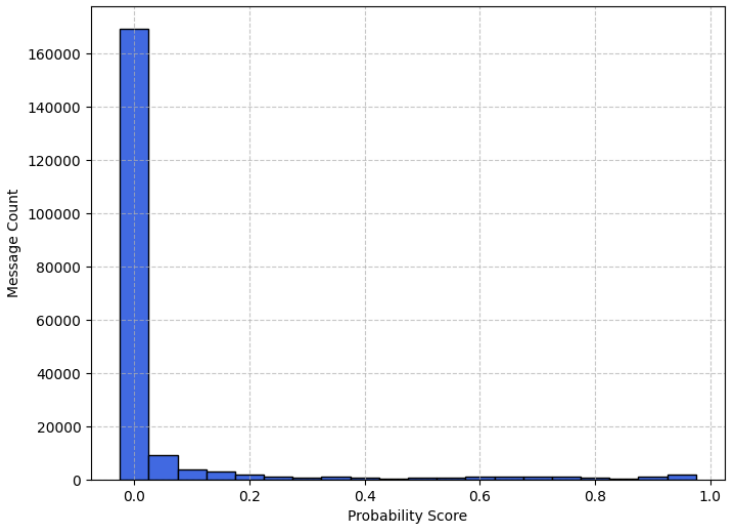


Figure A.3: Distribution of Probability Score - Normal Users - Node Features

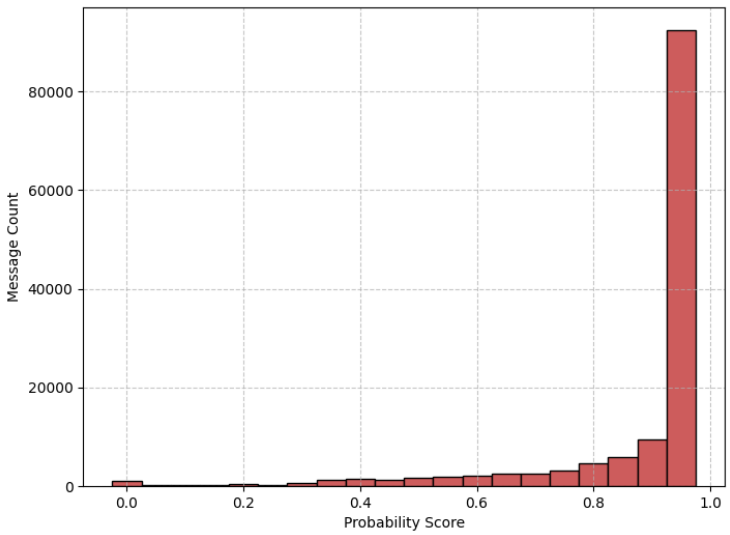


Figure A.4: Distribution of Probability Score - Abnormal Users - Node Features

A.2 Content

Table A.2: Content - SVM 5-fold cross validation results

Fold	PR	RC	ACC	$F_{0.5}$	F_1	F_2
1	0.872	0.305	0.534	0.636	0.452	0.350
2	0.401	0.600	0.605	0.430	0.481	0.546
3	0.891	0.502	0.812	0.772	0.642	0.550
4	0.538	0.164	0.593	0.369	0.251	0.190
5	0.718	0.261	0.694	0.532	0.383	0.299
Average	0.684	0.366	0.648	0.548	0.442	0.387

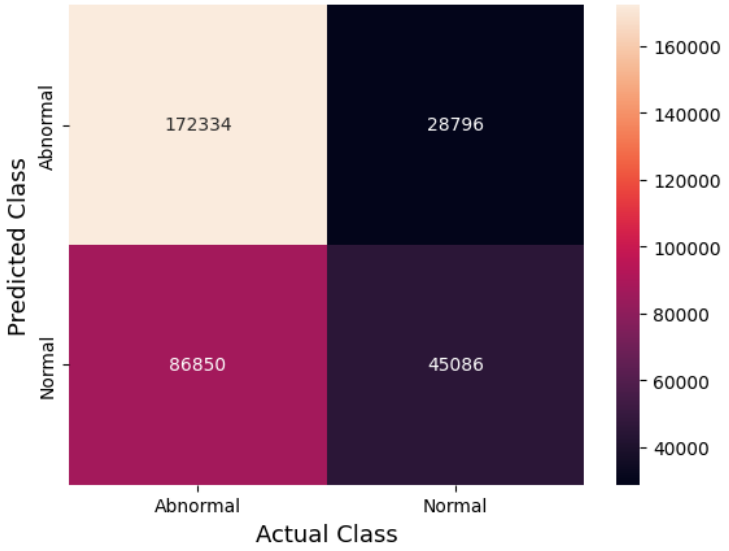
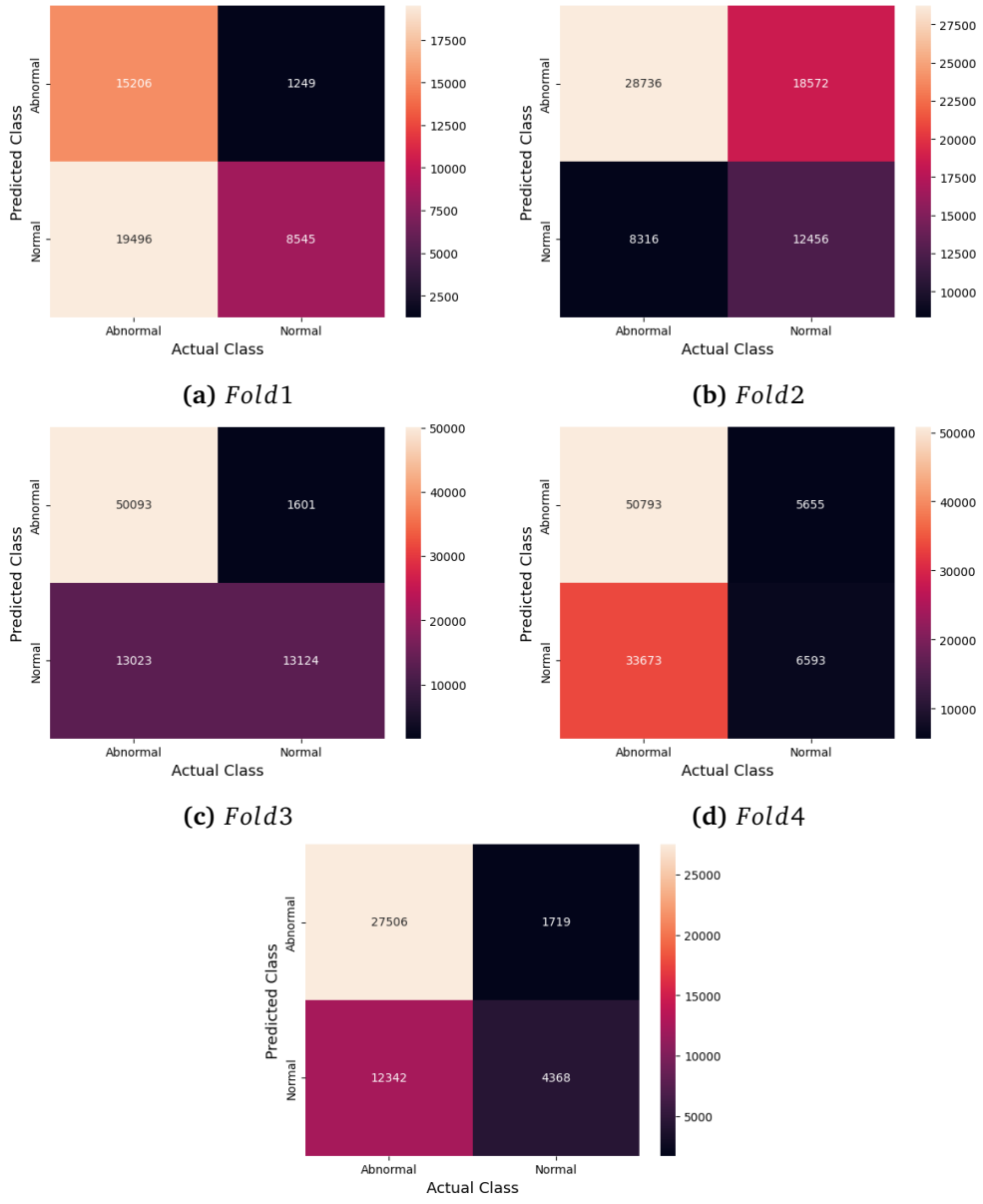


Figure A.5: Total Confusion Matrix for all 5 folds for Content Features



(a) $Fold1$

(b) $Fold2$

(c) $Fold3$

(d) $Fold4$

(e) $Fold5$

Figure A.6: Content features - All 5 folds

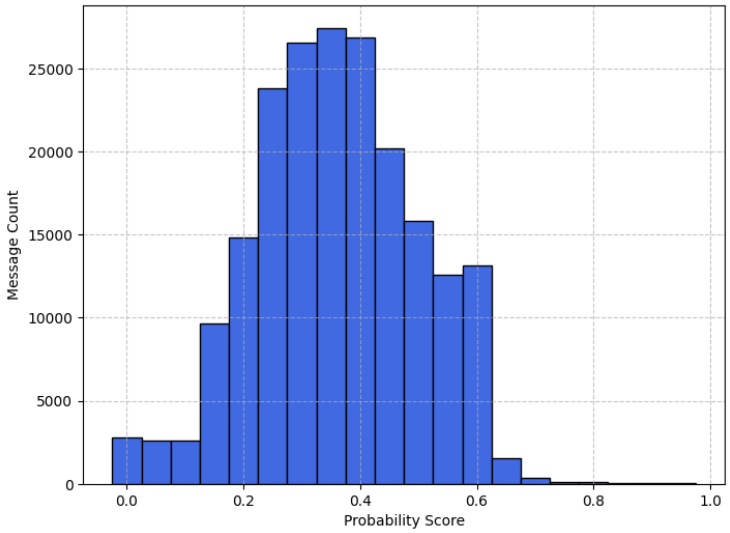


Figure A.7: Distribution of Probability Score - Normal Users - Content Features

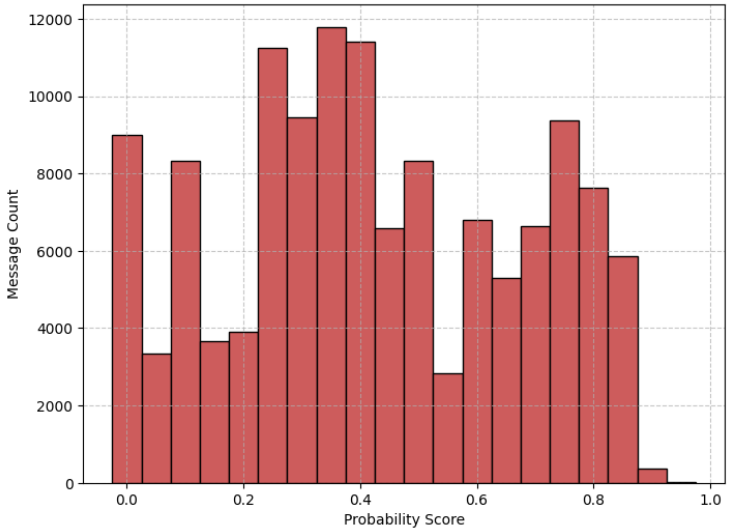


Figure A.8: Distribution of Probability Score - Abnormal Users - Content Features

A.3 Context

Table A.3: Context - SVM 5-fold cross validation results

Fold	PR	RC	ACC	$F_{0.5}$	F_1	F_2
1	0.690	0.738	0.626	0.699	0.713	0.728
2	0.765	1.000	0.906	0.803	0.867	0.942
3	0.734	0.893	0.855	0.761	0.806	0.856
4	0.768	0.564	0.747	0.716	0.650	0.595
5	0.686	0.870	0.808	0.716	0.767	0.826
Average	0.728	0.813	0.788	0.739	0.761	0.789

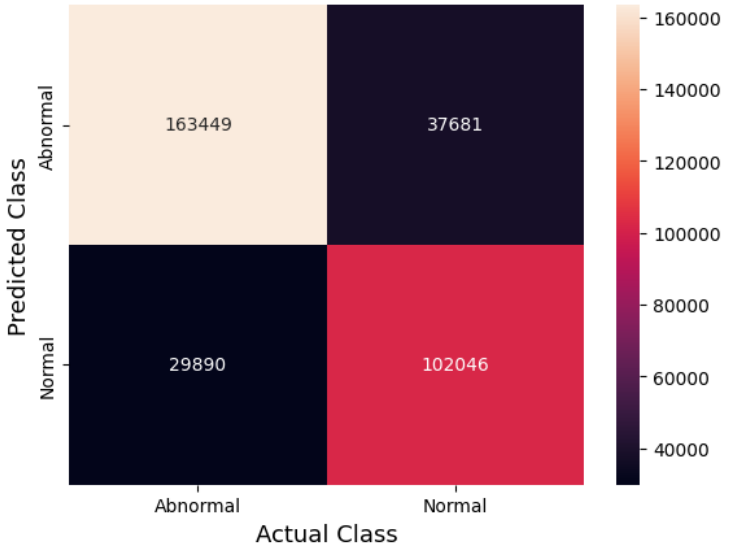
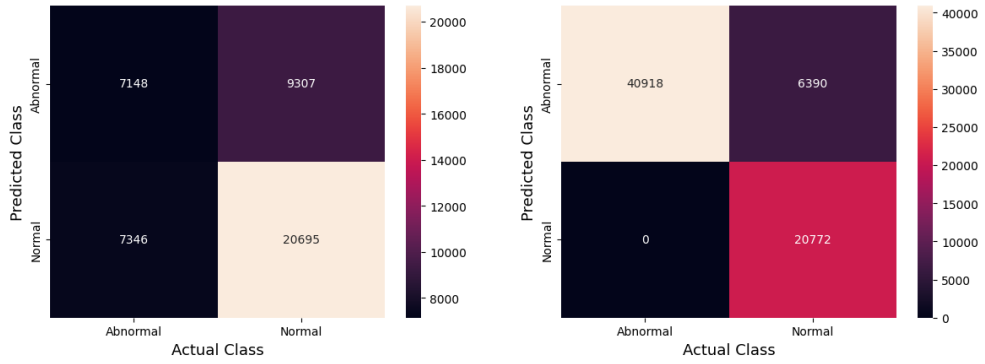
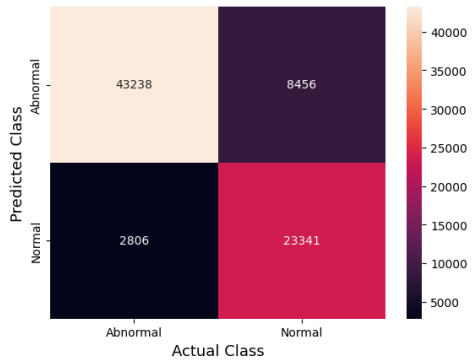


Figure A.9: Total Confusion Matrix for all 5 folds for Context Features

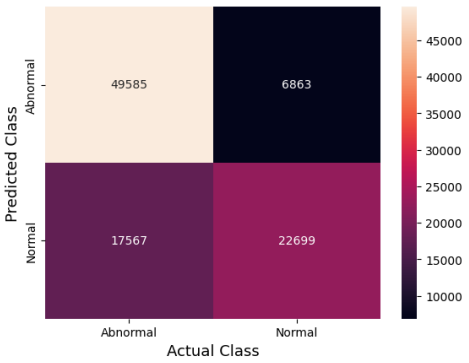


(a) $Fold1$

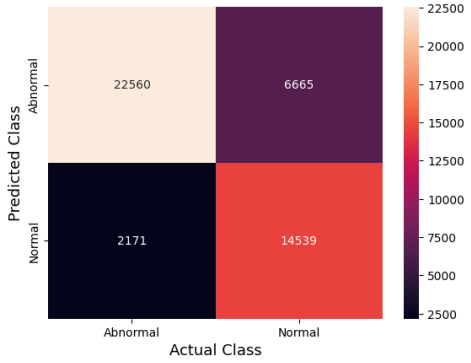
(b) $Fold2$



(c) $Fold3$



(d) $Fold4$



(e) $Fold5$

Figure A.10: Context features - All 5 folds

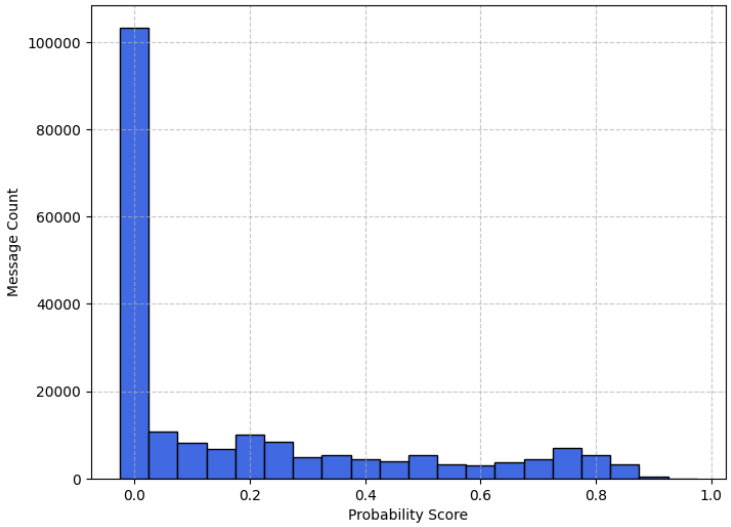


Figure A.11: Distribution of Probability Score - Normal Users - Context Features

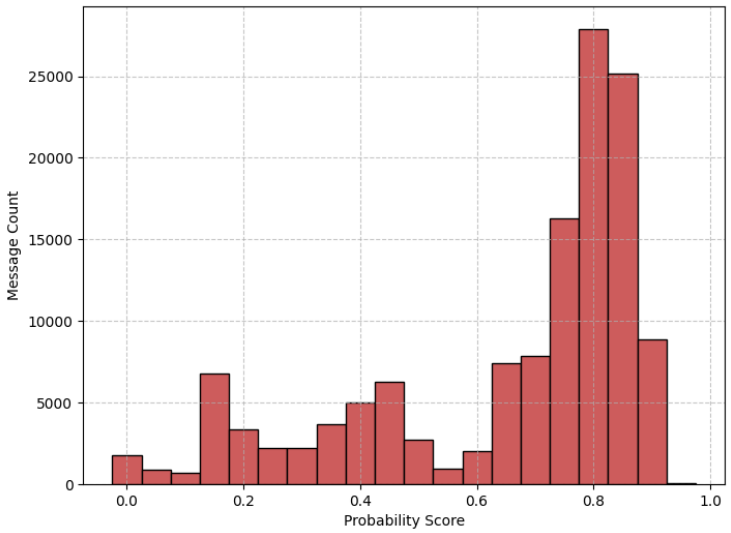


Figure A.12: Distribution of Probability Score - Abnormal Users - Context Features

A.4 Node and Content

Table A.4: Node and Content - SVM 5-fold cross validation results

Fold	PR	RC	ACC	$F_{0.5}$	F_1	F_2
1	0.987	0.890	0.923	0.966	0.936	0.908
2	0.908	0.995	0.968	0.924	0.950	0.976
3	0.961	0.991	0.983	0.966	0.975	0.985
4	0.981	0.982	0.985	0.981	0.981	0.982
5	0.728	0.915	0.845	0.759	0.811	0.871
Average	0.913	0.955	0.941	0.919	0.931	0.944

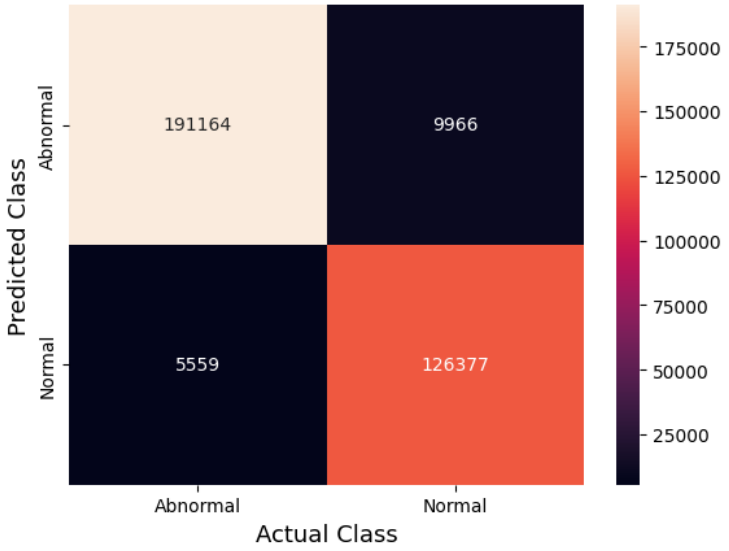
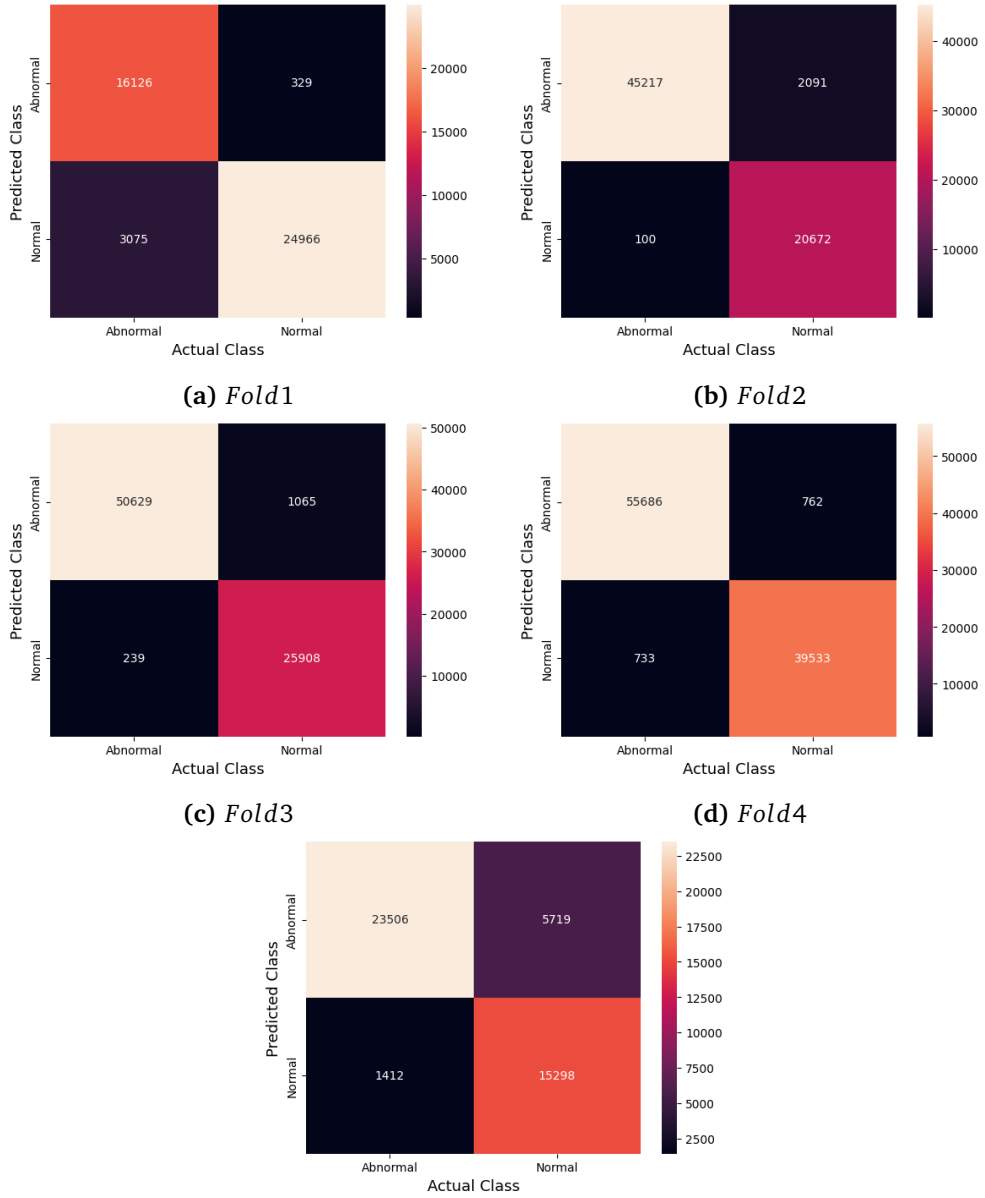


Figure A.13: Total Confusion Matrix for all 5 folds for Node and Content Features



(a) $Fold1$

(b) $Fold2$

(c) $Fold3$

(d) $Fold4$

(e) $Fold5$

Figure A.14: Node and Content features - All 5 folds

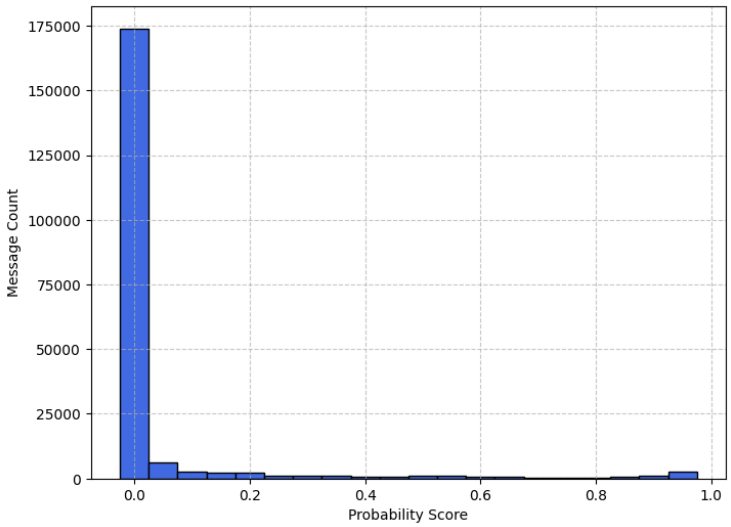


Figure A.15: Distribution of Probability Score - Normal Users - Node and Content Features

