DOCTORAL THESIS

Metrics and techniques for early detection in cybersecurity

Manuel Fernández López-Vizcaíno

2022

UNIVERSIDADE DA CORUÑA

Metrics and techniques for early detection in cybersecurity

Manuel Fernández López-Vizcaíno

DOCTORAL THESIS

December 2022

PhD Advisors:
Fidel Cacheda Seijo
Francisco Javier Nóvoa Manuel

PhD program in Information and Communications Technologies

UNIVERSIDADE DA CORUÑA

Dr. Fidel Cacheda Seijo
Catedrático de Universidad
Departamento de Ciencias de la Computación y Tecnologías de la Información
Universidade da Coruña

Dr. Francisco J. Nóvoa Manuel
Profesor contratado doctor
Departamento de Ciencias de la Computación y Tecnologías de la Información
Universidade da Coruña

Dr. José Carlos Dafonte Vázquez
Profesor titular de Universidad
Departamento de Ciencias de la Computación y Tecnologías de la Información
Universidade da Coruña

CERTIFICAN

Que la memoria titulada: *"Metrics and techniques for early detection in cybersecurity"* ha sido realizada por D. Manuel Fernández López-Vizcaíno bajo nuestra dirección en el Departamento de Ciencias de la Computación y Tecnologías de la Información de la universidade da Coruña, y concluye la Tesis Doctoral que presenta para optar al grado de Doctor en Ingeniería Informática con la Mención de Doctor Internacional.

En A Coruña, a 13 de diciembre de 2022

Fdo.: Fidel Cacheda Seijo
Director de la Tesis Doctoral

Fdo.: Francisco J. Nóvoa Manuel
Director de la Tesis Doctoral

Fdo.: José Carlos Dafonte Vázquez
Tutor de la Tesis Doctoral

Fdo.: Manuel Fernández López-Vizcaíno
Autor de la Tesis Doctoral

Para M., P. y M.

Acknowledgements

I want to thank my tutor Dafonte for introducing me to the world of research, to Fran and to Fidel for his guide and help to develop this Thesis. I also want to thank the LIA2 group (Dani, Marco, Iker, Ángel, María, ...) and the Telematics group for your good welcome. Specially I have to thank for all the help and apologize to Diego and specially to Laura for all these years coping with my sense of humour.

I would like to thank also Jorge Blasco and the Systems & Software Security Lab (S3Lab) for being so welcoming even when there were little opportunities to see each other faces.

Thanks to all my friends, particularly to David and Diego for dragging me out when needed and for letting me be when necessary, to Raquel and Uxío for caring. And to Nuria, Laura and Jorge, without them I would not had went to certain places, for lending me a hand when needed but also helping me to value activities beyond work.

Finally to the people without who I would not be able to achieve this, my family. First of all to Papá and Mamá for giving me the courage and opportunities to try things, to pursue them and to persevere. To Manuel and Aimée for always being interested in what I do and to encourage me to pursue it. To María for being there and supportive all the time. And to Marcos, thank you for continuing to share the road.

And as a good friend of mine would say... Thanks to all the people who contributed to this thesis, and to the ones who did not, well... Thank you too!

Manuel

This work has received financial support from the Xunta de Galicia (Centro singular de investigación de Galicia accreditation 2016-2019) and the European Union (European Regional Development Fund - ERDF).

I don't know. I can't tell the future, I just work there.

Dr. Who

Resumo

A importancia da detección temperá aumentou nos últimos anos conforme as redes de comunicacións pasaron a formar parte da vida diaria, e por tanto os perigos que supón aumentaron. Neste sentido, non só a seguridade das redes de comunicacións, sistemas e protección de datos están en perigo, senón tamén os seus usuarios. Coa proliferación das comunidades en liña e das redes sociais, os comportamentos que xa supoñían un problema atoparon unha plataforma que intensifica as súas capacidades, superando as limitacións do mundo físico. As probabilidades de producir un dano increméntanse no tempo para calquera tipo de ameaza de seguridade, polo tanto, canto antes se detecte e deteña, as probabilidades de mitigar os problemas xerados aumentan. Neste sentido, o ciberacoso converteuse nun problema urxente na Internet, especialmente nas redes sociais. Para abordar este problema deben definirse procedementos de detección temperá tanto en relación a métodos de detección coma métricas para medir o seu rendemento dende o punto de vista da detección consciente do tempo. Co obxectivo de alcanzar isto, por unha parte o problema da detección temperá definiuse formalmente e estudáronse diversas alternativas para a súa avaliación. En canto as métricas de detección temperá, estudáronse métricas de última xeración como Early Risk Detection Error (ERDE) e F-latency e propuxéronse alternativas coma NormalizedERDE, Time aware Precision (TaP) e Time aware F-score (TaF) para resolver problemas detectados nas outras métricas. Para mellorar os resultados obtidos coa utilización das métricas conscientes do tempo preséntanse tres modelos: de punto fixo, limiar e dual. Ademais, estudouse a incorporación de conxuntos de características para a detección temperá do ciberacoso en redes sociais: Doc2Vec e Multiple Instance Learning.

Resumen

La importancia de la detección temprana ha aumentado en los últimos años conforme las redes de comunicaciones han pasado a formar parte de la vida diaria, y por tanto los peligros que conllevan han aumentado. En ese sentido, no solo la seguridad de las redes de comunicaciones, sistemas y protección de datos están en peligro, si no también sus usuarios. Con la proliferación de las comunidades en línea y las redes sociales, aquellos comportamientos que ya suponían un problema han encontrado una plataforma que intensifica sus capacidades, superando las limitaciones del mundo físico. Las probabilidades de producir un daño se incrementan en el tiempo para cualquier tipo de amenaza de seguridad, por tanto, cuanto antes se detecte y detenga, las probabilidades de mitigar los problemas generados aumentan. En este sentido, el ciberacoso se ha convertido en un problema urgente en Internet, especialmente en las redes sociales. Para abordar este problema han de definirse procedimientos de detección temprana tanto en cuanto a métodos de detección como a métricas para medir su rendimiento desde el punto de vista de la detección consciente del tiempo. Con el objetivo de alcanzar esto, por una parte, el problema de la detección temprana se ha definido formalmente y se han estudiado diversas alternativas para su evaluación. En cuanto a las métricas de detección temprana, se han estudiado métricas de última generación como Early Detection Risk Error (ERDE) y F-latency y se han propuesto alternativas como NormalizedERDE, Time aware Precision (TAP) y Time aware F-score (TAF) para resolver problemas detectados en las otras métricas. Para mejorar los resultados obtenidos con la utilización de las métricas conscientes del tiempo se presentan tres modelos: modelo de punto fijo, modelo umbral y modelo dual. Además, se ha estudiado la incorporación de dos conjuntos de características para la detección temprana del ciberacoso en redes sociales: Doc2Vec y Multiple Instance Learning.

Abstract

Early detection importance has grown in the last years and, as communication networks had become part of everyday life, threats that come within had increased. In this sense, not only the security of networks, systems and data protection is in danger but also its users. With the spread of online communities and social networks, behaviours that already were a problem found a platform to grow as the capabilities of the platform allows to expand the limits of the physical world. Any security threat increases its chances of damage over time, so the earlier it is detected and stopped, there are more chances that outcome problems can be mitigated. In this sense, cyberbullying has become an urgent matter on the Internet and specially on social media networks. To approach this problem, a formal early detection procedure should be defined both in terms of methods of detection and metrics to measure the performance from the point of view of time aware detection. To achieve that, on the one hand, the early detection problem has been formally defined and multiple alternatives for the evaluation studied. Regarding early detection metrics, state of the art Early Detection Risk Error (ERDE) and F-latency had been reviewed and alternatives such as NormalizedERDE, Time aware Precision (TAP) and Time aware F-score (TAF) had been proposed to overcome problems detected. In order to improve results obtained with time aware metrics, the use of three early detection models is presented: fixed point model, threshold model and dual model. Finally, two sets of features for early detection of cyberbullying in social networks is considered: Doc2Vec and Multiple Instance Learning.

Acronyms

AB	**A**da **B**oost
BoW	**B**ag **o**f **W**ords
DDOS	**D**istributed **D**enial **O**f **S**ervice
DOS	**D**enial **O**f **S**ervice
EMDD	**E**xpectation-**M**aximization **D**iverse **D**ensity
ERDE	**E**arly **R**isk **D**etection **E**rror
ET	**E**xtra **T**ree
EWS	**E**arly **W**arning **S**ystems
FN	**F**alse **N**egative
FP	**F**alse **P**ositive
HIDS	**H**ost-based **I**ntrusion **D**etection **S**ystem
IDS	**I**ntrusion **D**etection **S**ystem
IoT	**I**nternet **o**f **T**hings
kNN	**k** **N**earest **N**eighbours
LDA	**L**atest **D**irichlet **A**llocation
LR	**L**ogistic **R**egression
LSVC	**L**inear **S**upport **V**ector **C**lassification
MIL	**M**ultiple **I**nstance **L**earning
MILES	**M**ultiple **I**nstance **L**earning via **E**mbedded Instance **S**election
ML	**M**achine **L**earning
NIDS	**N**etwork-based **I**ntrusion **D**etection **S**ystem
OS	**O**perative **S**ystem
RF	**R**andom **F**orest
SDN	**S**oftware **D**efined **N**etworks
TaF	**T**ime **a**ware **F**-score
TaP	**T**ime **a**ware **P**recision
TN	**T**rue **N**egative
TP	**T**rue **P**ositive

Contents

List of Figures

List of Tables

Chapter 1

Introduction

As communication networks become part of everyone life across the world, many threats come within. Therefore, cybersecurity is an essential part to maintain not only systems but also users with an appropriate level of security. In order to achieve that, those threats need to be detected and addressed as soon as possible, reducing the damage generated by them.

1.1 Aspects of cybersecurity

Cybersecurity can be defined as "things that are done to protect a person, organization, or country and their computer information against crime or attacks carried out using the internet" [1]. Even though usually this term is used only to refer to physical or logical systems that provide certain capabilities to users, it should be applied to the protection of users themselves too. Also, on appendix A of ISO 27001/2017 [2], which defines organizations information assets management, people (human resources) are classified as tangible assets of information. And points that they must be protected in the same way as the rest the of information assets of the organization.

Systems security constitutes the base of what is usually known as cybersecurity and addresses the preservation of integrity and working order of IT structures. Any attack towards these systems follows a set of phases that had been previously described [3]: Reconnaissance, Weaponization, Delivery, Exploitation, Installation, Command and Control (C2) and Actions on Objectives. If a threat is detected in an initial phase, further exploitation could be avoided.

From the point of view of user protection, an important threat comes from Social Media Networks where users interaction, amplified by the capability of these systems, could lead to mental-health related problems. In this environment, one of the events with more prevalence is cyberbullying [4] and it can lead to potentially devastating psychological consequences, such as depression, low self-esteem, suicide ideation or suicide [5].

In both cases, it would be beneficial the early detection of these situations, as it would reduce the damage by reducing the negative impact produced.

1.2 Early Detection Problem

The early detection problem could be divided in two main parts. How to detect and how to measure if the detection was performed not only in a proper manner but in an adequate time. The first point has to answer to *when* and *with which means* the detection will be made and the second to *which metrics* will be used and to *which data* will be applied.

As it has been mentioned, several fields could be affected on the moment a prediction is taken. In some cases even if the prediction is correct, the time used to take that prediction makes it useless or diminishes its value. In general, positive results are more sensitive to time delays as a negative one should not produce any problems in the short term.

The approach to this problem will be to study, asses and define, if needed, new metrics to score models. Also, when it must be selected when and how the evaluation will be applied. Defining the point of evaluation is not only not trivial but it presents different alternatives in granularity which alter the results obtained by the metric. Those approaches present more or less realistic representation of the way the information is acquired and processed by the systems, and therefore, evaluated.

In this case, penalization could be defined either by the time it takes to make a prediction or by the amount of items processed for a particular choice. As it will be seen later on, these two approaches have similarities in terms of how metrics behave and how the results represent the field idiosyncrasy.

1.3 Objectives

In this thesis the early detection problem is addressed in general, an applied in particular to the cybersecurity field. In specific, both network security and social networks user protection will be addressed.

To do so, metrics had been analysed and defined alternatives to address the problems found. These metrics had been used to study how well several methods perform. Starting with classic non time aware metrics, it follows the process of how existent time aware metrics are defined to reach the variations or new metrics proposed.

Then, different methods of evaluation based on granularity are studied for theses metrics, showing problems related to them and how if affects to the general outcome.

Also, in the case of cyberbullying, different approaches had been followed to improve the results in case of an early detection techniques for cyberbullying detection. Both based on different features and model definition.

1.4 Contributions

The publications related to this thesis and its contributions can be described as follows:

- *"Breaking the Cyber Kill Chain: Early Intrusion Detection for Scan Attacks"* [6]: Introduction to the early detection methodology, analysis of $ERDE$ and $F_{latency}$ metrics and presentation of new metric *NormalizedERDE*.
- *"Measuring early detection of anomalies"* [7]: Deep analysis of both batch and streaming evaluation methodologies and proposal of *Time aware Precision* or TaP.
- *"Time aware F-score for cybersecurity early detection evaluation"* [8]: Evaluation of new metric, *Time aware F-score* or TaF.
- *"Early detection of cyberbullying on social media networks"* [9]: Study of early detection methods for cyberbullying under a time aware evaluation.
- *"Site agnostic approach to early detection of cyberbullying on social media networks"* [10]: Analysis of three early detection models alternatives (fix point, threshold and dual models), Doc2Vec features and Multiple Instance Learning for early detection performance enhancement.

1.5 Thesis outline

From this chapter onwards the rest of the thesis is presented as follows:

- Chapter 2 Related work: a review of research and development in the field of early detection are presented, followed by a revision of the two facets of cybersecurity approached in this thesis: networks security and cyberbullying.

- Chapter 3 Early Detection Measure: Formal presentation of the early detection problem, presentation of different evaluation methodologies and metrics.
- Chapter 4 Metrics evaluation: Results of the metrics presented on the previous chapter with an analysis an comparison between them.
- Chapter 5 Early Detection Methods: Introduction of model alternatives for the early detection problem and feature analysis to improve results in a time aware evaluation.
- Chapter 6 Methods evaluation: Results for the methods and sets of features presented on Chapter 5 with a performance comparison between them by using time aware metrics.
- Chapter 7 Conclusions and future work: General conclusions for the metrics and methodologies presented on Chapter 3 and for the models and features presented on Chapter 5. Also a hint of future developments is introduced by the end of this chapter.

Chapter 2

Related work

In this chapter a thorough review of related work is presented and analysed from the point of view of the cybersecurity and early detection. For the first point and as described in Chapter 1, both systems and users security will be considered, as cybersecurity is described as the “things that are done to protect a person, organization, or country and their computer information against crime or attacks carried out using the internet” [1].

2.1 Early Detection

The early detection problem is present in different fields and although it has been explored in many of them there is limited research on its evaluation metrics and methodologies. The most researched topic of these two is the definition of evaluation metrics, among which the best effort was provided by *Early Risk Detection Error* (*ERDE*) and $F_{latency}$ metrics.

ERDE metric was defined for the workshop on early risk prediction on the Internet (*eRisk*), as part of the Conference and Labs for the Evaluation Forum (*CLEF*) in 2017. There, a task for early detection of different conditions (e.g. symptoms of depression, self-harm, anorexia, etc.) over social network data was proposed and results were measured with time aware methodology and metrics [11, 12]. The methodology proposed consisted on a series of batches where information for each user was divided into 10 splits and analysed sequentially with the proposed metric which is an error ratio.

$F_{latency}$ is a latency-weighted $F1$ defined to overcome some problems detected by the authors with the definition of *ERDE* metric for the evaluation of depression detection in social media [13]. One of the goals of this metric was to avoid heuristically defined parameters for the metrics as it was used in *ERDE* metric, replacing them by a dataset defined parameter. Also, the evaluation environment with this new metric definition does not propose any new evaluation methodologies.

In terms of metrics and methodologies those two, $ERDE$ [11] and $F_{latency}$ [13], are the best effort presented for early detection evaluation with specific latency dependent metrics. Both of them are more deeply described in Chapter 3, where alternatives to batch evaluation are also presented.

Disregarding specific latency aware metrics, several attempts have been made using traditional metrics as precision, recall or F-score [14] for time aware systems evaluation. In that sense, for the detection of cyberbullying, attempts like the presented by Samghabadi et al. [15] where the creation of a corpus for the early detection of cyberbullying is proposed but where the evaluation is limited to the use of $F1$ at fixed points without any kind of latency penalisation. Within the social media analysis for early detection, some works over rumours and fake news has also been presented. For fake news detection Zhou et al. [16] present an interdisciplinary study but the analysis of early detection is reduced to news articles and news content information. Also, in this case no proper time-aware evaluation in present. In the case of rumours, Zhao et al. [17] study different options for unconfirmed information detection but the latency evaluation is performed only by means of the time required to detect the rumour.

In the field of network and systems security some research has been presented for the early detection of attacks, as the threat increases with the time pass or the phases achieved [3]. Some works, such as [18] and [19], research into early detection of cyberattacks by means of a discovery of the attack in its early stages. Also, [20] presents a prototype for the early detection recognition of cyberattacks. But all of them without considering the time required for the detection with the use of non-time aware metrics. In order to avoid malware propagation some research on Early Warning Systems (EWS) has been presented using different alternatives such as bayesian inference [21], Kalman filter [22] or sensors [23]. These solutions also focus its evaluation on the identification of attacks in a timeline but without considering latency penalisations for the metrics.

Finally, related to network security, smart cities is another field where early detection is specially interesting. In this situation the increment on number of devices connected to the network as well as the substantial increment in traffic through the network leads to possible threats that could have critical effect on their performance of security. In this sense, Xu proposes the use of software-defined network function virtualization (SDNFV) architecture [24] and a traffic classification strategy specifically for early detection of attacks. Privalov et al. follows a similar approach for the detection of distributed denial of service attacks [25]. However in both cases time-aware evaluation is limited to the measure of the time. In the first case, to the measure of the average response time between attack and detection and in the second case, the time until the detection of a distributed denial of service attack (DDOS).

In short, there are several attempts on the literature towards achieving an early detection evaluation in general and in the field of cybersecurity in particular. Among them, the definition of $ERDE$ [11] and $F_{latency}$ [13] must be highlighted. Despite these approaches there are no formal definition of the problem nor systematic method for early detection evaluation, which leads to the use or no time-aware metrics.

2.2 Cybersecurity

2.2.1 Network Security

The number of devices connected to communication networks has increased steadily in the last years, just the number of *Internet of Things* (*IoT*) devices has reached 13 billions in 2022. Also, it is predicted that this number will triple from its value in 2020 to 2030 with 29 billions expected by the forecast of Statista and Transforma Insights [26] .

An intrusion detection system (IDS) is a mechanism meant to identify abnormal or suspect activities in the analysed object (a network or a host). It thus provides knowledge of successful intrusions as well as failed attempts. IDSs, which have gone through considerable evolutions, include many forms and can act at many levels. [27]. They can be classified into two groups depending if they are focused on locating anomalies in a host (Host based IDS or HIDS) or in a network (Network based IDS of NIDS). The last one, as said, focuses on the discovery of unauthorized or anomalous accesses to computer networks by analysing the traffic to detect harmful connections [28, 29].

Different approaches have been taken to this problem with solutions based in Machine Learning [30, 31], Artificial Neural Networks [32] and Deep Learning [33, 34] for detecting this kind of network traffic. Particularly in the case of Machine Learning and Neural Networks, both supervised and unsupervised approached had been explored [35, 36]. Supervised models had shown great performance in this field but some mention some difficulties regarding certain environments and particular attacks where no tagged data is to be found [37, 38, 39, 40].

On the other hand, unsupervised methods had been also widely applied to networks security as shown in [41, 42]. In [43], for example, an unsupervised anomaly detection system based on the use of k-nearest neighbout (kNN) is presented. Even a mixed semi-supervised learning system as presented on [44] or the approach used in [34] for feature extraction.

Although all these solutions are introduced as IDSs for network monitoring and that these systems are time dependent as presented on [3] their evaluation are not based on a time aware metric. Instead, they mostly use traditional correctness metrics such as *accuracy*, *precision* or $F - score$.

In fact, many research work presents early detection of network security threat as an improvement, particularly for specially time sensitive attacks as Distributed Denial os Service (DDOS) attacks are. Particularly, in [45] a system to automatically and collaboratively perform an early detection of DDoS threats at ISP level instead of the classical victim-end approach. Also, in [46] a method for early detection of low rate DDoS attacks is presented, particularly for Software Defined Networks (SDN). Which becomes relevant as SDN networks are widely used, even in hybrid approaches as it is explored in [47]. Even if this particular type of attack is more prone to be studied from a direct approach of early detection any security threat benefits from it, as shown for an different environment such as Internet of Things (IoT) in [48]. In this case presents a solution motivated by the Mirai-based DDoS attack performed by devices compromised with that malware. In this case a solution based on the detection of the infection in an early stage, such as scanning or infecting, is presented in order to avoid a future DDoS attack.

Related to the features used, even though traditional features such as source and destination address, as well as ports and protocols and other data extracted from network packets headers, other features are included in order to improve detection of attacks. For example, in [34], an autoencoder is used to extract features from header information in order to use its output on a classifier to detect if the traffic is normal or anomalous. Other research, as [49] presents the inclusion of time related features with the incorporation of inter-arrival times for packets. And [50] presents an approach to improve early detection of port scanning attacks, even if the procedure is achieved by a slow scan or when an increase or decrease in the frequency is present.

In terms of datasets there are numerous examples of available network data with different kinds of attacks to analyse or to develop models to detect that kind of traffic. Among others, UNB ISCX [51] which is a dataset for intrusion detection systems based on anomalies or a more recent on related to the profile of IoT devices [52]. Both developed by the Canadian Institute for Cybersecurity based on the University of New Brunswick [53]. Also, Kitsune dataset [34] which provides both a set of files containing captured network packets (PCAPs [54]) and a set of features dynamically extracted from network data by means of auto-encoder networks.

2.2.2 Cyberbullying

Within the user protection facet of cybersecurity cyberbullying has drawn the attention of researches in the last years, due to the increment in use of social media and the need of automatic detection of cyberbullying to keep these platforms safe for all the users.

In terms of used features for this task, Dinakar et al. [55] use for cyberbullying detection on YouTube comments term frequency-inverse document frequency combined with identification of profane words. The work presented by Zhong et al. [56] uses a combination of bag of words (BoW) with Word2Vec features, which poses a vectorial representation of texts as described in [57]. Added to the textual features, they conclude that images and captions can be used as a powerful predictor for future cyberbullying. Also, related to the use of alternatives to textual features, network-based features has been studied and the work presented in [58] shows how they can be a convenient tool for the detection of aggressive behaviours. Lexical syntactical features had also been used for offensive language prediction with Support Vector Machines (SVM) achieving high precision results [59].

Besides, other features commonly used for the detection of cyberbullying, as introduced in [55], are profanity [60, 61, 62, 63] and sentiment analysis [64, 65, 66, 67, 68, 69], which includes content polarity, emotion dictionaries or usser psychological characteristics.

Finally in terms of features used and the relevant research on the use of machine learning techniques for the detection of cyberbullying on social media, some thorough reviews had been published recently. For example, the ones provided by Arif in [70] and Singh et al. in [71].

From these works it can be generalized that most of them use content-based features as Tf-idf, Word2vec or other textual features as well as sentiment analysis. Others include features extracted from the user itself like their relation within the social network (e.g. number of followers or following profiles) or data from their profile (e.g. age or gender). It must be noticed that some of this, mostly the ones extracted from the user profile could not be present or be misleading.

It is interesting to note that those previous works presented do not use time aware evaluation or time related features. Although some of them acknowledge the importance of time features for cyberbullying detection [72, 73, 74, 75] not many of them include this type of characteristics or even perform a time aware evaluation. As such, works concentrate on obtaining a better performance in terms of post or session classification into cyberbullying or non-cyberbullying classes, disregarding the delay in the decision for the element being classified. To achieve so, usually standard evaluation metrics such as *precision*, *recall*, *F-score* or area under the curve (AUC) are used [76, 77, 78].

Among the works that provide a time-aware evaluation, and apart from the ones presented with the $ERDE$ [11, 12] and $F_{latency}$ [13] metrics, some such as [15] can be found. Here, the authors try to minimise the damage caused to victims of cyberbullying by performing an early detection. In order to provide a time-aware evaluation, the original dataset is divided in 10 independent subsets (or batches), as was presented in the $ERDE$ evaluation too [11], that are processed in a sequential manner. Even though a division of the dataset to provide intermediate scores is performed, the metric used is $F1$ without any penalisation for the delay in late predictions, and best scores are obtained usually when at least half the dataset has already been processed.

Lastly, some works refer directly to the early detection but fail to provide a proper evaluation framework of metrics and methods of evaluation. For example in [79] the evaluation provided for early detection of stress and depression is based on *precision*, *recall* and $F1$ without any delay penalisation. Other, such as, [80] claim to be working on the early field, particularly in the cyberbullying problem, but the results are focused on network security by means of cyberattacks detection. Also, their evaluation is based on the time to make the prediction without considering all the aspects of classification. Other authors proposed a method named *HENIN* [81], to provide early detection of cyberbullying. Still, the evaluation is based on *precision* and *accuracy* measured at a determined point k and without any penalty introduced in these metrics for delays in the prediction.

Although most of the presented works on this topic are focused on textual features there are some authors who suggest that audio-visual features can help to improve the performance of cyberbullying detectors that are based just on textual features [82]. But this has also some disadvantages as not all platforms use the same audiovisual resources or even they could not be present at all.

Regarding datasets, Rafiq et al. [83, 84] provide a dataset for the Vine social network [85, 86] a platform to share short videos with description and comments that was available until 2019. They also study the detection of cyberbullying obtaining the best models using *Ada Boost* and *Random Forest*. Following this example, Hosseinmardi et al. [87] describes an Instagram [88, 89] dataset, a social network were images and videos can be shared with descriptions and reactions to the posts can be added. Also, based on this dataset, they test a multi-modal text and images features as well as media session data for cyberbullying detection, obtaining relevant results with SVM, which outperformed the rest of proposed classifiers.

Chapter 3

Early Detection Measure

In this chapter the Early Detection Evaluation problem will be approached. First, a formal representation of the Early Detection problem will be presented, followed by different methods of evaluation. Finally, the state of the art early detection metrics will be analysed in detail and the newly defined ones will be introduced.

3.1 Formal Early Detection Representation

The problem of early detection for a generic system and non-specific type of elements and be formally defined as follows.

Let $E = \{e_1, e_2, ..., e_{|E|}\}$ be the set of entities susceptible of being classified, where $|E|$ denotes the number of entities. Each entity $e \in E$ is formed of a sequence of items, denoted as I_e, and an indicator l_e that denotes the class to which belongs the entity. In this case, the indicator l_e denotes a binary label marking whether the specific entity is considered anomalous ($l_e = true$) or not ($l_e = false$). It must be added that although this represents a binary classification task, early detection systems could provide a third value showing that the decision has not been taken yet (i.e. a delay).

In this scenario E^+ represents the set of anomalous entities (i.e. where $l_e = true$), and E^- the set of non-anomalous entities:

$$E^+ = \{e_1^+, e_2^+, ..., e_{|E^+|}^+\}, \forall\, e_i^+ \in E^+\; l_{e_i^+} = true$$

$$E^- = \{e_1^-, e_2^-, ..., e_{|E^-|}^-\}, \forall\, e_i^- \in E^-\; l_{e_i^-} = false$$

The sequence of items for a specific entity is supposed to change through time and is given by $I_e = (< I_1^e, t_1^e >, < I_2^e, t_2^e >, ..., < I_n^e, t_n^e >)$, where the tuple $< I_k^e, t_k^e >, k \in [1, n]$ represents the k-th item for entity e, and t_k^e is the timestamp associated with item I_k^e. Also, it is important to notice that the following statement must be true:

$$\forall < I_k^e, t_k^e > : \; t_k^e \; before \; t_{k+1}^e$$

Timestamps $t_1^e, t_2^e, ..., t_n^e$ can be equally and homogeneously distributed (e.g. in case of sensor emitting temperature and humidity data every second) or can be uneven and randomly distributed (e.g. in case of the posts written by a user).

An item, I_k^e, is characterized by a vector of features and, in this case, it can be assumed that all items associated with an entity, $I_k^e, k \in [1, n]$, are defined by the same vector of features, whose values may, and predictably will, change through time.

$$I_k^e = \left[f_{k_1}^e, f_{k_2}^e, ..., f_{k_m}^e\right], k \in [1, n]$$

Since entities are independent, each sequence of items I_e may have different lengths, n, for each entity $e \in E$. However, note that the number of features, m, would be the same for all items.

The superscript e in the specification of I_e (e.g. $I_e = (< I_1, t_1 >, < I_2, t_2 >, ..., < I_n, t_n >)$ could be dropped whenever e is clear from the context. Similarly, the vector of features for a specific item I_k^e, will drop the superscript when it is clear from the context: $I_k^e = [f_{k_1}, f_{k_2}, ..., f_{k_m}]$.

Given an entity e, the objective is to detect if the entity has an anomalous behaviour but scanning as few items from I_e as possible.

The objective function will be defined as $f(l_e, I_e \times [1..n]) \to \{0, 1, 2\}$. This function will return 1 (i.e. *positive*) if entity e is considered anomalous after processing items I_1 to I_k. In case the entity e is considered normal (i.e. non-anomalous or *negative*) after processing item k and previous ones, then $f(l_e, I_e, k) = 0$. Finally, $f(l_e, I_e, k) = 2$ if no definitive decision can be emitted on entity e after reading k items and more items must be processed (i.e. *delay*).

Therefore, for $f(l_e, I_e, k)$ outputs 0 and 1 are considered final and items $I_{k+1}, ..., I_n$ do not need to be processed. On the other side, if output 2 is provided, further items $I_{k+1}, ..., I_n$ must be processed, until a final output is achieved or the end of the items sequence is reached.

3.2 Batch vs Streaming

The literature collects mainly two types of evaluation on different early detection problems: batch or streaming, as described respectively in [11, 90].

In the batch evaluation the sequence of items, I_e, is divided into various homogeneous and consecutive groups, and each one is processed independently. Therefore, for each entity e, $I_e = (< I_1^e, I_1^e >, < I_2^e, t_2^e >, ..., < I_n^e, t_n^e >)$ is split into B batches, where batch j, B_j, is defined as the sequence of items occurring between time $[(j-1) \cdot n/B + 1]$ and $[j \cdot n/B]$:

$$B_j = \left(< I_{\left[\frac{(j-1)\cdot n}{B+1}\right]}, t_{\left[\frac{j-1)\cdot n}{B+1}\right]} >, ..., < I_{\left[\frac{j\cdot n}{B}\right]}, t_{\left[\frac{j\cdot n}{B}\right]} > \right)$$

Since entities in E are independent and can have different lengths, batches will be homogeneous for each entity e, but batches may have different sizes for each entity, as it can be seen in Figure 3.1.

Each batch is processed individually and, typically, all previous batches will be considered by the early detection model. Therefore, the function $f(l_e, B_j, \left[\frac{j\cdot n}{B}\right]),\ j \in [1, B]$ must be processed until a final output is obtained or until B is reached.

In the experiments included in Chapter 6, $B = 10$ has been selected for the test-set following previous works [11], with each batch containing 10% of the total *items* for the *entity*.

For the streaming evaluation the sequence of items, I_e, is processed individual and sequentially. Consequently, for each entity e, the function $f(l_e, I_e, k),\ k \in [1, n]$ must be processed, until a final output is obtained or until the maximum number of *items* (n) is reached. As in the previous cases, when processing item k, I_k, all previous items, $I_1, ..., I_{k-1}$, can be taken into account by the model, as shown in Figure 3.2.

It is interesting to note that, in a batch evaluation, when a final result is provided, the exact batch used is identified (e.g. B_j). However, the exact item from the batch used to produce this decision can not be identified and, therefore, for evaluation purposes, the last item of the batch will be considered (e.g. $I_{\left[\frac{j\cdot n}{B}\right]}$). On the other side, in streaming evaluation, the exact item, I_k, used to reach a final decision is certainly determined.

Lastly, and out of the scope of this thesis, time evaluation should be taken into account. In that case, the sequence of items, I_e would be processed individual and sequentially as for streaming evaluation, but with a slight difference when selecting which elements belong to the point of evaluation. Instead of selecting a predefined number of items for each entity, a timestamp will be provided, and all the items before that point will be included. That

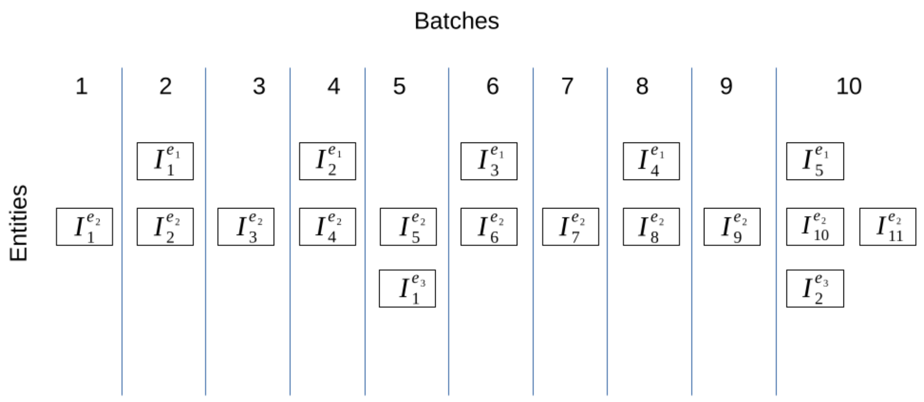


Figure 3.1: Item distribution for batch evaluation.

means that at a specific point in time each entity E could have a different number of items, as it can be seen in Figure 3.3.

3.3 Metrics

3.3.1 Non time aware metrics

In many cases non time aware metrics are employed to evaluate time dependent systems. Usually what is done is to measure the results at a fixed point but without taking into account the amount of items processed to that moment.

Some examples of this type of measures are classic metrics as precision, recall and F1 which is an harmonic mean of the previous two. For the sake of the description and as they will be mentioned later on, the definitions of these three metrics are included next:

$$precision = \frac{|TP|}{|TP| + |FP|}$$

$$recall = \frac{|TP|}{|TP| + |FN|}$$

$$F1 = 2 \cdot \frac{precision \cdot recall}{precision + recall}$$

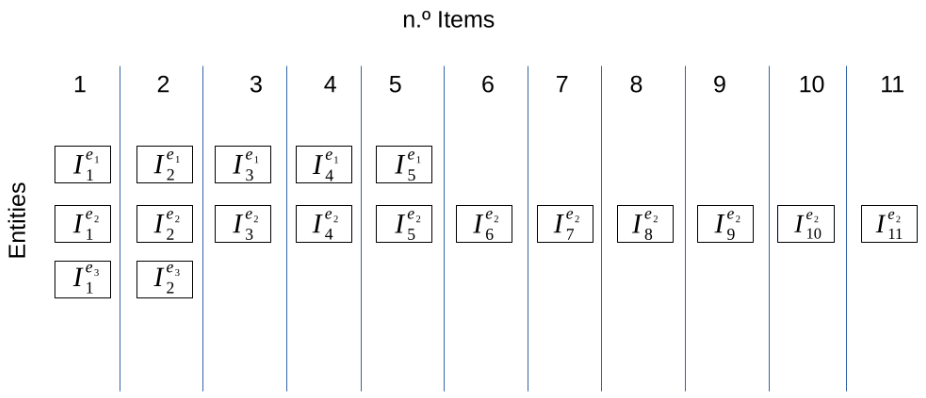


Figure 3.2: Item distribution for streaming evaluation.

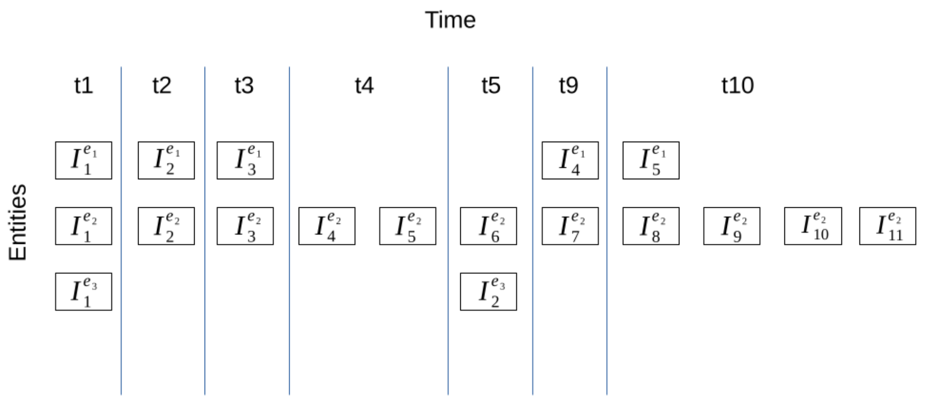


Figure 3.3: Item distribution for time evaluation.

