Cover Page – MSc Business Analytics Consultancy Project/Dissertation 2019-20

Title of Project: Generation of training data with Weak Supervision: An application in Cyber Threat Intelligence Feeds

Date: 22/08/2020

Word Count: 12424

Disclaimer:

I hereby declare that this dissertation is my individual work and to the best of my knowledge and confidence, it has not already been accepted in substance for the award of any other degree and is not concurrently submitted in candidature for any degree. It is the end product of my own independent study except where other acknowledgement has been stated in the text.

Marking Sheet – MSc Business Analytics Consultancy Project/Dissertation 2019-20

Criteria/Weight	Supervisor's comments
Topic, theoretical framework, literature, and methodology (30%): Topic is clearly identified and boundaries are asserted. Knowledge of relevant theories and their limitations. Current and relevant literature coming from reliable sources. Appropriate and adequate methodology for topics. Detailed methodology facilitating replication of project and reproducibility of results.	
Analysis and conclusions /recommendations (30%): Use of primary and/or secondary data. Rigorous analysis and interpretations. Alternative interpretations/arguments are considered. Limitations are identified and justified by reasonable arguments. Conclusions/recommendations are fully consistent with evidence presented.	
Structure, originality and presentation (10%): Provides a concise summary. Demonstrates an understanding of business context. Coherent and appropriate structure. Adequate presentation, language, style, graphs, tables, and referencing. Appropriate use of visualisation. Presents business recommendations.	
Complexity of project scope and progress made towards business goals (10%): Progress made towards overcoming technical and operational challenges encountered during the project. Progress made in overcoming problem framing and theoretical and data related problems encountered during the project.	
Project Management (10%): Good use of project management and communication tools. Use of Kanban board for structuring project work. Evidence of objectives being broken down in appropriate tasks and timely engagement with primary supervisor.	

General marking guidelines

85+	Outstanding work of publishable standard.
70-84	Excellent work showing mastery of the subject matter and excellent analytical skills.
60-69	Very good work. Interesting analysis with original insights. Some minor errors.
50-59	Good work which only covers a basic analysis. Some problems but no major omissions.
40-49	Inadequate work. Not sufficiently analytical. Some major omissions.
39-	Work seriously flawed. Lack of clarity and argumentation. Too descriptive.

Mark: __

Generation of training data with Weak Supervision: An application in Cyber Threat Intelligence Feeds

A dissertation submitted in partial fulfillment

of the requirements for the degree of

Master of Science

of

University College London

August 2020

Table of Contents

List of Tables

List of Figures

Nomenclature

Key terms that are used throughout this study are defined here:

- NLP: Natural Language Processing
- TTPs: Tactics, Techniques and Procedures
- LF: Labelling Function

Abstract

Creation of labelled data is one of the major bottlenecks for training Machine Learning models. Even if Data Science teams have access to state-of-the-art models, the lack of training data delay or even restrict the deployment of their Machine Learning applications. In this work, we provide an overview of the data labelling strategies, with a focus on textual data. We summarise their main properties and we relate them with the business problem of a cyber security team.

The team is interested in categorising records from Cyber Threat Intelligence Feeds based in their textual descriptions. With our solution, we demonstrate how Weak Supervision and Data Programming can create training data without having access to previously labelled instances. With the use of multiple logical conditions (Labelling Functions) we generate three labels that characterise the different textual descriptions of a Cyber Threat Intelligence Feed.

In order to examine the generalisation of the produced labels, we use them to train state-of-the-art Natural Language Models. Finally, we evaluate the performance of the latter with a hand-annotated dataset.

Keywords: Data Labelling, Weak Supervision, Data Programming, Snorkel,
Web Scrapping, Cyber Threat Intelligence Feeds, Natural Language Processing, BERT, RoBERTa

Chapter 1 Introduction

Both big data and the latest advancements in computational power have provided to Machine Learning an excellent feed to develop applications that reach or even exceed the human perception. ImageNet [1] has achieved performance higher than humans in image classification, where BERT [2] has surpassed the performance of expert human annotators in a common-sense inference task [3]. However, Machine Learning models in the supervised setting rely on large collections of data which are labelled.

These labelled datasets are usually annotated from humans and their creation normally require great amount of resources. Also, datasets may need new labels to be created in order to achieve new tasks. Thus, the labelling of the datasets is a critical bottleneck for training Supervised Machine Learning models.

Gartner, on its latest hype cycle for Data Science and Machine Learning [4] (fig. 1) places "Data Labelling and Annotation Services" in the stage of "Peak of Inflated Expectations" and expects that the field will reach a plateau in 2 to 5 years. This fact depicts that the data labelling approaches have recently started to appear commercially and we expect there will be more Data Science practitioners who will adopt them in the future.

In this work, we examine the data labelling approaches which deal with unstructured textual datasets. We aim to understand their main similarities and differences but also identify successful applications with them. This analysis will guide us to select

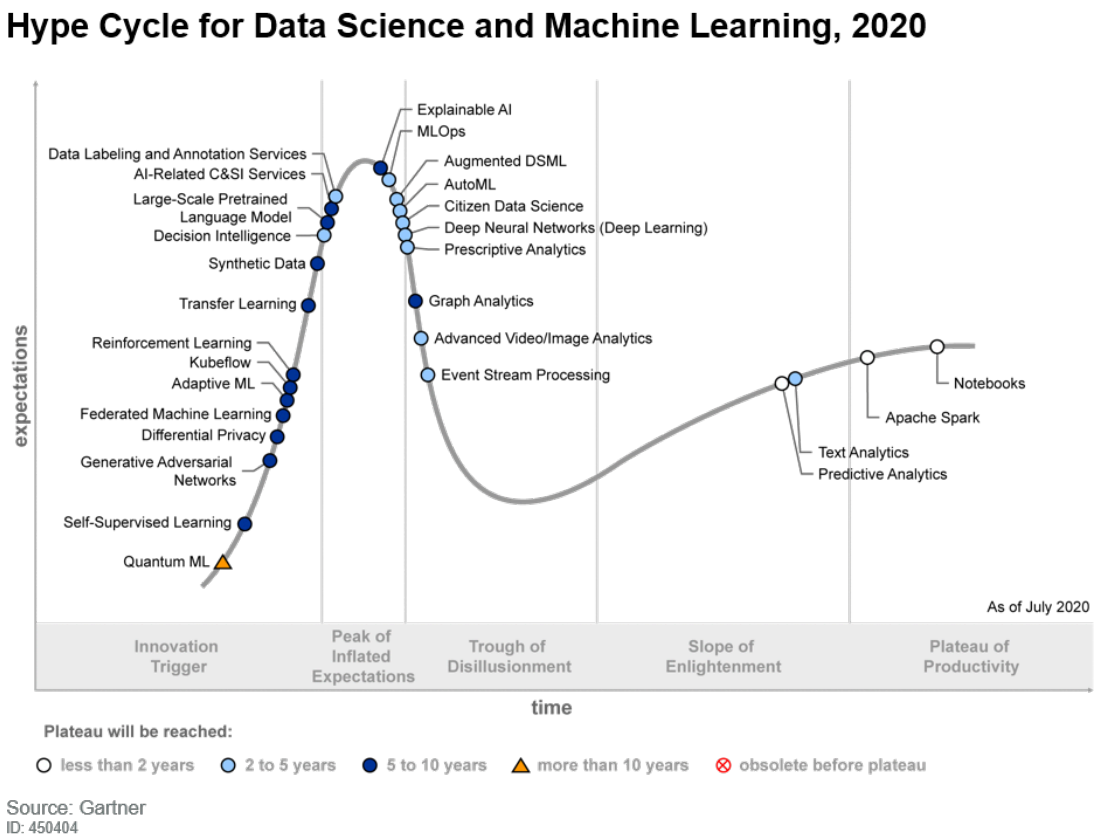


Figure 1: Hype Cycle for Data Science, Gartner, 2020

an appropriate data labelling approach that meets the business needs of a cyber-security team. Finally, we use state-of-the-art Natural Language models that utilise the new labels that we create.

Below, we provide a summary of the business problem that the cyber-security team faces, a problem formulation for our data labelling problem and an outline of the rest chapters of this work.

1.1 Business Problem

Currently, the cyber-security team is devoting a considerable amount of time into assessing cyber threat intelligence feeds that describe the activities of threat actors.

These feeds may contain documents that outline strategies of malicious agents, attack incidents as well as other types of information. These collections are particularly useful for the team as they offer insights for the way the threat agents act. Ultimately, this information helps the cyber-security team to better inform and protect its customers.

However, as this process requires substantial effort, team members would like to create two classifiers for documents that can be found in threat feeds. First, a classifier for the descriptions that can be found in them and secondly, a classifier for their accompanied external links.

Usually, in order to create these classifiers, the team would have first to annotate several descriptions, as well as the content of the external links. Then, these annotated sources could provide a proper labelled data source to create the final models.

In order to avoid this task, the team would like to exploit the available data labelling approaches in order create classifiers without having to annotate great quantities of data. Ideally, it would like to translate and transfer its domain knowledge to a classifier without having to annotate each data source explicitly.

1.2 Problem Formulation

In the Supervised Machine Learning task, a dataset consists of instances that have a feature-vector x1, x2 ..., xn and a ground-truth label y known also as response. Data labelling aims to generate the ground-truth label y, which in raw data is not available.

1.3 Outline

The rest of this study is organised as follows:

- Chapter 2 provides the background and a literature review of the main aspects of this study.
- Chapter 3 describes the design of our proposed solution for our business problem.
- Chapter 4 demonstrates how the solution was implemented.
- Chapter 5 summarises our work, discusses the limitations of our solution and provides recommendations for future work.

Chapter 2 Background & Literature Review

This chapter is organised as follows:

Section 2.1, discusses the background of the Cybersecurity domain, the Tactics, techniques and procedures (TTPs) of threat agents as well as the Cyber Threat Intelligence Feeds, the data collections we deal with in our solution.

Section 2.2, examines the data labelling strategies, by first collecting relevant studies and surveys and then by providing a summary for each approach.

Section 2.3, discusses the state of the art for Natural Language Processing models, which we exploit in our solution.

2.1 Cybersecurity

2.1.1 Background

Society, businesses and governments are increasingly relying on Information and Communication Technologies as new practical solutions emerge. Accordingly, cyber-

attacks are becoming more common to anyone who is relying on these technologies. The Global Risks Report of WEF [5] places cyber-attacks as the risk with the seventh highest global impact. Accenture on its latest report for the cost of cybercrime [6] estimates that security breaches grown by 67 per cent in the last five years and from 2019 to 2023 $5.2 trillion global value will be at risk from cyber-attacks.

A response to this threat is the turn to Cybersecurity. The International Telecommunication Union defines Cybersecurity as "the collection of tools, policies, security concepts [...] that can be used to protect the cyber environment and organisation and user's assets." [7]. In a sense, Cybersecurity aims to understand and take proactive measures against the cyber-attacks.

Cyber Attacks can have different targets. They can aim in the software, the hardware or the network of an IT infrastructure [8]. They are also diversified based on their purpose. They can aim for unauthorized access to a system, inappropriate usage, malicious activities or Denial of Service[1] [9]. Moreover, they can have as target a group of users, organisations, industries or governments [9].

These attacks can be seen as single events; however, the more sophisticated malicious activities can be seen as part of an overall strategy from the actor's side. Often, cybersecurity experts, refer to tactics, techniques and procedures (TTP) when they want to examine a strategy from a higher-level perspective.

[1] "Denial of Service attacks which come in many forms, are explicit attempts to block legitimate users' system access by reducing system availability" [125]

2.1.2 Tactics, techniques and procedures (TTP)

Although there is no a highly acknowledged definition for TTPs, the latter refers to "patterns of activities or methods associated with a specific threat actor or group of threat actors" [10]. In a sense, TTPs deal with how threat actors plan and execute their attacks.

TTPs can be considered as a collection of different types of knowledge for a specific threat agent. For example, TTPs may consist of:

- Documents that describe the strategies that the threat agent uses
- A list of specific domain names from webpages that the threat agent utilises
- Hash values[2] from the malicious files (such as a zipped file) that the agent shares to its targets

Several academic studies, cybersecurity companies [11] and organisations [12] refer to TTPs according to "The pyramid of pain" [13] (fig. 2) which was proposed by David J. Bianco. The illustration shows that the TTPs are comprised of several resources which may be easier or more difficult to be collected.

The TTPs are mainly created by cybersecurity companies but also from individual entities who want to archive or share their knowledge. TTPs, can help cybersecurity experts in several matters. Some examples, according to [14] are the following:

[2] "Hash values can be thought of as fingerprints for files. The contents of a file are processed through a cryptographic algorithm, and a unique numerical value – the hash value - is produced that identifies the contents of the file."[126]

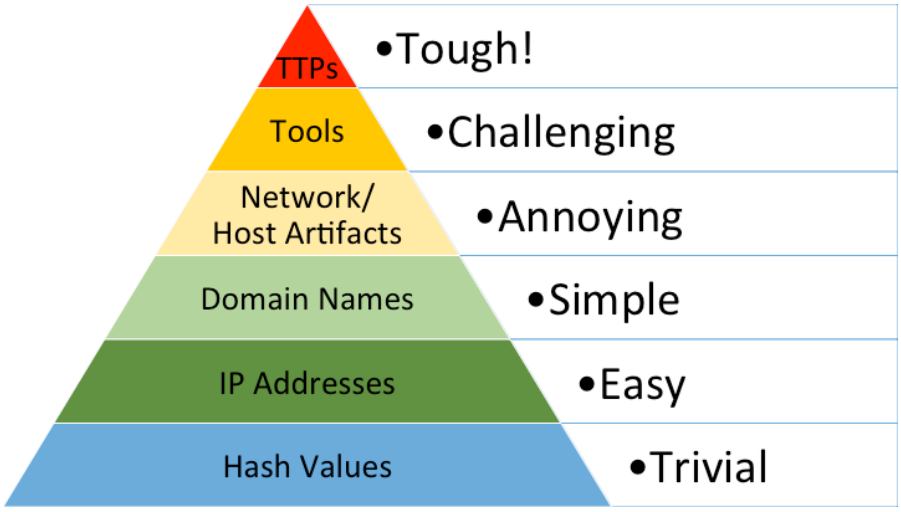


Figure 2: Pyramid of Pain, David J. Bianco, 2013

- Assess new suspicious activities in the IT infrastructure and relate them to TTPs of threat actors
- Provide structured knowledge to the security systems in order to help them identify potential threats
- Help security experts to take proactive measures but also evaluate the current state of their security systems against known attacks.

However, something that is challenging in the creation and potentially the distribution of TTPs is their structure. Cyber-security teams may follow a different framework to collect and synthesise the different forms of knowledge [15]. This issue in the long term may limit the actual benefits that TTPs can provide to a cybersecurity team.

For this reason, MITRE, an American not-for-profit organisation, with ATT&CK (Adversarial Tactics Techniques and Common Knowledge framework) project [16] aims to provide a comprehensive framework that can help security experts to identify, collect but also combine different sources of knowledge for the creation of their TTPs. Besides, MITRE with the STIX™ project (Structured Threat Information Expression) aims to define standards for sharing TTPs with other external partners.

TTPs are particularly useful when they are exchanged between different partners. For example, a TTP collection that describes the behaviour of a threat actor which targeted a bank in the past would be particularly useful for any other organisation within the same industry.

To this end, Threat Intelligence Feeds provide curated collections of TTPs. Threat Intelligence Feeds do not necessarily contain information only from the higher perspective of TTPs (where all the aspects of a threat actor are examined), but they can also provide information for single attacks.

2.1.3 Threat Intelligence Feeds

According to [17], threat intelligence refers to collections of evidence-based knowledge that describes mechanisms, context, implications and actionable advice about rising dangers for an IT infrastructure. A Threat Intelligence feed is not a prediction for a future danger but is based on existing incidents. As an incident, Gartner describes the result of a threat which exploited a vulnerability of an IT infrastructure [17]. (fig. 3) While threat intelligence feeds may contain broad knowledge about threat actors (for example a report or tools that an agent utilises) most feeds rely mainly on Indicators of Compromise (IOCs). According to [18], IOCs "serve as forensic evidence of potential intrusions on a host system or network. [...] Security researchers use IOCs to better analyse particular malware's techniques and behaviours. [...]". In a sense, IOCs describe verified malicious events.

Most of the well-known security companies such as Kaspersky [19] and Cisco [20] curate their own threat intelligence feeds for their customers. However, surveys [21] and studies [22] highlight that most consumers of intelligence feeds are not

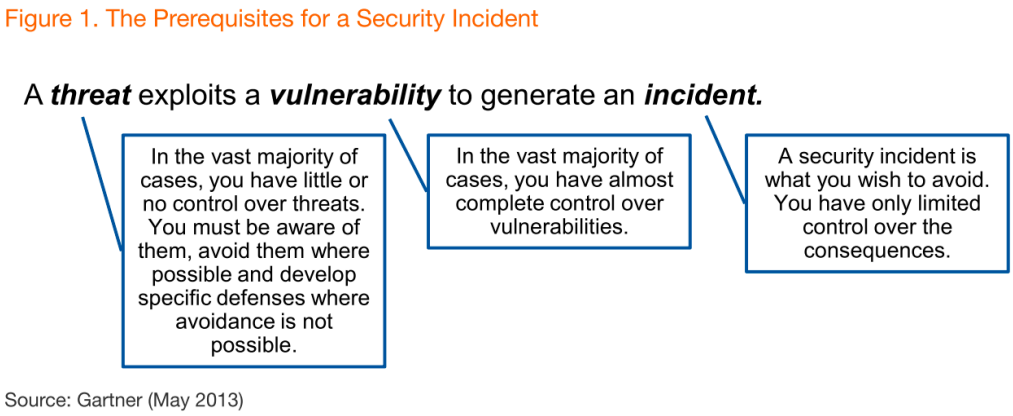


Figure 3: Definition of Threat Intelligence, Gartner, 2013

satisfied with their quality. Besides, most of the threat intelligence providers do not make their knowledge publicly available [17], [23]. However, there are a few open feeds available from some vendors and security organisations.

2.2 Data Labelling

In this section, we collect and categorise the main data labelling approaches. In order to be in alignment with our business problem, our study examines approaches which deal with textual data.

After a review of the available literature, we identified three primary sources which shed light in data labelling. These sources are the following:

- The Data Classification book [24]
- Work from Stanford's Hazy research group [25]
- A Survey on Data Collection for Machine Learning [26]

A more thorough analysis of these sources can be found in appendix A.

Next, we compare and synthesise the knowledge from these three sources.

2.2.1 A comparison of the three primary literature sources

The three literature sources cover the field of data labelling through different perspectives. In more detail, the book of "Data Classification" aims to give an overview of all the algorithms for solving classification and consequently data labelling problems. The Hazy Research team aims to solve problems related to data labelling with the use of Weak Supervision. Finally, the "Survey on Data Collection for Machine Learning" examines the problem of data labelling from a higher and a more commercial perspective.

In the next sub-chapters, we extend our analysis based in the "Survey on Data Collection for Machine Learning". We have chosen to continue our study with this survey for two reasons. First, because it provides a superset of the two other literature sources and secondly because it examines the topic from a commercial perspective. Below we provide a modified summary of the approaches proposed from the survey for data labelling:

I. Data with no labels
II. Manual Labelling (human-in-the-loop to annotate instances)
    a. Active Learning
    b. Crowdsourcing
III. Weak Labelling (no human-in-the-loop to annotate instances)[3]
    a. Data Programming
    b. Fact Extraction

[3] In order to evaluate the generated labels, we may need to label a small fraction of the unlabeled data manually. The reason we do this is clarified in the next chapters.

IV. Data with some Labels

V. Semi-Supervised Learning

VI. Previously labelled data

VII. Transfer Learning

2.2.2 Data with No Labels – Manual Labelling

Manually labelling refers to the methods were humans, or equivalently in literature "oracles", annotate entirely or partially a dataset. The first case refers to the Crowdsourcing concept [27]. For the second case, where humans partially annotate a dataset, specific strategies may be used (rather than randomly annotating a subset of the dataset). These strategies belong to the field of Active Learning [28].

Below we provide a summary for both approaches.

2.2.2.1 Crowdsourcing

Crowdsourcing outsources a labelling task to multiple workers who are not directly associated with it [29]. Related surveys [30], [31] also highlight the case where annotators with specific qualifications may be hired.

The produced labels from different annotators do not have to be same. Different approaches aim to synthesise and de-noise the multiple sets of labels [27].

A more thorough analysis of Crowdsourcing and its applications can be found in appendix B.

2.2.2.2 Active Learning

A survey [28] for Active Learning, describes it as a subfield of Machine Learning where learning algorithms use specific data to be trained. These algorithms actually,

request from humans to annotate instances which are least confident on how to label them.

According to B. Settles [32] and his sixth practical challenge, a side-effect of Active Learning is that the labelled instances are not an independent and identically distributed sample from the original dataset. In a sense, Active Learning assumes that the predictive model for our labelling task is already known and the labelling queries to humans are made according to that model.

In appendix C. we describe the ways that these algorithms pose queries and also we provide applications with Active Learning in the NLP domain.

2.2.3 Data with No Labels – Weak Supervision

For our problem of data labelling, Weak Supervision refers to the case where labelling is done in a semi-automatical way to produce labels. These labels are not as accurate as manual labels, where annotators label all of them.

In this setting, the authors distinguish two cases for creating weak labels; the case of Data Programming and Fact Extraction. Below we examine them in more detail.

2.2.3.1 Data Programming

Data Programming is the setting where users can generate weak labels with the use of "Labelling Functions" [33]. These functions, consider heuristics that are defined by domain experts, knowledge bases or even previous labels from an annotator. As these multiple Labelling Functions can be noisy, overlapping or conflicting, a generative model aims to give different weights to them.

Successful applications with Data Programming can found in appendix D.

2.2.3.2 Fact Extraction

Fact Extraction belongs to the broader field of Information Extraction. It aims to create weak labels with the use of web sources (such as an HTML web page) or extract relations from knowledge bases which already contain information of our interest [26].

2.2.3.2.1 Web Sources

For creating labels from web sources, web wrappers are used; procedures that extract information from unstructured or semi-structured web sources, transforming it to structured data [34]. In a sense, web wrappers are programmes which retrieve web sources in an automatic way, analyse the content of them and extract valuable information for the user.

2.2.3.2.2 Knowledge Bases

A knowledge base is a database which contains a collection of instances. Typically, these instances refer to "common" knowledge of the real world. In most cases, these records are registered in the database in the form of the triple "subject-predicate-object" [35].

For the data labelling setting, knowledge bases are used to extract information, or respectively labels, which are not available in our dataset. For example, for a dataset with socioeconomic indicators of cities, we could easily extract from a knowledge base the country where they belong to.

A more thorough analysis for these approaches and their applications with NLP can be found in appendix E.

2.2.4 Data with Some Labels – Semi-Supervised Learning

According to book "Semi-Supervised Learning" [36] the Semi-Supervised algorithms deal with problems where we have access to a large number of instances, but only few of them are labelled. In this setting, the semi-supervised algorithms aim to "learn" through the labelled subset and then generalise the classification task to the rest unlabelled subset.

A requirement for using such approaches is that the distribution of the unlabelled subset has to be relevant to the labelled data. Moreover, for each approach, additional assumptions may have to hold.

Approaches for dealing with such settings as well as applications in the NLP field can found in appendix F.

2.2.5 Previously Labelled Data – Transfer Learning

However, there are settings where the required assumptions for Semi-Supervised Learning do not hold. For example, in the case where the subsets of labelled and unlabelled data are not drawn from the same feature space or distribution. In this occasion, Transfer Learning approaches may be exploited [37].

Transfer Learning aims to extract knowledge from previous models (known as "source task" in literature) and apply it to a new "target task", which is in our case is the data labelling [38]. To understand better this concept, we share an example from real-life: For a person who already knows to code in Java, it will be easier to learn to code in Python, as both languages are object-oriented. In this case, the person transfers its knowledge to the new "target task".

The main aspects of the various Transfer Learning approaches and their applications in the NLP field can be found in appendix G.

2.2.6 Overview of the proposed approaches

In this part, we provide a summary of the examined approaches for data labelling. The following summary is not exhaustive and does not consider mixtures of approaches. Table 1 examines these approaches under five criteria:

i. Human annotation effort: How much effort is needed from humans to generate ground-truth labels
ii. Assumptions need to hold: Which refers to mathematical assumptions (such as smoothness) about the dataset that we want to annotate
iii. Requires previously annotated data: If for our data labelling task, we rely on previously labelled data
iv. Relies in supervised predictive model: Approaches which first train a model and then predict the labels of the instances that we want to annotate
v. Applicability: A subjective estimation based on our research and the applications that we have found in the NLP domain.

	Human annotation effort	Assumptions need to hold	Requires previously annotated data	Relies in supervised predictive model	Applicability
Crowd-Sourcing	Full	No	No	No	Wide
Active Learning	Partial	No‡	No	Yes	Wide
Weak Supervision	None*	No	No	No	Wide/Very Limited §
Semi-Supervised Learning	Partial†	Yes	Yes	Yes	Limited
Transfer Learning	Partial†	No	Yes	Yes	Limited

* None in these approaches, means that hand-labelling is not mandatory, however, human annotators may label a part of the dataset in order to evaluate the produced labels.

† Partial for Semi-Supervised and Transfer Learning refers to previously labelled data or predictive models that are used to label new instances.

‡ For Active Learning, there are no specific assumptions to hold, although Active Learning assumes that we know in advance which predictive model best fits our data.

§ Very Limited refers to Fact Extraction approaches (Web Sources or Knowledge Bases)

Table 1: A Comparison of the Data Labelling Approaches

2.3 State-of-the-Art for Natural Language Processing

2018 was a breakthrough year for NLP. Models such as ELMo [39], OpenAI GPT [40] and BERT [2] surpassed the human baseline performance in language task benchmarks such as GLUE [41] and SuperGLUE [42] (fig. 4).

In our work, we use BERT [2] and RoBERTa [43] language models. BERT is now widely acknowledged in academia, and Google has already started using it for improving its search results [44]. RoBERTa, developed by Facebook's Research

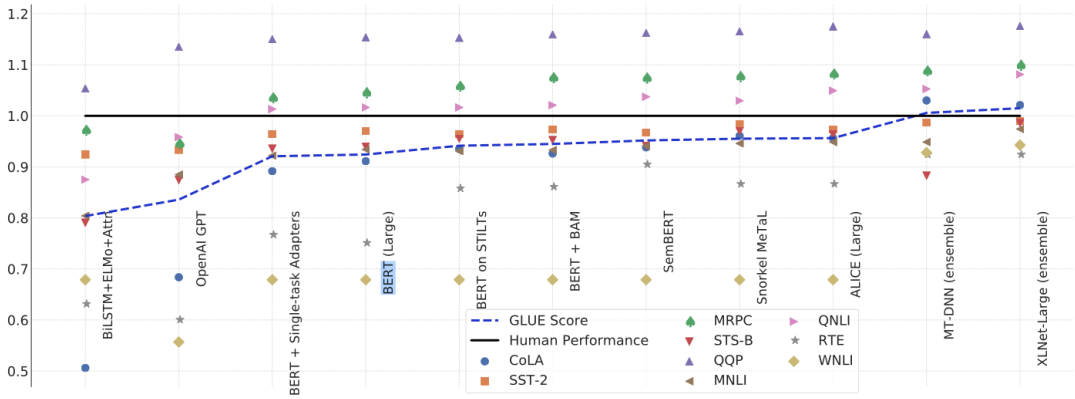


Figure 4: GLUE Benchmark for different NLP models

Team, is based on BERT's open-sourced architecture, but it further fine-tunes BERT's pre-training phase.

In our work, we use BERT [2] and RoBERTa [43] language models. BERT is now widely acknowledged in academia, and Google has already started using it for improving its search results [44]. RoBERTa, developed by Facebook's Research Team, is based on BERT's open-sourced architecture, but it further fine-tunes BERT's pre-training phase. Below we provide a summary for these models.

2.3.1 Bidirectional Encoder Representations from Transformers (BERT)

BERT, developed by Google, is a pre-trained model, which means that is created from an unlabelled Wikipedia and book corpus (16GB size in total [45]). This provides it with a broad knowledge of how English language is structured but also how words are used in different contexts.

However, the main advantage of it is that is "deeply bidirectional" thanks to its neural network Transformer architecture [46]. In a sense, this means that during training, it creates representations of words in relation to the whole context that they belong to (considering both previous and following words).

These representations are achieved with two strategies. The first is the Mask Language Model which masks 15% of words from different sentences, and the model aims to find the missing words for them. The second is the Next Sentence Prediction, where BERT, aims to predict for different sentence pairs if the second sentence is the sequence of the first one. Both tasks are pre-trained concurrently on the Wikipedia and book corpus, and the goal of the model is to minimise the combined loss function for them.

2.3.2 A Robustly Optimised BERT Pretraining Approach (RoBERTa)

RoBERTa, developed by Facebook, is a refined version of BERT model which focuses on a more efficient pre-training of the model. Its language corpus is 160GB [45], which is approximately ten times bigger than BERT. Moreover, during the pre-training phase, the Next Sentence Prediction task is omitted. Finally, RoBERTa, is using a dynamic masking pattern for the Mask Language Model and is trained in longer sentences.

RoBERTa as a single model has now the seventh highest overall ranking in the leaderboard of SuperGLUE benchmark, surpassing that of BERT's in all tasks.

Chapter 3 Design

This chapter is organised as follows:

Section 3.1, revisits our business problem and relates it with the background and literature review of chapter 2.

Section 3.2, examines the open cyber intelligence feeds, the data source that we exploit in this study.

Section 3.3, describes the data labelling method that matches to our business problem and data source.

Section 3.4, describes the NLP model distribution that we use in this study.

Section 3.5, provides the overall structure of our solution.

3.1 A revisit to the Business Problem

For our business problem, we explore threat intelligence feeds and we generate new labels for them. In more detail, we use open cyber intelligence feeds that contain short descriptions and external references (web pages) that describe either single attacks or the Tactics, Techniques and Procedures (TTPs) of threat actors. For these text sources, we generate new labels, by exploiting one of the available data labelling approaches. These labels aim to provide additional information for the attack or the

threat agent. For example, in our solution, we create a label that indicates if a threat agent has espionage intentions with its activities.

In a sense, instead of annotating all of the text sources, we use a data labelling technique in order to create labels (y) for two types of documents. Next, with Natural Language Models, we create a predictive model that can classify any description or analysis that is available in free text and provides information for an attack or a threat agent.

Ultimately, the generated labels for the cyber-attacks or the TTPs will offer to security experts a classified threat intelligence feed according to some topics.

3.2 Open Cyber Intelligence Feeds

The main criterion for selecting our open intelligence feed was to provide descriptions of attacks or TTPs and links to web pages. Besides, we would like have a feed that describes recent incidents.

AlienVault Open Threat Intelligence Portal has an active community of 100,000 participants [47], where users contribute their own cyber intelligence. Many of the intelligence feeds found on AlienVault portal provide a description but also external links.

Except AlienVault, we considered other open threat intelligence feeds. All of them can be found in appendix H. However, none of them was as informative as those that appear in AlienVault's portal.

3.3 Data Labelling approach of our solution

The data labelling approach we have chosen is highly connected with our business problem. The cyber-security team does not have any previous labelled sample of threat feeds, nor a previous classifier for these. Also, the cyber-intelligence team would like to create labels that are independent of a specific predictive model. This means that it would like to have labels that can be used with any state-of-the-art language model, as the field of NLP is rapidly evolving from 2018.

By consulting the table 1 from section 2.2.6, we find that Crowdsourcing solutions require all instances be hand-labelled from annotators. Active Learning, according to [32] and section 2.2.2.2, is not suggested when we want to generate labels that can be used with different predictive models. Semi-Supervised Learning and Transfer Learning approaches cannot be used, as they rely on previously labelled data and predictive models. For Weak Supervision, Fact Extraction would require to examine all descriptions and web pages in detail as the text sources are written in various structures.

Finally, we ended up with the weak labelling approach of Data Programming. According to Ratner et all. [33] and section 2.2.3.1, Data Programming allows users to use different labelling strategies, or equivalently Labelling Functions, to concurrently to label data.

These Labelling Functions can use:

I. Simple heuristic rules that are expressed with logical conditions, for example:
    a. if the word “spying” appears in the description of a threat agent then label it as one with espionage intentions
II. Knowledge bases, for example:

a. if a name from a conglomerate appears in the description of a threat agent, then label it as one with espionage intentions.

III. Previously labelled data

IV. Previous predictive models

As the produced weak labels from Labelling Functions may conflict, a generative model aims to denoise them by giving them different weights.

Finally, the authors of Data Programming propose to train a discriminative model of our choice with the produced labelled dataset. In this way, the predictive model will generalise beyond the weak labels, possibly by placing weights on co-occurring features or by exploiting pre-trained semantic knowledge (for example word embeddings in NLP algorithms) [48].

The main benefit of using lower-quality weak labels in discriminative models is described in more detail in a study from Google AI research team [49]; as the number of weak-labelled data increases the generalisation error of the models decrease at the same asymptotic rate as it would if ground-truth labels have been used. In other words, Data Programming aims to develop high quantities of noisy labelled data which ultimately can lead to better predictive models compared to using only a small proportion of hand-labelled data.

From the same study, empirical results from Snorkel DryBell[4], showed that human-labelling efforts could only exceed the performance of weak labels after 85000 hand-labelled instances for a topic classification task and after 12000 hand-labelled instances for a product classification task (fig. 5).

Several companies have presented successful applications with Data Programming. Some examples include applications from Intel [50], Apple [51] and IBM [52].