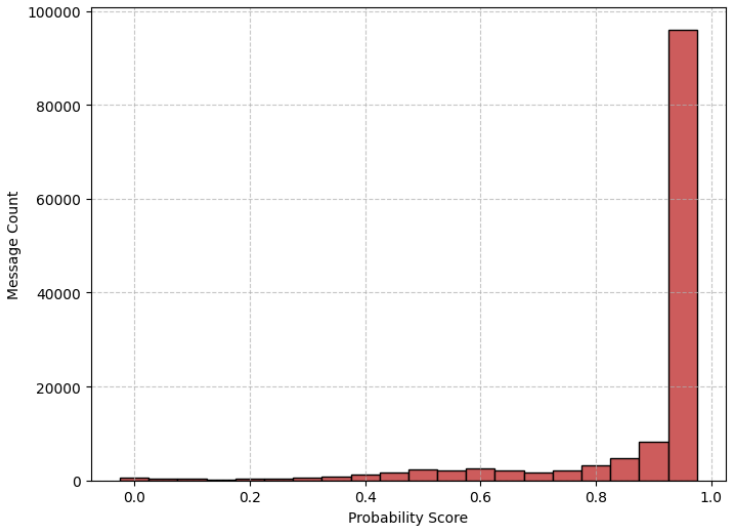


Figure A.16: Distribution of Probability Score - Abnormal Users - Node and Content Features

A.5 Node and Context

Table A.5: Node and Context - SVM 5-fold cross validation results

Fold	PR	RC	ACC	$F_{0.5}$	F_1	F_2
1	0.949	0.940	0.930	0.947	0.945	0.942
2	0.976	0.998	0.992	0.980	0.987	0.993
3	0.956	0.996	0.984	0.964	0.976	0.988
4	0.823	0.990	0.907	0.852	0.899	0.951
5	0.836	0.959	0.917	0.858	0.894	0.932
Average	0.908	0.977	0.946	0.920	0.940	0.961

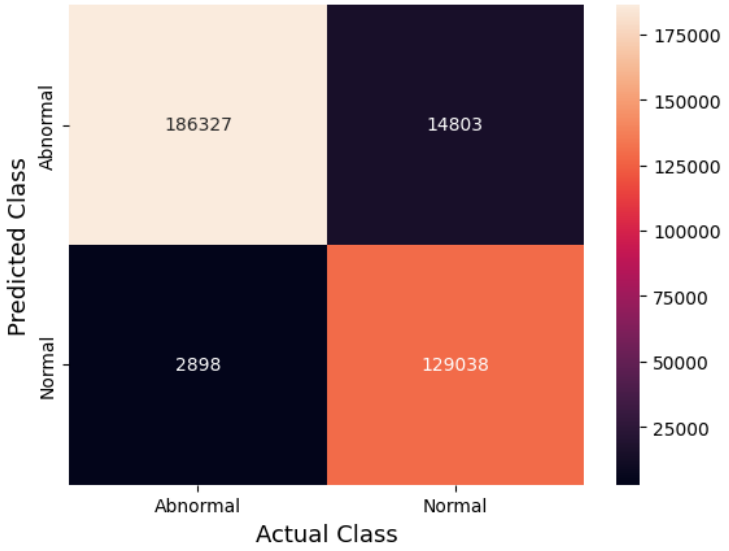
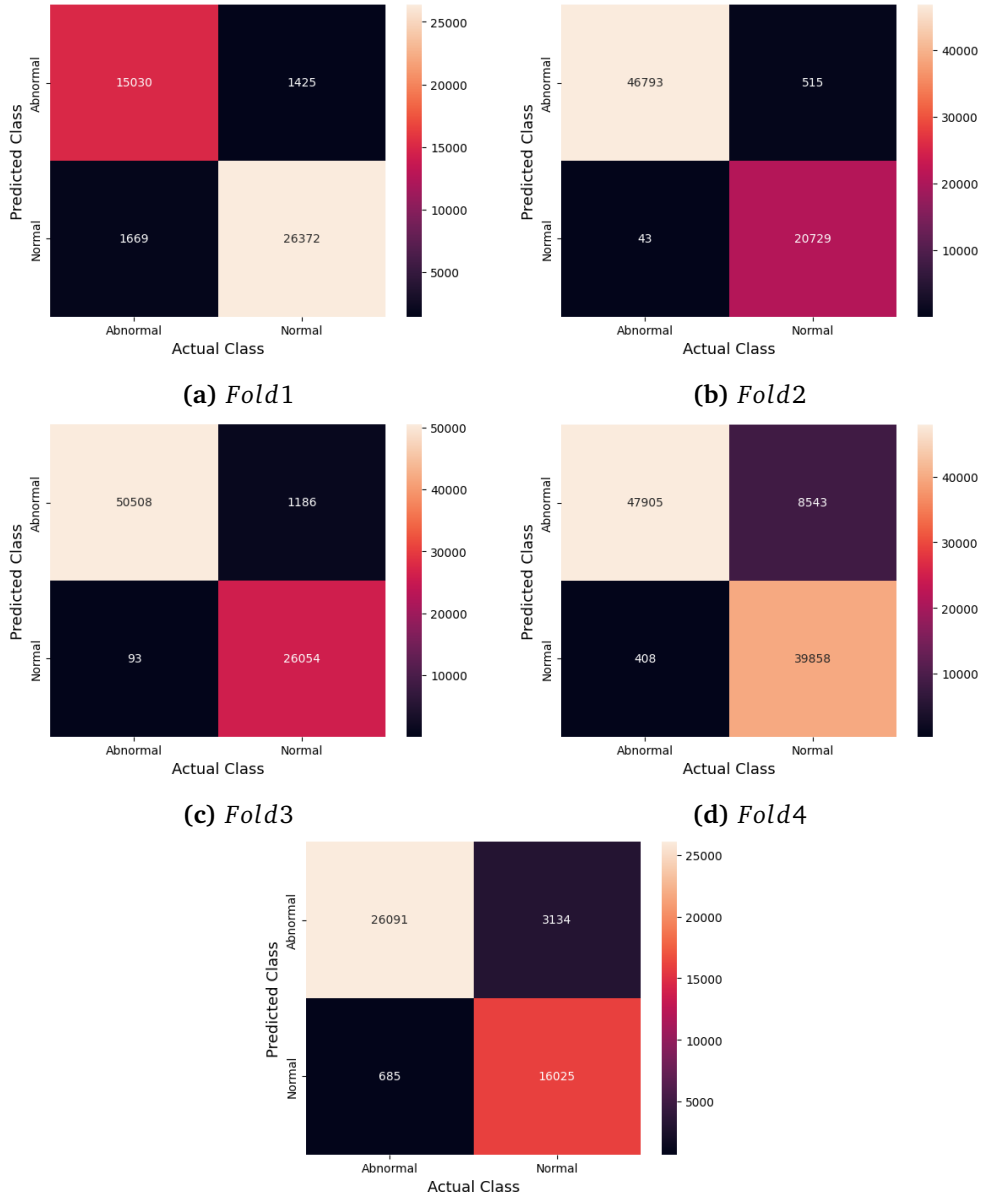


Figure A.17: Total Confusion Matrix for all 5 folds for Node and Context Features



(a) $Fold1$ **(b)** $Fold2$

(c) $Fold3$ **(d)** $Fold4$

(e) $Fold5$

Figure A.18: Node and Context features - All 5 folds

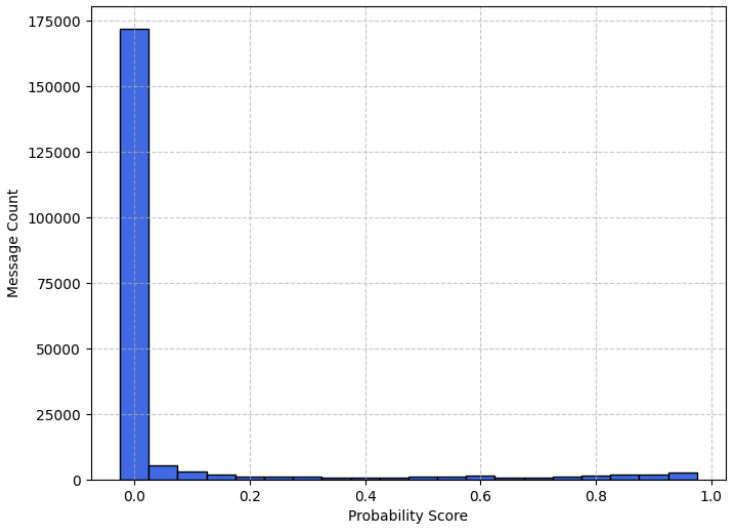


Figure A.19: Distribution of Probability Score - Normal Users - Node and Context Features

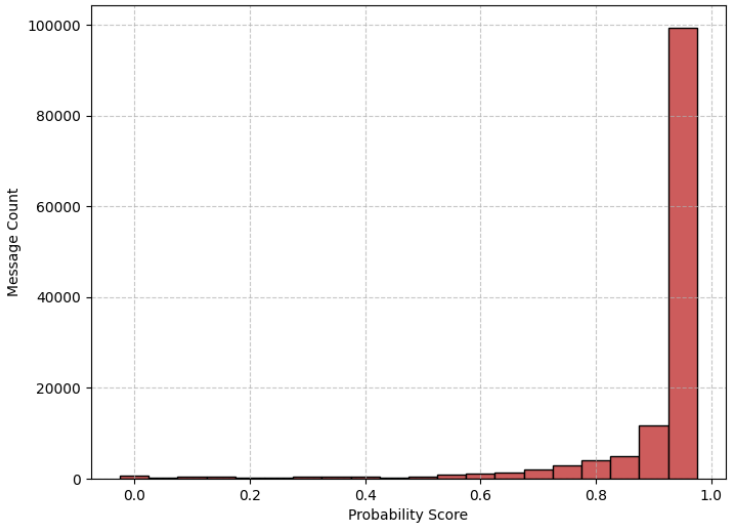


Figure A.20: Distribution of Probability Score - Abnormal Users - Node and Context Features

A.6 Content and Context

Table A.6: Content and Context - SVM 5-fold cross validation results

Fold	PR	RC	ACC	$F_{0.5}$	F_1	F_2
1	0.710	0.435	0.532	0.630	0.539	0.471
2	0.777	1.000	0.913	0.814	0.875	0.946
3	0.617	0.798	0.766	0.647	0.696	0.754
4	0.856	0.468	0.746	0.734	0.605	0.514
5	0.697	0.902	0.822	0.730	0.786	0.852
Average	0.731	0.721	0.756	0.711	0.700	0.707

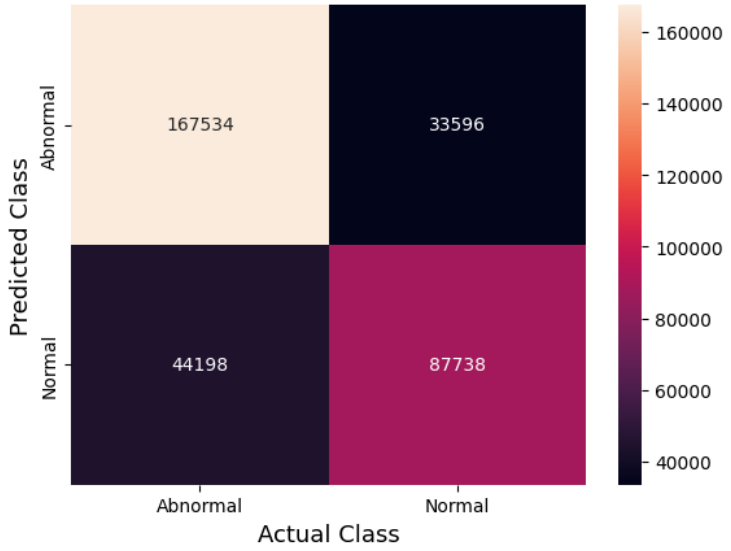
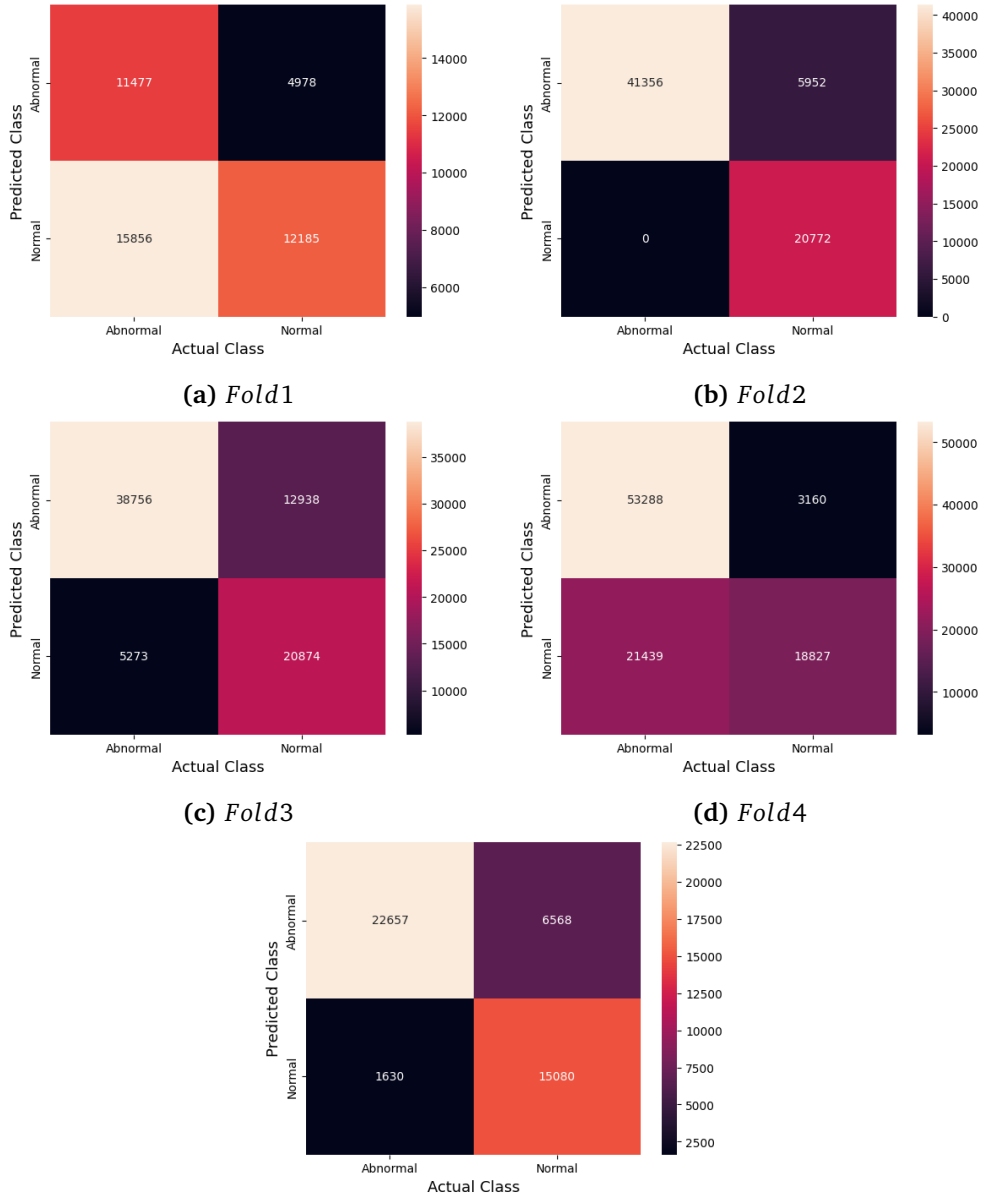


Figure A.21: Total Confusion Matrix for all 5 folds for Content and Context Features



(a) $Fold1$ **(b)** $Fold2$

(c) $Fold3$ **(d)** $Fold4$

(e) $Fold5$

Figure A.22: Content and Context features - All 5 folds

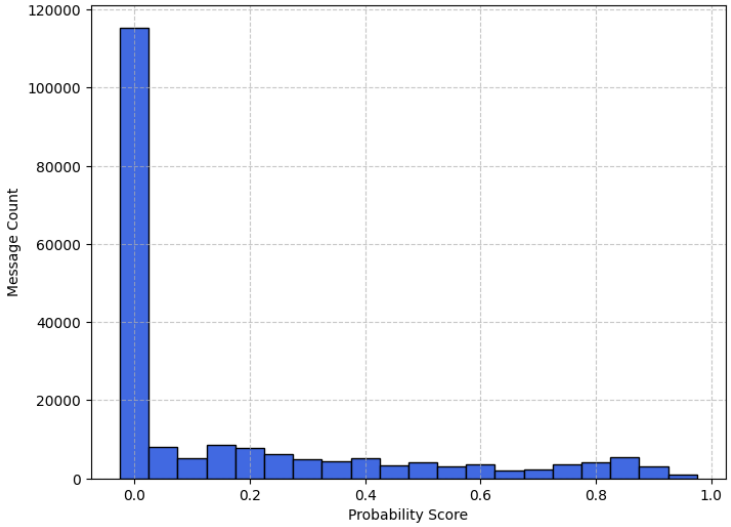


Figure A.23: Distribution of Probability Score - Normal Users - Node and Content Features

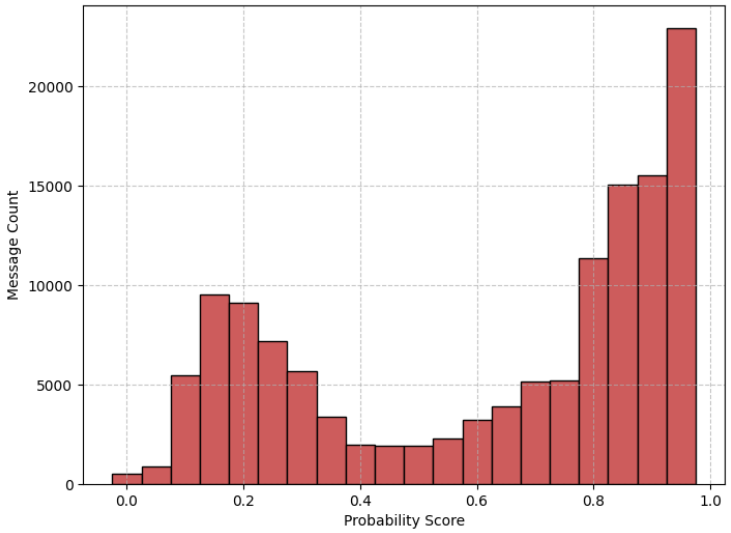


Figure A.24: Distribution of Probability Score - Abnormal Users - Node and Content Features

A.7 Node, Content and Context

Table A.7: Node, Content and Context - SVM 5-fold cross validation results

Fold	PR	RC	ACC	$F_{0.5}$	F_1	F_2
1	0.957	0.952	0.943	0.956	0.955	0.953
2	0.955	0.998	0.985	0.964	0.976	0.989
3	0.956	0.996	0.983	0.964	0.976	0.988
4	0.974	0.989	0.984	0.977	0.981	0.986
5	0.828	0.909	0.898	0.843	0.867	0.892
Average	0.934	0.969	0.959	0.941	0.951	0.962

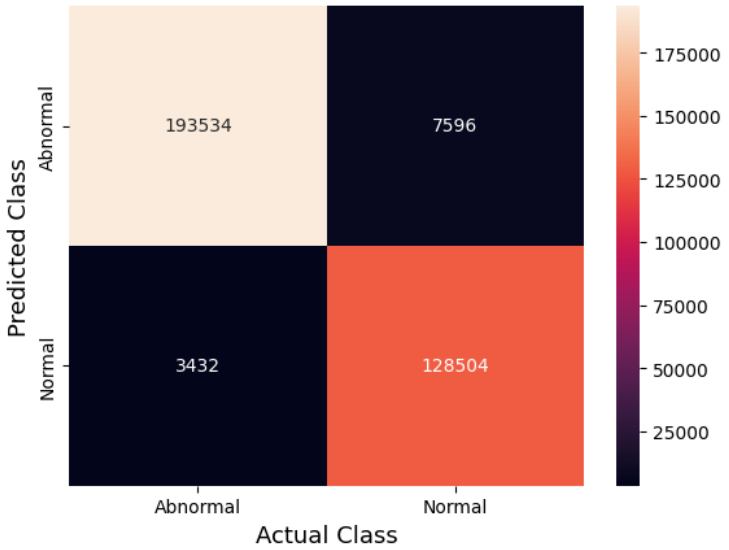
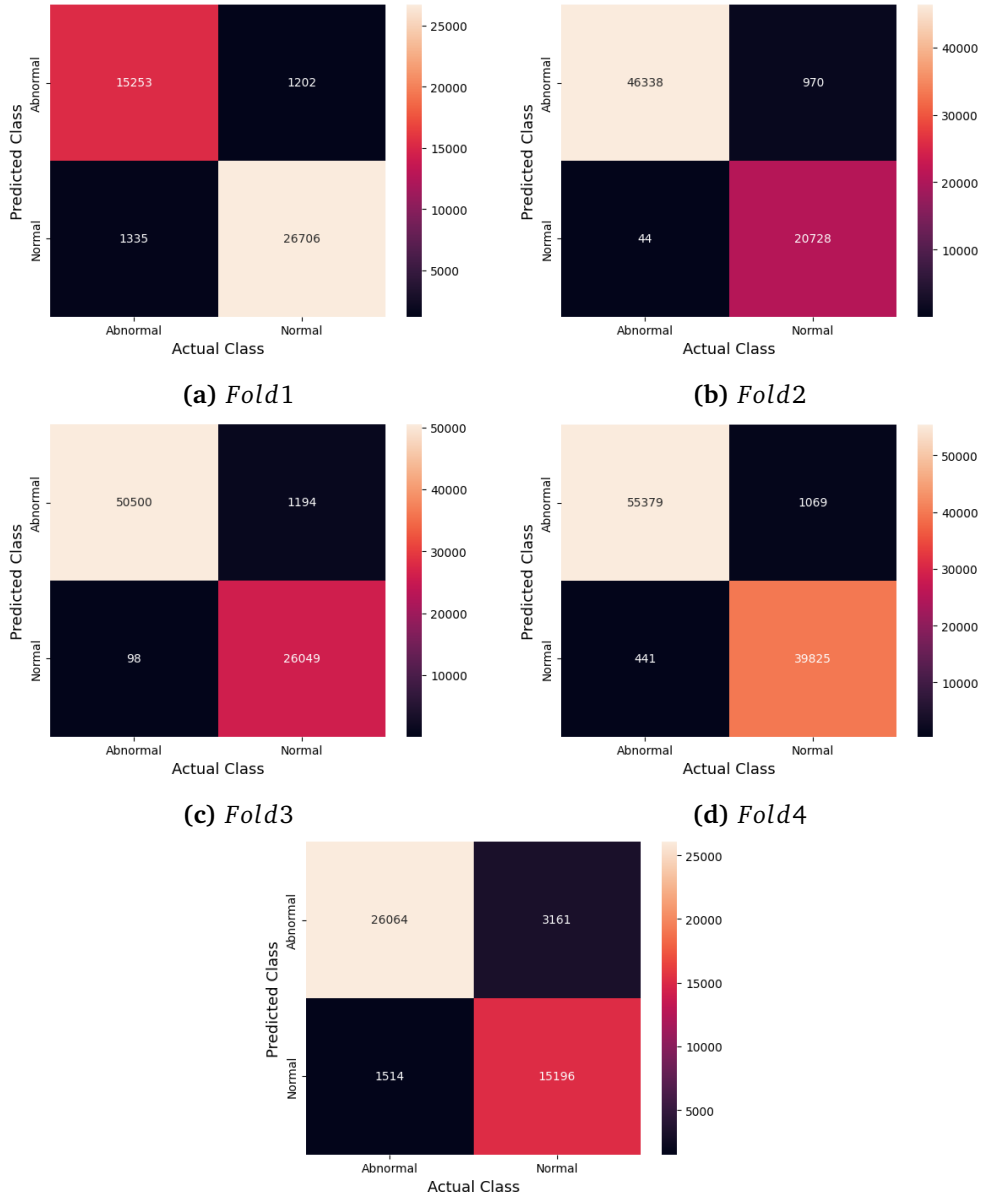


Figure A.25: Total Confusion Matrix for all 5 folds for Node, Content and Context Features



(a) $Fold1$ **(b)** $Fold2$

(c) $Fold3$ **(d)** $Fold4$

(e) $Fold5$

Figure A.26: Node, Content and Context features - All 5 folds

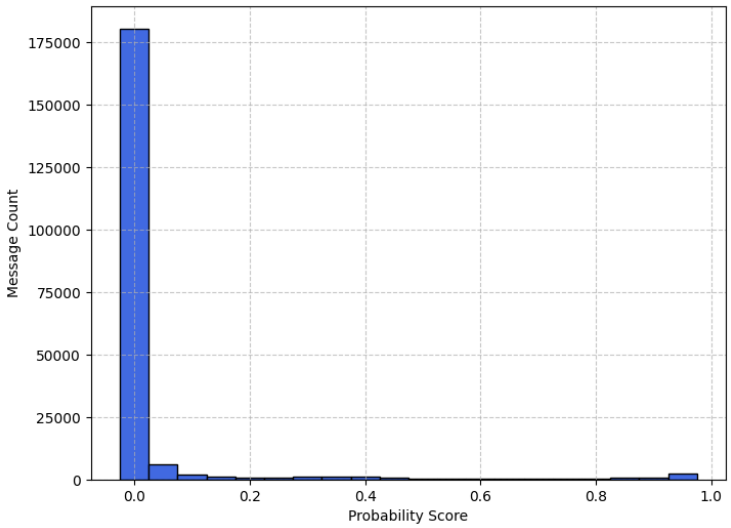


Figure A.27: Distribution of Probability Score - Normal Users - Node, Content and Context Features