Also a generalization of F1 called F_β can be found, where β can be used to weight in precision or recall for positive values.

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

3.3.2 ERDE

One of the main metrics identified in the state-of-the-art for early detection problem is *Early Risk Detection Error* or $ERDE$ defined in [11]. This metric is measured at a specific point denoted as o and considers four different cases:

$$ERDE_o(e_i, k) = \begin{cases} \frac{\sum\limits_{e_i \in E \wedge l_{e_i} = true} 1}{|E|} & if\ FP \\ 1 & if\ FN \\ 1 - \frac{1}{1+e^{k-o}} & if\ TP \\ 0 & if\ TN \end{cases}$$

In case of wrong predictions (false positive, FP, and false negative, FN), the error increases as expected, but in two different ways. False positives increase the error proportionally to the number of positive cases in the dataset, while false negatives increase the error by 1. A true negative (TN) prediction, does not increase the error independently of when it was produced. However, a true positive (TP) will impact negatively if the delay required to make the prediction exceeds the measuring point o (i.e. $k > o$), using a sigmoid function to introduce the penalty.

Note that $ERDE$ metric ranges from 0 to +1 and, as an error measure, values closer to zero are considered preferable values. This differs with how many well known metrics present their values, for example the ones presented previously.

3.3.3 F-latency

The latency-weighted F1, also denoted in short $F_{latency}$ or *F1-latency* metric, is proposed as an alternative to the ERDE metric that combines a measure of speed, latency and measures of accuracy as precision and recall [13]. Regarding the early detection problem, F-latency metric is defined as:

$$F_{latency}(f,k) = F1 \cdot (1 - \underset{l_f = true}{median}(-1 + \frac{2}{1+e^{-p \cdot (time(f,k)-1)}}))$$

Where, $F1$ is the standard *F-measure* that is calculated based on precision (P) and recall (R): $F1 = \frac{2 \cdot P \cdot R}{P \cdot R}$. $F_{latency}$ requires a parameter p that defines how quickly the penalty should be increased and is recommended to be set to achieve a 50% of latency penalty at the median number of items per entity. This parameter is dataset dependent, having an specific value for each set of entities evaluated. In order to compute the value ot this parameter and from the penalty function the following equation can be extracted:

$$p = -\frac{log(1/3)}{k_{median} - 1}$$

Where p is the parameter to configure the penalty in $F_{latency}$ metric and k_{median} the median of *items* per *entity* for this specific dataset.

3.3.4 Normalized ERDE

ERDE, as seen in section 3.3.2, increases the error as time required to make a correct prediction increases. However, this calculation depends on the specific dataset when computing the false negative penalization, as it is obtained from the proportion of positive cases in the dataset. Therefore, it does not allow a direct comparison of results when using different datasets or even when changing the o parameter.

In order to address this problem, a new metric, denoted as *Normalized-ERDE*, is defined [6]. To compute it, the maximum and minimum values of the metric are used to extract a transformed result by means of the standard min-max normalization for the $ERDE$ metric:

$$NormalizedERDE_o(f, k) = \frac{ERDE_o(f, k) - min(ERDE_o())}{max(ERDE_o() - min(ERDE_o())}$$

Where, $min(ERDE_o())$ and $max(ERDE_o())$ represent, respectively, the minimum and maximum of the $ERDE$ metrics, and $ERDE_o(f, k)$ the specific value of the original $ERDE$ measure being converted to *Normalized-ERDE*.

The proposed metric ranges from 0 to 1, with values closer to 0 representing a better performance, while values closer to 1 correspond to a poor result, as was expected from an $ERDE$ measure. In this sense, *NormalizedERDE* provides the possibility to validate the performance of a model over time by using different points of measure and with different datasets.

3.3.5 TaP

Time aware Precision or TaP in short, is a newly defined metric and one of the major contributions of this thesis. It was defined to address the issues remarked for previously early detection metrics, such as $ERDE$ and $F_{latency}$.

Output values will range between $+1$ and -1, where values close to $+1$ represent correct and in time predictions. Values close to -1 represent incorrect or late predictions and values arround 0 would correspond with non predictions portrayed as delays.

TaP is defined as a generic metric that could be applied to any early detection problem. For this purpose, one of the key aspects that makes it problem dependent instead of dataset dependent is how the moment when a correct prediction is considered to be taken late. To achieve this, and following $ERDE$ design, a point denoted by o is defined and used to start applying a penalisation to correct predictions.

Therefore, TaP at point o for an entity e_i is calculated as follows:

$$TaP_{o,\lambda}(e_i, k) = \begin{cases} -1 & \textit{if } FP \vee FN \\ 1 & \textit{if } TP \vee TN \wedge k \leq o \\ 1 - pf_{o,\lambda}(k) & \textit{if } TP \vee TN \wedge k > o \\ 0 & \textit{if delay} \end{cases}$$

In the case of an incorrect prediction (false positive, FP, and false negative, FN), the metric weighs -1 to represent an error. If a correct prediction is made (true positive, TP, and true negative, TN), then the delay required to generate the prediction is taken into account. In this last case, if the prediction was made before the defined point o (i.e. $k \leq o$), then the metric achieves its maximum value (i.e. $+1$). Otherwise (i.e. $k > o$) a penalty function, $pf_{o,\lambda}(k)$, is used to obtain the output by reducing the score of the metric with a predefined penalisation. Finally, if no decision has been made at that point (i.e. *delay*), TaP will take the value of 0.

The output of the metric TaP for a set of entities E is obtain by the aggregation of individual entities e_i by means of an average score, as shown next:

$$TaP_{o,\lambda}(E, k) = \frac{1}{|E|} \sum_{e_i \in E} TaP_{o,\lambda}(e_i, k)$$

The penalty function included in the calculation of $TaP_{o,\lambda}(e_i, k)$ metric as $pf_{o,\lambda}(k)$ is defined using a generalised logistic function scaled to operate on the defined range (i.e. -1 to $+1$) and the penalty is based on the number of items required to take the decision and where the parameter λ controls how the penalty is increased:

$$pf(k)_{o,\lambda} = 2 \cdot \left(-1 + \frac{2}{1 + e^{-\lambda(k-o)}} \right)$$

Figure 3.4 shows how parameter λ affects the calculus of TaP for a specific correct prediction (i.e. $TaP_{o,\lambda}(e_i, k)$). It must be noticed that in all cases, a correct prediction is emitted, but the number of *items* required to make the decision varies as shown on X-axis. In this case, the point of penalisation $o = 2$ has been chosen in order to properly display the output value of the function after and before the penalisation is introduced. Although as a generalisation it could be said that as the value of λ increases the penalisation increments too, particularly for $\lambda = 0$ no penalty is introduced no mater when the prediction was made, the maximum score is always achieved. When $\lambda = 0.1$ the metric decreases linearly as the delay increases. If the value selected is $\lambda = 10$, the full penalisation is applied in a single step and de minimum value possible is reached after just one more item has been considered, that is, when $k = 3$.

Also, and based on the original definition of TaP we describe , respectively, TaP^+ for the set of anomalous entities or positives cases, E^+, and TaP^- for the non-anomalous entities as:

$$TaP^+(E, k) = \frac{1}{|E^+|} \sum_{e_i \in E^+} TaP(e_i, k)$$

$$TaP^-(E, k) = \frac{1}{|E^-|} \sum_{e_i \in E^-} TaP(e_i, k)$$

Both values are combined in a *Time aware Precision* variant denoted as TaP_α and defined as:

$$TaP_\alpha(e_i, k) = \alpha \cdot TaP^+(e_i, k) + (1 - \alpha) \cdot TaP^-(e_i, k)$$

The parameter α controls the impact of anomalous and non anomalous cases in the final score. This value is problem related and will depend on each specific situation, with the restriction that it has to take a value in the range $[0, 1]$ ensuring that way that the added contribution of TaP^+ and TaP^- reaches the full value of TaP. Following this, $\alpha = 0.5$ will balance both cases, but other approaches could be considered, as shown in Chapter

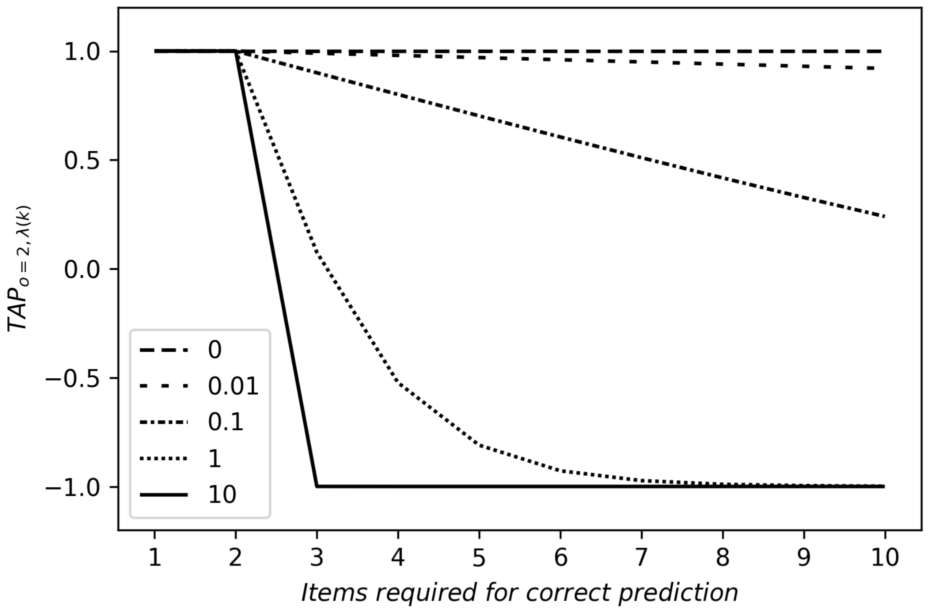


Figure 3.4: TaP for point of measure $o = 2$ and different values of λ. The X-axis represents the number of items required by the system to reach a correct prediction, assuming all previous items led to a delay. For example, for $x = 5$ the system generated the correct prediction on item $k = 5$ and, therefore, previously (i.e. $x < 5$) a delay was generated.

4. As a particular example, when $\alpha = 1$ is used, only positive, or anomalous, cases will be taken into account and the value of TaP_α would be the same as TaP^+. This might be interesting when studying, for instance, the early detection of a disease where non infected cases are non important.

3.3.6 TaF

The metric *Time aware F-score* or TaF for short, has been defined to overcome some limitations of the previous metrics and to ease the interpretation of the results, as it has a behaviour more similar to F-latency but improving the configuration parameters. Reflecting the same idea as TaP, a penalization point and the degree of penalization is defined as problem-based instead of based on the particular values of the dataset. In this sense, it is more natural to define a penalization point that to set it based on the mean of the values of the dataset. This last point is also crucial because in an real world streaming situation, it will not be possible to get the complete image or range or values before one specific item is processed.

The metric has been defined as follows:

$$TaF_{o,\lambda}(e_i, k) = \begin{cases} 1 & \textit{if } TP \wedge k \leq o \\ 1 - pf_{o,\lambda}(k) & \textit{if } TP \wedge k > o \\ 0 & \textit{if delay} \end{cases}$$

$$TaF(E, k) = \frac{\sum_{e_i \in E} TaF(e_i, k)}{|E_{TP}| + \frac{1}{2}(|E_{FP}| + |E_{FN}|)}$$

The cases defined in $TaF_{o,\lambda}$ use the same principle as in the previous metrics ($ERDE$ and TaP) but just for the correctly detected positive cases (true positives, TP). The maximum value is obtained if the item is identified before the point of measure o, otherwise a penalization function named $pf(k)_{o,\lambda}$ is applied.

For the final $TaF(E, k)$ value an aggregation of intermediate results is performed in order to use the final value as a penalized count of true positive cases. That is to use it instead of $|E_{TP}|$, applying, then, the F-score function. The number of real true positive cases (TP) is depicted as $|E_{TP}|$, whereas negative cases are shown as $|E_{FP}|$ and $|E_{FN}|$ for false positives and false negatives respectively.

For this metric, the penalty function is defined as follows, to give values in the range $[1, 0]$ which generates an output in the metric in the same range. This maintain the relation with the output values of F-score.

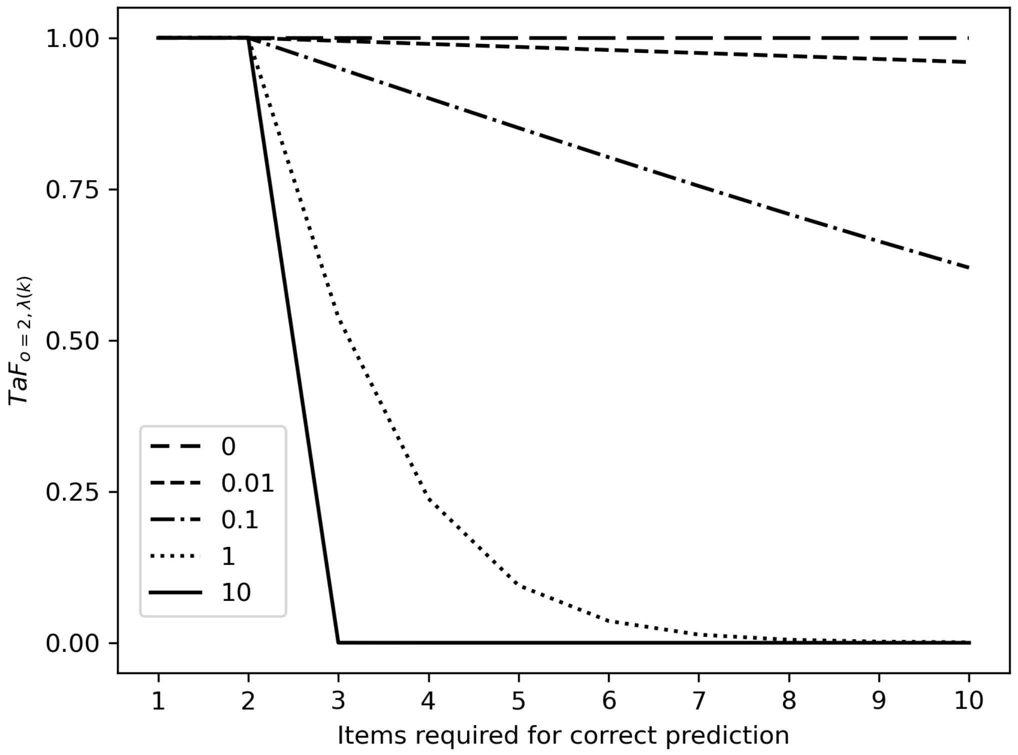


Figure 3.5: TaF for point of measure $o = 2$ and different values of λ. The X-axis represents the number of items required by the system to reach a correct prediction, assuming all previous items led to a delay. For example, for $x = 5$ the system generated the correct prediction on item $k = 5$ and, therefore, previously (i.e. $x < 5$) a delay was generated.

$$pf(k)_{o,\lambda} = -1 + \frac{2}{1 + e^{-\lambda(k-o)}}$$

The values of penalization shown in Figure 3.5 for different values of the parameter λ display how the results for individual entities e_i affects to the final value of TaF. As depicted in Section 3.3.5 for Figure 3.4, in all cases a correct prediction is emitted but the number of *items* required to achieve that prediction varies as presented on X-axis. The only significant difference present between Figure 3.4 and Figure 3.5 is the output range of the function, as TaP is $[-1, +1]$ and TaF is $[0, 1]$.

Chapter 4

Metrics evaluation

In this chapter the experiments conducted for the evaluation of the proposed metrics will be presented. Each one of them led to series of conclusions that were applied in the definition of developed solutions.

4.1 Baseline

4.1.1 Datasets

Two different datasets were used in the evaluation presented in this chapter, both their nature and characteristics differ, in order to study the outcome under particular circumstances. The first one is the OS Scan Attack from Kitsune dataset [34], which presents some particularities and its preprocess will be shown later in this section. The second one was especially collected for the Workshop on Early detection prediction on the Internet 2017 (eRisk) [11]. It contains posts from the website Reddit published by users and tagged as depressed or non-depressed. A deeper analysis on the characteristics and particularities of this dataset is included next.

Kitsune dataset is composed by data traffic from a video surveillance network and includes different attacks performed over the network through several days. It was designed and used to test de Network Intrusion Detection System (NIDS) described in [34]. In particular, the OS Scan Attack, that examine the network for hosts and their operating systems (OS) to reveal potential vulnerabilities, will be used and referred to as "Network Attack" from now on. This dataset is composed by a set of network packets from which a group of engineered features are extracted and tagged as "Attack" or "Normal". In order to further analyse the traffic, it has been divided into bidirectional flows as described in [91, 92], defined as the aggregation of packets with same pair source IP address - destination IP address, source port - destination port and protocol over a defined period of time. To account for the time division, timestamp of the packets was used as a split point using 0.1 seconds as threshold for the time between packets and 1 second as threshold for the flow span [6]. This task involved the redefinition of the assigned tags as now both request and reply packets of the same flow should be considered to have the same label. The reason behind this action is that the reply to a scan attack is delivering information to the attacker, and even if the individual packet would not pose a threat by itself it is, in this case, part of the intrusion. Hence, and following the naming described in 3.1, the Entities (E) corresponds to data flows where each flow is composed by a set of individual packets represented by a group of features and a timestamp that would be described as Items (I_e). For the experiments performed, features described in [34] had been used. As described in the original paper only characteristics of the packet itself and its relation in time with the rest of the traffic are used in the creation of the dataset without including specific information of the flow they belong to.

Table 4.1 summarizes the main statistics for the datasets used in this Chapter, among which OS Scan Attack from Kitsune dataset can be found under the name of Network attacks. The amount of individual packets reaches nearly 1.7 million, distributed in 75,700 bidirectional flows. As the dataset represents a OS scan attack, most of anomalous flows are going to be around 2 packets in size, like it can be seen in the average of packets per flow for attack class, which accounts for the request and the reply from the device under attack. This, results on getting the majority of *Entities* of Anomalous type even if the greater part of *items* or packets belong to the Normal class. Also, figure 4.1 displays the distribution of packets over flows, highlighting the evidence that most of the flows are formed of a few packets, but also that there are relevant groups around sizes of 100 and 200 packets. It must be observed that the Y-axis of the figure uses logarithmic scale in order to show the difference in sizes between the ones with more occurrences and the rest.

The eRisk depression dataset, referred as “Depression Dataset” from now on was specifically gathered for the Workshop on Early detection prediction on the Internet (eRisk) in its edition of 2017 [11]. It is composed by a set of publicly available Reddit posts published by users in about a year of use period. Those posts were later tagged as “Depressed” or “Non-depressed” following self-reports of diagnosed depression. In this case, the subjects correspond with the *Entities* (E) and each of the *items* (I_i^e) with the subject’s posts. As for the features used in the experiments, the ones defined in [93] and [94] wil be used. Although only individual post characteristics will be

Table 4.1: Datasets statistics for metrics evaluation

Dataset	*Entities & items*	*Normal*	*Anomalous*	*Total*
Depression	Entities	752	135	887
	Items	481,837	49,557	531,394
	Items per entity	640.7	367.1	599.1
Network attacks	Entities	10,045	65,655	75,700
	Items	1,566,602	131,249	1,697,851
	Items per entity	155.96	1.99	22.43

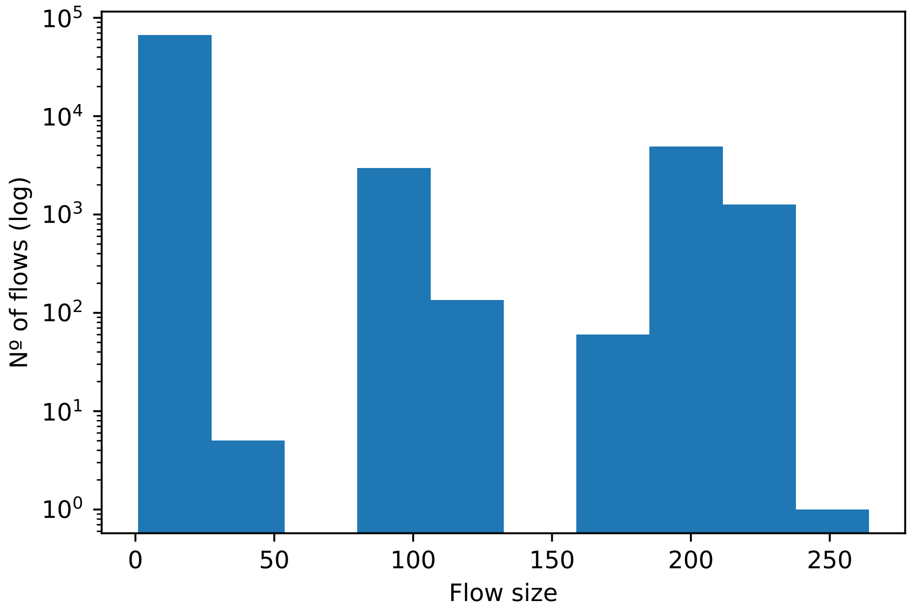


Figure 4.1: Distribution of packets per flow of Networks Attack dataset (log scale).

included and no features created by the aggregation of a sequence of posts are introduced.

Table 4.1 includes, along with the ones of the previous described dataset, the main statistics computed for the subjects and posts considered. This dataset is significantly smaller both in terms of number of *Entities* and *Items* but it has a higher average ratio of *items* per *entity*. With just 887 *Entities* and a little over half a million items, achieves an average or almost 600 items per entity. Being the amount of "Anomalous" cases the 15.22% of the total subjects, it is relevant to display that this ratio reaches almost half the value for "Anomalous" than for "Normal" cases.

4.1.2 Models

For evaluation purposes, several models have been defined to test the outcome of the different metrics under particular situations. Those models could be roughly divided into two groups: synthetic and machine learning. The first of them represents a group of models which based on the ground truth of the labels from the dataset, returns a specific response for all the cases. The other set are state-of-the-art machine learning models, trained with the datasets features and used to evaluate the metrics in a real world environment.

The first set is the one formed by five synthetic models who, based on the ground truth, return specific values for each case in order to model extreme situations difficult to consistently achieve with real detection models. These models are defined as follows:

- $Oracle_n$: produces delays before item n for all entities and then the correct prediction for each entity is generated. Therefore, an $Oracle_1$ would represent a best-case scenario where all correct predictions are produced after processing the first item of each entity. Following that, $Oracle_5$ for example, would represent a model where before item 5 only delays are emitted and after that point the correct case is selected.

- $Elcaro_n$: works as an inverse *Oracle*, delaying the prediction before item n and then providing the wrong prediction for each entity. In this case, any *Elcaro* would represent a worst-case scenario for every item in the entity, independently of the time required to generate the prediction.

- $Positive_n$: a delay is produced before item n as for previous models, and then all entities are tagged as positive (i.e. anomalous) cases.

- $Negative_n$: in this case, after item n, and opposite to $Positive_n$, all entities are predicted as negative (i.e. normal or non-anomalous).

- *Random*: in this case, the three possible outputs (positive, negative or delay) are generated randomly with equal probabilities. There is no need to the point where the decision is taken defining $Random_n$ because for any point the three possible labels have the same probabilities.

The second set includes state-of-the-art machine learning models and are divided into two groups. The first one used for $NormalizedERDE$ evaluation over Network Attack dataset was previously used in a non time-aware intrusion detection by [31]. The following models were selected because of their effectiveness in the detection of botnet activity and their implementation in *Weka* [95] was used:

- *ZeroR*: trivial classifier that assigns the most common class to all instances.
- *OneR*: standard classifier that uses the most relevant feature to predict the class.
- *JRip*: standard implementation of Repeated Incremental Pruning to Produce Error Reduction (RIPPER).
- *PART*: determines the rules based on the best leaf of C4.5 decision trees.
- *J48*: standard implementation of C4.5 decision trees.
- *Random Forest*: standard classifier based on multiple random trees.

The second group, used in the experimental evaluation of the rest of the proposed metrics is composed by some well known of-the-shelf state-of-the-art machine learning algorithms. The selection of these models was made based on the work presented in [84] for detection of cyberbullying in Vine social network. Particularly, the implementation by *scikit-learn* [96] was used, and the description of the models with the parameters used is listed below:

- *LinearSVC:* Support Vector Classification implementation with a 'linear' kernel that improves parameter selection and scalability.
  - Parameters: $C = 1$, $class\_weight$ = '$balanced$', $dual = False$, $max\_iter = 1000$
- *ExtraTree:* Meta estimator that uses averaging of randomized decision trees for classification.

  - Parameters: $n\_estimators = 50$, $bootstrap = False$, $class\_weight = None$

- *AdaBoost:* Meta estimator that refits a classifier by updating weights of incorrectly classified instances.
  - Parameters: $n\_estimators = 1000$, $learning\_rate = 2.0$, $algorithm = `SAMME.R'$

- *Random Forest:* Meta estimator which uses a determined number of decision trees and applies averaging to obtain the classifier.
  - Parameters: $n\_estimators = 500$, $class\_weight = None$, $max\_features = `sqrt'$, $max\_depth = 7$, $bootstrap = False$

- *Logistic Regression:* Conformed by a *logit MaxEnt* classifier, where the maximum entropy classifier is combined with a logistic function.
  - Parameters: $C = 0.1$, $class\_weight = `balanced'$, $dual = False$, $penalty = `l2'$, $solver = `sag'$

4.2 Normalized ERDE

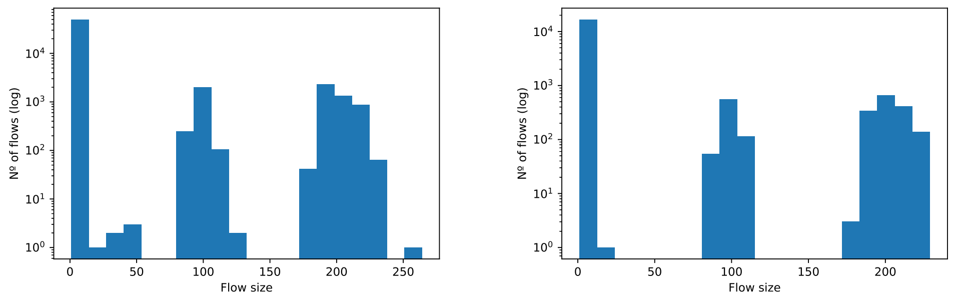
For the evaluation of *Normalized ERDE* metric, the Network Attack Dataset is used combined with the synthetic models and the first set of ML models conformed by: *ZeroR*, *OneR*, *JRip*, *PART*, *J48* and *Random Forest*. In this last case, the dataset has been divided into 75% and 25% for training and testing respectively. This division was performed maintaining the distribution and characteristics of the original dataset, as it can be observed in Table 4.2 and in Figure 4.2.

The first set of experiments explores the behaviour of different metrics for the *Oracle* models under batch evaluation and the results are detailed in Table 4.3. For the described synthetic models points 1 and 2 were chosen for *Oracle* model and point 1 for the rest of models. Although experiments with synthetic models were performed for points from 1 to 20 only relevant results are presented. This conclusion can be extracted from the fact that almost all attack flows (or positive *entities*) are about 2 packets long, any further decision point would not affect the outcome. For the same reason, penalization points for *ERDE* metric were defined as $o = 1$, $o = 2$ and $o = 5$.

First, focusing on the $Oracle_1$ and $Oracle_2$ models, that is, the ones who are able to always make the correct prediction at points 1 and 2 respectively, results of non time aware and time aware metrics significantly vary.

Table 4.2: Dataset (Train/Test)

	Train	Test	Total
Number of packets	1264351	433500	1697851
Number of flows	56775	18925	75700
Percentage of flows	75%	25%	100%
Average packets per flow	22.27	22.9	22.43
Standard deviation packets per flow	57.03	57.92	57.26
Minimum packets per flow	1	1	1
Maximum packets per flow	264	232	264



(a) Train (b) Test

Figure 4.2: Distribution of packets per flow for Train 4.2(a) and Test 4.2(b) (in logarithmic scale).

Table 4.3: Early detection performance for *Oracle* models

Model	Metrics	Batches									
		1	2	3	4	5	6	7	8	9	10
$Oracle_1$	ERDE (o=1)	0.0	0.0	0.0	0.0002	0.4331	0.4331	0.4331	0.4332	0.4332	0.6338
	ERDE (o=2)	0.0	0.0	0.0	0.0001	0.2329	0.2329	0.2330	0.2330	0.2330	0.4335
	ERDE (o=5)	0.0	0.0	0.0	0.0	0.0156	0.0156	0.0156	0.0156	0.0156	0.0411
	F1-latency	0.0	0.0	0.0	-0.0004	0.0	0.0	0.0	0.0	0.0	0.5
	Precision	0.0	0.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0
	Recall	0.0	0.0	0.0	0.0004	0.9987	0.9987	0.9987	0.9987	0.9987	1.0
	F1	0.0	0.0	0.0001	0.0008	0.9993	0.9993	0.9993	0.9993	0.9993	1.0
$Oracle_2$	ERDE (o=1)	0.0	0.0	0.0	0.0	0.0	0.0	0.0002	0.0002	0.0002	0.6338
	ERDE (o=2)	0.0	0.0	0.0	0.0	0.0	0.0	0.0002	0.0002	0.0002	0.4335
	ERDE (o=5)	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0411
	F1-latency	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.5
	Precision	0.0	0.0	0.0	0.0	1.0	1.0	1.0	1.0	1.0	1.0
	Recall	0.0	0.0	0.0	0.0	0.0	0.0	0.0004	0.0004	0.0004	1.0
	F1	0.0	0.0	0.0	0.0	0.0001	0.0001	0.0008	0.0008	0.0008	1.0
$Elcaro_1$	ERDE (o=1)	0.1068	0.1068	0.1068	0.1075	0.9733	0.9733	0.9733	0.9733	0.9733	0.9824
	ERDE (o=2)	0.1068	0.1068	0.1068	0.1075	0.9733	0.9733	0.9733	0.9733	0.9733	0.9824
	ERDE (o=5)	0.1068	0.1068	0.1068	0.1075	0.9733	0.9733	0.9733	0.9733	0.9733	0.9824
	F1-latency	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	Precision	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	Recall	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	F1	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
$Positive_1$	ERDE (o=1)	0.1068	0.1068	0.1068	0.1073	0.5402	0.5402	0.5403	0.5403	0.5403	0.7489
	ERDE (o=2)	0.1068	0.1068	0.1068	0.1072	0.3401	0.3401	0.3402	0.3402	0.3402	0.5486
	ERDE (o=5)	0.1068	0.1068	0.1068	0.1071	0.1227	0.1227	0.1227	0.1227	0.1227	0.1562
	F1-latency	0.0	0.0	0.0	-0.0003	0.0	0.0	0.0	0.0	0.0	0.4645
	Precision	0.0	0.0	0.0002	0.0027	0.8752	0.8752	0.8752	0.8752	0.8752	0.8673
	Recall	0.0	0.0	0.0	0.0004	0.9987	0.9987	0.9987	0.9987	0.9987	1.0
	F1	0.0	0.0	0.0001	0.0007	0.9328	0.9328	0.9328	0.9328	0.9328	0.9289
$Negative_1$	ERDE (o=1)	0.0	0.0	0.0	0.0003	0.8661	0.8661	0.8661	0.8661	0.8661	0.8673
	ERDE (o=2)	0.0	0.0	0.0	0.0003	0.8661	0.8661	0.8661	0.8661	0.8661	0.8673
	ERDE (o=5)	0.0	0.0	0.0	0.0003	0.8661	0.8661	0.8661	0.8661	0.8661	0.8673
	F1-latency	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	Precision	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	Recall	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	F1	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

Traditional metrics, represented in this case for $Precison$, $Recall$ and $F1$, consistently achieve the perfect score in the last batch. These metrics start increasing its value as soon as the $Oracle$ starts making correct predictions, that is on batch 3 for $Oracle_1$ and on batch 5 for $Oracle_2$. That is mainly due to the size of attack flows, as very few of them have 4 or more packets, and the lack of penalization for late correct decisions in this kind of metrics. Time aware metrics, on the other hand, behave as expected not achieving the better score at the last batch as a result of the penalisation introduced for late predictions. $ERDE$ for example, increases its value as the number of evaluated batches increase and that can be explained by the fact that even if the entity was properly predicted, there is a latency that introduces a penalty in the result. That penalty is applied at different points depending on the value of the parameter o for the metric, and in this case, values 1, 2 and 5 were studied. The higher the point of penalty the lower the value of $ERDE$ which means a better result for this metric. $F_{latency}$ on its side, shows a less dynamic behaviour maintaining a close to 0 value until the last batch where it reaches a value of 0.5.

Following, the reverse $Oracle$, $Elcaro_1$, achieves the worst results for all metrics as all the predictions are incorrect no mater the point where they are taken. As in the previous case with the $Oracle$ models, non time aware metrics do not provide any information about the delay in the decision and display a 0.0 value for all the batches. These values are obtained as a result of the combination of not taking into account delay information and all the predictions getting the opposite value to the one expected. Also, and as a result of the outcome obtained for $F1$ metric, $F_{latency}$ is 0.0 for all the batches because independently of the value of the penalty applied, if $F1$ is 0.0 its product is going to be the same. Opposite to this is the behaviour shown by $ERDE$ as an evolution on the outcome is shown for different batches as the number of packets processed increases. In this case and as all predictions are wrong it has the same value independently of the point of penalization ($o = 1$, $o = 2$ or $o = 5$).

Next, $Positive_1$ model achieves a perfect $Recall$ and fairly good results for $Precision$ and $F1$ on the last batch, increasing their values as the number of batches advances. However, time-aware metrics show a decreasing performance opposite to the other metrics growth. It must be pointed out the difference between the results of $ERDE_{o=1}$ which highlights a worse score than $ERDE_{o=2}$ and $ERDE_{o=5}$ as it is far more restrictive when the penalization is applied.

Finally, the last of the proposed synthetic models, $Negative_1$, emulates the results of $Elcaro_1$ achieving the second worst score for all the results in the synthetic models. The first batches obtain low values for $ERDE$ as there is no penalization since flows correspond to normal traffic, until when the batch where the decision is emitted is reached and the model starts to generate false negative predictions. Also, as in $Elcaro_1$ results for the three values of o considered obtain the same values as no positive penalizations are involved because of the nature of this model.

As a result of the analysis of this first set of experiments, there is enough evidence to confirm the unsuitability of non time aware metrics such as *Precision*, *Recall* and $F1$ for time aware evaluation. Concerning $F1_{latency}$, the lack of evolution, severe penalization and unintuitive selection of parameter configuration could limit its use, implying also some problems when working with small number of items.

Regarding $ERDE$, the output values of the metric increase, as expected, as the error should rise when the time required to make a correct prediction increases. Also, something to notice is that $ERDE$ requires calculations that depend on the dataset used for the evaluation when computing the penalty for false negatives, therefore it is not dataset agnostic and it does not depend just on the o parameter which could be defined as problem dependent. Finally, in relation with the values obtained for the synthetic models for this metric, no perfect scores are achieved in the worst of cases, being $Elcaro_1$ with a value of 0.9824 the closest one.

NormalizedERDE, the metric proposed in Section 3.3.4, ranges from 0 to 1 being, in the same sense as *ERDE*, the values closer to 0 the better performance and values closer to 1 poor results. In the case of *ERDE*, the minimum is achieved by the *Oracle* model in the first batch while the maximum value corresponds to the *Elcaro* model in the last batch. Although these values should correspond with 0 and 1, as it can be seen in Table 4.3 that is not true for all the model and cases. The best example for this situation is $Elcaro_1$ on batch 10 for whatever point o of measure for $ERDE$ metric. To deal with this, *NormalizedERDE* applies *MinMax normalization* over *ERDE* results.