[4] A framework that leverages Data Programming

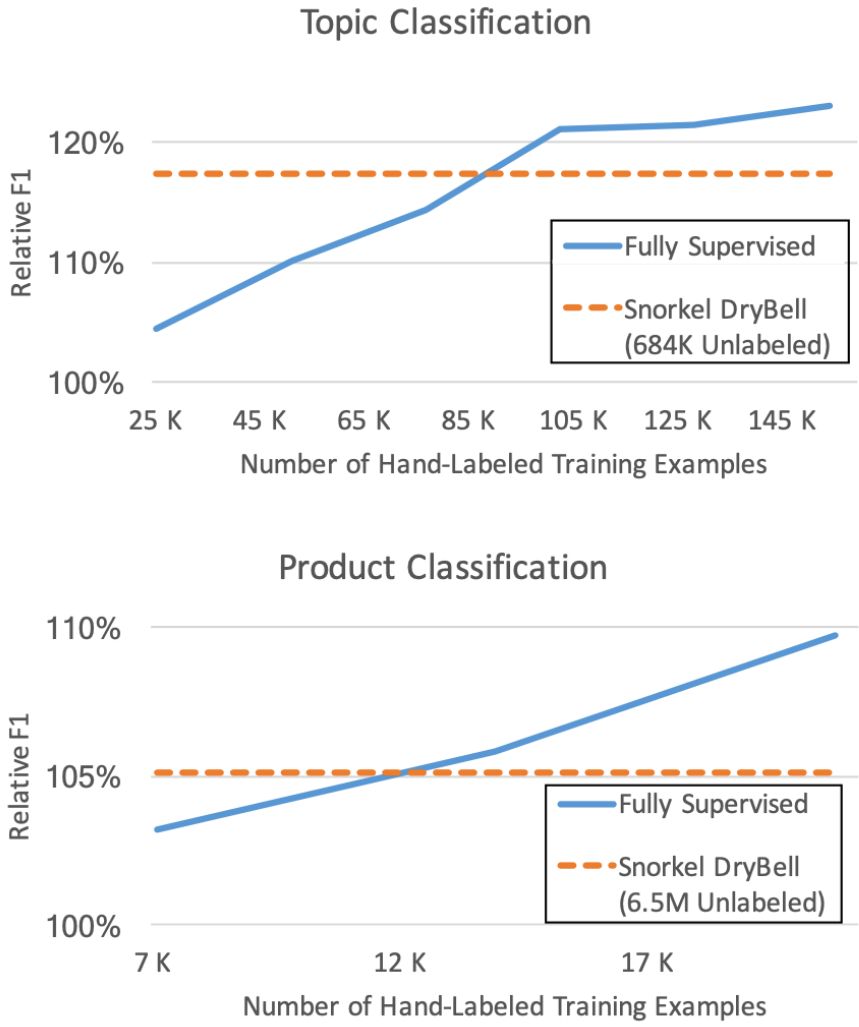


Figure 5: Empirical Results from Snorkel DryBell, S. H. Back et al., 2019

For our solution, we examined several frameworks in Python which leverage Data Programming. Some examples include Ddlite [53], Snuba [54] and Snorkel [55]. However, Snorkel proved to have extended functionalities and is the only library that is maintained currently.

Snorkel's Architecture is provided in figure 6. Snorkel, relying in Data Programming and in a more recent study for multi-task-aware models [56] aims to collect the different sources of weak signals, denoise them and provide a labelling model, without having access to ground-truth labels.

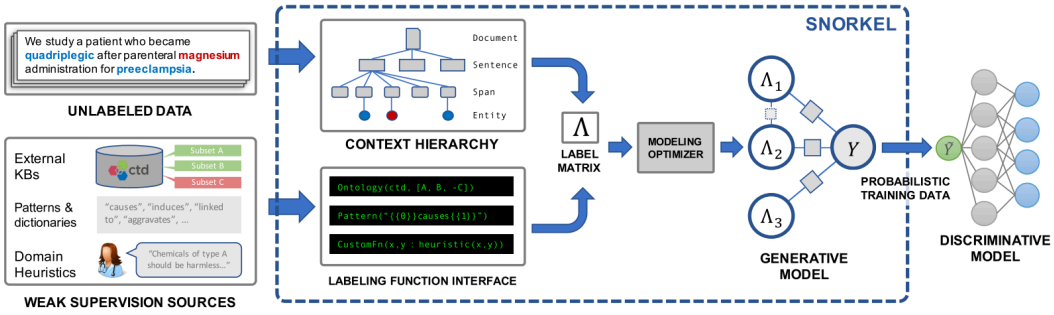


Figure 6: Snorkel's Architecture, Ratner et all. , 2017

3.4 NLP model distribution

In our solution, we use the Transformers Library from HuggingFace. HuggingFace is a team which delivers state-of-the-art NLP models in easy to use distributions. The NLP models provided by Transformers library can be trained with Tensorflow or PyTorch deep learning libraries. There are no great differences between them, although Tensorflow seems to be used more in deployment (deliver real applications) while PyTorch more in research [57]. In our solution, we use the PyTorch library.

3.5 Overall Solution Design

Our solution in chapter 4 consists of the following stages (fig. 7):

1. The selection of cyber intelligence feed, its exploratory data analysis and the definition of training, development and test subsets.
2. The hand-annotation of the development set and the selection of the final labels for our solution.

3. The creation of labelling functions (LFs) for these labels.
4. The evaluation of the labelling functions (LFs) that we have created and the hand-annotation of the test subset.
5. The selection of labelling functions (LFs) based on our test subset and the creation of our weak labels for the training subset.
6. The training and evaluation of BERT and RoBERTa language models with the use of the weakly labelled training set for short descriptions.
7. The web scraping of the external references (web pages), the definition of training and test subsets and the hand-annotation of the test subset.
8. The selection of labelling functions (LFs) and text window based on our test subset. The creation of weak labels for the training subset.
9. The training and evaluation of BERT and RoBERTa language models with the use of the weakly labelled training set for web pages.

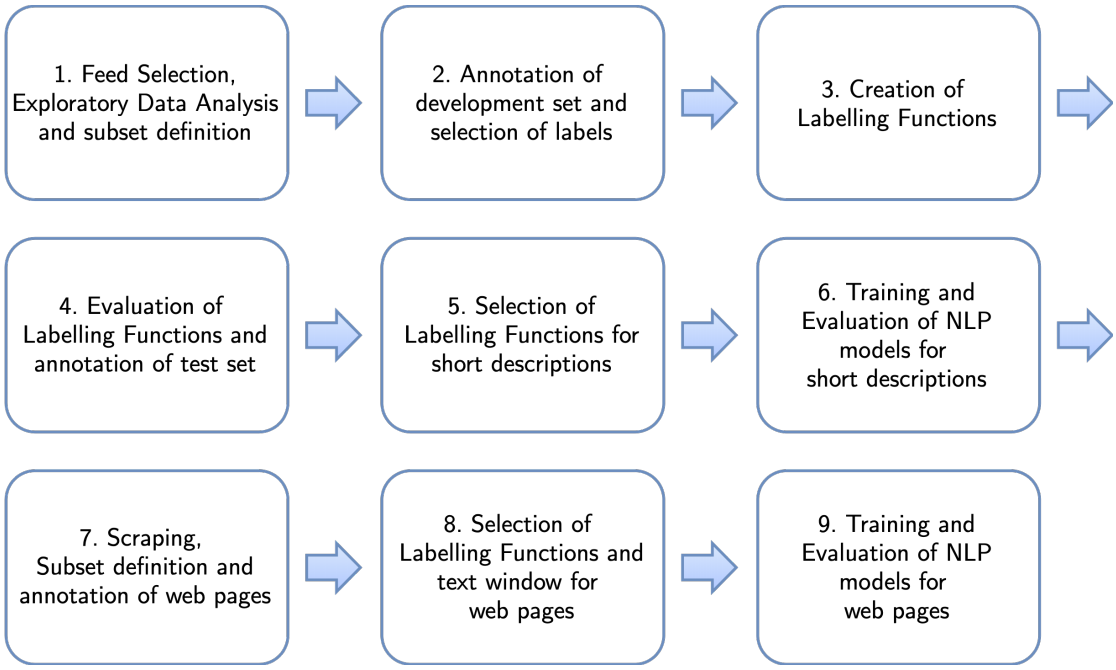


Figure 7: Overall Solution Design

Chapter 4 Solution

In this chapter, we provide our solution as it was described in the section 3.5.

The files and the dataset of our solution can be found in the following repository: https://github.com/mgmtbasnorkel/dissertation

4.1 Feed Selection, Exploratory Data Analysis and subset definition

In this stage, we:

1. Collect our candidate intelligence feeds through AlienVault's API.
2. Calculate basic assessment criteria and select the final intelligent feed.
3. Perform an Exploratory Data Analysis.
4. Remove records that are not satisfying specific criteria (data cleaning).
5. Define development, test and train subsets for short threat descriptions.

4.1.1 Candidate Intelligence Feeds

For choosing our candidate intelligence feeds, we visited the portal of AlienVault[5]. Our aim was to find collections which contain a large number of records. The platform allows to subscribe to other users who provide their own threat intelligence collections. Finally, we selected the seven accounts with most records (pulses) (fig. 8). It is noteworthy that one of the accounts is the official intelligence feed provided by AlienVault.

The collections provided by AlienVault are not directly available for download through its website. Users, have to generate their own API key and retrieve the collections of their interest through a programming language.

The A-get_subscriptions_job.py script file, connects to the API of AlienVault and retrieves the records from the collection that we are subscribed to. The total retrieval time for each collection can be found in appendix I.

4.1.2 Calculate assessment criteria and select our final intelligent feed

In B-Select_and_compare_feeds.ipynb notebook, we load all the intelligent feeds that we had collected from AlienVault and we calculate quality metrics for them.

[5] otx.alienvault.com.

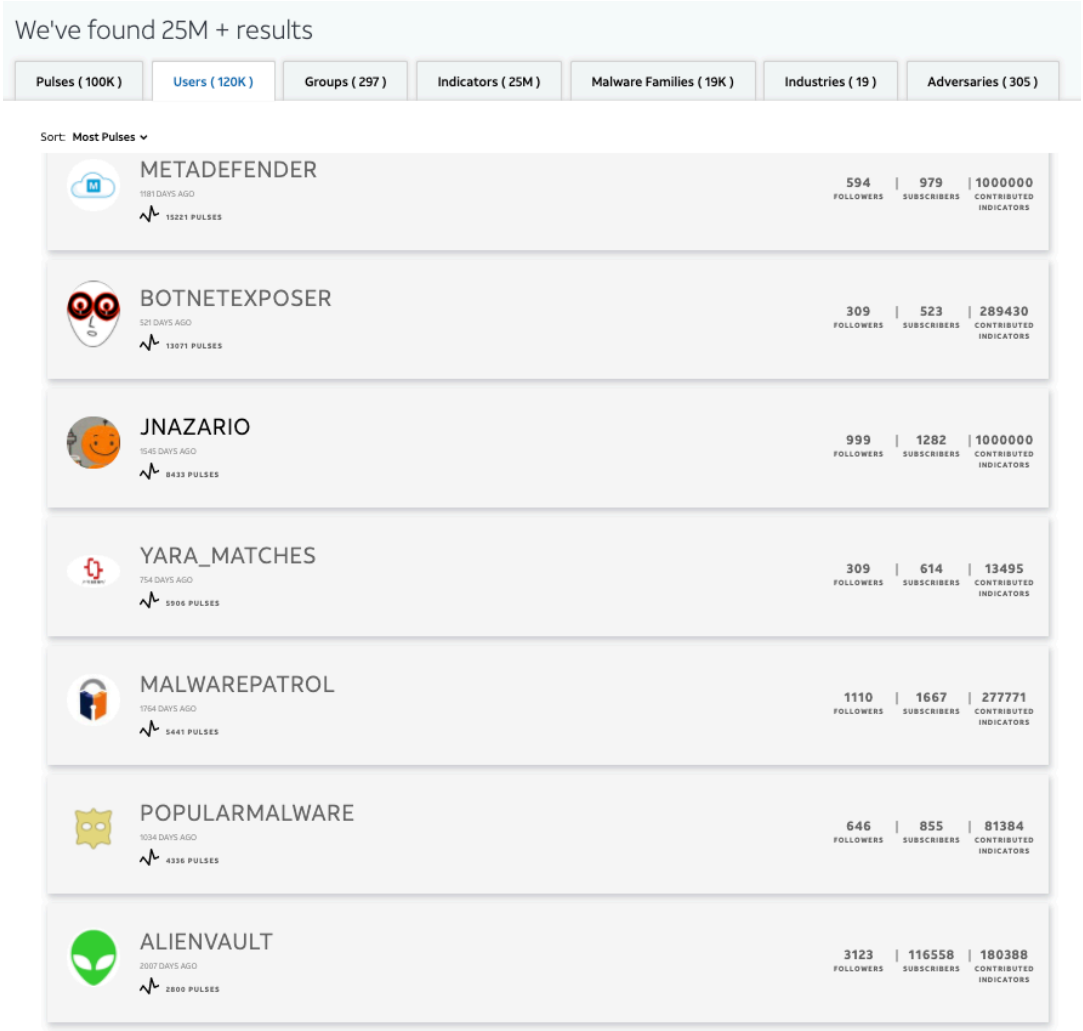


Figure 8: The accounts that we have subsribed to in AlienVault

In more detail, we extract for each collection, the date of the first and last record and we calculate the per cent of missing values for threat descriptions and external links. In addition, we calculate the average number and the standard deviation of words in the short descriptions. Appendix J. provides these metrics accompanied by the number of total subscribers in the AlienVault Portal.

After an agreement with the cyber-security team, we decided to proceed with AlienVault's feed, as it has provided adequate scores across all metrics.

4.1.3 Exploratory Data Analysis

In C-EDA.ipynb notebook, we perform an exploratory data analysis in the threat collection of AlienVault. The collection consists of 2670 records and 19 columns. The records describe the overall tactics, techniques and procedures (TTPs) of a threat agent or they focus on a single cyber-attack. Every record in AlienVault is referred to as "pulse".

4.1.3.1 Initial Exploration of Attributes

Below we provide a description for each column of the dataset (fig. 9):

- Industries: Refers to affected industries. More than 2000 records do not have a value for this attribute.
- TLP (Traffic Light Protocol): "a set of designations used to ensure that sensitive information is shared with the appropriate audience" [58]. It can take four values (white, green, amber, red). It's an indication which shows the level of confidential information the attack has access to.
- Description: It describes the attack or the agent briefly. Throughout our solution, we deal extensively with this attribute.
- Created: The date when the pulse submitted
- Tags: Arbitrary keywords that describe the pulse. Around three-quarters of the dataset contain at least one tag
- Malware Families: Attacks may use previous malicious code, or they may belong to a broader family. This attribute provides this information. However, very few pulses (around three hundred) contain such information.
- Modified: The last date when the pulse modified
- Author Name: The user on AlienVault who submitted the pulse
- Public: A redundant column, as all pulses have a value equal to one

- Extract Source: A redundant column, as all pulses have no value
- References: External web sites, pdf documents or tweets that describe in more detail a pulse. We deal extensively with this source in the rest of our work.
- Targeted Countries: Refers to the countries that are affected. Less than 600 records have a value for this attribute.
- Indicators : A collection of Indicators of Compromise (IoCs). The IoCs provide hash values for malicious files or other information.
- Attack IDs: There is no available information for this attribute in the portal of AlienVault. Most pulses do not have value for this attribute.
- More Indicators: A redundant column, as all pulses have False value
- Revision: The number of revisions for the pulse. Around 800 pulses have at least one revision.
- Adversary: Group of people that follow a specific malicious strategy. When a pulse is associated with a specific group, its name is provided. More information about malicious groups can be found in MITRE documentation[6]
- ID: A unique ID provided by AlienVault
- Name: The name of the pulse.

Overall, we speculate that these attributes are not exhaustive. By studying different external references (web pages) for a pulse, we may find information for an attribute that is not provided in the dataset.

[6] https://attack.mitre.org/groups/

	industries	tlp	description
1	[Technology]	white	In February 2020 and May 2020, Zscaler observed four malicious macro-based Microsoft Word documents hosted on newly registered sites with top-level domains of .space and .xyz. We attribute these attacks to the same threat actor due to the similar tactics, techniques and procedures (TTPs) used to deploy the final payload.\n\nThe final .NET payload, to the best of our knowledge, has not been observed in the wild before. It has a small code section in it that overlaps with the QuasarRAT. However, this code was not used at runtime. We have assigned the name - ShellReset to this RAT based on the unique strings found inside the final payload.

	created	tags	malware_families	modified	author_name	public	extract_source
1	2020-06-01 14:35:25.141	[QuasarRAT, ShellReset RAT]	[ShellReset RAT, QuasarRAT - S0262]	2020-06-01 14:35:25.141	AlienVault	1	[]

	references	targeted_countries
1	[https://www.zscaler.com/blogs/research/shellreset-rat-spread-through-macro-based-documents-using-applocker-bypass]	[]

	indicators	attack_ids	more_indicators	revision	adversary	id	name
1	[{'indicator': 'datacoup.com', 'description': '', 'title': '', 'created': '2020-06-01T14:35:26', 'is_active': 1, 'content': '', 'expiration': None, 'type': 'domain', 'id': 2260344996}, {'indicator': 'consumerspost.xyz', 'description': '', 'title': '', 'created': '2020-06-01T14:35:26', 'is_active': 1, 'content': '', 'expiration': None, 'type': 'domain', 'id': 2260344997}, {'indicator': 'documentsharing.space', 'description': '', 'title': '', 'created': '2020-06-01T14:35:26', 'is_active': 1, 'content': '', 'expiration': None, 'type': 'domain', 'id': 2260344998}, {'indicator': 'centeralfiles.xyz', 'description': '', 'title': '', 'created': '2020-06-01T14:35:26', 'is_active': 1, 'content': '', 'expiration': None, 'type': 'domain', 'id': 2260344999}, {'indicator': 'theashyggdrasil.xyz', 'de...	[T1059, T1060, T1064, T1083, T1113, T1127]	False	1		5ed5122d4b46071c940688ab	New RAT in Macro-Based Docs Using AppLocker Bypass

Figure 9: First observation from AlienVault's feed

4.1.3.2 Analysis of attributes

Below we provide notable findings from the analysis of the different attributes.

TLP codes

All pulses have either white or green TLP code. This means that the attacks did not gain access to highly confidential information.

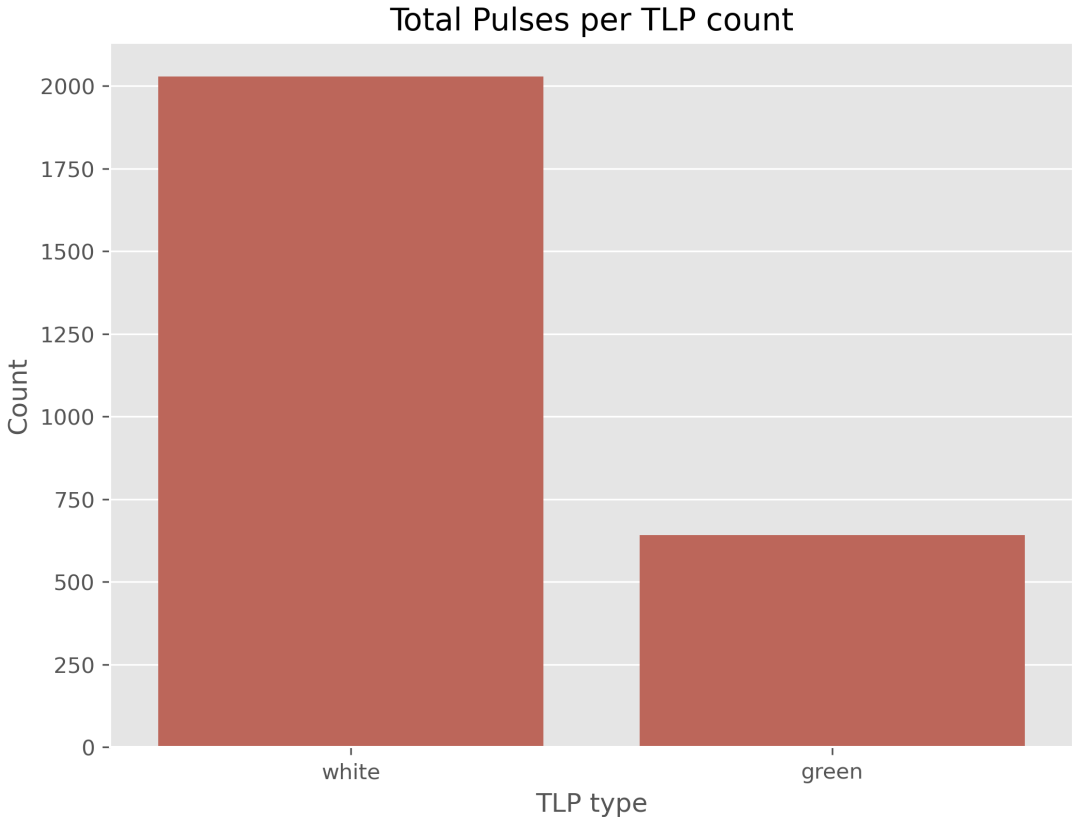


Figure 10: Total pulses per TLP code

Most common tags

Below we provide the ten most common tags among all pulses.

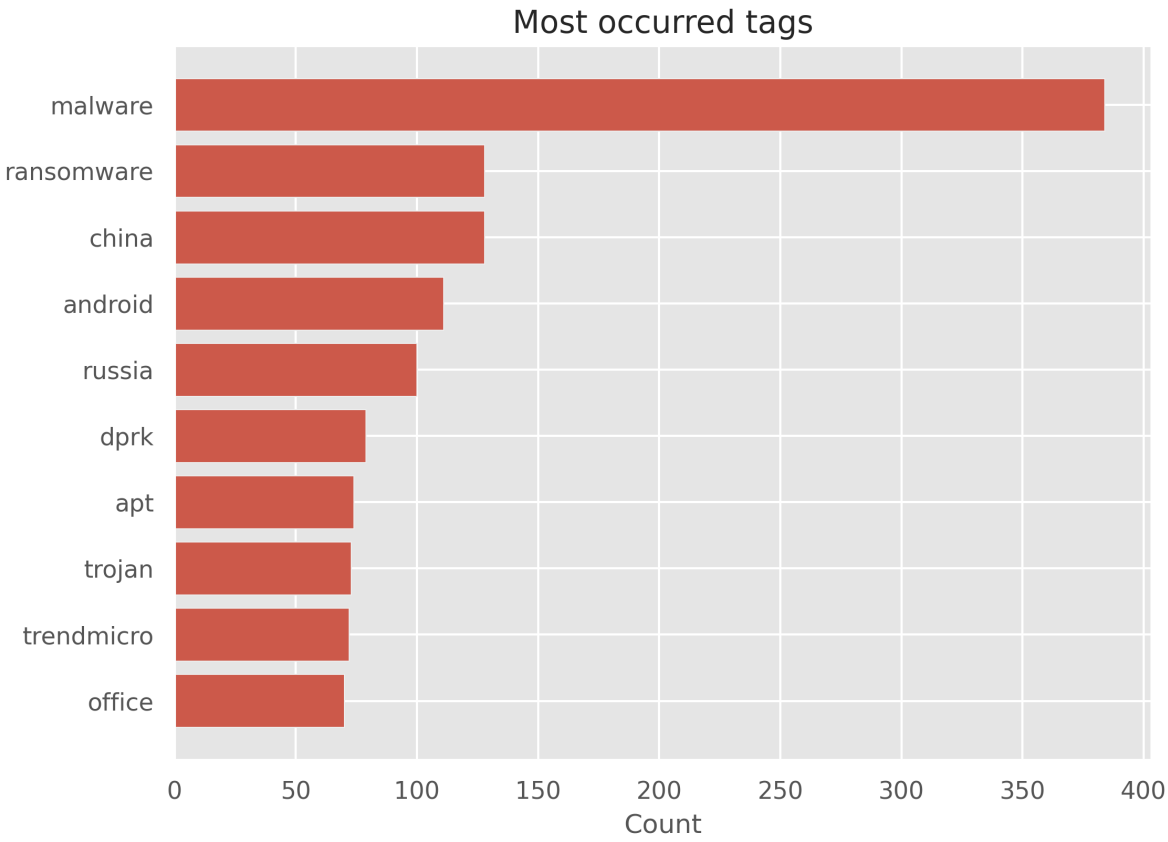


Figure 11: Most common tags for pulses

Cumulative number of pulses per day

Below we demonstrate that from late 2014, AlienVault has been updating its collection on a constant basis.

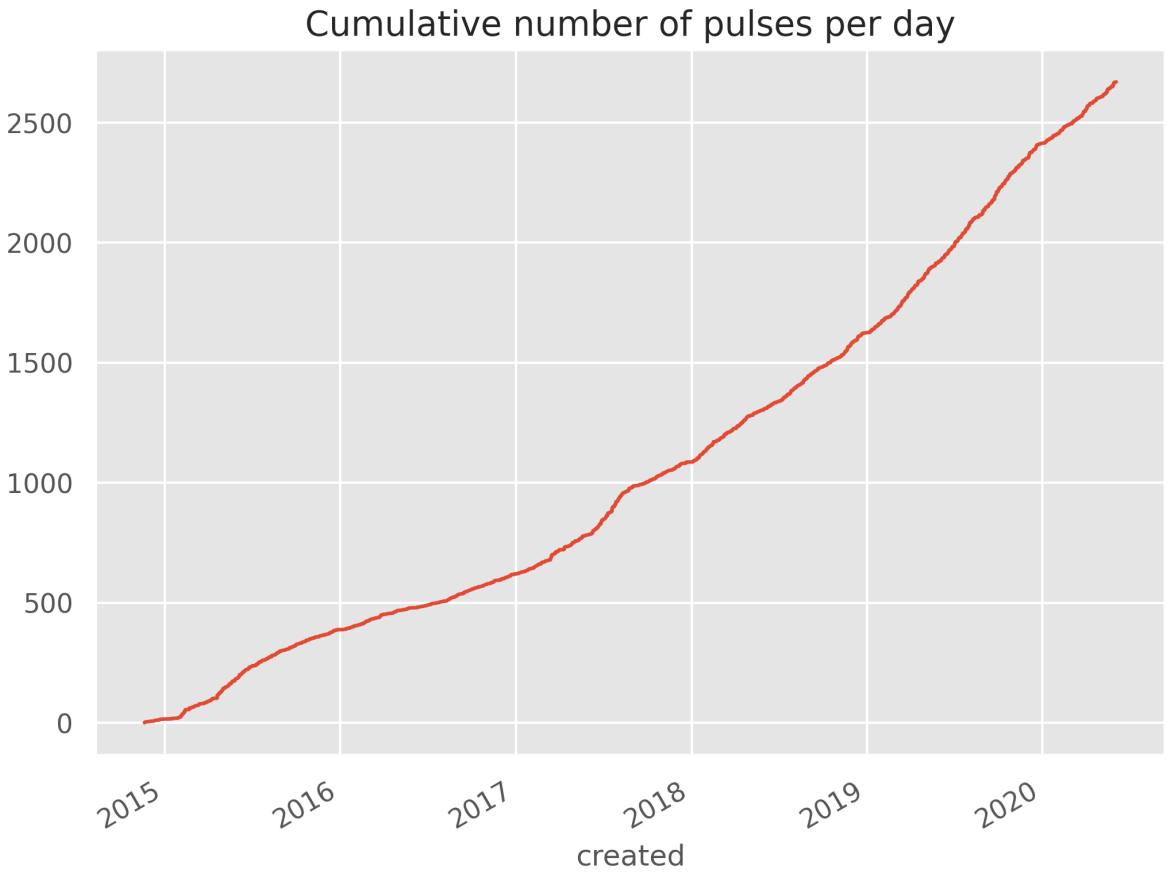


Figure 12: Cumulative number of pulses per day

Targeted Countries

Below we provide a map which shows how many pulses refer as target a given country. Pulses refer as the most targeted country the United States (104), following by South Korea (83) and Ukraine (51).

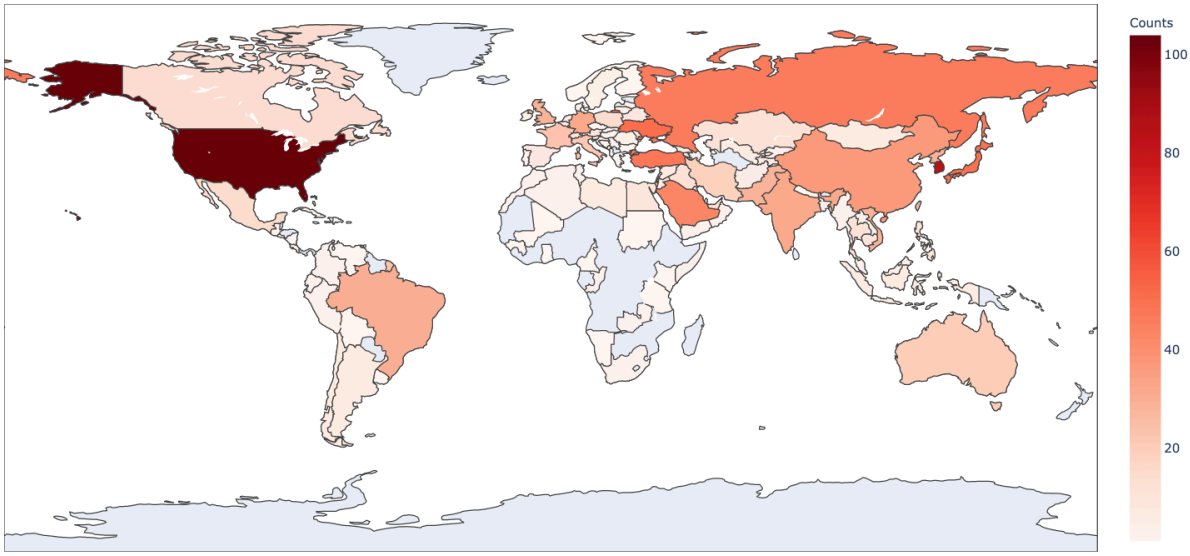


Figure 13: Number of attacks per country

Number of IoCs per type

Finally, we provide the types and the total number of different IoCs. Most IoCs provide hash values of malicious files.

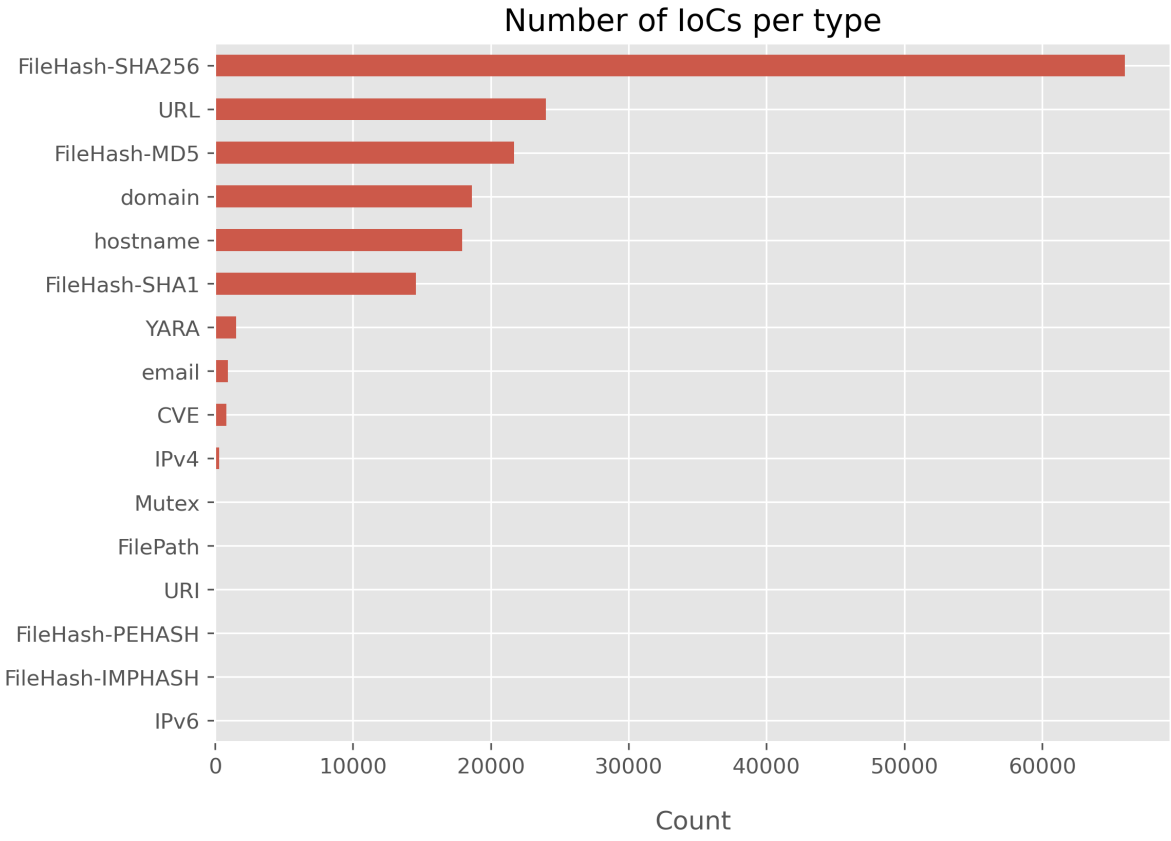


Figure 14: Number of IoCs per type

4.1.4 Data Cleaning

In the D-Subsets.ipynb notebook we proceeded with the cleaning of the dataset:

- From the 2670 pulses, we removed 121 records which had either missing descriptions or reference.
- From the 2549 remaining pulses, we removed 519 records which have a description of short length (less than 15 words).
- From the 2030 remaining pulses, we removed 11 records which have descriptions of significant extended length (more than 159 words). We performed this step as the size of the longest description defines the size of the matrix that we use in our NLP models (section 4.6). Trials have shown that the size of this matrix plays a detrimental role in the feasibility to train our models (as a bigger matrix requires more memory). Moreover, bigger matrices require more training time.

The cleaned dataset ended up with 2019 records.