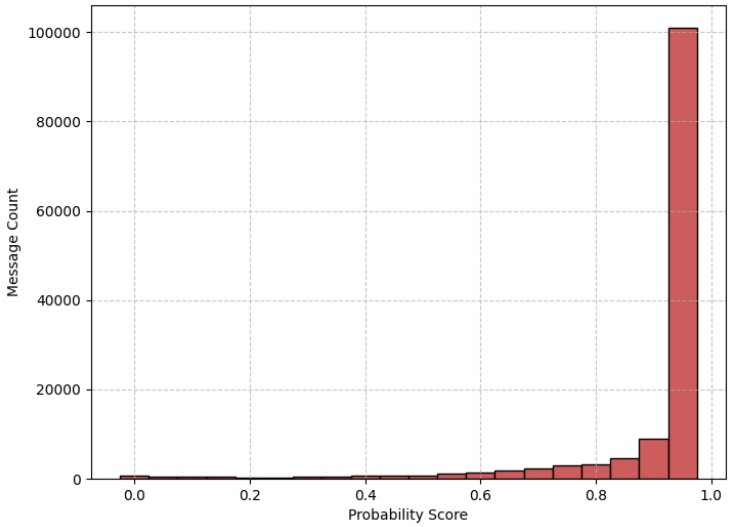


Figure A.28: Distribution of Probability Score - Abnormal Users - Node, Content and Context Features

Appendix B

Probability timelines - Three best performing feature set combinations

B.1 Node, Content, and Context

B.1.1 All Normal Users

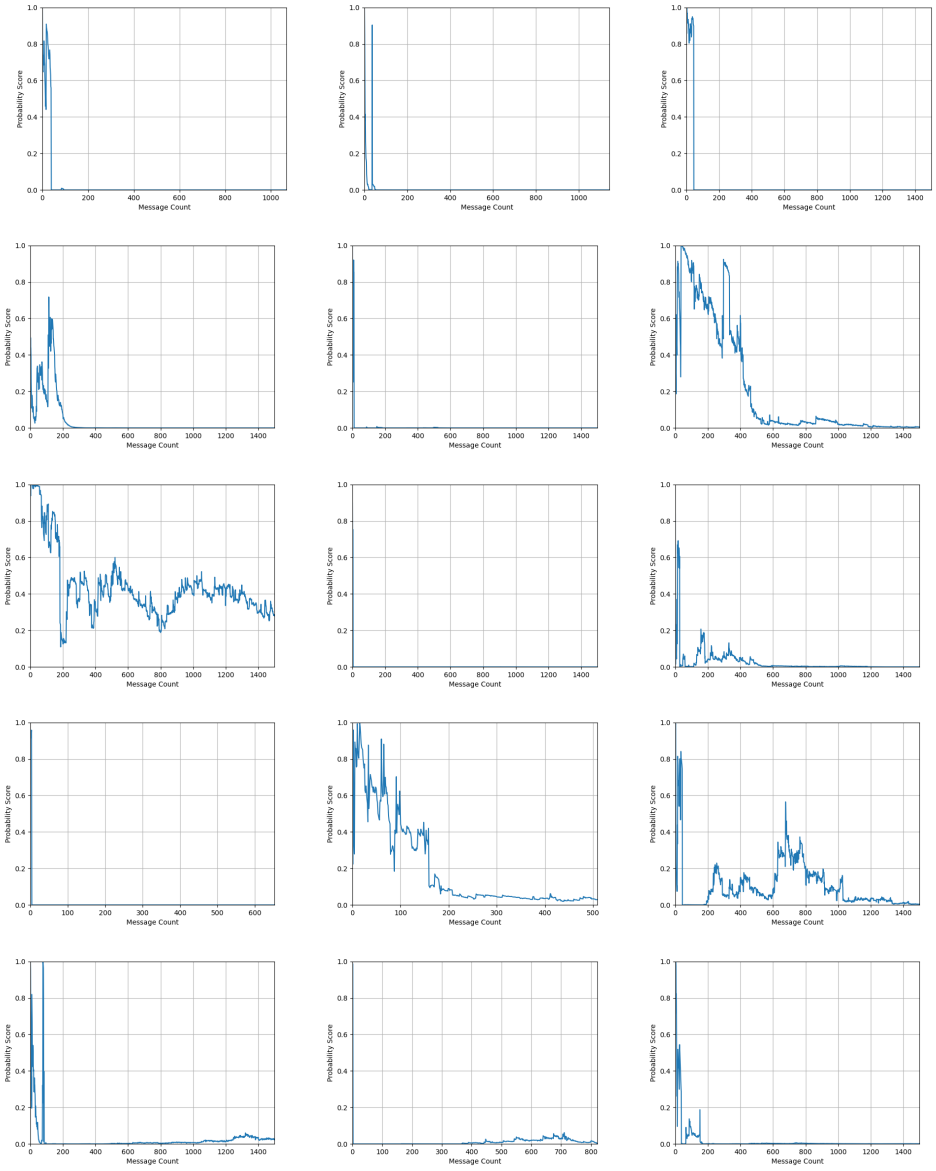


Figure B.1: Probability timelines for normal users, after SVM - Node, Context and Content

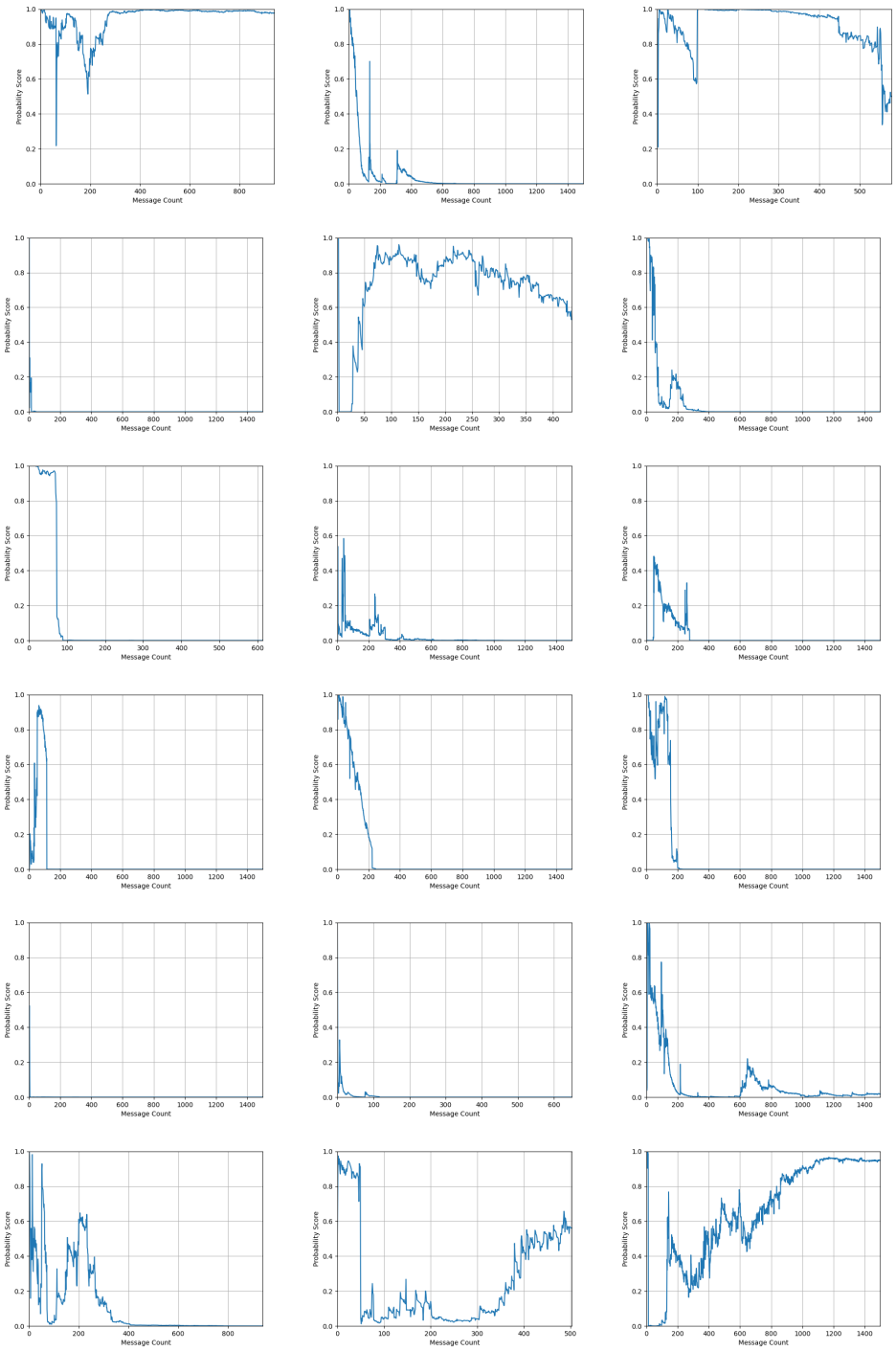


Figure B.2: Probability timelines for normal users, after SVM - Node, Context and Content

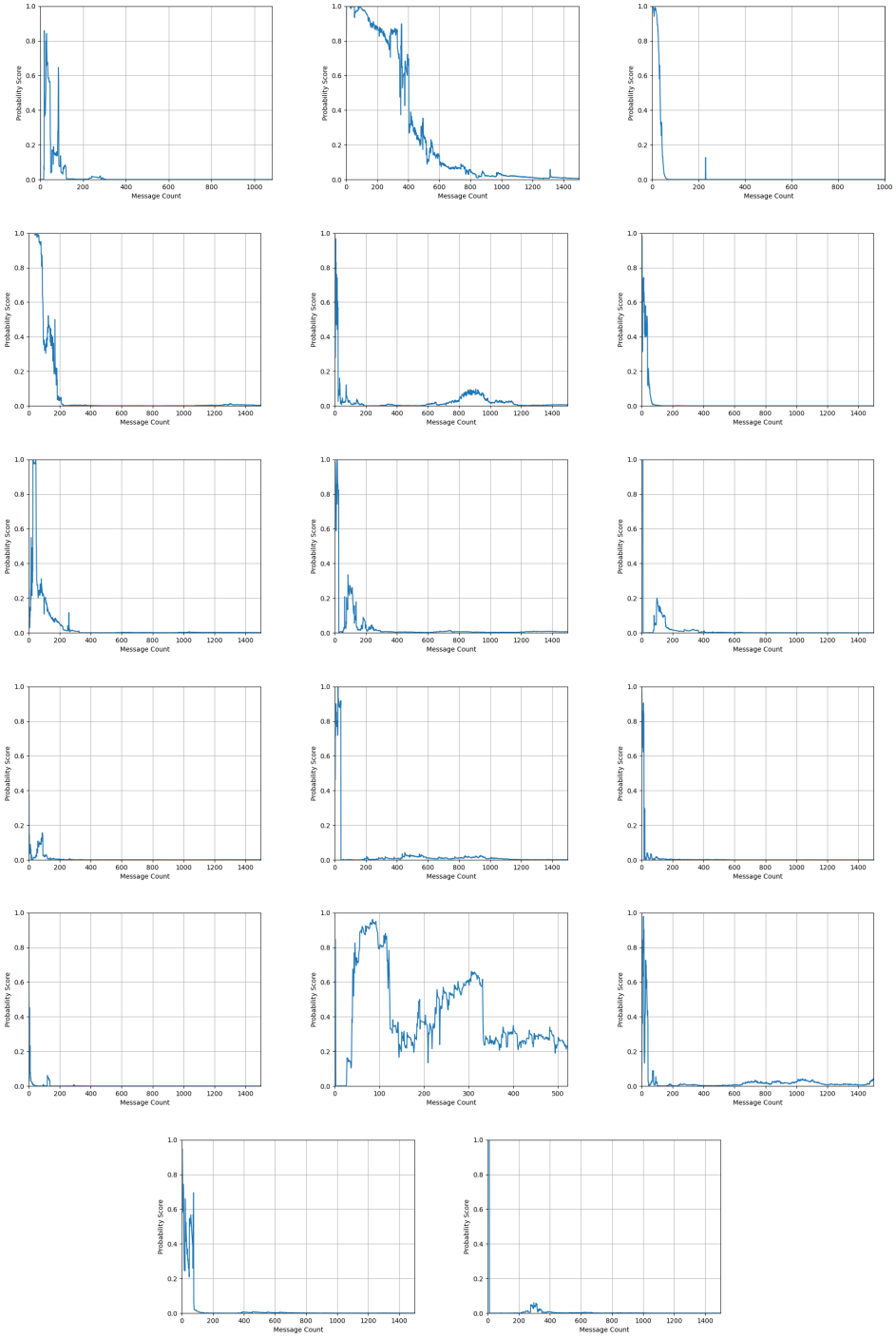


Figure B.3: Probability timelines for normal users, after SVM - Node, Context and Content

B.1.2 All Abnormal Users

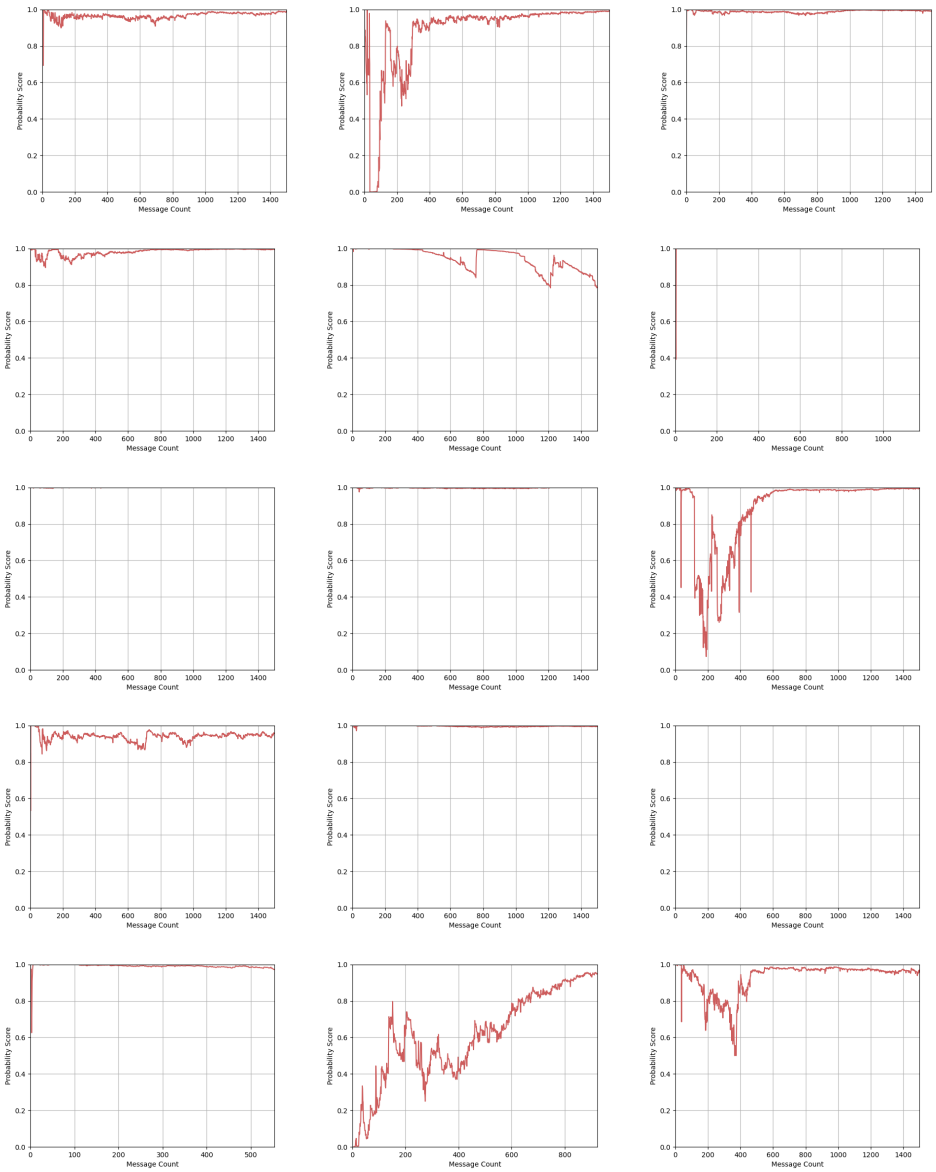


Figure B.4: Probability timelines for abnormal users, after SVM - Node, Context and Content

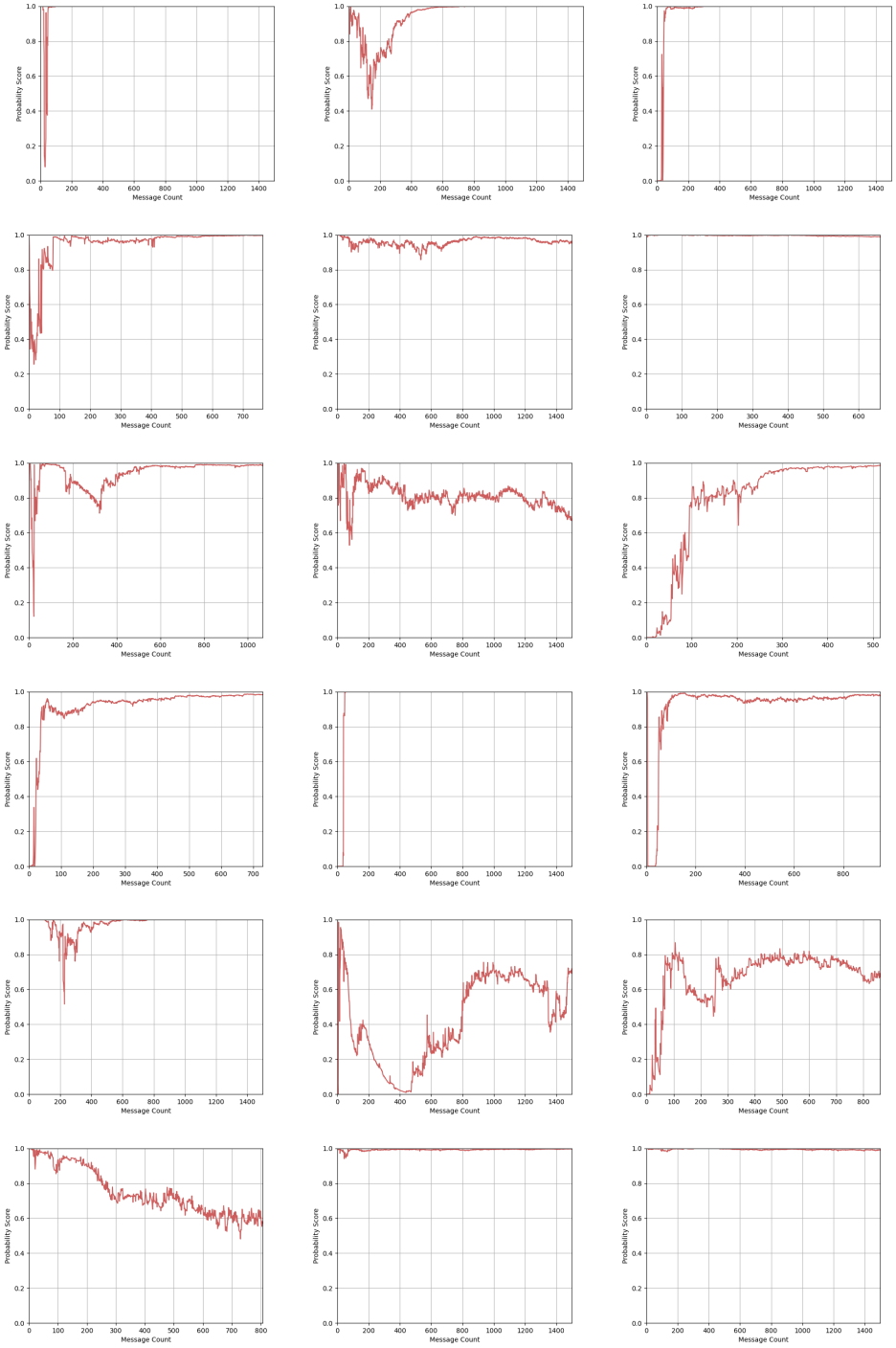


Figure B.5: Probability timelines for abnormal users, after SVM - Node, Context and Content

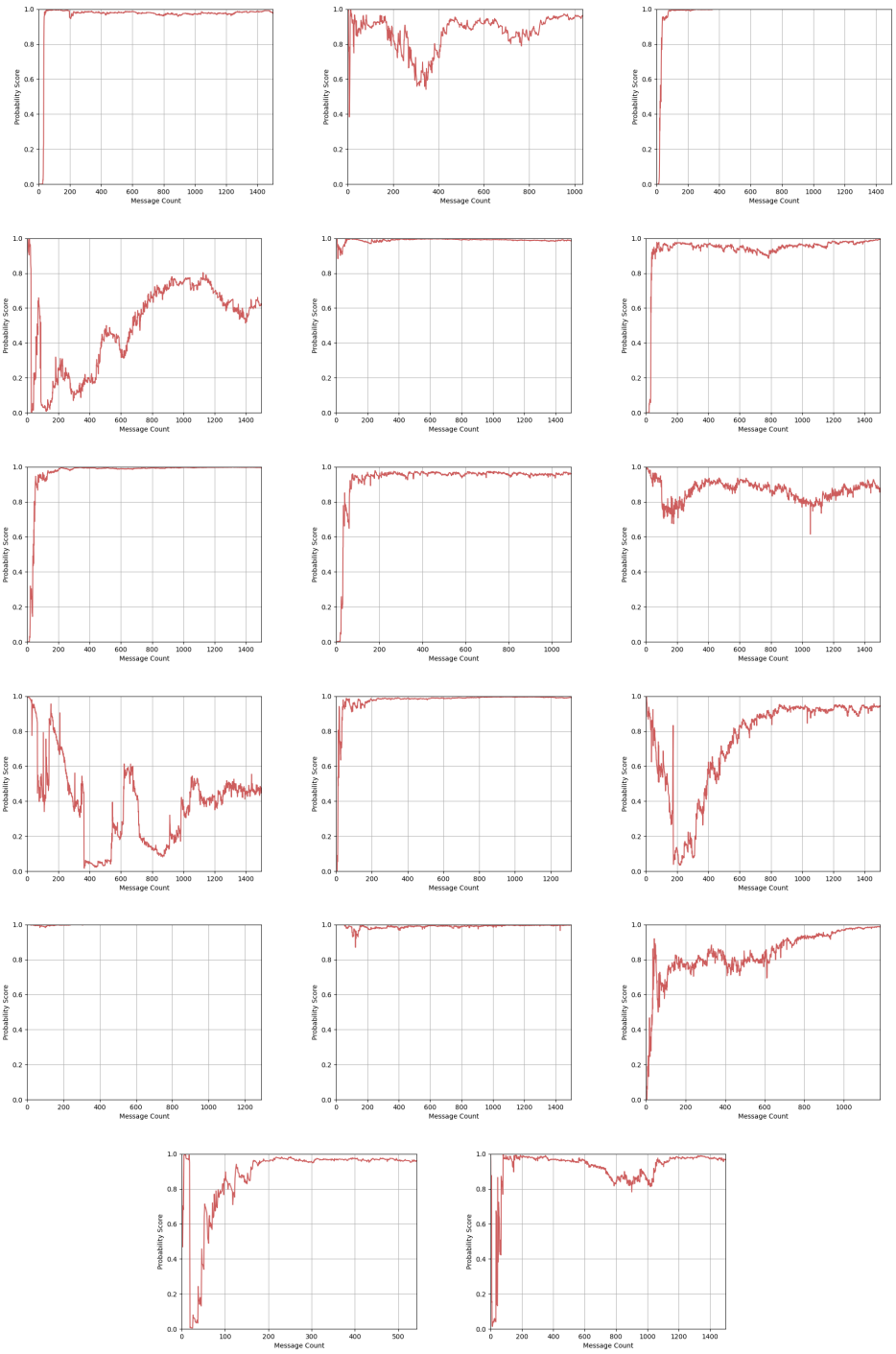


Figure B.6: Probability timelines for abnormal users, after SVM - Node, Context and Content