In the next set of experiments the performance of the different oracle alternatives (i.e. *Oracle*, *Elcaro*, *Negative* and *Positive*) was evaluated over *NormalizedERDE* metric, the results are shown in Figures 4.3, 4.4 and 4.5. As it can be observed, the figures show the expected behaviour for $ERDE$ metric but with slight differences. In general a difference between models where positive cases are correctly classified (*Oracle* and *Positive*) and the ones where negative cases are correctly classified (*Elcaro* and *Negative*) can be seen in all three figures, being that difference higher when the penalisation point (o) is bigger (i.e. $o = 5$) than with an earlier penalization point. Results for *Oracle* and *Positive* are located on the lower part of the graph due to because in those cases all positive cases are correctly classified. *Elcaro* and

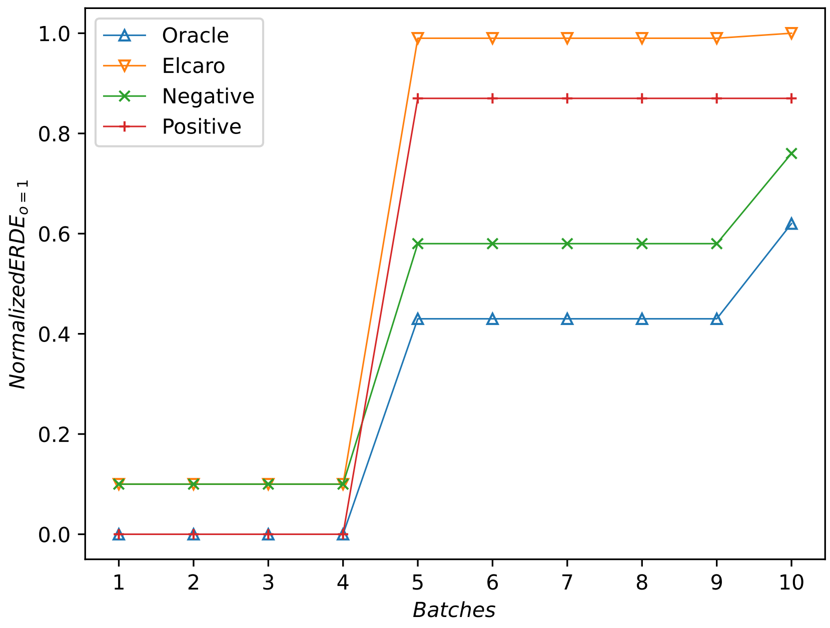


Figure 4.3: *Normalized ERDE* results at $o = 1$ for synthetic models.

Negative, on the other hand, achieve worst results and are located in the upper part of the graphs.

Taking $NormalizedERDE_{o=1}$ (Figure 4.3) as a reference, a step is present in batch 5 and generalized for the other values of o that can be explained by the equation definition of *ERDE* metric and its behaviour with low values of o and k. From the definition of *ERDE* on Section 3.3.2 it can be observed that even in the case of a true positive there is a penalisation that causes a decrease in performance of around 0.5 and which is reflected in *NormalizedERDE* behaviour. This same situation is present for the values of $NormalizedERDE_{o=2}$ (Figure 4.4) but here the differences between positive and negative models are bigger, something that can also be explained by value of o and the definition of the metric. For $NormalizedERDE_{o=5}$ (Figure 4.5) it is noticeable that the penalisation is softened for *Oracle* and *Positive* models which introduce a more interesting behaviour for the metric in these cases. Although *Elcaro* and *Negative* maintain their values leading to a low performance. Finally, *Oracle* model shows a slightly better performance than *Positive* as all the cases are properly classified and in *Positive* model negative cases are classified as positive too. For the same reason, *Elcaro* provides a worst performance than *Negative* as even the negative cases are incorrectly classified.

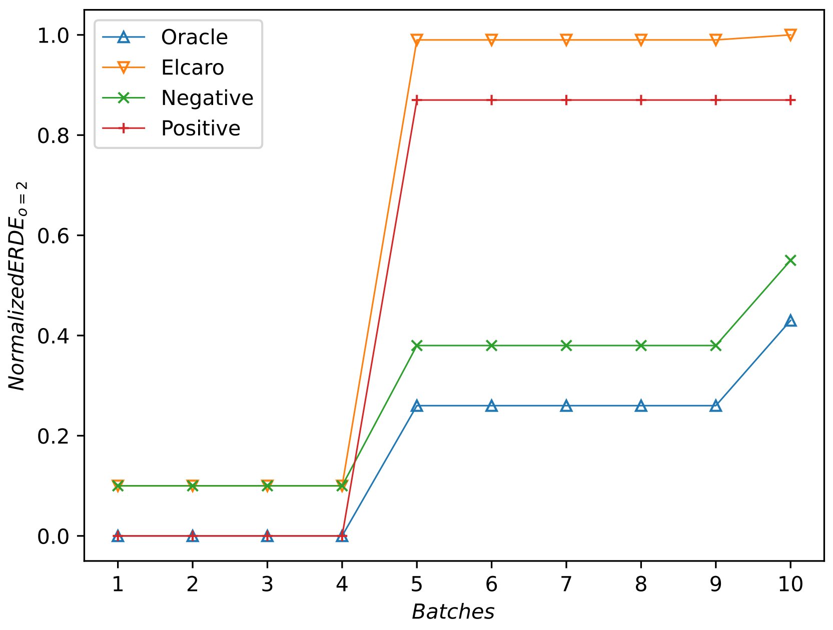


Figure 4.4: *Normalized ERDE* results at $o = 2$ for synthetic models.

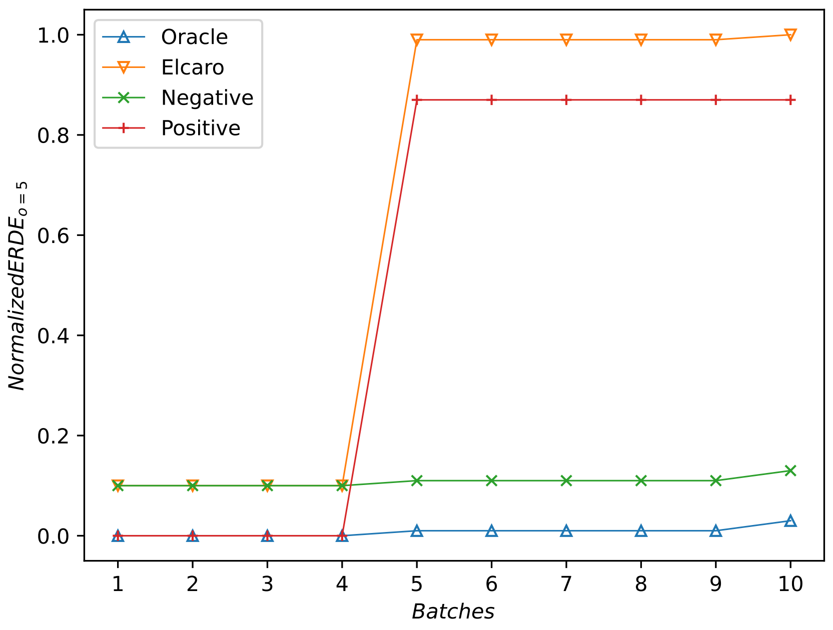


Figure 4.5: *Normalized ERDE* results at $o = 5$ for synthetic models.

For these reasons, $NormalizedERDE_{o=5}$ provides a better evaluation metric for the assessment of time aware intrusion detection systems. It provides a measure where a slight penalisation if present but not to steep that turns a late prediction into a wrong one in about just one batch. One feature provided by *NormalizedERDE* is the ability to measure performance of detection systems over time through different batches. Not only that, but the ability to penalise the results when they are taken past a defined point. As it can be observed from the presented figures and the statistics for the dataset (Table 4.1), the values of the metrics represent the characteristics of the attack. Most anomalous flows consist only in two packets, due to the scan nature of the attack, corresponding with the request from the intruder and the reply given by the system. The fact that each batch contains 10% of the total of packets for each flow makes that on batch 5 attack flows contain on average 1 packet. Also, as on the last batch all decisions must be made, there is another step present.

The next set of experiments are focused on evaluating the performance over some state-of-the-art models, in order to provide a better real world assessment of the metrics. The set of algorithms was selected from previous work [31] to evaluate classification of IoT network traffic to detect Botnet Attacks. Even though the main objective of these experiments are the evaluation of *NormalizedERDE* for the sake of completeness and to compare with other metrics results the values or *ERDE* and $F_{latency}$ are presented as well. From the Tables 4.4 and 4.5, it can be observed from the results of *ZeroR*, *OneR*, *RandomForest*, *J48*, *JRip* and *PART*, that this last model obtains the worst performance of the lot, confirming the results presented in [31]. Particularly, $NormalizedERDE_{o=5}$ shows that there is room for improvement for all models, since the best value obtained is 0.2846 for *Random Forest*. This is far from the best possible value obtained with the synthetic models where $Oracle_1$ obtained 0.0418 (Figure 4.5). This indicates that standard machine learning models have a low performance for early detection tasks as they might not be able to model these conditions. In order to improve these results, more alternatives for specific early detection models must be studied and proposed in order to improve results with time-aware metrics, as it will be approached in Chapter 5.

Noteworthy is also the low performance for all models in the evaluation of the first batches. A further analysis of those cases showed that it might be due to the models providing early detection of attacks when dealing with normal data flows. This aspect could be further improved if time-aware machine learning models were applied to try to decrease the impact of wrongly performed early detections.

Results shown in this section highlight the importance of time-aware met-

Table 4.4: Early detection performance for state-of-the-art ML models, part 1

Classifier	Metrics	Batches									
		1	2	3	4	5	6	7	8	9	10
ZeroR	$ERDE_{o=1}$	0.1201	0.1201	0.1201	0.1207	0.605	0.605	0.605	0.6051	0.6051	0.7832
	$ERDE_{o=2}$	0.1201	0.1201	0.1201	0.1206	0.435	0.435	0.435	0.4351	0.4351	0.613
	$ERDE_{o=5}$	0.1201	0.1201	0.1201	0.1205	0.2503	0.2503	0.2503	0.2503	0.2503	0.2796
	$F_{latency}$	0.0	0.0	-0.0001	-0.0004	0.0	0.0	0.0	0.0	0.0	0.4243
	$NormalizedERDE_{o=1}$	0.0	0.0	0.0	0.0008	0.6509	0.6509	0.6509	0.651	0.651	0.8901
	$NormalizedERDE_{o=2}$	0.0	0.0	0.0	0.0007	0.4227	0.4227	0.4227	0.4228	0.4228	0.6616
	$NormalizedERDE_{o=5}$	0.0	0.0	0.0	0.0005	0.1748	0.1748	0.1748	0.1748	0.1748	0.2141
OneR	$ERDE_{o=1}$	0.1201	0.1201	0.1201	0.1207	0.605	0.605	0.6051	0.6051	0.6051	0.7833
	$ERDE_{o=2}$	0.1201	0.1201	0.1201	0.1207	0.435	0.435	0.4351	0.4351	0.4351	0.613
	$ERDE_{o=5}$	0.1201	0.1201	0.1201	0.1206	0.2504	0.2504	0.2504	0.2504	0.2504	0.2797
	$F_{latency}$	0.0	0.0	-0.0001	-0.0004	0.0	0.0	0.0	0.0	0.0	0.4243
	$NormalizedERDE_{o=1}$	0.0	0.0	0.0	0.0008	0.6509	0.6509	0.651	0.651	0.651	0.8902
	$NormalizedERDE_{o=2}$	0.0	0.0	0.0	0.0008	0.4227	0.4227	0.4228	0.4228	0.4228	0.6616
	$NormalizedERDE_{o=5}$	0.0	0.0	0.0	0.0007	0.1749	0.1749	0.1749	0.1749	0.1749	0.2142
RandomForest	$ERDE_{o=1}$	0.1201	0.1201	0.1201	0.1207	0.605	0.605	0.605	0.6051	0.6051	0.7832
	$ERDE_{o=2}$	0.1201	0.1201	0.1201	0.1206	0.435	0.435	0.435	0.4351	0.4351	0.613
	$ERDE_{o=5}$	0.1201	0.1201	0.1201	0.1205	0.2503	0.2503	0.2503	0.2503	0.2503	0.2796
	$F_{latency}$	0.0	0.0	-0.0001	-0.0004	0.0	0.0	0.0	0.0	0.0	0.4243
	$NormalizedERDE_{o=1}$	0.1223	0.1223	0.1223	0.1229	0.6158	0.6158	0.6158	0.6159	0.6159	0.7972
	$NormalizedERDE_{o=2}$	0.1223	0.1223	0.1223	0.1228	0.4428	0.4428	0.4428	0.4429	0.4429	0.6240
	$NormalizedERDE_{o=5}$	0.1223	0.1223	0.1223	0.1227	0.2548	0.2548	0.2548	0.2548	0.2548	0.2846

Table 4.5: Early detection performance for state-of-the-art ML models, part 2

Classifier	Metrics	Batches									
		1	2	3	4	5	6	7	8	9	10
J48	$ERDE_{o=1}$	0.1201	0.1201	0.1201	0.1207	0.605	0.605	0.605	0.6051	0.6051	0.7833
	$ERDE_{o=2}$	0.1201	0.1201	0.1201	0.1206	0.435	0.435	0.435	0.4351	0.4351	0.613
	$ERDE_{o=5}$	0.1201	0.1201	0.1201	0.1205	0.2503	0.2503	0.2503	0.2503	0.2503	0.2797
	$F_{latency}$	0.0	0.0	-0.0001	-0.0004	0.0	0.0	0.0	0.0	0.0	0.4243
	$NormalizedERDE_{o=1}$	0.1223	0.1223	0.1223	0.1229	0.6158	0.6158	0.6158	0.6159	0.6159	0.7973
	$NormalizedERDE_{o=2}$	0.1223	0.1223	0.1223	0.1228	0.4428	0.4428	0.4428	0.4429	0.4429	0.6240
	$NormalizedERDE_{o=5}$	0.1223	0.1223	0.1223	0.1227	0.2548	0.2548	0.2548	0.2548	0.2548	0.2847
JRip	$ERDE_{o=1}$	0.1201	0.1201	0.1201	0.1207	0.605	0.605	0.605	0.6051	0.6051	0.7833
	$ERDE_{o=2}$	0.1201	0.1201	0.1201	0.1206	0.435	0.435	0.435	0.4351	0.4351	0.613
	$ERDE_{o=5}$	0.1201	0.1201	0.1201	0.1205	0.2503	0.2503	0.2503	0.2503	0.2503	0.2797
	$F_{latency}$	0.0	0.0	-0.0001	-0.0004	0.0	0.0	0.0	0.0	0.0	0.4243
	$NormalizedERDE_{o=1}$	0.1223	0.1223	0.1223	0.1229	0.6158	0.6158	0.6158	0.6159	0.6159	0.7973
	$NormalizedERDE_{o=2}$	0.1223	0.1223	0.1223	0.1228	0.4428	0.4428	0.4428	0.4429	0.4429	0.6240
	$NormalizedERDE_{o=5}$	0.1223	0.1223	0.1223	0.1227	0.2548	0.2548	0.2548	0.2548	0.2548	0.2847
PART	$ERDE_{o=1}$	0.1201	0.1201	0.1202	0.1208	0.8562	0.8562	0.8562	0.8562	0.8562	0.8651
	$ERDE_{o=2}$	0.1201	0.1201	0.1202	0.1208	0.8562	0.8562	0.8562	0.8562	0.8562	0.8651
	$ERDE_{o=5}$	0.1201	0.1201	0.1202	0.1208	0.8562	0.8562	0.8562	0.8562	0.8562	0.8651
	$F_{latency}$	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	$NormalizedERDE_{o=1}$	0.1223	0.1223	0.1224	0.123	0.8715	0.8715	0.8715	0.8715	0.8715	0.8806
	$NormalizedERDE_{o=2}$	0.1223	0.1223	0.1224	0.123	0.8715	0.8715	0.8715	0.8715	0.8715	0.8806
	$NormalizedERDE_{o=5}$	0.1223	0.1223	0.1224	0.123	0.8715	0.8715	0.8715	0.8715	0.8715	0.8806

rics for a proper model evaluation. Particularly, *NormalizedERDE* as a modification of *ERDE* metric provides a suitable manner to assess early intrusion detection systems. As shown, batch evaluation with $NormalizedERDE_{o=5}$ could improve models evaluation from the point off view of time evaluation. It is important also to emphasize the importance of providing intermediate results for different batches and not just a final value. On this matter, a final and global score could lead to biased conclusions so a more dynamic and time-aware analysis of the model should be performed.

4.3 TaP

In order to improve metrics evaluation, a dual approach was taken both in terms of methods of evaluation and in phases of analysis. For the former, batch and streaming evaluation has been performed to study differences in behaviour between them, providing as explained in Section 3.2 a more realistic representation of systems operation. For the latter, parameter definition for the metric was made on a first step where different values were studied for the synthetic metric. Those values were later used for evaluation with the results of machine learning models. Next follows the analysis of parameters and results for Batch and Streaming evaluation for TaP metric defined in Section 3.3.

4.3.1 Batch

First, the analysis of λ parameter for the metric will be performed with Depression dataset. As shown in Figure 3.4 the following values of λ are considered: 0, 0.01, 0.1, 1 and 10. Values $\lambda = 0.01$ and $\lambda = 10$ were chosen as representative for the effect of this parameter and the results for $TaP_{0=5}$ can be observed in Figure 4.6. The reason behind the selection of these two λ values is to allow both a smooth but noticeable penalty, for 0.01, and a severe one, for 10. For both selected values, and as shown earlier, *Oracle* and *Elcaro* models should take the higher and lower values. Although, the maximum value for *Oracle* and minimum for *Elcaro* is not reached in both cases due to the penalisation introduced by the λ parameter. When $\lambda = 10$, the penalisation introduced is high, therefore, as some entities require more than 5 items to make the correct prediction due to the decision point and the batch evaluation, TaP is unable to reach its perfect score. On the other hand, when $\lambda = 0.01$ the penalisation is much smother which leads to TaP reaching almost a perfect score. When analysing the penalisation introduced by these two values, it is interesting to note that the differences between *Positive*

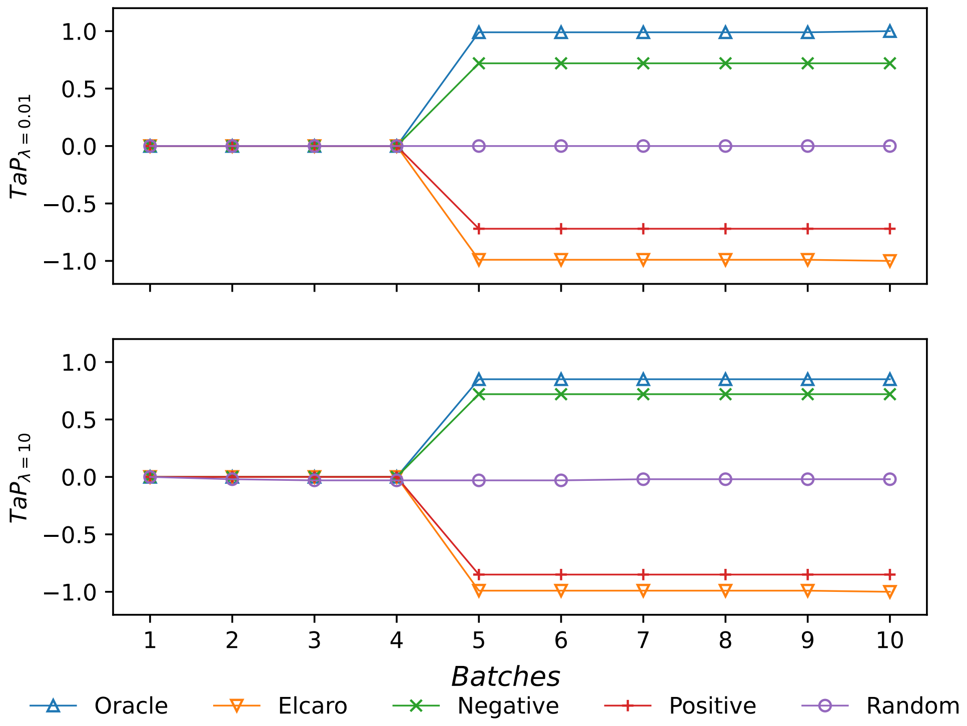


Figure 4.6: $TaP_{o=5}$ for Depression dataset with $\lambda = 0.01$ in the upper figure and $\lambda = 10$ in the lower figure. $Oracle_5$, $Elcaro_5$, $Positive_5$, $Negative_5$ and $Random$ models are represented in each figure.

and $Oracle$, and $Negative$ and $Elcaro$ are smaller for $\lambda = 10$ because the penalisation for a late prediction is equivalent to a wrong prediction. On the contrary, smaller values of λ decrease the penalisation over late predictions which modify the behaviour in case of late correct positive predictions.

The next parameter to be studied is α, defined to combine TaP^+ and TaP^- in order to obtain the final TaP_α metric. For this purpose, $\lambda = 0.01$ will be used due to the slight but noticeable penalty introduced as discussed previously. Both TaP^+ and TaP^- behave independently regarding its definition over positive and negative entities (i.e. E^+ and E^-) as it can be observed in Figure 4.7. In this Figure it can be noticed how TaP^+ focuses only on positive cases, for this reason, $Oracle_5$ and $Positive_5$ achieve equally highest scores. TaP^- obtains the same results but for negative cases, therefore, $Oracle_5$ and $Negative_5$ reach the highest scores. It is interesting to see how TaP^+ achieves a -1 score both for $Negative_5$ and $Elcaro_5$ and in the case of TaP^- this value is achieved for $Positive_5$ and $Elcaro_5$. Meanwhile, 1 is not reached by any model due to the penalisation introduced by the

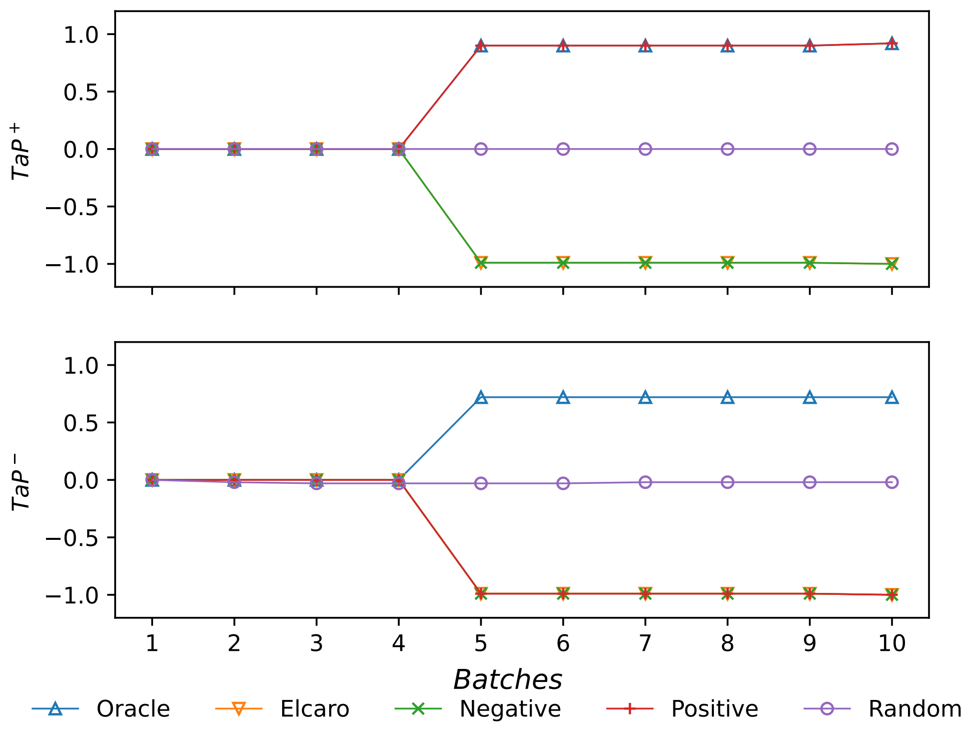


Figure 4.7: $TaP^{+}_{o=5}$ in the upper section of the figure and $TaP^{-}_{o=5}$ in the lower section for Depression dataset with $\lambda = 0.01$. $Oracle_5$, $Elcaro_5$, $Positive_5$, $Negative_5$ and $Random$ models are represented in each figure.

number of items processed and the chosen value of λ. Finally, in order to study the effect of α for TaP_α metric, different values of the parameter are used with $TaP_{o=5}$ for all baseline models. Specifically in Figure 4.8, where the behaviour of the metric with values of 0.5, 0.75, 0.9 and 0.95 is shown, it can be observed that the variation on the parameter has no effect on $Oracle_5$ and $Elcaro_5$ because the former makes no wrong predictions and the latter predicts all positive cases incorrectly. However, in $Positive_5$ and $Negative_5$ the effect is more clearly seen as models display different values depending on the parameter. When $alpha$ increases the difference between $Positive_5$ and $Oracle_5$, and $Negative_5$ an $Elcaro_5$ become smaller as more importance is given to only positive predictions which decrease the difference between these models.

For the remaining experiments, except stated otherwise, previously defined parameters for TaP metric will be fixed as: $\lambda = 0.01$ and $alpha = 0.90$.

Next, an evaluation based on these parameters will be performed both for synthetic models and for the selected state - of - the - art ML models.

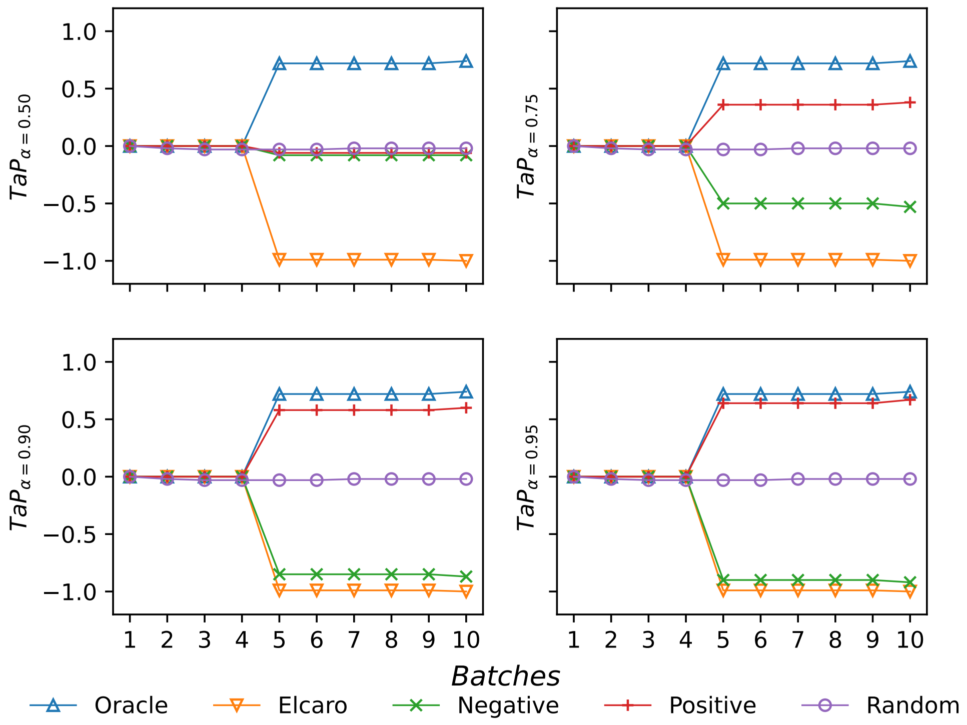


Figure 4.8: TaP_α for Depression dataset with $\alpha = 0.5$ (up left), $\alpha = 0.75$ (up right), $\alpha = 0.9$ (down left) and $\alpha = 0.95$ (down right). Parameters o and λ are fixed at 5 and 0.01, respectively. $Oracle_5$, $Elcaro_5$, $Positive_5$, $Negative_5$ and $Random$ models are represented on each figure.

A comparison with other presented time aware metrics will be performed as well as an analysis of the results over different values of o for the two selected datasets.

First, the performance of $ERDE$, $F_{latency}$ and TaP for the baseline models is presented over the two selected datasets introduced in Section 4.1.1. For each synthetic model two outputs were defined at items 1 and 5 obtaining for example with $Oracle$: $Oracle_1$ and $Oracle_5$. Also, two points of penalization (i.e. o parameter) were defined both for $ERDE$ and TaP and the values selected depended on the dataset. For the Depression dataset $o = 5$ and $o = 50$ were chosen following [11] and for the Network Attacks dataset $o = 1$ and $o = 10$ as presented in [6]. Results for these experiments are summarized in Table 4.6.

Regarding $ERDE$ results $Oracle_1$ and $Oracle_5$ were expected to reach almost perfect scores for all cases, specially when measured at the higher point of penalization. Although this is true for Depression dataset at $ERDE_{o=50}$, reaching even 0.0, in the case of Networks attacks, both $Oracle_1$ and $Oracle_5$ obtain much worst values, with scores up to 0.4337 and 0.6338. These results are justified by the number of items per entity for anomalous cases. In this dataset and as shown in Table 4.1 anomalous flows contain approximately 2 packets per flow, which generates a high penalization even in the case of correct predictions. All of this leads to difficulty of interpretation and comparability issues among datasets as shown with *NormalizedERDE* analysis.

In the case of $F_{latency}$ metric, the model $Oracle_1$ reaches 1.0, the maximum value, but $Oracle_5$ presents a much lower result. In those cases both models achieve almost perfect $F1$ score and the penalisation is introduced by the latency factor of $F_{latency}$. The harsh penalisation introduced by the metric, specially in the case of Network attack dataset, makes that even when the number of items k is relatively small such as around 2 items, the results of the metric decrease around a 50% from $Oracle_1$ to $Oracle_5$. Further, when studying this metric it must be noticed that it is a final metric and no multiple point results could be provided as for $ERDE$ and TaP. The penalty starting point in this case was introduced through the use of the parameter p which is defined based on the median of the items per entity on the dataset. Therefore, $F_{latency}$ is a dataset-dependent metric without point o penalization (i.e. o) parameter.

TaP metric, as defined, is able to provide the maximum score (i.e. 1) for the $Oracle$ models and the lowest score (i.e. -1) in the case of $Elcaro$ model for all datasets. These results are comparable among datasets as similar models produce the similar scores on different datasets. In some cases, for example $Oracle_1$ in the Depression dataset with $o = 5$ which corresponds

Table 4.6: Performance for baseline models on batch evaluation. TaP and TaF use $\lambda = 0.01$ and $\alpha = 0.90$. Parameter o is set to two values, $(low, high)$, for $ERDE$ and TaP in each dataset: $(5, 50)$ for Depression and $(1, 10)$ for Network Attacks.

Dataset	Model	$ERDE_{o=low}$	$ERDE_{o=high}$	$F_{latency}$	$TaP_{o=low}$	$TaP_{o=high}$	$TaF_{o=low}$	$TaF_{o=high}$
Depression	$Oracle_1$	0.047	0.0	1.0	0.9849	1.0	0.9935	1.0
	$Oracle_5$	0.1002	0.0	0.9227	0.9814	1.0	0.9917	1.0
	$Elcaro_1$	0.2812	0.2812	0.0	-1.0	-1.0	0.0	0.0
	$Positive_1$	0.176	0.129	0.2642	0.7883	0.8	0.2625	0.2642
	$Negative_1$	0.1522	0.1522	0.0	-0.8033	-0.8	0.0	0.0
	$Random_1$	0.1699	0.1361	0.2376	0.0212	0.0413	0.2376	0.2431
Network Attacks	$Oracle_1$	0.4337	0.0001	1.0	0.9858	0.9939	1.0	1.0
	$Oracle_5$	0.6338	0.0003	0.5002	0.9768	0.9939	0.995	1.0
	$Elcaro_1$	0.9824	0.9824	0.0	-1.0	-1.0	0.0	0.0
	$Positive_1$	0.5487	0.1152	0.9289	0.8	0.8	0.9289	0.9289
	$Negative_1$	0.8673	0.8673	0.0	-0.8142	-0.8061	0.0	0.0
	$Random_1$	0.7088	0.4921	0.6353	-0.0091	-0.0083	0.636	0.6345

with $TaP_{o=low}$ the maximum score is not obtained. This is because of the batch evaluation that might include more than 5 items which leads to a penalisation, as the last item of the batch is the one used for the metric evaluation. For Network Attack dataset this circumstance is more visible as both case of $TaP_{o=low}$ and $TaP_{o=high}$ are affected by this behaviour. The performance shown between different values of o with different models help to emphasize the differences between de models and for example in the case of Depression dataset a slightly worse performance for $Oracle_5$ is highlighted due to the score obtained.

Next, an analysis of the results for the standard set of state of the art machine learning models selected is included using the same two datasets, metrics and parameters. The results for these experiments can be found in Table 4.7, where is shown that for each machine learning model, once the training phase has been performed, two points of prediction were defined on items 1 and 5, where the model takes the decision with the information gathered until that moment. Resulting in two models per each machine learning model named as follows: $Model_1$ and $Model_5$ (e.g. $AdaBoost_1$ and $AdaBoost_5$). When studying the synthetic models, the best results were expected to be obtained when a higher point of penalty (i.e. $o = high$) were used as TaP parameter. This would indicate that a lower penalty is applied in the evaluation and the same results are found on the evaluation of machine learning models too. Also, as shown in the synthetic models, best results were expected the sooner the decision was taken, but in the case of machine learning models might not operate the same way. This is is due to the case of some models performing better with less information than others so in some cases it could be possible to achieve worst results with more information. Related to that is the fact that for Network attacks dataset, all models performance improves the sooner the decision is taken, but in the case of Depression dataset, both *AdaBoost* and *LogisticRegression* improve their results in terms of $TaP_{o=low}$. Changes observed between the two datasets display how with the previously selected parameters for the metrics there is a difference between scores for $o = low$ and $o = high$ as TaP obtains a lightly bigger difference than $ERDE$ when evaluating bat performance systems. If the system displays a better performance, higher differences could be found for $ERDE$ but as changes are small it could difficult the selection of the better model. This can be spotted for Network attack dataset in the case of $ExtraTree_1$ and $AdaBoost_1$ where $ERDE$ metric achieve 0.4337 for both models while other metrics display different scores. Particularly, TaP obtains 0.9858 and 0.9857, improving the results in terms of definition. Lastly, the behaviour shown by $F_{latency}$ in relation to the values of TaP metric when results from both baseline and machine learning models are studied must be noted, for example, the penalisation introduced for the second model in the case of Network attack dataset.

4.3.2 Streaming

In streaming evaluation, instead of batches, individual items are processed for each entity and evaluated sequentially to obtain the score for the metric at that point, as described in Section 3.2. Since the gap between two consecutive measures will be smaller in this case than for batch evaluation, the parameter λ has been redefined to take the value $\lambda = 0.1$ instead of the $\lambda = 0.01$ used for batch evaluation. In batch evaluation the difference between measures were typically of a few items at least, but in this case the granularity can be reduced to a single item.

First, the results for the baseline models in the Depression dataset are shown in Figure 4.9, where X-axis represent the amount of individual items used by the baseline model to make a prediction and the Y-axis represent the score for each of the metrics. As a result, at each point a different model is presented defined by the amount of items, for example for *Oracle* model, at $x = 10$ point in the figure, would act as a $Oracle_{10}$ and its TaP value is about 0.5. Regarding TaP metric on the same figure it can be observed how after the penalisation point $o = 5$ the performance decreases as the penalty contribute less to the final value. This is the expected behaviour as the predictions, although correct, are generated late. $ERDE$ values display little difference between models only being able to differentiate almost only two groups of models, which makes more complex the evaluation task. Lastly on this same Figure, $F_{latency}$ shows a very good result, close to a perfect score, for *Oracle* which stabilises around 20 items. *Oracle* model shows a big difference to the rest of the models that fall together into two groups that makes hard their evaluation.

For TaP metric, a more thorough assessment of its parameters for streaming evaluation can be see in Figure 4.10, setting $\lambda = 10$ for a coarse penalisation that highlights both point of decision o and selected model. Particularly $TaP_{o=5}$ and $TaP_{o=50}$ are presented for *Oracle*, *Elcaro*, *Positive*, *Negative* and *Random* models in the Depression dataset. As the penalization point o changes, the decrease on the performance moves from between measures in items 5 to 10 in the case of $o = 5$ to between items 50 and 100 for $o = 50$. The bigger penalisation in both cases is the one suffered by *Oracle* and *Positive* as it is in positive cases where the delay penalisation affects the result. In this sense it is important to see that the decrease for $TaP_{o=5}$ is bigger than the one in $TaP_{o=50}$ due to the amount of items included in each case. On the first case, $Oracle_{10}$ and the following ones obtain values close to -1 while for

Table 4.7: Performance for Machine Learning models on batch evaluation. TaP and TaF use $\lambda = 0.01$ and $\alpha = 0.90$. Parameter o is set to two values, $()low, high)$, for $ERDE$ and TaP in each dataset: $(5, 50)$ for Depression and $(1, 10)$ for Network Attacks.