4.1.4.1 Meta-analysis of attributes

Below we provide findings that come from the cleaned dataset. We focus mainly on the short pulse descriptions and the references links as we will deal with them extensively in the rest of our solution.

Pulses' Description Length

Below we demonstrate how the length of the pulse descriptions differs. The removal of descriptions with limited length has led to a collection that has on its vast majority ($> 95\%$), descriptions of more than twenty words.

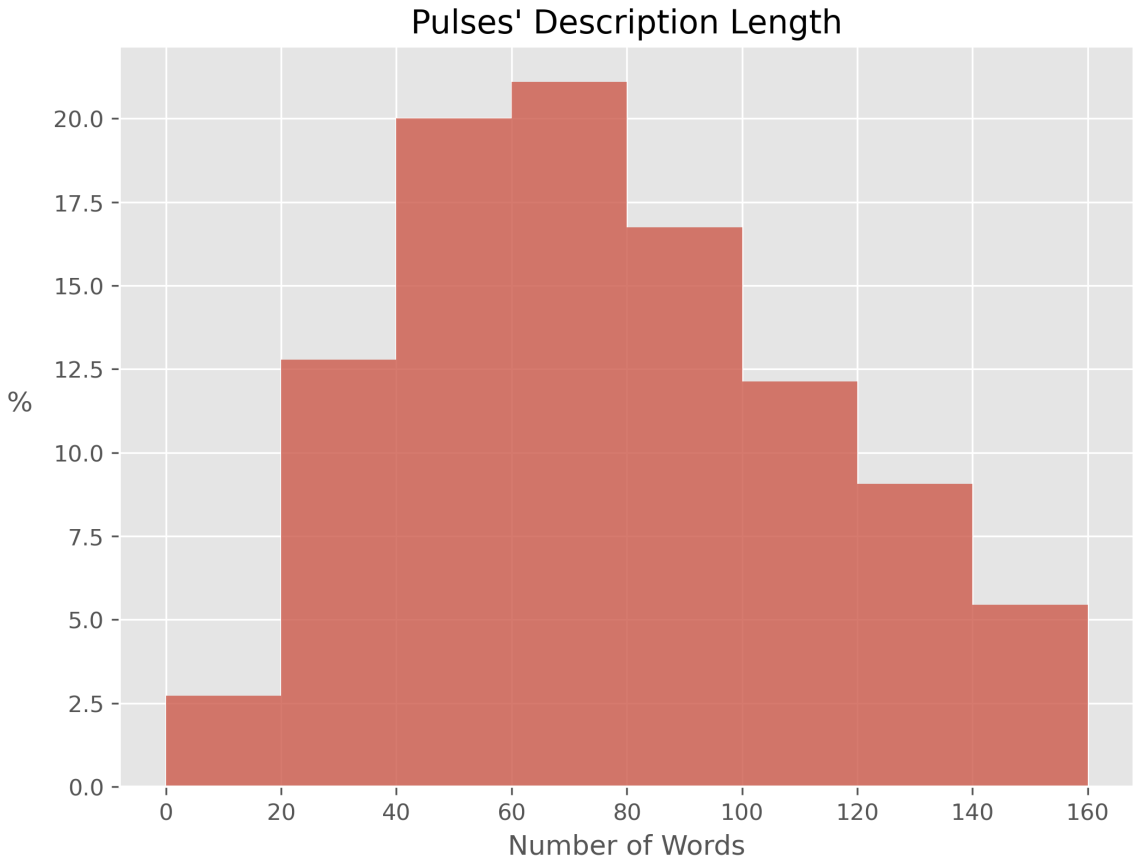


Figure 15: Pulses' description length for bins of a fixed size

Number of references per pulse

Most pulses have one external reference. References can be links to web pages, tweets or pdf documents. These references are further investigated in the next graph.

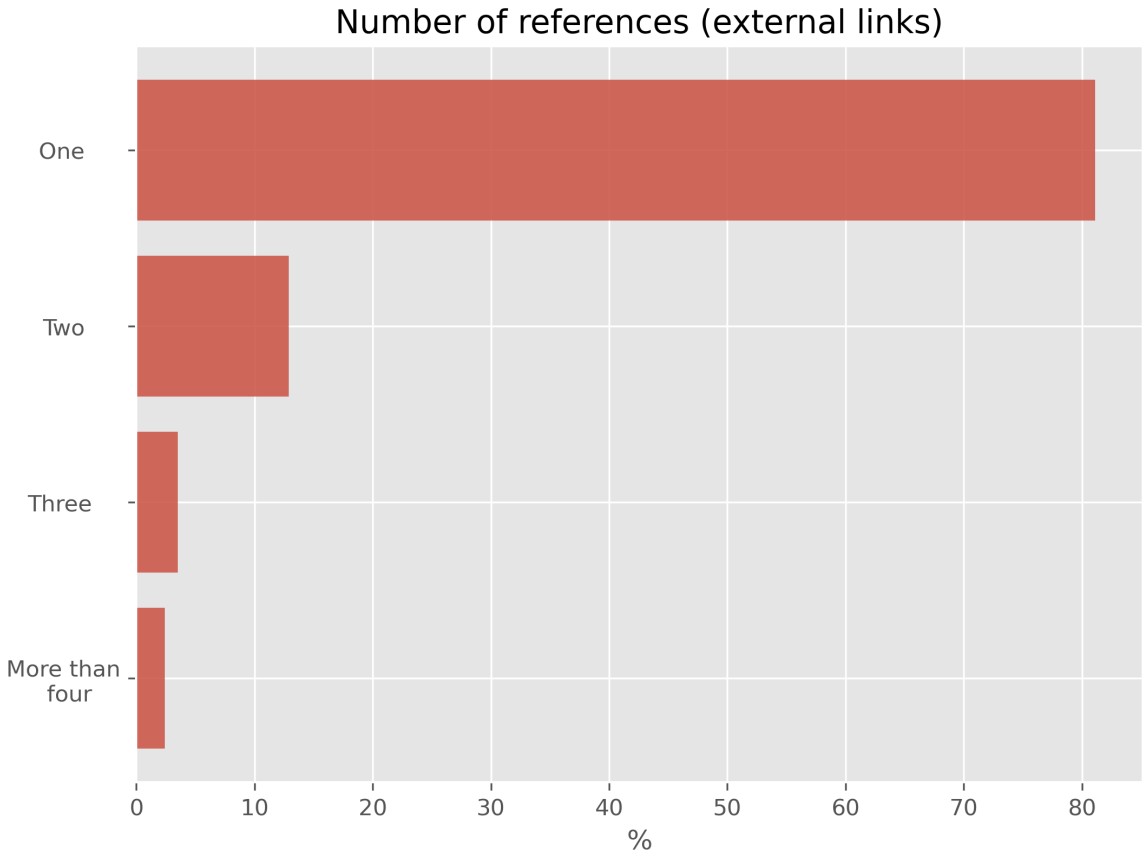


Figure 16: Number of pulses for different number of references

Types of references

Figure 17 presents the type of the first reference for each pulse. We distinguish four types of external references. These are the web pages in English, web pages in other languages, pdf documents and tweets. The different types of references are further investigated in section 4.7

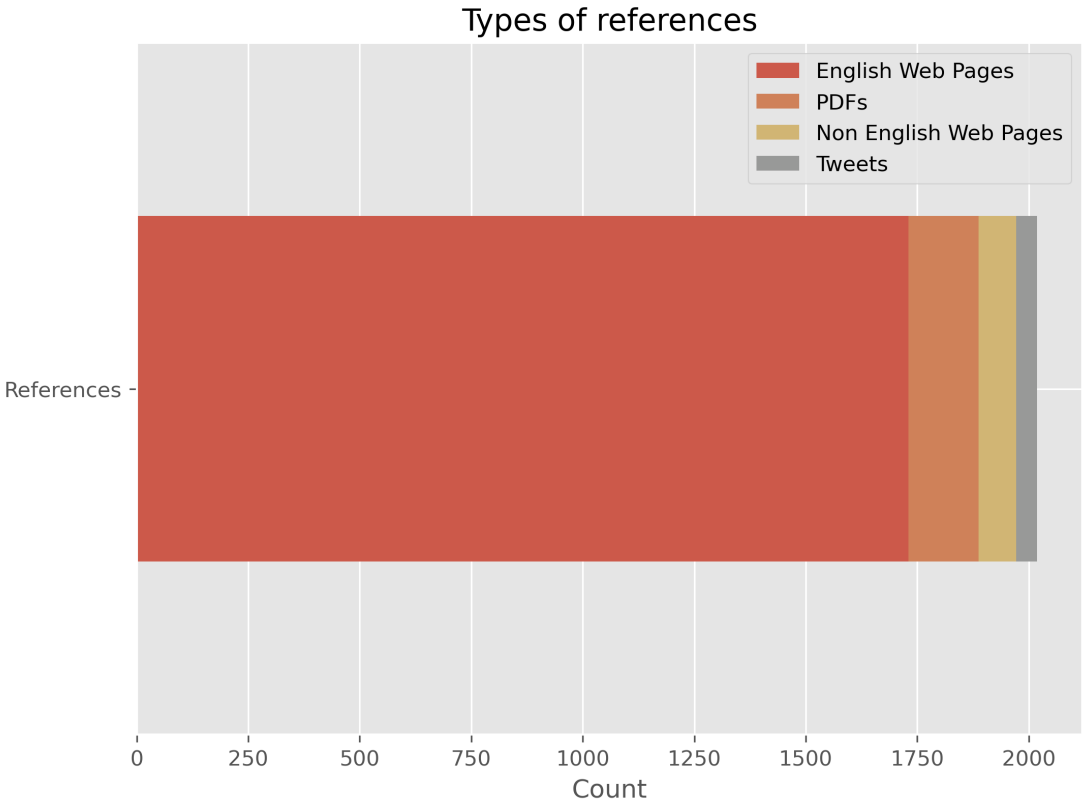


Figure 17: Types of references

4.1.5 Definition of subsets

From the final dataset of the 2019 records, around 12% will be used as development subset, 8% as a test subset and the rest 80% as training subsets. This regards only the case pulses' descriptions and not the references (web pages). The latter case is revisited in the section 4.7 "Web Scraping and subset definition". Finally, it is noteworthy that none of these subsets had previously annotated instances. Below we describe in more detail their actual usage.

4.1.5.1 Description of Subsets

Development Subset

This subset will be hand-annotated and is used mostly for the creation of Labelling Functions. For the hand-annotation of the first half of the dataset, we did not pre-specified any labels, but we tried to figure recurring topics that are discussed over all observations. This task is described in the next stage ("Annotation of the first 100 instances of development set"). The second half of this subset has been annotated only for the labels that we decided to examine in our solution.

Test Subset

Once we have finished the development of our Labelling functions, we hand-annotated the test subset. This subset is used to evaluate the different Labelling Functions that we have created up to the point of creating our final NLP models.

Train Subset

This dataset will be labelled from our Labelling Functions and Snorkel's generative model. Finally, the created weak labels y together with the textual descriptions (x), will be used as a training dataset for our NLP models.

4.1.5.2 Creation of Stratified Subsets

In the D-Subsets.ipynb notebook, we calculated the quantiles of descriptions' length (table 2).

Range of Words	Number of pulses
(16, 50]	530
(50, 73]	489
(73, 102]	502
(102, 159]	498

Table 2: Quantiles of descriptions' length

Then we created a new attribute which indicates in which quantile each description belongs to. This feature was considered in the creation of our subsets.

In more detail, for creating our subsets, we exploited a stratified sampling function which is open-sourced on GitHub [59]. The function creates strata based on categorical attributes. In our solution, these categorical attributes are the pulses' year creation, month and the quantile range of descriptions' length. We used these attributes as we wanted our development and test subsets to have a good representation of the dataset. A more thorough analysis of the different strata can be found in appendix J.

4.1.5.3 Final Subsets for pulses' descriptions

In figure 18 we provide an overview of the three subsets:

- Development Set (A): 278 observations
- Test Set (B): 167 observations
- Training Set (C): 1574 observations

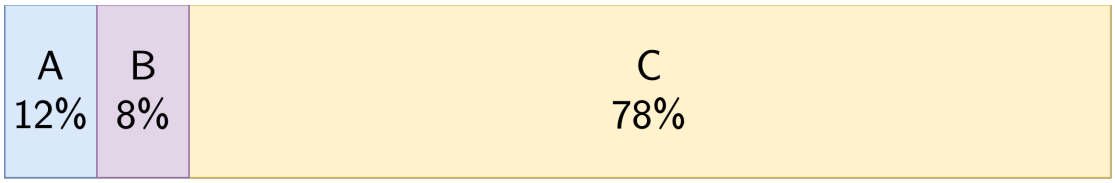


Figure 18: Final subsets for pulses' descriptions

4.2 Annotation of development set and selection of labels

In this stage, we describe how we annotated our development set and how we selected our final labels for our solution.

4.2.1 Annotation of the first one hundred instances

In the second stage of our solution, we hand-annotated one-hundred pulses' descriptions from the development set. At this phase, we were not aware of any specific label, but instead, we defined them while we were iterating through the different observations. Examples of our annotations can be found in appendix K.

At the end of the annotation we resulted in 17 labels, that can be found in table 3.

After a discussion with the cyber-security team, we decided to proceed with three of them:

- Targeted: If the description refers to an attack or an activity that targeted a specific entity or group of people

Label	Positive Rate	Label	Positive Rate (%)
Targeted	39%	Deal with cryptocurrencies	7%
Physical Target Known	35%	Physical Creator Known	7%
Instance refers to a previous attack	25%	Targets financial institutions	5%
Attack to IT infrastructure	12%	Targets media	4%
Targets a business	9%	Targeted to banks	4%
Attack with espionage intentions	9%	Targets energy companies	2%
Targets governments	8%	Targeted to critical infrastructure	2%
Targets Military	8%	Target Healthcare	1%
Aims to steal money	7%		

Table 3: Created labels from the annotation of the first one hunder instances

- Instance refers to a previous attack: If the malicious activity is based on previous malicious code, malware families or if the document describes a similarity with other previous attacks.
- Attack with Espionage Intentions: Attack which aimed to spy specific persons or organisations.

The selection of the labels was based first in the specific interests of our cyber-security team, but also on their diverging positive rates (39%, 25%, 9%).

4.2.2 Collection of keywords and annotation of the rest development set

Considering the three selected labels, we returned to our first one hundred instances, and we collected specific keywords and expressions that indicate a positive label.

For example, for the first label ('Targeted') verbs such as "targeting", prepositions as "against" or nouns as "company" indicated a positive label. In the same fashion, we collected several keywords and expressions for the rest labels. These were used to develop Labelling Functions which are covered in the next stage.

This process was also done concurrently while we annotated the rest observations of the development set.

The full list of keywords for each label can found in appendix L.

4.3 Creation of Labelling Functions

Below we describe how we defined our Labelling Functions (LFs) for the different labels. In a sense, a Labelling Function is a logical condition which labels an observation as positive or negative. Labelling functions may also abstain to label an observation.

In general, we classify the Labelling Functions into two categories. Functions that utilize attributes other than the actual text and functions that deal only with the actual text. The first type of functions, as it is prevalent, can only be used if the rest attributes are available during weak labelling. In our case, we did not refrain from using them as our final goal was to create a labelled-training set from the original threat feed.

4.3.1 Labelling Functions for "Targeted" label

All the LFs developed for this label were based only on the actual text. For some keywords, we declared LFs that first check if a specific word appears in the text and then label the instance as positive or negative.

For example:

```
@labeling_function()
def VERB_targets(x):
    return POSITIVE if re.search(r"targets.*", x.description, flags=re.I) else NEGATIVE
```

In this case, the labelling function will search for the verb "targets" and will classify as positive (1) the instance if the condition is fulfilled, otherwise will classify it as negative (0). The regular expression operations package from Python ("re" in code), searches dynamically throughout the text and also considers other forms of the same word.

In addition, we defined LFs which labels positive an instance if a keyword appears, otherwise, they abstain (-1) to label this instance.

For example:

```
@labeling_function()
def PROPN_focus(x):
    return POSITIVE if re.search(r"focus.*", x.description, flags=re.I) else ABSTAIN
```

The labelling function for this setting will search for the word “focus” and if it appears in the description will label it as positive. Otherwise, it will abstain to label it as negative.

All of our Labelling Functions had as a certain condition the positive label.

For deciding if an LF can label an instance as negative or refrain from doing it, we consulted the empirical results which are presented in the next section. It is noteworthy that empirical results had shown that LFs are better to label an instance as negative if a condition is not met. This is due to the Snorkel’s generative model which will consider the agreements or disagreements of the LFs (for the same instance) and will give a different weight to them in order to increase or reduce their signals. The topic is revisited in section 4.4.

Beyond the LFs which deal with specific keywords, we also defined two more advanced LFs. The LFs take advantage of the “spacy” NLP package and perform a named entity recognition (NER) in the document. In more detail, they search for mentions of geopolitical entities or languages.

Below we provide the example of the first case:

```
@labeling_function(pre=[spacy])
def SPACY_GPE(x):
    if any([ent.label_ == “GPE” for ent in  x.doc.ents]):
        return POSITIVE
 else:
        return NEGATIVE
```

In this case, if the document contains a mention to a geopolitical entity (GPE), then the LF classifies the description as positive (that indeed describes a targeted activity), otherwise it classifies it as negative (although it might target a different type of entity). Again, the empirical results which are provided in section 4.4 have shown that classifying the instance as negative, provides better labelling results.

All of the created LFs can be found in appendix M. and E-1_targeted.ipynb notebook.

4.3.2 Labelling Functions for "Refers to previous attack" label

For this label, we created LFs that rely on the textual description but also on other attributes of the dataset.

4.3.2.1 LFs based on other attributes

Snorkel suggests that users may use several resources to label the same instance. In order to do so, they may leverage aggregated statistics, that are derived from the dataset that we want to annotate. In our case, we selected the 76 positive instances of the development set, and we compared them with another 76 negative instances (randomly sampled).

For example, we have seen that between instances that refer to previous attacks and those that do not, the latter have a higher number of records with information for affected industries (industry label in the dataset). The finding is illustrated in table 4.

Refers to a previous Attack	Mention for affected industries in the dataset	Total Records	Percentage allocation of total records between positive and negative instances
Positive	Yes	15	15/(15+17)=46.80%
	No	61	61/(61+59)= 50.8%
Negative	Yes	17	17/(15+17)=53.20%
	No	59	59/(61+59)=49.2%

Table 4: A comparison between instances with mentions to industries

Although their difference cannot be considered great, we defined an LF that examines the industry attribute, and if it finds information on it, then labels the instance as negative. Similar to the above example, we created another four LFs that can be found in appendix M.

Finally, we created an LF that compares the years mentioned in the text with the creation date of the pulse (that can be found in the attribute "created" in the dataset). If a year before the creation year of the pulse is mentioned in the description, then the labelling function labels the description as positive (that indeed refers to a previous malicious activity).

4.3.2.2 LFs based on the actual text

Except for LFs that seek for specific keywords, we created one more labelling function that uses a list of known threat groups[7]. If the description mentions one of these groups, then the labelling functions classify the instance as positive.

Moreover, and only for the case of "refers to a previous attack" label, we created nine addition keyword-LFs after reviewing the initial empirical results which are

[7] https://attack.mitre.org/groups/

presented in the next chapter. The additional LFs of the second round has proved to improve the performance of our labelling model.

All of the LFs can be found in appendix M. and in the E-2_previous_attack.ipynb notebook.

4.3.3 Labelling Functions for "Espionage" label

For the case of espionage label, due to its rare positive class, we developed five labelling functions that seek for specific keywords. These LFs can be found in appendix M. and E-3_espionage.ipynb notebook.

4.4 Evaluation of Labelling Functions and annotation of the test set

In this stage, we evaluate the labelling functions that we have created based on the development set. In the end, we describe how the test set was labelled and its usage for the next stage ("Selection of Labelling Functions").

4.4.1 Coverage, Accuracy and the utility of Development Set in Snorkel

Coverage and empirical accuracy metrics that we exhibit in the evaluation results for each label are particularly important in Snorkel. Authors suggest that for high

performing labelling models, users should seek for LFs that have high coverage and empirical accuracy [60].

Coverage is the fraction of the instances an LF has labelled compared to the whole dataset. Empirical accuracy shows the percentage of the correct labels. While coverage is calculated without any ground-truth labels, empirical accuracy compares the predicted labels of Snorkel with the ground-truth labels that we have made (only for the development or test subsets).

The generated labels for any subset, are created in a weakly supervised way based only with the usage of the LFs. The hand-annotated subsets that we have created are used to evaluate our labelling model with ground-truth labels.

Finally, it is worth to mention that achieving both high coverage and empirical accuracy from a single LF, in reality is not possible. If a labelling function had an absolute coverage and maximum accuracy, then we wouldn't need a Machine Learning model, and we could rely only upon the logical condition that the labelling function uses [60].

4.4.2 Evaluation of LFs for "Targeted" label

4.4.2.1 A trivial example

Before we proceed with the evaluation of all LFs that were used for "targeted" label, we demonstrate a trivial example that uses five LFs with high accuracy. Table 5 shows their performance in labelling the development set and consists of the following columns:

- j: an index of the LF
- Polarity: if the LF is providing positive (1) and/or negative (0) labels
- Coverage: the fraction of the dataset that the LF has labelled
- Overlaps: the fraction of the dataset that this LF has labelled with at least one other
- Conflicts: the fraction of the dataset that this LF disagrees with at least one other LF
- Correct/Incorrect (based in development set): number of correct and incorrect produced labels
- Empirical Accuracy (based in development set): The fraction of the dataset with correct labels

	j	Polarity	Coverage	Overlaps	Conflicts	Correct	Incorrect	Emp. Acc.
VERB_stolen	15	[1]	0.023474	0.023474	0.023474	4	1	0.800000
NOUN_target	8	[1]	0.384977	0.384977	0.384977	63	19	0.768293
SPACY_GPE	20	[0, 1]	1.000000	1.000000	0.713615	153	60	0.718310
VERB_targeting	17	[0, 1]	1.000000	1.000000	0.713615	146	67	0.685446
VERB_steal	14	[1]	0.075117	0.075117	0.075117	10	6	0.625000

Table 5: The five LFs with the highest Empirical Accuracy

From the analysis of the LFs we see that VERB_stolen, which labels an instance as positive [1], has coverage 2.3%, which equals to 5 out of 278 observations of development set. Four from these labels were correct (empirical accuracy: 80%). In this case, the LF is predicting most of the times correct, but it gives labels only if the word "stolen" appears. In the case of LF SPACY_GPE, the function labels an instance as either positive or negative (coverage 100%) and in general has delivered 71% correct labels. It is evident that an LF with abstains, will have a low coverage;

something that can also leave an observation unlabelled if none LF can provide a label for an instance.

In table 6 we present a case, where all of these functions abstain if their condition is not fulfilled. In this case, LF VERB_targeting has surpassed VERB_stolen in accuracy, but its coverage has turned from 100% to 15%. Also, by turning the LF SPACY_GPE to abstain, we see that its accuracy remains almost the same (compared of labelling also negative), but its coverage drops by around 60%.

The actual gains of using LFs with negative labels are shown in table 7. For the two previous settings, we use a majority vote model and Snorkel's Generative model, which gives different weight to each LF. In both models, we see that using LFs with negative labels, provide better classification results. Again, we would like to highlight that this concerns our specific labelling problem. From our research in papers with

	j	Polarity	Coverage	Overlaps	Conflicts	Correct	Incorrect	Emp. Acc.
VERB_targeting	17	[1]	0.154930	0.154930	0.154930	31	2	0.939394
VERB_stolen	15	[1]	0.023474	0.023474	0.023474	4	1	0.800000
NOUN_target	8	[1]	0.384977	0.384977	0.384977	63	19	0.768293
SPACY_GPE	20	[1]	0.422535	0.422535	0.422535	63	27	0.700000
VERB_steal	14	[1]	0.075117	0.075117	0.075117	10	6	0.625000

Table 6: Labelling Functions that all abstain

	5 LFs with:	Accuracy	F1	AUC
Majority Vote Model	NEGATIVES	0.75	0.70	0.80
	ABSTAIN only	0.59	0.67	0.73
Snorkel's Generative Model	NEGATIVES	0.73	0.73	0.83
	ABSTAIN only	0.60	0.67	0.82

Table 7: A comparison beteween negative and abstaining Labelling Functions

Snorkel [50], [55], [61] and Snorkel's forum [62] we have not found guidance on how to define our LFs based on our labelling problem.

4.4.2.2 Evaluation of all LFs

The complete evaluation analysis of all LFs for "targeted" label can be found in appendix N. An example of how different LFs label a given instance as positive can be found in appendix O.

In table 8 we provide the results of a majority vote model (equal weight to LFs) and Snorkel's Generative model (different weights to LFs). For these labelling models, all LFs were used.

4.4.3 Evaluation of LFs for "Refers to a previous attack" label

The analysis for all LFs can be found in appendix N. An example of a labelled instance can be found in appendix O. In table 9 we provide the classification results for the two models using all of the LFs.

Labelling Model for Development Set "Targeted" (22 LFs)	Accuracy	F1	AUC
Majority Vote	0.53	0.02	0.50
Snorkel's Generative	0.76	0.73	0.82

Table 8: Evaluation of LFs for label "Targeted"

As we were not satisfied with the classification scores for this label, we examined mislabelled examples, and we included nine additional keyword-LFs to our models. The improved classification results can be found in table 10.

Labelling Model for Development Set "Refers to a previous attack." (50 LFs)	Accuracy	F1	AUC
Majority Vote	0.64	0*	0.5
Snorkel's Generative	0.67	0.59	0.70

*due to precision's denominator (TP+FP) equal to zero

Table 9: Evaluation of first fifty LFs for "Refers to a previous attack"

Labelling Model for Development Set "Refers to a previous attack" (59 LFs)	Accuracy	F1	AUC
Majority Vote	0.64	0*	0.5
Snorkel's Generative	0.76	0.67	0.76

*due to precision's denominator (TP+FP) equal to zero

Table 10: Evaluation of all LFs for "Refers to a previous attack"

4.4.4 Evaluation of LFs for “Espionage” label

The analysis for all LFs of espionage can be found in appendix N. In table 11 we provide the classification results for the two models using all of the LFs.

4.4.5 Annotation of Test Set

Above, we demonstrated how the LFs perform in the development set. As these LFs were developed concurrently with the hand-annotation of the development set, we hand-annotated a new subset of 167 observations which will be examined in the next section.

During the annotation of the test set, we did not write any new LF, as we wanted to examine how the produced LFs from the first round perform in a new unseen subset.

Labelling Model for Development Set “Espionage” (5 LFs)	Accuracy	F1	AUC
Majority Vote	0.91	0.18	0.55
Snorkel’s Generative	0.95	0.79	0.94

Table 11: Evaluation of all LFs for "Espionage"

4.5 Selection of Labelling Functions for short descriptions

In the first stage of this section, we use the produced generative models to evaluate it with our test set. Then we introduce an approach for selecting an appropriate set of LFs that offers a high AUC in our test set. The selected LFs will be used to label our training subset, which in section 4.6 will be used to train our NLP models.

4.5.1 Evaluation of all LFs in the test set

As we have seen in the previous section, Snorkel's model outperforms majority vote model in all labels. For this reason, in table 12 we present the evaluation results only for the Snorkel's Generative labelling model in the test set.

Generative Labelling Model in test set	Accuracy	F1	AUC
"Targeted"	0.73	0.72	0.82
"Refers to a previous attack"	0.67	0.52	0.72
"Espionage"	0.91	0.53	0.91

Table 12: Evaluation of all LFs in test set for short descriptions

Compared to the development set (were all LFs were based in), the evaluation in test set returns slightly lower scores across all metrics. As accuracy and F1 can differ due to different decision thresholds [24] in the next section, we focus only on the AUC.

4.5.2 Selection of LFs

In the previous results, we have used all of our produced LFs to create our generative model. As the model, does not rely on any ground-truth labels in order to optimize its performance, there might be powersets of LFs that can potentially offer better labelling results.

In general, the Snorkel's generative model requires at least three LFs to be produced. This means that the total number of LF-powersets is given from the following formula:

$$2^n - \binom{n}{1} - \binom{n}{2}$$

where n the number of total LFs

As the number of powersets increases exponentially when LFs increase, evaluating all of the possible powersets would be computationally prohibitive.

In order to overcome this problem, we devised an approach that examines randomly powersets of different sizes and LFs.

4.5.2.1 Label "Targeted"

In figure 19 we show how different powersets of LFs result in a different AUC. Every thin line represents an experiment, where sequentially, a new randomly selected LF is introduced. The black line represents the average AUC of all thin lines. In the case, where few LFs are used (3 to 9 LFs), an introduction of one additional LF can significantly increase or deteriorate the AUC. In addition, there might be settings, where a smaller number of LFs can yield to a higher AUC. Compared to using all of our LFs, we see that with 12 specific LFs we can get a slightly better AUC.

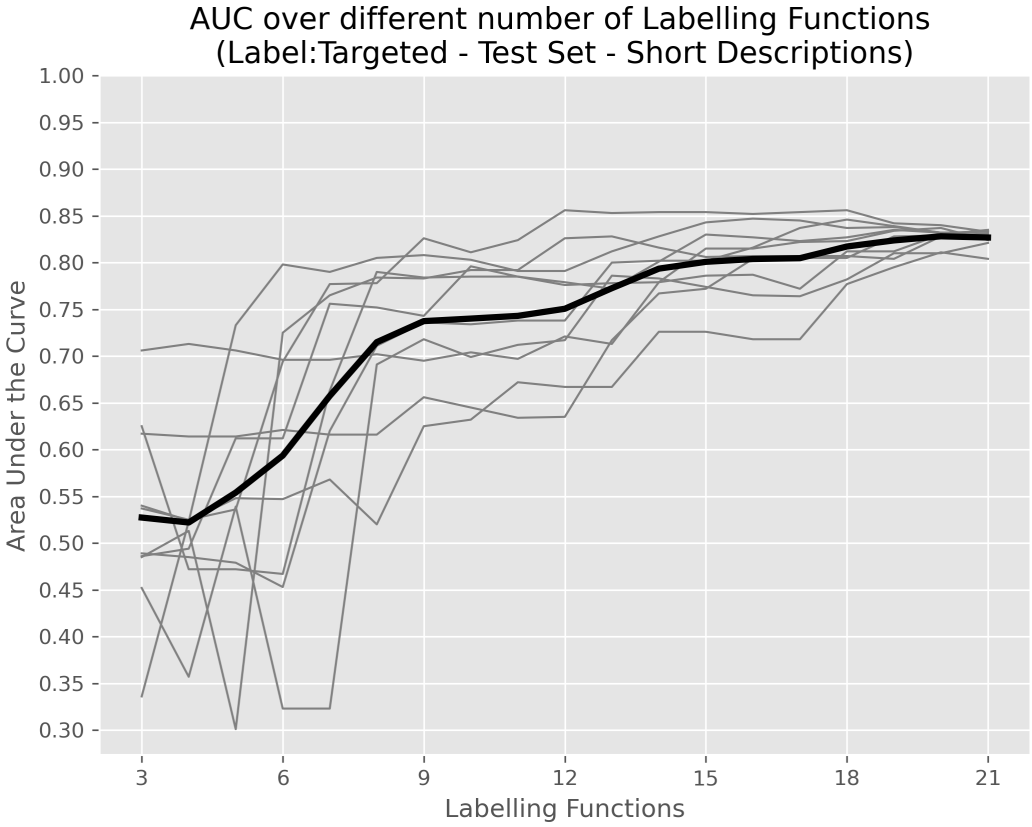


Figure 19: AUC over different powersets for short descriptions of "Targeted"

Although this approach is not exhaustive (in terms of assessing all of the possible powersets), we selected the powerset, which resulted in the highest AUC.

The ROC curve of the final powerset is presented in figure 20.

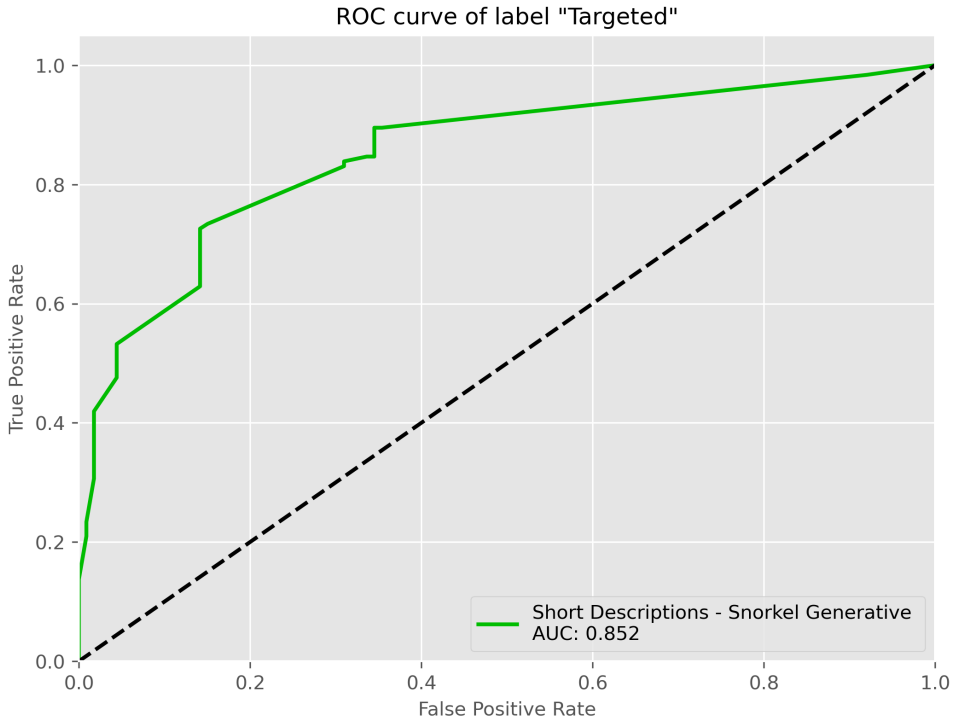


Figure 20: ROC curve of the final selected LFs for short descriptions of "Targeted"

4.5.2.2 Label “Refers to a previous attack.”

In the same fashion as with label “targeted” we performed a randomised search for the best powerset of the label “refers to a previous attack” (fig. 21). In this setting, 49 LFs resulted in the highest AUC.