B.2 Node and Context

B.2.1 All Normal Users

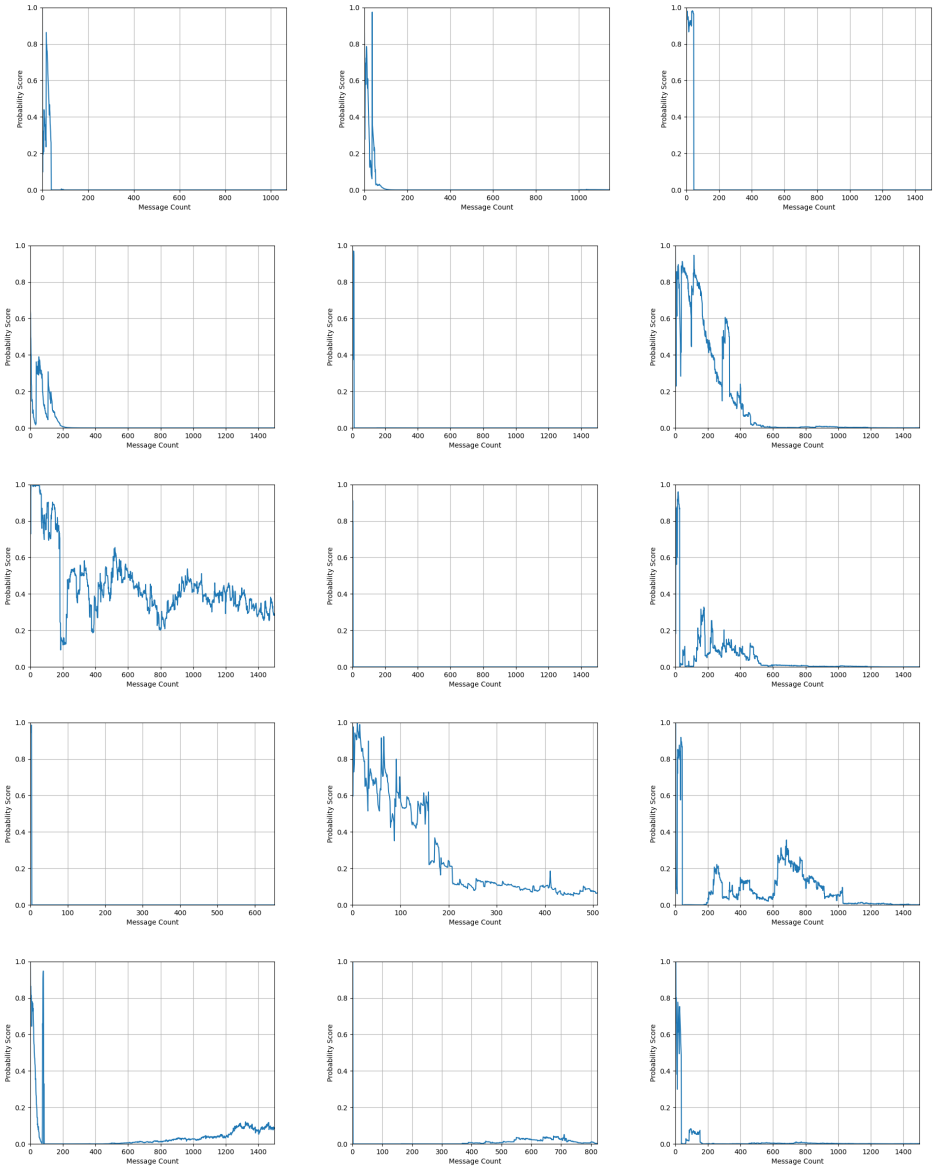


Figure B.7: Probability timelines for normal users, after SVM - Node and Context

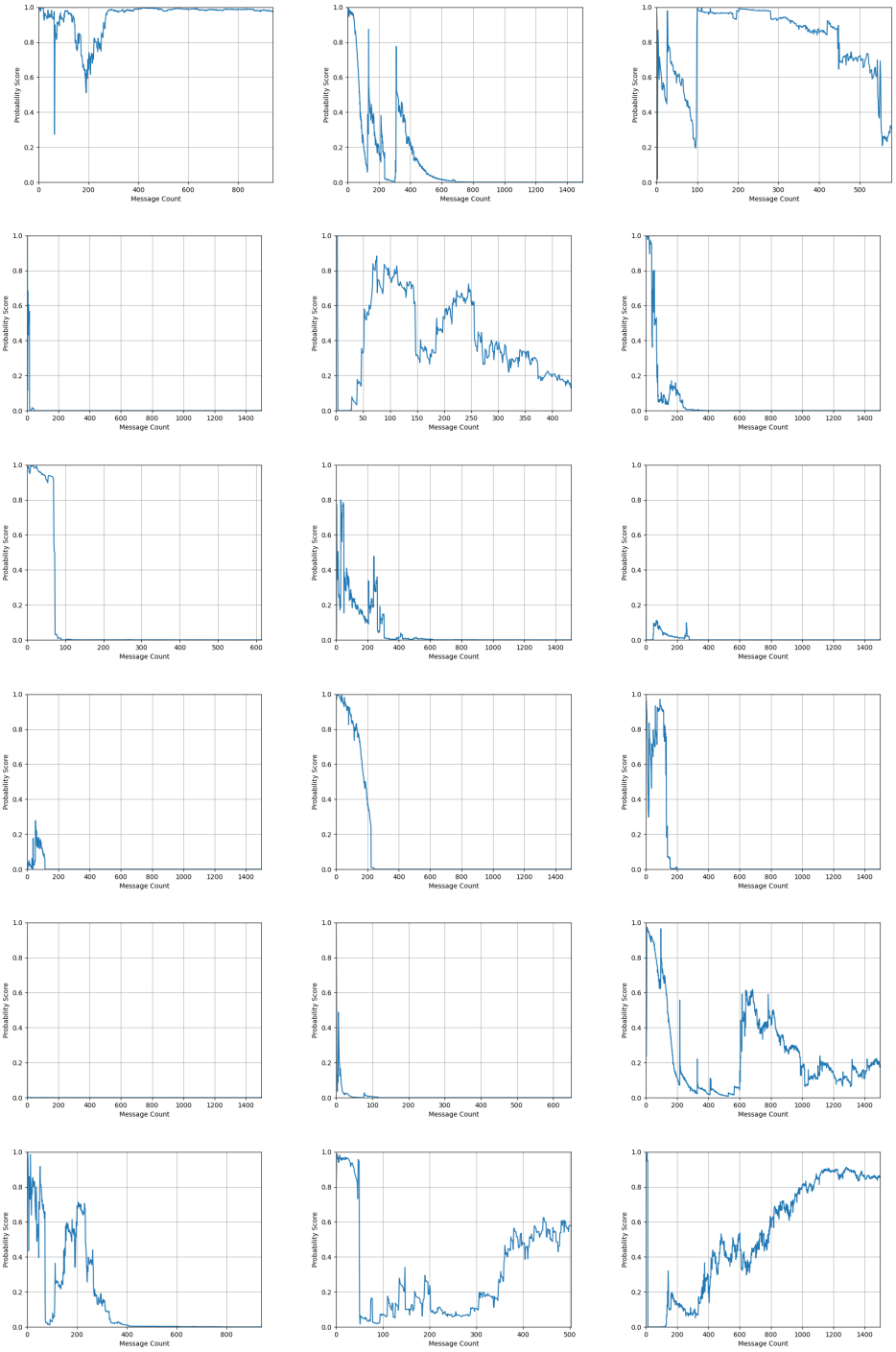


Figure B.8: Probability timelines for normal users, after SVM - Node and Context

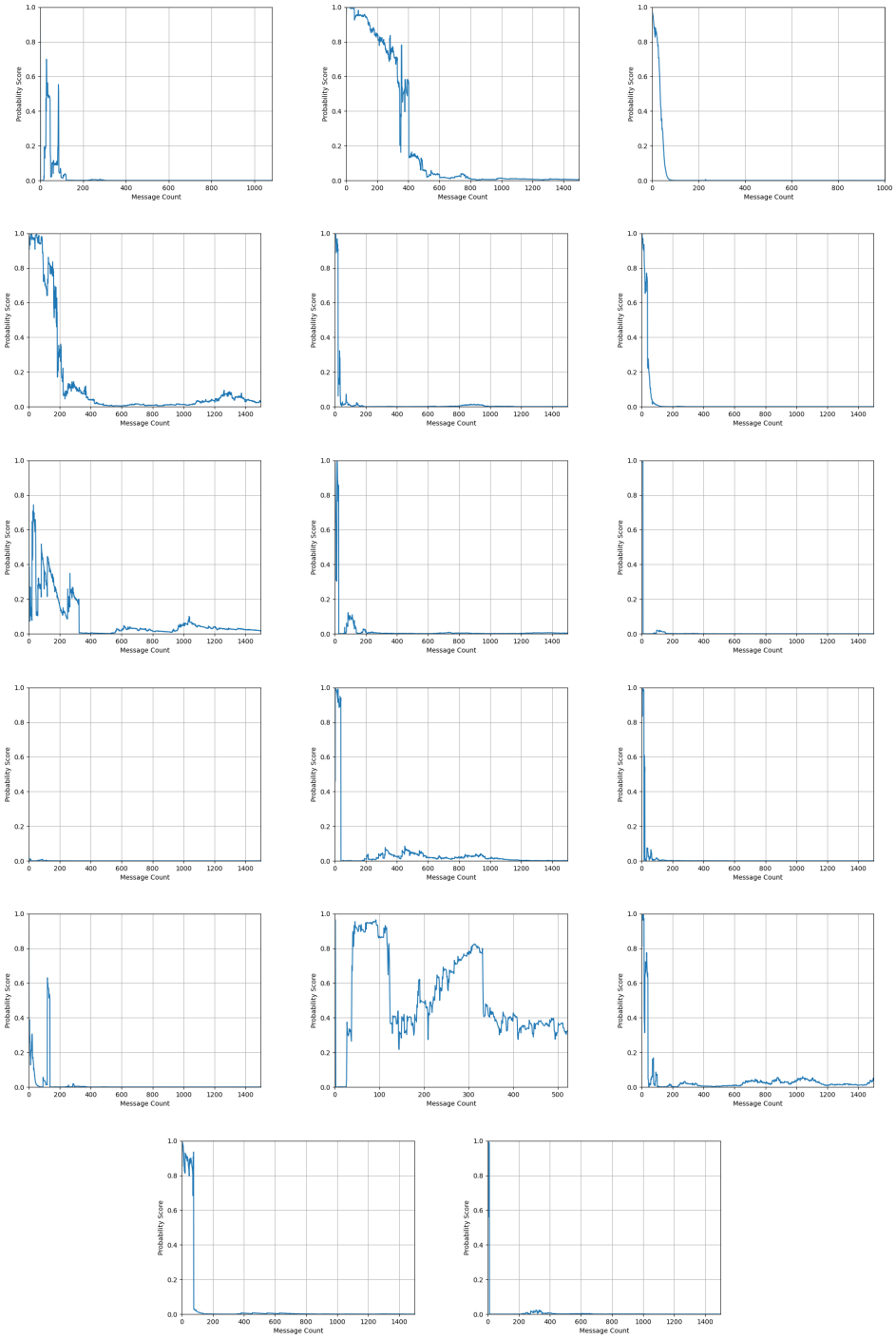


Figure B.9: Probability timelines for normal users, after SVM - Node and Context

B.2.2 All Abnormal Users

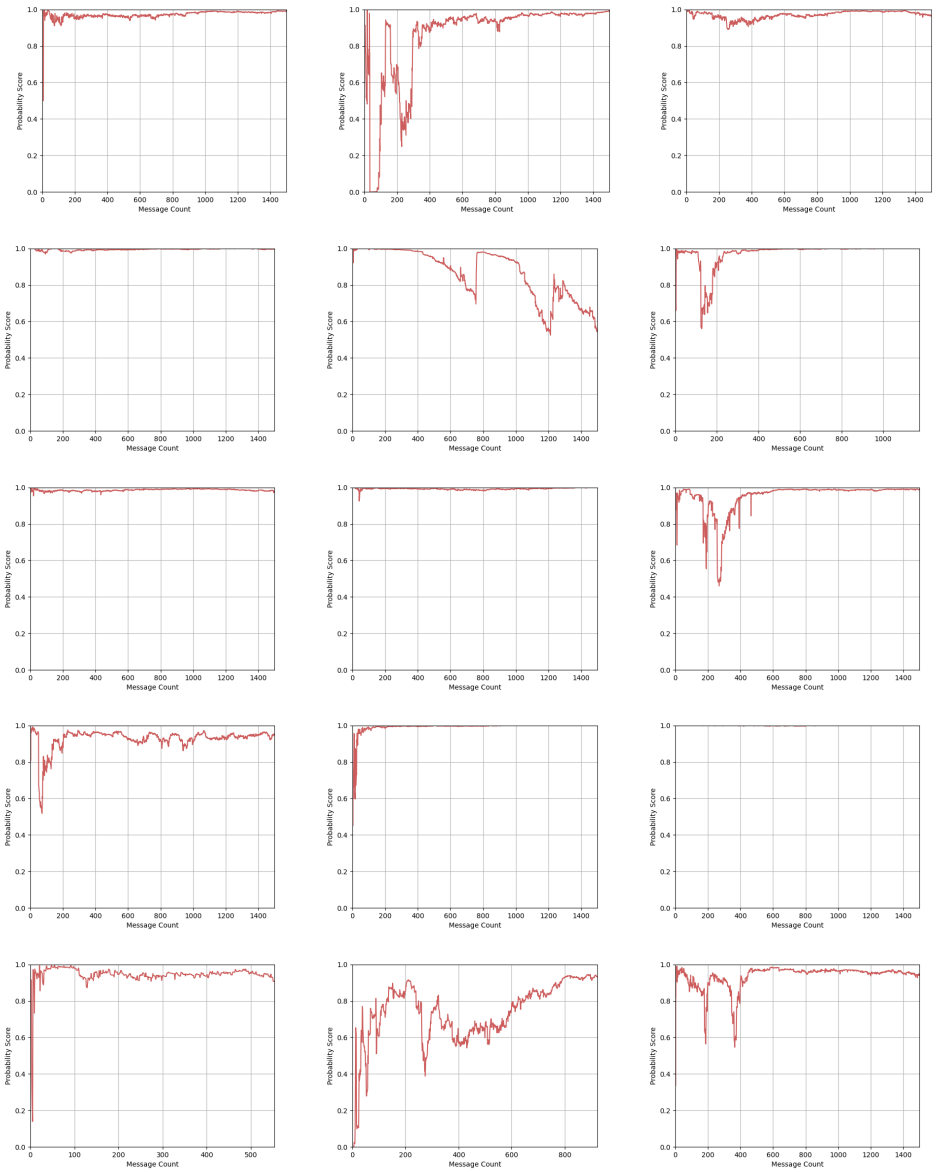


Figure B.10: Probability timelines for abnormal users, after SVM - Node and Context

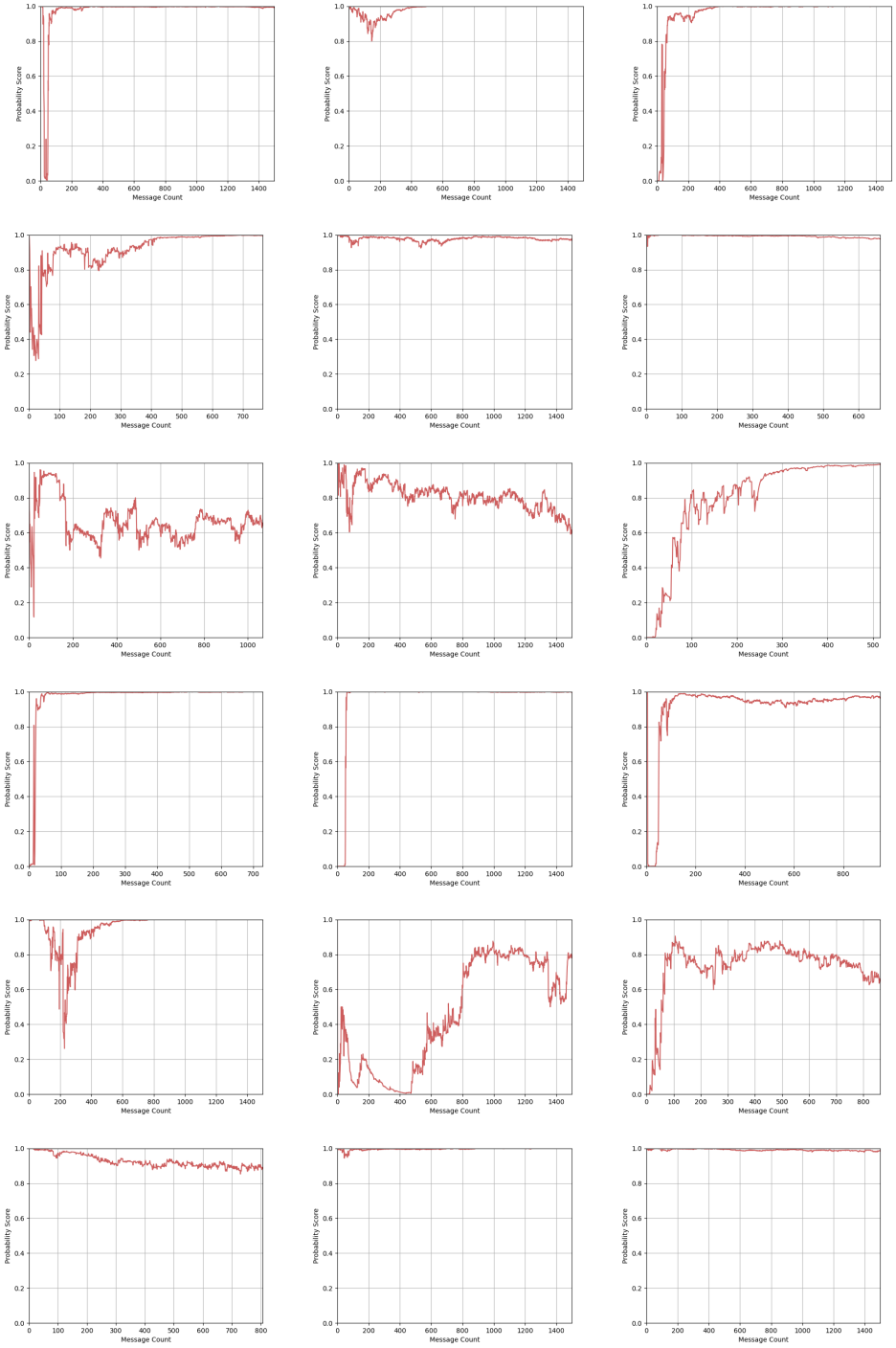


Figure B.11: Probability timelines for abnormal users, after SVM - Node and Context

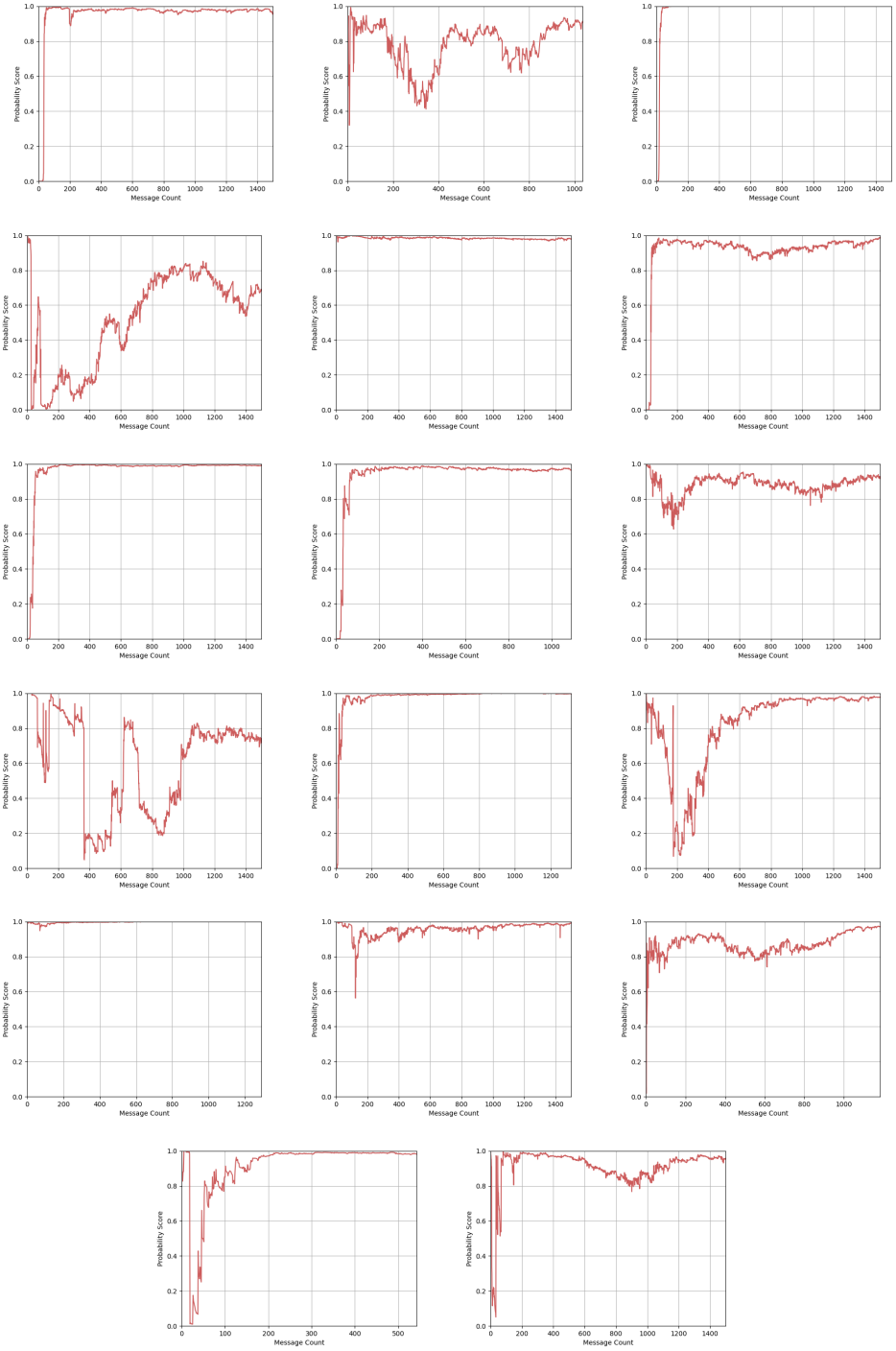


Figure B.12: Probability timelines for abnormal users, after SVM - Node and Context