Dataset	Model	$ERDE_{o=low}$	$ERDE_{o=high}$	$F_{latency}$	$TaP_{o=low}$	$TaP_{o=high}$	$TaF_{o=low}$	$TaF_{o=high}$
Depression	AB_1	0.1407	0.1097	0.3096	0.2762	0.2855	0.3096	0.3116
	AB_5	0.1737	0.1081	0.3016	0.2878	0.2994	0.3222	0.3247
	ET_1	0.1458	0.1457	0.0929	-0.7121	-0.7088	0.0938	0.0938
	ET_5	0.1497	0.149	0.0378	-0.764	-0.7603	0.0409	0.0409
	$LSVC_1$	0.1513	0.1513	0.0141	-0.7903	-0.7869	0.0143	0.0143
	$LSVC_5$	0.1522	0.1522	0.0	-0.8037	-0.8	0.0	0.0
	LR_1	0.1298	0.0951	0.363	0.354	0.366	0.3624	0.3652
	LR_5	0.1537	0.0776	0.394	0.5177	0.5322	0.4206	0.424
	RF_1	0.1489	0.1488	0.0414	-0.7633	-0.76	0.0419	0.0419
	RF_5	0.1511	0.1511	0.0132	-0.7904	-0.7867	0.0143	0.0143
Network Attacks	AB_1	0.4337	0.0001	1.0	0.9857	0.9939	1.0	1.0
	AB_5	0.6338	0.0003	0.5002	0.9767	0.9939	0.995	1.0
	ET_1	0.4337	0.0001	1.0	0.9858	0.9939	1.0	1.0
	ET_5	0.6338	0.0003	0.5002	0.9768	0.9939	0.995	1.0
	$LSVC_1$	0.4393	0.0113	0.9935	0.9625	0.9707	0.9935	0.9935
	$LSVC_5$	0.637	0.0115	0.497	0.9536	0.9706	0.9885	0.9935
	LR_1	0.4403	0.0068	0.9956	0.9743	0.9824	0.9956	0.9956
	LR_5	0.6405	0.0069	0.498	0.9653	0.9824	0.9907	0.9956
	RF_1	0.4337	0.0001	1.0	0.9857	0.9939	1.0	1.0
	RF_5	0.6338	0.0003	0.5002	0.9767	0.9939	0.995	1.0

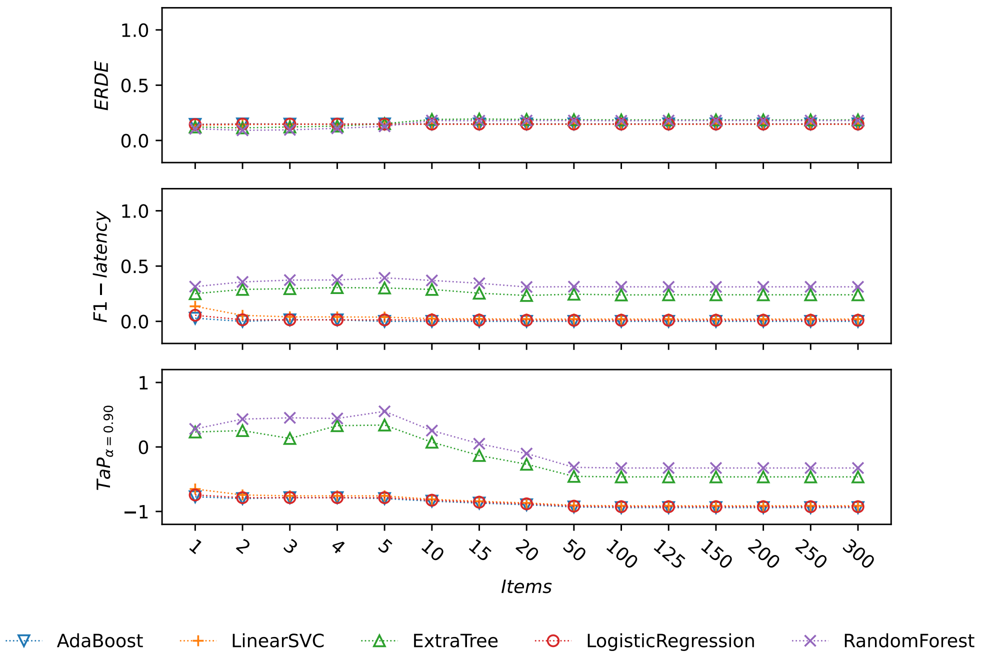


Figure 4.9: TaP (bottom), $F_{latency}$ (middle), $ERDE$ (top) for Depression dataset with $o = 5$, $\lambda = 0.1$ and $\alpha = 0.9$. Baseline models, *Oracle*, *Elcaro*, *Positive*, *Negative* and *Random*, are showed on each figure. The X-axis represents the number of items used by the baseline models to compute the predictions for all entities.

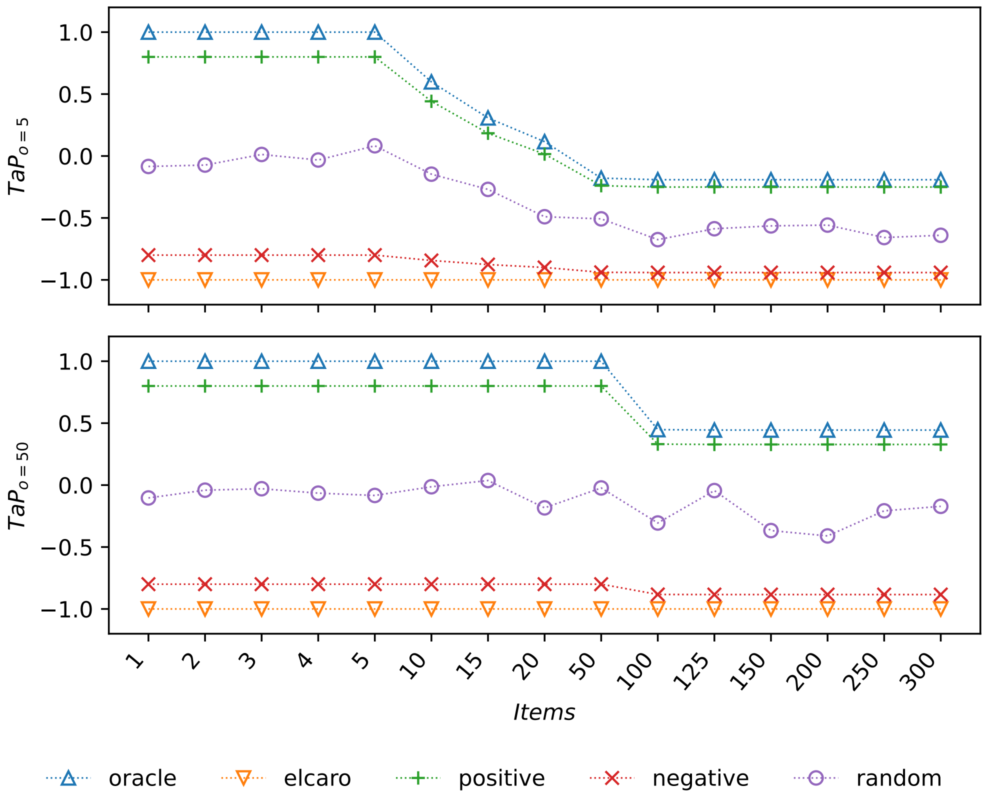


Figure 4.10: TaP penalisation for streaming evaluation over Depression dataset with $\lambda = 10$ and $\alpha = 0.90$. TaP with $o = 5$ (top) and $o = 50$ (bottom) for Depression dataset. Parameters λ and α are fixed to 10 and 0.9, respectively. Baseline models, *Oracle*, *Elcaro*, *Positive*, *Negative* and *Random*, are showed on each figure. The X-axis represents the number of items used by the baseline models to compute the predictions for all entities.

the second one, the decrement on $Oracle_{100}$ is only about 0.25. This can be explained by the logistic function in the penalty which provide higher values for lower values of o.

Finally the results for the machine learning models selected and already studied in the batch evaluation are shown for streaming evaluation in Figure 4.11 for the Depression dataset and in Figure 4.12 for the Network Attack dataset. In those figures, the behaviour of $ERDE$, $F_{latency}$ and $TaP_{\alpha=0.9}$ metrics are displayed. In the case of $ERDE$ and TaP the point of penalisation chosen is the one corresponding with the $o = low$ previously defined value (i.e. $o = 5$ for Depression and $o = 1$ for Network Attacks). Also, and as shown the previous streaming evaluation analysis of synthetic models, λ

parameter will be set to 0.1 in order to better display the impact of penalisation when the number of items where the measure is taken are close to each other. This is the situation of the first measures in the presented graphics as points are spaced as their value increase. As a result of the distribution of the measurement points and mostly due to the performance of the algorithms, the metric output is almost completely stable. Some exceptions can be found in Figure 4.12 for Network Attack dataset, where a harsh change is present for all the metrics but specially for $ERDE$ and $F_{latency}$ as the majority of anomalous entities have around 2 items. That combined with the point $o = low$ where the penalty is introduced, generates the decrease of the metrics. Also, in the case of Depression dataset, as shown in Figure 4.11, *RandomForest* and *ExtraTree* display a good performance both for $F_{latency}$ and TaP metrics, getting the highest value at 5 items. Meanwhile, $ERDE$ cannot clearly differentiate between the results obtained by those models. These results demonstrate, as seen for baseline models, the higher granularity provided by TaP metric against $ERDE$ and $F_{latency}$ and how it can help to discriminate between models with similar performances.

Overall, changes observed on the output value of TaP metric for different models and datasets can be taken into account and explained. The increase of the metric value up to item 5 in Depression dataset, for example, could be explained by some algorithms taking better decisions after at least two items being processed. These situations appear also for the rest of the sequence but as penalisation increases from item 5 onward it does not improve the final value. After that number of items a stabilisation of the outcome can be observed due to the x-axis does not show a proportion of the total amount of items but a more natural approach to the distribution of items in time.

4.4 TaF

As performed for TaP metric in previous section, an analysis of *Time aware F-score* or TaF for short, defined in Section 3.3.6, will be performed both for batch and streaming evaluation as presented in Section 3.2. For this particular metric, observations will be made mostly with $F_{latency}$ as both represent a *F-score* with a delay penalisation.

4.4.1 Batch

First, the results for TaF metric for synthetic models can be found on Table 4.6 were both results for Depression and Networks Attacks datasets are included. It is interesting to notice how the small differences of both metrics

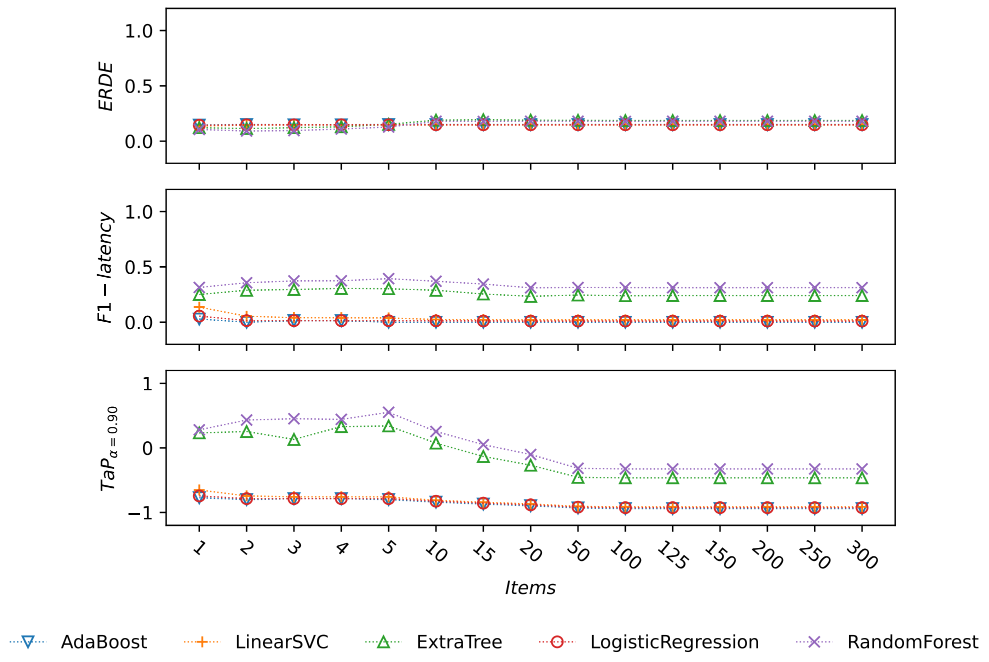


Figure 4.11: Comparison between TaP, $ERDE$ and $F1-latency$ for streaming evaluation over Depression dataset with $\lambda = 0.1$ and $\alpha = 0.90$. Machine Learning selected models are showed in each figure. The X-axis represents the number of items used by the baseline models to compute the predictions for all entities.

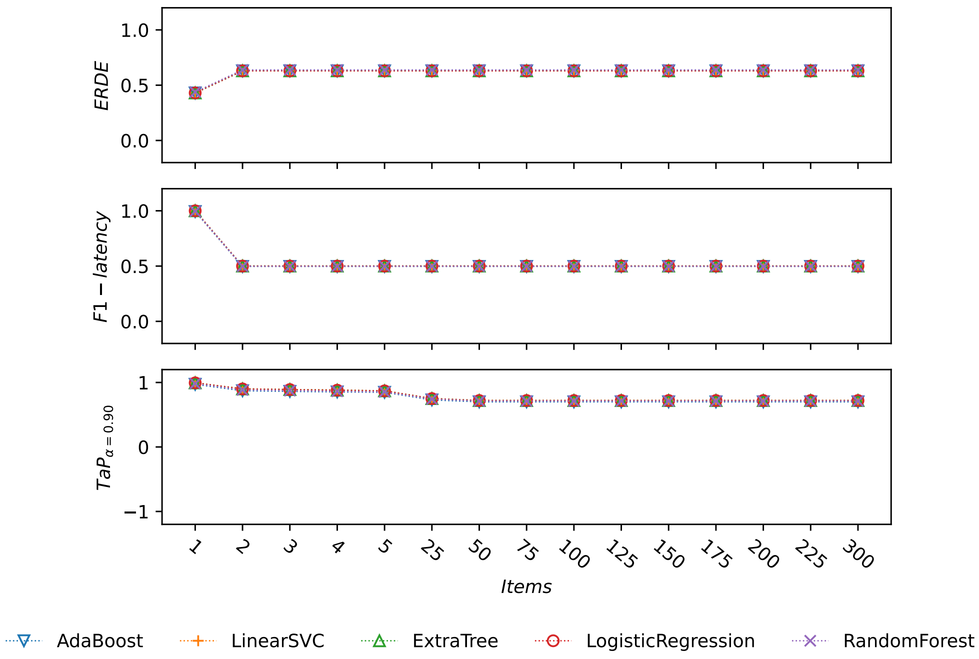


Figure 4.12: Comparison between TaP, $ERDE$ and $F1-latency$ for streaming evaluation over Network Attacks dataset with $\lambda = 0.1$ and $\alpha = 0.90$. Machine Learning selected models are showed in each figure. The X-axis represents the number of items used by the baseline models to compute the predictions for all entities.

values when analysing the results for all models but $Oracle_1$ and $Oracle_5$. In these cases, as the penalty function varies and is much steep for $F - latency$ some differences can be spotted, being the most evident one the variation between $Oracle_1$ and $Oracle_5$ for $F_{latency}$ metric in the Network Attacks dataset. As introduced in the analysis of TaP metric, this is due to the distribution of the number of items per entity and how $F_{latency}$ generates its penalty.

Then, on Table 4.7 the results for the selected set of ML models are presented. Here, and following the same explanation given for synthetic models, both metrics behave similarly for models with poor results, and the bigger variations are defined by the different penalisation approach of both metrics. Focusing on Network Attacks dataset, all models present a high performance for both metrics on its first item version (e.g. $AdaBoost_1$) but when the second point of decision (5) is evaluated $F_{latency}$ scores decrease almost to 50% of the original score, while TaF maintain smaller differences.

4.4.2 Streaming

For streaming evaluation and in order to better display differences between TaF and $F_{latency}$ results, a plot for each one of the selected models and metrics are included. First, results for the synthetic models are shown in Figure 4.13 for Depression dataset, there it can be observed that all but *oracle* model, obtain similar results for both metrics with some slight differences in the case of *positive* and *random* models. This can be explained by the metrics penalizing different on the correctly detected positive cases, that can be expected both on *positive* and *random* models. *Oracle* model shows bigger differences resulting in higher penalties with TaF metric. This difference increases from the penalisation point ($o = 5$) onwards.

Machine Learning models results are shown in Figure 4.14 for Depression dataset and in Figure 4.15 for Network Attacks. Each one of the plots portray the behaviour of TaP and $F_{latency}$, as presented for the synthetic models, over the amount of items used for the prediction for the following models: *Ada Boost (AB)*, *Extra Tree (ET)*, *Linear Support Vector Classification (LSVC)*, *Linear Regression (LR)* and *Random Forest (RF)*.

It must be noticed that as presented in batch evaluation, differences for both models are small when the models perform poorly as it can be clearly seen for Figure 4.14. But, when the results achieved are better, the difference in penalty clearly differentiate both scores. This can be observed in Figure 4.15 where $F_{latency}$ results abruptly decrease after item number 2, where its penalty indicated that it should achieve 50% of the value. TaF maintains a higher value and the penalisation could be increased by means of the λ penalisation factor.

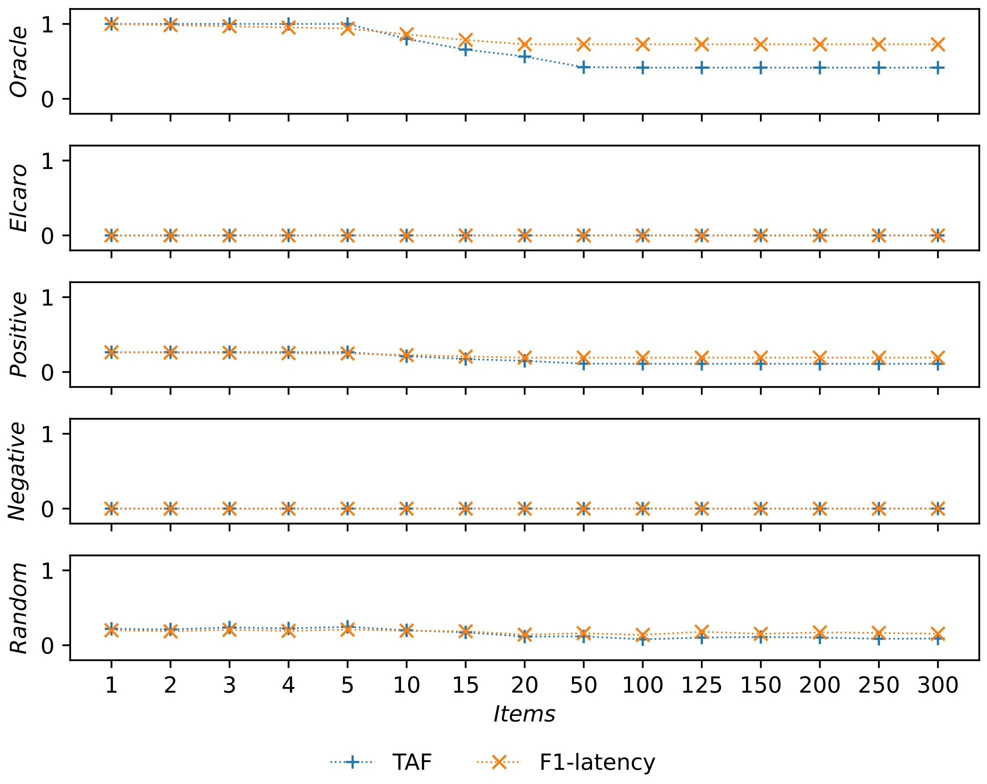


Figure 4.13: Comparison between TaF and $F1_{latency}$ for streaming evaluation over Depression dataset with $\lambda = 0.1$ and $\alpha = 0.90$. Synthetic generated models are showed in each figure from up to bottom: *Oracle*, *Elcaro*, *Positive*, *Negative* and *Random*. The X-axis represents the number of items used by the models to compute the predictions for all entities.

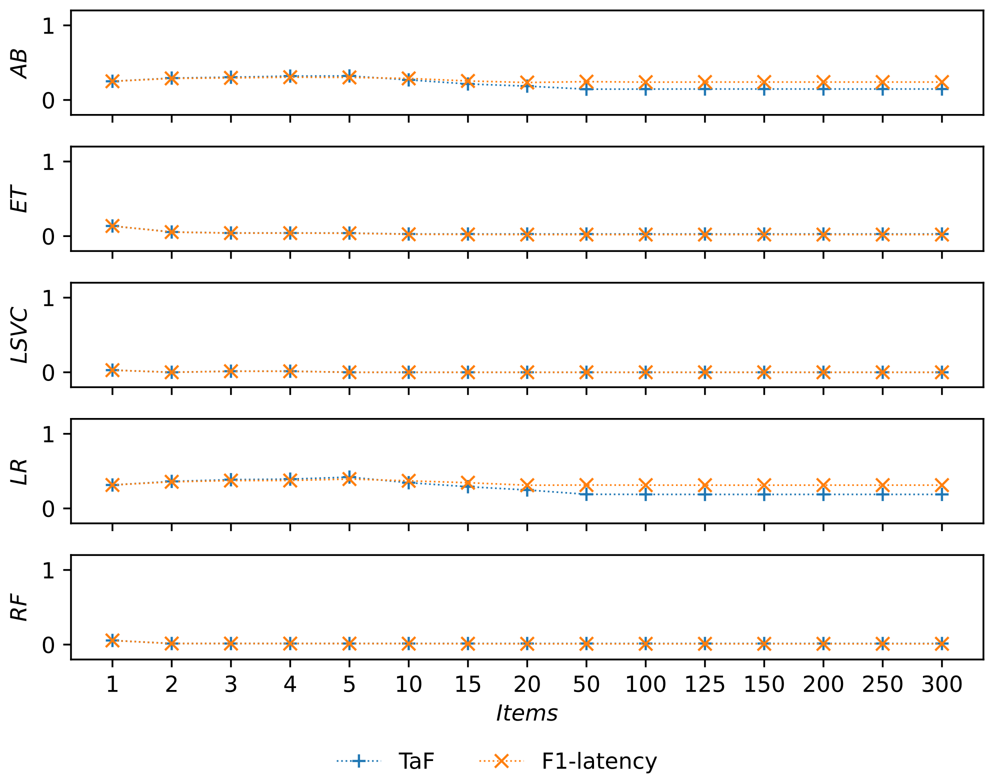


Figure 4.14: Comparison between TaF and $F1_{latency}$ for streaming evaluation over Depression dataset with $\lambda = 0.1$ and $\alpha = 0.90$. Machine Learning selected models are showed in each figure from up to bottom: Ada Boost (AB), Extra Tree (ET), Linear SVC (LSVC), Logistic Regression (LR), Random Forest (RF). The X-axis represents the number of items used by the baseline models to compute the predictions for all entities.

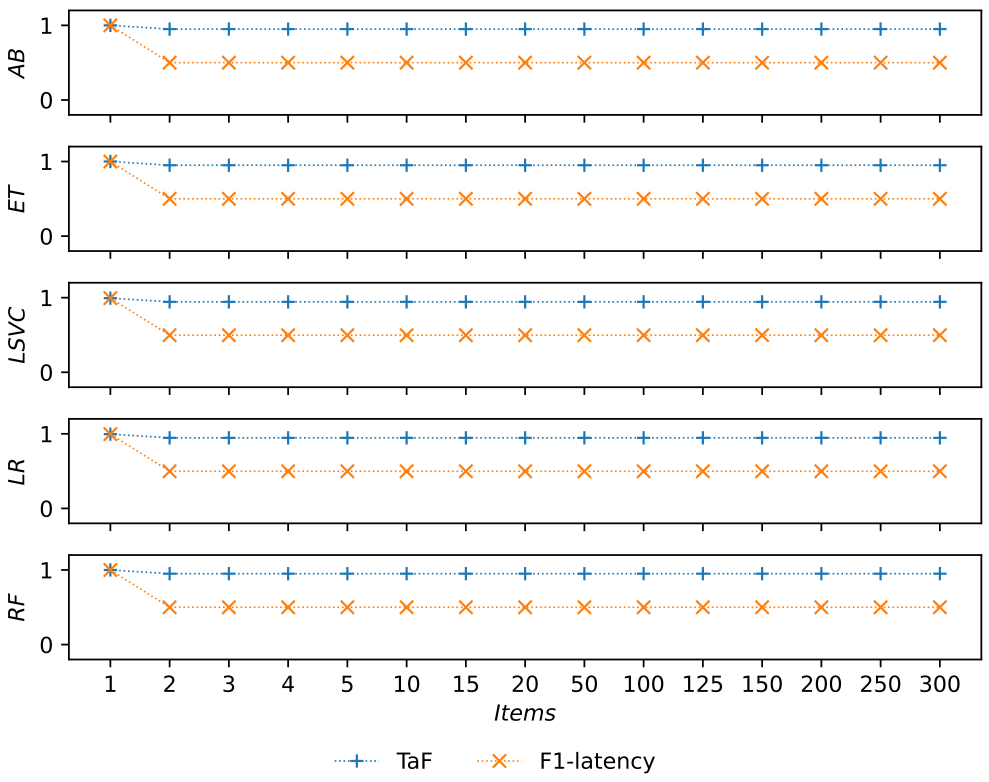


Figure 4.15: Comparison between TaF and $F1_{latency}$ for streaming evaluation over Network dataset with $\lambda = 0.1$ and $\alpha = 0.90$. Machine Learning selected models are showed in each figure from up to bottom: Ada Boost (AB), Extra Tree (ET), Linear SVC (LSVC), Logistic Regression (LR), Random Forest (RF). The X-axis represents the number of items used by the baseline models to compute the predictions for all entities.

Chapter 5

Early Detection Methods

In this chapter an introduction to early detection methods is provided. Three specifically defined early detection methods are presented: fix, threshold and dual. Also, it includes a review of features for cyberbullying detection on social networks with an early detection aim.

5.1 Models

All the presented models take into account the fact that this is not a classical binary classification problem, because a non-final decision can be emitted. The output of the models could be a negative decision, a positive one, meaning that the anomaly or situation has been detected, and a delay which models the will of non taking a decision yet by the model. This situation will avoid taking a decision too early and waiting for more information to decide properly about the output.

For the baseline of all experiments regarding the models and features presented in this chapter, which results are shown in Chapter 6 a set of models has been selected. The following algorithms were used as baselines for fixed, threshold and dual models as they achieved good results in previous works [9, 84]:

- *AdaBoost (AB)*: Meta estimator that refits a classifier by updating weights of incorrectly classified instances.
- *Extra Trees (ET)*: Meta estimator that uses averaging of randomized decision trees for classification.
- *Logistic Regresion (LR)*: Conformed by a logit MaxEnt classifier, where the maximum entropy classifier is combined with a logistic function.

- *Naïve Bayes (NB)*: Uses Bayes' theorem with the assumption of conditional independence between features.

- *Support-Vector Machine (SVM)*: Defines a set of hyperplanes in a high dimensionality space in order to divide different classes of input.

The models considered in this section and presented next are: fix model, a basic one where the decision is taken into a predefined point; threshold model, where a level of confidence is defined for the classes and dual, where an specialized model is trained for the different classes and combined to obtain the final output of the model. Those last two models in particular have reported good results in previous works [97, 82]

5.1.1 Fix model

This model has the simplest mechanism of all the early detection methods proposed and uses a predefined point to decide whether the entity E is classified as positive (E^+) or negative (E^-).

$$\delta_{fix}(m, i)$$

Where i is the point where the decision is taken and represent the amount of elements processed before the decision is emitted. In this case, and following the naming presented in Section 3.1, i represents an *item* of the Entity E being evaluated.

As it will be shown in Chapter 6, the granularity definition for the point of decision can be changed. The first option, or coarse grain would be to use the *item* as subdivision of the entity. The second option would be to use a smaller portion of the entity, a subset of the *item* to fix the point where the decision will be taken. In the latter, more intermediate decisions could be defined, achieving a fine grain fix decision model.

5.1.2 Threshold model

Threshold model, as presented in [9], is a variation of the singleton model from [82] based on one learning model, which is trained with the whole set of entities used for training. It then integrates a decision function based on the class probabilities to determine if enough evidence is available to proceed with a firm decision. Otherwise a delay is emitted and the model awaits to get more information before making a decision.

Initially, the decision function for the threshold model ($\delta_{threshold}$) is defined as:

$$\delta_{threshold}(m, th^{+}(), th^{-}())$$

Here, m designates a specific machine learning model (e.g. one of the presented baseline models) over which the threshold model will be built. The thresholds are defined as $th^{+}()$ and $th^{-}()$, where the first one denotes the threshold function to set the limit for a positive final decision and the second one represents the same for the negative cases.

As threshold functions, several options have been explored, from independent functions for positive an negative decisions to several decreasing functions depending on the number of items processed. Opposite from the results shown in [82], best performance shown in the Chapter 6 was obtained with a constant function both for positive and negative cases. The function, defined as: $th() = I$ was tested for different values of I for the experiments provided. The difference in performance from the one shown in [82] and the ones provided in this document may be motivated to the fine grain evaluation performed, as a streaming approach was followed instead of a batch in the "original paper".

Regarding the order of application of decision functions, there was no difference whatsoever and the same results were obtained if the positive threshold function $th^{+}()$ was applied prior or subsequently to the negative one ($th^{-}()$) . For the sake of reproducibility in future experiments, in these experiments first the positive function was applied, followed by the negative if there was no decision on the positive threshold.

5.1.3 Dual Model

Dual model defines a different and independent model for each one of the options (i.e. positive and negative). Inspired by multiclass classifiers one-versus-all, it establishes two independent learning models defined with each set of features. One of them is trained to detect positive cases and the other negative cases (represented by m^{+} and m^{-} respectively). It must be noted that each of the models could use a different set of features in order to improve an specific class detection. The decision function for this model is defined, following the proposal from [82], as shown next:

$$\delta_{dual}(m+, m-, th+(), th-())$$

In the representation of δ_{dual}, m^+ is the model trained for positive predictions and m^- the one for negative predictions. As defined in 5.1.2, a threshold function is defined. In this case, a specific function for each one of the models, $th^+()$ for the positive model responsible of positive cases and $th^-()$ the threshold model for the negative model. In the same sense as presented, different values of I must be explored in these models.

As it can be seen in the definition of this model, a combination of models and techniques is applied in order to define threshold and dual model implementations. These are expected to capture de special characteristics of early detection problem in a better way than the standard baselines and therefore to improve the performance in terms of early detection metrics.

5.2 Features

As important as model definition is the data input used to feed the model, specifically the features used. This selection of features depends both on the information available and the methods applied for the generation or extraction of new features.

To study the cyberbullying early detection problem, as displayed on the experiments of Chapter 6 a dual approach can be taken. Firstly, if particular features from specific environments (i. e. the specific social network) are used, and then if those particular features are dismissed losing that bit of information in return of a more generalized method that can be applied for early detection of cyberbullying regardless the specific platform where it is applied.

A social media session of the datasets used in Chapter 6 is composed by a posted element, that in the presented datasets could be represented by an image or a video along with a description and all the likes and comments associated. These sessions represent entities (E) that are labelled as positive or negative. And the decisions will me made in relation to the number of comments (or words if specified) used to take the decision.

For the experiments included in Chapter 6 the starting point will be the set of features which better performed in [84], classified as follows:

- Profile owner features: that capture the characteristics of the user who posted the initial video. These features include numbers of followers and following, polarity and subjectivity of the user's profile description.

- Media session features: number of likes, comments and sharing and polarity and subjectivity of media caption.

- Comment features: that are intended to determine the negativity associated with the comment. These features include percentage of negative comments, profane words in the comment, average polarity and subjectivity for the comments, differentiating between owner and other comments.

- Video features: intended to capture the nature of the video, these features validate the emotions and content in the video.

- LDA features: top ten topics extracted using Latent Dirichlet Allocation from all comments.

These features had also been extended with two new groups of characteristics that may be relevant for the early detection problem: Bag-of-Words (BoW) similarity and time aspects.

BoW similarity is considered following previous works such as [98, 59, 64], and the aim is to achieve the computation of these features without supervision. To do so, the training part of the dataset is divided into two disjunctive sets by its label: cyberbullying and non-cyberbullying sessions. These features will represent the likeliness of a comment belonging to cyberbullying or normal set of comments without considering a set o predefined terms (e.g. profane words).

To compute these features, for each comment, the average, standard deviation, minimum, maximum and median of the Term Frequency-Inverse Document Frequency (TF-IDF) similarity is computed by analysing each comment in contrast ot other cyberbullying comments. This same process is performed for the non-cyberbullying comments and in both cases the comment from which de values are being computed is removed from the sample before the values are obtained.

The early detection problem has clear time dependencies and cyberbullying implies a certain repetition over time, as it can be observed from the statistics analysis presented over the datasets used in Chapter 4. There, the data shows that there is a shorter time span for cyberbullying sessions. Therefore, it was considered relevant to include features that are able to capture time aspects of the input.

For this purpose, features that measure the time difference between consecutive comments were generated. There are two cases:

- Time difference with the last comment.

- Time difference from all previous comments.

In both cases, values are aggregated by calculating the average, median, maximum and minimum values, and included as new features in for the item processed, or in this specific context, the comment of the media session.

5.2.1 Doc2Vec

Doc2Vec was presented in [57] as an alternative vector representation for text features that provides a semantic wise and dense result vector which outperforms state of the art methods for text classification and sentiment analysis tasks. By using an unsupervised algorithm it is capable of generating fixed-length feature representations for variable length pieces of text.

This method is composed by three steps, where first the unsupervised learning model is trained with the pieces of text from the datasets. On this step, information is extracted by means of sliding a fixed length window over the text. The resulting word vectors obtained after training can be used directly as input for machine learning algorithms in order to classify by its input labels. Also, although not applied in this case, after an inference step to generate paragraph vectors, it an be used on his third step, to make predictions about particular labels.

The paragraph vectors allows also to address some of the problems of other word representations such as BoW models. One of its main characteristics is to be able to maintain word semantics on the representation. Also, another advantage of this method is the preservation of word order, at least in a small context, which leads to allegedly to a good representation with a better generalization than other methods.

5.2.2 Multiple Instance Learning

Multiple Instance Learning (MIL) [99] paradigm assumes that labels are assigned to sets or bags, or in the references in this chapter and the evaluation of the methods, to media sessions from Social Media Networks. This possess a problem, as training examples could be considered ambiguous because single objects, posts of media sessions in the presented experiments, are expected to have multiple alternative instances that conforms them. Only some of of the posts may be responsible for the classification of the media session as either positive or negative.

In order to include these capabilities, an early phase is introduced to generate new media session representation that can then be supplied to early detection models an executed as usual. Specifically the use of MIL will be focused on instance-space methods that obtains the labels assigned to the bags by aggregating individual instance features, as shown in [100].

This aggregation will be achieved by means of the following functions:

- *minimum*
- *maximum*
- *average*
- *median*
- *average* between *minimum* and *maximum*

On each case, the named function will be used to determine which label represents the set of instances that were assigned to the bag that it is being processed. After this task is performed, and as previously introduced, the generated representation is later used on the following phase where a machine learning model is trained to classify the media sessions.

Chapter 6

Methods Evaluation

The following sections present the results for each of the three sets of experiments performed. First, we study how does the inclusion of Doc2Vec features affect all the proposed models. Then, a fine-grain reduction is proposed for fixed model switching from post counting to word number. Finally, we incorporate Multiple Instance Learning to fixed early detection and threshold models. Complementary data for $F_{latency}$ plots, where non present in this chapter, can be found on the Appendix Additional Data (7.2).

6.1 Baseline

6.1.1 Datasets

For the evaluation of the methods for early detection proposed on Chapter 5 two datasets tagged for cyberbullying and obtained from different social networks have been used: Instagram and Vine.

The Instagram dataset, which is a media-based social networks mainly focused on sharing images but also videos, where users can comment on the posts that contain those resources, was created as described by Hosseinmardi et al. in [101, 87]. To collect the dataset the authors started with an initial sample of 25 thousand users with public profiles and gathered a set of more than 3 million media sessions. In order to achieve a more balanced dataset, sessions with less than 15 comments were discarded and also only those with at least one profane word on the comments were considered. The result of this process is a dataset with $2,218$ media sessions that were labelled for cyberbullying with human contributors. To achieve this, a version of the majority voting method is used where each contributor receives a level of trust based on a series of tests ans quizzes and a minimum of 60% is required to keep the session in the dataset. Each one of the sessions for this dataset include their own set of comments.

A similar approach was followed to create the other used dataset, Vine, which was a social network based on a mobile application to record, edit and publish short videos on user profiles. As for the Instagram, in this case users can watch and comment on these videos. In order to create the dataset Rafiq et al. [84, 83] started with a sample of more than 650 thousand media sessions that were filtered, in the same way as for Instagram dataset, to obtain just session with at least 15 comments, which reduced the group to 436 thousand sessions. Later, 969 media sessions were selected after classifying them by the percentage of comments with profane words and selecting a representative subsample for all the labelled groups. Each one of the selected sessions were then tagged by 5 reviewers and if the confidence level for the session was below 60% it was discarded, obtaining a final set of 747 session from Vine social network.

The main statistics for both datasets can be found in Table 6.1. When comparing both datasets it can be seen that Vine dataset is smaller both in terms of number of sessions and comments but it provides a higher rate of comments per session.

Table 6.1: Statistics for Instagram and Vine cyberbullying datasets.