The ROC curve of the final powerset is presented in figure 22.

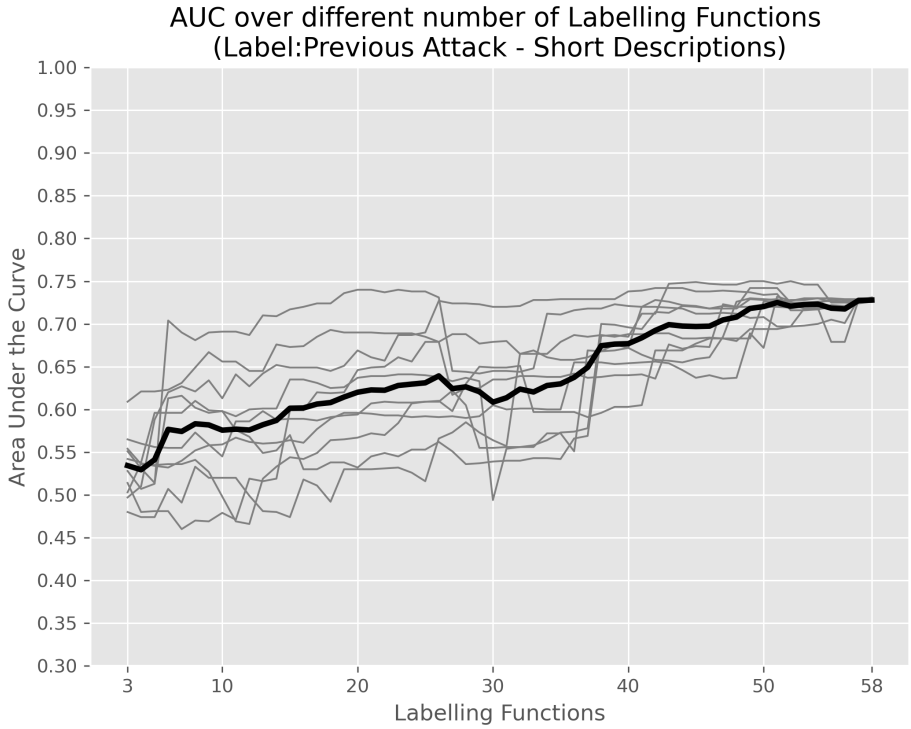


Figure 22: AUC over different powersets for short descriptions of "Refers to a previous attack"

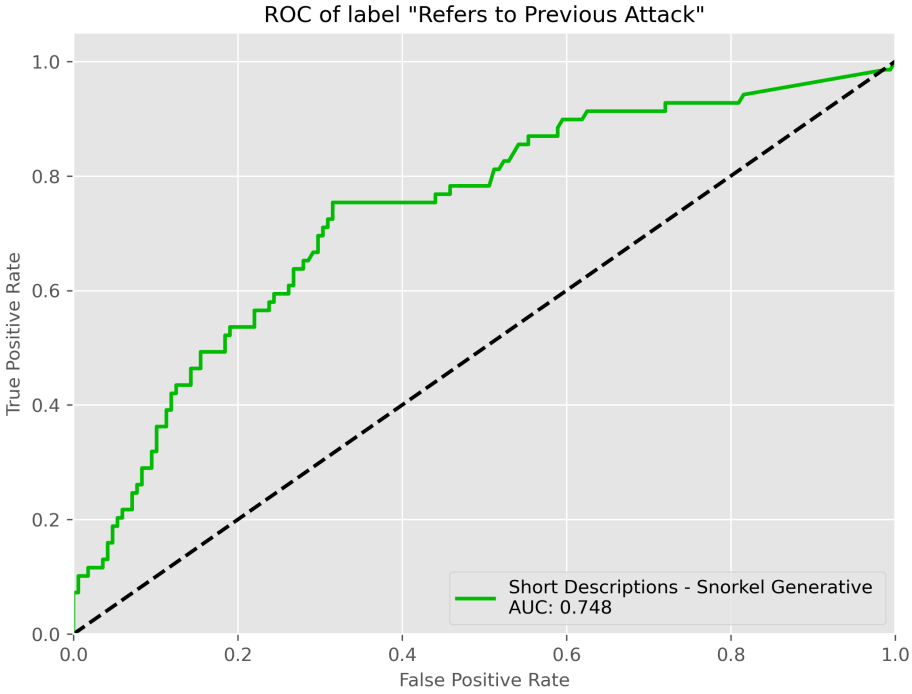


Figure 21: ROC curve of the final selected LFs for short descriptions of "Refers to a previous attack"

4.5.2.3 Label “Espionage”

For this label, we defined only 5 LFs, which means that the total powersets were 16. As the number were small, we performed an exhaustive search over all powersets. The results are presented in figure 23.

The ROC curve of the best performing powerset (3 LFs) is presented in figure 24.

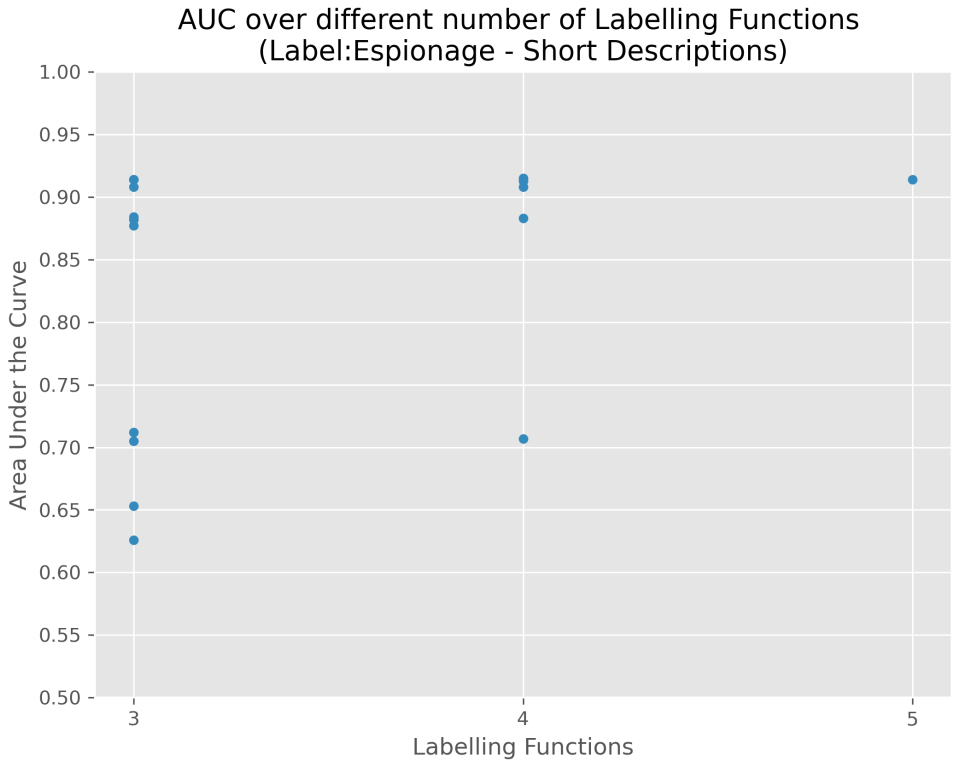


Figure 23: AUC over different powersets for short descriptions of "Espionage”

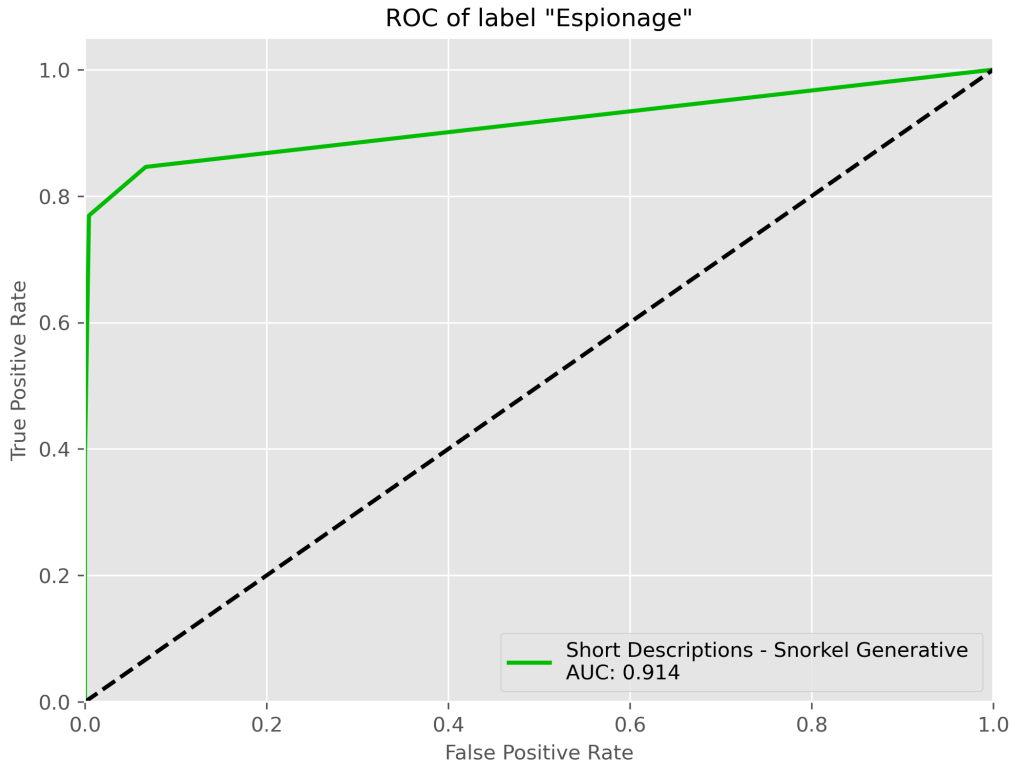


Figure 24: ROC curve of the final selected LFs for short descriptions of "Espionage"

All of the final selected LFs for the three labels can be found in appendix P.

4.6 Training and Evaluation of NLP models for short descriptions

The selected LFs in the previous section were used to label our training subset. With the produced training set containing only the actual textual description and the weak labels, we trained two NLP models (namely BERT and RoBERTa). As Snorkel suggests, training a discriminative model can further generalise beyond the LFs, increasing the classification performance on unseen data [55].

In order to develop our NLP models, we consulted their release papers [2], [43] but also two guides that are available on the web [63], [64]. In the development of the models, we performed trials with the suggested parameters from the developers of BERT [2]. From our experience, two aspects were prominent during the training of the models. The first was the number of tokens that were used to train the algorithm. Both for BERT and RoBERTa, the total word tokens for each description were at most 260. This aspect, has not posed any limitation to train our models. However, a dataset with more than 512 tokens would be prohibitive to run with the current language models. The second aspect regards the parameter of the number of epochs. In a sense, the number of epochs is the number of times the model sees the training data. The developers of BERT suggest that this number should vary from 1 to 4. In our models, we have seen that tuning this parameter has offered notable gains in our evaluation results.

The GPU we used was an Nvidia Tesla T4 [65], and the models ran in the environment of Google Colab [66].

4.6.1 NLP models for pulse descriptions

In figures 25-27 we present the results of our best performing models in the test set for the three labels. Table 13 summarize and compare them with Snorkel's generative model. Appendix Q. describes how the models were produced.

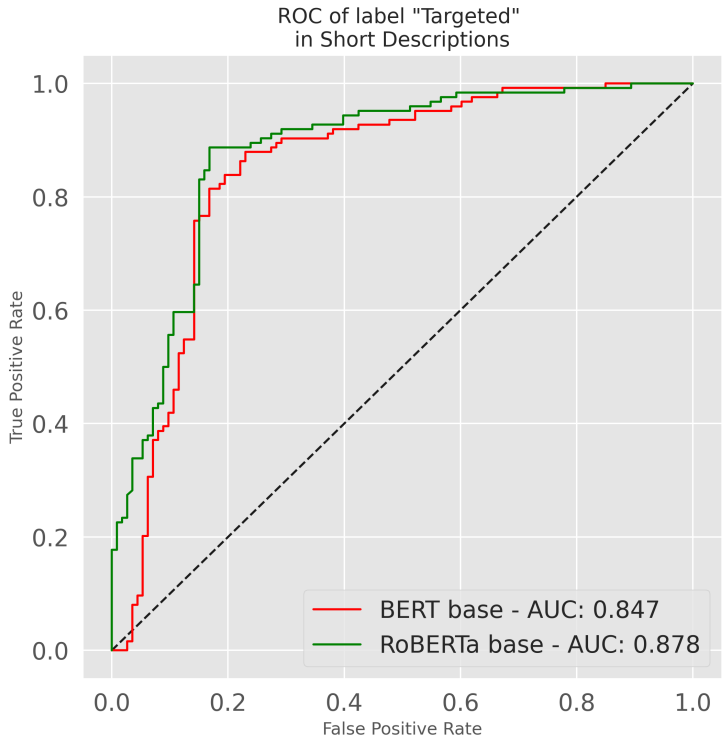


Figure 25: ROC curves of final models for short descriptions of "Targeted"

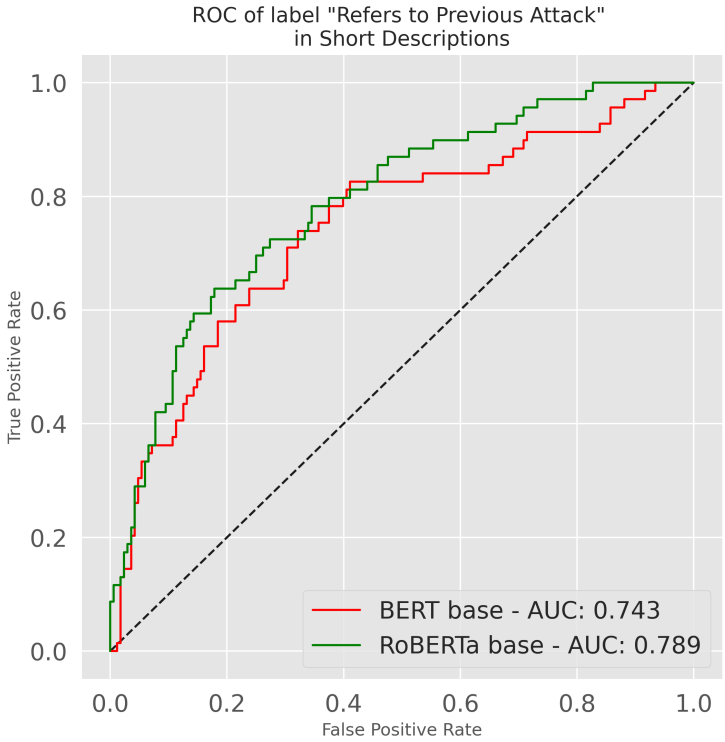


Figure 26: ROC curves of final models for short descriptions of "Refers to a previous attack"

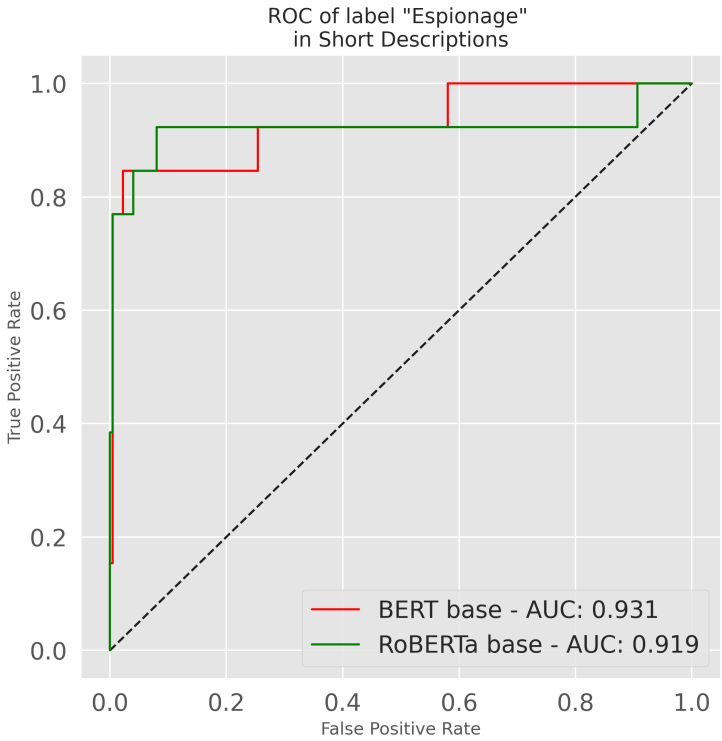


Figure 27: ROC curves of final models for short descriptions of "Espionage"

Test Set	Generative Labelling Model AUC	BERT AUC	RoBERTa AUC
“Targeted”	0.82	0.847	0.878
“Refers to previous attack”	0.72	0.743	0.789
“Espionage”	0.91	0.931	0.919

Table 13: Final scores for short descriptions

4.6.2 Training with a balanced dataset for label "espionage"

From the above the results, we see that the produced NLP models for "espionage" label did not offer the same lifts compared to the models of the other labels. The weak label "espionage" is imbalanced with 16% positive rate in the training subset. Training NLP models with imbalanced datasets may affect their performance. A recent study with imbalanced datasets in BERT models [67] suggests data augmentation as a way to deal with imbalanced data.

Snorkel, with its Transformation Functions [68] offers a framework to augment and balance textual datasets. Augmentation can be done in several ways, for example by swapping adjectives within an instance, by using synonyms or by changing country names.

In our case, we reduced our negative instances by half, and we used Snorkel to augment the positively labelled instances to have a balanced dataset. More technical about this process can be found in appendix R.

In figure 28 and table 14, we present the results with the balanced training set.

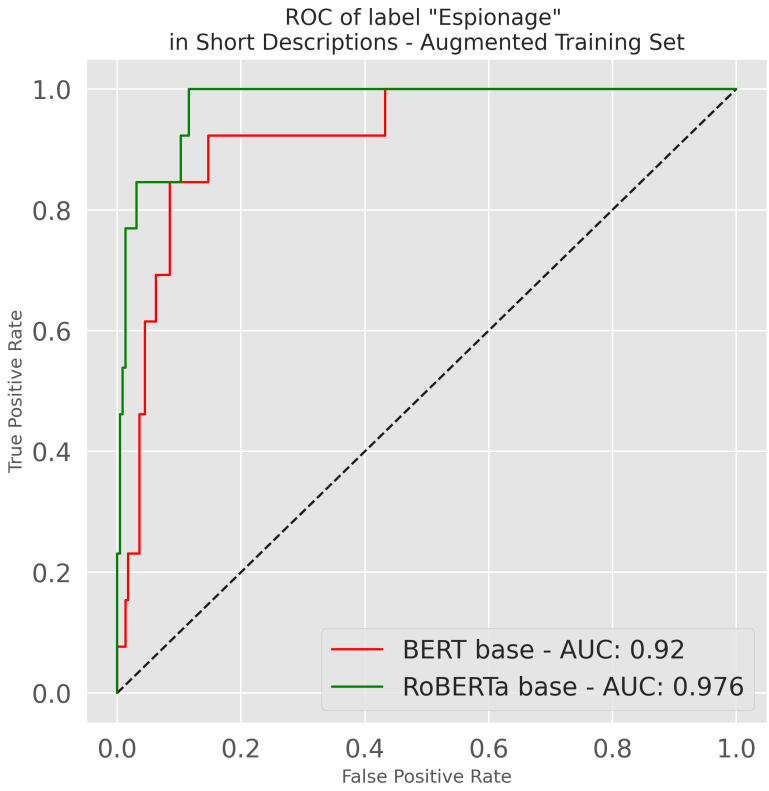


Figure 28: ROC curves of final models for short descriptions of "Espionage" with balanced dataset

Label “Espionage”	BERT AUC	RoBERTa AUC
Original Training Set	0.931	0.919
Augmented – Balanced Training Set	0.920	0.976

Table 14: Final Scores of original and balanced dataset for short descriptions of "Espionage"

4.7 Scraping, Subset definition and annotation of web pages

In the previous section, we have dealt with short descriptions that come with every pulse in our feed. In this section, we describe how we collect content from external references that come with every pulse. In the next sections, we will follow a similar approach to that of short descriptions to create NLP models that can classify different web pages according to our previously selected labels.

4.7.1 Scraping of Web Pages

In the F-web_scraping.py script, we developed a web scraper that uses Python's BeatifulSoup package.

In subsection 4.1.5.1 we have shown that the references that come with every pulse, may direct to different types of sources (web pages, tweets, pdfs). As the number of tweets and pdfs, links were comparatively small to the whole dataset, we developed our solution only for web pages. In addition, as web pages may come in various languages, we decided to keep only those that use the English language. In this way, we ended up with 1730 observations (out of the 2019 observations). As every pulse may had multiple references, we focused mainly on the first reference (link) of every pulse. If the first link was not active, then our scrapper proceeded by retrieving information from the second link and so on.

For these 1730 web pages, we kept their main body text, excluding elements that may come from headers, footers or sidebars. Also, as some web pages had more advanced systems to prevent web scrapping, we performed some quality checks on

the web scraped content, to keep only observations that provide a meaningful description of a pulse. More technical details about these quality checks can be found in appendix S. and D-Subsets.ipynb notebook. Finally, our data collection consisted of 1371 web pages.

4.7.2 Subset definition

From the original dataset of 2019 observations, we ended up with a dataset of 1371 web pages (68%). From these 1371 observations, we defined a stratified test subset of 134 records (around 10%). The stratification has been done as per section 4.1.6.2 ("Creation of Stratified Subsets"). The selected observations did not interfere with the observations of development set for short descriptions, as the LFs were made according to this subset.

In figure 29 we provide an overview of the new subsets:

- Test subset (A): 134 observations
- Training subset (B): 1237 observations

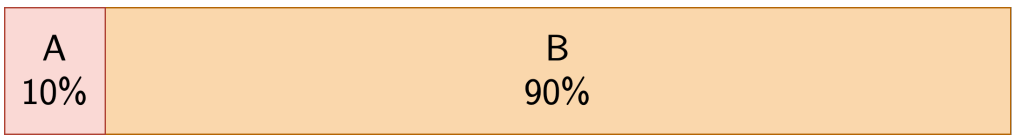


Figure 29: Final subsets for web pages

4.7.3 Annotation of Test Subset

For annotating our test subset, we visited the links of the extracted web sites. We have considered the whole web page before we decide for a label.

4.8 Selection of Labelling Functions and text window for web pages

In this stage, we describe how we selected the LFs for generating our three weak labels for web pages. Next, we describe the challenges that we faced with the long textual data of the web pages and the way we have dealt with them.

4.8.1 Evaluation of all LFs in the test set of web pages

In this section, we evaluate the produced LFs in the test subset of webpages, following the same approach as with section 4.4. The results are presented in table 15.

Generative Labelling Model in the test set of web pages	Accuracy	F1	AUC
"Targeted"	0.66	0.69	0.67
"Refers to previous attack"	0.67	0.72	0.70
"Espionage"	0.68	0.19	0.84

Table 15: Evaluation of all LFs in test set for web pages

4.8.2 Selection of LFs

For all three labels, we follow the same randomised approach as with section 4.5. Note that the LFs are applied in the whole body of the extracted web pages which varies from 400 to 7500 words.

4.8.2.1 Label "Targeted"

For label "targeted", the powerset with the highest AUC contained 14 LFs. The selected LFs can be found in appendix T.

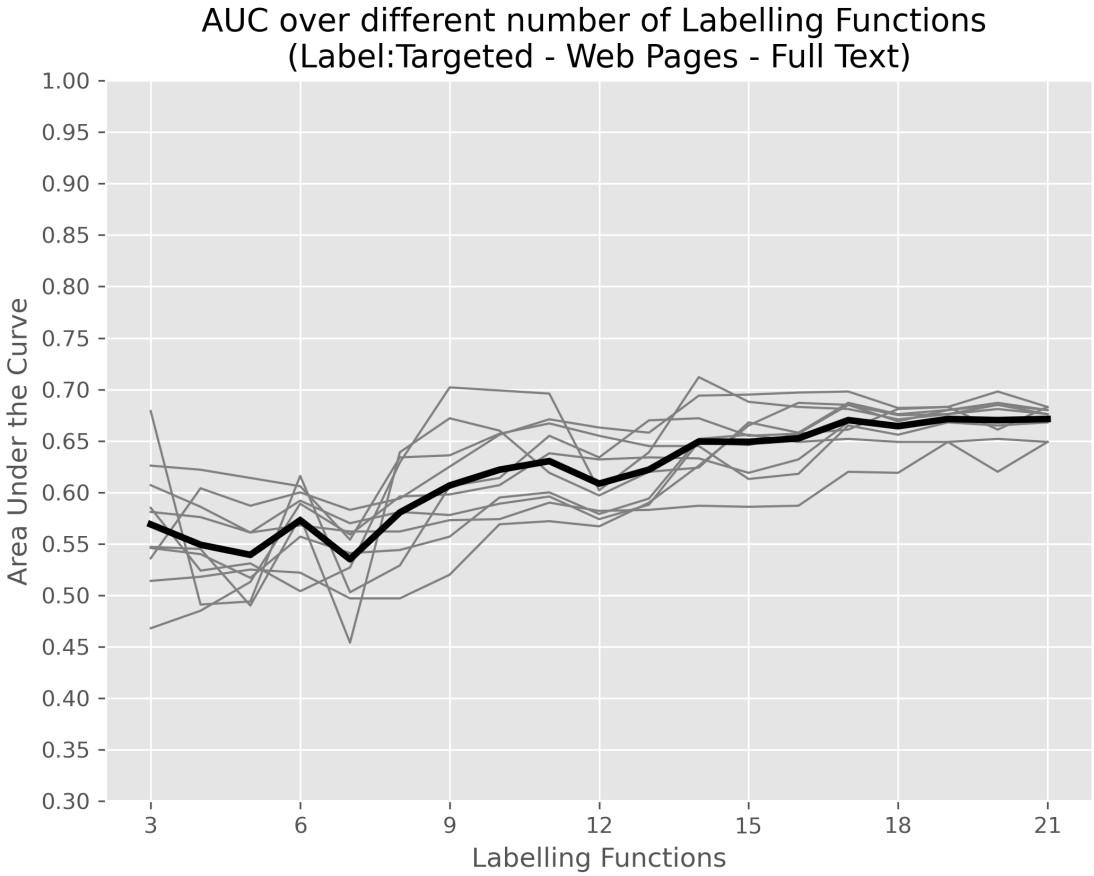


Figure 30: AUC over different powersets for web pages of "Targeted"

4.8.2.2 Label “Refers to a previous attack”

For label “Refers to a previous attack”, the powerset with the highest AUC contained 51 LFs.

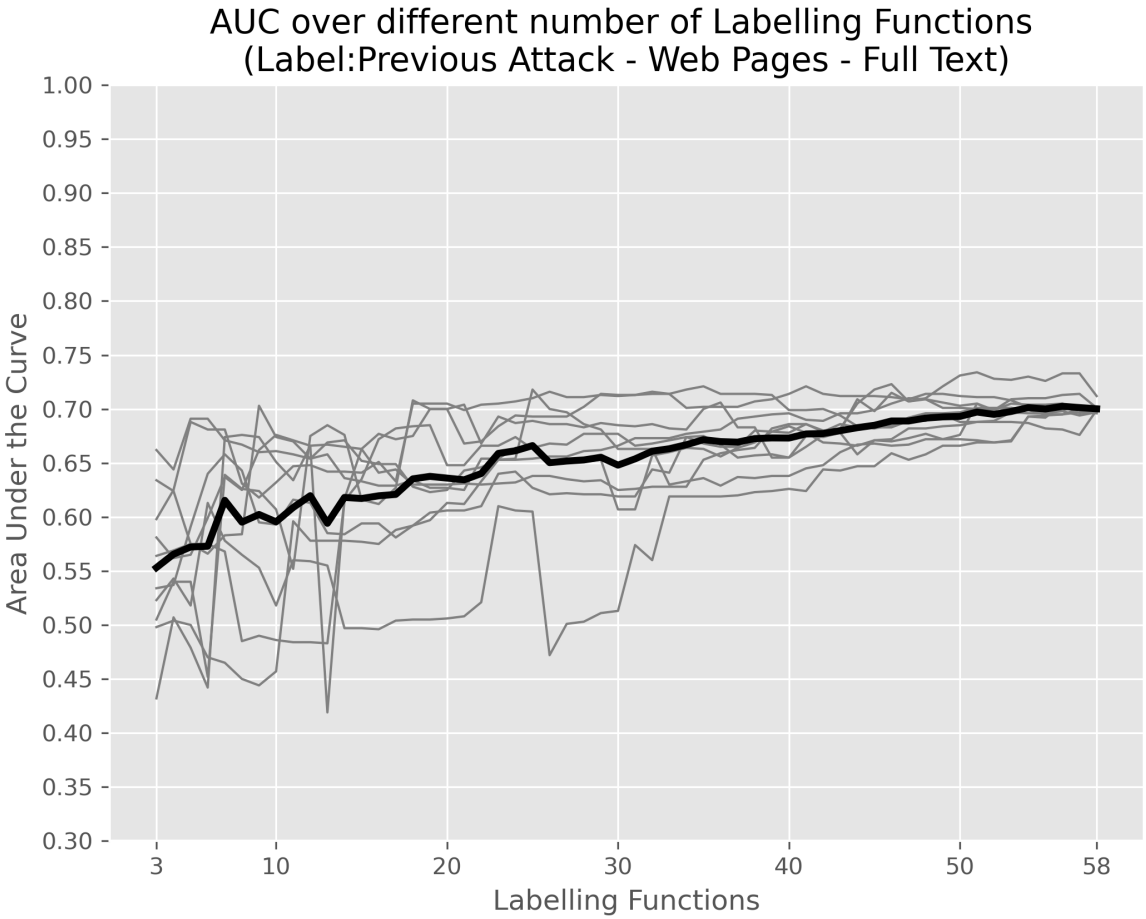


Figure 31: AUC over different powersets for web pages of "Refers to a previous attack"

4.8.2.3 Label “Espionage”

For label “espionage” we performed an exhaustive search over all possible powersets. The powerset with the highest AUC contained 3 LFs.

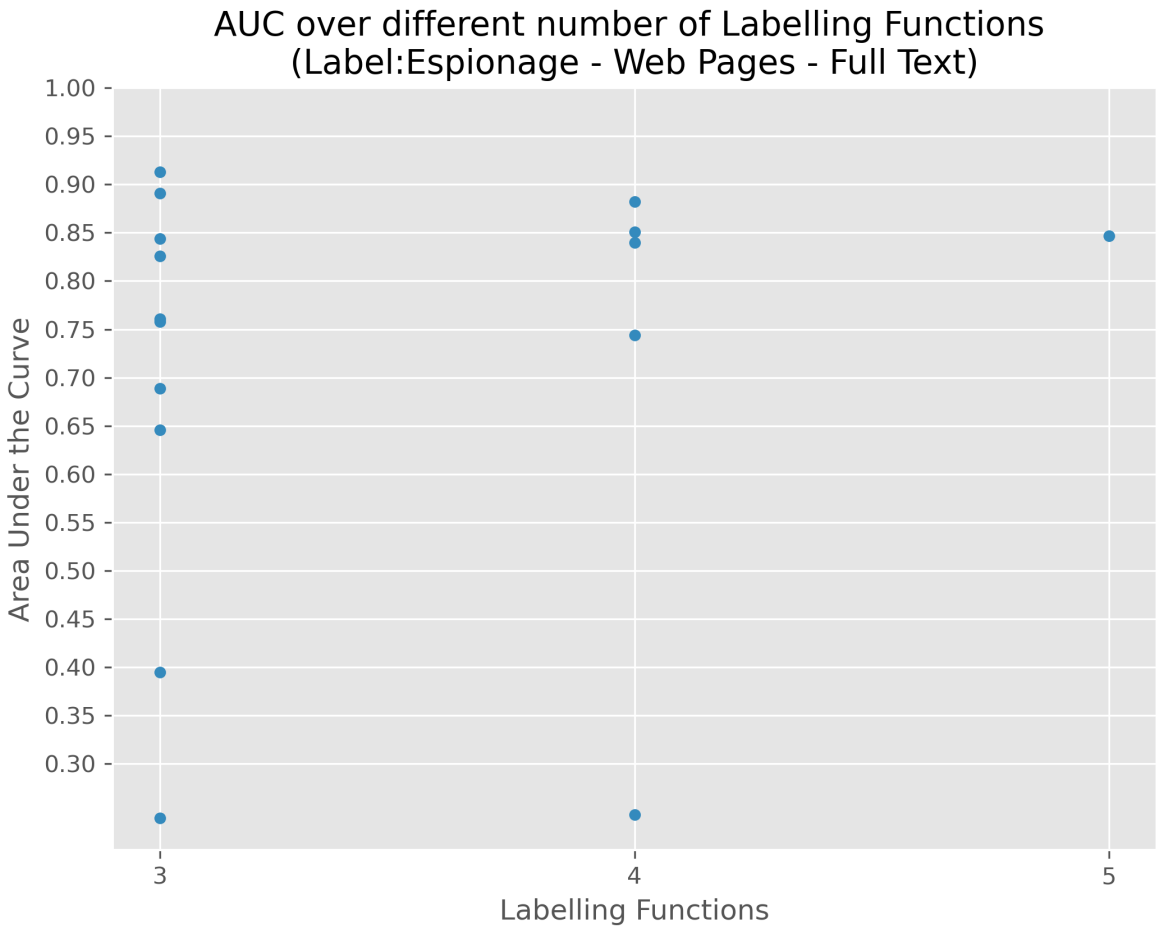


Figure 32: AUC over different powersets for web pages of "Espionage”

4.8.3 Selection of text window

Before we proceed with the training of our NLP models, we recall a limitation of BERT and RoBERTa models, that was discussed in section 4.6. Both models can handle at most 512 tokens, which means that using the long documents of web pages

to train our models was not possible. Also, due to limited GPU sources, training a model with 512 tokens was not feasible in our case. Typically, for this kind of problem, users from knowledge sharing forums such as StackOverflow, suggest using the first n tokens or the last n tokens of the dataset to train an NLP model [69]. The rationale is that essential information can be found in the first or the last sentences of a document. (introduction or conclusion).