B.3 Node

B.3.1 All Normal Users

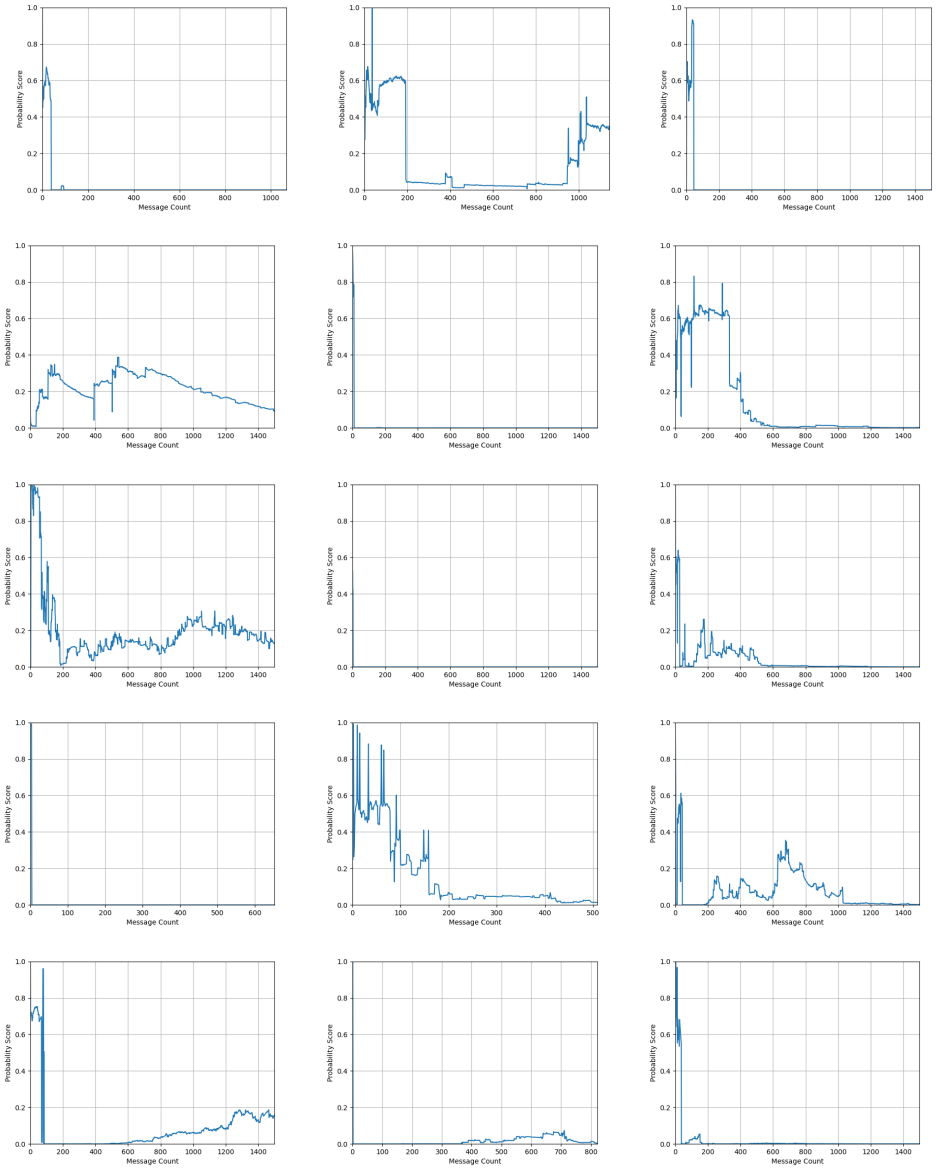


Figure B.13: Probability timelines for normal users, after SVM - Node

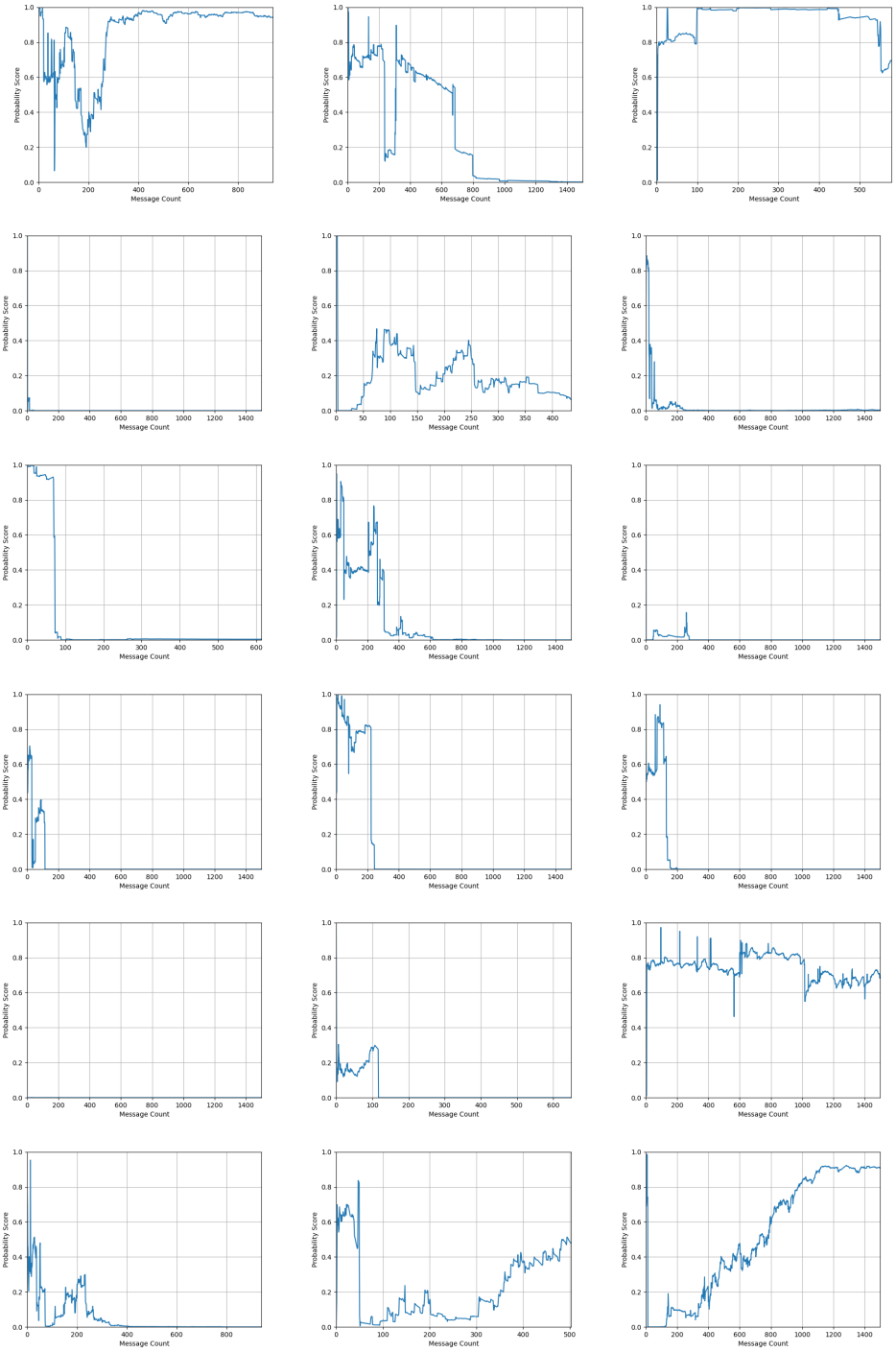


Figure B.14: Probability timelines for normal users, after SVM - Node

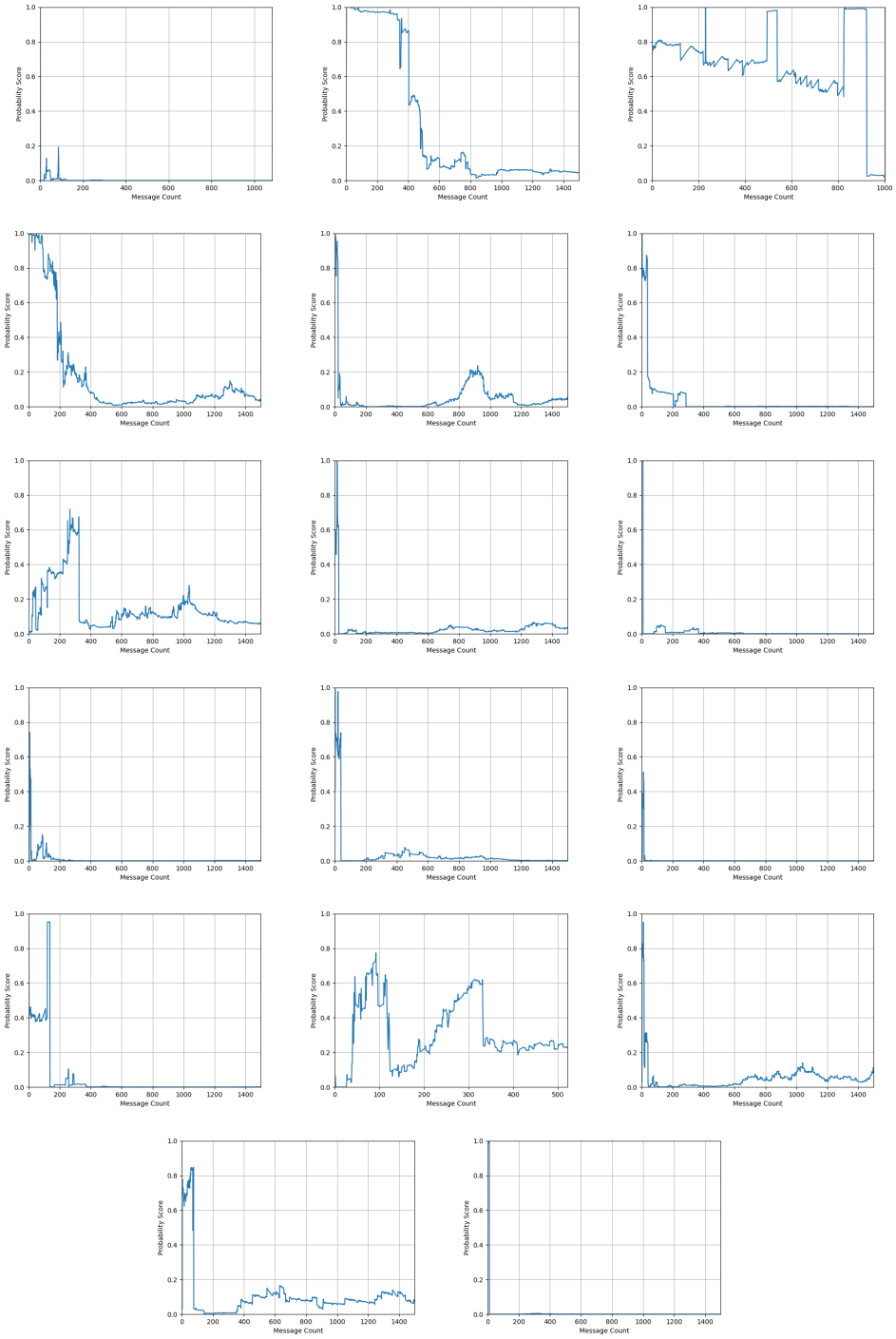


Figure B.15: Probability timelines for normal users, after SVM - Node

B.3.2 All Abnormal Users

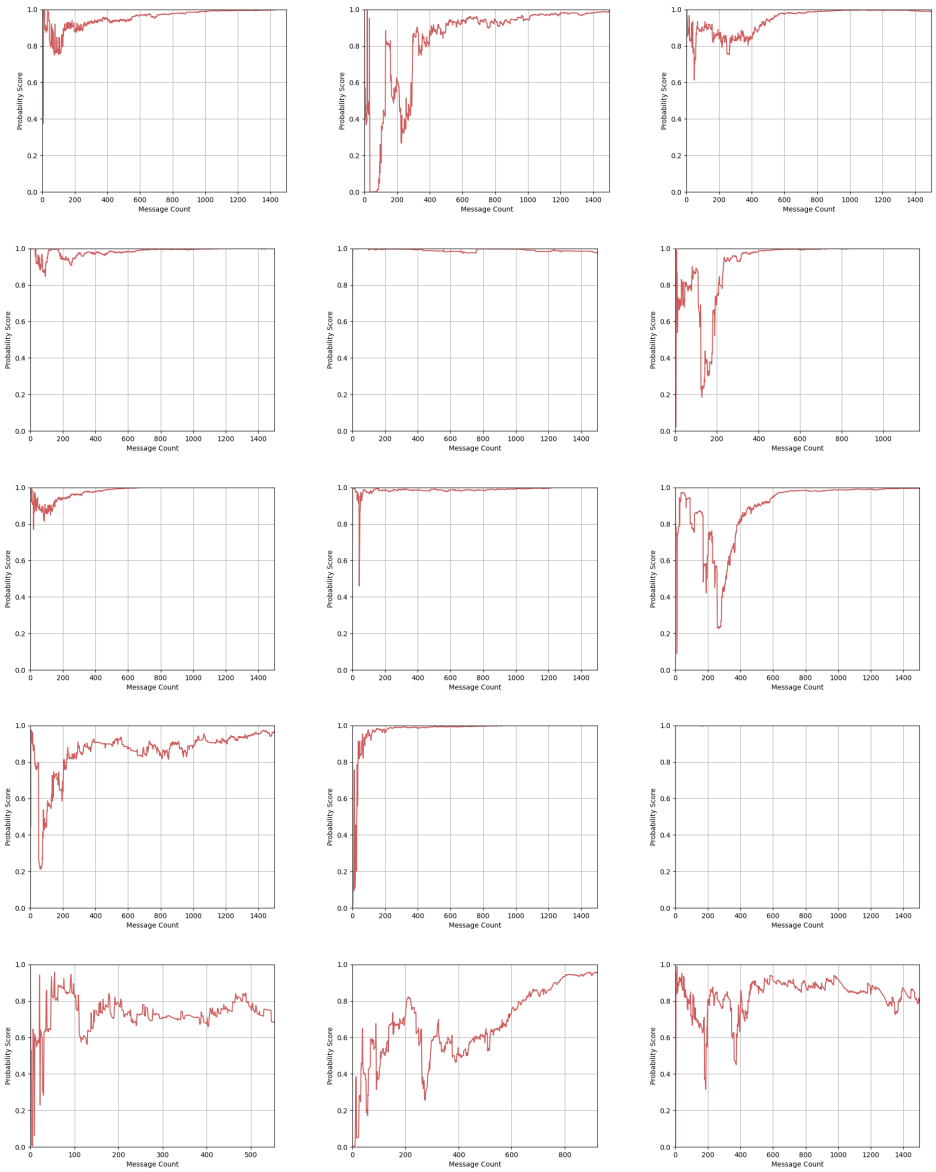


Figure B.16: Probability timelines for abnormal users, after SVM - Node

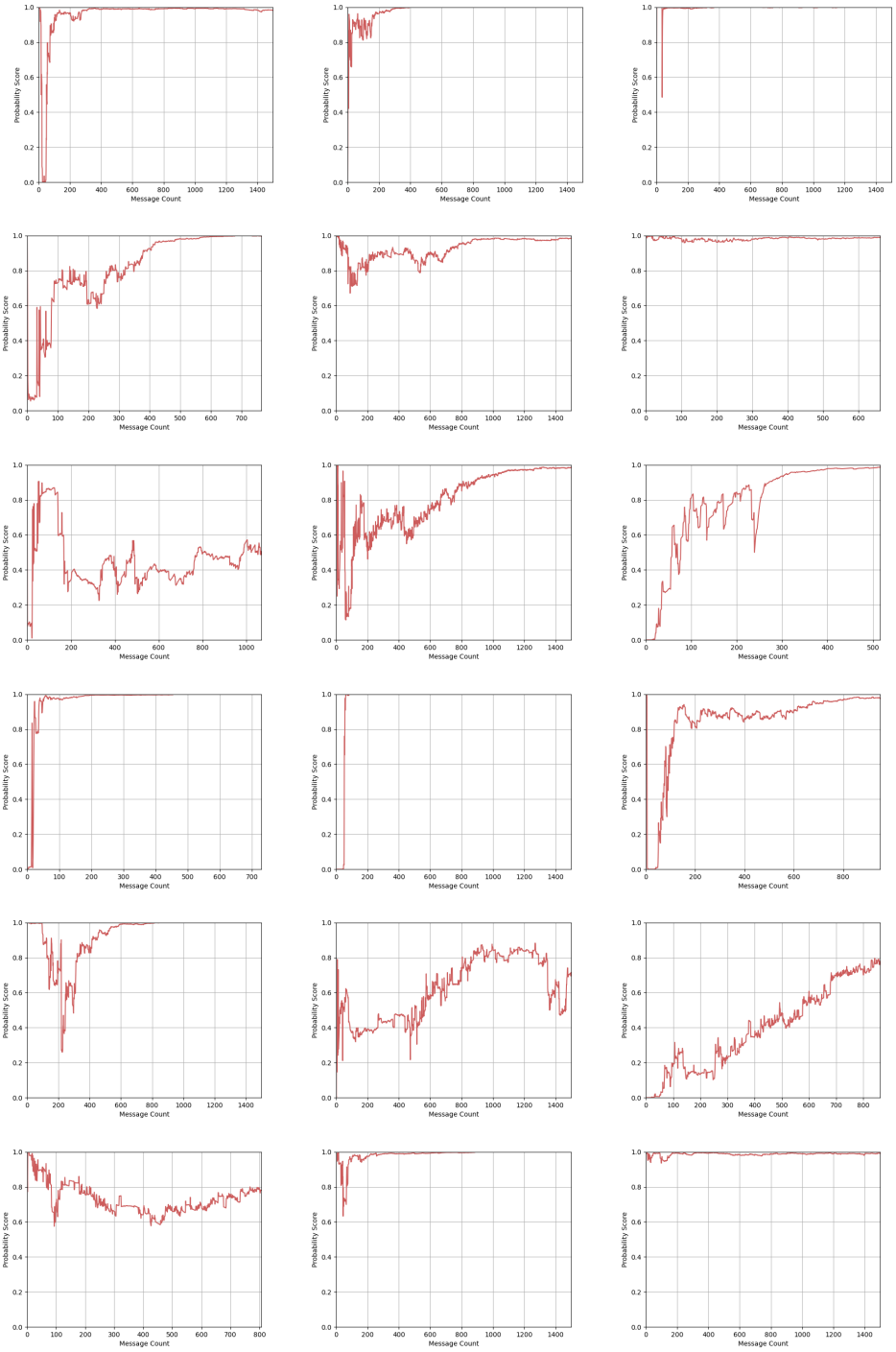


Figure B.17: Probability timelines for abnormal users, after SVM - Node

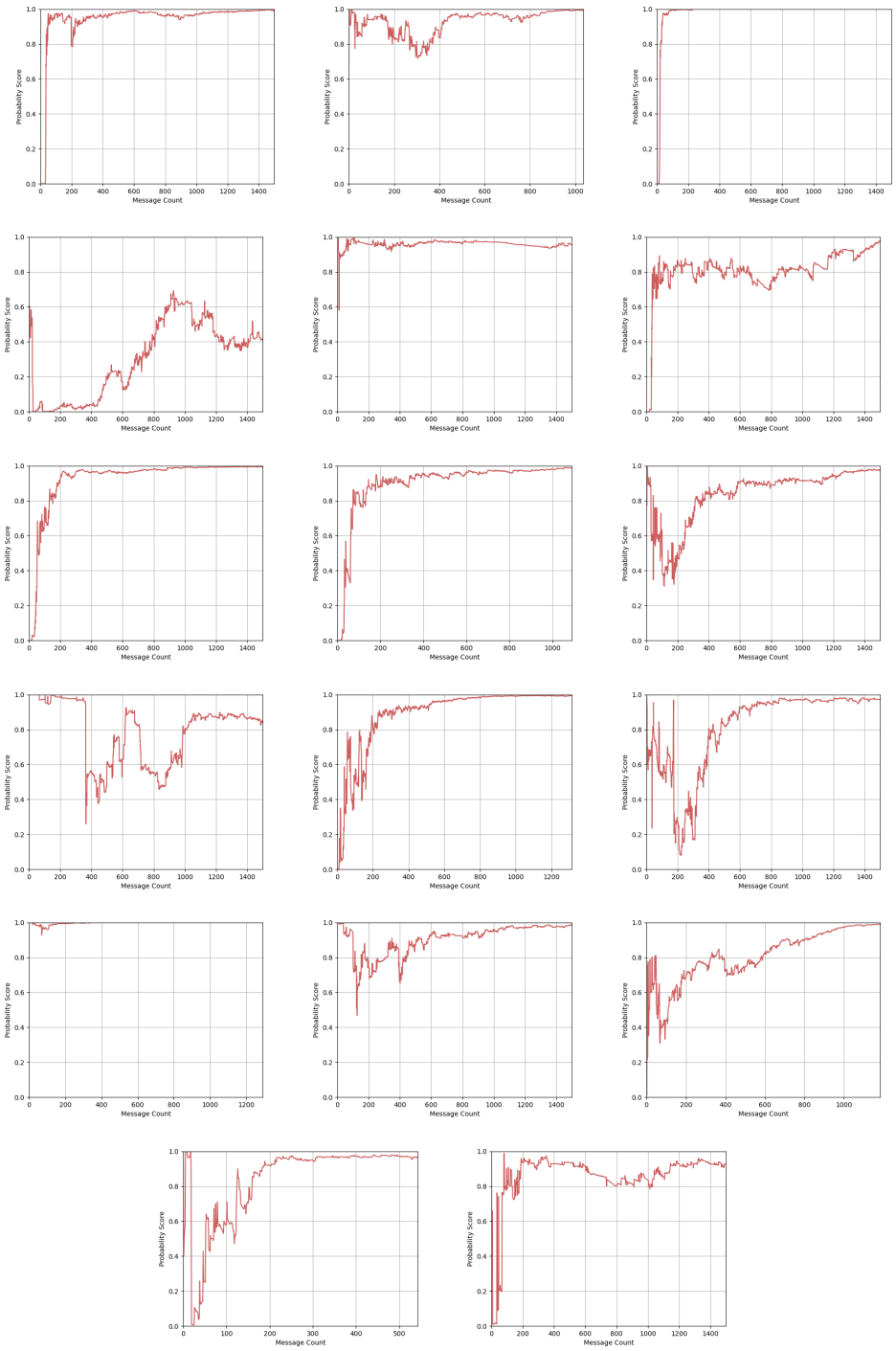


Figure B.18: Probability timelines for abnormal users, after SVM - Node

Appendix C

Results from Dynamic Detection Mechanism

C.1 Node

Table C.1: Node - Dynamic Detection Mechanism Results

Start Msg	Win Size	STAB	DEC	Avg Msg	PR	RC	ACC	$F_{0.5}$	F_1	F_2
0	10	0.01	0.25	31.24	0.585	0.620	0.590	0.592	0.602	0.613
0	10	0.01	0.50	31.24	0.604	0.580	0.600	0.599	0.592	0.585
0	10	0.01	0.75	31.24	0.703	0.520	0.650	0.657	0.598	0.549
0	10	0.02	0.25	20.82	0.517	0.620	0.520	0.534	0.564	0.596
0	10	0.02	0.50	20.82	0.536	0.600	0.540	0.547	0.566	0.586
0	10	0.02	0.75	20.82	0.735	0.500	0.660	0.672	0.595	0.534
0	10	0.04	0.25	16.55	0.515	0.700	0.520	0.543	0.593	0.653
0	10	0.04	0.50	16.55	0.525	0.620	0.530	0.542	0.569	0.598
0	10	0.04	0.75	16.55	0.703	0.520	0.650	0.657	0.598	0.549
0	20	0.01	0.25	75.55	0.679	0.720	0.690	0.687	0.699	0.711
0	20	0.01	0.50	75.55	0.708	0.680	0.700	0.702	0.694	0.685
0	20	0.01	0.75	75.55	0.765	0.520	0.680	0.699	0.619	0.556
0	20	0.02	0.25	49.52	0.667	0.720	0.680	0.677	0.692	0.709
0	20	0.02	0.50	49.52	0.702	0.660	0.690	0.693	0.680	0.668
0	20	0.02	0.75	49.52	0.811	0.600	0.730	0.758	0.690	0.633
0	20	0.04	0.25	35.03	0.574	0.700	0.590	0.595	0.631	0.670
0	20	0.04	0.50	35.03	0.608	0.620	0.610	0.610	0.614	0.618
0	20	0.04	0.75	35.03	0.778	0.560	0.700	0.722	0.651	0.593
0	30	0.01	0.25	140.49	0.768	0.860	0.800	0.785	0.811	0.840
0	30	0.01	0.50	140.49	0.792	0.840	0.810	0.802	0.816	0.830
0	30	0.01	0.75	140.49	0.864	0.760	0.820	0.841	0.809	0.779
0	30	0.02	0.25	88.63	0.712	0.840	0.750	0.734	0.771	0.811
0	30	0.02	0.50	88.63	0.745	0.820	0.770	0.759	0.781	0.804
0	30	0.02	0.75	88.63	0.800	0.640	0.740	0.762	0.711	0.667
0	30	0.04	0.25	60.09	0.678	0.800	0.710	0.699	0.734	0.772
0	30	0.04	0.50	60.09	0.714	0.700	0.710	0.711	0.707	0.703
0	30	0.04	0.75	60.09	0.800	0.560	0.710	0.737	0.659	0.596
0	40	0.01	0.25	175.78	0.776	0.900	0.820	0.798	0.833	0.872
0	40	0.01	0.50	175.78	0.815	0.880	0.840	0.827	0.846	0.866
0	40	0.01	0.75	175.78	0.851	0.800	0.830	0.840	0.825	0.810
0	40	0.02	0.25	117.58	0.738	0.900	0.790	0.765	0.811	0.862
0	40	0.02	0.50	117.58	0.786	0.880	0.820	0.803	0.830	0.859
0	40	0.02	0.75	117.58	0.848	0.780	0.820	0.833	0.813	0.793

Continued on next page

Table C.1: Node - Dynamic Detection Mechanism Results continued

Start Msg	Win Size	STAB	DEC	Avg Msg	PR	RC	ACC	$F_{0.5}$	F_1	F_2
0	40	0.04	0.25	77.97	0.710	0.880	0.760	0.738	0.786	0.840
0	40	0.04	0.50	77.97	0.750	0.840	0.780	0.766	0.792	0.820
0	40	0.04	0.75	77.97	0.789	0.600	0.720	0.743	0.682	0.630
0	50	0.01	0.25	205.93	0.807	0.920	0.850	0.827	0.860	0.895
0	50	0.01	0.50	205.93	0.833	0.900	0.860	0.846	0.865	0.886
0	50	0.01	0.75	205.93	0.894	0.840	0.870	0.882	0.866	0.850
0	50	0.02	0.25	145.55	0.767	0.920	0.820	0.793	0.836	0.885
0	50	0.02	0.50	145.55	0.815	0.880	0.840	0.827	0.846	0.866
0	50	0.02	0.75	145.55	0.889	0.800	0.850	0.870	0.842	0.816
0	50	0.04	0.25	97.64	0.708	0.920	0.770	0.742	0.800	0.868
0	50	0.04	0.50	97.64	0.772	0.880	0.810	0.791	0.822	0.856
0	50	0.04	0.75	97.64	0.857	0.600	0.750	0.789	0.706	0.638
50	10	0.01	0.25	72.84	0.698	0.880	0.750	0.728	0.779	0.837
50	10	0.01	0.50	72.84	0.714	0.800	0.740	0.730	0.755	0.781
50	10	0.01	0.75	72.84	0.800	0.640	0.740	0.762	0.711	0.667
50	10	0.02	0.25	67.58	0.703	0.900	0.760	0.735	0.789	0.852
50	10	0.02	0.50	67.58	0.727	0.800	0.750	0.741	0.762	0.784
50	10	0.02	0.75	67.58	0.795	0.620	0.730	0.752	0.697	0.649
50	10	0.04	0.25	65.01	0.708	0.920	0.770	0.742	0.800	0.868
50	10	0.04	0.50	65.01	0.722	0.780	0.740	0.733	0.750	0.768
50	10	0.04	0.75	65.01	0.821	0.640	0.750	0.777	0.719	0.669
50	20	0.01	0.25	117.32	0.734	0.940	0.800	0.768	0.825	0.890
50	20	0.01	0.50	117.32	0.789	0.900	0.830	0.809	0.841	0.875
50	20	0.01	0.75	117.32	0.818	0.720	0.780	0.796	0.766	0.738
50	20	0.02	0.25	93.17	0.746	0.940	0.810	0.778	0.832	0.894
50	20	0.02	0.50	93.17	0.782	0.860	0.810	0.796	0.819	0.843
50	20	0.02	0.75	93.17	0.822	0.740	0.790	0.804	0.779	0.755
50	20	0.04	0.25	80.44	0.719	0.920	0.780	0.752	0.807	0.871
50	20	0.04	0.50	80.44	0.759	0.820	0.780	0.771	0.788	0.807
50	20	0.04	0.75	80.44	0.814	0.700	0.770	0.788	0.753	0.720
50	30	0.01	0.25	170.88	0.810	0.940	0.860	0.833	0.870	0.911
50	30	0.01	0.50	170.88	0.836	0.920	0.870	0.852	0.876	0.902
50	30	0.01	0.75	170.88	0.875	0.840	0.860	0.868	0.857	0.847
50	30	0.02	0.25	122.67	0.783	0.940	0.840	0.810	0.855	0.904
50	30	0.02	0.50	122.67	0.821	0.920	0.860	0.839	0.868	0.898
50	30	0.02	0.75	122.67	0.864	0.760	0.820	0.841	0.809	0.779
50	30	0.04	0.25	97.10	0.746	0.940	0.810	0.778	0.832	0.894
50	30	0.04	0.50	97.10	0.824	0.840	0.830	0.827	0.832	0.837
50	30	0.04	0.75	97.10	0.816	0.620	0.740	0.767	0.705	0.651
50	40	0.01	0.25	199.24	0.800	0.960	0.860	0.828	0.873	0.923