	Instagram			Vine		
	Cyberbullying	Normal	Total	Cyberbullying	Normal	Total
Media sessions	585	1,369	1,954	190	557	747
	29.94%	70.06%	100%	25.44%	74.56%	100%
Comments	45,372	76,862	122,234	15,810	40,389	56,199
	37.12%	62.88%	100%	28.13%	71.87%	100%
Comments/session	77.56	56.14	62.56	83.21	72.51	75.23
Words/comment	14.00	7.13	9.68	7.64	4.96	5.72
English	74.16%	73.22%	73.74%	47.62%	34.43%	38.69%

The distribution of the number of words per comment are shown in Figure 6.1(a) for the Instagram dataset and in Figure 6.1(b) for Vine dataset. As it can be observed on these graphs, Instagram comments are much longer than Vine, with an average of around 10 words per comment, while Vine just reaches around 5 words per comment. It is also interesting to see that in the case of Instagram there is a long queue reaching up to 400 words which leads to a standard deviation in this case of 17.46 although Vine just reaches to nearly 30 words that produces a much smaller standard deviation of 5.41. Regarding the size of the comments, as it can be seen in Figure 6.1, bullying comments on Instagram have the tendency of being larger but this behaviour is less noticeable, just appearing on the tail of words distribution for comments in Vine dataset.Also, an analysis of the language used on these social media platforms has been performed and, as shown in the Table 6.1, Instagram comments are mostly in English but Vine shows a higher variability of the languages used. This was seen as a representation of the characteristics of the dataset because Vine comments are smaller in size, as displayed in Figure 6.1, and that lead to a higher use of abbreviations and slang.

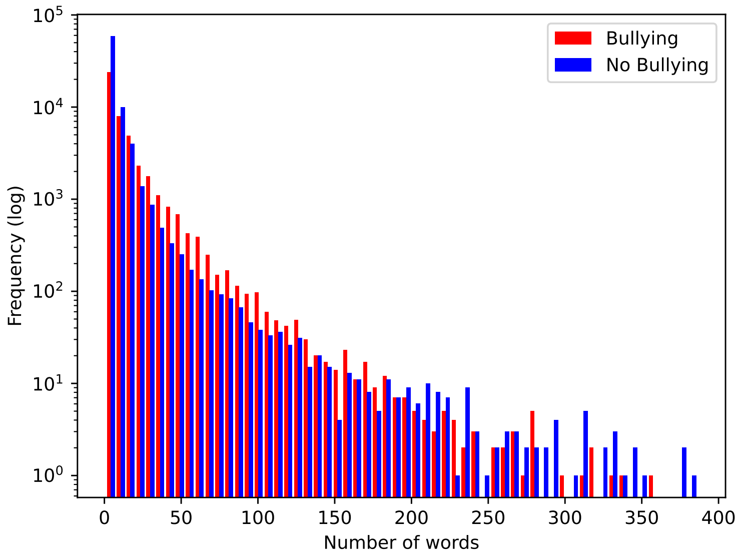
Finally, regarding the features used, as described in Section 5.2, in order to study the development of early detection models ignoring the specific social network, from the original set (i.e. *Profile owner features*, *Media session features*, *Comment features*, *LDA features*, *Video features*, *BoW features* and *Time features*), and the proposed ones, the following set has been considered:

- LDA features: Latent Dirchlet Allocation
- BoW features: Bag of Words
- Tf-idf: Term frequency - inverse document frequency
- Doc2Vec: Vectorial representation of textual sets.
- MIL: Multiple Instance Learning

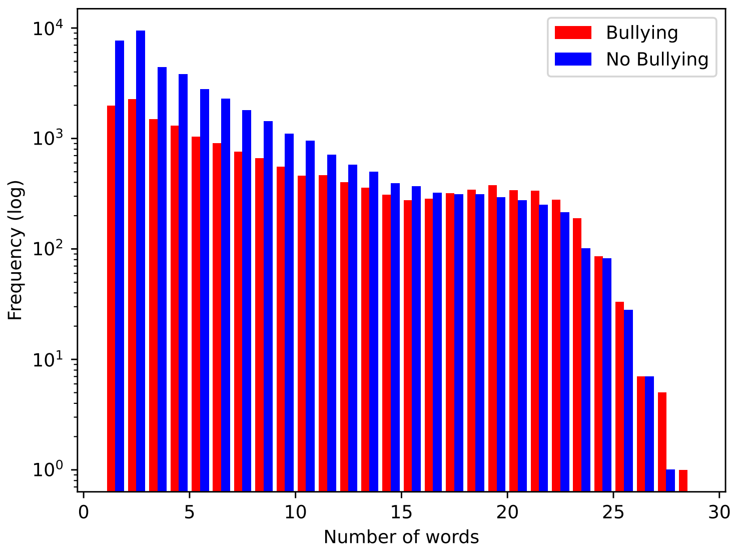
6.1.2 Models

As previously stated, there are no extensive research in cyberbullying early detection with an early detection metrics evaluation, that only uses data from comments. Some research found on this topic with results over Instagram and Vine datasets are [72] which reaches an $F1$ value of 0.62 ± 0.03 (*precision* 0.79 ± 0.03 and *recall* 0.55 ± 0.03) using a Linear Regression model for Instagram dataset. Similar result were achieved by [74] with an $F1$ score of 0.65 (*precision* 0.873 ± 0.04 and *recall* 0.517 ± 0.05). Also, for Vine dataset, analogous results were obtained with an $F1$ score of 0.59 ± 0.04 (*precision* 0.62 ± 0.05 and *recall* 0.57 ± 0.05) using Linear Regression too.

In order to stablish a baseline to compare following results and as stated in [9], some experiments comparable with the results presented here were conducted, and the results are shown on Table 6.2. There, time aware metrics ($ERDE$ and $F_{latency}$) and precision and recall values are included to allow both a comparison between them and with future results. To achieve this a selection of machine learning models extracted from the ones which presented better results for Vine dataset in [84] were used in a simple early detection fix model evaluating its performance at a fixed point. Considering this, the resulting models are:



(a) Instagram



(b) Vine

Figure 6.1: Histograms for comments number of words for the Instagram dataset (6.1(a)) and Vine dataset (6.1(b)). Logarithmic scale is employed for Y-axis (frequency). Bullying and non-bullying comments are differentiated on each histogram.

- *Random Forest* at point 5: RF_5
- *Ada Boost* at point 5: AB_5
- *Extra Trees* at point 5: ET_5
- *Support Vector Classification* at point 5: SVC_5
- *Logistic Regression* at point 5: LR_5

As base models for the rest of experiments included in this chapter regarding the early detection models proposed in Chapter 5 results from previous works with early detection model were considered [93, 94, 9]. Particularly, the following models were selected as they shown relevant results for fixed, threshold and dual early detection models:

- *Ada Boost* (ADA)
- *Extra Trees* (ET)
- *Logistic Regression* (LR)
- *Naïve Bayes* (NB)
- *Support-Vector Machine* (SVM)

6.2 Baseline Features

6.2.1 Fix Model

Fixed early detection model, as described in 5.1.1, is a simple adaptation of any standard machine learning model considering a determined fix number of items as entry to the model before making the final decision, while this point is not reached a delay is emitted. On the experiments analysed on this section 1, 5, 10, 15 and 25 comments are used to define the models.

For simplicity models are named with the name of the machine learning model and a subscript specifying the number of posts used to make the decision. Following that, for example an *Ada Boost* model that takes the decision with 5 comments will be denoted by ADA_5 or an *Extra Trees* model that uses 15 comments, ET_{15}. Each one of these models will produce a different value of performance on the point where the decision is taken.

The evaluation of the proposed models in the defined points (i.e. 1, 5, 10, 15 and 25) can be found in Figure 6.2 for Instagram dataset and in Figure 6.3 for Vine dataset. In those figures, the X-axis represent the features used

Table 6.2: Results for baseline models for 5 comments using individual groups of features in Vine dataset. Best results for each group of features are highlighted for time aware metrics and underlined for precision and recall.

	RF_5	AB_5	ET_5	SVC_5	LR_5
Profile owner features					
$ERDE$	0.1757	**0.1685**	0.1617	0.2054	0.1712
$F_{latency}$	0.1414	0.1363	0.2672	**0.2759**	0.1272
$Precision$	0.1905	0.3333	<u>0.4118</u>	0.1805	0.2500
$Recall$	0.1212	0.0909	0.2121	<u>0.7273</u>	0.0909
Media session features					
$ERDE$	0.1609	0.1746	**0.1599**	0.1967	0.1710
$F_{latency}$	**0.2881**	0.0465	0.2783	0.2196	0.0000
$Precision$	0.4000	0.1250	<u>0.4667</u>	0.1625	0.0000
$Recall$	0.2424	0.0303	0.2121	<u>0.3939</u>	0.0000
Comment features					
$ERDE$	0.1642	0.1694	**0.1556**	0.1627	0.1575
$F_{latency}$	0.2121	0.1332	**0.3368**	0.2776	0.3348
$Precision$	0.4167	0.3000	<u>0.5000</u>	0.3636	0.4167
$Recall$	0.1515	0.0909	0.2727	0.2424	<u>0.3030</u>
LDA features					
$ERDE$	**0.1633**	0.1738	0.1668	0.1686	0.1705
$F_{latency}$	0.1909	0.1193	0.1735	0.1660	**0.2045**
$Precision$	<u>0.5714</u>	0.2000	0.3636	0.3077	0.2609
$Recall$	0.1212	0.0909	0.1212	0.1212	<u>0.1818</u>
Video features					
$ERDE$	0.1728	**0.1719**	0.1728	**0.1719**	**0.1719**
$F_{latency}$	**0.0489**	0.0000	**0.0489**	0.0000	0.0000
$Precision$	<u>0.1667</u>	0.0000	<u>0.1667</u>	0.0000	0.0000
$Recall$	<u>0.0303</u>	0.0000	<u>0.0303</u>	0.0000	0.0000
BoW features					
$ERDE$	**0.1659**	0.1719	0.1685	0.1710	0.1710
$F_{latency}$	**0.1468**	0.0502	0.1363	0.0000	0.0000
$Precision$	<u>0.5000</u>	0.2000	0.3333	0.0000	0.0000
$Recall$	<u>0.0909</u>	0.0303	<u>0.0909</u>	0.0000	0.0000
Time features					
$ERDE$	0.1730	0.1745	**0.1581**	0.1740	0.1710
$F_{latency}$	0.1497	0.0000	**0.2726**	0.1909	0.0000
$Precision$	0.2222	0.0000	<u>0.6667</u>	0.2222	0.0000
$Recall$	0.1212	0.0000	<u>0.1818</u>	<u>0.1818</u>	0.0000

on the model, the first three corresponds with standard features introduced in Section 5.2 which are as described the baseline to compare to for all the models being proposed. The rest represent Doc2Vec and combinations with baseline features (i.e. Tf-idf and LDA). For the sake of completeness, and to allow comparisons with previously obtained results, both $F_{latency}$ and TaP results are included for these experiments.

When Figures 6.2(a) and 6.2(b), for Instagram dataset results, are observed it can be noticed how best results are obtained when less than 10 of 15 items are used. These results are present for both metrics, but in the case of TaP, as a higher penalty is applied over 15 items, both 15 and 25 models present lower performance. Best results are found for 10 models with both metrics, but some cases as *ExtraTrees* for TI+LDA which gets similar values with $F_{latency}$ for 10 and 15 items, obtains its maximum value on 10 for TaP metric. This behaviour can be seen also in Figures 6.3(a) and 6.3(b), for Vine dataset. As seen in Instagram dataset, models 15 and 25 present higher penalties for TaP metric but the maximum value overall is still obtained with *Naïve Bayes* for the 10 item model, which gives also the best result for *Ada Boost* and *Logistic Regression*.

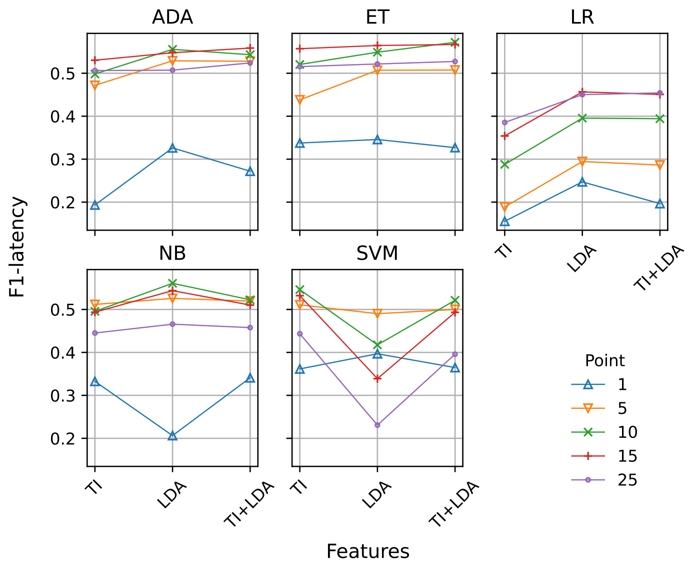
6.2.2 Threshold model

Threshold model, as described in Section 5.1.2, uses a standard machine learning model with a positive th^+ and a negative th^- threshold, based on the class probability given by the model to decide if the decision is final (i.e. cyberbullying or non-cyberbullying) or if it keeps emitting delays.

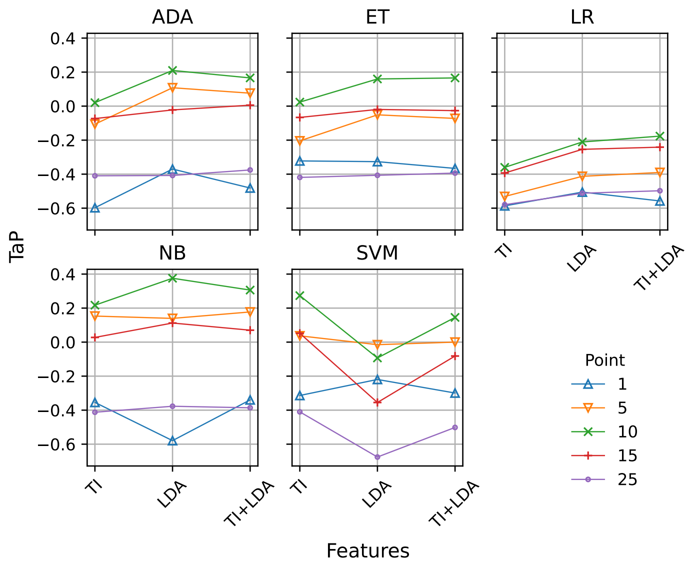
In order to decide if an specific entity is classified as cyberbullying or not at a determined item, the class probability will be first compared with the positive threshold (th^+) to decide if a cyberbullying decision is emitted. Otherwise, negative threshold (th^-) is compared for the non-cyberbullying prediction. If none of these conditions are fulfilled a delay is generated and more items will be processed before the final decision is emitted.

Experiments presented in this section will evaluate both the general performance of the threshold model and different values for positive and negative thresholds. To achieve this, values from 0.5 to 0.9 will be used, specifically low positive thresholds (i.e. 0.5 and 0.6) and high negative thresholds (i.e. 0.8 and 0.9) since previous best results were obtained with this combination of values [9].

Figure 6.4 and Figure 6.5 display the results for both datasets and metrics, following the experiments performed for fix model in the previous section. In the figures, in order to represent different combinations of positive and negative thresholds, different types of lines have been included in each one of

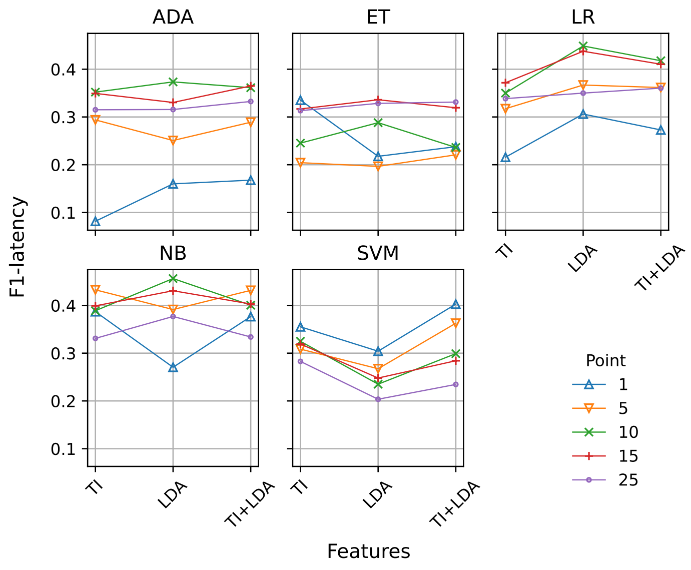


(a) $F_{latency}$

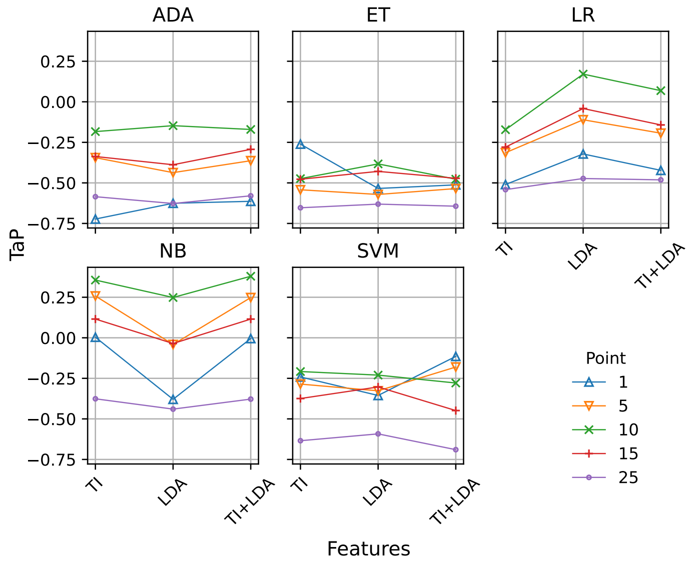


(b) TaP

Figure 6.2: $F1_{latency}$ (6.2(a)) and TaP (6.2(b)) for fixed early detection model at points $1, 5, 10, 15$ and 25 using Instagram dataset. One graph is presented for each machine learning model: AdaBoost (ADA), Extra Trees (ET), Logistic Regression (LR), Naïve Bayes (NB) and Support-Vector Machine (SVM). Features are represented on the X-axis.Tf-idf (TI) and Latent Dirichlet Allocation (LDA).



(a) $F_{latency}$



(b) TaP

Figure 6.3: $F1_{latency}$ (6.3(a)) and TaP (6.3(b)) for fixed early detection model at points $1, 5, 10, 15$ and 25 using Vine dataset. One graph is presented for each machine learning model: AdaBoost (ADA), Extra Trees (ET), Logistic Regression (LR), Naïve Bayes (NB) and Support-Vector Machine (SVM). Features are represented on the X-axis.Tf-idf (TI) and Latent Dirichlet Allocation (LDA).

the plots to represent the performance of the combination of model, features, and pair of thresholds.

As it can be observed on the figures, both metrics show similar behaviour for models performance with slight differences mostly due to changes in penalisation as for the same threshold not all models and feature combinations take the decision at the same point. Specifically, for Instagram dataset it can be observed how all models but *SVM* obtain better results when a higher negative threshold is applied, that is when $th^- = 0.9$. In the case of *SVM* can also be observed but it is due to the selection of a low positive threshold, $th^+ = 0.5$. Regarding Vine dataset, this behaviour is present in the majority of models as *Extra Trees*, *Logistic Regression* and *Naïve Bayes* obtain their best values with the combination of a high negative threshold and a low positive threshold ($th^+ = 0.5$ and $th^- = 0.9$) but for *Ada Boost* and again for *SVM* which achieve their best performance for $th^+ = 0.5$ and $th^- = 0.8$ although both cases are below the scores for the rest of the model disregarding the thresholds used.

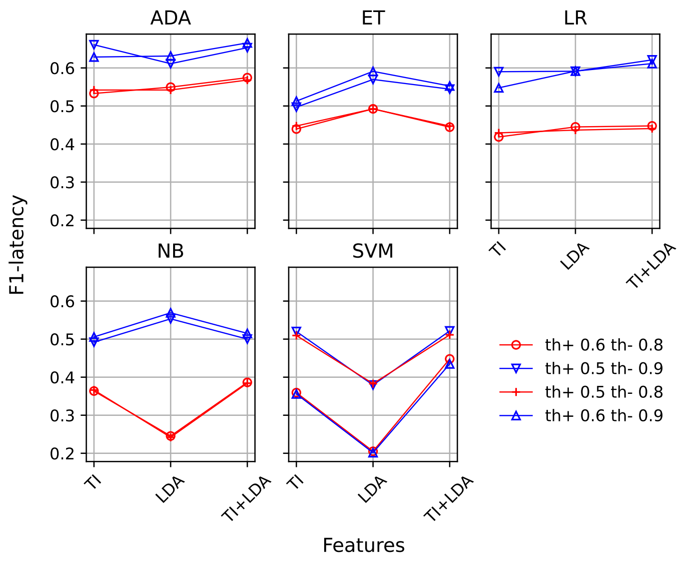
The motivation of the selection of these values as thresholds can be easily explained by the need to provide an easy pass threshold in the case of cyberbullying cases when little information is present. When negative cases are studied, a higher degree of confidence is required to decrease the possible amount of false positives from the set of cases that are supposed to be non-cyberbullying.

As previously observed for Fix model on Section 6.2.1 and on the results of this section, conclusions about the relation and representation of models performance using both $F_{latency}$ and TaP metrics can be extracted, and TaP results are representative of the time aware model evaluation.

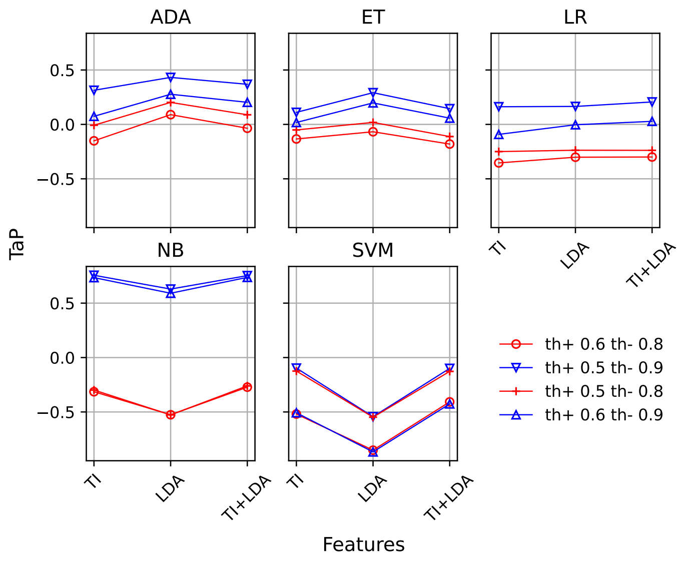
6.2.3 Dual model

Dual model, presented in Section 5.1.3, uses two independent machine learning models where one of them is trained to detect positive or cyberbullying cases m^+ while the other model is trained to detect negative or non-cyberbullying cases m^-. For each one of them both positive and negative thresholds are needed. As described for threshold model, when processing a new item, the class probability for m^+ must be higher than the defined threshold and in the same way, when m^- process the item its class probability must be above the negative threshold th^- in order to be considered ass non-cyberbullying. If those conditions are not fulfilled a delay will be emitted and more items will be required to obtain a final decision.

In order to study the performance of dual model, and in the same way did for previous models (i.e. fix and threshold), different combinations of

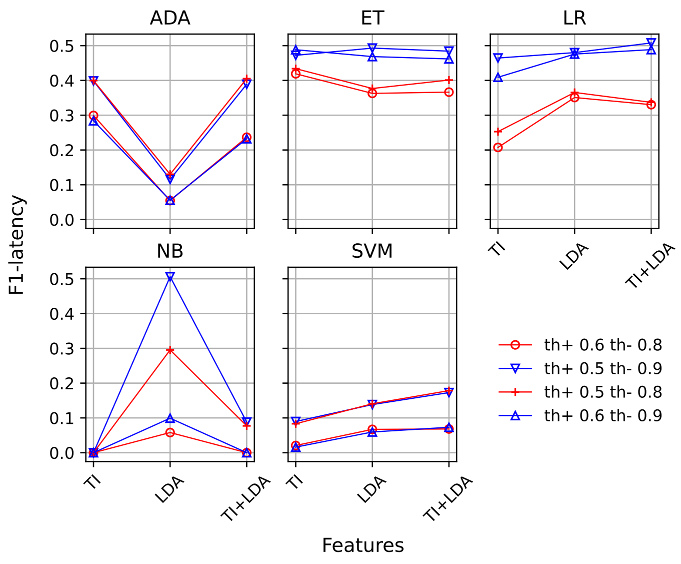


(a) $F_{latency}$

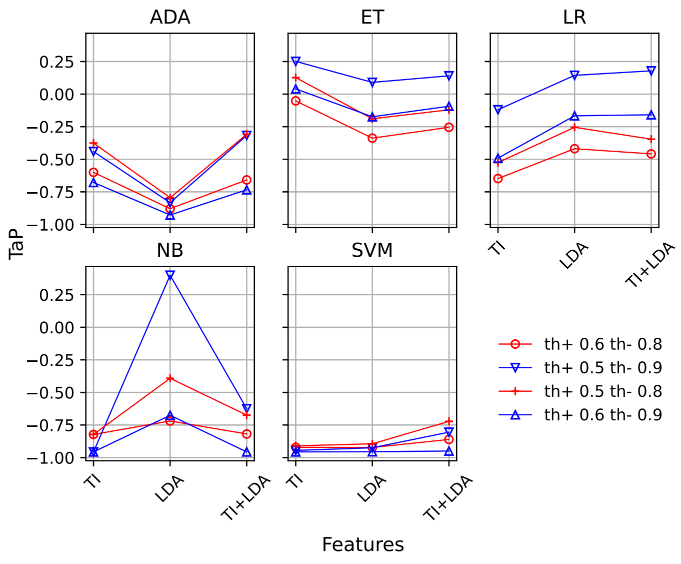


(b) TaP

Figure 6.4: $F1-latency$ (6.4(a)) and TaP (6.4(b)) for threshold early detection model using Instagram dataset. One graph is presented for each machine learning model. Different values for positive and negative thresholds are represented on each graph. Features are represented on the X-axis.



(a) $F_{latency}$



(b) TaP

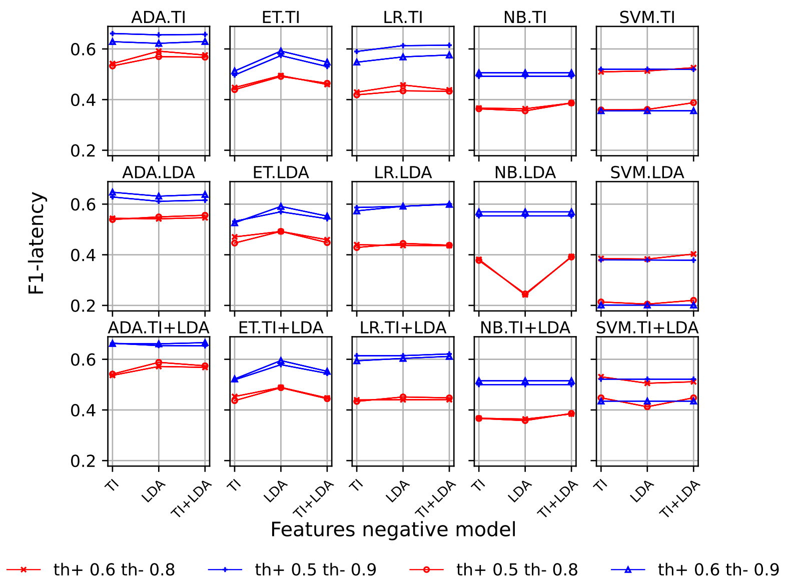
Figure 6.5: $F1 - latency$ (6.5(a)) and TaP (6.5(b)) for threshold early detection model using Vine dataset. One graph is presented for each machine learning model. Different values for positive and negative thresholds are represented on each graph. Features are represented on the X-axis.

features and thresholds are used for positive and negative models. Results for Instagram dataset using both $F_{latency}$ and TaP metrics are shown in Figure 6.6, and for Vine dataset in Figure 6.7. Each row of plots contains all the models with different features for positive model, while features for negative models are displayed as the x-axis for each of the plots. Within each combination of positive and negative sets of features for each model, four combinations of positive and negative thresholds are defined and displayed.

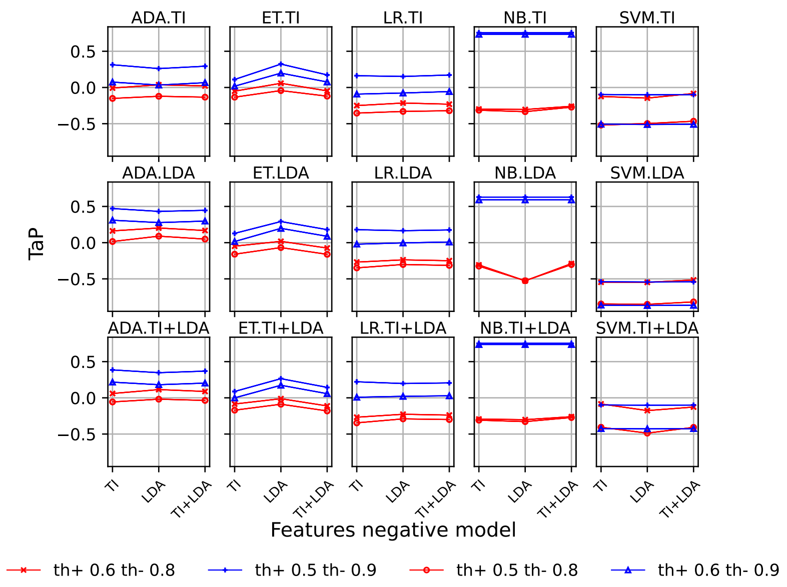
Generally, in the case of Instagram dataset, *Ada Boost*, *Extra Trees* and *Logistic Regression* obtain consistent results for both metrics also maintaining the behaviour explained in previous section where Threshold model was analysed. The principal change in performance depending on the metric used appears on model *Naïve Bayes* when a high negative penalty is introduced ($th^{-} = 0.9$), reaching 0.7561, this might be explained by the model taking late but correct predictions for non-cyberbullying cases which receive a different penalisation for the two metrics. Another generalisation that can be extracted is that the lower negative threshold value th^{-} (i.e. 0.8) obtain worst scores than whatever the positive threshold is. When comparing positive thresholds, *Ada Boost*, *Extra Tree* and *Logistic Regression* achieve higher values with lower positive threshold (i.e. 0.5) despite the features selection used on the model. One exception to this is the case of *Naïve Bayes* where both positive thresholds achieve really close scores a high negative thresholds is used.

For Vine dataset, even though performances are lower than the ones obtained for Instagram dataset it is interesting to see how feature combinations show bigger differences regarding the positive features used for the model. The more stable behaviour can be found on *Extra Trees* and *Logistic Regression* disregarding the set of features used both for positive an negative models. Both *Naïve Bayes* and *SVM* present, in general, a low performance for all combinations of positive and negative features. The exception to this behaviour can be found for *Naïve Bayes* model, where the use of only LDA as positive features achieves the best result, 0.399, for the dataset with TaP metric when a low positive threshold and a high negative threshold are used (i.e. $th^{+} = 0.5$ and $th^{-} = 0.9$)

Compared with previous results it can be seen that dual model outperforms fix model for both datasets in most of the presented combinations, while obtaining comparable and competitive values in the case of threshold model.

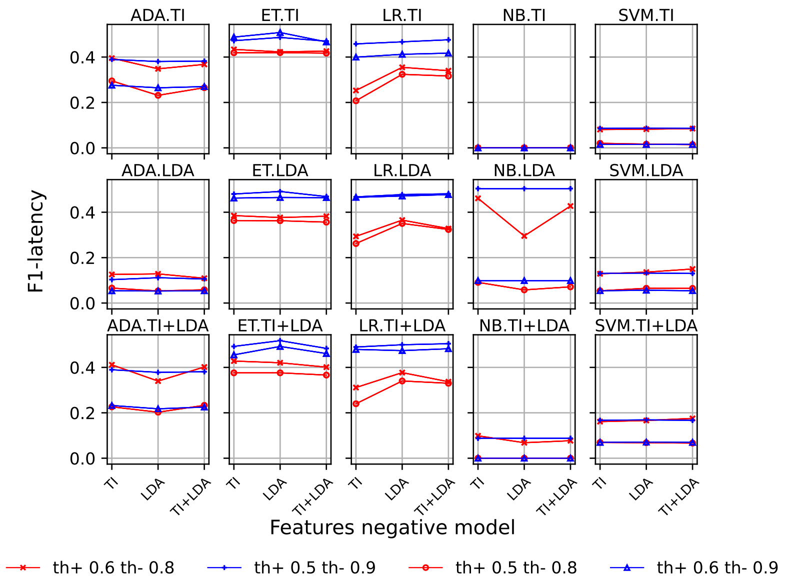


(a) $F_{latency}$

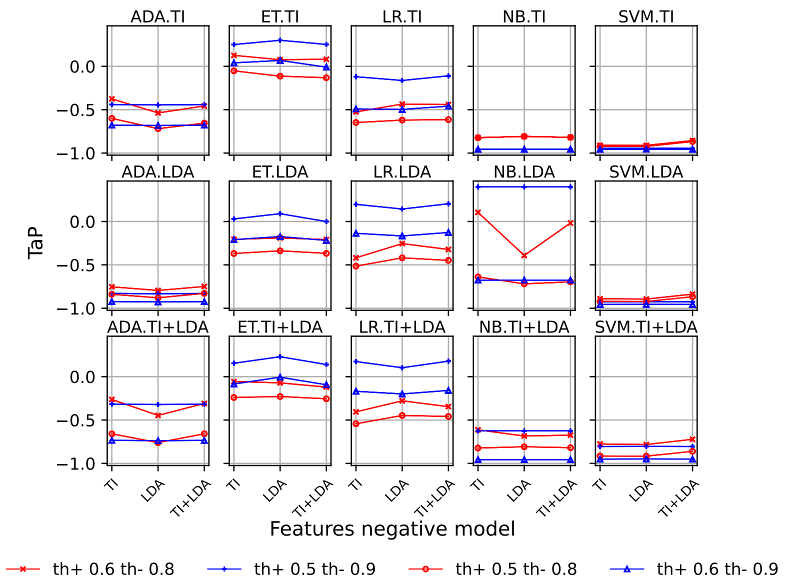


(b) TaP

Figure 6.6: $F1 - latency$ (6.6(a)) and TaP (6.6(b)) for dual early detection model using Instagram dataset. Columns correspond to machine learning models. Rows correspond to positive features. Negative features are represented on the X-axis. Different values for positive and negative thresholds are represented on each graph.



(a) $F_{latency}$



(b) TaP

Figure 6.7: $F1 - latency$ (6.7(a)) and TaP (6.7(b)) for dual early detection model using Vine dataset. Columns correspond to machine learning models. Rows correspond to positive features. Negative features are represented on the X-axis. Different values for positive and negative thresholds are represented on each graph.

6.3 Doc2Vec features

Doc2Vec features, as presented in Section 5.2.1, had been tested on the three previously presented models in order to evaluate its effect over performance. In the experiments, baseline features as sentiment and profane analysis were included in different combinations with syntactic and semantic features, such as Tf-if, LDA and Doc2Vec.

6.3.1 Fix model

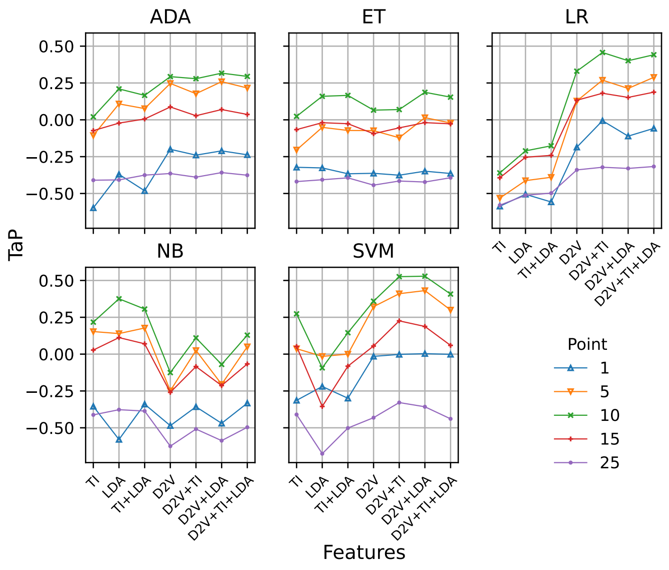
The first set of experiments evaluates the performance obtained when including Doc2Vec on fix early detection model. For the Instagram dataset, as displayed in Figure 6.8(a), the best baseline performance, 0.4127, is obtained when using LDA features for NB_{10}, or *Naïve Bayes* at point 10 model. Vine dataset, as shown in Figure 6.8(b), obtains its best baseline score, 0.3794, when combining LDA and TI features also for NB_{10} or *Naïve Bayes* at point 10 model.