However, as our web pages, did not follow the standard structure of a document (as viable information might not be at the beginning or the end of the web page), we developed an algorithm that evaluates the performance of the LFs in different textual windows.

Usually, these textual windows are called sliding windows and have as main parameter the total number of words we want each window to have.

For example, for the sentence:

"BERT is a cool NLP model."

The sliding windows of three words would be the following:

- "BERT is a"
- "is a cool"
- "a cool NLP"
- "cool NLP model"

Although for our problem, as the total length of web pages differed significantly, we developed an approach that defines the size of the window according to the size of the document. In a sense, for long documents, there will be windows of great size and vice-versa. Besides, this approach would allow us to define a fixed number of sliding text windows.

The number of windows has been determined first by the maximum available number of word tokens that we can feed to our NLP models (300 tokens due to GPU) and

secondly by the web page with the most extended text (around 7500 words). In this way, we defined the total number of text windows to be 25 (7500/300). Below we show how the selected LFs performed in the different sliding windows in the test set.

4.8.3.1 Label "Targeted"

From the results presented in figure 33 we see that the highest performing windows were in the beginning and in the 20^{th} window. We selected the window two as it had the highest AUC.

In figure 34 we display the ROC curve with the best performing LFs and text window for the test set.

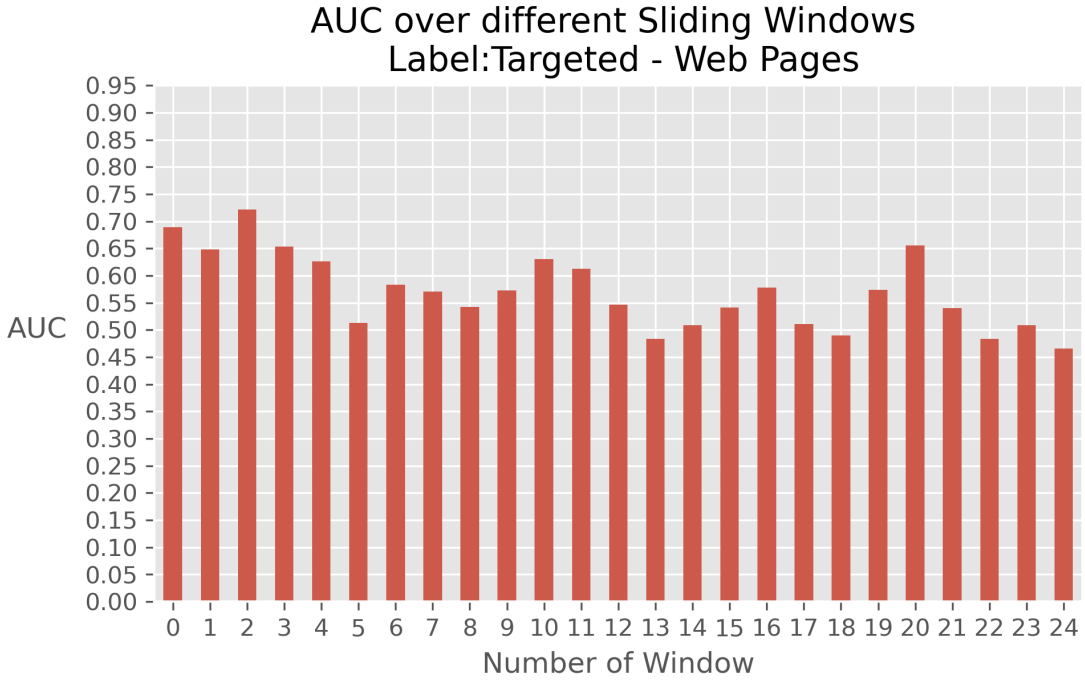


Figure 33: AUC over different sliding windows for "Targeted"

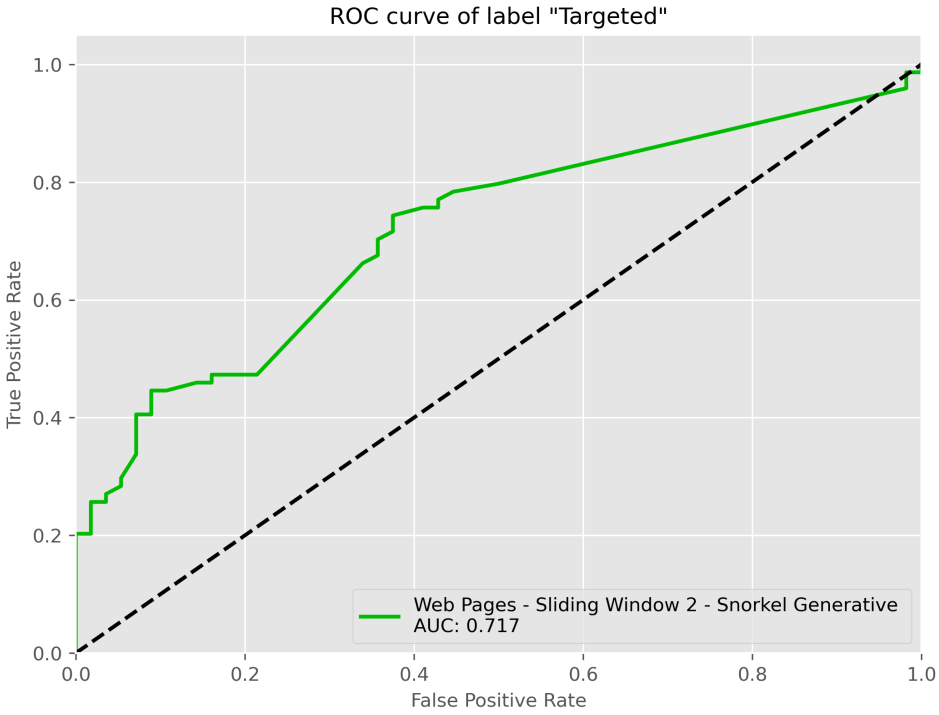


Figure 34: ROC curve for best performing window of "Targeted"

4.8.3.2 Label “Refers to a previous attack”

In the same fashion, in figures 35-36 we display the performance over different text windows and the ROC curve of the best LFs and sliding window.

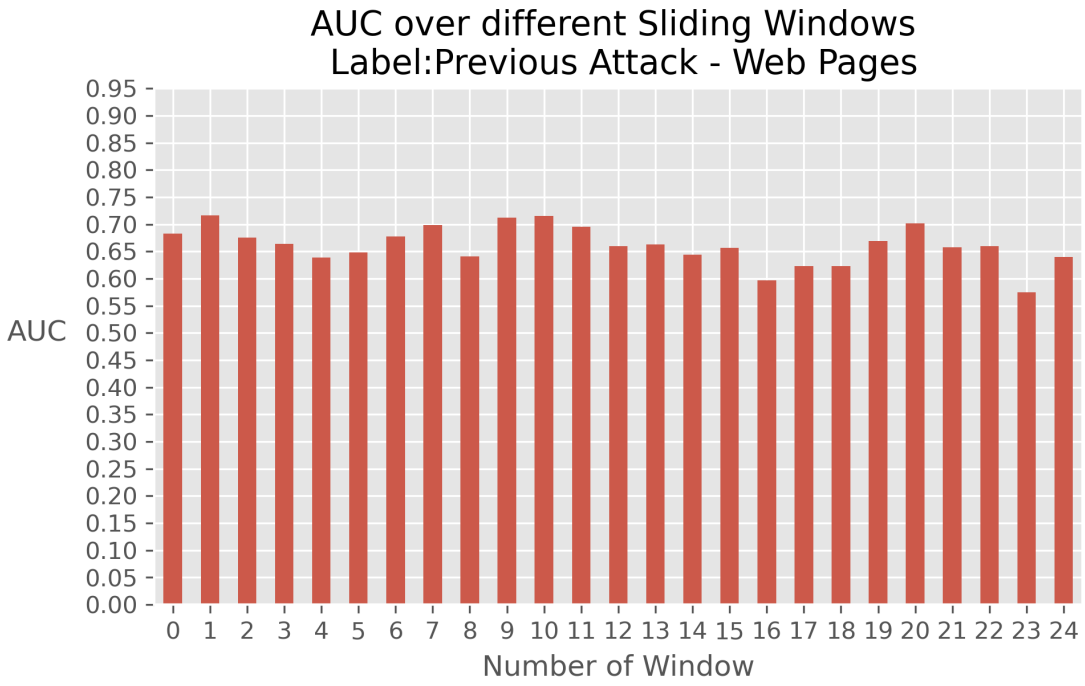


Figure 35: AUC over different sliding windows for "Refers to a previous attack"

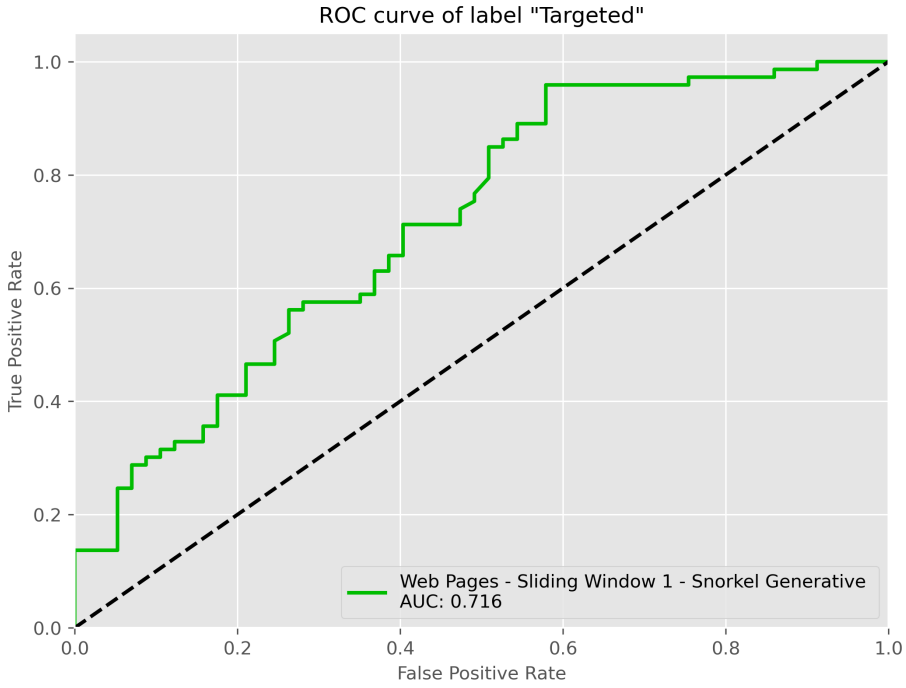


Figure 36: ROC curve for best performing window of "Refers to a previous attack"

4.8.3.3 Label “Espionage”

Finally, in fig 37-38 we provide the results for label “espionage”.

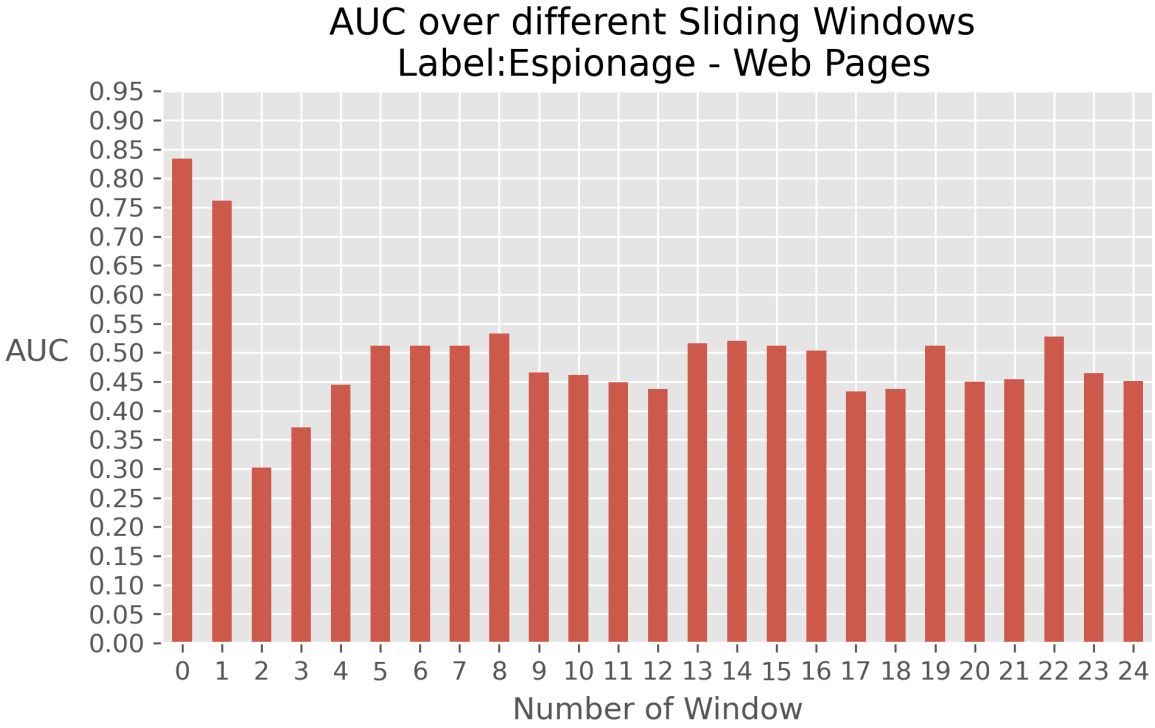


Figure 37: AUC over different sliding windows for "Espionage"

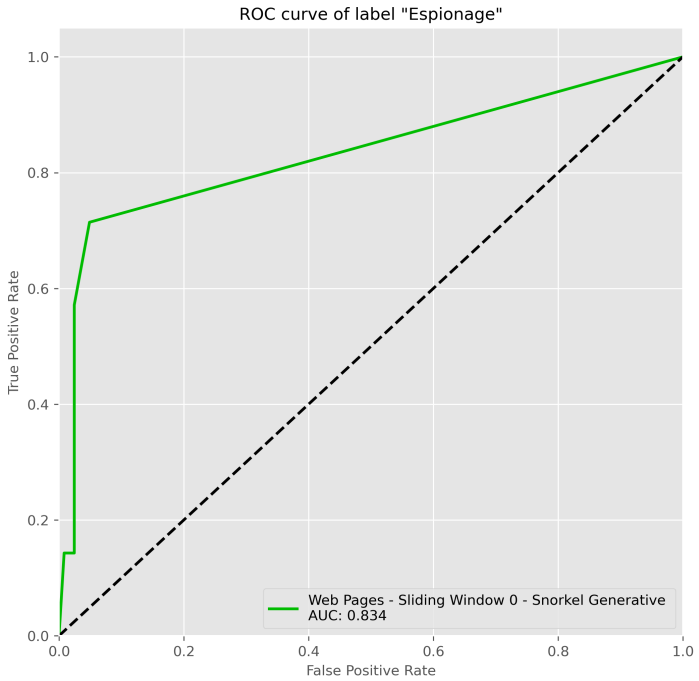


Figure 38: ROC curve for best performing window of "Espionage"

4.9 Training and Evaluation of NLP models for web pages

For training and evaluating our NLP models, we follow the same approach as per section 4.6. The parameters and the technical details of them can be found in appendix U.

4.9.1 NLP models for web pages

In figures 39-41 we present the results of our best performing models in the test set for the three labels. Table 16 summarize and compare them with Snorkel's generative model. The latter model is developed with the best performing set of LFs in the full document of the web pages.

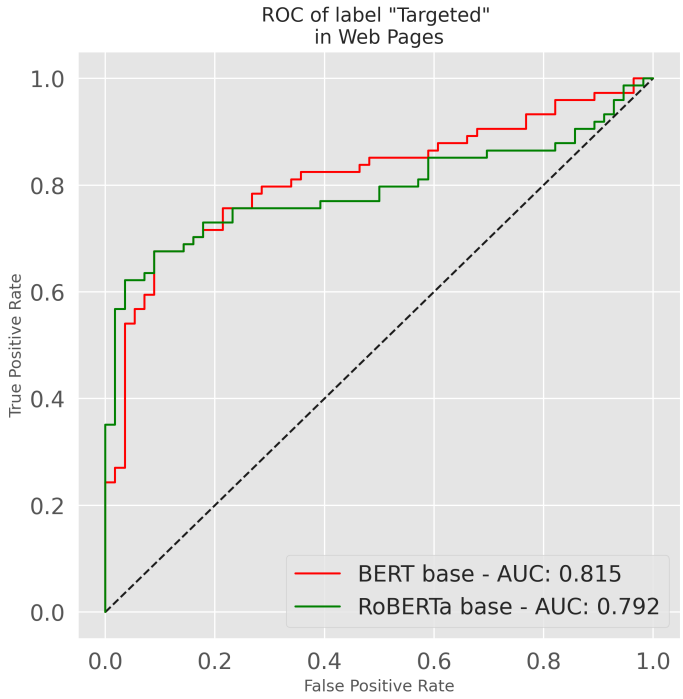


Figure 40: ROC curves of final models for web pages of "Targeted"

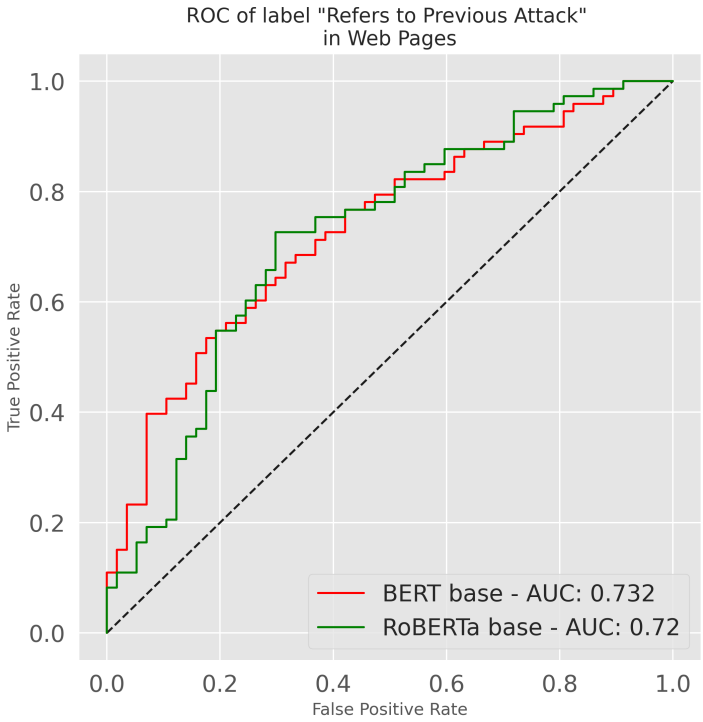


Figure 39: ROC curves of final models for web pages of "Refers to a previous attack"

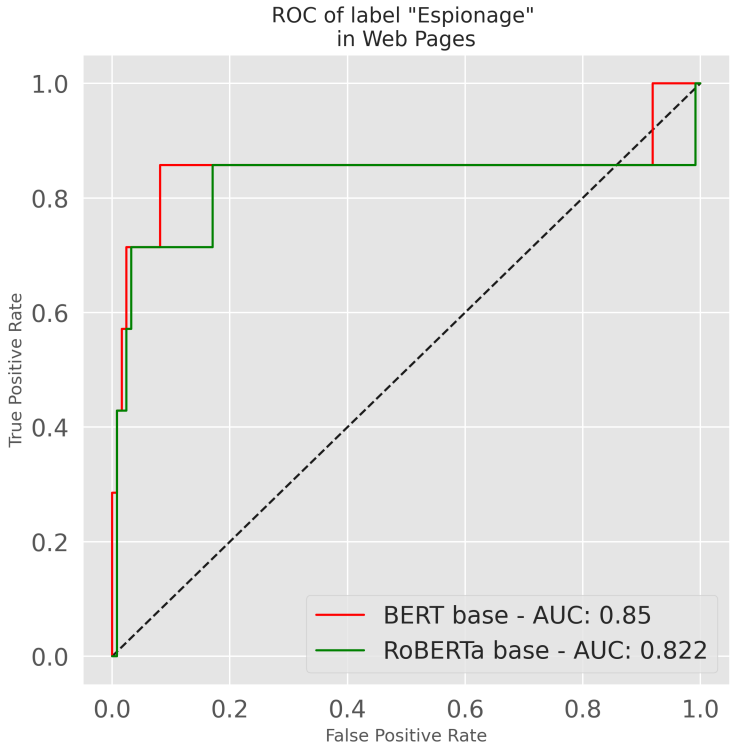


Figure 41: ROC curves of final models for web pages of "Espionage"

Test Set (web pages)	Generative Labelling Model AUC	BERT AUC	RoBERTa AUC
"Targeted"	0.67	0.815	0.792
"Refers to a previous attack"	0.70	0.732	0.72
"Espionage"	0.84	0.85	0.822

Table 16: Final scores for web pages

4.9.2 Training with a balanced dataset for label "espionage"

As per section 4.6.2 for short descriptions, the label "espionage" is highly imbalanced. In figure 42 and table 17, we display the results of the models trained with a balanced dataset. The balanced dataset of 1165 observations has been produced with the same approach as for short descriptions and details for it can be found in appendix R.

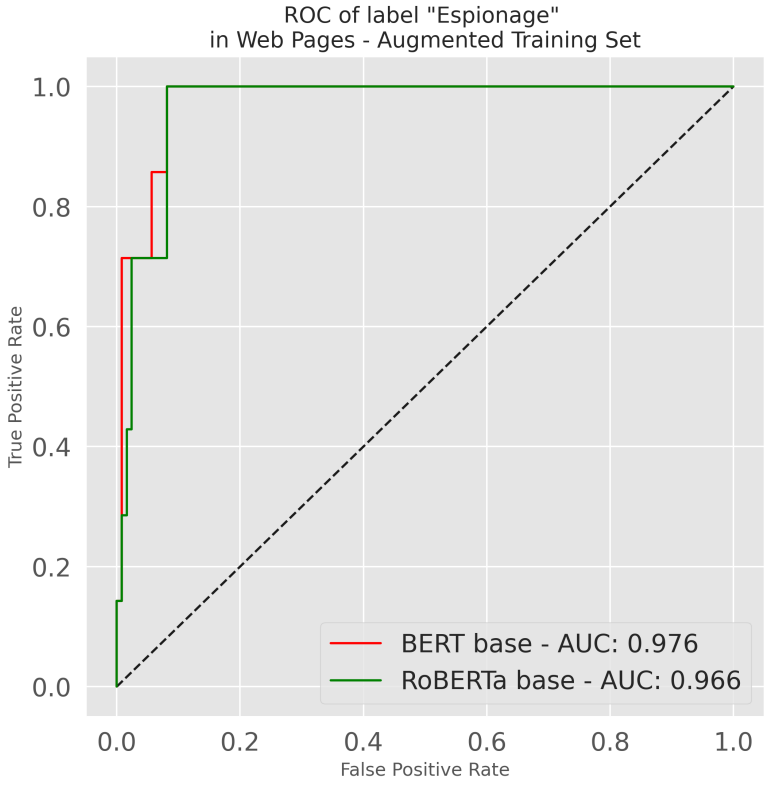


Figure 42: ROC curves of final models for web pages of "Espionage" with balanced dataset

Label “Espionage”	BERT AUC	RoBERTa AUC
Original Training Set	0.850	0.822
Augmented – Balanced Training Set	0.976	0.966

Table 17: Final Scores of original and balanced dataset for web pages of "Espionage"

4.10 Summary of Results

In table 18 we present the results of all produced models. In general, both BERT and RoBERTa achieved higher AUC compared to Snorkel’s generative model. RoBERTa, in most cases, has provided the best performing models for the short descriptions. Conversely, BERT has proved to overperform RoBERTa in all models for web pages. Finally, the augmented datasets for the label “espionage” have shown to offer notable lifts in the performance of the NLP models.

Label	Source	Train Set Size	Test Set Size	Total LFs	Selected LFs	Snorkel Model AUC	BERT Base AUC	RoBERTa Base AUC
Targeted	Short Descriptions	1574	167	21	12	0.82	0.847	0.878
Targeted	Web Pages	1237	134	21	14	0.67	0.815	0.792
Previous Attack	Short Descriptions	1574	167	59	49	0.72	0.743	0.789
Previous Attack	Web Pages	1237	134	59	44	0.70	0.732	0.720
Espionage	Short Descriptions	1574	167	5	3	0.91	0.931	0.919
Espionage	Short Descriptions	1543*	167	5	3		0.920	0.976
Espionage	Web Pages	1237	134	5	3	0.84	0.850	0.822
Espionage	Web Pages	1165*	134	5	3		0.976	0.966

* balanced dataset

Table 18: Summary of results

Chapter 5 Conclusion

In this chapter, we provide a summary of this study, limitations and possible areas for further research.

5.1 Summary

This study aimed to examine the available data labelling approaches and apply an appropriate one to a data labelling problem of a cyber-security team.

In chapter 2, we provided the background of the cybersecurity domain, and its threat intelligence feeds, a literature review of the available data labelling approaches and the state-of-the-art for NLP models.

Next, in chapter 3, we linked this knowledge with the data labelling problem that the cyber-security team faces; label threat intelligence feeds according to their textual descriptions. After relating it with the different data labelling techniques, we proceeded with Weak Supervision and Data Programming approach.

Finally, in chapter 4, we presented our solution, which comprised of:

- Collecting and selecting an appropriate threat intelligence feed
- Selecting labels and creating Labelling Functions for them with Snorkel framework

- Weak label descriptions and web pages of our selected feed
- Train and evaluate state-of-the-art NLP models with the produced weak labels

5.2 Limitations

One major limitation that we faced while developing our solution was the limited computational resources for training our NLP models. As both BERT and RoBERTa rely on neural networks, training big datasets in terms of both text size and number of instances require access to GPUs with great RAM. Also, this limitation has confined us not to train more heavy models such as BERT and RoBERTa large.

5.3 Areas for further research

The size of the threat intelligence feed that we used, which can also be considered as a limitation, is an aspect of this solution which could be further investigated. Data Programming, our selected data labelling approach, claims through theoretical and empirical studies that having access to big collections of data can further improve the quality of the produced labels. This topic has been examined in more detail in section 3.3. Having access to a more extensive threat intelligence feed could potentially increase the quality of our produced labels.

Furthermore, we identify three more areas for further research.

5.3.1 Creation of Labelling Functions

In section 4.4.2.1 we described why Labelling Functions are preferred to provide a negative label if a condition is not fulfilled (compared to abstaining from providing a label). Although this approach has improved the quality of our labels, we could further investigate under which conditions a Labelling Function should aim not to provide a label.

5.3.2 Selection of Labelling Functions

Especially, for labels "targeted" and "refers to a previous attack", we selected our Labelling Functions based in a randomised and non-exhaustive approach. Inspired by stepwise regression models, we could formulate an algorithm that evaluates and chooses different powersets of LFs based on a criterion.

5.3.3 Selection of Text Window

In section 4.8.3 to overcome the problem of the diverging and large in length web pages, we devised a strategy that creates a fixed number of windows. The selection of the text windows could be further investigated and tailored to the particular characteristics of every web page. Also, the windows could be defined according to the LFs and where in-text these are triggered.

References

[1] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," *Proc. IEEE Int. Conf. Comput. Vis.*, vol. 2015 Inter, pp. 1026–1034, 2015, doi: 10.1109/ICCV.2015.123.

[2] J. Devlin, M. W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," 2018.

[3] N. Nangia and S. R. Bowman, "Human vs. Muppet: A conservative estimate of human performance on the GLUE benchmark," *ACL 2019 - 57th Annu. Meet. Assoc. Comput. Linguist. Proc. Conf.*, pp. 4566–4575, 2019, doi: 10.18653/v1/p19-1449.

[4] Gartner, "Hype Cycle for Data Science and Machine Learning, 2020," Jul. 28, 2020. https://www.gartner.com/document/code/450404?ref=ddisp&refval=450404 (accessed Aug. 19, 2020).

[5] World Economic Forum, "The Global Risks Report 2020," pp. 1–114, 2020, [Online]. Available: http://wef.ch/risks2019.

[6] Accenture and Ponemon Institute, "The Cost of Cybercrime: Ninth Annual Cost of Cybercrime Study," *Ninth Annu. Cost Cybercrime Study*, p. 18, 2019, [Online]. Available: https://www.accenture.com/_acnmedia/PDF-96/Accenture-2019-Cost-of-Cybercrime-Study-Final.pdf#zoom=50.

[7] ITU, "Recommendation ITU-T X.1205," Geneva, 2008. doi: 10.22215/timreview835.

[8] J. Jang-Jaccard and S. Nepal, "A survey of emerging threats in cybersecurity," *J. Comput. Syst. Sci.*, vol. 80, no. 5, pp. 973–993, 2014, doi: 10.1016/j.jcss.2014.02.005.

[9] N. Sun, J. Zhang, P. Rimba, S. Gao, L. Y. Zhang, and Y. Xiang, "Data-Driven Cybersecurity Incident Prediction: A Survey," *IEEE Commun. Surv. Tutorials*, vol. 21, no. 2, pp. 1744–1772, 2019, doi: 10.1109/COMST.2018.2885561.

[10] J. Friedman and M. Bouchard, *Definitive Guide to Cyber Threat Intelligence: Using Knowledge about Adversaries to Win the War against Targeted Attacks*. 2015.

[11] CISCO, "Hunting for Hidden Threats," 2019.

[12] MITRE, "Putting MITRE ATT&CK into Action with What You Have, Where You Are," 2019. https://www.slideshare.net/KatieNickels/putting-mitre-attck-into-action-with-what-you-have-where-you-are (accessed Jul. 24, 2020).

[13] D. J.Bianco, "The Pyramid of Pain | Enterprise Detection & Response," 2013. http://detect-

respond.blogspot.com/2013/03/the-pyramid-of-pain.html (accessed Aug. 21, 2020).

[14] Optiv, "Tactics, Techniques and Procedures (TTPs) Within Cyber Threat Intelligence | Optiv," 2017. https://www.optiv.com/explore-optiv-insights/blog/tactics-techniques-and-procedures-ttps-within-cyber-threat-intelligence (accessed Jul. 24, 2020).

[15] V. Mavroeidis and S. Bromander, "Cyber threat intelligence model: An evaluation of taxonomies, sharing standards, and ontologies within cyber threat intelligence," *Proc. - 2017 Eur. Intell. Secur. Informatics Conf. EISIC 2017*, vol. 2017-Janua, pp. 91–98, 2017, doi: 10.1109/EISIC.2017.20.

[16] B. E. Strom, D. P. Miller, K. C. Nickels, A. G. Pennington, and C. B. Thomas, "MITRE ATT&CK ^{TM}: Design and Philosophy Authors:," no. March, 2020.

[17] Gartner, "Definition: Threat Intelligence," 2014. .

[18] Trend Micro USA, "Indicators of compromise - Definition - Trend Micro USA." https://www.trendmicro.com/vinfo/us/security/definition/indicators-of-compromise (accessed Jul. 24, 2020).

[19] Kaspersky, "Kaspersky Private Security Network - Datasheet," 2019. [Online]. Available: https://media.kaspersky.com/en/business-security/enterprise/KPSN-Datasheet-GLOBAL-EN.pdf.

[20] Cisco, "Cisco Talos - Whitepaper," *Cisco*, pp. 1–7, 2019, [Online]. Available: https://www.cisco.com/c/en/us/products/security/talos.html%0Ahttps://talosintelligence.com/docs/Talos_WhitePaper.pdf.

[21] Ponemon institute, "Third Annual Study on Exchanging Cyber Threat Intelligence: There Has to Be a Better Way," no. November, 2018.

[22] H. Griffioen, T. Booij, and C. Doerr, "Quality Evaluation of Cyber Threat Intelligence Feeds," 2018.

[23] T. Ring, "Threat intelligence: Why people don't share," *Comput. Fraud Secur.*, vol. 2014, no. 3, pp. 5–9, 2014, doi: 10.1016/S1361-3723(14)70469-5.

[24] C. C. Aggarwal, *Data Classification*. Chapman and Hall/CRC, 2014.

[25] "Hazy Research Research Group," 2020. http://hazyresearch.stanford.edu/ (accessed Jul. 13, 2020).

[26] Y. Roh, G. Heo, and S. E. Whang, "A Survey on Data Collection for Machine Learning: A Big Data - AI Integration Perspective," *IEEE Trans. Knowl. Data Eng.*, vol. 4347, no. c, pp. 1–1, 2019, doi: 10.1109/tkde.2019.2946162.

[27] V. C. Raykar *et al.*, "Learning from crowds," *J. Mach. Learn. Res.*, vol. 11, pp. 1297–1322, 2010.

[28] B. Settles, “Computer Sciences Active Learning Literature Survey,” no. January, 2009.

[29] P. Whitla, “Crowdsourcing and Its Application in Marketing Activities,” vol. 5, no. 1, pp. 15–28, 2009.

[30] A. I. Chittilappilly, L. Chen, and S. Amer-Yahia, “A Survey of General-Purpose Crowdsourcing Techniques,” *IEEE Trans. Knowl. Data Eng.*, vol. 28, no. 9, pp. 2246–2266, 2016, doi: 10.1109/TKDE.2016.2555805.