Continued on next page

Table C.1: Node - Dynamic Detection Mechanism Results continued

Start Msg	Win Size	STAB	DEC	Avg Msg	PR	RC	ACC	$F_{0.5}$	F_1	F_2
50	40	0.01	0.50	199.24	0.839	0.940	0.880	0.858	0.887	0.918
50	40	0.01	0.75	199.24	0.878	0.860	0.870	0.874	0.869	0.863
50	40	0.02	0.25	146.09	0.783	0.940	0.840	0.810	0.855	0.904
50	40	0.02	0.50	146.09	0.852	0.920	0.880	0.865	0.885	0.906
50	40	0.02	0.75	146.09	0.891	0.820	0.860	0.876	0.854	0.833
50	40	0.04	0.25	110.32	0.758	0.940	0.820	0.789	0.839	0.897
50	40	0.04	0.50	110.32	0.818	0.900	0.850	0.833	0.857	0.882
50	40	0.04	0.75	110.32	0.795	0.620	0.730	0.752	0.697	0.649
50	50	0.01	0.25	228.48	0.800	0.960	0.860	0.828	0.873	0.923
50	50	0.01	0.50	228.48	0.839	0.940	0.880	0.858	0.887	0.918
50	50	0.01	0.75	228.48	0.898	0.880	0.890	0.894	0.889	0.884
50	50	0.02	0.25	172.13	0.814	0.960	0.870	0.839	0.881	0.927
50	50	0.02	0.50	172.13	0.868	0.920	0.890	0.878	0.893	0.909
50	50	0.02	0.75	172.13	0.894	0.840	0.870	0.882	0.866	0.850
50	50	0.04	0.25	132.05	0.750	0.960	0.820	0.784	0.842	0.909
50	50	0.04	0.50	132.05	0.852	0.920	0.880	0.865	0.885	0.906
50	50	0.04	0.75	132.05	0.861	0.620	0.760	0.799	0.721	0.657
100	10	0.01	0.25	116.23	0.727	0.960	0.800	0.764	0.828	0.902
100	10	0.01	0.50	116.23	0.800	0.880	0.830	0.815	0.838	0.863
100	10	0.01	0.75	116.23	0.842	0.640	0.760	0.792	0.727	0.672
100	10	0.02	0.25	113.42	0.727	0.960	0.800	0.764	0.828	0.902
100	10	0.02	0.50	113.42	0.792	0.840	0.810	0.802	0.816	0.830
100	10	0.02	0.75	113.42	0.825	0.660	0.760	0.786	0.733	0.688
100	10	0.04	0.25	111.90	0.727	0.960	0.800	0.764	0.828	0.902
100	10	0.04	0.50	111.90	0.796	0.860	0.820	0.808	0.827	0.846
100	10	0.04	0.75	111.90	0.821	0.640	0.750	0.777	0.719	0.669
100	20	0.01	0.25	158.00	0.787	0.960	0.850	0.816	0.865	0.920
100	20	0.01	0.50	158.00	0.821	0.920	0.860	0.839	0.868	0.898
100	20	0.01	0.75	158.00	0.864	0.760	0.820	0.841	0.809	0.779
100	20	0.02	0.25	135.81	0.778	0.980	0.850	0.811	0.867	0.932
100	20	0.02	0.50	135.81	0.818	0.900	0.850	0.833	0.857	0.882
100	20	0.02	0.75	135.81	0.884	0.760	0.830	0.856	0.817	0.782
100	20	0.04	0.25	124.02	0.783	0.940	0.840	0.810	0.855	0.904
100	20	0.04	0.50	124.02	0.804	0.820	0.810	0.807	0.812	0.817
100	20	0.04	0.75	124.02	0.838	0.620	0.750	0.783	0.713	0.654
100	30	0.01	0.25	205.53	0.800	0.960	0.860	0.828	0.873	0.923
100	30	0.01	0.50	205.53	0.836	0.920	0.870	0.852	0.876	0.902
100	30	0.01	0.75	205.53	0.894	0.840	0.870	0.882	0.866	0.850
100	30	0.02	0.25	161.59	0.787	0.960	0.850	0.816	0.865	0.920
100	30	0.02	0.50	161.59	0.839	0.940	0.880	0.858	0.887	0.918

Continued on next page

Table C.1: Node - Dynamic Detection Mechanism Results continued

Start Msg	Win Size	STAB	DEC	Avg Msg	PR	RC	ACC	$F_{0.5}$	F_1	F_2
100	30	0.02	0.75	161.59	0.905	0.760	0.840	0.872	0.826	0.785
100	30	0.04	0.25	139.69	0.787	0.960	0.850	0.816	0.865	0.920
100	30	0.04	0.50	139.69	0.824	0.840	0.830	0.827	0.832	0.837
100	30	0.04	0.75	139.69	0.865	0.640	0.770	0.808	0.736	0.675
100	40	0.01	0.25	228.18	0.800	0.960	0.860	0.828	0.873	0.923
100	40	0.01	0.50	228.18	0.836	0.920	0.870	0.852	0.876	0.902
100	40	0.01	0.75	228.18	0.894	0.840	0.870	0.882	0.866	0.850
100	40	0.02	0.25	183.71	0.787	0.960	0.850	0.816	0.865	0.920
100	40	0.02	0.50	183.71	0.839	0.940	0.880	0.858	0.887	0.918
100	40	0.02	0.75	183.71	0.911	0.820	0.870	0.891	0.863	0.837
100	40	0.04	0.25	156.21	0.787	0.960	0.850	0.816	0.865	0.920
100	40	0.04	0.50	156.21	0.811	0.860	0.830	0.821	0.835	0.850
100	40	0.04	0.75	156.21	0.838	0.620	0.750	0.783	0.713	0.654
100	50	0.01	0.25	250.64	0.800	0.960	0.860	0.828	0.873	0.923
100	50	0.01	0.50	250.64	0.836	0.920	0.870	0.852	0.876	0.902
100	50	0.01	0.75	250.64	0.915	0.860	0.890	0.903	0.887	0.870
100	50	0.02	0.25	205.00	0.814	0.960	0.870	0.839	0.881	0.927
100	50	0.02	0.50	205.00	0.849	0.900	0.870	0.859	0.874	0.889
100	50	0.02	0.75	205.00	0.909	0.800	0.860	0.885	0.851	0.820
100	50	0.04	0.25	173.70	0.766	0.980	0.840	0.801	0.860	0.928
100	50	0.04	0.50	173.70	0.827	0.860	0.840	0.833	0.843	0.853
100	50	0.04	0.75	173.70	0.853	0.580	0.740	0.780	0.690	0.620
150	10	0.01	0.25	167.23	0.783	0.940	0.840	0.810	0.855	0.904
150	10	0.01	0.50	167.23	0.843	0.860	0.850	0.846	0.851	0.857
150	10	0.01	0.75	167.23	0.900	0.720	0.820	0.857	0.800	0.750
150	10	0.02	0.25	163.97	0.787	0.960	0.850	0.816	0.865	0.920
150	10	0.02	0.50	163.97	0.843	0.860	0.850	0.846	0.851	0.857
150	10	0.02	0.75	163.97	0.895	0.680	0.800	0.842	0.773	0.714
150	10	0.04	0.25	161.42	0.774	0.960	0.840	0.805	0.857	0.916
150	10	0.04	0.50	161.42	0.818	0.900	0.850	0.833	0.857	0.882
150	10	0.04	0.75	161.42	0.868	0.660	0.780	0.817	0.750	0.693
150	20	0.01	0.25	202.38	0.800	0.960	0.860	0.828	0.873	0.923
150	20	0.01	0.50	202.38	0.836	0.920	0.870	0.852	0.876	0.902
150	20	0.01	0.75	202.38	0.884	0.760	0.830	0.856	0.817	0.782
150	20	0.02	0.25	180.40	0.787	0.960	0.850	0.816	0.865	0.920
150	20	0.02	0.50	180.40	0.830	0.880	0.850	0.840	0.854	0.870
150	20	0.02	0.75	180.40	0.905	0.760	0.840	0.872	0.826	0.785
150	20	0.04	0.25	172.81	0.770	0.940	0.830	0.799	0.847	0.900
150	20	0.04	0.50	172.81	0.827	0.860	0.840	0.833	0.843	0.853
150	20	0.04	0.75	172.81	0.865	0.640	0.770	0.808	0.736	0.675

Continued on next page

Table C.1: Node - Dynamic Detection Mechanism Results continued

Start Msg	Win Size	STAB	DEC	Avg Msg	PR	RC	ACC	$F_{0.5}$	F_1	F_2
150	30	0.01	0.25	241.09	0.800	0.960	0.860	0.828	0.873	0.923
150	30	0.01	0.50	241.09	0.836	0.920	0.870	0.852	0.876	0.902
150	30	0.01	0.75	241.09	0.875	0.840	0.860	0.868	0.857	0.847
150	30	0.02	0.25	202.30	0.787	0.960	0.850	0.816	0.865	0.920
150	30	0.02	0.50	202.30	0.836	0.920	0.870	0.852	0.876	0.902
150	30	0.02	0.75	202.30	0.884	0.760	0.830	0.856	0.817	0.782
150	30	0.04	0.25	186.87	0.783	0.940	0.840	0.810	0.855	0.904
150	30	0.04	0.50	186.87	0.846	0.880	0.860	0.853	0.863	0.873
150	30	0.04	0.75	186.87	0.886	0.620	0.770	0.816	0.729	0.660
150	40	0.01	0.25	260.02	0.800	0.960	0.860	0.828	0.873	0.923
150	40	0.01	0.50	260.02	0.836	0.920	0.870	0.852	0.876	0.902
150	40	0.01	0.75	260.02	0.894	0.840	0.870	0.882	0.866	0.850
150	40	0.02	0.25	224.89	0.787	0.960	0.850	0.816	0.865	0.920
150	40	0.02	0.50	224.89	0.836	0.920	0.870	0.852	0.876	0.902
150	40	0.02	0.75	224.89	0.889	0.800	0.850	0.870	0.842	0.816
150	40	0.04	0.25	200.37	0.800	0.960	0.860	0.828	0.873	0.923
150	40	0.04	0.50	200.37	0.827	0.860	0.840	0.833	0.843	0.853
150	40	0.04	0.75	200.37	0.861	0.620	0.760	0.799	0.721	0.657
150	50	0.01	0.25	282.53	0.842	0.960	0.890	0.863	0.897	0.934
150	50	0.01	0.50	282.53	0.868	0.920	0.890	0.878	0.893	0.909
150	50	0.01	0.75	282.53	0.915	0.860	0.890	0.903	0.887	0.870
150	50	0.02	0.25	244.52	0.814	0.960	0.870	0.839	0.881	0.927
150	50	0.02	0.50	244.52	0.865	0.900	0.880	0.872	0.882	0.893
150	50	0.02	0.75	244.52	0.889	0.800	0.850	0.870	0.842	0.816
150	50	0.04	0.25	215.41	0.800	0.960	0.860	0.828	0.873	0.923
150	50	0.04	0.50	215.41	0.846	0.880	0.860	0.853	0.863	0.873
150	50	0.04	0.75	215.41	0.879	0.580	0.750	0.797	0.699	0.622
200	10	0.01	0.25	215.10	0.783	0.940	0.840	0.810	0.855	0.904
200	10	0.01	0.50	215.10	0.846	0.880	0.860	0.853	0.863	0.873
200	10	0.01	0.75	215.10	0.881	0.740	0.820	0.849	0.804	0.764
200	10	0.02	0.25	211.99	0.797	0.940	0.850	0.822	0.862	0.907
200	10	0.02	0.50	211.99	0.863	0.880	0.870	0.866	0.871	0.876
200	10	0.02	0.75	211.99	0.878	0.720	0.810	0.841	0.791	0.747
200	10	0.04	0.25	211.30	0.797	0.940	0.850	0.822	0.862	0.907
200	10	0.04	0.50	211.30	0.863	0.880	0.870	0.866	0.871	0.876
200	10	0.04	0.75	211.30	0.872	0.680	0.790	0.825	0.764	0.711
200	20	0.01	0.25	244.76	0.814	0.960	0.870	0.839	0.881	0.927
200	20	0.01	0.50	244.76	0.836	0.920	0.870	0.852	0.876	0.902
200	20	0.01	0.75	244.76	0.860	0.740	0.810	0.833	0.796	0.761
200	20	0.02	0.25	227.92	0.797	0.940	0.850	0.822	0.862	0.907

Continued on next page

Table C.1: Node - Dynamic Detection Mechanism Results continued

Start Msg	Win Size	STAB	DEC	Avg Msg	PR	RC	ACC	$F_{0.5}$	F_1	F_2
200	20	0.02	0.50	227.92	0.846	0.880	0.860	0.853	0.863	0.873
200	20	0.02	0.75	227.92	0.886	0.780	0.840	0.863	0.830	0.799
200	20	0.04	0.25	222.25	0.810	0.940	0.860	0.833	0.870	0.911
200	20	0.04	0.50	222.25	0.860	0.860	0.860	0.860	0.860	0.860
200	20	0.04	0.75	222.25	0.872	0.680	0.790	0.825	0.764	0.711
200	30	0.01	0.25	281.18	0.860	0.980	0.910	0.881	0.916	0.953
200	30	0.01	0.50	281.18	0.887	0.940	0.910	0.897	0.913	0.929
200	30	0.01	0.75	281.18	0.913	0.840	0.880	0.897	0.875	0.854
200	30	0.02	0.25	250.65	0.828	0.960	0.880	0.851	0.889	0.930
200	30	0.02	0.50	250.65	0.868	0.920	0.890	0.878	0.893	0.909
200	30	0.02	0.75	250.65	0.907	0.780	0.850	0.878	0.839	0.802
200	30	0.04	0.25	233.81	0.800	0.960	0.860	0.828	0.873	0.923
200	30	0.04	0.50	233.81	0.878	0.860	0.870	0.874	0.869	0.863
200	30	0.04	0.75	233.81	0.895	0.680	0.800	0.842	0.773	0.714
200	40	0.01	0.25	300.68	0.860	0.980	0.910	0.881	0.916	0.953
200	40	0.01	0.50	300.68	0.868	0.920	0.890	0.878	0.893	0.909
200	40	0.01	0.75	300.68	0.913	0.840	0.880	0.897	0.875	0.854
200	40	0.02	0.25	267.67	0.857	0.960	0.900	0.876	0.906	0.938
200	40	0.02	0.50	267.67	0.882	0.900	0.890	0.886	0.891	0.896
200	40	0.02	0.75	267.67	0.907	0.780	0.850	0.878	0.839	0.802
200	40	0.04	0.25	249.21	0.828	0.960	0.880	0.851	0.889	0.930
200	40	0.04	0.50	249.21	0.878	0.860	0.870	0.874	0.869	0.863
200	40	0.04	0.75	249.21	0.897	0.700	0.810	0.850	0.787	0.732
200	50	0.01	0.25	322.83	0.860	0.980	0.910	0.881	0.916	0.953
200	50	0.01	0.50	322.83	0.868	0.920	0.890	0.878	0.893	0.909
200	50	0.01	0.75	322.83	0.915	0.860	0.890	0.903	0.887	0.870
200	50	0.02	0.25	283.65	0.857	0.960	0.900	0.876	0.906	0.938
200	50	0.02	0.50	283.65	0.882	0.900	0.890	0.886	0.891	0.896
200	50	0.02	0.75	283.65	0.907	0.780	0.850	0.878	0.839	0.802
200	50	0.04	0.25	263.22	0.814	0.960	0.870	0.839	0.881	0.927
200	50	0.04	0.50	263.22	0.882	0.900	0.890	0.886	0.891	0.896
200	50	0.04	0.75	263.22	0.923	0.720	0.830	0.874	0.809	0.753
250	10	0.01	0.25	265.27	0.855	0.940	0.890	0.870	0.895	0.922
250	10	0.01	0.50	265.27	0.860	0.860	0.860	0.860	0.860	0.860
250	10	0.01	0.75	265.27	0.902	0.740	0.830	0.864	0.813	0.768
250	10	0.02	0.25	263.31	0.839	0.940	0.880	0.858	0.887	0.918
250	10	0.02	0.50	263.31	0.860	0.860	0.860	0.860	0.860	0.860
250	10	0.02	0.75	263.31	0.900	0.720	0.820	0.857	0.800	0.750
250	10	0.04	0.25	261.31	0.842	0.960	0.890	0.863	0.897	0.934
250	10	0.04	0.50	261.31	0.860	0.860	0.860	0.860	0.860	0.860

Continued on next page

Table C.1: Node - Dynamic Detection Mechanism Results continued

Start Msg	Win Size	STAB	DEC	Avg Msg	PR	RC	ACC	$F_{0.5}$	F_1	F_2
250	10	0.04	0.75	261.31	0.923	0.720	0.830	0.874	0.809	0.753
250	20	0.01	0.25	286.40	0.857	0.960	0.900	0.876	0.906	0.938
250	20	0.01	0.50	286.40	0.849	0.900	0.870	0.859	0.874	0.889
250	20	0.01	0.75	286.40	0.905	0.760	0.840	0.872	0.826	0.785
250	20	0.02	0.25	275.65	0.842	0.960	0.890	0.863	0.897	0.934
250	20	0.02	0.50	275.65	0.863	0.880	0.870	0.866	0.871	0.876
250	20	0.02	0.75	275.65	0.905	0.760	0.840	0.872	0.826	0.785
250	20	0.04	0.25	272.12	0.842	0.960	0.890	0.863	0.897	0.934
250	20	0.04	0.50	272.12	0.857	0.840	0.850	0.854	0.848	0.843
250	20	0.04	0.75	272.12	0.900	0.720	0.820	0.857	0.800	0.750
250	30	0.01	0.25	315.95	0.875	0.980	0.920	0.894	0.925	0.957
250	30	0.01	0.50	315.95	0.870	0.940	0.900	0.883	0.904	0.925
250	30	0.01	0.75	315.95	0.915	0.860	0.890	0.903	0.887	0.870
250	30	0.02	0.25	290.81	0.842	0.960	0.890	0.863	0.897	0.934
250	30	0.02	0.50	290.81	0.852	0.920	0.880	0.865	0.885	0.906
250	30	0.02	0.75	290.81	0.909	0.800	0.860	0.885	0.851	0.820
250	30	0.04	0.25	283.09	0.845	0.980	0.900	0.869	0.907	0.950
250	30	0.04	0.50	283.09	0.860	0.860	0.860	0.860	0.860	0.860
250	30	0.04	0.75	283.09	0.900	0.720	0.820	0.857	0.800	0.750
250	40	0.01	0.25	338.39	0.891	0.980	0.930	0.907	0.933	0.961
250	40	0.01	0.50	338.39	0.887	0.940	0.910	0.897	0.913	0.929
250	40	0.01	0.75	338.39	0.915	0.860	0.890	0.903	0.887	0.870
250	40	0.02	0.25	306.38	0.875	0.980	0.920	0.894	0.925	0.957
250	40	0.02	0.50	306.38	0.882	0.900	0.890	0.886	0.891	0.896
250	40	0.02	0.75	306.38	0.911	0.820	0.870	0.891	0.863	0.837
250	40	0.04	0.25	293.81	0.845	0.980	0.900	0.869	0.907	0.950
250	40	0.04	0.50	293.81	0.865	0.900	0.880	0.872	0.882	0.893
250	40	0.04	0.75	293.81	0.900	0.720	0.820	0.857	0.800	0.750
250	50	0.01	0.25	369.52	0.891	0.980	0.930	0.907	0.933	0.961
250	50	0.01	0.50	369.52	0.887	0.940	0.910	0.897	0.913	0.929
250	50	0.01	0.75	369.52	0.915	0.860	0.890	0.903	0.887	0.870
250	50	0.02	0.25	322.61	0.891	0.980	0.930	0.907	0.933	0.961
250	50	0.02	0.50	322.61	0.900	0.900	0.900	0.900	0.900	0.900
250	50	0.02	0.75	322.61	0.911	0.820	0.870	0.891	0.863	0.837
250	50	0.04	0.25	306.43	0.842	0.960	0.890	0.863	0.897	0.934
250	50	0.04	0.50	306.43	0.865	0.900	0.880	0.872	0.882	0.893
250	50	0.04	0.75	306.43	0.925	0.740	0.840	0.881	0.822	0.771

C.2 Node and Context

Table C.2: Node and Context - Dynamic Detection Mechanism Results

Start Msg	Win Size	STAB	DEC	Avg Msg	PR	RC	ACC	$F_{0.5}$	F_1	F_2
0	10	0.01	0.25	32.38	0.661	0.740	0.680	0.675	0.698	0.723
0	10	0.01	0.50	32.38	0.661	0.740	0.680	0.675	0.698	0.723
0	10	0.01	0.75	32.38	0.667	0.720	0.680	0.677	0.692	0.709
0	10	0.02	0.25	24.10	0.610	0.720	0.630	0.629	0.661	0.695
0	10	0.02	0.50	24.10	0.603	0.700	0.620	0.621	0.648	0.678
0	10	0.02	0.75	24.10	0.625	0.700	0.640	0.639	0.660	0.684
0	10	0.04	0.25	17.39	0.547	0.700	0.560	0.572	0.614	0.663
0	10	0.04	0.50	17.39	0.548	0.680	0.560	0.570	0.607	0.649
0	10	0.04	0.75	17.39	0.586	0.680	0.600	0.603	0.630	0.659
0	20	0.01	0.25	70.32	0.792	0.840	0.810	0.802	0.816	0.830
0	20	0.01	0.50	70.32	0.808	0.840	0.820	0.814	0.824	0.833
0	20	0.01	0.75	70.32	0.808	0.840	0.820	0.814	0.824	0.833
0	20	0.02	0.25	46.34	0.702	0.800	0.730	0.719	0.748	0.778
0	20	0.02	0.50	46.34	0.727	0.800	0.750	0.741	0.762	0.784
0	20	0.02	0.75	46.34	0.727	0.800	0.750	0.741	0.762	0.784
0	20	0.04	0.25	35.09	0.619	0.780	0.650	0.646	0.690	0.741
0	20	0.04	0.50	35.09	0.633	0.760	0.660	0.655	0.691	0.731
0	20	0.04	0.75	35.09	0.660	0.700	0.670	0.668	0.680	0.692
0	30	0.01	0.25	125.58	0.846	0.880	0.860	0.853	0.863	0.873
0	30	0.01	0.50	125.58	0.863	0.880	0.870	0.866	0.871	0.876
0	30	0.01	0.75	125.58	0.863	0.880	0.870	0.866	0.871	0.876
0	30	0.02	0.25	74.16	0.754	0.860	0.790	0.773	0.804	0.837
0	30	0.02	0.50	74.16	0.768	0.860	0.800	0.785	0.811	0.840
0	30	0.02	0.75	74.16	0.764	0.840	0.790	0.778	0.800	0.824
0	30	0.04	0.25	55.08	0.672	0.860	0.720	0.703	0.754	0.814
0	30	0.04	0.50	55.08	0.717	0.860	0.760	0.741	0.782	0.827
0	30	0.04	0.75	55.08	0.719	0.820	0.750	0.737	0.766	0.798
0	40	0.01	0.25	160.64	0.885	0.920	0.900	0.891	0.902	0.913
0	40	0.01	0.50	160.64	0.902	0.920	0.910	0.906	0.911	0.916
0	40	0.01	0.75	160.64	0.920	0.920	0.920	0.920	0.920	0.920
0	40	0.02	0.25	107.72	0.807	0.920	0.850	0.827	0.860	0.895
0	40	0.02	0.50	107.72	0.836	0.920	0.870	0.852	0.876	0.902
0	40	0.02	0.75	107.72	0.836	0.920	0.870	0.852	0.876	0.902

Continued on next page

Table C.2: Node and Context - Dynamic Detection Mechanism Results continued

Start Msg	Win Size	STAB	DEC	Avg Msg	PR	RC	ACC	$F_{0.5}$	F_1	F_2
0	40	0.04	0.25	75.05	0.697	0.920	0.760	0.732	0.793	0.865
0	40	0.04	0.50	75.05	0.738	0.900	0.790	0.765	0.811	0.862
0	40	0.04	0.75	75.05	0.741	0.860	0.780	0.762	0.796	0.833
0	50	0.01	0.25	184.48	0.885	0.920	0.900	0.891	0.902	0.913
0	50	0.01	0.50	184.48	0.920	0.920	0.920	0.920	0.920	0.920
0	50	0.01	0.75	184.48	0.918	0.900	0.910	0.915	0.909	0.904
0	50	0.02	0.25	124.90	0.852	0.920	0.880	0.865	0.885	0.906
0	50	0.02	0.50	124.90	0.885	0.920	0.900	0.891	0.902	0.913
0	50	0.02	0.75	124.90	0.885	0.920	0.900	0.891	0.902	0.913
0	50	0.04	0.25	90.13	0.734	0.940	0.800	0.768	0.825	0.890
0	50	0.04	0.50	90.13	0.780	0.920	0.830	0.804	0.844	0.888
0	50	0.04	0.75	90.13	0.782	0.860	0.810	0.796	0.819	0.843
50	10	0.01	0.25	77.86	0.758	0.940	0.820	0.789	0.839	0.897
50	10	0.01	0.50	77.86	0.797	0.940	0.850	0.822	0.862	0.907
50	10	0.01	0.75	77.86	0.807	0.920	0.850	0.827	0.860	0.895
50	10	0.02	0.25	67.11	0.742	0.980	0.820	0.780	0.845	0.921
50	10	0.02	0.50	67.11	0.774	0.960	0.840	0.805	0.857	0.916
50	10	0.02	0.75	67.11	0.782	0.860	0.810	0.796	0.819	0.843
50	10	0.04	0.25	63.37	0.710	0.980	0.790	0.752	0.824	0.911
50	10	0.04	0.50	63.37	0.750	0.960	0.820	0.784	0.842	0.909
50	10	0.04	0.75	63.37	0.772	0.880	0.810	0.791	0.822	0.856
50	20	0.01	0.25	117.00	0.887	0.940	0.910	0.897	0.913	0.929
50	20	0.01	0.50	117.00	0.904	0.940	0.920	0.911	0.922	0.933
50	20	0.01	0.75	117.00	0.922	0.940	0.930	0.925	0.931	0.936
50	20	0.02	0.25	87.76	0.783	0.940	0.840	0.810	0.855	0.904
50	20	0.02	0.50	87.76	0.810	0.940	0.860	0.833	0.870	0.911
50	20	0.02	0.75	87.76	0.821	0.920	0.860	0.839	0.868	0.898
50	20	0.04	0.25	77.80	0.738	0.960	0.810	0.774	0.835	0.906
50	20	0.04	0.50	77.80	0.770	0.940	0.830	0.799	0.847	0.900
50	20	0.04	0.75	77.80	0.774	0.820	0.790	0.782	0.796	0.810
50	30	0.01	0.25	161.96	0.887	0.940	0.910	0.897	0.913	0.929
50	30	0.01	0.50	161.96	0.922	0.940	0.930	0.925	0.931	0.936
50	30	0.01	0.75	161.96	0.940	0.940	0.940	0.940	0.940	0.940
50	30	0.02	0.25	113.61	0.870	0.940	0.900	0.883	0.904	0.925
50	30	0.02	0.50	113.61	0.887	0.940	0.910	0.897	0.913	0.929
50	30	0.02	0.75	113.61	0.882	0.900	0.890	0.886	0.891	0.896
50	30	0.04	0.25	93.76	0.746	0.940	0.810	0.778	0.832	0.894
50	30	0.04	0.50	93.76	0.825	0.940	0.870	0.845	0.879	0.914
50	30	0.04	0.75	93.76	0.827	0.860	0.840	0.833	0.843	0.853
50	40	0.01	0.25	205.96	0.889	0.960	0.920	0.902	0.923	0.945