The use of Doc2Vec features improves performance in general for all combinations and models, except in the case of *Naïve Bayes* model. The best performance for Instagram dataset, 0.5292, is obtained for SVM_{10} when using Doc2Vec combined with LDA, followed by LR_{10} with Doc2Vec and TI. Vine dataset reaches its best result, 0.5716, for just Doc2Vec features on SVM model, both at point 10, closely followed by SVM_5. The improvement in the performance obtained as a result of the inclusion of Doc2Vec features for both datasets is significant.

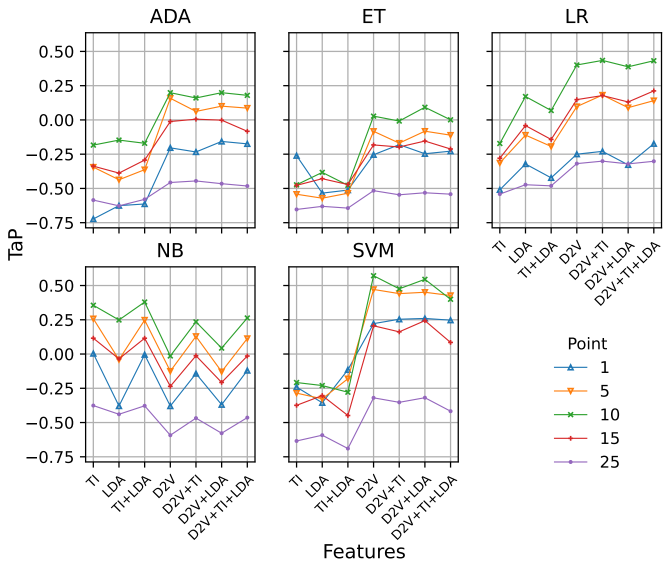
6.3.2 Fine-grain fix model

The fine-grain fix model consists on an extension of the fixed early detection model as an attempt to provide a finer level of granularity when deciding the stopping decision point. Instead of using the *items* of the *entity*, a subsection of the *item* will be used. In this case, the number of posts will be replaced by the accumulative number of words processed. As the proposed datasets have distinctive characteristics in terms of words per post, the purpose of this is to determine if the number of words provide a better option for fix model definition than simple post counting.

The experiments show results for Doc2Vec features alone and in combination with others, such as LDA and Tf-idf (TI), since the best results for fixed early detection models were obtained with these combinations in previous section.



(a) Instagram



(b) Vine

Figure 6.8: TaP for fixed early detection model at points $1, 5, 10, 15$ and 25 using Instagram (6.8(a)) and (6.8(b)) datasets. One graph is presented for each machine learning model: AdaBoost (ADA), Extra Trees (ET), Logistic Regression (LR), Naïve Bayes (NB) and Support-Vector Machine (SVM). Features are represented on the X-axis.Tf-idf (TI) and Latent Dirichlet Allocation (LDA) and Doc2Vec.

In the experiments shown in this section, values in the range between 1 and 100 will be used, in particular the following values were selected: 1, 3, 5, 10, 15, 25, 35, 50, 75 and 100. Results for Instagram and Vine datasets are shown in Figure 6.9.

There, it can be seen that performance increase exponentially on both datasets for the first points of decision, from words 1 to 10, and that it keeps increasing beyond that point at a slower pace, even showing some valleys in the case of Vine dataset. The maximum value for Instagram is located between 50 and 75 words, depending on the model, with the exception of *Naïve Bayes* where this value is found earlier on but below the scores of other models at that point. For Vine the maximum value is located between 35 and 50, except again in the case of NB where high values can be found close to 10 words. The general reduction in the number of words used to achieve the highest score reflects the smaller post size of the latter dataset.

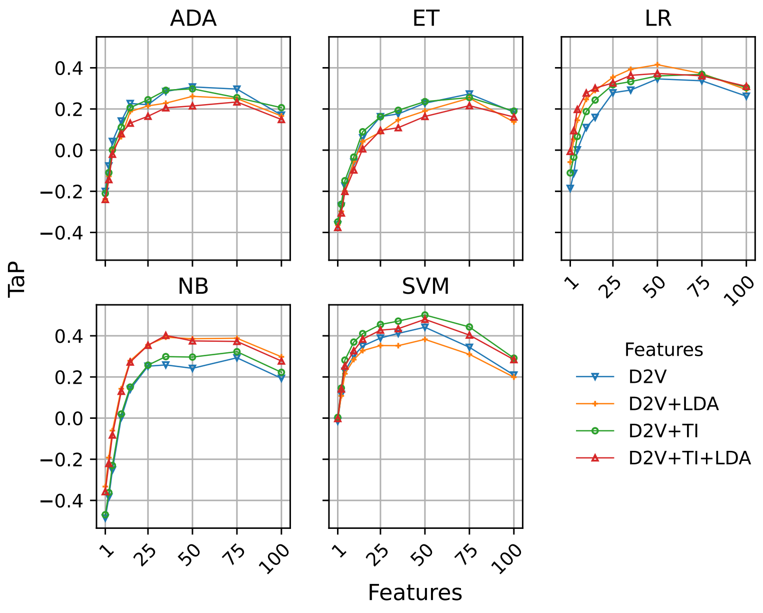
In most cases and for both datasets, the combination of different features explored do not provide significative differences in terms of performance. For example, in the case of Instagram dataset, as it can be observed in Figure 6.9(a), SVM model provides the best performance at 50 words point with Doc2Vec and LDA features, reaching 0.5012. When comparing the results with the ones obtained for the early detection fixed model on the previous section, which uses post counting instead of words for the definition of the stopping point, there is no significant difference with the best models in each case.

When analysing the results for Vine dataset, as shown in Figure 6.9(b), the best score, 0.5589, is achieved also with SVM model but at the 35 words point and using both Doc2Vec alone and Doc2Vec and LDA features combined. Still, and as in the case of Instagram dataset, there is no improvement when compared with the best results obtained by regular fix point model presented on the previous section.

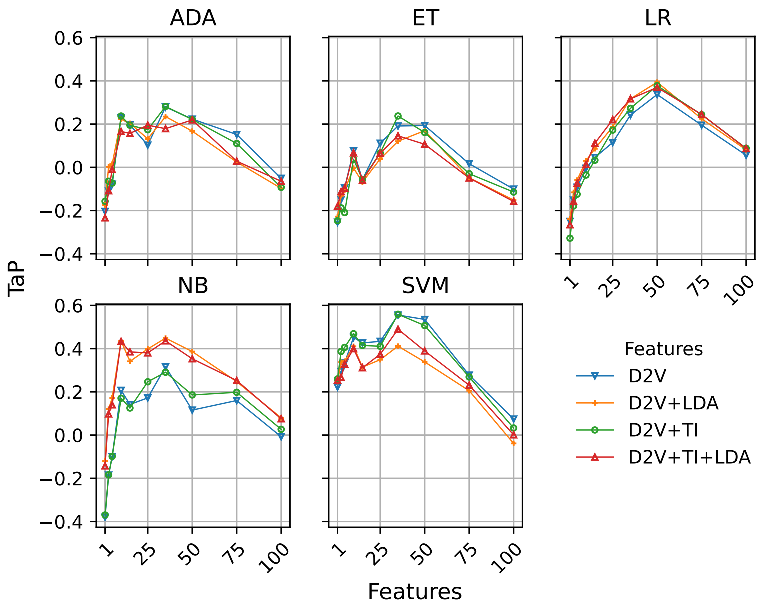
To summarise, with these results it can be concluded that the reduction from number of posts to number of words in the definition of early detection fix models provides an equivalent performance for both datasets. Therefore, for the remaining references to fix point methods number of posts will be used as a reference.

6.3.3 Threshold model

Next, the same experiment is repeated for threshold early detection model in order to study how the use of Doc2Vec features influences performance for this model. Results for Instagram and Vine datasets are shown in Figure 6.10. Since positive and negative thresholds must be defined and to simplify



(a) Instagram



(b) Vine

Figure 6.9: *TaP* for fine-grain fixed early detection model using Instagram (6.9(a)) and Vine (6.9(b)) datasets. One graph is presented for each machine learning model. Different features combinations are represented on each graph.

the analysis, as described in Section 6.2.2, only two low positive ($th^+ = 0.5$ and $th^+ = 0.6$) and two high negative ($th^- = 0.8$ and $th^- = 0.9$) thresholds are used, since they have provided good results for previous evaluations [9].

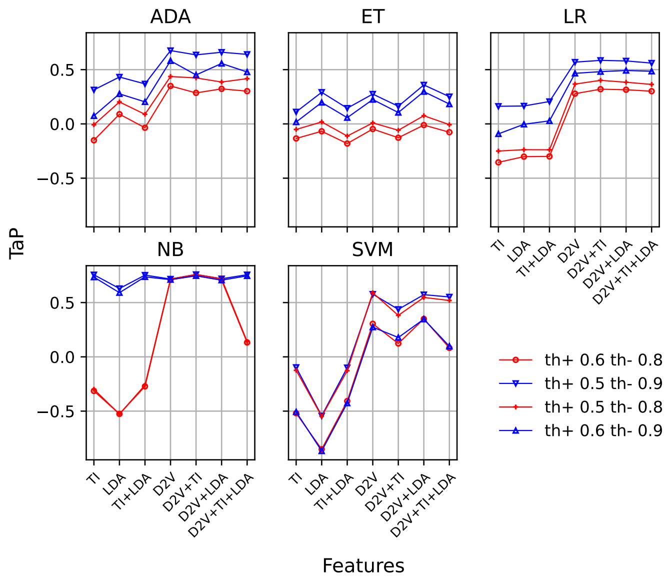
For both datasets a similar behaviour as for fix model is found in the figures, a general performance increase when Doc2Vec features are included, either by themselves or in combination with others. In the case of Instagram dataset (Figure 6.10(a)), the best baseline, 0.7561, was obtained when combining Tf-idf and LDA with *Naïve Bayes* and the following thresholds: $th^+ = 0.5$ and $th^- = 0.9$, but $th^+ = 0.6$ obtains a similar performance. For Vine it was also *Naïve Bayes* model, with a value of 0.399 when just LDA is used. Also the thresholds values are the same as for Instagram but with the difference that $th^+ = 0.6$ do not achieve a value close to the best performance.

In this model, when Doc2Vec features are included, the performance increases to a certain extent on all models but *Naïve Bayes.* Particularly, with this model on the Instagram dataset there is an slight improvement when Doc2Vec features are included except in the case where they are combined both with TI and LDA, and it affects only to $th^- = 0.8$, while on the Vine dataset a small increase is also present when $th^+ = 0.5$ is used with Doc2Vec. It should be noticed that in this last case, results for Doc2Vec alone are better than any combination of the rest of features. Also, the best case results in any combination for *Naïve Bayes* model are significantly lower than the results obtained by the other models with this combination of parameters. Specifically, on Instagram, *Naïve Bayes* obtain its best results with slight improvements when Doc2Vec features are included, but the rest of the models present important improvements with the inclusion of Doc2Vec, being the highest model *AB* using Doc2Vec with LDA features and thresholds of $th^+ = 0.5$ and $th^- = 0.9$. In the case of Vine results, the best model requires Doc2Vec features combined with LDA for *Ada Boost* model with $th^+ = 0.5$ and $th^- = 0.9$ reaching its highest score, 0.7097, which outperforms baseline results.

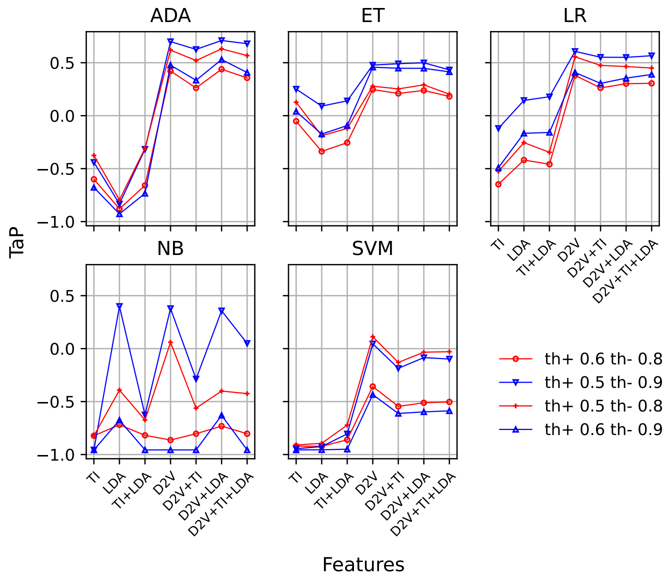
6.3.4 Dual model

Lastly, the behaviour of dual model with the inclusion of Doc2Vec features is examined in this section. Different values of threshold had been explored, using low values for positive and high for negative as reported on Sections 6.2.2 and 6.2.3.

Generally, the best results in this experiment are obtained with *Naïve Bayes* with all baseline features as negative features and Tf-idf with or without LDA as positive features for the Instagram dataset. On the other hand,



(a) Instagram



(b) Vine

Figure 6.10: *TaP* for threshold early detection model using Instagram (6.10(a)) and Vine (6.10(b)) datasets. One graph is presented for each machine learning model. Different values for positive and negative thresholds are represented on each graph. Features are represented on the X-axis.

for the Vine dataset, the best score is obtained by *Ada Boost*, again with just TI features as positive features and using just LDA as negative features.

Since best scores for positive models were obtained when Doc2Vec features were used, results presented will be limited to those combinations. Figures 6.11 and 6.12 contain the performance of machine learning results with different combinations of features. As presented in Dual model results analysis, each column represent the machine learning algorithm being evaluated while positive features are displayed for rows and negative features as the x-axis of each graph. Also, as different thresholds had been studied for both positive and negative (i.e. $0.5-0.6$ and $0.8-0.9$ respectively), multiple lines represents its values.

When results for Instagram dataset are examined (Figure 6.11) it can be observed that best performing models are *Naïve Bayes*, 0.7585, followed by *Ada Boost* and that a high negative threshold ($th^{-} = 0.9$) reach the better values on most cases.

In the case of Vine dataset, results are shown with the same structure in Figure 6.12 and where a smaller impact of newly introduced features (i.e. Doc2Vec) both for positive and negative cases. There is only two cases where a noticeable improvement can be spotted, *Logistic Regression* and *SVM* when Doc2Vec are included as part of the negative set of features. Even on those cases higher differences are found only when the lower negative threshold is used (i.e. $th^{-} = 0.8$). The behaviour of the rest of models is mainly stable and *Ada Boost* and *Extra Trees*, disregarding the threshold values, even achieve good results for all combinations overall. Best results for this dataset is obtained by *Ada Boost*, 0.7168, when combining Doc2Vec and LDA as positive features and using all features as negative. In this case, thresholds used were $th+ = 0.5$ and $th^{-} = 0.9$ and its results significantly outperforms baseline dual model for this dataset.

To summarize, best results when comparing between the best baseline score obtained and the result when Doc2Vec features were include, show that the inclusion of these features generally improves the models when using only baseline features.This can be observed in Table 6.3, where both best values are included along with the improvement shown.

In view of these results and conclusions, for the remaining experiments, threshold early detection model will be used instead of dual model, as the configuration is simpler and its performance results can be considered equivalent.

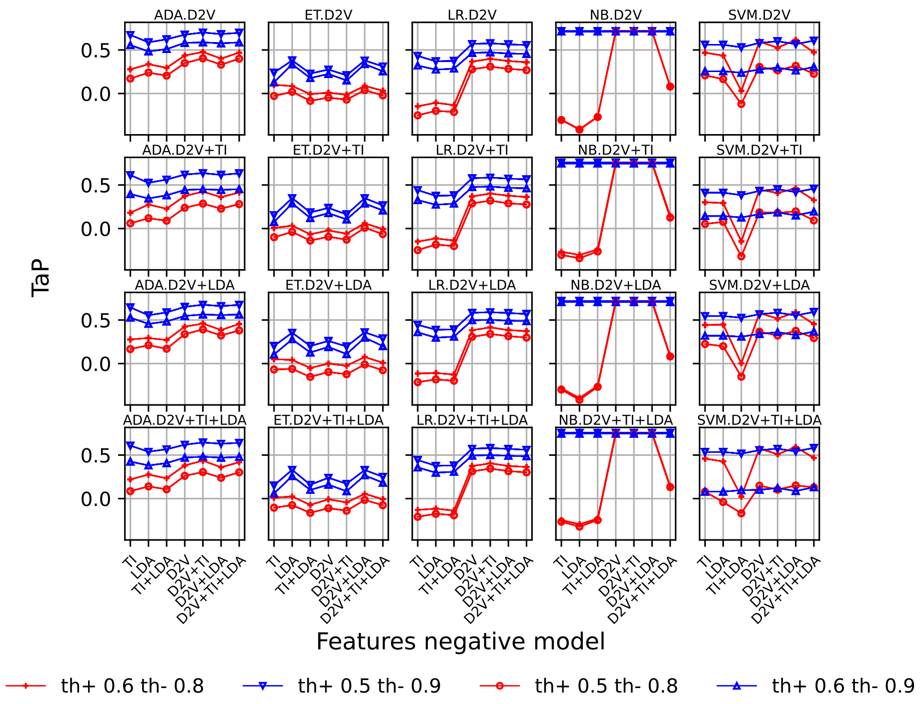


Figure 6.11: *TaP* for dual early detection model using Instagram dataset. Columns correspond to machine learning models. Rows correspond to positive features. Negative features are represented on the X-axis. Different values for positive and negative thresholds are represented on each graph.

6.4 Multiple Instance Learning

This section approaches the inclusion of Multiple Instance Learning (MIL) features, as introduced in Section 5.2.2, will be explored in order to determine if they could improve early detection cyberbullying models.

In this sense, in order to include MIL features to the presented early detection methods a bag representation of the posts used for the prediction will be added. The features from the items (i.e. posts) in the bag will be aggregated using the following functions: minimum, maximum, average between minimum and maximum, arithmetic mean and median. Also, in order to allow a better comparison with previous obtained results, an extra aggregation function called *None* is included as a representation of the early detection model without the use of MIL features.

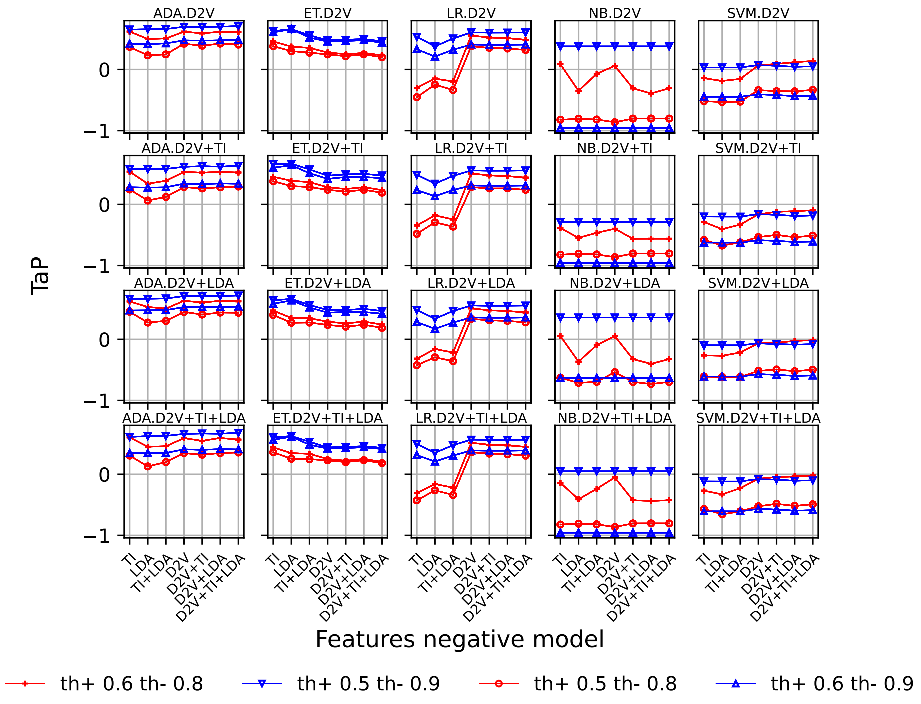


Figure 6.12: TaP for dual early detection model using Vine dataset. Columns correspond to machine learning models. Rows correspond to positive features. Negative features are represented on the X-axis. Different values for positive and negative thresholds are represented on each graph.

6.4.1 Fix model

Results for fix model experiments are shown in Figure 6.13 for the Instagram and Vine datasets, featuring combination of: profane words, sentiment analysis, Tf-idf, LDA and Doc2Vec features.

For the Instagram dataset, the use of MIL, decreases model performance in many cases. Although it is not true for all combinations as *Naïve Bayes* and *SVM* show some slight improvements for some models at the higher points. Actually, *NB* obtains its best score using MIL at point 10 when using *median* as aggregation function. Despite this, there is no difference when compared with the fixed model with *None* function, that is, without the use of MIL, and it achieves significantly worst results than threshold and dual.

On the contrary, the results for Vine dataset, display a significant change with MIL models, particularly for *Ada Boost* and *Extra Tree* models where

Table 6.3: *TaP* summary results for early detection models.

	Instagram			Vine		
Model	Baseline	Doc2Vec	%	Baseline	Doc2Vec	%
Fixed	0.3755 (NB)	0.529 (SVM)	40.8%	0.3794 (NB)	0.5716 (SMV)	50.6%
Threshold	0.7561 (NB)	0.7585 (NB)	0.3%	0.399 (NB)	0.7097 (ADA)	77.8%
Dual	0.7561 (NB)	0.7585 (NB)	0.3%	0.399 (NB)	0.7167 (ADA)	91.5%

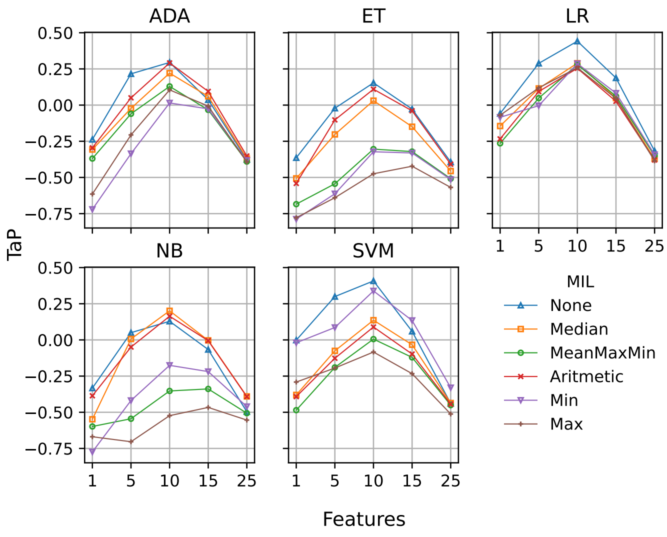
major improvements are shown. *Ada Boost* obtains the best score, 0.7926, using the *arithmetic mean* at point 10. This significantly outperforms the results obtained without MIL for fixed, threshold and dual models.

This relevant performance improvement with the inclusion of multiple instance learning (MIL) can be explained due to the reduced post size in this network and the lower use of standard English, circumstances that specially profits from this technique. As each post of this social network is composed on average of around 5, the information provided to the model and that can be extracted by the semantic an syntactic features is very limited. Nonetheless, the use of simple aggregation function, like the ones provided in MIL, for a certain amount of posts allow to improve the information provided by combining the information from different posts better than the simple concatenation of them. Also, as Instagram posts are larger and standard English is usually found, the presented features are able to extract enough information form the post itself with render the aggregation functions less effective.

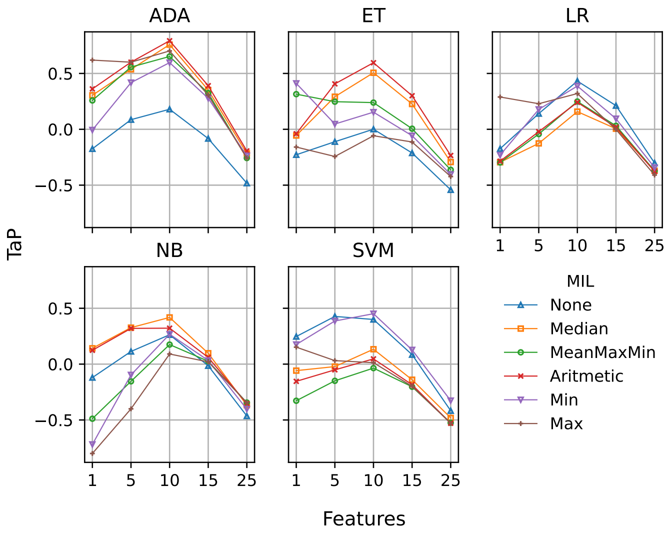
For the sake of completeness experiments with different sampling alternatives had been performed instead of using all posts in each case. Specifically, the first 10 posts, the last 10 posts and 10 random posts from the session had been selected, but without noticeable improvements when using fixed model with MIL as early detection model.

6.4.2 Threshold model

Also, the impact of threshold selection on model performance with the inclusion of MIL has been tested. Presented experiments which results are displayed in Figure 6.14 for the Instagram and Vine datasets are limited to *Ada Boost* and *Extra Trees* models, since they obtained the best results in previous observations. Likewise, and as in previous MIL experiments, all features had been included. In these figures and in order to display the full range of values obtained with the introduction of MIL features, the whole range of positive and negative thresholds are used: 0.5, 0.6, 0.7, 0.8 and 0.9. Each



(a) Instagram



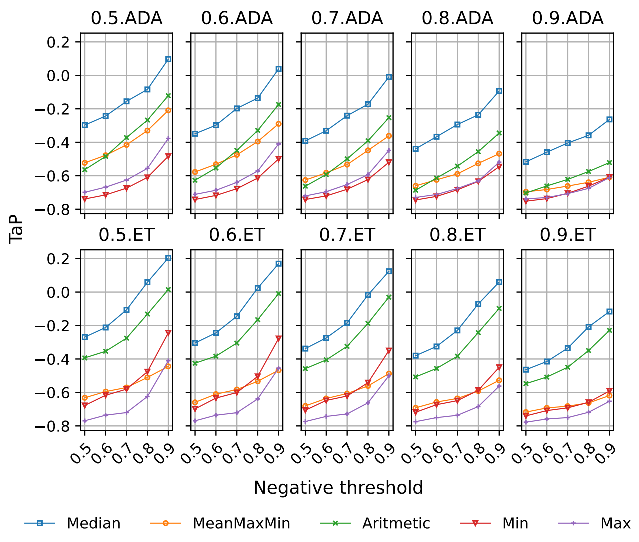
(b) Vine

Figure 6.13: TaP for fixed early detection model based using MIL for Instagram (6.13(a)) and Vine (6.13(b)) datasets. One graph is presented for each machine learning model. The X-axis represents the points where the fixed model produces its decision. Different aggregation functions are represented on each graph: no MIL (None), minimum (Min), maximum (Max), average maximum and minimum (MeanMaxMin), arithmetic mean (Arithmetic) and median (Median).

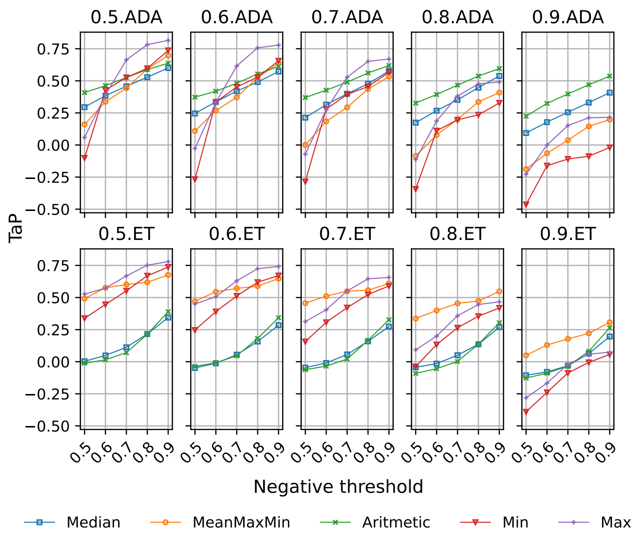
row represents a different ML model, being the upper one *Ada Boost* (*AB*) and the lower one *Extra Trees* (*ET*). Within the row, each plot represents a different positive threshold, while negative thresholds are on the x-axis of each plot, and aggregation functions are shown as different line types.

For Instagram dataset (Figure 6.14(a)), and as expected due to the results obtained previously for threshold models, the combination of low positive and high negative threshold values improves performance. Also, for MIL, *median* aggregation function provides consistently the best results for all combinations, followed by arithmetic mean, specially for *Extra Trees* models, which reaches 0.2035 when using $th^+ = 0.5$ and $th^- = 0.9$.

On the other hand and regarding Vine dataset (Figure 6.14(b)), *Extra Trees* provides, in general, a slightly worst performance than the results obtained by *Ada Boost* model. As in the case of Instagram, the increase of negative thresholds improves the performance and in regard to positive ones, the best results are obtained with 0.7 and 0.8. Also, as related for the previous MIL experiments, this provides a meaningful change regarding the standard threshold model results, as the aggregation of the information from multiple posts increases the class probability for cyberbullying cases detected. In terms of aggregation functions, *maximum* obtains the best results for both models when low positive thresholds are used, and this concurs with the best overall results for both models. When higher positive threshold values are used, *arithmetic mean* consistently provides the best results for *Ada Boost* model, while *MeanMaxMin* does the same for *Extra Trees* model. In fact, best score, 0.8149, is obtained by *Ada Boost* with *maximum* aggregation function for a positive threshold $th^+ = 0.5$ and negative threshold $th^- = 0.9$. This result, significantly improves the best performance obtained by fixed, threshold and dual models displayed in previous sections.



(a) Instagram



(b) Vine

Figure 6.14: TaP for threshold early detection model based using MIL for Instagram (6.14(a)) and Vine (6.14(b)) datasets. Rows correspond to machine learning models: ADA and ET. Columns correspond to positive thresholds. The X-axis represents negative thresholds. Different aggregation functions are represented on each graph:minimum (Min), maximum (Max), average maximum and minimum (MeanMaxMin), arithmetic mean (Arithmetic) and median (Median).

Chapter 7

Conclusions and future work

This chapter contains a summary of the conclusions extracted in previous sections, as well as some general ideas of the early detection problem and the performance measure of the methods applied. Also, a sight of future work is described next for different lines of study within the topic covered in this research.

7.1 Conclusions

The early detection problem has been defined and posses as an urgent matter in many fields, in this case, specifically, to cybersecurity. In this particular area, different segments had been identified, from systems or networks security to the protection of the users.

In the first case, it is clear that an early stop of any threat against the infrastructure will minimise the costs of operation and reduce the impact on the business or organization that they are giving support to. This early stop can protect or avoid damages to data or infrastructure and shorten downtime.

In the latter, both depression and cyberbullying have a definite relation with the duration of the action, in terms of the risk that it posses or the fact that they are defined by a repetition in time. This last case, more relevant in the case of cyberbullying, was treated in more detail in Chapters 5 and 6 where specific models and methods are defined in order to capture the time relation or to expedite the detection.

The approach to early detection was to first formally describe the early detection problem, then to identify the phases or actions that come into work when creating a detection system, to finally define, if needed, modifications to improve the outcome. Two key points were identified: metrics and methods. The first and essential aspect was to establish a form of measure for this kind of system. Also, appropriate methods had been applied to the early detection problem to improve the performance in terms of the predefined metrics. Although these two points are essential for this kind of problem there is a related matter that include both of them. That is the methodology used for the evaluation, and to do so, two different approaches had been taken into account: batch and streaming. When using batch evaluation, a percentage of the total amount of items of each entity are included in the evaluation, whereas when streaming is used, the same number of items are used for all entities being evaluated.

In terms of metrics definition, the starting point was the previously defined time aware metrics, among which *Early Risk Detection Error* ($ERDE$) and $F_{latency}$ were studied. As the outcome of this analysis, some problems arise such as dataset dependency, inter dataset results analysis or difficulty of interpretation. To deal with this, several metrics were defined to overcome the detected problems, first of them was a modification of $ERDE$ called $NormalizedERDE$ which by means of a normalization of $ERDE$ result based on dataset values, tried to allow inter dataset analysis of results. Next, a new time-aware metric was defined, following the principles of definition of $ERDE$ but using classic non time aware metrics as reference. The result was *Time aware Precision* (TaP) which was defined to be problem dependent not dataset dependent and as it has as a reference well know metrics it should ease the study of the output. Although TaP solved some of the detected problems it had still mainly some interpretation issues and to deal with that a new metric was defined: *Time aware F-score* (TaF), following the same directives as for TaP was defined to achieve an output similar to $F - score$ but to be time aware. In summary the objective was to achieve a metric which worked independently from the specific dataset.

As for detection methods in the early detection problem, the focus was on the specific aspect of cyberbullying in social media networks.

In particular, Vine and Instagram datasets were used to validate proposed features and models. The three proposed models are: fix point, threshold and dual. For the first one, different levels of granularity were tested but no significant differences were found when decreasing the grain from post to word count.

Concerning the features used, two approaches had been shown and both a specialized and generic approach to social media networks had been taken. To achieve so, information from user profiles, and particular to each social media network, has been included or disregarded to demonstrate particular or generic performance. Firstly, the use of particular sets of features from specific social media networks, with or without the use of text similarities and time features, led to show how the threshold model was able to improve the baseline by 26% and that the dual model was able to further increase this difference up to 42% both by using Extra Tree as base model. Then, the inclusion of the proposed features along with the ones extracted from the social network and used as baseline, improved both threshold and dual models. As a summary, in this case, the dual model provides the best performance for the early detection when using all features for the identification of positive cases. And simpler features, leaving out the new time and textual features, with low thresholds to produce early detection for the negative models.

In terms of generalizing the outcome to different social networks, the next results were obtained without the use of particular features extracted from the platforms and specific peculiarities from the user profiles. Instead, only information extracted from users' comments as it is the case of Doc2Vec and MIL features were used. When applying the former it showed improvements on all early detection models and better results for smaller post size and lower use of English language was achieved for the latter.

Finally and specifically when dealing with the network attack dataset presented in Chapter 4, something must be noticed. The kind of attack presented is an OS Scan Attack, where the aggressor is trying to extract information from a machine connected to the network (e.g. which ports are open, what services are running or what is the OS of the device). When studying network traffic it is quite common to proceed as shown in Section 4.1 by creating flows. In this case those flows would contain as little as two packets for attack traffic and that will affect not just the detection but the evaluation of the system. To avoid that it could be necessary to perform a higher hierarchy aggregation, for example as shown in [102]. This point, joined with the streaming nature of traffic evaluation, where there is no complete information about the entities, also reflects the fact that the penalization should be problem related and not based on the specific dataset and that the penalisation introduced should be according to the elements required to make that decision.

In summary, the results obtained are:

- Assess the importance of early detection, specially in different facets of cybersecurity.
- Analysis of evaluation methodology, both for batches and streaming.
- Proposed more realistic evaluation based on streaming and pure time evaluation.
- Presented three models based on ML learning basic models, that could be combined to improve the early detection:

  - Fix model (with different levels of granularity)
  - Threshold model
  - Dual model

- Selected a set of features based on user comments and included specific ones to represent time related aspects:
  - Doc2Vec
  - Multiple Instance Learning

7.2 Future Work

Several points are key to the development of early detection systems, starting with the measure of systems performance, it has been presented both batches and streaming evaluation but a more realistic approach will be tackled and it is pure time based evaluation. In this case, an specific amount of items will be used in the evaluation at each time point, defined by the timestamp of each one of the items. Only items in the defined period of time will be used on the evaluation, this will give the system the most real image of a real world situation in terms of online evaluation.

Also, even considering that the early detection problem can be applied in multiple and different environments, the proposed formalization, metrics and methods, expect to provide a generic approach to this problem. Because of that other points where this research could be expanded is the fields of application. Even if diverse facets of cybersecurity had been engaged in this thesis such as the systems and humans resources, there are more fields that could benefit from an early detection approach. Firstly, it would be interesting to delve into the use of different datasets of diverse attacks to compare the performance of models. Likewise, unrelated fields such as medicine of emergencies/disasters prevention are some of the examples where it would be interesting to develop further research.

In terms of the presented models, several ways could broaden the presented results. First, regarding the definition of machine learning models, heterogeneous combinations of different base machine learning models could be used. It would be also interesting to study how Deep Learning performs in terms of time aware metrics an to analyse its capabilities in terms of time aware feature extraction.

Finally, in relation to the extracted information from users, the expansion of features used to feed the model for training could be improved by increasing the study of the extracted features and analysing the results. Feature extraction would be focused on user comments, as described on the section 5.2, as this would keep the model site agnostic and also because these concentrate most of the information for the early detection of cyberbullying. Also, Multiple Instance Learning could be further expanded by using more advanced instance-space MIL models. Among them it could be used Expectation-Maximization Diverse Density (EMDD), or even bag-space and embedded-space model could be considered. Amid which Normalized Set Kernel (NST-SMV) or Multiple Instance Learning via Embedded Instance Selection (MILES) could be used.