[31] F. Daniel, P. Kucherbaev, C. Cappiello, B. Benatallah, and M. Allahbakhsh, “Quality control in crowdsourcing: A survey of quality attributes, assessment techniques, and assurance actions,” *ACM Comput. Surv.*, vol. 51, no. 1, 2018, doi: 10.1145/3148148.

[32] B. Settles, “From Theories to Queries: Active Learning in Practice,” in *Active Learning and Experimental Design workshop In conjunction with AISTATS 2010*, 2011, vol. 16, pp. 1–18, [Online]. Available: http://proceedings.mlr.press/v16/settles11a.html.

[33] A. Ratner, C. De Sa, S. Wu, D. Selsam, and C. Ré, “Data programming: Creating large training sets, quickly,” *Adv. Neural Inf. Process. Syst.*, no. Nips, pp. 3574–3582, 2016.

[34] E. Ferrara, P. De Meo, G. Fiumara, and R. Baumgartner, “Web data extraction, applications and techniques: A survey,” *Knowledge-Based Syst.*, vol. 70, pp. 301–323, Nov. 2014, doi: 10.1016/j.knosys.2014.07.007.

[35] H. Bast, B. Buchhold, and E. Haussmann, “Semantic search on text and knowledge bases,” *Found. Trends Inf. Retr.*, vol. 10, no. 2–3, pp. 119–271, 2016, doi: 10.1561/1500000032.

[36] O. Chapelle, B. Schölkopf, and A. Zien, *Semi-Supervised Learning*. The MIT Press, 2006.

[37] S. J. Pan and Q. Yang, “A Survey on Transfer Learning,” *IEEE Trans. Knowl. Data Eng.*, vol. 22, no. 10, pp. 1345–1359, Oct. 2010, doi: 10.1109/TKDE.2009.191.

[38] and M. K. Han, Jiawei, Jian Pei, *Data mining concepts and techniques*, 3rd Editio. Elsevier, 2011.

[39] M. E. Peters *et al.*, “Deep contextualized word representations,” in *NAACL HLT 2018 - 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies - Proceedings of the Conference*, 2018, vol. 1, pp. 2227–2237, doi: 10.18653/v1/n18-1202.

[40] A. Radford, N. Karthik, S. Tim, and S. Ilya, “Improving Language Understanding by Generative Pre-Training,” 2018.

[41] A. Wang, A. Singh, J. Michael, F. Hill, O. Levy, and S. R. Bowman, “Glue: A multi-task benchmark and analysis platform for natural language understanding,” pp. 1–20, 2018.

[42] A. Wang *et al.*, “SuperGLUE: A Stickier Benchmark for General-Purpose Language

Understanding Systems," vol. 2019, no. July, pp. 1–29, 2019, [Online]. Available: http://arxiv.org/abs/1905.00537.

[43] Y. Liu *et al.*, "RoBERTa: A Robustly Optimized BERT Pretraining Approach," no. 1, 2019, [Online]. Available: http://arxiv.org/abs/1907.11692.

[44] Pandu Nayak, "Understanding searches better than ever before," 2019. https://www.blog.google/products/search/search-language-understanding-bert/ (accessed Jul. 27, 2020).

[45] S. Khan, "BERT, RoBERTa, DistilBERT, XLNet — which one to use? | by Suleiman Khan, Ph.D. | Towards Data Science," 2019. https://towardsdatascience.com/bert-roberta-distilbert-xlnet-which-one-to-use-3d5ab82ba5f8 (accessed Jul. 28, 2020).

[46] A. Vaswani *et al.*, "Attention is all you need," *Adv. Neural Inf. Process. Syst.*, vol. 2017-Decem, no. Nips, pp. 5999–6009, 2017.

[47] "AlienVault - Open Threat Exchange." https://otx.alienvault.com/ (accessed Jul. 28, 2020).

[48] A. Ratner, "Question: Why use the discriminative model at all? · Issue #1059 · snorkel-team/snorkel," 2019. https://github.com/snorkel-team/snorkel/issues/1059 (accessed Jul. 17, 2020).

[49] S. H. Bach *et al.*, "Snorkel Drybell: A case study in deploying weak supervision at industrial scale," in *Proceedings of the ACM SIGMOD International Conference on Management of Data*, 2019, pp. 362–375, doi: 10.1145/3299869.3314036.

[50] E. Bringer, A. Israeli, Y. Shoham, A. Ratner, and C. Ré, "Osprey: Weak Supervision of Imbalanced Extraction Problems without Code," *Proc. ACM SIGMOD Int. Conf. Manag. Data*, 2019, doi: 10.1145/3329486.3329492.

[51] C. Ré, F. Niu, P. Gudipati, and C. Srisuwananukorn, "Overton: A Data System for Monitoring and Improving Machine-Learned Products," pp. 1–13, 2019, [Online]. Available: http://arxiv.org/abs/1909.05372.

[52] N. Mallinar *et al.*, "Bootstrapping Conversational Agents with Weak Supervision," *Proc. AAAI Conf. Artif. Intell.*, vol. 33, pp. 9528–9533, 2019, doi: 10.1609/aaai.v33i01.33019528.

[53] H. R. Ehrenberg, J. Shin, A. J. Ratner, J. A. Fries, and C. Ré, "Data programming with DDLite: Putting humans in a different part of the loop," *HILDA 2016 - Proc. Work. Human-In-the-Loop Data Anal.*, 2016, doi: 10.1145/2939502.2939515.

[54] P. Varma and C. Ré, "Snuba," *Proc. VLDB Endow.*, vol. 12, no. 3, pp. 223–236, Nov. 2018, doi: 10.14778/3291264.3291268.

[55] A. Ratner, S. H. Bach, H. Ehrenberg, J. Fries, S. Wu, and C. Ré, "Snorkel: Rapid training data

creation with weak supervision," *Proc. VLDB Endow.*, vol. 11, no. 3, pp. 269–282, 2017, doi: 10.14778/3157794.3157797.

[56] A. Ratner, B. Hancock, J. Dunnmon, F. Sala, S. Pandey, and C. Ré, "Training Complex Models with Multi-Task Weak Supervision," *Proc. AAAI Conf. Artif. Intell.*, vol. 33, pp. 4763–4771, 2019, doi: 10.1609/aaai.v33i01.33014763.

[57] S. Welch, "Pytorch vs. TensorFlow: What You Need to Know | Udacity," 2020. https://blog.udacity.com/2020/05/pytorch-vs-tensorflow-what-you-need-to-know.html (accessed Jul. 29, 2020).

[58] "Traffic Light Protocol (TLP) Definitions and Usage | CISA." https://us-cert.cisa.gov/tlp (accessed Aug. 12, 2020).

[59] flaboss, "Stratified Sampling in Pandas," 2019. https://github.com/flaboss/python_stratified_sampling/blob/master/stratifiedSample.py (accessed Aug. 13, 2020).

[60] Snorkel, "Intro to Labeling Functions · Snorkel." https://www.snorkel.org/use-cases/01-spam-tutorial (accessed Aug. 14, 2020).

[61] S. H. Bach, B. He, A. Ratner, and C. Ré, "Learning the structure of generative models without labeled data," *34th Int. Conf. Mach. Learn. ICML 2017*, vol. 1, pp. 434–449, 2017.

[62] "Snorkel Community." https://spectrum.chat/snorkel (accessed Aug. 15, 2020).

[63] McCormick Chris and Ryan Nick, "BERT Fine-Tuning Tutorial with PyTorch," 2019. https://mccormickml.com/2019/07/22/BERT-fine-tuning/#3-tokenization--input-formatting (accessed Aug. 15, 2020).

[64] C. Tran, "Tutorial: Fine-tuning BERT for Sentiment Analysis - Skim AI." https://skimai.com/fine-tuning-bert-for-sentiment-analysis/ (accessed Aug. 15, 2020).

[65] "NVIDIA T4 Tensor Core GPU for AI Inference | NVIDIA Data Center." https://www.nvidia.com/en-us/data-center/tesla-t4/ (accessed Aug. 15, 2020).

[66] "Welcome To Colaboratory - Colaboratory." https://colab.research.google.com/notebooks/intro.ipynb#scrollTo=5fCEDCU_qrC0 (accessed Aug. 15, 2020).

[67] H. Tayyar Madabushi, E. Kochkina, and M. Castelle, "Cost-Sensitive BERT for Generalisable Sentence Classification on Imbalanced Data," pp. 125–134, 2019, doi: 10.18653/v1/d19-5018.

[68] "Data Augmentation with Snorkel · Snorkel." https://www.snorkel.org/blog/tanda (accessed Aug. 19, 2020).

[69] "nlp - How to use Bert for long text classification? - Stack Overflow," 2019.

https://stackoverflow.com/questions/58636587/how-to-use-bert-for-long-text-classification (accessed Aug. 16, 2020).

[70] A. Ratner, S. Bach, P. Varma, and C. Ré, "Weak Supervision: The New Programming Paradigm for Machine Learning Our AI is Hungry: Now What?," *Stanford DAWN*, 2018. https://dawn.cs.stanford.edu/2017/07/16/weak-supervision/ (accessed Jun. 24, 2020).

[71] G. Li, J. Wang, Y. Zheng, and M. Franklin, "Crowdsourced data management: A survey," in *Proceedings - International Conference on Data Engineering*, 2017, vol. 28, no. 9, pp. 39–40, doi: 10.1109/ICDE.2017.26.

[72] M. C. Yuen, I. King, and K. S. Leung, "A survey of crowdsourcing systems," *Proc. - 2011 IEEE Int. Conf. Privacy, Secur. Risk Trust IEEE Int. Conf. Soc. Comput. PASSAT/SocialCom 2011*, pp. 766–773, 2011, doi: 10.1109/PASSAT/SocialCom.2011.36.

[73] O. F. Zaidan and C. Callison-burch, "Crowdsourcing Translation: Professional Quality from Non-Professionals," pp. 1220–1229, 2011.

[74] K. Ikeda, A. Morishima, H. Rahman, and S. B. Roy, "Collaborative Crowdsourcing with Crowd4U," pp. 1497–1500, 2016.

[75] J. B. P. Vuurens and A. P. De Vries, "Obtaining High-Quality Relevance Judgments Using Crowdsourcing," 2012.

[76] R. Borromeo *et al.*, "Crowdsourcing Strategies for Text Creation Tasks," 2019.

[77] D. Angluin, "Queries and Concept Learning," 1988.

[78] D. Cohn, L. Atlas, and R. Ladner, "Improving Generalization with Active Learning," *Mach. Learn.*, vol. 15, no. 2, pp. 201–221, 1994, doi: 10.1023/A:1022673506211.

[79] D. D. Lewis and W. A. Gale, "A sequential algorithm for training text classifiers," in *Proceedings of the 17th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 1994*, Aug. 1994, pp. 3–12, doi: 10.1007/978-1-4471-2099-5_1.

[80] A. Fujii, T. Tokunaga, K. Inui, and H. Tanaka, "Selective Sampling for Example-based Word Sense Disambiguation," *Comput. Linguist.*, vol. 24, no. 4, pp. 573–597, 1998.

[81] S. C. H. Hoi, R. Jin, and M. R. Lyu, "Large-scale text categorization by batch mode active learning," *Proc. 15th Int. Conf. World Wide Web*, pp. 633–642, 2006, doi: 10.1145/1135777.1135870.

[82] S. Tong and D. Koller, "Support Vector Machine Active Learning with Applications to Text Classification," *J. Mach. Learn. Res.*, pp. 45–66, 2001.

[83] L. Qian, H. Hui, Y. Hu, G. Zhou, and Q. Zhu, "Bilingual active learning for relation classification via pseudo parallel corpora," *52nd Annu. Meet. Assoc. Comput. Linguist. ACL*

2014 - Proc. Conf., vol. 1, no. 2013, pp. 582–592, 2014, doi: 10.3115/v1/p14-1055.

[84] K. Kowsari, D. E. Brown, M. Heidarysafa, K. Jafari Meimandi, M. S. Gerber, and L. E. Barnes, "HDLTex: Hierarchical Deep Learning for Text Classification," *Proc. - 16th IEEE Int. Conf. Mach. Learn. Appl. ICMLA 2017*, vol. 2017-Decem, pp. 364–371, 2017, doi: 10.1109/ICMLA.2017.0-134.

[85] Y. Chen, T. A. Lasko, Q. Mei, J. C. Denny, and H. Xu, "A study of active learning methods for named entity recognition in clinical text," *J. Biomed. Inform.*, vol. 58, pp. 11–18, 2015, doi: 10.1016/j.jbi.2015.09.010.

[86] F. Laws, C. Scheible, and H. Schütze, "Active learning with amazon mechanical turk," *EMNLP 2011 - Conf. Empir. Methods Nat. Lang. Process. Proc. Conf.*, pp. 1546–1556, 2011.

[87] J. Kranjc, J. Smailović, V. Podpečan, M. Grčar, M. Žnidaršič, and N. Lavrač, "Active learning for sentiment analysis on data streams: Methodology and workflow implementation in the ClowdFlows platform," *Inf. Process. Manag.*, vol. 51, no. 2, pp. 187–203, 2015, doi: 10.1016/j.ipm.2014.04.001.

[88] Y. Shen, H. Yun, Z. C. Lipton, Y. Kronrod, and A. Anandkumar, "Deep active learning for named entity recognition," *6th Int. Conf. Learn. Represent. ICLR 2018 - Conf. Track Proc.*, pp. 1–15, 2018.

[89] J. A. Dunnmon *et al.*, "Cross-Modal Data Programming Enables Rapid Medical Machine Learning," *Patterns*, vol. 1, no. 2, p. 100019, 2020, doi: 10.1016/j.patter.2020.100019.

[90] R. R. Fayzrakhmanov, E. Sallinger, B. Spencer, T. Furche, and G. Gottlob, "Browserless web data extraction: Challenges and opportunities," *Web Conf. 2018 - Proc. World Wide Web Conf. WWW 2018*, no. c, pp. 1095–1104, 2018, doi: 10.1145/3178876.3186008.

[91] T. Gogar, O. Hubacek, and J. Sedivy, "Deep Neural Networks for Web Page Information Extraction," *IFIP Adv. Inf. Commun. Technol.*, vol. 475, pp. VI–VIII, 2016, doi: 10.1007/978-3-319-44944-9.

[92] J. L. Martinez-Rodriguez, A. Hogan, and I. Lopez-Arevalo, *Information Extraction meets the Semantic Web: A Survey*, vol. 11, no. 2. 2020.

[93] J. Hoffart, F. M. Suchanek, K. Berberich, and G. Weikum, "YAGO2: A spatially and temporally enhanced knowledge base from Wikipedia," *Artif. Intell.*, vol. 194, pp. 28–61, 2013, doi: 10.1016/j.artint.2012.06.001.

[94] D. Vrandečić and M. Krötzsch, "Wikidata: A free collaborative knowledgebase," *Commun. ACM*, vol. 57, no. 10, pp. 78–85, Sep. 2014, doi: 10.1145/2629489.

[95] X. Dong *et al.*, "Knowledge vault: A web-scale approach to probabilistic knowledge fusion,"

Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Min., pp. 601–610, 2014, doi: 10.1145/2623330.2623623.

[96] M. Wick, "GeoNames," 2005. http://www.geonames.org/ (accessed Jul. 18, 2020).

[97] W3C, "SPARQL Query Language for RDF," 2008. https://www.w3.org/TR/rdf-sparql-query/ (accessed Jul. 18, 2020).

[98] M. Qu, X. Ren, and Ha, "Automatic Synonym Discovery with Knowledge Bases," vol. 7, no. 1, pp. 45–56, 2017.

[99] B. Yang and T. Mitchell, "Leveraging knowledge bases in LSTMs for improving machine reading," *ACL 2017 - 55th Annu. Meet. Assoc. Comput. Linguist. Proc. Conf. (Long Pap.*, vol. 1, pp. 1436–1446, 2017, doi: 10.18653/v1/P17-1132.

[100] D. Yarowsky, "Unsupervised word sense disambiguation rivaling supervised methods," pp. 189–196, 1995, doi: 10.3115/981658.981684.

[101] I. Triguero, S. García, and F. Herrera, "Self-labeled techniques for semi-supervised learning: Taxonomy, software and empirical study," *Knowl. Inf. Syst.*, vol. 42, no. 2, pp. 245–284, 2015, doi: 10.1007/s10115-013-0706-y.

[102] Z. H. Zhou and M. Li, "Tri-training: Exploiting unlabeled data using three classifiers," *IEEE Trans. Knowl. Data Eng.*, vol. 17, no. 11, pp. 1529–1541, 2005, doi: 10.1109/TKDE.2005.186.

[103] S. Goldman and S. Louis, "Democratic Co-Learning," no. Ictai, 2004.

[104] A. Blum and T. Mitchell, "Combining labeled and unlabeled data with co-training," *Proc. Annu. ACM Conf. Comput. Learn. Theory*, pp. 92–100, 1998, doi: 10.1145/279943.279962.

[105] K. Tomanek and U. Hahn, "Semi-Supervised Active Learning for Sequence Labeling," no. August, pp. 1039–1047, 2009.

[106] M. R. Bouguelia, Y. Belaid, and A. Belaid, "A stream-based semi-supervised active learning approach for document classification," *Proc. Int. Conf. Doc. Anal. Recognition, ICDAR*, pp. 611–615, 2013, doi: 10.1109/ICDAR.2013.126.

[107] M. E. Peters, W. Ammar, C. Bhagavatula, and R. Power, "Semi-supervised sequence tagging with bidirectional language models," *ACL 2017 - 55th Annu. Meet. Assoc. Comput. Linguist. Proc. Conf. (Long Pap.*, vol. 1, pp. 1756–1765, 2017, doi: 10.18653/v1/P17-1161.

[108] T. Miyato, A. M. Dai, and I. Goodfellow, "Adversarial training methods for semi-supervised text classification," *5th Int. Conf. Learn. Represent. ICLR 2017 - Conf. Track Proc.*, no. 2015, pp. 1–10, 2019.

[109] S. Bickel, M. Brückner, and T. Scheffer, "Discriminative learning for differing training and test distributions," *ACM Int. Conf. Proceeding Ser.*, vol. 227, pp. 81–88, 2007, doi:

10.1145/1273496.1273507.

[110] H. Daume III, "Frustratingly easy domain adaptation," *ACL 2007 - Proc. 45th Annu. Meet. Assoc. Comput. Linguist.*, no. June, pp. 256–263, 2007.

[111] W. Dai, Q. Yang, G. R. Xue, and Y. Yu, "Boosting for transfer learning," *ACM Int. Conf. Proceeding Ser.*, vol. 227, pp. 193–200, 2007, doi: 10.1145/1273496.1273521.

[112] A. Argyriou, E. Theodoros, and P. Massimiliano, "Multi-task feature learning," *Adv. Neural Inf. Process. Syst.*, pp. 41–48, 2007.

[113] E. V. Bonilla, K. M. A. Chai, and C. K. I. Williams, "Multi-task Gaussian Process prediction," *Adv. Neural Inf. Process. Syst. 20 - Proc. 2007 Conf.*, 2009.

[114] L. Mihalkova, T. Huynh, and R. J. Mooney, "Mapping and revising Markov logic networks for transfer learning," *Proc. Natl. Conf. Artif. Intell.*, vol. 1, no. July, pp. 608–614, 2007.

[115] J. Blitzer, R. McDonald, and F. Pereira, "Domain adaptation with structural correspondence learning," *COLING/ACL 2006 - EMNLP 2006 2006 Conf. Empir. Methods Nat. Lang. Process. Proc. Conf.*, no. July, pp. 120–128, 2006, doi: 10.3115/1610075.1610094.

[116] J. Blitzer, M. Dredze, and F. Pereira, "Biographies, Bollywood, Boom-boxes and Blenders," *Assoc. Comput. Linguist. - ACL 2007*, no. June, pp. 440–447, 2007.

[117] X. Liao, Y. Xue, and L. Carin, "Logistic regression with an auxiliary data source," *ICML 2005 - Proc. 22nd Int. Conf. Mach. Learn.*, pp. 505–512, 2005, doi: 10.1145/1102351.1102415.

[118] J. Lu, V. Behbood, P. Hao, H. Zuo, S. Xue, and G. Zhang, "Transfer learning using computational intelligence: A survey," *Knowledge-Based Syst.*, vol. 80, pp. 14–23, 2015, doi: 10.1016/j.knosys.2015.01.010.

[119] S. Tan, Y. Wang, G. Wu, and X. Cheng, "Using unlabeled data to handle domain-transfer problem of semantic detection," *Proc. ACM Symp. Appl. Comput.*, pp. 896–903, 2008, doi: 10.1145/1363686.1363893.

[120] M. Ciaramita, S. Zürich, and O. Chapelle, "Adaptive Parameters for Entity Recognition with Perceptron HMMs," *Acl 2010*, no. July, pp. 1–7, 2010, [Online]. Available: http://www.aclweb.org/anthology/W/W10/W10-26.pdf#page=11.

[121] S. Ruder, M. Peters, S. Swayamdipta, and T. Wolf, "Transfer learning in natural language processing tutorial," in *NAACL HLT 2019 - 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies - Tutorial Abstracts*, 2019, no. 2010, pp. 15–18, [Online]. Available: https://www.aclweb.org/anthology/N19-5004/.

[122] S. Ruder, "Neural Transfer Learning for Natural Language Processing," 2019.

[123] "New Cyber Espionage Campaigns Targeting Palestinians - Part 1: The Spark Campaign," 2020. https://www.cybereason.com/blog/new-cyber-espionage-campaigns-targeting-palestinians-part-one (accessed Aug. 21, 2020).

[124] "WildPressure targets industrial-related entities in the Middle East | Kaspersky ICS CERT," 2020. https://ics-cert.kaspersky.com/reports/2020/03/26/wildpressure-targets-industrial-related-entities-in-the-middle-east/ (accessed Aug. 21, 2020).

[125] G. Carl, G. Kesidis, R. R. Brooks, and Suresh Rai, "Denial-of-service attack-detection techniques," *IEEE Internet Comput.*, vol. 10, no. 1, pp. 82–89, Jan. 2006, doi: 10.1109/MIC.2006.5.

[126] Trend Micro USA, "Hash values - Definition - Trend Micro USA." https://www.trendmicro.com/vinfo/us/security/definition/hash-values (accessed Jul. 24, 2020).

Appendices

Appendix A. The three primary literature sources for data labelling

1. The book “Data Classification”

The Data Classification book [24] in the chapters 1.4.4 (Enhancing Classification Methods with Additional Data), 1.4.5 (Incorporating Human Feedback), 11.10 (Leveraging Additional Training Data), 20 (Semi-Supervised Learning), 21 (Transfer Learning) and 22 (Active Learning: A Survey) describe the different fields and methods that deal with unlabelled data.

In summary, they depict four main fields:

Semi-Supervised Learning

Which refer to methods that they leverage unlabelled data when only a small fraction of the labelled data is available. These approaches, aim to gain knowledge from the manifolds that the unlabelled data belong to, as well as from their dense and correlated regions.

Transfer Learning

Learning methods that use previously labelled data from other domains to deal with unlabelled data. The concept is here is to bring knowledge from previous similar classification tasks to generate to label a new set of instances. Transfer Learning differs from Semi-Supervised Learning mainly because the former uses a feature set from a different source while the latter deals with the same set of features.

Active Learning

Methods which aim to get more labels at the learning stage in order to improve the performance of the classification task. In this setting, the Active Learning algorithms typically request from a human to annotate specific instances to

develop a classification model. In reality, these approaches rely on human annotators.

Visual Learning

Algorithms which share similarities with Active Learning Algorithms and request visual feedback from a human to develop a model. In this case, Visual Learning may request from a human to give specific parameters for developing a classification model (after reviewing a visual representation of the data), or it may ask feedback for dealing with specific instances.

2. Hazy Research Team

Hazy research is Stanford's research group led by Professor Chris Ré [25]. The team focuses on weak forms of supervision in the field of Machine Learning.

In their paper of Snorkel framework for data labelling [55] as well as in their corresponding blog post [70] the authors depicted the different forms of learning algorithms and types of supervision for generating labelled training data. In their study, they defined three main types of learning algorithms for data labelling; Active Learning, Semi-Supervised Learning and Transfer Learning.

Weak Supervision with Data Programming

Machine Learning algorithms which deal with imprecise labels. In the traditional supervised setting, the labels (y) for developing a predictive model are regarded as ground-truth. In the case of Weak Supervision, these labels may be imprecise, contain noise, and generally are considered of lower-quality compared to ground-truth labels. However, the cost in terms of human effort for generating these labels is comparatively lower to ground-truth labels. This method can be beneficial when we have a large amount of data.

3. Survey on Data Collection for Machine Learning

The survey conducted by [26] defines three sources for gaining data for Machine Learning tasks. These sources are a) the data acquisition of new datasets, b) the data labelling of existing datasets and c) the use of existing labelled data. In our study, we examined more thoroughly the second and the third source.

For the part of data labelling of existing datasets, the authors provide a summary of the different approaches to deal with unlabelled data. They classify these approaches according to the following structure:

I. Data with no labels
    a. Manual Labelling (human-in-the-loop to annotate instances)
        i. Active Learning
        ii. Crowdsourcing
    b. Weak Labelling (no human-in-the-loop to annotate instances)
        i. Data Programming
        ii. Fact Extraction

II. Data with some Labels
        i. Semi-Supervised Learning

In this section, many elements of Active Learning (1.1.1) and Semi-Supervised Learning (2.1) are similar to those of the book "Data Classification" [24]. The same applies to the Data Programming (1.2.1) of Weak Labelling, which relies on the work of the Hazy Research team [25] and more specifically on the work of [33].

However, the study includes additional ways to label data. For Manual Labelling, this is the Crowdsourcing (1.1.2) and for the Weak Labelling is the Fact Extraction (1.2.2). Below we provide a brief overview for them.

Crowdsourcing

Refer to techniques where many human annotators label instances. These annotators do not need to have expertise in annotating the labels. In reality, crowdsourcing techniques expect that annotators may make mistakes. The different crowdsourcing approaches aim to overcome this drawback with several strategies (such us with the majority vote from all annotations for the same instance).

Fact Extraction

The field of Fact Extraction aims to create weak labels with the use of knowledge bases or web sources. The knowledge bases may be in our possession, or other providers may offer them. These knowledge bases can provide indirect information for the instance that we want to annotate. For example, if we have a dataset which has socio-economic indicators for different cities, a knowledge base can easily provide us with the country which they belong to. Web sources, on the other hand, can be exploited with the use of web scrappers. The latter are programs that autonomously access web sources and extract specific knowledge from them.

In the third part of the survey ("use of existing labelled data"), the authors classify the approaches as following:

I. Improve data
    a. Data Cleaning
    b. Re-labelling
II. Improve model
    a. Make model robust
    b. Transfer Learning

For our particular problem of data labelling, we distinguish the Transfer Learning (2.2) from this section. Again many of the concepts introduced in this chapter are in alignment with the book "Data Classification" [24].

Overall, the survey examines the data labelling problem from a higher perspective. In reality, it aligns the different approaches with different business settings. For example, a company which has a limited budget can directly realise that Active Learning and Crowdsourcing approaches may not be feasible. In another setting, a business which owns already a predictive model for a specific task can exploit Transfer Learning to develop a new model for a similar task.

Appendix B. Some additional points for Crowdsourcing

Crowdsourcing task can be hosted in one of the many available online Crowdsourcing Platforms, and two kinds of users will interact with it; the "requester" of the labelling task and the "workers" who will annotate the dataset [71].

Several aspects have to be considered from the "requester" of the task. For example, which Crowdsourcing Platform is appropriate for the annotation task, what kind and how many "workers" should be employed for the task, as well as others other aspects to ensure the high quality of the final labels.

The survey of [31] examines the aspects which ensure the high quality of the labels. The dataset provided for annotation should be timely and recent to the final classification task. The annotating task itself has to have a clear description, an easy interface for the annotators to work on, as well as incentives for the annotators in order to deliver high-quality results. Finally, the annotators should be selected with caution, considering their prior experience in labelling tasks as well as their societal and educational background. These aspects are not exhaustive, but they give a sense of how the quality of labels may be impacted by all the labelling process.

As it was mentioned, there are several Crowdsourcing Platforms to host such tasks. Many surveys [31], [71], [72] refer as a first general-purpose choice the Amazon Mechanical Turk. In contrast, the survey of [30] covers Crowdsourcing Platforms which support special requirements or labelling tasks.

In the fields of NLP and text mining, many applications have been presented for translation tasks [73], [74]. In addition, studies for crowdsourcing judgements about the relevance of documents to a query [75], as well as for summarisation of documents [76].

Appendix C. Active Learning query strategies and applications in NLP.

There are three well-known query strategies in Active Learning, these are the Membership Query Synthesis [77], the Stream-based Selective Sampling [78] and the Pool-based Active Learning [79].

Membership query synthesis

In this occasion, the active learner may make queries with new instances that are synthesised from a combination of other existing unlabelled instances. However, this approach may lead to problems for a human annotator. For example, in the field of natural language processing, the annotator may have to classify reviews that are a combination of others. These potentially generated instances may be nonsensical and unclear for the annotator to label. Successful applications with this approach are used in the biology field, where robots have to conduct experiments in order to annotate a new instance.

Stream-based Selective Sampling

Referred also as Sequential Active Learning, is the category of algorithms which examine each unlabelled instance one-by-one and decide if they want to ask the annotator to classify it. This approach has been used for NLP tasks such as the task of [80], which aimed to learn the ambiguines of same words in different contexts (for example the meaning of the word “bank” for a river or a business). In general, this approach is preferred when the computational power is limited.

Pool-Based Active Learning

This approach calculates a metric for all unlabelled instances, ranks them and then select the best candidate to annotate. Successful applications in the NLP domain have been presented in [81] which categorise web sites based on Logistic

Regression, as well as in [82] which classifies news articles with the use of Support Vector Machines. It is worth to mention that Pool-Based Active Learning can be considered as the opposite case of Stream-based Selective Sampling, where each instance is examined one-by-one. As it is evident, the former approach requires higher computational resources [28].In general, the greatest amount of literature deals with Pool-Based Active Learning [24].

In NLP domain, we found several applications with Active Learning. Active Learning for translation tasks [83], text classification [84] and Named Entity Recognition [85].

Also, Active Learning in the domain of NLP has been examined in applications together with the crowdsourcing platform Amazon Mechanical Turk for Named Entity Recognition [86], with streaming data and Crowdsourcing for sentiment analysis [87] and with Deep Networks for Named Entity Recognition [88].

Appendix D. Applications with Data Programming in NLP

Some applications with Data Programming in the NLP field include a topic and a product classification [49], a relation extraction from tweets [50], a classification of diagnoses in the medical field [89], as well as an application for question-answering tasks [51].

Appendix E. Fact Extraction methods and their applications in NLP

Web Sources

The different web wrappers may differ in the way that they are developed, operate but also maintained. The creation and the execution of them may rely on definitions that were manually created from the user (for example, regular-expression or logic rules) or they may exploit already pre-defined higher-level automation strategies. For the maintenance of them (as the web sources may change after some period) also different strategies have been proposed [34].

In general, the aforementioned survey of [34] on Web Data Extraction applications, notes several challenges for using such approaches. For example, different web wrapper techniques should eliminate their needs for human interventions. Compared to our data labelling problem, if the Fact Extraction process takes more to be deployed or maintained rather than annotating our dataset directly, then the former approach may not offer direct benefits. Also, web wrapper approaches should be able to process large chunks of data promptly. This point makes sense, especially for our Machine Learning tasks, where we need a high number of labels.

Some studies which use wrappers to extract textual information are the following:

- The study [90] which demonstrates a wrapper that interacts with the web pages that visits; it uses search boxes, click on links and extract information from the results page.

- The study of [91] which uses a hybrid approach and considers both the HTML code together with the visual representation (screenshot) of the web page [91]

However, we believe that these are only a small portion of all the successful applications in the wild. Companies or individuals may develop their wrapper methods to complete specific tasks as there is diversity on the available web content.

Knowledge Bases

The survey of Data Collection [26], but also relevant surveys for knowledge bases [35], [92] refer as most popular knowledge bases those that rely on data from WikiPedia. These are the YAGO2 [93] and Wikidata [94] a successor of Google's knowledge base Freebase. Knowledge Vault [95] is another knowledge base, which combines data from the world wide web with knowledge from previous knowledge bases. Except these, there are other more specialised knowledge bases such as GeoNames [96] which offer geospatial data.

Nevertheless, as the size of these knowledge bases can be significantly large, different "Entity Extraction and Linking systems" may be used to exploit them [92]. Based on the way that we want to access records from these knowledge bases (for example through keywords, substrings or exact keywords) and the tasks that we want to carry over, we may need to select a specific system (in a sense a database system) that use one of the available knowledge bases. Alternatively, knowledge bases may be accessed remotely with the use of query languages such as SPARQL [97].

Some successful applications in the field of text mining are the automatic synonym discovery from knowledge bases [98] as well as the use of knowledge bases for building LSTM neural networks for entity extraction [99]. Again, as with the knowledge extraction from web sources, we believe that they are much

more successful tailored-made applications for specific problems, however, not all of them are shared in academia.

Appendix F. Semi-Supervised learning approaches and applications in NLP.

Several approaches have been proposed for dealing with problems in the Semi-Supervised setting. The book of "Data Mining concepts and techniques" [38] describes the very early concept of self-training [100], which is considered as the simplest form of Semi-Supervision. With this approach, a classifier is trained with the labelled subset, and the model classifies all of the unlabelled data. Then, the most confident predictions are selected, and the model is re-trained, including the pre-existing labelled data. Finally, the same concept is repeated to the rest of the unlabelled data. In a sense, the concept is to incrementally add the most confident predictions to the model [24].

The empirical survey of [101] in self-labelled techniques for semi-supervised learning examines different techniques, especially for the data labelling problem. The authors tested 55 datasets from different domains with several semi-supervised techniques. Below we provide a short description of the best-performing techniques across all datasets.