Continued on next page

Table C.2: Node and Context - Dynamic Detection Mechanism Results continued

Start Msg	Win Size	STAB	DEC	Avg Msg	PR	RC	ACC	$F_{0.5}$	F_1	F_2
50	40	0.01	0.50	205.96	0.923	0.960	0.940	0.930	0.941	0.952
50	40	0.01	0.75	205.96	0.940	0.940	0.940	0.940	0.940	0.940
50	40	0.02	0.25	143.80	0.873	0.960	0.910	0.889	0.914	0.941
50	40	0.02	0.50	143.80	0.923	0.960	0.940	0.930	0.941	0.952
50	40	0.02	0.75	143.80	0.923	0.960	0.940	0.930	0.941	0.952
50	40	0.04	0.25	110.20	0.774	0.960	0.840	0.805	0.857	0.916
50	40	0.04	0.50	110.20	0.839	0.940	0.880	0.858	0.887	0.918
50	40	0.04	0.75	110.20	0.846	0.880	0.860	0.853	0.863	0.873
50	50	0.01	0.25	241.66	0.906	0.960	0.930	0.916	0.932	0.949
50	50	0.01	0.50	241.66	0.923	0.960	0.940	0.930	0.941	0.952
50	50	0.01	0.75	241.66	0.920	0.920	0.920	0.920	0.920	0.920
50	50	0.02	0.25	160.44	0.889	0.960	0.920	0.902	0.923	0.945
50	50	0.02	0.50	160.44	0.941	0.960	0.950	0.945	0.950	0.956
50	50	0.02	0.75	160.44	0.941	0.960	0.950	0.945	0.950	0.956
50	50	0.04	0.25	127.53	0.831	0.980	0.890	0.857	0.899	0.946
50	50	0.04	0.50	127.53	0.873	0.960	0.910	0.889	0.914	0.941
50	50	0.04	0.75	127.53	0.880	0.880	0.880	0.880	0.880	0.880
100	10	0.01	0.25	119.93	0.783	0.940	0.840	0.810	0.855	0.904
100	10	0.01	0.50	119.93	0.825	0.940	0.870	0.845	0.879	0.914
100	10	0.01	0.75	119.93	0.836	0.920	0.870	0.852	0.876	0.902
100	10	0.02	0.25	112.35	0.787	0.960	0.850	0.816	0.865	0.920
100	10	0.02	0.50	112.35	0.800	0.960	0.860	0.828	0.873	0.923
100	10	0.02	0.75	112.35	0.865	0.900	0.880	0.872	0.882	0.893
100	10	0.04	0.25	111.29	0.774	0.960	0.840	0.805	0.857	0.916
100	10	0.04	0.50	111.29	0.800	0.960	0.860	0.828	0.873	0.923
100	10	0.04	0.75	111.29	0.800	0.880	0.830	0.815	0.838	0.863
100	20	0.01	0.25	160.42	0.889	0.960	0.920	0.902	0.923	0.945
100	20	0.01	0.50	160.42	0.923	0.960	0.940	0.930	0.941	0.952
100	20	0.01	0.75	160.42	0.941	0.960	0.950	0.945	0.950	0.956
100	20	0.02	0.25	134.55	0.810	0.940	0.860	0.833	0.870	0.911
100	20	0.02	0.50	134.55	0.839	0.940	0.880	0.858	0.887	0.918
100	20	0.02	0.75	134.55	0.885	0.920	0.900	0.891	0.902	0.913
100	20	0.04	0.25	123.89	0.787	0.960	0.850	0.816	0.865	0.920
100	20	0.04	0.50	123.89	0.828	0.960	0.880	0.851	0.889	0.930
100	20	0.04	0.75	123.89	0.830	0.880	0.850	0.840	0.854	0.870
100	30	0.01	0.25	200.66	0.889	0.960	0.920	0.902	0.923	0.945
100	30	0.01	0.50	200.66	0.941	0.960	0.950	0.945	0.950	0.956
100	30	0.01	0.75	200.66	0.940	0.940	0.940	0.940	0.940	0.940
100	30	0.02	0.25	160.03	0.889	0.960	0.920	0.902	0.923	0.945
100	30	0.02	0.50	160.03	0.923	0.960	0.940	0.930	0.941	0.952

Continued on next page

Table C.2: Node and Context - Dynamic Detection Mechanism Results continued

Start Msg	Win Size	STAB	DEC	Avg Msg	PR	RC	ACC	$F_{0.5}$	F_1	F_2
100	30	0.02	0.75	160.03	0.939	0.920	0.930	0.935	0.929	0.924
100	30	0.04	0.25	138.41	0.797	0.940	0.850	0.822	0.862	0.907
100	30	0.04	0.50	138.41	0.855	0.940	0.890	0.870	0.895	0.922
100	30	0.04	0.75	138.41	0.880	0.880	0.880	0.880	0.880	0.880
100	40	0.01	0.25	249.84	0.889	0.960	0.920	0.902	0.923	0.945
100	40	0.01	0.50	249.84	0.941	0.960	0.950	0.945	0.950	0.956
100	40	0.01	0.75	249.84	0.939	0.920	0.930	0.935	0.929	0.924
100	40	0.02	0.25	182.38	0.889	0.960	0.920	0.902	0.923	0.945
100	40	0.02	0.50	182.38	0.941	0.960	0.950	0.945	0.950	0.956
100	40	0.02	0.75	182.38	0.941	0.960	0.950	0.945	0.950	0.956
100	40	0.04	0.25	157.07	0.810	0.940	0.860	0.833	0.870	0.911
100	40	0.04	0.50	157.07	0.852	0.920	0.880	0.865	0.885	0.906
100	40	0.04	0.75	157.07	0.875	0.840	0.860	0.868	0.857	0.847
100	50	0.01	0.25	288.05	0.940	0.940	0.940	0.940	0.940	0.940
100	50	0.01	0.50	288.05	0.959	0.940	0.950	0.955	0.949	0.944
100	50	0.01	0.75	288.05	0.957	0.880	0.920	0.940	0.917	0.894
100	50	0.02	0.25	201.32	0.889	0.960	0.920	0.902	0.923	0.945
100	50	0.02	0.50	201.32	0.941	0.960	0.950	0.945	0.950	0.956
100	50	0.02	0.75	201.32	0.940	0.940	0.940	0.940	0.940	0.940
100	50	0.04	0.25	179.01	0.828	0.960	0.880	0.851	0.889	0.930
100	50	0.04	0.50	179.01	0.887	0.940	0.910	0.897	0.913	0.929
100	50	0.04	0.75	179.01	0.913	0.840	0.880	0.897	0.875	0.854
150	10	0.01	0.25	170.25	0.855	0.940	0.890	0.870	0.895	0.922
150	10	0.01	0.50	170.25	0.922	0.940	0.930	0.925	0.931	0.936
150	10	0.01	0.75	170.25	0.938	0.900	0.920	0.930	0.918	0.907
150	10	0.02	0.25	163.43	0.783	0.940	0.840	0.810	0.855	0.904
150	10	0.02	0.50	163.43	0.870	0.940	0.900	0.883	0.904	0.925
150	10	0.02	0.75	163.43	0.917	0.880	0.900	0.909	0.898	0.887
150	10	0.04	0.25	161.38	0.774	0.960	0.840	0.805	0.857	0.916
150	10	0.04	0.50	161.38	0.855	0.940	0.890	0.870	0.895	0.922
150	10	0.04	0.75	161.38	0.882	0.900	0.890	0.886	0.891	0.896
150	20	0.01	0.25	203.90	0.889	0.960	0.920	0.902	0.923	0.945
150	20	0.01	0.50	203.90	0.922	0.940	0.930	0.925	0.931	0.936
150	20	0.01	0.75	203.90	0.939	0.920	0.930	0.935	0.929	0.924
150	20	0.02	0.25	182.64	0.855	0.940	0.890	0.870	0.895	0.922
150	20	0.02	0.50	182.64	0.887	0.940	0.910	0.897	0.913	0.929
150	20	0.02	0.75	182.64	0.938	0.900	0.920	0.930	0.918	0.907
150	20	0.04	0.25	173.12	0.787	0.960	0.850	0.816	0.865	0.920
150	20	0.04	0.50	173.12	0.889	0.960	0.920	0.902	0.923	0.945
150	20	0.04	0.75	173.12	0.913	0.840	0.880	0.897	0.875	0.854

Continued on next page

Table C.2: Node and Context - Dynamic Detection Mechanism Results continued

Start Msg	Win Size	STAB	DEC	Avg Msg	PR	RC	ACC	$F_{0.5}$	F_1	F_2
150	30	0.01	0.25	243.76	0.906	0.960	0.930	0.916	0.932	0.949
150	30	0.01	0.50	243.76	0.941	0.960	0.950	0.945	0.950	0.956
150	30	0.01	0.75	243.76	0.940	0.940	0.940	0.940	0.940	0.940
150	30	0.02	0.25	208.04	0.906	0.960	0.930	0.916	0.932	0.949
150	30	0.02	0.50	208.04	0.923	0.960	0.940	0.930	0.941	0.952
150	30	0.02	0.75	208.04	0.938	0.900	0.920	0.930	0.918	0.907
150	30	0.04	0.25	186.32	0.810	0.940	0.860	0.833	0.870	0.911
150	30	0.04	0.50	186.32	0.904	0.940	0.920	0.911	0.922	0.933
150	30	0.04	0.75	186.32	0.933	0.840	0.890	0.913	0.884	0.857
150	40	0.01	0.25	290.54	0.925	0.980	0.950	0.935	0.951	0.968
150	40	0.01	0.50	290.54	0.961	0.980	0.970	0.965	0.970	0.976
150	40	0.01	0.75	290.54	0.960	0.960	0.960	0.960	0.960	0.960
150	40	0.02	0.25	225.06	0.889	0.960	0.920	0.902	0.923	0.945
150	40	0.02	0.50	225.06	0.941	0.960	0.950	0.945	0.950	0.956
150	40	0.02	0.75	225.06	0.939	0.920	0.930	0.935	0.929	0.924
150	40	0.04	0.25	203.68	0.839	0.940	0.880	0.858	0.887	0.918
150	40	0.04	0.50	203.68	0.902	0.920	0.910	0.906	0.911	0.916
150	40	0.04	0.75	203.68	0.955	0.840	0.900	0.929	0.894	0.861
150	50	0.01	0.25	333.49	0.941	0.960	0.950	0.945	0.950	0.956
150	50	0.01	0.50	333.49	0.960	0.960	0.960	0.960	0.960	0.960
150	50	0.01	0.75	333.49	0.958	0.920	0.940	0.950	0.939	0.927
150	50	0.02	0.25	243.96	0.889	0.960	0.920	0.902	0.923	0.945
150	50	0.02	0.50	243.96	0.941	0.960	0.950	0.945	0.950	0.956
150	50	0.02	0.75	243.96	0.940	0.940	0.940	0.940	0.940	0.940
150	50	0.04	0.25	221.56	0.857	0.960	0.900	0.876	0.906	0.938
150	50	0.04	0.50	221.56	0.904	0.940	0.920	0.911	0.922	0.933
150	50	0.04	0.75	221.56	0.932	0.820	0.880	0.907	0.872	0.840
200	10	0.01	0.25	217.68	0.870	0.940	0.900	0.883	0.904	0.925
200	10	0.01	0.50	217.68	0.887	0.940	0.910	0.897	0.913	0.929
200	10	0.01	0.75	217.68	0.935	0.860	0.900	0.919	0.896	0.874
200	10	0.02	0.25	212.81	0.855	0.940	0.890	0.870	0.895	0.922
200	10	0.02	0.50	212.81	0.885	0.920	0.900	0.891	0.902	0.913
200	10	0.02	0.75	212.81	0.956	0.860	0.910	0.935	0.905	0.878
200	10	0.04	0.25	211.48	0.839	0.940	0.880	0.858	0.887	0.918
200	10	0.04	0.50	211.48	0.904	0.940	0.920	0.911	0.922	0.933
200	10	0.04	0.75	211.48	0.957	0.880	0.920	0.940	0.917	0.894
200	20	0.01	0.25	246.10	0.906	0.960	0.930	0.916	0.932	0.949
200	20	0.01	0.50	246.10	0.940	0.940	0.940	0.940	0.940	0.940
200	20	0.01	0.75	246.10	0.957	0.900	0.930	0.945	0.928	0.911
200	20	0.02	0.25	227.54	0.870	0.940	0.900	0.883	0.904	0.925

Continued on next page

Table C.2: Node and Context - Dynamic Detection Mechanism Results continued

Start Msg	Win Size	STAB	DEC	Avg Msg	PR	RC	ACC	$F_{0.5}$	F_1	F_2
200	20	0.02	0.50	227.54	0.885	0.920	0.900	0.891	0.902	0.913
200	20	0.02	0.75	227.54	0.935	0.860	0.900	0.919	0.896	0.874
200	20	0.04	0.25	223.27	0.836	0.920	0.870	0.852	0.876	0.902
200	20	0.04	0.50	223.27	0.885	0.920	0.900	0.891	0.902	0.913
200	20	0.04	0.75	223.27	0.955	0.840	0.900	0.929	0.894	0.861
200	30	0.01	0.25	280.55	0.923	0.960	0.940	0.930	0.941	0.952
200	30	0.01	0.50	280.55	0.960	0.960	0.960	0.960	0.960	0.960
200	30	0.01	0.75	280.55	0.959	0.940	0.950	0.955	0.949	0.944
200	30	0.02	0.25	248.47	0.889	0.960	0.920	0.902	0.923	0.945
200	30	0.02	0.50	248.47	0.923	0.960	0.940	0.930	0.941	0.952
200	30	0.02	0.75	248.47	0.938	0.900	0.920	0.930	0.918	0.907
200	30	0.04	0.25	235.17	0.855	0.940	0.890	0.870	0.895	0.922
200	30	0.04	0.50	235.17	0.885	0.920	0.900	0.891	0.902	0.913
200	30	0.04	0.75	235.17	0.955	0.840	0.900	0.929	0.894	0.861
200	40	0.01	0.25	326.02	0.923	0.960	0.940	0.930	0.941	0.952
200	40	0.01	0.50	326.02	0.960	0.960	0.960	0.960	0.960	0.960
200	40	0.01	0.75	326.02	0.959	0.940	0.950	0.955	0.949	0.944
200	40	0.02	0.25	265.85	0.889	0.960	0.920	0.902	0.923	0.945
200	40	0.02	0.50	265.85	0.941	0.960	0.950	0.945	0.950	0.956
200	40	0.02	0.75	265.85	0.938	0.900	0.920	0.930	0.918	0.907
200	40	0.04	0.25	247.33	0.870	0.940	0.900	0.883	0.904	0.925
200	40	0.04	0.50	247.33	0.920	0.920	0.920	0.920	0.920	0.920
200	40	0.04	0.75	247.33	0.956	0.860	0.910	0.935	0.905	0.878
200	50	0.01	0.25	368.52	0.942	0.980	0.960	0.950	0.961	0.972
200	50	0.01	0.50	368.52	0.960	0.960	0.960	0.960	0.960	0.960
200	50	0.01	0.75	368.52	0.958	0.920	0.940	0.950	0.939	0.927
200	50	0.02	0.25	282.10	0.889	0.960	0.920	0.902	0.923	0.945
200	50	0.02	0.50	282.10	0.941	0.960	0.950	0.945	0.950	0.956
200	50	0.02	0.75	282.10	0.939	0.920	0.930	0.935	0.929	0.924
200	50	0.04	0.25	265.28	0.873	0.960	0.910	0.889	0.914	0.941
200	50	0.04	0.50	265.28	0.922	0.940	0.930	0.925	0.931	0.936
200	50	0.04	0.75	265.28	0.935	0.860	0.900	0.919	0.896	0.874
250	10	0.01	0.25	265.13	0.842	0.960	0.890	0.863	0.897	0.934
250	10	0.01	0.50	265.13	0.904	0.940	0.920	0.911	0.922	0.933
250	10	0.01	0.75	265.13	0.932	0.820	0.880	0.907	0.872	0.840
250	10	0.02	0.25	262.83	0.842	0.960	0.890	0.863	0.897	0.934
250	10	0.02	0.50	262.83	0.902	0.920	0.910	0.906	0.911	0.916
250	10	0.02	0.75	262.83	0.933	0.840	0.890	0.913	0.884	0.857
250	10	0.04	0.25	261.29	0.842	0.960	0.890	0.863	0.897	0.934
250	10	0.04	0.50	261.29	0.902	0.920	0.910	0.906	0.911	0.916

Continued on next page

Table C.2: Node and Context - Dynamic Detection Mechanism Results continued

Start Msg	Win Size	STAB	DEC	Avg Msg	PR	RC	ACC	$F_{0.5}$	F_1	F_2
250	10	0.04	0.75	261.29	0.953	0.820	0.890	0.923	0.882	0.844
250	20	0.01	0.25	286.85	0.906	0.960	0.930	0.916	0.932	0.949
250	20	0.01	0.50	286.85	0.940	0.940	0.940	0.940	0.940	0.940
250	20	0.01	0.75	286.85	0.957	0.900	0.930	0.945	0.928	0.911
250	20	0.02	0.25	275.97	0.857	0.960	0.900	0.876	0.906	0.938
250	20	0.02	0.50	275.97	0.904	0.940	0.920	0.911	0.922	0.933
250	20	0.02	0.75	275.97	0.935	0.860	0.900	0.919	0.896	0.874
250	20	0.04	0.25	272.71	0.873	0.960	0.910	0.889	0.914	0.941
250	20	0.04	0.50	272.71	0.902	0.920	0.910	0.906	0.911	0.916
250	20	0.04	0.75	272.71	0.930	0.800	0.870	0.901	0.860	0.823
250	30	0.01	0.25	315.52	0.923	0.960	0.940	0.930	0.941	0.952
250	30	0.01	0.50	315.52	0.960	0.960	0.960	0.960	0.960	0.960
250	30	0.01	0.75	315.52	0.959	0.940	0.950	0.955	0.949	0.944
250	30	0.02	0.25	293.36	0.889	0.960	0.920	0.902	0.923	0.945
250	30	0.02	0.50	293.36	0.923	0.960	0.940	0.930	0.941	0.952
250	30	0.02	0.75	293.36	0.957	0.900	0.930	0.945	0.928	0.911
250	30	0.04	0.25	284.41	0.873	0.960	0.910	0.889	0.914	0.941
250	30	0.04	0.50	284.41	0.904	0.940	0.920	0.911	0.922	0.933
250	30	0.04	0.75	284.41	0.933	0.840	0.890	0.913	0.884	0.857
250	40	0.01	0.25	361.27	0.906	0.960	0.930	0.916	0.932	0.949
250	40	0.01	0.50	361.27	0.941	0.960	0.950	0.945	0.950	0.956
250	40	0.01	0.75	361.27	0.959	0.940	0.950	0.955	0.949	0.944
250	40	0.02	0.25	311.89	0.906	0.960	0.930	0.916	0.932	0.949
250	40	0.02	0.50	311.89	0.941	0.960	0.950	0.945	0.950	0.956
250	40	0.02	0.75	311.89	0.957	0.900	0.930	0.945	0.928	0.911
250	40	0.04	0.25	296.57	0.889	0.960	0.920	0.902	0.923	0.945
250	40	0.04	0.50	296.57	0.922	0.940	0.930	0.925	0.931	0.936
250	40	0.04	0.75	296.57	0.936	0.880	0.910	0.924	0.907	0.891
250	50	0.01	0.25	402.15	0.925	0.980	0.950	0.935	0.951	0.968
250	50	0.01	0.50	402.15	0.942	0.980	0.960	0.950	0.961	0.972
250	50	0.01	0.75	402.15	0.959	0.940	0.950	0.955	0.949	0.944
250	50	0.02	0.25	324.24	0.906	0.960	0.930	0.916	0.932	0.949
250	50	0.02	0.50	324.24	0.960	0.960	0.960	0.960	0.960	0.960
250	50	0.02	0.75	324.24	0.958	0.920	0.940	0.950	0.939	0.927
250	50	0.04	0.25	308.28	0.873	0.960	0.910	0.889	0.914	0.941
250	50	0.04	0.50	308.28	0.922	0.940	0.930	0.925	0.931	0.936
250	50	0.04	0.75	308.28	0.936	0.880	0.910	0.924	0.907	0.891

C.3 Node, Context, and Content

Table C.3: Node, Context, and Content - Dynamic Detection Mechanism Results

Start Msg	Win Size	STAB	DEC	Avg Msg	PR	RC	ACC	$F_{0.5}$	F_1	F_2
0	10	0.01	0.25	35.72	0.667	0.680	0.670	0.669	0.673	0.677
0	10	0.01	0.50	35.72	0.673	0.660	0.670	0.671	0.667	0.663
0	10	0.01	0.75	35.72	0.667	0.640	0.660	0.661	0.653	0.645
0	10	0.02	0.25	23.98	0.636	0.700	0.650	0.648	0.667	0.686
0	10	0.02	0.50	23.98	0.648	0.700	0.660	0.658	0.673	0.689
0	10	0.02	0.75	23.98	0.667	0.680	0.670	0.669	0.673	0.677
0	10	0.04	0.25	20.10	0.617	0.740	0.640	0.638	0.673	0.712
0	10	0.04	0.50	20.10	0.627	0.740	0.650	0.647	0.679	0.714
0	10	0.04	0.75	20.10	0.623	0.660	0.630	0.630	0.641	0.652
0	20	0.01	0.25	75.67	0.833	0.800	0.820	0.826	0.816	0.806
0	20	0.01	0.50	75.67	0.830	0.780	0.810	0.819	0.804	0.789
0	20	0.01	0.75	75.67	0.826	0.760	0.800	0.812	0.792	0.772
0	20	0.02	0.25	46.33	0.731	0.760	0.740	0.736	0.745	0.754
0	20	0.02	0.50	46.33	0.740	0.740	0.740	0.740	0.740	0.740
0	20	0.02	0.75	46.33	0.735	0.720	0.730	0.732	0.727	0.723
0	20	0.04	0.25	35.46	0.691	0.760	0.710	0.704	0.724	0.745
0	20	0.04	0.50	35.46	0.691	0.760	0.710	0.704	0.724	0.745
0	20	0.04	0.75	35.46	0.712	0.740	0.720	0.717	0.725	0.734
0	30	0.01	0.25	139.24	0.846	0.880	0.860	0.853	0.863	0.873
0	30	0.01	0.50	139.24	0.863	0.880	0.870	0.866	0.871	0.876
0	30	0.01	0.75	139.24	0.860	0.860	0.860	0.860	0.860	0.860
0	30	0.02	0.25	84.52	0.796	0.860	0.820	0.808	0.827	0.846
0	30	0.02	0.50	84.52	0.808	0.840	0.820	0.814	0.824	0.833
0	30	0.02	0.75	84.52	0.792	0.760	0.780	0.785	0.776	0.766
0	30	0.04	0.25	57.51	0.737	0.840	0.770	0.755	0.785	0.817
0	30	0.04	0.50	57.51	0.745	0.820	0.770	0.759	0.781	0.804
0	30	0.04	0.75	57.51	0.750	0.780	0.760	0.756	0.765	0.774
0	40	0.01	0.25	183.90	0.855	0.940	0.890	0.870	0.895	0.922
0	40	0.01	0.50	183.90	0.870	0.940	0.900	0.883	0.904	0.925
0	40	0.01	0.75	183.90	0.868	0.920	0.890	0.878	0.893	0.909
0	40	0.02	0.25	123.56	0.852	0.920	0.880	0.865	0.885	0.906
0	40	0.02	0.50	123.56	0.868	0.920	0.890	0.878	0.893	0.909
0	40	0.02	0.75	123.56	0.860	0.860	0.860	0.860	0.860	0.860

Continued on next page

Table C.3: Node, Context, and Content - Dynamic Detection Mechanism Results continued

Start Msg	Win Size	STAB	DEC	Avg Msg	PR	RC	ACC	$F_{0.5}$	F_1	F_2
0	40	0.04	0.25	75.71	0.746	0.940	0.810	0.778	0.832	0.894
0	40	0.04	0.50	75.71	0.767	0.920	0.820	0.793	0.836	0.885
0	40	0.04	0.75	75.71	0.778	0.840	0.800	0.789	0.808	0.827
0	50	0.01	0.25	205.11	0.873	0.960	0.910	0.889	0.914	0.941
0	50	0.01	0.50	205.11	0.889	0.960	0.920	0.902	0.923	0.945
0	50	0.01	0.75	205.11	0.887	0.940	0.910	0.897	0.913	0.929
0	50	0.02	0.25	142.77	0.852	0.920	0.880	0.865	0.885	0.906
0	50	0.02	0.50	142.77	0.868	0.920	0.890	0.878	0.893	0.909
0	50	0.02	0.75	142.77	0.880	0.880	0.880	0.880	0.880	0.880
0	50	0.04	0.25	104.28	0.780	0.920	0.830	0.804	0.844	0.888
0	50	0.04	0.50	104.28	0.849	0.900	0.870	0.859	0.874	0.889
0	50	0.04	0.75	104.28	0.840	0.840	0.840	0.840	0.840	0.840
50	10	0.01	0.25	81.16	0.793	0.920	0.840	0.816	0.852	0.891
50	10	0.01	0.50	81.16	0.804	0.900	0.840	0.821	0.849	0.879
50	10	0.01	0.75	81.16	0.796	0.860	0.820	0.808	0.827	0.846
50	10	0.02	0.25	66.60	0.712	0.940	0.780	0.748	0.810	0.883
50	10	0.02	0.50	66.60	0.797	0.940	0.850	0.822	0.862	0.907
50	10	0.02	0.75	66.60	0.796	0.860	0.820	0.808	0.827	0.846
50	10	0.04	0.25	63.99	0.716	0.960	0.790	0.755	0.821	0.899
50	10	0.04	0.50	63.99	0.783	0.940	0.840	0.810	0.855	0.904
50	10	0.04	0.75	63.99	0.788	0.820	0.800	0.795	0.804	0.813
50	20	0.01	0.25	127.61	0.868	0.920	0.890	0.878	0.893	0.909
50	20	0.01	0.50	127.61	0.882	0.900	0.890	0.886	0.891	0.896
50	20	0.01	0.75	127.61	0.898	0.880	0.890	0.894	0.889	0.884
50	20	0.02	0.25	88.17	0.770	0.940	0.830	0.799	0.847	0.900
50	20	0.02	0.50	88.17	0.793	0.920	0.840	0.816	0.852	0.891
50	20	0.02	0.75	88.17	0.792	0.840	0.810	0.802	0.816	0.830
50	20	0.04	0.25	78.94	0.746	0.940	0.810	0.778	0.832	0.894
50	20	0.04	0.50	78.94	0.783	0.940	0.840	0.810	0.855	0.904
50	20	0.04	0.75	78.94	0.774	0.820	0.790	0.782	0.796	0.810
50	30	0.01	0.25	174.31	0.885	0.920	0.900	0.891	0.902	0.913
50	30	0.01	0.50	174.31	0.920	0.920	0.920	0.920	0.920	0.920
50	30	0.01	0.75	174.31	0.918	0.900	0.910	0.915	0.909	0.904
50	30	0.02	0.25	120.97	0.852	0.920	0.880	0.865	0.885	0.906
50	30	0.02	0.50	120.97	0.865	0.900	0.880	0.872	0.882	0.893
50	30	0.02	0.75	120.97	0.854	0.820	0.840	0.847	0.837	0.827
50	30	0.04	0.25	95.13	0.787	0.960	0.850	0.816	0.865	0.920
50	30	0.04	0.50	95.13	0.821	0.920	0.860	0.839	0.868	0.898
50	30	0.04	0.75	95.13	0.820	0.820	0.820	0.820	0.820	0.820
50	40	0.01	0.25	227.30	0.889	0.960	0.920	0.902	0.923	0.945