Bibliography

[1] "Cybersecurity — meaning, definition in cambridge english dictionary." Available at: `https://dictionary.cambridge.org/dictionary/english/cybersecurity`.

[2] ISO, "Information security, cybersecurity and privacy protection 27001," 2017.

[3] E. M. Hutchins, M. J. Cloppert, and R. M. Amin, "Intelligence-driven computer network defense informed by analysis of adversary campaigns and intrusion kill chains," *Proceedings of the 6th International Conference on Information Warfare and Security*, vol. 1, pp. 113–125, 2011.

[4] "Cyber bullying: common types of bullying 2019 — statista."

[5] K. V. Royen, K. Poels, W. Daelemans, and H. Vandebosch, "Automatic monitoring of cyberbullying on social networking sites: From technological feasibility to desirability," *Telematics and Informatics*, vol. 32, pp. 89–97, 2 2015.

[6] M. Lopez-Vizcaino, F. J. Novoa, D. Fernandez, V. Carneiro, and F. Cacheda, "Early intrusion detection for os scan attacks," *2019 IEEE 18th International Symposium on Network Computing and Applications, NCA 2019*, 9 2019.

[7] M. F. Lopez-Vizcaino, F. J. Novoa, D. Fernandez, and F. Cacheda, "Measuring early detection of anomalies," *IEEE Access*, vol. 10, pp. 127695–127707, 2022.

[8] M. López-Vizcaíno, F. J. Novoa, D. Iglesias, and F. Cacheda, "Time aware f-score for cybersecurity early detection evaluation," *Preprint*, 2023.

[9] M. F. López-Vizcaíno, F. J. Nóvoa, V. Carneiro, and F. Cacheda, "Early detection of cyberbullying on social media networks," *Future Generation Computer Systems*, vol. 118, pp. 219–229, 5 2021.

[10] M. López-Vizcaíno, F. J. Novoa, T. Artieres, and F. Cacheda, "Site agnostic approach to early detection of cyberbullying on social media networks," vol. Submitted, 2022.

[11] D. E. Losada and F. Crestani, "A test collection for research on depression and language use," pp. 28–39, Springer, Cham, 2016.

[12] D. E. Losada, F. Crestani, and J. Parapar, "erisk 2020: Self-harm and depression challenges," pp. 557–563, Springer International Publishing, 2020.

[13] F. Sadeque, D. Xu, and S. Bethard, "Measuring the latency of depression detection in social media," *WSDM 2018 - Proceedings of the 11th ACM International Conference on Web Search and Data Mining*, vol. 2018-Febuary, pp. 495–503, 2 2018.

[14] N. Chinchor, "Muc-4 evaluation metrics," pp. 22–29. Available at: http://www.aclweb.org/anthology-new/M/M92/M92-1002.pdf.

[15] N. S. Samghabadi, A. P. L. Monroy, and T. Solorio, "Detecting early signs of cyberbullying in social media," pp. 144–149, European Language Resources Association (ELRA), 5 2020.

[16] X. Zhou, A. Jain, V. V. Phoha, and R. Zafarani, "Fake news early detection: A theory-driven model," *Digital Threats: Research and Practice*, vol. 1, 6 2020.

[17] Z. Zhao, P. Resnick, and Q. Mei, "Enquiring minds: Early detection of rumors in social media from enquiry posts," pp. 1395–1405, International World Wide Web Conferences Steering Committee, 2015.

[18] S. N. Narayanan, A. Ganesan, K. Joshi, T. Oates, A. Joshi, and T. Finin, "Early detection of cybersecurity threats using collaborative cognition," *Proceedings - 4th IEEE International Conference on Collaboration and Internet Computing, CIC 2018*, pp. 354–363, 11 2018.

[19] M. Pivarníková, P. Sokol, and T. Bajtoš, "Early-stage detection of cyber attacks," *Information 2020, Vol. 11, Page 560*, vol. 11, p. 560, 11 2020.

[20] S. Hosokawa, M. Enomoto, K. Matsumoto, and M. Takahashi, "A prototype of cyber incident diagnosis mechanism for cyber attacks early recognition support system," *2015 10th Asian Control Conference: Emerging Control Techniques for a Sustainable World, ASCC 2015*, 9 2015.

[21] H. K. Kalutarage, C. Lee, S. A. Shaikh, and F. L. B. Sung, "Towards an early warning system for network attacks using bayesian inference," *Proceedings - 2nd IEEE International Conference on Cyber Security and Cloud Computing, CSCloud 2015 - IEEE International Symposium of Smart Cloud, IEEE SSC 2015*, pp. 399–404, 1 2016.

[22] C. C. Zou, L. Gao, W. Gong, and D. Towsley, "Monitoring and early warning for internet worms," *Proceedings of the 10th ACM conference on Computer and communication security - CCS '03*, 2003.

[23] K. Bsufka, O. Kroll-Peters, and S. Albayrak, "Intelligent network-based early warning systems," vol. 4347 LNCS, pp. 103–111, Springer, Berlin, Heidelberg, 2006.

[24] C. Xu, H. Lin, Y. Wu, X. Guo, and W. Lin, "An sdnfv-based ddos defense technology for smart cities," *IEEE Access*, vol. 7, pp. 137856–137874, 2019.

[25] A. Privalov, V. Lukicheva, I. Kotenko, and I. Saenko, "Method of early detection of cyber-attacks on telecommunication networks based on traffic analysis by extreme filtering," *Energies 2019, Vol. 12, Page 4768*, vol. 12, p. 4768, 12 2019.

[26] Statista, "Iot connected devices worldwide 2019-2030 — statista." Available at `https://www.statista.com/statistics/1183457/iot-connected-devices-worldwide/`.

[27] A. S. Zaidoun, *Computer Science Security: Concepts and Tools.* Wiley, 2022.

[28] M. K. Asif, T. A. Khan, T. A. Taj, U. Naeem, and S. Yakoob, "Network intrusion detection and its strategic importance," *BEIAC 2013 - 2013 IEEE Business Engineering and Industrial Applications Colloquium*, pp. 140–144, 2013.

[29] P. García-Teodoro, J. Díaz-Verdejo, G. Maciá-Fernández, and E. Vázquez, "Anomaly-based network intrusion detection: Techniques, systems and challenges," *Computers & Security*, vol. 28, pp. 18–28, 2 2009. Combination of parameters better than individual values.

[30] A. L. Buczak and E. Guven, "A survey of data mining and machine learning methods for cyber security intrusion detection," *IEEE Communications Surveys and Tutorials*, vol. 18, pp. 1153–1176, 2016.

[31] S. S. Chawathe, "Monitoring iot networks for botnet activity," *NCA 2018 - 2018 IEEE 17th International Symposium on Network Computing and Applications*, 11 2018.

[32] S. Mukkamala, A. H. Sung, and A. Abraham, "Intrusion detection using an ensemble of intelligent paradigms," *Journal of Network and Computer Applications*, vol. 28, pp. 167–182, 4 2005.

[33] M. Yousefi-Azar, V. Varadharajan, L. Hamey, and U. Tupakula, "Autoencoder-based feature learning for cyber security applications," *Proceedings of the International Joint Conference on Neural Networks*, vol. 2017-May, pp. 3854–3861, 6 2017.

[34] Y. Mirsky, T. Doitshman, Y. Elovici, and A. Shabtai, "Kitsune: An ensemble of autoencoders for online network intrusion detection," 2 2018.

[35] M. Ahmed, A. N. Mahmood, and J. Hu, "A survey of network anomaly detection techniques," *Journal of Network and Computer Applications*, vol. 60, pp. 19–31, 1 2016.

[36] S. Eltanbouly, M. Bashendy, N. Alnaimi, Z. Chkirbene, and A. Erbad, "Machine learning techniques for network anomaly detection: A survey," *2020 IEEE International Conference on Informatics, IoT, and Enabling Technologies, ICIoT 2020*, pp. 156–162, 2 2020.

[37] N. Görnitz, M. Kloft, K. Rieck, and U. Brefeld, "Toward supervised anomaly detection," *Journal of Artificial Intelligence Research*, vol. 46, pp. 235–262, 2 2013.

[38] G. Prashanth, V. Prashanth, P. Jayashree, and N. Srinivasan, "Using random forests for network-based anomaly detection at active routers," *Proceedings of ICSCN 2008 - International Conference on Signal Processing Communications and Networking*, pp. 93–96, 2008.

[39] M. C. Belavagi and B. Muniyal, "Performance evaluation of supervised machine learning algorithms for intrusion detection," *Procedia Computer Science*, vol. 89, pp. 117–123, 1 2016.

[40] D. Bhamare, T. Salman, M. Samaka, A. Erbad, and R. Jain, "Feasibility of supervised machine learning for cloud security," *ICISS 2016 - 2016 International Conference on Information Science and Security*, 3 2017.

[41] W. Chen, F. Kong, F. Mei, G. Yuan, and B. Li, "A novel unsupervised anomaly detection approach for intrusion detection system," pp. 69–73, 2017.

[42] L. Yang, A. Moubayed, A. Shami, P. Heidari, A. Boukhtouta, A. Larabi, R. Brunner, S. Preda, and D. Migault, "Multi-perspective content delivery networks security framework using optimized unsupervised anomaly detection," *IEEE Transactions on Network and Service Management*, vol. 19, pp. 686–705, 3 2022.

[43] M. Xie, J. Hu, S. Han, and H. H. Chen, "Scalable hypergrid k-nn-based online anomaly detection in wireless sensor networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 24, pp. 1661–1670, 2013.

[44] J. Ran, Y. Ji, and B. Tang, "A semi-supervised learning approach to ieee 802.11 network anomaly detection," *IEEE Vehicular Technology Conference*, vol. 2019-April, 4 2019.

[45] S. Behal, K. Kumar, and M. Sachdeva, "D-face: An anomaly based distributed approach for early detection of ddos attacks and flash events," *Journal of Network and Computer Applications*, vol. 111, pp. 49–63, 6 2018.

[46] K. S. Sahoo, D. Puthal, M. Tiwary, J. J. Rodrigues, B. Sahoo, and R. Dash, "An early detection of low rate ddos attack to sdn based data center networks using information distance metrics," *Future Generation Computer Systems*, vol. 89, pp. 685–697, 12 2018.

[47] R. Amin, M. Reisslein, and N. Shah, "Hybrid sdn networks: A survey of existing approaches," *IEEE Communications Surveys and Tutorials*, vol. 20, pp. 3259–3306, 10 2018.

[48] A. Kumar and T. J. Lim, "Edima: Early detection of iot malware network activity using machine learning techniques," *IEEE 5th World Forum on Internet of Things, WF-IoT 2019 - Conference Proceedings*, pp. 289–294, 4 2019.

[49] M. D. Mauro, G. Galatro, G. Fortino, and A. Liotta, "Supervised feature selection techniques in network intrusion detection: A critical review," *Engineering Applications of Artificial Intelligence*, vol. 101, p. 104216, 5 2021.

[50] M. U. Nisa and K. Kifayat, "Detection of slow port scanning attacks," *1st Annual International Conference on Cyber Warfare and Security, ICCWS 2020 - Proceedings*, 10 2020.

[51] A. Shiravi, H. Shiravi, M. Tavallaee, and A. A. Ghorbani, "Toward developing a systematic approach to generate benchmark datasets for intrusion detection," *Computers and Security*, vol. 31, pp. 357–374, 2012.

[52] S. Dadkhah, H. Mahdikhani, P. K. Danso, A. Zohourian, K. A. Truong, and A. A. Ghorbani, "Towards the development of a realistic multidimensional iot profiling dataset," *2022 19th Annual International Conference on Privacy, Security and Trust, PST 2022*, 2022.

[53] "Canadian institute for cybersecurity — unb." `https://www.unb.ca/cic/`.

[54] "Pcap capture file format." Available at `https://www.ietf.org/archive/id/draft-gharris-opsawg-pcap-01.html`.

[55] K. Dinakar, R. Reichart, and H. Lieberman, "Modeling the detection of textual cyberbullying," 2011. Use @techreport for APA citation.

[56] H. Zhong, H. Li, A. C. Squicciarini, S. M. Rajtmajer, C. Griffin, D. J. Miller, and C. Caragea, "Content-driven detection of cyberbullying on the instagram social network.," vol. 16, pp. 3952–3958, 2016.

[57] Q. Le and T. Mikolov, "Distributed representations of sentences and documents," vol. 32, pp. 1188–1196, PMLR, 7 2014.

[58] J. Blackburn, S. Binghamton, A. Vakali, D. Chatzakou, I. Leontiadis, E. D. Cristofaro, G. Stringhini, and N. Kourtellis, "Detecting cyberbullying and cyberaggression in social media," *ACM Transactions on the Web*, vol. 13, p. 51, 2019.

[59] Y. Chen, Y. Zhou, S. Zhu, and H. Xu, "Detecting offensive language in social media to protect adolescent online safety," *Proceedings - 2012 ASE/IEEE International Conference on Privacy, Security, Risk and Trust and 2012 ASE/IEEE International Conference on Social Computing, SocialCom/PASSAT 2012*, pp. 71–80, 2012.

[60] S. Alsafari, S. Sadaoui, and M. Mouhoub, "Hate and offensive speech detection on arabic social media," *Online Social Networks and Media*, vol. 19, p. 100096, 9 2020.

[61] M. A. Al-Garadi, K. D. Varathan, and S. D. Ravana, "Cybercrime detection in online communications: The experimental case of cyberbullying detection in the twitter network," *Computers in Human Behavior*, vol. 63, pp. 433–443, 10 2016.

[62] K. Reynolds, A. Kontostathis, and L. Edwards, "Using machine learning to detect cyberbullying," *Proceedings - 10th International Conference on Machine Learning and Applications, ICMLA 2011*, vol. 2, pp. 241–244, 2011.

[63] V. Nahar, X. Li, and C. Pang, "An effective approach for cyberbullying detection," *Communications in information science and management engineering*, vol. 3, p. 238, 2013.

[64] C. V. Hee, E. Lefever, B. Verhoeven, J. Mennes, B. Desmet, G. D. Pauw, W. Daelemans, and V. Hoste, "Detection and fine-grained classification of cyberbullying events," pp. 672–680, 2015.

[65] H. Dani, J. Li, and H. Liu, "Sentiment informed cyberbullying detection in social media," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 10534 LNAI, pp. 52–67, 2017.

[66] J. Zhang, T. Otomo, L. Li, and S. Nakajima, "Cyberbullying detection on twitter using multiple textual features," *2019 IEEE 10th International Conference on Awareness Science and Technology, iCAST 2019 - Proceedings*, 10 2019.

[67] V. Balakrishnan, S. Khan, and H. R. Arabnia, "Improving cyberbullying detection using twitter users' psychological features and machine learning," *Computers & Security*, vol. 90, p. 101710, 3 2020.

[68] N. Yuvaraj, K. Srihari, G. Dhiman, K. Somasundaram, A. Sharma, S. Rajeskannan, M. Soni, G. S. Gaba, M. A. Alzain, and M. Masud, "Nature-inspired-based approach for automated cyberbullying classification on multimedia social networking," *Mathematical Problems in Engineering*, vol. 2021, 2021.

[69] B. A. Talpur and D. O'Sullivan, "Cyberbullying severity detection: A machine learning approach," *PLOS ONE*, vol. 15, p. e0240924, 10 2020.

[70] M. Arif, "A systematic review of machine learning algorithms in cyberbullying detection: Future directions and challenges," *Journal of*

Information Security and Cybercrimes Research, vol. 4, pp. 01–26, 6 2021.

[71] N. Singh and S. K. Sharma, "Review of machine learning methods for identification of cyberbullying in social media," *Proceedings - International Conference on Artificial Intelligence and Smart Systems, ICAIS 2021*, pp. 284–288, 3 2021.

[72] L. Cheng, K. Shu, S. Wu, Y. N. Silva, D. L. Hall, and H. Liu, "Unsupervised cyberbullying detection via time-informed gaussian mixture model," pp. 185–194, Association for Computing Machinery, 2020.

[73] A. Gupta, W. Yang, D. Sivakumar, Y. Silva, D. Hall, and M. N. Barioni, "Temporal properties of cyberbullying on instagram," *The Web Conference 2020 - Companion of the World Wide Web Conference, WWW 2020*, pp. 576–583, 4 2020.

[74] L. Cheng, R. Guo, Y. Silva, D. Hall, and H. Liu, "Hierarchical attention networks for cyberbullying detection on the instagram social network," *Proceedings*, pp. 235–243, 2019.

[75] D. Soni and V. Singh, "Time reveals all wounds: Modeling temporal characteristics of cyberbullying," *Proceedings of the International AAAI Conference on Web and Social Media*, vol. 12, 6 2018.

[76] M. A. Al-Garadi, M. R. Hussain, N. Khan, G. Murtaza, H. F. Nweke, I. Ali, G. Mujtaba, H. Chiroma, H. A. Khattak, and A. Gani, "Predicting cyberbullying on social media in the big data era using machine learning algorithms: Review of literature and open challenges," *IEEE Access*, vol. 7, pp. 70701–70718, 2019.

[77] S. Salawu, Y. He, and J. Lumsden, "Approaches to automated detection of cyberbullying: A survey," *IEEE Transactions on Affective Computing*, vol. 11, pp. 3–24, 1 2020. Date of publication: 10 October 2017 ¡br/¿But published in: IEEE Transactions on Affective Computing (Volume: 11, Issue: 1, 01 Jan.-March 2020).

[78] H. Rosa, N. Pereira, R. Ribeiro, P. C. Ferreira, J. P. Carvalho, S. Oliveira, L. Coheur, P. Paulino, A. M. V. Simão, and I. Trancoso, "Automatic cyberbullying detection: A systematic review," *Computers in Human Behavior*, vol. 93, pp. 333–345, 4 2019.

[79] K. Yang, T. Zhang, and S. Ananiadou, "A mental state knowledge–aware and contrastive network for early stress and depression

detection on social media," *Information Processing & Management*, vol. 59, p. 102961, 7 2022.

[80] M. H. Zaib, F. Bashir, K. N. Qureshi, S. Kausar, M. Rizwan, and G. Jeon, "Deep learning based cyber bullying early detection using distributed denial of service flow," *Multimedia Systems*, vol. 1, pp. 1–20, 3 2021.

[81] H.-Y. Chen and C.-T. Li, "Henin: Learning heterogeneous neural interaction networks for explainable cyberbullying detection on social media," pp. 2543–2552, Association for Computational Linguistics, 11 2020.

[82] D. Soni and V. Singh, "See no evil, hear no evil: Audio-visual-textual cyberbullying detection," *Proceedings of the ACM on Human-Computer Interaction*, vol. 2, 11 2018.

[83] R. I. Rafiq, H. Hosseinmardi, R. Han, Q. Lv, S. Mishra, and S. A. Mattson, "Careful what you share in six seconds: Detecting cyberbullying instances in vine," *Proceedings of the 2015 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining, ASONAM 2015*, pp. 617–622, 8 2015.

[84] R. I. Rafiq, H. Hosseinmardi, S. A. Mattson, R. Han, Q. Lv, and S. Mishra, "Analysis and detection of labeled cyberbullying instances in vine, a video-based social network," *Social Network Analysis and Mining*, vol. 6, pp. 1–16, 12 2016.

[85] "Vine." `https://vine.co/`.

[86] "Vine faqs." `https://help.twitter.com/en/using-twitter/vine-faqs`.

[87] H. Hosseinmardi, S. A. Mattson, R. I. Rafiq, R. Han, Q. Lv, and S. Mishra, "Detection of cyberbullying incidents on the instagram social network," 3 2015. Use @techreport for APA citation.

[88] "Instagram." `https://www.instagram.com/`.

[89] "Instagram help center." `https://help.instagram.com/`.

[90] D. E. Losada, F. Crestani, and J. Parapar, "Overview of erisk 2019 early risk prediction on the internet," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 11696 LNCS, pp. 340–357, 2019.

[91] P. Aitken, B. Claise, and B. Trammell, "Specification of the ip flow information export (ipfix) protocol for the exchange of flow information." Available at `https://rfc-editor.org/rfc/rfc7011.txt`.

[92] B. Trammell and E. Boschi, "Bidirectional flow export using ip flow information export (ipfix)," 1 2008. Bidirectional Flow export Proposed Standar from 2008 referenced by RFC7011 which is the Internet Standar for IPFIX (IP Flow Information Export).

[93] F. Cacheda, D. Fernández, F. J. Novoa, and V. Carneiro, "Analysis and experiments on early detection of depression," 2018.

[94] F. Cacheda, D. Fernandez, F. J. Novoa, and V. Carneiro, "Early detection of depression: Social network analysis and random forest techniques," *Journal of Medical Internet Research*, vol. 21, p. e12554, 6 2019.

[95] E. Frank, M. A. Hall, I. H. Witten, and M. Kaufmann, "Weka workbench online appendix for "data mining: Practical machine learning tools and techniques"," 2016.

[96] "scikit-learn: machine learning in python — scikit-learn 1.1.2 documentation." `https://scikit-learn.org/stable`.

[97] P. K. Smith, J. Mahdavi, M. Carvalho, S. Fisher, S. Russell, and N. Tippett, "Cyberbullying: its nature and impact in secondary school pupils," *Journal of Child Psychology and Psychiatry*, vol. 49, pp. 376–385, 4 2008.

[98] J.-M. Xu, K.-S. Jun, X. Zhu, and A. Bellmore, "Learning from bullying traces in social media," 2012.

[99] T. G. Dietterich, R. H. Lathrop, and T. Lozano-Pérez, "Solving the multiple instance problem with axis-parallel rectangles," *Artificial Intelligence*, vol. 89, pp. 31–71, 1 1997.

[100] J. Amores, "Multiple instance classification: Review, taxonomy and comparative study," *Artificial Intelligence*, vol. 201, pp. 81–105, 8 2013.

[101] H. Hosseinmardi, S. A. Mattson, R. I. Rafiq, R. Han, Q. Lv, and S. Mishra, "Analyzing labeled cyberbullying incidents on the instagram social network," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 9471, pp. 49–66, 2015.

[102] D. Fernandez, L. Vigoya, F. Cacheda, F. J. Novoa, M. F. Lopez-Vizcaino, and V. Carneiro, “A practical application of a dataset analysis in an intrusion detection system,” pp. 1–5, 11 2018.

Resumen extendido en castellano

En la actualidad, las redes de comunicaciones han pasado a formar parte fundamental de la vida diaria, tanto para las personas usuarias de las mismas como para todos los organismos, instituciones y empresas que dependen de ellas para realizar sus funciones más esenciales. Esto provoca que los peligros que conlleva su utilización hayan aumentado y por tanto tengan mayor impacto en aquellas situaciones que requieran su uso. En este sentido, no solo la seguridad de los propios sistemas que permiten la comunicación o de los datos que procesan y transportan están en peligro, las personas usuarias de las mismas se vuelven objetivos de ataques. El aumento del uso de plataformas sociales de comunicación y comunidades en línea en forma de redes sociales permite que comportamientos problemáticos en el mundo real aumenten sus capacidades mediante las herramientas que ofrece Internet. De esta forma, las limitaciones geográficas y temporales se ven difuminadas y las probabilidades de producir daños se incrementan. Es por ello, junto a que la extensión en el tiempo de un ataque aumenta la posibilidad de que las consecuencias derivadas del mismo sean más graves, que se debe priorizar la detección temprana de los mismos, de manera que se puedan mitigar los problemas que estos generan. En relación a esto, el ciberacoso se ha convertido en un problema en Internet y en particular en las redes sociales.

Para abordar el problema de la detección de amenazas contra la ciberseguridad y limitar su expansión en el tiempo, han de definirse procedimientos de detección temprana tanto en relación a los métodos de detección como a las métricas empleadas para medir su rendimiento desde un punto de vista consciente del tiempo. Con el objetivo de conseguir esto, se ha definido formalmente el problema de la detección temprana y se han establecido diferentes metodologías alternativas para la evaluación. En cuanto a las métricas de detección temprana, se ha partido del estudio de métricas de última generación como Early Detection Risk Error (ERDE) y F-latency y se han propuesto alternativas como NormalizedERDE, Time aware Precision (TaP) y Time aware F-score (TaF) para resolver los problemas detectados en las primeras. En cuanto a la mejora de los modelos de detección temprana, se han utilizado las métricas conscientes del tiempo definidas para la construcción y evaluación de tres modelos: modelo de punto fijo, modelo, umbral y modelo dual. Además se han estudiado las características utilizadas para el entrenamiento de estos modelos, incluyendo conjuntos de características con el objetivo de mejorar la detección temprana del ciberacoso en redes sociales. En particular en este aspecto se ha analizado el rendimiento de los sistemas propuestos ante la inclusión de características Doc2Vec y Multiple Instance Learning.

Introducción

Conforme las redes de comunicaciones han pasado a formar parte de la vida se han incrementado los riesgos que estas conllevan. Por ello, es primordial garantizar un nivel adecuado de seguridad tango para los sistemas que interconectan como para las personas usuarias de los mismos. Teniendo como objetivo detectar estos riesgos lo antes posible, reduciendo de esta manera posibles daños que puedan producir.

La ciberseguridad se puede definir como las acciones tomadas para proteger a una persona, organización o país y sus sistemas de información contra crímenes y ataques realizados mediante el uso de Internet [1]. Como se puede observar, pese a que habitualmente este término se aplique únicamente a los sistemas físicos y lógicos que proporcionan las funcionalidades, debería aplicarse así mismo a las propias personas usuarias.

La seguridad de sistemas es la base de que lo que habitualmente se conoce como ciberseguridad y se ocupa de preservar la integridad y el correcto funcionamiento de las infraestructuras de las tecnologías de la información. Los ataques contra estos sistemas siguen una serie de pasos que han sido descritos como [3]: Reconocimiento, Militarización, Distribución, Explotación, Instalación, Comando y Control (C2) y Acciones en Objetivos. Si se detecta una amenaza en una fase inicial, se podrá evitar el aprovechamiento de las siguientes.

Desde el punto de vista de la protección de las personas usuarias, un gran peligro proviene de las redes sociales, donde las interacciones ven amplificadas por las capacidades de estos sistemas para generar posibles problemas mentales derivados de su uso. En este entorno, una de las problemáticas con mayor presencia es el ciberacoso [4] y puede dar lugar a consecuencias psicológicas devastadoras como: depresión, disminución de la autoestima, idealizaciones suicidas o suicidio [5].

En ambos casos, resultaría beneficioso realizar una detección temprana de estas situaciones dado que reduciría el daño disminuyendo el impacto negativo producido.

Objetivos

En esta tesis se aborda el problema de la detección temprana desde un punto de vista genérico y se aplica en particular al ámbito de la ciberseguridad. En concreto se evalúan diferentes facetas de este ámbito. Se estudia tanto la seguridad de las redes de comunicaciones como la protección de las personas usuarias de redes sociales frente al ciberacoso. Para conseguir esto, se analizan las métricas existentes para la evaluación de modelos de detección y se definen alternativas que resuelvan problemas los detectados si es necesario. Estas métricas se usan a su vez para analizar el rendimiento de diferentes métodos de detección.

Además, para la detección de ciberacoso en redes sociales, se proponen diferentes aproximaciones para la mejora de los modelos de detección temprana, tanto en términos de definición de modelos como de características usadas. En el primer caso analiza el rendimiento de los modelos de punto fijo, umbral y dual y en el segundo características como Doc2Vec y Multiple Instance Learning.

Los objetivos pueden resumirse como sigue:

- Análisis de métricas conscientes y no conscientes del tiempo
- Estudio de metodos de evaluación
- Propuesta de modelos de detección temprana
- Evaluación de características para modelos de detección temprana

Conclusiones

Como se ha introducido, el problema de la detección temprana se ha definido y supone un asunto de suma importancia en diversos campos, y en este caso en particular para la ciberseguridad. En este ámbito se han detectado diversos aspectos desde la seguridad de las redes de comunicaciones a la protección de las personas usuarias de sus sistemas.

Dentro del primer caso es evidente que una interrupción temprana de cualquier amenaza contra la infraestructura reducirá los costes de operación y minimizará el impacto en los negocios u organizaciones a los que presta servicio. Esta acción puede proteger o evitar daños sobre los datos, infraestructuras y reducir el tiempo de inactividad.

En el segundo caso, tanto la depresión como el ciberacoso tienen una relación directa con la duración de la acción, en relación al riesgo que generan o por su definición como una repetición en el tiempo de una serie de acciones. Con respecto a esta repetición en el tiempo, y con mayor relevancia para el caso del ciberacoso, se han definido modelos y métodos específicos con el fin de capturar esta relación con el tiempo para agilizar su detección.

En un primer lugar se ha definido de manera formal la detección temprana, luego se han identificado las fases o acciones que entran en funcionamiento en la creación de un sistema de detección para finalmente definir, si era necesario, modificaciones para mejorar sus resultados. En este aspecto se han detectado dos puntos clave: las métricas y los métodos. La primera cuestión esencial se trata de la definición de un sistema de medida para este tipo de aplicaciones. Además, diferentes métodos han sido aplicados para mejorar el rendimiento de los sistemas de detección cuando se aplican las métricas definidas. Pese a que estos dos puntos son imprescindibles en esta clase de problemas, hay un asunto que incluye a ambos y que debe ser tenido en cuenta, se trata de la metodología usada para la evaluación. Para ello se han aplicado dos aproximaciones diferentes: procesamiento por lotes y procesamiento en flujo. Cuando se aplica procesamiento por lotes, se evalúa en cada punto un porcentaje del número total de items de cada entidad, en cambio con la procesamiento en flujo, el mismo número de items se usa para todas las entidades que se están evaluando.

En relación a la definición de las métricas, el punto de inicio se basó en métricas previamente definidas, entre las cuales se estudiaron *Early Risk Detection Error* ($EREDE$) y $F_{latency}$. Como resultado de este estudio se detectaron algunos problemas tales que la dependencia del conjunto de datos, el análisis de resultados entre conjuntos de datos y la dificultad de interpretación de los resultados. Para lidiar con esto, se definieron diferentes alternativas a las métricas estudiadas. La primera de ellas fue una modificación de $ERDE$ denominada $NormalizedERDE$ o $ERDE$ normalizado, la que por medio de una normalización de los resultados de la métrica $ERDE$ para diferentes valores del mismo conjunto de datos, trataba de permitir el análisis de sus resultados entre diferentes conjuntos de datos. La segunda métrica consciente del tiempo, *Time aware Precision* (TaP) fue definida siguiendo los principios de la definición de $ERDE$ pero utilizando métricas no dependientes del tiempo como referencia. Como resultado se obtuvo una métrica dependiente del problema en lugar de una métrica dependiente del conjunto de datos. Pese a que TaP resuelve algunos de los problemas detectados hasta el momento, mantiene algunos relacionados con la interpretación de sus resultados y para lidiar con ellos y siguiendo los mismos criterios que con TaP se ha definido una nueva métrica, *Time aware F-score* (TaF), con una salida similar a la obtenida con $F - score$ pero incluyendo una dependencia del tiempo en su penalización. En resumen, el objetivo era obtener una métrica dependiente del tiempo que operase de manera independiente al conjunto de datos utilizado.

Los modelos de detección para detección temprana se estudiaron principalmente en el ámbito del ciberacoso en redes sociales, en particular, conjuntos de datos obtenidos de Vine e Instagram fueron usados para validar las características y modelos que se proponen. En particular se proponen tres modelos: punto fijo, umbral y dual. Para el primer caso se estudiaron diferentes niveles de granularidad pero sin detectar diferencias significativas al disminuir el grano de publicación a número de palabras.

Para el análisis de las características utilizadas se siguió una aproximación en la que se han tenido en cuenta tanto características específicas de la red social en particular como una aproximación genérica que obvia particularidades de estas. Con el objetivo de conseguir este doble análisis, la información obtenida de los perfiles de usuario de cada red social en particular, se ha tenido en cuenta o se ha descartado para estudiar el rendimiento de los sistemas en cada caso. En primer lugar, se incluyeron características particulares de cada red social, incluyendo o no características de similitud textual y características temporales, lo cual demostró como el modelo de umbral era capaz de mejorar los resultados obtenidos con los experimentos de referencia en un 26% y que el modelo dual podía incrementar esta diferencia hasta un 42% usando *Extra Tree* como modelo base en ambos casos. Tras esto, la inclusión de las características propuestas junto con las extraídas de la red social y que se usaron para los experimentos de referencia, mejoran los resultados tanto del modelo umbral como el dual. En resumen, en este caso, el modelo dual proporciona el mejor rendimiento para la detección temprana cuando se aplican todas las características para la identificación de casos positivos. Y empleando características más simples que obvian las características temporales y textuales que junto con umbrales bajos producen mejores resultados para los modelos negativos.

Para la generalización de los resultados en diferentes redes sociales, los resultados siguientes fueron obtenidos sin el uso de características particulares de cada plataforma ni particularidades de los perfiles de usuario. En su lugar, solo se incluyó información extraída de los comentarios realizados por las personas usuarias de la red social, como es el caso de las características Doc2Vec y MIL. El uso del primer conjunto de características mostró una mejoría para todos los modelos de detección y el segundo, mejores resultados en aquellos casos en los que el tamaño de las publicaciones era menor y el uso del idioma inglés estándar más escaso.

Finalmente se debe resaltar un detalle del análisis de los resultados en relación a los experimentos mostrados con el conjunto de datos de ataques en red. El tipo de ataque representado en este conjunto de datos es un escaneo de sistemas operativos, en el que el agresor trata de extraer información de una máquina conectada a la red (e.g. que puertos se encuentran abiertos, que servicios están disponibles o cual es el sistema operativo que utiliza el dispositivo). Al estudiar el tráfico de red es bastante habitual crear flujos para su análisis. En este caso particular la gran mayoría de flujos de ataque tendrían tan solo unos dos paquetes y esto afecta no solo en la detección si no también en la evaluación de los sistemas. Para evitar esto sería necesario realizar una agregación de más alto nivel, por ejemplo como se muestra en [102]. Esto, unido a la naturaleza de flujo de datos de la evaluación del tráfico de red en donde no existe información completa de las entidades, influye así mismo en que la penalización debería ser dependiente del problema en lugar de dependiente del conjunto de datos. Además, esta penalización debería estar definida conforme el numero de elementos necesarios para tomar la decisión.

En resumen, los resultados obtenidos son:

- Evaluación de la importancia de la detección temprana, específicamente en las diferentes facetas de la ciberseguridad.
- Análisis de la metodología de evaluación, tanto para procesado por lotes como para procesado por flujos.
- Propuesta de evaluaciones más realistas basadas en el procesado por flujos y en evaluación basada puramente en tiempo.
- Presentación de tres modelos de aprendizaje máquina base que pueden ser combinados para mejorar la detección temprana:
  - Modelo fijo (con diferentes niveles de granularidad)
  - Modelo umbral
  - Modelo dual
- Selección de un conjunto de características basadas en comentarios de usuarios e inclusión de características específicas para representar aspectos relacionados con el tiempo:
  - Doc2Vec
  - Multiple Instance Learning

Contribuciones

Las publicaciones relacionadas con esta tesis y sus contribuciones se pueden resumir de la siguiente manera:

- *"Breaking the Cyber Kill Chain: Early Intrusion Detection for Scan Attacks"* [6]: Introducción a la metodología de detección temprana, análisis de las métricas $ERDE$ y $F_{latency}$ y presentación de una nueva métrica *NormalizedERDE*.
- *"Measuring early detection of anomalies"* [7]: Análisis profundo tanto de las metodologías de evaluación en lotes como por flujos y propuesta de la métrica *Time aware Precision* or TaP.
- *"Time aware F-score for cybersecurity early detection evaluation"* [8]: Evaluación de la nueva métrica *Time aware F-score* or TaF.
- *"Early detection of cyberbullying on social media networks"* [9]: Estudio de la detección de métodos de detección temprana del ciberacoso mediante una evaluación consciente del tiempo.
- *"Site agnostic approach to early detection of cyberbullying on social media networks"* [10]: Análisis de tres alternativas de modelos de detección temprana (punto fijo, umbral y dual), así como características Doc2Vec y Multiple Instance Learning para la mejora de la detección temprana.

Trabajo futuro

Algunos puntos son claves en el desarrollo de sistemas de detección temprana, partiendo de la medición del rendimiento de estos sistemas. Se han presentado tanto sistemas de evaluación basados en lotes como los basados en flujos de datos pero una aproximación más realista sería aquella basada únicamente en tiempo. En este caso, un número determinado de items se usaría para la evaluación en cada punto temporal, definido como la marca de tiempo de cada uno de los items. Unicamente aquellos que pertenezcan al período definido para la evaluación se tendrán en cuenta, esto proporciona al sistema una imagen más realista de la situación en el mundo real en cuanto a la evaluación en línea.