- Self-Training [100], an approach which was described above. Although it follows a simple concept, it has shown to provide high-performing results in many different datasets.
- Tri-training [102], a method which trains three models of the same type on the labelled dataset using the bagging method. The final labels are generated through majority voting.
- Democratic Co-learning [103], which trains different types of models (for example k-neighbours with naïve Bayes) on the same labelled dataset. The annotation of the unlabelled dataset is based on weighted voting

from these models. The new annotated data are re-trained with the pre-labelled data.

- Co-training [104], which trains two models with different splits of the feature set of the labelled data. These models aim to denoise their disagreements over their predictions on the unlabelled data.

Semi-Supervised Learning has also been studied with Active Learning from several studies. Some examples in the NLP domain include the work of [105] for sequence labelling task as well as the application of [106] which deals with stream-based data for document classification.

In more recent studies, Semi-Supervised learning has been used with neural networks. Some examples include the work of [107], which deals with bidirectional language models for sequence tagging as well as the work of [108], which deals with adversarial examples for text classification.

Appendix G. Transfer Learning approaches and applications in NLP

The survey on Transfer Learning of [37], classify the Transfer Learning approaches according to two aspects. The first refers to the source and target domains and the second to the source and target tasks. In this setting, domain refers to the actual data collection (for examples university webpages), and the task refers to the actual predictive task (for example, categorise different webpages) [26]. Based on these aspects, we have three main branches of "Transfer Learning".

The first branch, called "Inductive Transfer Learning" refers to the case where the source and the target domain are the same (for example when we have two sets of documents of the same language) but our source and target tasks are different (for example those documents focus on different topics).

The second branch, called "Transductive Transfer Learning" is the opposite case of "Inductive Transfer Learning"; the source and the target domain are different (for example documents from different languages), and the source and the target tasks are the same (for example documents that focus on a common topic).

Finally, the last branch, "Unsupervised Transfer Learning" refers to the setting where both source and target domains and tasks are different.

One additional aspect that we have to consider for these branches is the availability of labels. For our data labelling problem, we expect that we will have labelled data in our source domain and what we seek for, are labels in our target domain.

With this scenario, all approaches which belong to the branch of "Transductive Transfer Learning" can be examined, where for "Inductive Transfer Learning"

only some of them. This suggestion is based on Table 2 of the work of [37]. Below we collect some approaches for each branch that match with our labelling problem:

Transductive Transfer Learning

1. Discriminative Learning for Differing Training and Test Distributions [109] which uses re-weighting methods for co-variate shift.
2. Domain Adaptation [110], which uses a kernel-mapping function to map data to a high-dimensional feature space from both the source and target data. This approach aims mostly on NLP problems.

Inductive Transfer Learning

1. Boosting for Transfer Learning (TrAdaBoost) [111] which aims through several iterations to adjust the weights of the classifier of the labelled data so to fit with the unlabeled data.
2. Multi-Task Feature Learning [112] which tries to find a proper representation of features that reduces the classification error between the models for the source and target domains.
3. Multi-task Gaussian Process prediction [113] which belong to a category of approaches which aim to find common parameters between the source and domain model.
4. Markov Logic Networks for Transfer Learning [114] which deals mainly with network data and transfers the relationships of the source domain data to the target domain.

The survey of [37] also provides some successful applications of Transfer Learning in the NLP domain with the absence of target labels. The work of [115] shows how a part of speech tagger for financial news can be used for biomedical abstracts with the use of Transfer Learning. The paper of [116] shows to adjust sentiment classifiers to products of other categories.

Transfer Learning has also been studied with Active Learning for data labelling problems. The work of [117] demonstrated an application of Transfer Learning together with Active Learning for creating labels for unlabelled datasets.

A more recent survey from [118] collected NLP applications with Self-Labeling. Methods, which belong to the field of domain adaptation which is part of Transductive Transfer Learning. These methods generate labels for the target domain through the source domain. Some applications with Self-Labelling have been presented in [119] where different sentiment classifiers for computer, education and house reviews are adjusted for the rest topics. Also, in the work of [120], an entity recognition model for general news documents is adjusted and evaluated for financial documents.

It is noteworthy that in the NLP field, Transfer Learning is not useful only for the case where we have labels for our source domain. The Proceedings of a recent conference [121] and a PhD thesis [122] refer to NLP models which are based in Sequential Transfer Learning. Sequential Transfer Learning which is part of the Inductive Transfer Learning, works with unlabelled data from the source domain and labelled data in the target domain. Several well-known pre-trained language models such as ULMFit and BERT (which are discussed in chapter 2.3) also belong to this field.

Appendix H. Open Cyber Threat Intelligence Feeds

Below we provide a list of the open threat feeds that we have examined.

Name	Link	Comments
Bruteforce blocker	http://danger.rulez.sk/projects/bruteforceblocker/	No recent data
Bambenek Consulting	http://osint.bambenekconsulting.com/feeds/	No batch access
Blocklist	http://lists.blocklist.de/lists/all.txt	No descriptions, only IOCs
Project Honeypot	https://www.projecthoneypot.org/list_of_ips.php?t=p	Only recent IP, no descriptions
OpenPhish	https://openphish.com/	Only domains, no descriptions
PhishTank	https://www.phishtank.com/phish_archive.php	Only domains, no descriptions
ScumWare	https://www.scumware.org/#	Contains threat types, but no descriptions or batch download
Charles B. Haley's collection	http://charles.the-haleys.org/ssh_dico_attack_hdeny_format.php/hostsdeny.txt	Only past events
SSL Blacklist	https://sslbl.abuse.ch/blacklist/#botnet-c2-ips-suricata	Wide coverage of threats although no descriptions, and only events of the last 30 days
Cybercrime Tracker	http://cybercrime-tracker.net/	No descriptions
VirusTotal	https://www.virustotal.com/	Informative, but considers IOCs independently. Batch download of last year only for academics
SANS Internet	https://isc.sans.edu/	Includes False Positives; the threat data are not reliable.

These feeds come from articles that have collections of open feeds:

- https://www.misp-project.org/feeds/
- https://threatfeeds.io/
- https://www.senki.org/operators-security-toolkit/open-source-threat-intelligence-feeds/

Appendix I. Total retrieval time and quality metrics for different intelligence feeds from AlienVault API

Threat Intelligent Feeds	Total Retrieval Time
METADEFNDER	1hr 50min
BOTNETEXPOSER	27min
JNAZARIO	14min
YARA_MATCHES	23min
MALWAREPATROL	15min
POPULARMALWARE	15min
ALIENVAULT	3min

Threat Intelligent Feeds	Date of first record	Date of last record	% missing short descriptions	% missing external links	Average number of words in short descriptions	Standard deviation of number of words in short descriptions	Number of Subscribers in AlienVault
METADEFNDER	05-2017	04-2020	0.24	0	24.8	4.4	594
BOTNETEXPOSER	03-2019	04-2020	0.01	100	24.7	5.4	309
JNAZARIO	06-2016	05-2020	0.15	0	10	4.2	999
YARA_MATCHES	07-2018	07-2020	0	100	7	0.5	309
MALWAREPATROL	08-2016	08-2020	0	100	24	6.6	1110
POPULARMALWARE	08-2016	08-2020	0	100	24	6.6	646
ALIENVAULT	11-2014	06-2020	9	3	67	40	3123

Appendix J. Stratified Sampling Analysis of pulses' descriptions

For creating our development and test subsets, we calculated how many observations equal to 20%. In this case, we ended up with 404 pulses in total.

Then we requested from the sampling algorithm [59] to select the appropriate instances which represent all years, months and description's length quantiles.

Below we provide a comparison of first level (year) and the third level (length quantiles) between the full dataset and the stratified sample.

Year		Descriptions' length quantiles	
Full Dataset %	Stratified Sample %	Full Dataset %	Stratified Sample %
2020 9.86	2020 10.11		
2019 30.06	2019 30.34	(16.0, 50.0] 26.25	(16.0, 50.0] 26.52
2018 21.64	2018 22.25	(50.0, 73.0] 24.22	(50.0, 73.0] 24.04
2017 17.14	2017 16.85	(73.0, 102.0] 24.86	(73.0, 102.0] 24.94
2016 10.65	2016 9.89	(102.0, 159.0] 24.67	(102.0, 159.0] 24.49
2015 10.60	2015 10.56		
2014 0.05			

From the table, we realise that the subset is a representation of the full dataset. In order to have a proper representation, the algorithm used more observations to create the stratified sample. The stratified sample contained 445 observations.

In the same fashion, from the stratified sample of the 445 observations, we generated our final development and test subsets. The development set ended up with 278 observations and test set with 167.

Descriptions' length quantiles		
Full Dataset %	Development Set %	Test Set %
(16.0, 50.0] 26.25	(15.999, 50.0] 25.18	(15.999, 50.0] 28.74
(50.0, 73.0] 24.22	(50.0, 73.0] 24.82	(50.0, 73.0] 22.75
(73.0, 102.0] 24.86	(73.0, 102.0] 25.18	(73.0, 102.0] 24.55
(102.0, 159.0] 24.67	(102.0, 159.0] 24.82	(102.0, 159.0] 23.95

Appendix K. Examples of short descriptions and annotations

The suggested labels have been produced while we annotated the first one hundred observations of the development subset.

1. *"Over the last several months, the Cybereason Nocturnus team has been tracking recent espionage campaigns targeting the Middle East. These campaigns are specifically directed at entities and individuals in the Palestinian territories. This investigation shows multiple similarities to previous attacks attributed to a group called MoleRATs (aka The Gaza Cybergang), an Arabic-speaking, politically motivated group that has operated in the Middle East since 2012."* [123].

Targeted	Media	Government	Industry/Businesses
1	0	0	0
Finance	Banking	Energy	Crypto
0	0	0	0
Military	Healthcare	Critical Infrastructure	Espionage
0	0	0	1
Refers to a previous attack	Creator Physical Known	Target Physical Known	IT Infrastructure
1	1	1	0

2. "*In August 2019, Kaspersky discovered a malicious campaign distributing a fully fledged C++ Trojan that we call Milum. All the victims we registered were organizations from the Middle East. At least some of them are related to industrial sector. Our Kaspersky Threat Attribution Engine (KTAE) doesn't show any code similarities with known campaigns. Nor have we seen any target intersections. In fact, we found just three almost unique samples, all in one country. So we consider the attacks to be targeted and have currently named this operation WildPressure.*"[124]

Targeted	Media	Government	Industry/Businesses
1	0	0	1
Finance	Banking	Energy	Crypto
0	0	0	0
Military	Healthcare	Critical Infrastructure	Espionage
0	0	0	0
Refers to a previous attack	Creator Physical Known	Target Physical Known	IT Infrastructure
0	0	1	0

Appendix L. Keywords that indicate a positive label

The parts of speech (PoS) are defined according to the Universal Dependencies Framework[8]

Label Targeted: If the description refers to an attack or an activity that targeted a specific entity or group of people.

	Keyword	PoS
0	military	ADJ
1	firm	ADJ
2	against	ADP
3	defence	NOUN
4	company	NOUN
5	country	NOUN
6	industry	NOUN
7	politic	NOUN
8	target	NOUN
9	victim	NOUN
10	government	NOUN
11	bank	PROPN
12	breach	PROPN
13	focus	PROPN
14	steal	VERB
15	target	VERB
16	aim	VERB
17	focus	VERB

[8] https://universaldependencies.org/u/pos/

Label Instance refers to previous attack: If the malicious activity is based on previous malicious code, malware families or if the documents describe a similarity with other previous attacks.

	Keyword	PoS
0	recent	ADJ
1	previous	ADJ
2	similar	ADJ
3	old	ADJ
4	ongoing	ADJ
5	variant	ADJ
6	last	ADJ
7	ago	ADV
8	previously	ADV
9	originally	ADV
10	first	ADV
11	early	ADV
12	back	ADV
13	be	AUX
14	another	DET
15	borrowing	NOUN
16	reemergence	NOUN
17	predecessor	NOUN
18	Appeared	PROPN
19	Successor	PROPN
20	since	SCONJ
21	while	SCONJ
22	wave	VERB
23	observe	VERB
24	tie	VERB
25	reuse	VERB
26	develop	VERB
27	update	VERB

Label Attack with Espionage Intentions: Attack which aims to spy specific persons or organisations.

	Keyword	PoS
0	data	NOUN
1	spy	NOUN
2	cyberespionage	NOUN
3	espionage	NOUN
4	breach	PROPN

Appendix M. Labelling Functions

Label Targeted: If the description refers to an attack or an activity that targeted a specific entity or group of people.

```
ABSTAIN = -1
NEGATIVE = 0
POSITIVE = 1

@labeling_function()
def ADJ_military(x):
    return POSITIVE if re.search(r"military.*", x.description, flags=re.I) else NEGATIVE

@labeling_function()
def ADJ_firm(x):
    return POSITIVE if re.search(r"firm.*", x.description, flags=re.I) else NEGATIVE

@labeling_function()
def ADP_against(x):
    return POSITIVE if re.search(r"against.*", x.description, flags=re.I) else NEGATIVE

@labeling_function()
def NOUN_defense(x):
    return POSITIVE if re.search(r"defense.*", x.description, flags=re.I) else NEGATIVE

@labeling_function()
def NOUN_company(x):
    return POSITIVE if re.search(r"company.*", x.description, flags=re.I) else NEGATIVE

@labeling_function()
def NOUN_countries(x):
    return POSITIVE if re.search(r"countries.*", x.description, flags=re.I) else NEGATIVE

@labeling_function()
def NOUN_industry(x):
    return POSITIVE if re.search(r"industry.*", x.description, flags=re.I) else NEGATIVE
```

```
@labeling_function()
def NOUN_politics(x):
    return POSITIVE if re.search(r"politics.*", x.description, flags=re.I) else NEGATIVE

@labeling_function()
def NOUN_target(x):
    return POSITIVE if re.search(r"target.*", x.description, flags=re.I) else ABSTAIN

@labeling_function()
def NOUN_government(x):
    return POSITIVE if re.search(r"government.*", x.description, flags=re.I) else NEGATIVE

@labeling_function()
def PROPN_bank(x):
    return POSITIVE if re.search(r"bank.*", x.description, flags=re.I) else NEGATIVE

@labeling_function()
def PROPN_breach(x):
    return POSITIVE if re.search(r"breach.*", x.description, flags=re.I) else NEGATIVE

@labeling_function()
def PROPN_focus(x):
    return POSITIVE if re.search(r"focus.*", x.description, flags=re.I) else ABSTAIN

@labeling_function()
def VERB_focusing(x):
    return POSITIVE if re.search(r"focusing.*", x.description, flags=re.I) else ABSTAIN

@labeling_function()
def VERB_steal(x):
    return POSITIVE if re.search(r"steal.*", x.description, flags=re.I) else ABSTAIN

@labeling_function()
def VERB_stolen(x):
    return POSITIVE if re.search(r"stolen.*", x.description, flags=re.I) else ABSTAIN
```

```
@labeling_function()
def VERB_targets(x):
    return POSITIVE if re.search(r"targets.*", x.description, flags=re.I) else NEGATIVE

@labeling_function()
def VERB_targeting(x):
    return POSITIVE if re.search(r"targeting.*", x.description, flags=re.I) else NEGATIVE

@labeling_function()
def VERB_aim(x):
    return POSITIVE if re.search(r"aim.*", x.description, flags=re.I) else NEGATIVE

@labeling_function()
def VERB_aimed(x):
    return POSITIVE if re.search(r"aimed.*", x.description, flags=re.I) else NEGATIVE

#Named Entity Recognition with SPACY
@labeling_function(pre=[spacy])
def SPACY_GPE(x):
    if any([ent.label_ == "GPE" for ent in  x.doc.ents]):
        return POSITIVE
    else:
        return NEGATIVE

@labeling_function(pre=[spacy])
def SPACY_LANGUAGE(x):
    if any([ent.label_ == "LANGUAGE" for ent in x.doc.ents]):
        return POSITIVE
    else:
        return NEGATIVE
```

Label Instance refers to previous attack: If the malicious activity is based on previous malicious code, malware families or if the documents describe a similarity with other previous attacks.

```
ABSTAIN = -1
NEGATIVE = 0
POSITIVE = 1

#ROUND ONE
#FROM SUMMARY STATISTICS

@labeling_function()
def SUM_word_count(x):
    return POSITIVE if x.desc_word_count>71 else NEGATIVE

@labeling_function()
def SUM_targeted_countries(x):
    return NEGATIVE if x.targeted_countries_ind==1 else ABSTAIN

@labeling_function()
def SUM_tags_ind(x):
    return POSITIVE if x.tags_ind==0 else ABSTAIN

@labeling_function()
def SUM_industries_ind(x):
    return NEGATIVE if x.industries_ind==1 else ABSTAIN

@labeling_function()
def SUM_total_ref(x):
    return NEGATIVE if x.total_ref>2 else POSITIVE

#CUSTOM YEAR COMPARISON HEURISTIC
@labeling_function()
def CUSTOM_year(x):
extract=np.array(re.findall(r"\D(\d{4})\D",x.description.lower()),np.int64)
    extract = extract[(extract>1950) & (extract<2020)]
    return POSITIVE if any(extract<x.year) else ABSTAIN
```

```
#SEARCH IN THESAURUS
threat_theausarus = [
   'admin@338', 'APT-C-36', 'APT1', 'APT12', 'APT16', 'APT17', 'APT18', 'APT19', 'APT28', 'APT29',
    'APT3', 'APT30', 'APT32', 'APT33', 'APT37', 'APT38', 'APT39', 'APT41', 'Axiom', 'BlackOasis',
    'BlackTech', 'Blue Mockingbird', 'Bouncing Golf', 'BRONZE BUTLER', 'Carbanak', 'Charming Kitten',
    'Cleaver', 'Cobalt Group', 'CopyKittens', 'Dark Caracal', 'Darkhotel', 'DarkHydrus', 'DarkVishnya',
    'Deep Panda', 'Dragonfly', 'Dragonfly 2.0', 'DragonOK', 'Dust Storm', 'Elderwood', 'Equation',
    'FIN10','FIN4', 'FIN5', 'FIN6', 'FIN7', 'FIN8', 'Frankenstein', 'Gallmaker', 'Gamaredon Group',
    'GCMAN', 'Gorgon Group', 'Group5', 'Honeybee', 'Inception', 'Ke3chang', 'Kimsuky', 'Lazarus Group',
    'Leafminer', 'Leviathan', 'Lotus Blossom', 'Machete', 'Magic Hound', 'menuPass', 'Moafee',
    'Mofang', 'Molerats', 'MuddyWater', 'Naikon', 'NEODYMIUM', 'Night Dragon', 'OilRig', 'Orangeworm',
    'Patchwork', 'PittyTiger', 'PLATINUM', 'Poseidon Group', 'PROMETHIUM', 'Putter Panda', 'Rancor',
    'Rocke', 'RTM', 'Sandworm Team', 'Scarlet Mimic', 'Sharpshooter', 'Silence', 'SilverTerrier',
    'Soft Cell', 'Sowbug', 'Stealth Falcon', 'Stolen Pencil', 'Strider', 'Suckfly', 'TA459', 'TA505',
    'Taidoor', 'TEMP.Veles', 'The White Company', 'Threat Group-1314', 'Threat Group-3390', 'Thrip',
    'Tropic Trooper', 'Turla', 'Whitefly', 'Windshift', 'Winnti Group', 'WIRTE', 'Wizard Spider'
]


@labeling_function()
def THESAURUS_threat_agents(x):
    return POSITIVE if any(word in x.description_lemma_spacy for word in threat_theausarus) else
NEGATIVE



# KEYWORDS
@labeling_function()
def ADJ_recent(x):
    return POSITIVE if "recent" in x.description_lemma_spacy else ABSTAIN


@labeling_function()
def ADJ_previous(x):
    return POSITIVE if "previous" in x.description_lemma_spacy else NEGATIVE


@labeling_function()
def ADJ_similar(x):
    return POSITIVE if "similar" in x.description_lemma_spacy else NEGATIVE
```

```
@labeling_function()
def ADJ_old(x):
    return POSITIVE if "old" in x.description_lemma_spacy else NEGATIVE

@labeling_function()
def ADJ_ongoing(x):
    return POSITIVE if "ongoing" in x.description_lemma_spacy else NEGATIVE

@labeling_function()
def ADJ_variant(x):
    return POSITIVE if "variant" in x.description_lemma_spacy else NEGATIVE

@labeling_function()
def ADJ_last(x):
    return POSITIVE if "last" in x.description_lemma_spacy else ABSTAIN

@labeling_function()
def ADV_previously(x):
    return POSITIVE if "previously" in x.description_lemma_spacy else NEGATIVE

@labeling_function()
def ADV_originally(x):
    return POSITIVE if "originally" in x.description_lemma_spacy else ABSTAIN

@labeling_function()
def ADV_first(x):
    return POSITIVE if "first" in x.description_lemma_spacy else ABSTAIN

@labeling_function()
def ADV_ago(x):
    return POSITIVE if "ago" in x.description_lemma_spacy else ABSTAIN

@labeling_function()
def ADV_early(x):
    return POSITIVE if "early" in x.description_lemma_spacy else ABSTAIN
```

```
@labeling_function()
def ADV_back(x):
    return POSITIVE if "back" in x.description_lemma_spacy else ABSTAIN

@labeling_function()
def NOUN_reemergence(x):
    return POSITIVE if "reemergence" in x.description_lemma_spacy else NEGATIVE

@labeling_function()
def NOUN_predecessor(x):
    return POSITIVE if "predecessor" in x.description_lemma_spacy else NEGATIVE

@labeling_function()
def NOUN_borrowing(x):
    return POSITIVE if "borrowing" in x.description_lemma_spacy else ABSTAIN

@labeling_function()
def DET_another(x):
    return POSITIVE if "another" in x.description_lemma_spacy else ABSTAIN

@labeling_function()
def PROPN_successor(x):
    return POSITIVE if "successor" in x.description_lemma_spacy else ABSTAIN

@labeling_function()
def PROPN_appeared(x):
    return POSITIVE if "appeared" in x.description_lemma_spacy else NEGATIVE

@labeling_function()
def VERB_wave(x):
    return POSITIVE if "wave" in x.description_lemma_spacy else ABSTAIN

@labeling_function()
def VERB_observe(x):
    return POSITIVE if "observe" in x.description_lemma_spacy else ABSTAIN
```

```
@labeling_function()
def VERB_tie(x):
    return POSITIVE if "tie" in x.description_lemma_spacy else ABSTAIN

@labeling_function()
def VERB_develop(x):
    return POSITIVE if "develop" in x.description_lemma_spacy else ABSTAIN

@labeling_function()
def VERB_update(x):
    return POSITIVE if "update" in x.description_lemma_spacy else NEGATIVE

@labeling_function()
def FIX_was_first(x):
    return POSITIVE if "was first" in x.description.lower() else ABSTAIN

@labeling_function()
def FIX_dating_back(x):
    return POSITIVE if "dating back" in x.description.lower() else ABSTAIN

@labeling_function()
def FIX_first_observed(x):
    return POSITIVE if "first observed" in x.description.lower() else ABSTAIN

@labeling_function()
def FIX_was_active(x):
    return POSITIVE if "was active" in x.description.lower() else ABSTAIN

@labeling_function()
def FIX_source_code(x):
    return POSITIVE if "source code" in x.description.lower() else ABSTAIN

@labeling_function()
def FIX_until_now(x):
    return POSITIVE if "until now" in x.description.lower() else ABSTAIN
```

```
@labeling_function()
def ORIG_earlier(x):
    return POSITIVE if "earlier" in x.description.lower() else ABSTAIN

@labeling_function()
def ORIG_observed(x):
    return POSITIVE if "observed" in x.description.lower() else ABSTAIN

@labeling_function()
def ORIG_tied(x):
    return POSITIVE if "tied" in x.description.lower() else ABSTAIN

@labeling_function()
def ORIG_developing(x):
    return POSITIVE if "developing" in x.description.lower() else ABSTAIN

@labeling_function()
def ORIG_updated(x):
    return POSITIVE if "updated" in x.description.lower() else ABSTAIN

@labeling_function()
def EXP_groups(x):
    return POSITIVE if "groups" in x.description.lower() else NEGATIVE

@labeling_function()
def EXP_family(x):
    return POSITIVE if "family" in x.description.lower() else NEGATIVE

@labeling_function()
def EXP_used(x):
    return POSITIVE if "used" in x.description.lower() else ABSTAIN

@labeling_function()
def EXP_share(x):
    return POSITIVE if "share" in x.description.lower() else ABSTAIN
```

```
@labeling_function()
def EXP_modified(x):
    return POSITIVE if "modified" in x.description.lower() else NEGATIVE

@labeling_function()
def EXP_subsequent(x):
    return POSITIVE if "subsequent" in x.description.lower() else ABSTAIN

@labeling_function()
def EXP_first_appeared(x):
    return POSITIVE if "first appeared" in x.description.lower() else ABSTAIN

@labeling_function()
def EXP_based(x):
    return POSITIVE if "based" in x.description.lower() else NEGATIVE

#SECOND ROUND
#KEYWORDS

@labeling_function()
def SCONJ_since(x):
    return POSITIVE if "since" in x.description_lemma_spacy else NEGATIVE

@labeling_function()
def SCONJ_while(x):
    return POSITIVE if "while" in x.description_lemma_spacy else ABSTAIN

@labeling_function()
def ADD_new_version(x):
    return POSITIVE if re.search(r"new.*version", x.description, flags=re.I) else NEGATIVE

@labeling_function()
def ADD_earlier_this_year(x):
    return POSITIVE if re.search(r"earlier.*this", x.description, flags=re.I) else ABSTAIN
```

```
@labeling_function()
def ADD_last_year_month(x):
    return POSITIVE if ((re.search(r"last.*year", x.description, flags=re.I)) or (re.search(r"last.*month", x.description, flags=re.I))) else ABSTAIN


@labeling_function()
def ADD_since_our_last(x):
    return POSITIVE if re.search(r"since.*our.*last", x.description, flags=re.I) else NEGATIVE


@labeling_function()
def ADD_has_been(x):
    return POSITIVE if re.search(r"has.*been", x.description, flags=re.I) else NEGATIVE


@labeling_function()
def ADD_new_versions(x):
    return POSITIVE if re.search(r"new.*versions", x.description, flags=re.I) else NEGATIVE


@labeling_function()
def ADD_dating_back(x):
    return POSITIVE if re.search(r"dating.*back", x.description, flags=re.I) else ABSTAIN
```

Label Attack with Espionage Intentions: Attack which aim to spy specific persons or organisations.

```
#KEYWORDS
@labeling_function()
def NOUN_data(x):
    return POSITIVE if re.search(r"data.*", x.description, flags=re.I) else NEGATIVE


@labeling_function()
def NOUN_spy(x):
    return POSITIVE if re.search(r"spy.*", x.description, flags=re.I) else NEGATIVE


@labeling_function()
def NOUN_cyberespionage(x):
    return POSITIVE if re.search(r"cyberespionage.*", x.description, flags=re.I) else NEGATIVE
```

```
@labeling_function()
def NOUN_espionage(x):
    return POSITIVE if re.search(r"espionage.*", x.description, flags=re.I) else NEGATIVE


@labeling_function()
def PROPN_breach(x):
    return POSITIVE if re.search(r"breach.*", x.description, flags=re.I) else NEGATIVE
```

Appendix N. Evaluation Analysis for each label and LF

Label Targeted: If the description refers to an attack or an activity that targeted a specific entity or group of people.

	j	Polarity	Coverage	Overlaps	Conflicts	Correct	Incorrect	Emp. Acc.
VERB_stolen	15	[1]	0.023474	0.023474	0.023474	4	1	0.800000
NOUN_target	8	[1]	0.384977	0.384977	0.384977	63	19	0.768293
SPACY_GPE	20	[0, 1]	1.000000	1.000000	0.713615	153	60	0.718310
VERB_targeting	17	[0, 1]	1.000000	1.000000	0.713615	146	67	0.685446
VERB_steal	14	[1]	0.075117	0.075117	0.075117	10	6	0.625000
NOUN_government	9	[0, 1]	1.000000	1.000000	0.713615	131	82	0.615023
VERB_targets	16	[0, 1]	1.000000	1.000000	0.713615	130	83	0.610329
NOUN_defense	3	[0, 1]	1.000000	1.000000	0.713615	127	86	0.596244
NOUN_countries	5	[0, 1]	1.000000	1.000000	0.713615	126	87	0.591549
ADP_against	2	[0, 1]	1.000000	1.000000	0.713615	125	88	0.586854
NOUN_company	4	[0, 1]	1.000000	1.000000	0.713615	124	89	0.582160
SPACY_LANGUAGE	21	[0, 1]	1.000000	1.000000	0.713615	122	91	0.572770
PROPN_bank	10	[0, 1]	1.000000	1.000000	0.713615	121	92	0.568075
ADJ_military	0	[0, 1]	1.000000	1.000000	0.713615	120	93	0.563380
PROPN_breach	11	[0, 1]	1.000000	1.000000	0.713615	120	93	0.563380
VERB_aim	18	[0, 1]	1.000000	1.000000	0.713615	118	95	0.553991
VERB_aimed	19	[0, 1]	1.000000	1.000000	0.713615	118	95	0.553991
ADJ_firm	1	[0, 1]	1.000000	1.000000	0.713615	117	96	0.549296
NOUN_politics	7	[0, 1]	1.000000	1.000000	0.713615	117	96	0.549296
NOUN_industry	6	[0, 1]	1.000000	1.000000	0.713615	116	97	0.544601
PROPN_focus	12	[1]	0.061033	0.061033	0.061033	7	6	0.538462
VERB_focusing	13	[1]	0.009390	0.009390	0.009390	1	1	0.500000

Label Instance refers to previous attack: If the malicious activity is based on previous malicious code, malware families or if the documents describe a similarity with other previous attacks.

(Part 1/2)

	j	Polarity	Coverage	Overlaps	Conflicts	Correct	Incorrect	Emp. Acc.
EXP_first_appeared	48	[1]	0.004695	0.004695	0.004695	1	0	1.000000
ORIG_developing	40	[1]	0.004695	0.004695	0.004695	1	0	1.000000
FIX_until_now	36	[1]	0.009390	0.009390	0.009390	2	0	1.000000
FIX_source_code	35	[1]	0.014085	0.014085	0.014085	3	0	1.000000
FIX_was_active	34	[1]	0.004695	0.004695	0.004695	1	0	1.000000
FIX_first_observed	33	[1]	0.009390	0.009390	0.009390	2	0	1.000000
FIX_dating_back	32	[1]	0.014085	0.014085	0.014085	3	0	1.000000
ADV_first	16	[1]	0.103286	0.103286	0.103286	18	4	0.818182
FIX_was_first	31	[1]	0.023474	0.023474	0.023474	4	1	0.800000
ORIG_updated	41	[1]	0.018779	0.018779	0.018779	3	1	0.750000
VERB_wave	26	[1]	0.018779	0.018779	0.018779	3	1	0.750000
ADV_previously	14	[0, 1]	1.000000	1.000000	1.000000	147	66	0.690141
DET_another	23	[1]	0.075117	0.075117	0.075117	11	5	0.687500
EXP_based	49	[0, 1]	1.000000	1.000000	1.000000	144	69	0.676056
VERB_tie	28	[1]	0.014085	0.014085	0.014085	2	1	0.666667
ADV_ago	17	[1]	0.042254	0.042254	0.042254	6	3	0.666667
SUM_word_count	0	[0, 1]	1.000000	1.000000	1.000000	142	71	0.666667
ADJ_previous	8	[0, 1]	1.000000	1.000000	1.000000	142	71	0.666667
ADJ_variant	12	[0, 1]	1.000000	1.000000	1.000000	142	71	0.666667
EXP_family	43	[0, 1]	1.000000	1.000000	1.000000	141	72	0.661972
ADJ_ongoing	11	[0, 1]	1.000000	1.000000	1.000000	139	74	0.652582
NOUN_reemergence	20	[0, 1]	1.000000	1.000000	1.000000	138	75	0.647887
THESAURUS_threat_agents	6	[0, 1]	1.000000	1.000000	1.000000	138	75	0.647887

(continues on next page)

(Part 2/2)

ADJ_similar	9	[0, 1]	1.000000	1.000000	1.000000	138	75	0.647887
ADJ_old	10	[0, 1]	1.000000	1.000000	1.000000	138	75	0.647887
PROPN_appeared	25	[0]	1.000000	1.000000	1.000000	137	76	0.643192
NOUN_predecessor	21	[0]	1.000000	1.000000	1.000000	137	76	0.643192
SUM_industries_ind	3	[0]	0.197183	0.197183	0.197183	27	15	0.642857
SUM_targeted_countries	1	[0]	0.258216	0.258216	0.258216	35	20	0.636364
ADV_back	19	[1]	0.051643	0.051643	0.051643	7	4	0.636364
EXP_modified	46	[0, 1]	1.000000	1.000000	1.000000	135	78	0.633803
VERB_update	30	[0, 1]	1.000000	1.000000	1.000000	135	78	0.633803
EXP_groups	42	[0, 1]	1.000000	1.000000	1.000000	133	80	0.624413
CUSTOM_year	5	[1]	0.295775	0.295775	0.295775	39	24	0.619048
ADV_originally	15	[1]	0.023474	0.023474	0.023474	3	2	0.600000
ADJ_last	13	[1]	0.056338	0.056338	0.056338	7	5	0.583333
VERB_observe	27	[1]	0.159624	0.159624	0.159624	19	15	0.558824
ADJ_recent	7	[1]	0.084507	0.084507	0.084507	10	8	0.555556
EXP_used	44	[1]	0.225352	0.225352	0.225352	25	23	0.520833
ORIG_observed	38	[1]	0.136150	0.136150	0.136150	15	14	0.517241
ORIG_tied	39	[1]	0.009390	0.009390	0.009390	1	1	0.500000
VERB_develop	29	[1]	0.037559	0.037559	0.037559	4	4	0.500000
ORIG_earlier	37	[1]	0.042254	0.042254	0.042254	4	5	0.444444
SUM_total_ref	4	[0, 1]	1.000000	1.000000	1.000000	83	130	0.389671
ADV_early	18	[1]	0.089202	0.089202	0.089202	7	12	0.368421
EXP_share	45	[1]	0.051643	0.051643	0.051643	4	7	0.363636
SUM_tags_ind	2	[1]	0.309859	0.309859	0.309859	23	43	0.348485
NOUN_borrowing	22	[]	0.000000	0.000000	0.000000	0	0	0.000000
PROPN_successor	24	[]	0.000000	0.000000	0.000000	0	0	0.000000
EXP_subsequent	47	[1]	0.014085	0.014085	0.014085	0	3	0.000000

Labelling Functions of the second round for label "Refers to a previous attack":

	j	Polarity	Coverage	Overlaps	Conflicts	Correct	Incorrect	Emp. Acc.
ADD_dating_back	8	[1]	0.018779	0.018779	0.018779	4	0	1.000000
SCONJ_since	0	[0, 1]	1.000000	1.000000	0.441315	160	53	0.751174
ADD_new_version	2	[0, 1]	1.000000	1.000000	0.441315	142	71	0.666667
ADD_earlier_this_year	3	[1]	0.028169	0.028169	0.028169	4	2	0.666667
ADD_last_year_month	4	[1]	0.028169	0.028169	0.028169	4	2	0.666667
ADD_new_versions	7	[0, 1]	1.000000	1.000000	0.441315	141	72	0.661972
ADD_since_our_last	5	[0, 1]	1.000000	1.000000	0.441315	138	75	0.647887
ADD_has_been	6	[0, 1]	1.000000	1.000000	0.441315	134	79	0.629108
SCONJ_while	1	[1]	0.093897	0.093897	0.093897	9	11	0.450000

Label Attack with Espionage Intentions: Attack which aims to spy specific persons or organisations.

	j	Polarity	Coverage	Overlaps	Conflicts	Correct	Incorrect	Emp. Acc.
NOUN_espionage	3	[0, 1]	1.0	1.0	0.234742	207	6	0.971831
NOUN_cyberespionage	2	[0, 1]	1.0	1.0	0.234742	199	14	0.934272
NOUN_spy	1	[0, 1]	1.0	1.0	0.234742	191	22	0.896714
PROPN_breach	4	[0, 1]	1.0	1.0	0.234742	190	23	0.892019
NOUN_data	0	[0, 1]	1.0	1.0	0.234742	176	37	0.826291

Appendix O. An example of how different LFs label an instance as positive for each label

Label Targeted: If the description refers to an attack or an activity that targeted a specific entity or group of people.