Continued on next page

Table C.3: Node, Context, and Content - Dynamic Detection Mechanism Results continued

Start Msg	Win Size	STAB	DEC	Avg Msg	PR	RC	ACC	$F_{0.5}$	F_1	F_2
50	40	0.01	0.50	227.30	0.906	0.960	0.930	0.916	0.932	0.949
50	40	0.01	0.75	227.30	0.922	0.940	0.930	0.925	0.931	0.936
50	40	0.02	0.25	150.68	0.887	0.940	0.910	0.897	0.913	0.929
50	40	0.02	0.50	150.68	0.922	0.940	0.930	0.925	0.931	0.936
50	40	0.02	0.75	150.68	0.917	0.880	0.900	0.909	0.898	0.887
50	40	0.04	0.25	112.04	0.803	0.980	0.870	0.833	0.883	0.939
50	40	0.04	0.50	112.04	0.842	0.960	0.890	0.863	0.897	0.934
50	40	0.04	0.75	112.04	0.843	0.860	0.850	0.846	0.851	0.857
50	50	0.01	0.25	240.06	0.906	0.960	0.930	0.916	0.932	0.949
50	50	0.01	0.50	240.06	0.906	0.960	0.930	0.916	0.932	0.949
50	50	0.01	0.75	240.06	0.922	0.940	0.930	0.925	0.931	0.936
50	50	0.02	0.25	171.57	0.887	0.940	0.910	0.897	0.913	0.929
50	50	0.02	0.50	171.57	0.922	0.940	0.930	0.925	0.931	0.936
50	50	0.02	0.75	171.57	0.938	0.900	0.920	0.930	0.918	0.907
50	50	0.04	0.25	134.95	0.810	0.940	0.860	0.833	0.870	0.911
50	50	0.04	0.50	134.95	0.902	0.920	0.910	0.906	0.911	0.916
50	50	0.04	0.75	134.95	0.894	0.840	0.870	0.882	0.866	0.850
100	10	0.01	0.25	121.29	0.842	0.960	0.890	0.863	0.897	0.934
100	10	0.01	0.50	121.29	0.855	0.940	0.890	0.870	0.895	0.922
100	10	0.01	0.75	121.29	0.878	0.860	0.870	0.874	0.869	0.863
100	10	0.02	0.25	114.48	0.790	0.980	0.860	0.822	0.875	0.935
100	10	0.02	0.50	114.48	0.842	0.960	0.890	0.863	0.897	0.934
100	10	0.02	0.75	114.48	0.860	0.860	0.860	0.860	0.860	0.860
100	10	0.04	0.25	111.82	0.766	0.980	0.840	0.801	0.860	0.928
100	10	0.04	0.50	111.82	0.807	0.920	0.850	0.827	0.860	0.895
100	10	0.04	0.75	111.82	0.827	0.860	0.840	0.833	0.843	0.853
100	20	0.01	0.25	165.02	0.887	0.940	0.910	0.897	0.913	0.929
100	20	0.01	0.50	165.02	0.900	0.900	0.900	0.900	0.900	0.900
100	20	0.01	0.75	165.02	0.898	0.880	0.890	0.894	0.889	0.884
100	20	0.02	0.25	134.07	0.842	0.960	0.890	0.863	0.897	0.934
100	20	0.02	0.50	134.07	0.852	0.920	0.880	0.865	0.885	0.906
100	20	0.02	0.75	134.07	0.894	0.840	0.870	0.882	0.866	0.850
100	20	0.04	0.25	124.18	0.790	0.980	0.860	0.822	0.875	0.935
100	20	0.04	0.50	124.18	0.852	0.920	0.880	0.865	0.885	0.906
100	20	0.04	0.75	124.18	0.854	0.820	0.840	0.847	0.837	0.827
100	30	0.01	0.25	216.74	0.870	0.940	0.900	0.883	0.904	0.925
100	30	0.01	0.50	216.74	0.904	0.940	0.920	0.911	0.922	0.933
100	30	0.01	0.75	216.74	0.902	0.920	0.910	0.906	0.911	0.916
100	30	0.02	0.25	160.36	0.870	0.940	0.900	0.883	0.904	0.925
100	30	0.02	0.50	160.36	0.902	0.920	0.910	0.906	0.911	0.916

Continued on next page

Table C.3: Node, Context, and Content - Dynamic Detection Mechanism Results continued

Start Msg	Win Size	STAB	DEC	Avg Msg	PR	RC	ACC	$F_{0.5}$	F_1	F_2
100	30	0.02	0.75	160.36	0.891	0.820	0.860	0.876	0.854	0.833
100	30	0.04	0.25	138.43	0.875	0.980	0.920	0.894	0.925	0.957
100	30	0.04	0.50	138.43	0.887	0.940	0.910	0.897	0.913	0.929
100	30	0.04	0.75	138.43	0.896	0.860	0.880	0.888	0.878	0.867
100	40	0.01	0.25	267.69	0.889	0.960	0.920	0.902	0.923	0.945
100	40	0.01	0.50	267.69	0.906	0.960	0.930	0.916	0.932	0.949
100	40	0.01	0.75	267.69	0.922	0.940	0.930	0.925	0.931	0.936
100	40	0.02	0.25	189.66	0.870	0.940	0.900	0.883	0.904	0.925
100	40	0.02	0.50	189.66	0.904	0.940	0.920	0.911	0.922	0.933
100	40	0.02	0.75	189.66	0.917	0.880	0.900	0.909	0.898	0.887
100	40	0.04	0.25	156.63	0.857	0.960	0.900	0.876	0.906	0.938
100	40	0.04	0.50	156.63	0.887	0.940	0.910	0.897	0.913	0.929
100	40	0.04	0.75	156.63	0.898	0.880	0.890	0.894	0.889	0.884
100	50	0.01	0.25	281.27	0.889	0.960	0.920	0.902	0.923	0.945
100	50	0.01	0.50	281.27	0.906	0.960	0.930	0.916	0.932	0.949
100	50	0.01	0.75	281.27	0.922	0.940	0.930	0.925	0.931	0.936
100	50	0.02	0.25	219.57	0.870	0.940	0.900	0.883	0.904	0.925
100	50	0.02	0.50	219.57	0.904	0.940	0.920	0.911	0.922	0.933
100	50	0.02	0.75	219.57	0.918	0.900	0.910	0.915	0.909	0.904
100	50	0.04	0.25	175.00	0.839	0.940	0.880	0.858	0.887	0.918
100	50	0.04	0.50	175.00	0.902	0.920	0.910	0.906	0.911	0.916
100	50	0.04	0.75	175.00	0.911	0.820	0.870	0.891	0.863	0.837
150	10	0.01	0.25	170.61	0.857	0.960	0.900	0.876	0.906	0.938
150	10	0.01	0.50	170.61	0.887	0.940	0.910	0.897	0.913	0.929
150	10	0.01	0.75	170.61	0.936	0.880	0.910	0.924	0.907	0.891
150	10	0.02	0.25	163.43	0.831	0.980	0.890	0.857	0.899	0.946
150	10	0.02	0.50	163.43	0.887	0.940	0.910	0.897	0.913	0.929
150	10	0.02	0.75	163.43	0.896	0.860	0.880	0.888	0.878	0.867
150	10	0.04	0.25	161.66	0.803	0.980	0.870	0.833	0.883	0.939
150	10	0.04	0.50	161.66	0.868	0.920	0.890	0.878	0.893	0.909
150	10	0.04	0.75	161.66	0.878	0.860	0.870	0.874	0.869	0.863
150	20	0.01	0.25	212.07	0.870	0.940	0.900	0.883	0.904	0.925
150	20	0.01	0.50	212.07	0.882	0.900	0.890	0.886	0.891	0.896
150	20	0.01	0.75	212.07	0.915	0.860	0.890	0.903	0.887	0.870
150	20	0.02	0.25	181.19	0.842	0.960	0.890	0.863	0.897	0.934
150	20	0.02	0.50	181.19	0.868	0.920	0.890	0.878	0.893	0.909
150	20	0.02	0.75	181.19	0.915	0.860	0.890	0.903	0.887	0.870
150	20	0.04	0.25	173.12	0.817	0.980	0.880	0.845	0.891	0.942
150	20	0.04	0.50	173.12	0.870	0.940	0.900	0.883	0.904	0.925
150	20	0.04	0.75	173.12	0.875	0.840	0.860	0.868	0.857	0.847

Continued on next page

Table C.3: Node, Context, and Content - Dynamic Detection Mechanism Results continued

Start Msg	Win Size	STAB	DEC	Avg Msg	PR	RC	ACC	$F_{0.5}$	F_1	F_2
150	30	0.01	0.25	250.60	0.870	0.940	0.900	0.883	0.904	0.925
150	30	0.01	0.50	250.60	0.904	0.940	0.920	0.911	0.922	0.933
150	30	0.01	0.75	250.60	0.920	0.920	0.920	0.920	0.920	0.920
150	30	0.02	0.25	209.24	0.870	0.940	0.900	0.883	0.904	0.925
150	30	0.02	0.50	209.24	0.902	0.920	0.910	0.906	0.911	0.916
150	30	0.02	0.75	209.24	0.894	0.840	0.870	0.882	0.866	0.850
150	30	0.04	0.25	185.19	0.828	0.960	0.880	0.851	0.889	0.930
150	30	0.04	0.50	185.19	0.902	0.920	0.910	0.906	0.911	0.916
150	30	0.04	0.75	185.19	0.894	0.840	0.870	0.882	0.866	0.850
150	40	0.01	0.25	297.05	0.873	0.960	0.910	0.889	0.914	0.941
150	40	0.01	0.50	297.05	0.906	0.960	0.930	0.916	0.932	0.949
150	40	0.01	0.75	297.05	0.920	0.920	0.920	0.920	0.920	0.920
150	40	0.02	0.25	231.60	0.873	0.960	0.910	0.889	0.914	0.941
150	40	0.02	0.50	231.60	0.904	0.940	0.920	0.911	0.922	0.933
150	40	0.02	0.75	231.60	0.917	0.880	0.900	0.909	0.898	0.887
150	40	0.04	0.25	201.33	0.857	0.960	0.900	0.876	0.906	0.938
150	40	0.04	0.50	201.33	0.904	0.940	0.920	0.911	0.922	0.933
150	40	0.04	0.75	201.33	0.898	0.880	0.890	0.894	0.889	0.884
150	50	0.01	0.25	311.36	0.889	0.960	0.920	0.902	0.923	0.945
150	50	0.01	0.50	311.36	0.906	0.960	0.930	0.916	0.932	0.949
150	50	0.01	0.75	311.36	0.922	0.940	0.930	0.925	0.931	0.936
150	50	0.02	0.25	256.90	0.870	0.940	0.900	0.883	0.904	0.925
150	50	0.02	0.50	256.90	0.904	0.940	0.920	0.911	0.922	0.933
150	50	0.02	0.75	256.90	0.918	0.900	0.910	0.915	0.909	0.904
150	50	0.04	0.25	217.18	0.839	0.940	0.880	0.858	0.887	0.918
150	50	0.04	0.50	217.18	0.902	0.920	0.910	0.906	0.911	0.916
150	50	0.04	0.75	217.18	0.894	0.840	0.870	0.882	0.866	0.850
200	10	0.01	0.25	216.29	0.860	0.980	0.910	0.881	0.916	0.953
200	10	0.01	0.50	216.29	0.887	0.940	0.910	0.897	0.913	0.929
200	10	0.01	0.75	216.29	0.915	0.860	0.890	0.903	0.887	0.870
200	10	0.02	0.25	212.11	0.860	0.980	0.910	0.881	0.916	0.953
200	10	0.02	0.50	212.11	0.870	0.940	0.900	0.883	0.904	0.925
200	10	0.02	0.75	212.11	0.913	0.840	0.880	0.897	0.875	0.854
200	10	0.04	0.25	211.29	0.860	0.980	0.910	0.881	0.916	0.953
200	10	0.04	0.50	211.29	0.885	0.920	0.900	0.891	0.902	0.913
200	10	0.04	0.75	211.29	0.933	0.840	0.890	0.913	0.884	0.857
200	20	0.01	0.25	252.64	0.868	0.920	0.890	0.878	0.893	0.909
200	20	0.01	0.50	252.64	0.882	0.900	0.890	0.886	0.891	0.896
200	20	0.01	0.75	252.64	0.913	0.840	0.880	0.897	0.875	0.854
200	20	0.02	0.25	227.14	0.857	0.960	0.900	0.876	0.906	0.938

Continued on next page

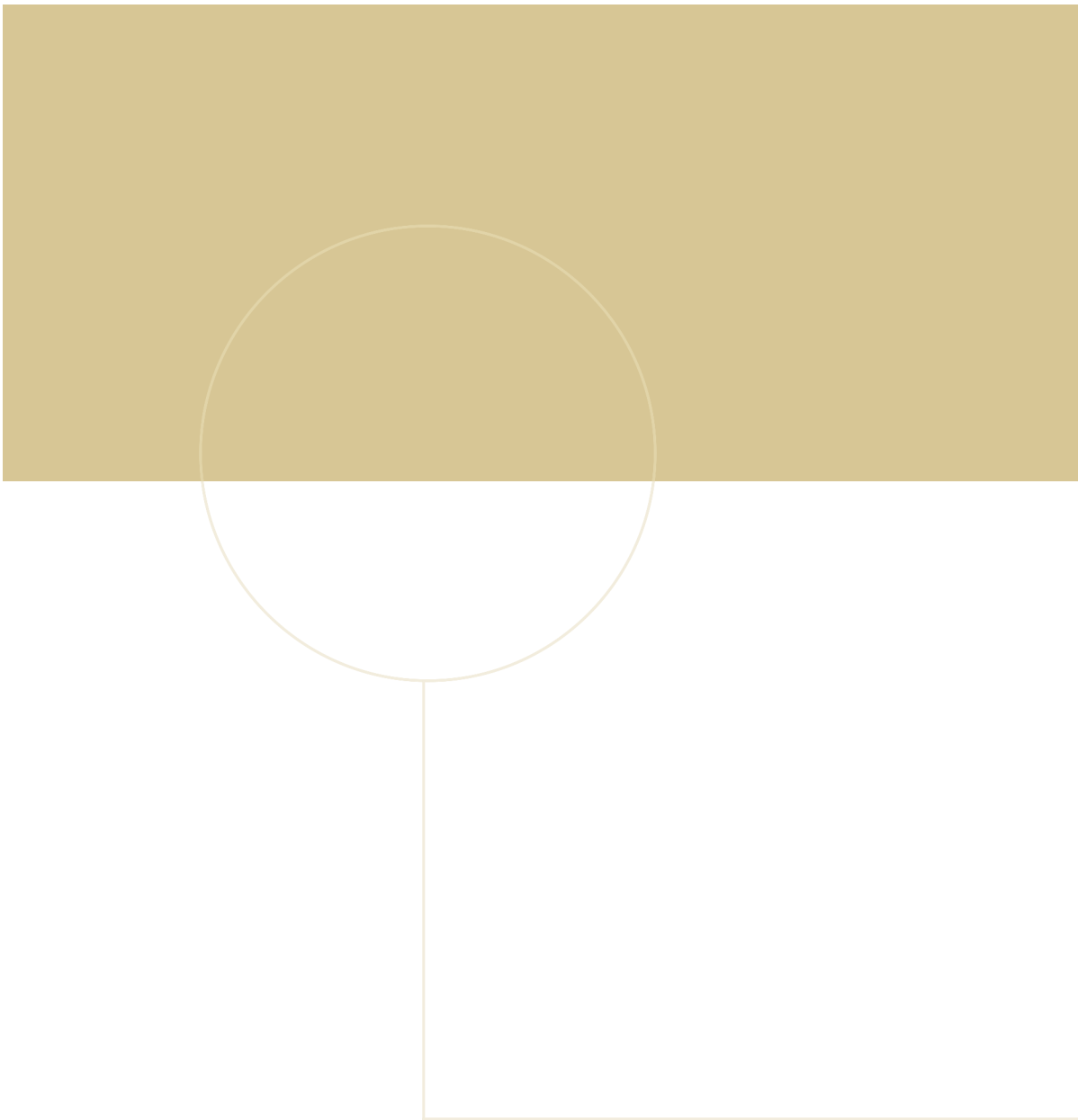
Table C.3: Node, Context, and Content - Dynamic Detection Mechanism Results continued

Start Msg	Win Size	STAB	DEC	Avg Msg	PR	RC	ACC	$F_{0.5}$	F_1	F_2
200	20	0.02	0.50	227.14	0.865	0.900	0.880	0.872	0.882	0.893
200	20	0.02	0.75	227.14	0.911	0.820	0.870	0.891	0.863	0.837
200	20	0.04	0.25	222.47	0.860	0.980	0.910	0.881	0.916	0.953
200	20	0.04	0.50	222.47	0.885	0.920	0.900	0.891	0.902	0.913
200	20	0.04	0.75	222.47	0.932	0.820	0.880	0.907	0.872	0.840
200	30	0.01	0.25	289.00	0.870	0.940	0.900	0.883	0.904	0.925
200	30	0.01	0.50	289.00	0.904	0.940	0.920	0.911	0.922	0.933
200	30	0.01	0.75	289.00	0.920	0.920	0.920	0.920	0.920	0.920
200	30	0.02	0.25	249.35	0.852	0.920	0.880	0.865	0.885	0.906
200	30	0.02	0.50	249.35	0.902	0.920	0.910	0.906	0.911	0.916
200	30	0.02	0.75	249.35	0.894	0.840	0.870	0.882	0.866	0.850
200	30	0.04	0.25	234.38	0.845	0.980	0.900	0.869	0.907	0.950
200	30	0.04	0.50	234.38	0.882	0.900	0.890	0.886	0.891	0.896
200	30	0.04	0.75	234.38	0.911	0.820	0.870	0.891	0.863	0.837
200	40	0.01	0.25	335.39	0.906	0.960	0.930	0.916	0.932	0.949
200	40	0.01	0.50	335.39	0.923	0.960	0.940	0.930	0.941	0.952
200	40	0.01	0.75	335.39	0.939	0.920	0.930	0.935	0.929	0.924
200	40	0.02	0.25	269.57	0.873	0.960	0.910	0.889	0.914	0.941
200	40	0.02	0.50	269.57	0.904	0.940	0.920	0.911	0.922	0.933
200	40	0.02	0.75	269.57	0.915	0.860	0.890	0.903	0.887	0.870
200	40	0.04	0.25	247.38	0.845	0.980	0.900	0.869	0.907	0.950
200	40	0.04	0.50	247.38	0.902	0.920	0.910	0.906	0.911	0.916
200	40	0.04	0.75	247.38	0.915	0.860	0.890	0.903	0.887	0.870
200	50	0.01	0.25	351.88	0.923	0.960	0.940	0.930	0.941	0.952
200	50	0.01	0.50	351.88	0.923	0.960	0.940	0.930	0.941	0.952
200	50	0.01	0.75	351.88	0.940	0.940	0.940	0.940	0.940	0.940
200	50	0.02	0.25	296.93	0.870	0.940	0.900	0.883	0.904	0.925
200	50	0.02	0.50	296.93	0.904	0.940	0.920	0.911	0.922	0.933
200	50	0.02	0.75	296.93	0.917	0.880	0.900	0.909	0.898	0.887
200	50	0.04	0.25	262.62	0.842	0.960	0.890	0.863	0.897	0.934
200	50	0.04	0.50	262.62	0.902	0.920	0.910	0.906	0.911	0.916
200	50	0.04	0.75	262.62	0.913	0.840	0.880	0.897	0.875	0.854
250	10	0.01	0.25	266.97	0.870	0.940	0.900	0.883	0.904	0.925
250	10	0.01	0.50	266.97	0.900	0.900	0.900	0.900	0.900	0.900
250	10	0.01	0.75	266.97	0.932	0.820	0.880	0.907	0.872	0.840
250	10	0.02	0.25	262.82	0.855	0.940	0.890	0.870	0.895	0.922
250	10	0.02	0.50	262.82	0.882	0.900	0.890	0.886	0.891	0.896
250	10	0.02	0.75	262.82	0.913	0.840	0.880	0.897	0.875	0.854
250	10	0.04	0.25	261.54	0.839	0.940	0.880	0.858	0.887	0.918
250	10	0.04	0.50	261.54	0.882	0.900	0.890	0.886	0.891	0.896

Continued on next page

Table C.3: Node, Context, and Content - Dynamic Detection Mechanism Results continued

Start Msg	Win Size	STAB	DEC	Avg Msg	PR	RC	ACC	$F_{0.5}$	F_1	F_2
250	10	0.04	0.75	261.54	0.913	0.840	0.880	0.897	0.875	0.854
250	20	0.01	0.25	295.73	0.855	0.940	0.890	0.870	0.895	0.922
250	20	0.01	0.50	295.73	0.885	0.920	0.900	0.891	0.902	0.913
250	20	0.01	0.75	295.73	0.917	0.880	0.900	0.909	0.898	0.887
250	20	0.02	0.25	275.61	0.870	0.940	0.900	0.883	0.904	0.925
250	20	0.02	0.50	275.61	0.880	0.880	0.880	0.880	0.880	0.880
250	20	0.02	0.75	275.61	0.913	0.840	0.880	0.897	0.875	0.854
250	20	0.04	0.25	272.26	0.855	0.940	0.890	0.870	0.895	0.922
250	20	0.04	0.50	272.26	0.882	0.900	0.890	0.886	0.891	0.896
250	20	0.04	0.75	272.26	0.913	0.840	0.880	0.897	0.875	0.854
250	30	0.01	0.25	325.33	0.870	0.940	0.900	0.883	0.904	0.925
250	30	0.01	0.50	325.33	0.904	0.940	0.920	0.911	0.922	0.933
250	30	0.01	0.75	325.33	0.920	0.920	0.920	0.920	0.920	0.920
250	30	0.02	0.25	299.65	0.855	0.940	0.890	0.870	0.895	0.922
250	30	0.02	0.50	299.65	0.887	0.940	0.910	0.897	0.913	0.929
250	30	0.02	0.75	299.65	0.913	0.840	0.880	0.897	0.875	0.854
250	30	0.04	0.25	283.65	0.839	0.940	0.880	0.858	0.887	0.918
250	30	0.04	0.50	283.65	0.900	0.900	0.900	0.900	0.900	0.900
250	30	0.04	0.75	283.65	0.933	0.840	0.890	0.913	0.884	0.857
250	40	0.01	0.25	369.17	0.906	0.960	0.930	0.916	0.932	0.949
250	40	0.01	0.50	369.17	0.923	0.960	0.940	0.930	0.941	0.952
250	40	0.01	0.75	369.17	0.939	0.920	0.930	0.935	0.929	0.924
250	40	0.02	0.25	312.14	0.870	0.940	0.900	0.883	0.904	0.925
250	40	0.02	0.50	312.14	0.904	0.940	0.920	0.911	0.922	0.933
250	40	0.02	0.75	312.14	0.936	0.880	0.910	0.924	0.907	0.891
250	40	0.04	0.25	296.46	0.839	0.940	0.880	0.858	0.887	0.918
250	40	0.04	0.50	296.46	0.885	0.920	0.900	0.891	0.902	0.913
250	40	0.04	0.75	296.46	0.935	0.860	0.900	0.919	0.896	0.874
250	50	0.01	0.25	386.25	0.923	0.960	0.940	0.930	0.941	0.952
250	50	0.01	0.50	386.25	0.923	0.960	0.940	0.930	0.941	0.952
250	50	0.01	0.75	386.25	0.940	0.940	0.940	0.940	0.940	0.940
250	50	0.02	0.25	336.96	0.887	0.940	0.910	0.897	0.913	0.929
250	50	0.02	0.50	336.96	0.922	0.940	0.930	0.925	0.931	0.936
250	50	0.02	0.75	336.96	0.936	0.880	0.910	0.924	0.907	0.891
250	50	0.04	0.25	308.07	0.836	0.920	0.870	0.852	0.876	0.902
250	50	0.04	0.50	308.07	0.885	0.920	0.900	0.891	0.902	0.913
250	50	0.04	0.75	308.07	0.936	0.880	0.910	0.924	0.907	0.891



NTNU

Norwegian University of
Science and Technology