Además, considerando que el problema de la detección temprana se puede aplicar en múltiples y diferentes entornos, la formalización propuesta, las métricas y métodos, esperan proveer una aproximación genérica a este problema. Debido a esto, otros puntos donde la investigación presentada podría ampliarse es en diversos campos de aplicación. Aunque en esta tesis se han presentado diversos aspectos de la ciberseguridad, tales como los sistemas y los recursos humanos, hay múltiples campos que pueden beneficiarse de la aproximación de la detección temprana. En un primer lugar, sería interesante profundizar en el uso de diferentes conjuntos de datos con ataques diversos para comparar el rendimiento de los modelos presentados. Igualmente, otros campos no relacionados con la ciberseguridad, como la medicina o la prevención de emergencias o desastres son algunos ejemplos donde podría resultar interesante estudiar la aplicación de la detección temprana.

En cuanto a los modelos presentados, existen diversas maneras de ampliar la investigación realizada. Por una parte, en cuanto a la definición de los métodos de aprendizaje máquina, donde podrían utilizarse combinaciones heterogéneas de los diferentes modelos base aplicados. Así mismo sería interesante estudiar el comportamiento de modelos basados en Deep Learning mediante métricas conscientes del tiempo para analizar sus capacidades de extracción de información temporal en estos casos.

Por último, y relacionado con la información extraída de los usuarios, la expansión de las características usadas para alimentar el modelos para su entrenamiento podría mejorarse incrementando el estudio de la extracción de características y su análisis. La extracción de características estaría centrada en los comentarios, ya que mantendría el modelo independiente de la plataforma y porque estos concentran la mayor parte de la información para la detección temprana del ciberacoso. Además, la aplicación de Multiple Instance Learning podría extenderse usando modelos del espacio de instancias MIL mas avanzados. Entre ellos, podrían aplicarse Expectation-Maximization Diverse Density (EMDD), o incluso modelos de espacios de instancias de conjuntos o espacios incrustados. Entre estos, podrían aplicarse Normalized Set Kernel (NST-SMV) o Multiple Instance Learning via Embedded Instance Selection (MILES).

Resumo estendido en galego

Na actualidade, as redes de comunicacións forman parte fundamental da vida diaria, tanto para as persoas usuarias das mesmas com para tódolos organismos, institucións e empresas que dependen delas para realizar as súas funcións máis esenciais. Isto provoca que os perigos que trae consigo a súa utilización aumentaran e teñan por tanto un maior impacto naquelas situacións que requiran o seu uso. Neste sentido, non só a seguridade dos propios sistemas que permiten a comunicación, ou o dos datos que procesan e transportan están en perigo, as persoas usuarias das mesmas vólvense obxectivos de ataques. O aumento do uso de plataformas sociais de comunicación e comunidades en liña en forma de redes sociais permite que comportamentos xa problemáticos no mundo real aumenten as súas capacidades mediante as ferramentas que ofrece Internet. Desta forma, as limitacións xeográficas e temporais vense esvaídas e as probabilidades de producir danos incrementase. É por isto, xunto a que a extensión do tempo dun ata que aumenta a posibilidade de que as consecuencias derivadas do mesmo sexan mais graves, que se debe priorizar a detección temperá dos mesmos, de forma que se podan mitigar os problemas que estes xeran. En relación a isto, o ciberacoso converteuse nun problema en Internet e en particular nas redes sociais.

Para abordar o problema da detección de ameazas contra a ciberseguridade e limitar a súa expansión no tempo, deben definirse procedementos de detección temperá tanto en relación aos métodos de detección como as métricas empregadas para medir o seu rendemento dende un punto de vista consciente do tempo. Co obxectivo de conseguir isto, definiuse formalmente o problema da detección temperá e establécense diferentes metodoloxías alternativas para a avaliación. En canto as métricas de detección tempera, partiuse do estudo de métricas de última xeración como Early Detection Risk Error (ERDE) e F-latency e propuxéronse novas alternativas coma NormalizedERDE, Time aware Precision (TaP) e Time aware F-score (TaF) para resolver os problemas detectados nas primeiras. En canto a mellora dos modelos de detección temperá, utilizáronse as métricas conscientes do tempo definidas para a construción e avaliación de tres modelos: modelos de punto fixo, modelo limiar e modelo dual. Ademais, estudáronse as características utilizas para o adestramento destes modelos, incluíndo conxuntos de características co obxectivo de mellorar a detección temperá do ciberacoso en redes sociais. En particular neste aspecto analizouse o rendemento dos sistemas propostos ante a inclusión de características Doc2Vec e Multiple Instance Learning.

Introdución

Conforme as redes de comunicacións pasaron a formar parte da vida os riscos que estas traen consigo incrementáronse. Por iso, é primordial garantir un nivel adecuado de seguridade tanto para os sistemas que interconectan como para as persoas usuarias dos mesmos. Tendo como obxectivo detectar estes riscos o antes posible, reducindo desta maneira posibles danos que se podan producir.

A ciberseguridade pódese definir como as accións tomadas para protexer a unha persoa, organización ou país e os seus sistemas de información contra crimes e ataques realizados mediante o uso de Internet [1]. Como se pode observar, pese a que normalmente este término so se aplique aos sistemas físicos e lóxicos que proporcionan as funcionalidades, debería aplicarse así mesmo as propias persoas usuarias.

A seguridade de sistemas é a base do que se coñece comunmente como ciberseguridade e ocupase de preservar a integridade e o correcto funcionamento das infraestruturas das tecnoloxías da información. Os ataques contra estes sistemas seguen unha serie de pasos que foron descritos coma [3]: Recoñecemento, Militarización, Distribución, Explotación, Instalación, Comando e Control (C2) e Accións en Obxectivos. Se se detecta unha ameaza nunha fase inicial, poderase evitar o aproveitamento das seguintes.

Dende o punto de vista da protección das persoas usuarias, un gran perigo ven das rede sociais, onde as interaccións en amplifícaas polas capacidades destes sistemas para xeras posibles problemas mentais derivados dos seu uso. Neste entorno, unha das problemáticas con maior presencia é o ciberacoso [4] e pode dar lugar a consecuencias psicolóxicas devastadoras coma: depresión, diminución da autoestima, ideacións suicidas ou suicidio [5].

En ambos casos, resultaría beneficioso realizar unha detección temperá desta situacións dado que reduciría o dano diminuíndo o impacto negativo producido.

Obxectivos

Nesta tesis abordase o problema da detección temperá dende un punto de vista xenérico e aplicase en particular ao ao ámbito da ciberseguridade. En concreto avalíanse diferentes facetas deste ámbito. Estudase tanto a seguridade nas redes de comunicacións como a protección das persoas usuarias das redes sociais fronte ao ciberacoso. Para conseguir isto, analízanse as métricas existentes para a avaliación de modelos de detección e defínense alternativas que resolvan problemas dos detectados se fose necesario. Estas métricas úsanse a sua vez para analizar o rendemento de diferentes métodos de detección.

Ademais, para a detección do ciberacoso en redes sociais, propóñense diferentes aproximacións para a mellora dos modelos de detección temperá, tanto en términos de definición de modelos como de características usadas. No primeiro caso analizase o rendemento dos modelos de punto fixo, limiar e dual e no segundo características coma Doc2Vec e Multiple Instance Learning.

Os obxectivos poden resumirse como segue:

- Análise de métricas conscientes e non conscientes do tempo
- Estudo de métodos de avaliación.
- Proposta de modelos de detección temperá
- Avaliación de características para modelos de detección temperá

Conclusións

Como se introduciu, o problema da detección temperá definiuse e supón un asunto de suma importancia en diversos campos, e neste caso en particular para a ciberseguridade. Neste ámbito detectáronse diversos aspectos dende a seguridade das redes de comunicacións á protección das persoas usuarias dos seus sistemas.

Dentro do primeiro caso é evidente que unha interrupción temperá de calquera ameaza contra a infraestrutura reducirá os custos de operación e minimizará o impacto nos negocios ou organizacións aos que presta servizo. Esta acción pode protexer ou evitar danos sobre os datos, infraestruturas e reducir o tempo de inactividade.

No segundo caso, tanto a depresión coma o ciberacoso teñen unha relación directa coa duración da acción, en relación ao risco que xeran ou pola súa definición coma unha repetición no tempo dunha serie de accións. Con respecto a esta repetición no tempo, e con maior relevancia para o caso do ciberacoso, definíronse modelos e métodos específicos co fin de capturar esta relación co tempo para axilizar a súa detección.

En primeiro lugar definiuse de maneira formal a detección temperá, logo identificáronse as fases ou accións que entran en funcionamento na creación dun sistema de detección para finalmente definir, se era necesario, modificacións para mellorar os seus resultados. Neste aspecto detectáronse dous puntos chave: as métricas e o métodos. A primeira cuestión esencial tratase da definición dun sistema de medida para este tipo de aplicacións. Ademais, aplicáronse diferentes métodos para mellorar o rendemento dos sistemas de detección cando se aplican as métricas definidas. Pese a que estes dous puntos son imprescindibles nesta clase de problemas, hai un asunto que inclúe a ambos e que debe terse en conta, trátase da metodoloxía usada para a avaliación. Para iso, definíronse dúas aproximacións diferentes: procesamento por lotes e procesamento en fluxo. Cando se aplica procesamento por lotes, avalíase en cada punto unha porcentaxe do número total de items de cada entidade, en cambio co procesamento en fluxo, o mesmo número de items úsanse para todas as entidades que se están a avaliar.

En relación a definición das métricas, o punto de inicio baseouse en métricas previamente definidas, entre as cales estudáronse *Early Risk Detection Error* ($ERDE$) e $F_{latency}$. Como resultado deste estudio detectáronse algúns problemas tales como a dependencia do conxunto de datos, o análise de resultado entre conxuntos de datos e a dificultade de interpretación dos resultados. Para facer fronte a isto, definíronse diferentes alternativas as métricas estudadas. A primeira delas foi unha modificación de $ERDE$ denominada $NormalizedERDE$ ou $ERDE$ normalizado, a que por medio dunha normalización dos resultados da métrica $ERDE$ para diferentes valores do mesmo conxunto de datos, trataba de permitir o análise dos seus resultados entre diferentes conxuntos de datos. A segunda métrica consciente do tempo, *Time aware Precision* TaP foi definida seguindo os principios da definición de $ERDE$ pero utilizando métricas non dependentes do tempo como referencia. Como resultado obtívose unha métrica dependente do problema en lugar de unha métrica dependente do conxunto de datos. Pese a que TaP resolve algúns dos problemas detectados ata o momento, mantén algúns relacionados coa interpretación dos seus resultados e para facerlles fronte e seguindo os mesmos criterio que con TaP definiuse unha nova métrica, *Time aware F-score* (TaF), con unha saída similar a obtida con $F-score$ pero incluíndo unha dependencia do tempo na súa penalización. En resumo, o obxectivo era obter unha métrica dependente do tempo que operase dunha maneira independente do conxunto de datos.

Os modelos de detección para detección temperá estudáronse principalmente no ámbito do ciberacoso en rede sociais, en particular, conxuntos de datos obtidos de Vine e Instagram foron usados para validar as características e modelos que se propoñen. En particular, propóñense tres modelos: punto fixo, limiar e dual. Para o primeiro caso estudáronse diferentes niveles de granularidade paro sen detectar diferencias significativas ao diminuír o grao de publicación ao número de palabras.

Para a análise das características utilizadas seguiuse unha aproximación na que tanto se tiveron en conta características específicas da rede social en particular como unha aproximación xenérica na que obvia as particularidades destas. Co obxectivo de conseguir esta dobre análise, a información obtida dos perfís de usuario de cada rede social en particular, tívose en conta ou descartouse para estudar o rendemento dos sistemas en cada caso. Nun primeiro lugar, incluíronse características particulares de caa rede social, incluíndo ou non características de similitude textual e características temporais, o que demostrou como o modelo limiar era capaz de mellorar os resultados obtidos cos experimentos de referencia nun 26% e que o modelo dual podía incrementar esta diferencia ata nun 42% usando *Extra Tree* como modelo base en ambos casos. Tras isto, a inclusión das características proposta xunto coas extraídas da rede social e que se usaron para os experimentos de referencia, melloran os resultados tato do modelo limiar como do dual. En resumo, neste caso, o modelo dual proporciona mellor rendemento para detección temperá cando se aplican tódalas características para a identificación de casos positivos. E empregando características mais simples que obvian as características temporais e textuais que xunto con limiares baixos producen mellores resultados para os modelos negativos.

Para a xeneralización dos resultados en diferentes redes sociais, os resultados seguintes foron obtidos sen o uso de características particulares de cada plataforma nin particularidades dos perfís de usuario. No seu lugar, so se incluíu información extraída dos comentarios realizados polas persoas usuarias da rede social, como é o caso das características Doc2Vec e MIL. O uso do primeiro conxunto de características mostrou unha melloría para tódolos modelos de detección e no segundo, mellores características naqueles casos en que o tamaño das publicacións era menor e o uso do idioma inglés estándar máis escaso.

Finalmente débese resaltar un detalle da análise dos resultados en relación aos experimentos mostrados co conxunto de datos de ataques en rede. Este tipo de ataque representado neste conxunto de datos é un escaneo de sistemas operativos, no que o agresor trata de extraer información dunha máquina conectada a rede (e.g. que portos se atopan abertos, que servizos están dispoñibles ou cal é o sistema operativos que utiliza o dispositivo). Ao estudar o tráfico de rede é bastante normal crear fluxos para a súa análise. Neste caso particular a gran maioría de fluxos de ataque terían tan só uns dous paquetes e isto afecta non só na detección senón tamén na avaliación destes sistemas. Para evitar isto sería necesario realizar unha agregación de mais alto nivel, por exemplo como se mostra en [102]. Isto, unido a natureza do fluxo de datos da avaliación do trafico de rede onde non existe información completa das entidades, inflúe así mesmo en que a penalización debería ser dependente do problema en lugar de dependente do conxunto de datos. Ademais, esta penalización debería estar definida conforme o número de elementos necesarios para a tomar a decisión.

En resumo, os resultados obtidos son:

- Avaliación da importancia da detección temperá, especificamente nas diversas facetas da ciberseguridade.
- Análise da metodoloxía de avaliación, tanto para o procesado por lotes como no procesado en fluxos e na avaliación baseada puramente en tempo.
- Proposta de avaliacións mais realistas baseadas en procesado en fluxos e en avaliación baseada puramente en tempo.
- Presentación de tres modelos de aprendizaxe máquina base que poden ser combinados para mellorar a detección temperá:
  - Modelo fixo (con diferentes niveis de granularidade)
  - Modelo limiar
  - Modelo dual
- Selección dun conxunto de características baseadas en comentarios das persoas usuarias e inclusión de características específicas para representar aspectos relacionados co tempo.
  - Doc2Vec
  - Multiple Instance Learning

Contribucións

As publicacións relacionadas con esta tese e as súas contribucións pódense resumir da seguinte maneira:

- *"Breaking the Cyber Kill Chain: Early Intrusion Detection for Scan Attacks"* [6]: Introdución a metodoloxía de detección temperá, análise das métricas $ERDE$ y $F_{latency}$ e presentación dunha nova métrica *NormalizedERDE*.

- *"Measuring early detection of anomalies"* [7]: Análise profundo tanto das metodoloxías de avaliación en lotes com por fluxos e proposta da métrica *Time aware Precision* or TaP.

- *"Time aware F-score for cybersecurity early detection evaluation"* [8]: Avaliación da nova métrica *Time aware F-score* or TaF.

- *"Early detection of cyberbullying on social media networks"* [9]: Estudo da detección de métodos de detección temperá do ciberacoso mediante unha avaliación consciente do tempo.

- *"Site agnostic approach to early detection of cyberbullying on social media networks"* [10]: Análise de tres alternativas de modelos de detección temperá (punto fixo, limiar e dual), así como características Doc2Vec e Multiple Instance Learning para a mellora da detección temperá.

Traballo futuro

Algúns puntos son chave no desenvolvemento de sistemas de detección temperá, partindo da medición do rendemento destes sistemas. Presentáronse tanto sistemas de avaliación baseados en lotes como os baseados en fluxos de datos, pero unha aproximación mais realista sería aquela baseada unicamente no tempo. Neste caso, un número determinado de items usaríanse para a avaliación en cada punto temporal, definido como a marca de tempo de cada un dos items. Unicamente aqueles que pertenzan ao período definido para a avaliación teranse en conta. Isto proporciona ao sistema unha imaxe mais realista da situación o mundo real en canto a avaliación en liña.

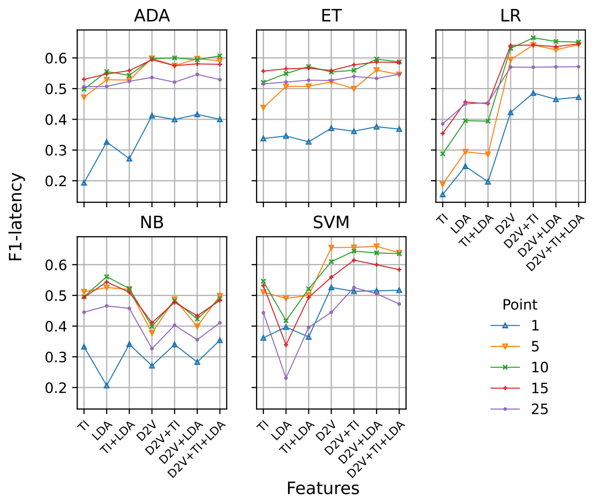
Ademais, considerando que o problema da detección temperá pode aplicarse en múltiples e diferentes contornas, a formalización proposta, as métricas e métodos, esperan prover unha aproximación xenérica a este problema. Debido a isto, outros puntos onde a investigación presentada podería ampliarse é nos diversos campos de aplicación. Aínda que nesta tese, presentase diversos aspectos da ciberseguridade, tales como os sistemas e os recursos humanos, hai múltiples campos que poden beneficiarse da aproximación da detección temperá. Nun primeiro lugar, sería interesante afondar no uso de diferentes conxuntos de datos con ataques diversos pra comparar o rendemento dos modelos presentados. Igualmente, outros campos non relacionados coa ciberseguridade, como a medicina ou a prevención de emerxencias ou desastres son algúns dos exemplos onde podería resultar interesante estudar a aplicación da detección temperá.

En canto aos modelos presentados, existen diversas maneiras de ampliar a investigación realizada. Por unha parte, en canto a definición dos métodos de aprendizaxe máquina, onde poderían utilizarse combinacións heteroxéneas dos diferentes modelos base aplicados. Así mesmo sería interesante estudar o comportamento de modelos baseados en Deep Learning mediante métricas conscientes do tempo para analizar as súas capacidades de extracción de información temporal nestes casos.

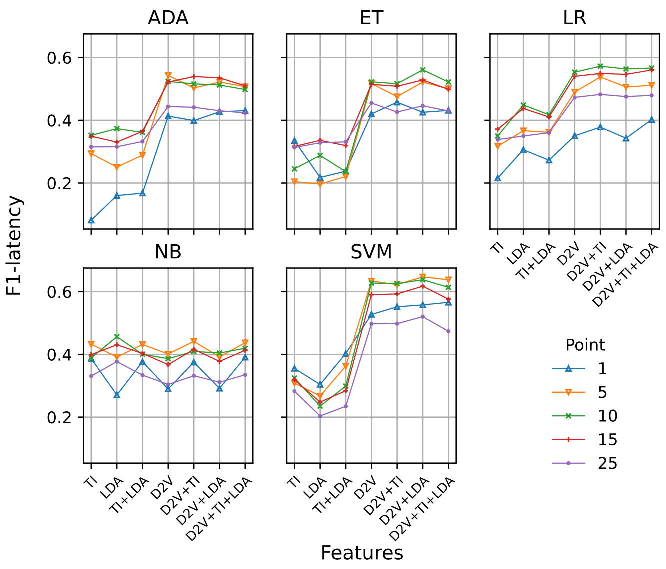
Por último, e relacionado coa información extraída dos usuarios, a expansión das características usadas para alimentar os modelos para o seu adestramento podería mellorarse incrementando o estudo da extracción de características e a sua análise. A extracción de características estaría centrada nos comentarios, xa que mantería o modelo independente da plataforma e porque estes concentran a maior parte da información para a detección temperá do ciberacoso. Ademais, a aplicación de Multiple Instance Learning podería estenderse usando modelos do espazo de instancias MIL mais avanzados. Entre eles, poderían aplicarse Expectation-Maximization Diverse Density (EMDD), ou incluso modelos de espazos de instancias de conxuntos ou espazos incrustados. Entre estas, poderían aplicarse Normalized Set Kernel (NST-SMV) ou Multiple Instance Learning via Embedded Instance Selection (MILES).

Additional data

This Appendix contains additional data for experiments of chapter 6, specifically $F_{latency}$ plots for TaP comparison.

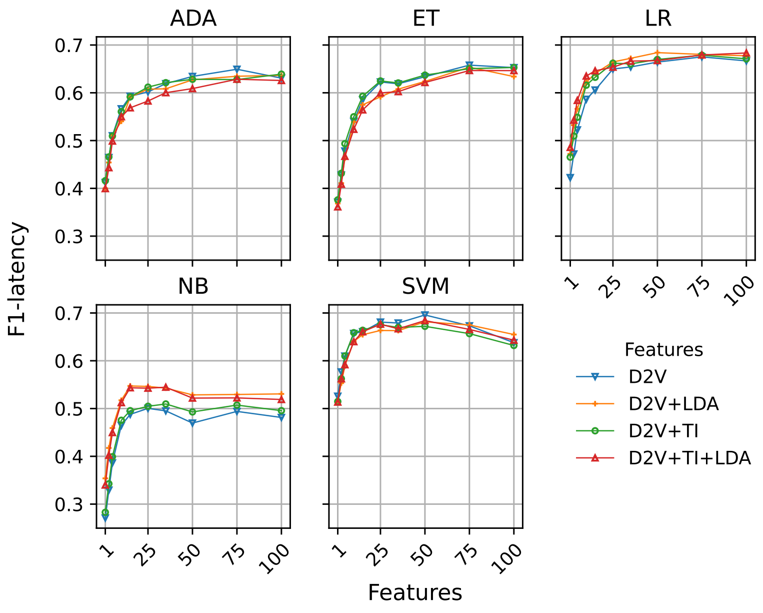


(a) Instagram

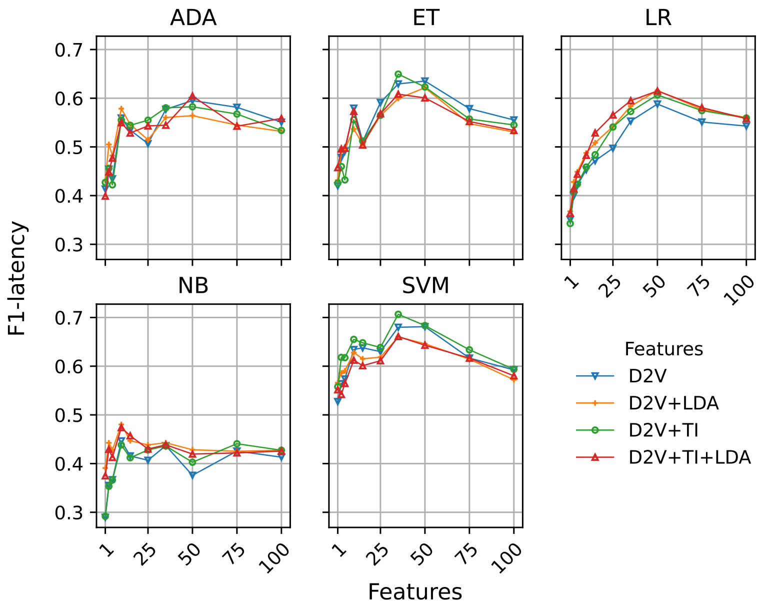


(b) Vine

Figure 1: $F_{latency}$ for fixed early detection model at points $1, 5, 10, 15$ and 25 using Instagram (1(a)) and (1(b)) datasets. One graph is presented for each machine learning model: AdaBoost (ADA), Extra Trees (ET), Logistic Regression (LR), Naïve Bayes (NB) and Support-Vector Machine (SVM). Features are represented on the X-axis.Tf-idf (TI) and Latent Dirichlet Allocation (LDA) and Doc2Vec.

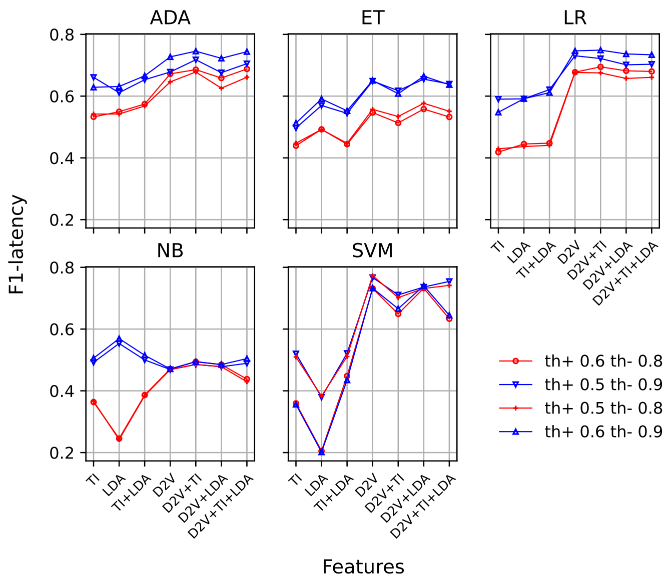


(a) Instagram

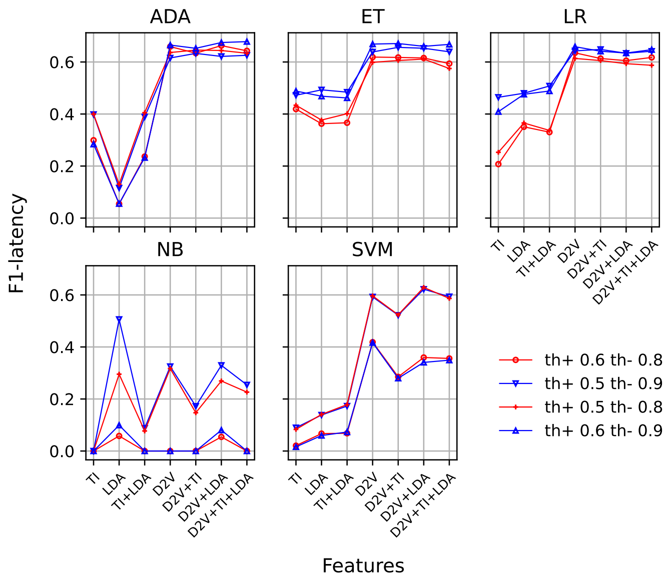


(b) Vine

Figure 2: $F_{latency}$ for fine-grain fixed early detection model using Instagram (2(a)) and Vine (2(b)) datasets. One graph is presented for each machine learning model. Different features combinations are represented on each graph.



(a) Instagram



(b) Vine

Figure 3: $F_{latency}$ for threshold early detection model using Instagram (3(a)) and Vine (3(b)) datasets. One graph is presented for each machine learning model. Different values for positive and negative thresholds are represented on each graph. Features are represented on the X-axis.

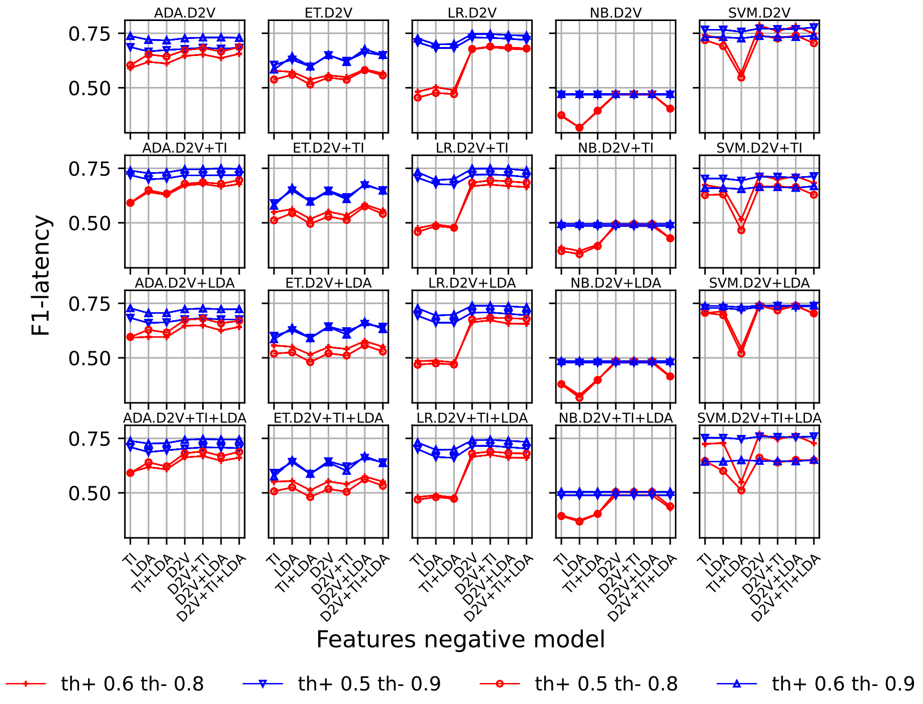


Figure 4: $F_{latency}$ for dual early detection model using Instagram dataset. Columns correspond to machine learning models. Rows correspond to positive features. Negative features are represented on the X-axis. Different values for positive and negative thresholds are represented on each graph.

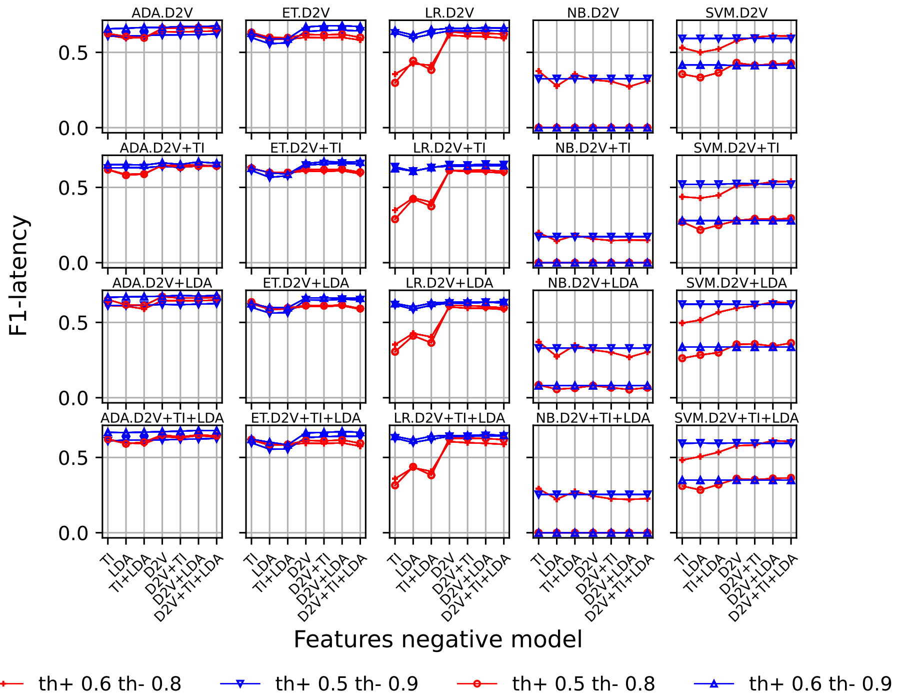
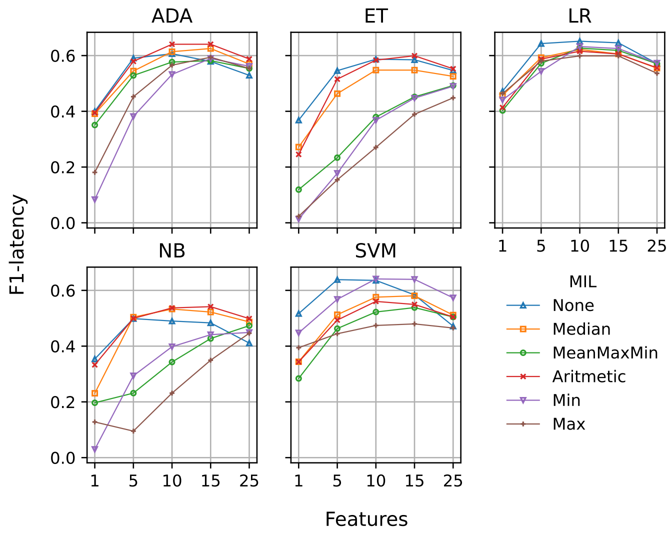
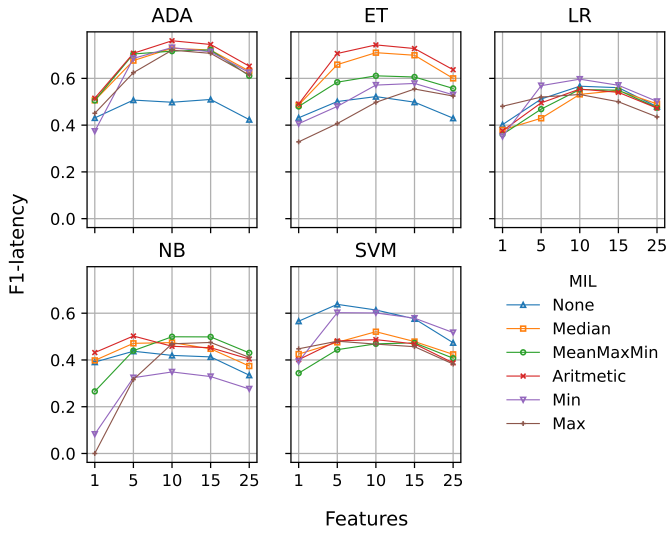


Figure 5: $F_{latency}$ for dual early detection model using Vine dataset. Columns correspond to machine learning models. Rows correspond to positive features. Negative features are represented on the X-axis. Different values for positive and negative thresholds are represented on each graph.

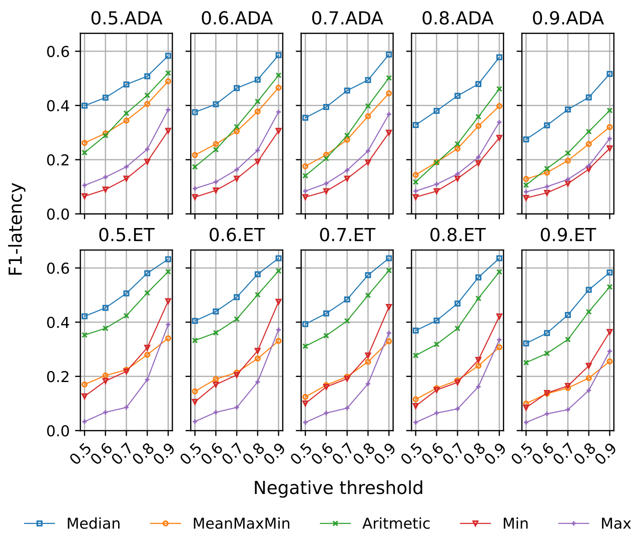


(a) Instagram

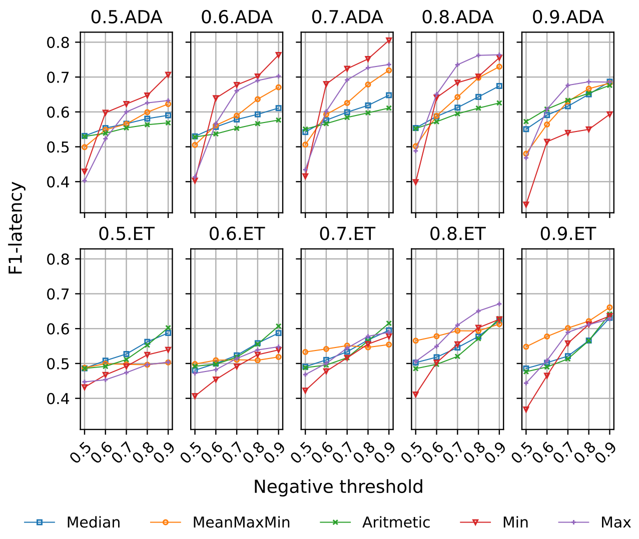


(b) Vine

Figure 6: $F_{latency}$ for fixed early detection model based using MIL for Instagram (6(a)) and Vine (6(b)) datasets. One graph is presented for each machine learning model. The X-axis represents the points where the fixed model produces its decision. Different aggregation functions are represented on each graph: no MIL (None), minimum (Min), maximum (Max), average maximum and minimum (MeanMaxMin), arithmetic mean (Arithmetic) and median (Median).



(a) Instagram



(b) Vine

Figure 7: $F_{latency}$ for threshold early detection model based using MIL for Instagram (7(a)) and Vine (7(b)) datasets. Rows correspond to machine learning models: ADA and ET. Columns correspond to positive thresholds. The X-axis represents negative thresholds. Different aggregation functions are represented on each graph:minimum (Min), maximum (Max), average maximum and minimum (MeanMaxMin), arithmetic mean (Arithmetic) and median (Median).