Description:

'The adversary group known as Group123, APT37, and ScarCruft has been observed conducting spear phishing attacks with a fake resume against South Korean ***targets****. Resume includes link which drops second file, which then conducts network communication with various URLs.'*

Triggered LFs (weight is provided by the generative model of Snorkel):

Labelling Function	Produced Label	Weight
NOUN_target	1	1
SPACY_GPE	1	0.712

Label Instance refers to previous attack: If the malicious activity is based on previous malicious code, malware families or if the documents describe a similarity with other previous attacks.

Description:

'On August 1, 2018, the US Department of Justice announced that it had arrested several individuals suspected of having ties to the FIN7 cybercrime rig. FIN7 operations are linked to numerous intrusion attempts having targeted hundreds of companies since at least as early as 2015. Interestingly, this threat actor created fake companies in order to hire remote pentesters, developers and interpreters to participate in their malicious business. The main goal behind its malicious activities was to steal financial assets from companies, such as debit cards, or get access to

financial data or computers of finance department employees in order to conduct wire transfers to offshore accounts.'

Triggered LFs (weight is provided by the generative model of Snorkel):

Labelling Function	Produced Label	Weight
CUSTOM_year	1	0.927
ADV_early	1	0.856
VERB_tie	1	0.795
SUM_word_count	1	0.712
SCONJ_since	1	0.665
SUM_total_ref	1	0.49

Label Attack with Espionage Intentions: Attack which aim to spy specific persons or organisations.

Description:

'TrendMicro uncovered a cyber espionage campaign targeting Middle Eastern countries. We named this campaign "Bouncing Golf" based on the malware's code in the package named "golf." The malware involved, which Trend Micro detects as AndroidOS_GolfSpy.HRX, is notable for its wide range of cyber espionage capabilities. Malicious codes are embedded in apps that the operators repackaged from legitimate applications. Monitoring the command and control (C&C) servers used by Bouncing Golf, we've so far observed more than 660 Android devices infected with GolfSpy. Much of the information being stolen appear to be military-related.'

Triggered LFs (weight is provided by the generative model of Snorkel):

Labelling Function	Produced Label	Weight
NOUN_espionage	1	0.624
NOUN_spy	1	0.538

Appendix P. Final selected LFs for labelling the training set of pulse's descriptions

Label Targeted:

VERB_steal, VERB_focusing, NOUN_countries, NOUN_politics, VERB_aimed, NOUN_target, NOUN_government, ADJ_military, PROPN_bank, NOUN_company, VERB_targeting, SPACY_GPE

Label Instance refers to a previous attack:

ADD_new_version, ADV_early, SCONJ_while, VERB_tie, EXP_based, EXP_first_appeared, NOUN_borrowing, ORIG_observed, ADJ_ongoing, EXP_modified, SUM_industries_ind, ADD_earlier_this_year, ADJ_recent, PROPN_successor, ADJ_old, SUM_targeted_countries, ORIG_tied, EXP_family, FIX_until_now, ADV_originally, SCONJ_since, EXP_groups, FIX_dating_back, SUM_total_ref, DET_another, FIX_source_code, ADD_last_year_month, ADV_back, EXP_subsequent, EXP_used, FIX_first_observed, FIX_was_active, ORIG_earlier, ADD_since_our_last, ADJ_last, ADJ_variant, ADV_first, SUM_word_count, EXP_share, VERB_observe, ADD_has_been, VERB_develop, CUSTOM_year, ADJ_previous, ADD_dating_back, NOUN_predecessor, PROPN_appeared, ORIG_developing, ADV_previously

Label Attack with Espionage Intentions:

NOUN_data, NOUN_cyberespionage, NOUN_espionage

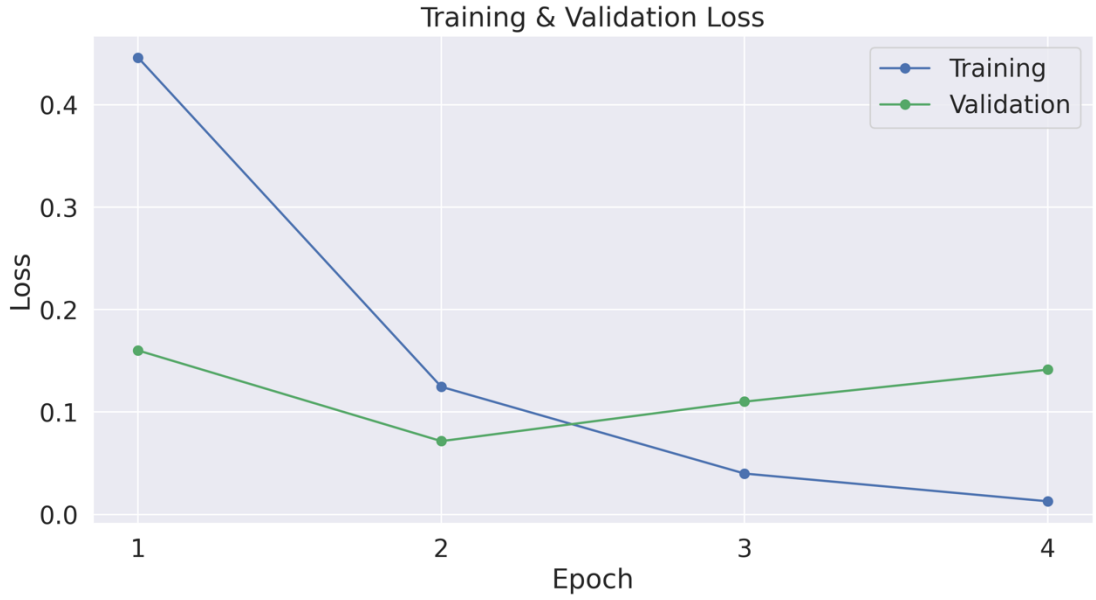
Appendix Q. NLP models for pulse's descriptions

For all models we used:

- Training set – Validation set (90%-10%). Validation set is part of the original training subset, and it is used only during the training of the model.
- BertTokenizer/ RobertaTokenizer from Transformers package to tokenize our documents
- BertForSequenceClassification / RoBERTaForSequenceClassification pretrained models from Transformers package
- Adam optimizer with a fixed learning rate of 5e-5 and epsilon value 1e-8
- Batch size equal to 32

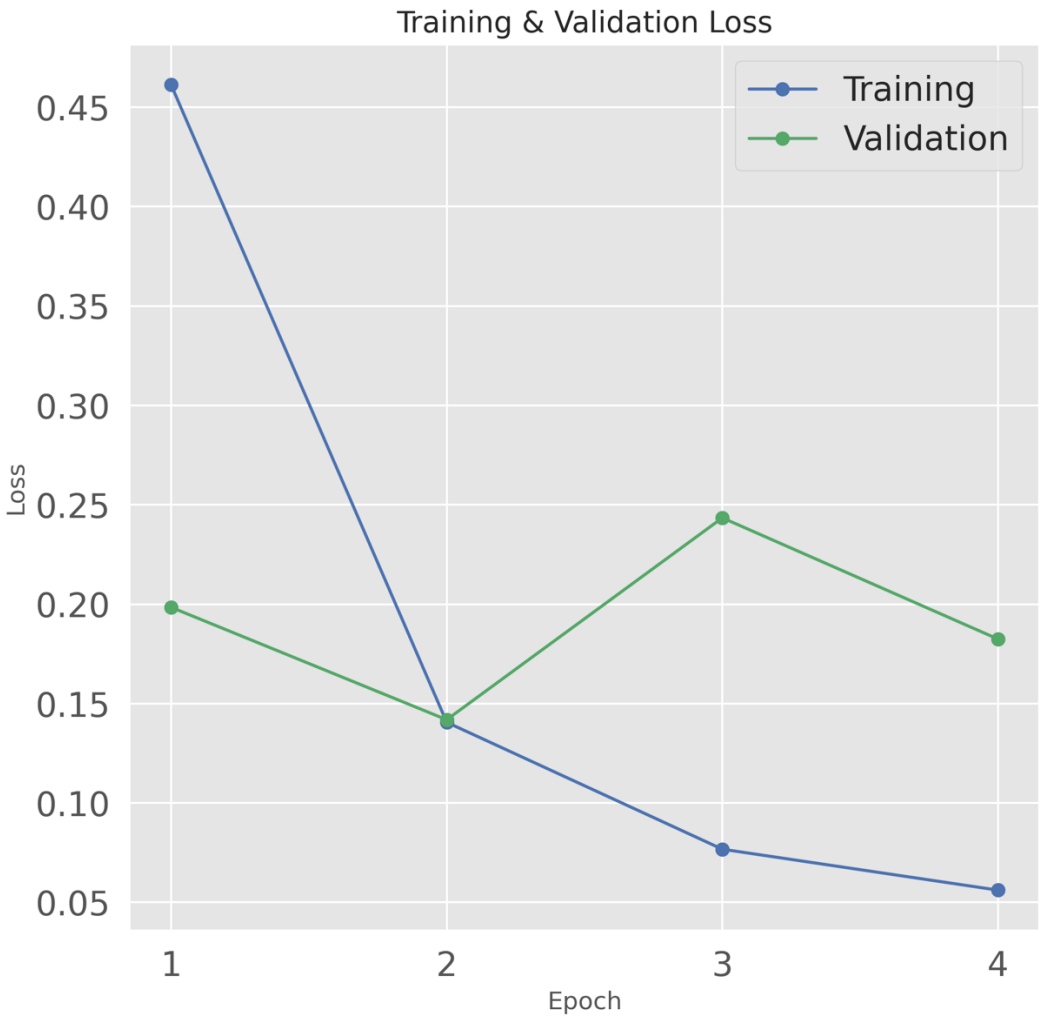
Label "targeted"

BERT train-validation loss for a different number of epochs:



Two epochs yielded the best results in the test set.

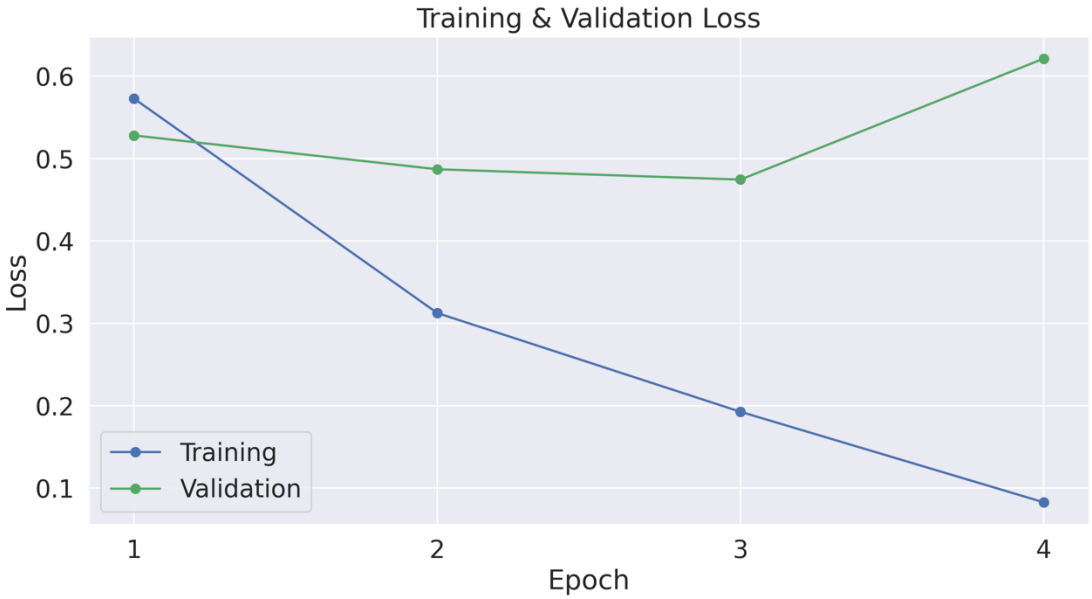
RoBERTa train-validation loss for a different number of epochs:



Two epochs yielded the best results in the test set.

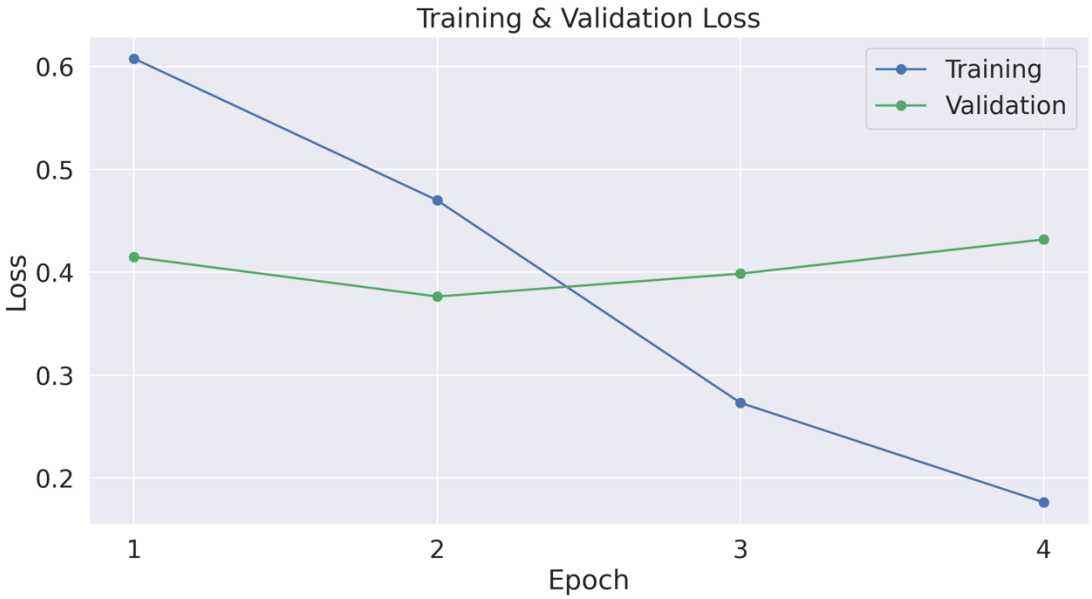
Label "refers to a previous attack"

BERT train-validation loss for a different number of epochs:



One epoch yielded the best results in the test set.

RoBERTa train-validation loss for a different number of epochs:



Three epochs yielded the best results in the test set.

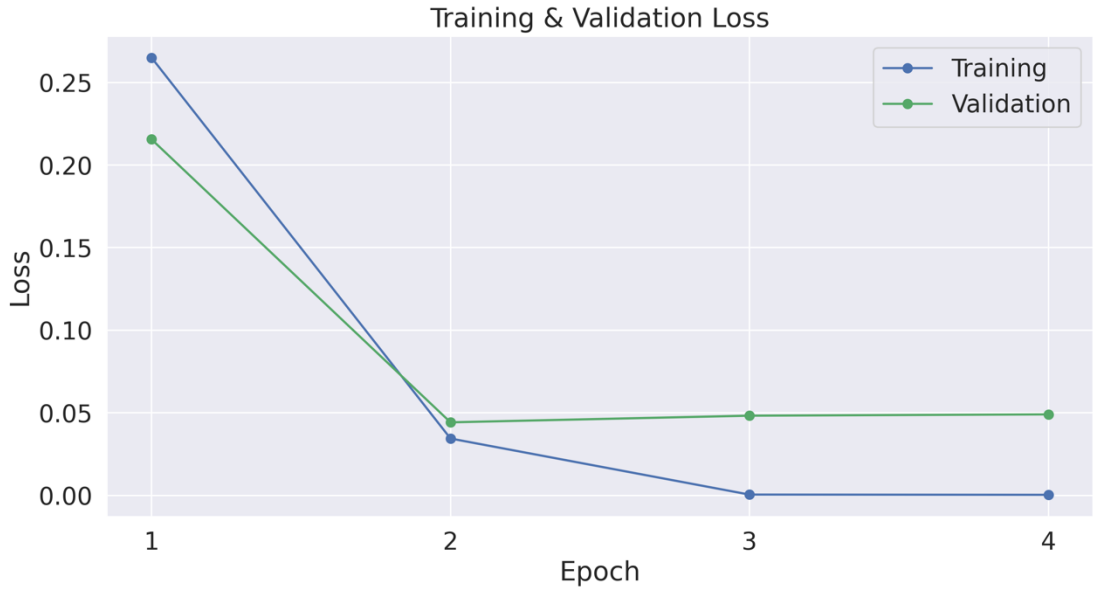
Label “espionage”

Original training subset

BERT train-validation loss for a different number of epochs:

Four epochs yielded the best results in the test set.

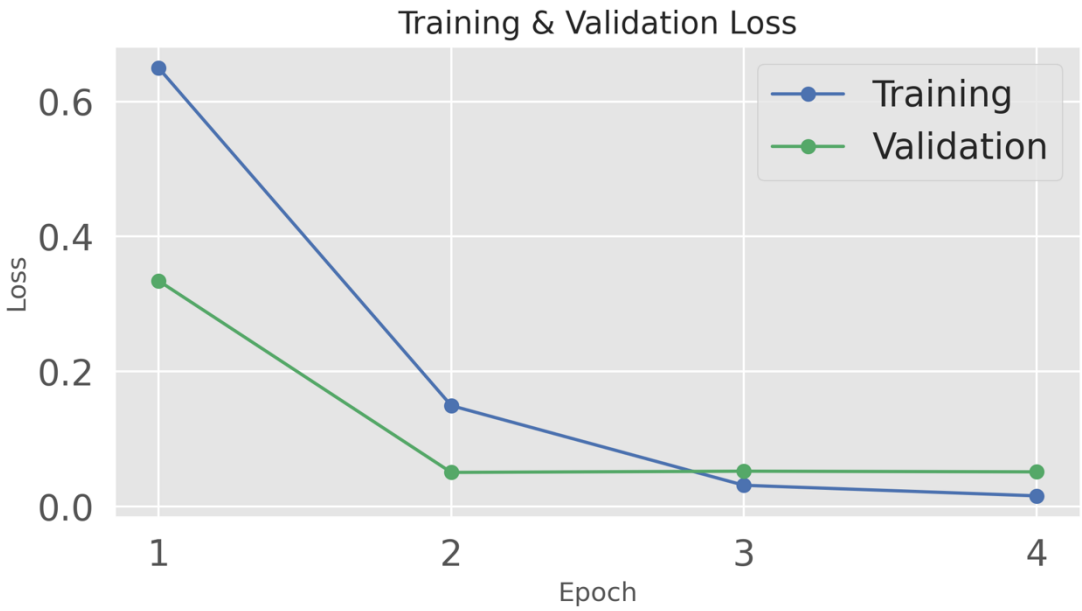
RoBERTa train-validation loss for a different number of epochs:



Two epochs yielded the best results in the test set.

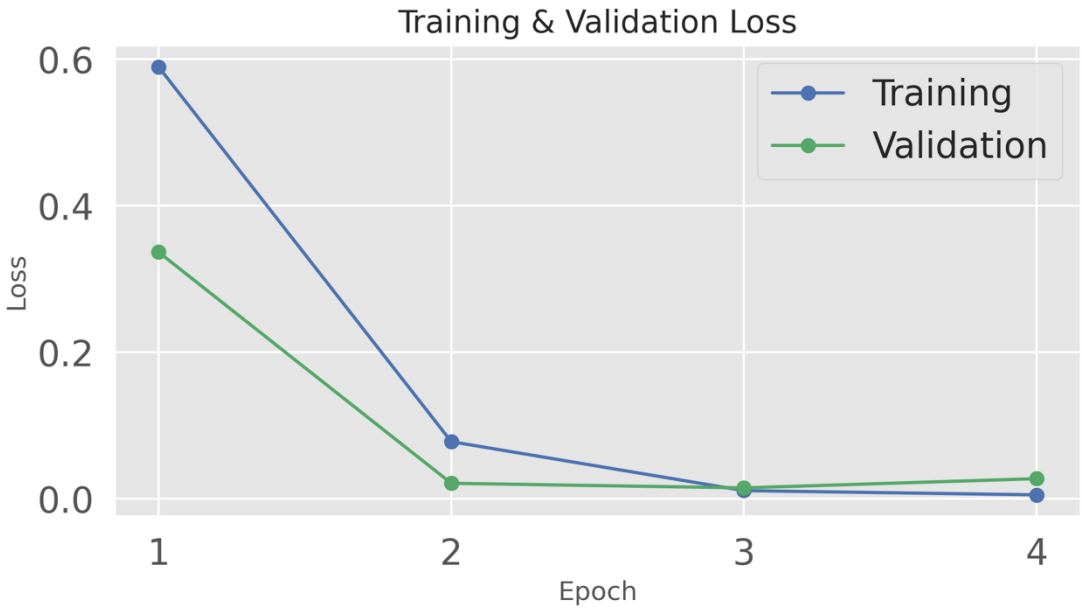
Balanced training subset

BERT train-validation loss for a different number of epochs:



Two epochs yielded the best results in the test set.

RoBERTa train-validation loss for a different number of epochs:



Four epochs yielded the best results in the test set.

Appendix R. Data Augmentation with Transformation Functions for short descriptions

First, we examined how imbalanced is our training set. From the 1569 records, 1316 were negative and 253 positives (83%, 16% respectively). From the 1316 negative records, we randomly removed half of them (under-sampled), ending up with 784 observations. For the 253 positives, we augmented them with snorkel 2 additional augmented examples per instance. In this way, we ended up with 253x3 = 759 observations. The final augmented training set of 1543 records were almost balanced.

The augmentation was done with five transformation functions as they are proposed from Snorkel[9]. The transformation functions change country names within the text string, swap adjectives, or replace verbs, nouns and adjectives with synonyms. Below we provide an example of how the transformation functions could modify one of our instances.

[9] https://www.snorkel.org/use-cases/02-spam-data-augmentation-tutorial

	TF Name	Original Text	Transformed Text
0	change_country	In May 2019, ESET researchers observed a spike in ESET telemetry data regarding malware targeting France. After further investigations, they identified malware that distributes various types of spam. One of them is leading to a survey that redirects to a dodgy smartphone promotion while the other is a sextortion campaign. The spam targets the users of Orange S.A., a French ISP.	In May 2019, ESET researchers observed a spike in ESET telemetry data regarding malware targeting Panama. After further investigations, they identified malware that distributes various types of spam. One of them is leading to a survey that redirects to a dodgy smartphone promotion while the other is a sextortion campaign. The spam targets the users of Orange S.A., a French ISP.
1	swap_adjectives	In May 2019, ESET researchers observed a spike in ESET telemetry data regarding malware targeting France. After further investigations, they identified malware that distributes various types of spam. One of them is leading to a survey that redirects to a dodgy smartphone promotion while the other is a sextortion campaign. The spam targets the users of Orange S.A., a French ISP.	In May 2019, ESET researchers observed a spike in ESET telemetry data regarding malware targeting France. After further investigations, they identified malware that distributes other types of spam. One of them is leading to a survey that redirects to a dodgy smartphone promotion while the various is a sextortion campaign. The spam targets the users of Orange S.A., a French ISP.
2	replace_verb_with_synonym	In May 2019, ESET researchers observed a spike in ESET telemetry data regarding malware targeting France. After further investigations, they identified malware that distributes various types of spam. One of them is leading to a survey that redirects to a dodgy smartphone promotion while the other is a sextortion campaign. The spam targets the users of Orange S.A., a French ISP.	In May 2019, ESET researchers observed a spike in ESET telemetry data see malware targeting France. After further investigations, they identified malware that distributes various types of spam. One of them is leading to a survey that redirects to a dodgy smartphone promotion while the other is a sextortion campaign. The spam targets the users of Orange S.A., a French ISP.
3	replace_noun_with_synonym	In May 2019, ESET researchers observed a spike in ESET telemetry data regarding malware targeting France. After further investigations, they identified malware that distributes various types of spam. One of them is leading to a survey that redirects to a dodgy smartphone promotion while the other is a sextortion campaign. The spam targets the users of Orange S.A., a French ISP.	In May 2019, ESET researchers observed a spike in ESET telemetry data regarding malware targeting France. After further investigations, they identified malware that distributes various types of spam. One of them is leading to a survey that redirects to a dodgy smartphone promotion while the other is a sextortion political campaign . The spam targets the users of Orange S.A., a French ISP.
4	replace_adjective_with_synonym	A sophisticated hacking group with suspected ties to cybercrime gangs operating in Eastern Europe is now actively targeting and breaching prominent, brand name restaurants in the U.S.\n\nA recently disclosed data breach suffered by Mexican fast food restaurant Chipotle was carried out by hackers linked to a group known as FIN7 or Carbanak Group, CyberScoop has learned. In addition to Chipotle, the hackers appears to be targeting national restaurant franchises Baja Fresh and Ruby Tuesday, according to malware samples and other evidence CyberScoop obtained.	A sophisticated hacking group with suspected ties to cybercrime gangs operating in Eastern Europe is now actively targeting and breaching outstanding , brand name restaurants in the U.S.\n\nA recently disclosed data breach suffered by Mexican fast food restaurant Chipotle was carried out by hackers linked to a group known as FIN7 or Carbanak Group, CyberScoop has learned. In addition to Chipotle, the hackers appears to be targeting national restaurant franchises Baja Fresh and Ruby Tuesday, according to malware samples and other evidence CyberScoop obtained.

As we wanted two additional augmented instances per example, we randomly have chosen two transformation functions for each record.

Appendix S. Quality checks on the web scrapped content

In D-subsets.ipynb we developed the function substring_after(s) which check different conditions and accordingly performs some actions in the scrapped text. For example, if the extracted content contained expressions such as "Page not found" or "Skip to main content", then these documents were discarded. In addition, in the same function, we removed content which was coming from the headers of the websites.

Appendix T. Final selected LFs for labelling the training set of web pages

Label Targeted:

NOUN_politics, VERB_targeting, NOUN_defense, NOUN_company, VERB_steal, PROPN_focus, PROPN_breach, NOUN_government, NOUN_countries, PROPN_bank, VERB_stolen, ADJ_military, VERB_targets, SPACY_GPE

Label Instance refers to a previous attack:

SUM_word_count, ORIG_tied, SUM_tags_ind, CUSTOM_year, SUM_total_ref, ORIG_observed, THESAURUS_threat_agents, EXP_modified, ADJ_previous, ORIG_updated, EXP_groups, ADJ_ongoing, ADJ_similar, ADJ_last, NOUN_predecessor, ADV_first, ADV_originally, ADV_ago, VERB_observe, FIX_until_now, NOUN_reemergence, ADJ_variant, NOUN_borrowing, DET_another, PROPN_successor, EXP_used, VERB_wave, ADV_early, EXP_first_appeared, EXP_subsequent, VERB_update, FIX_was_first, FIX_dating_back, FIX_first_observed, FIX_was_active, ADD_dating_back, ADV_back, ORIG_earlier, PROPN_appeared, SUM_targeted_countries, ADJ_old, VERB_tie, ADJ_recent, EXP_family, ORIG_developing, FIX_source_code, VERB_develop, ADV_previously, EXP_share, EXP_based, SCONJ_since

Label Attack with Espionage Intentions:

NOUN_spy, NOUN_cyberespionage, NOUN_espionage

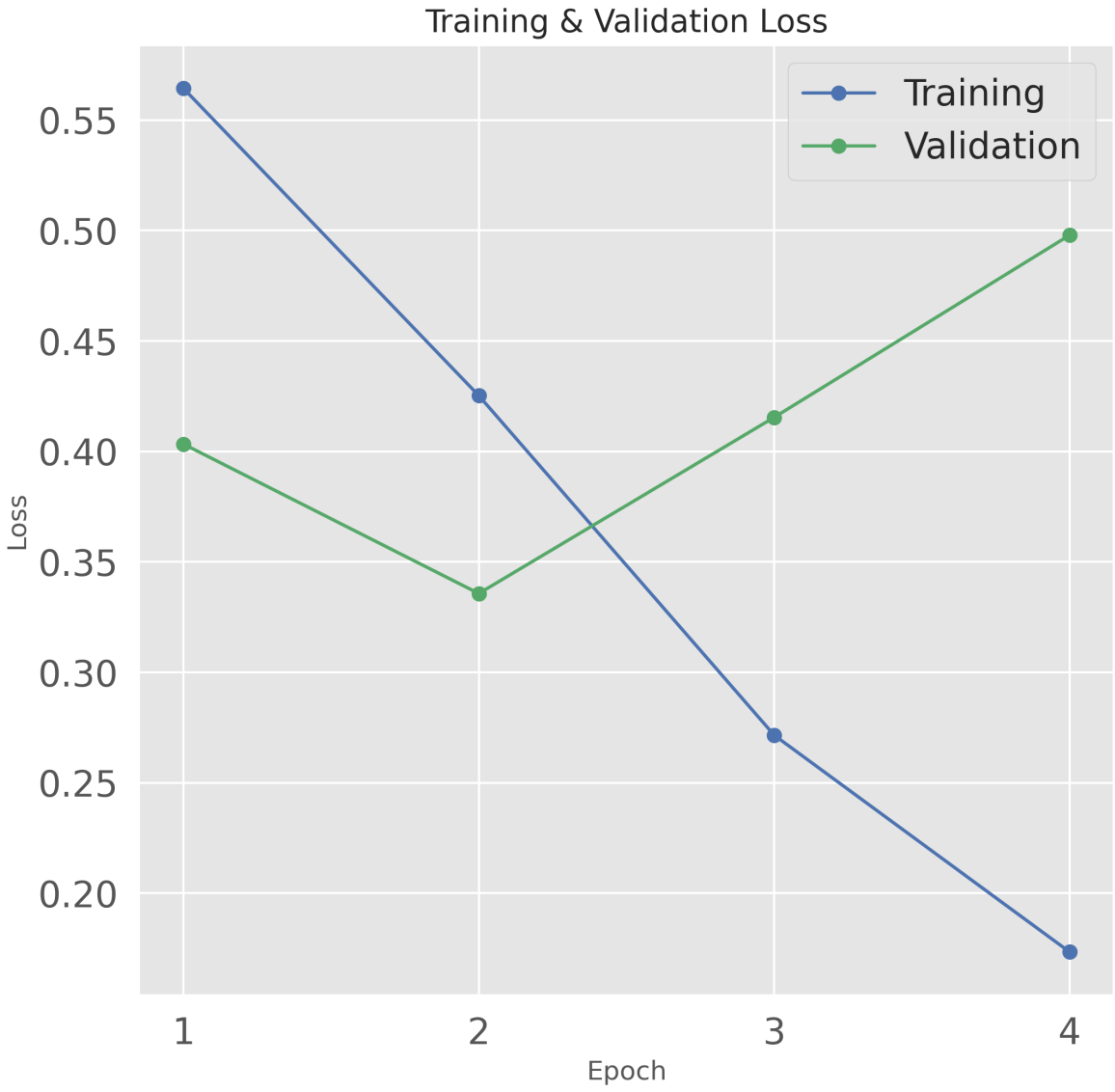
Appendix U. NLP models for web pages

For all models we used:

- The number of tokens was fixed to 300 (as the sliding windows were calculated according to this number)
- Training set – Validation set (90%-10%). Validation set is part of the original training subset, and it is used only during the training of the model
- BertTokenizer/ RobertaTokenizer from Transformers package to tokenize our documents
- BertForSequenceClassification / RoBERTaForSequenceClassification pretrained models from Transformers package
- Adam optimizer with a fixed learning rate of 5e-5 and epsilon value 1e-8
- Batch size equal to 32

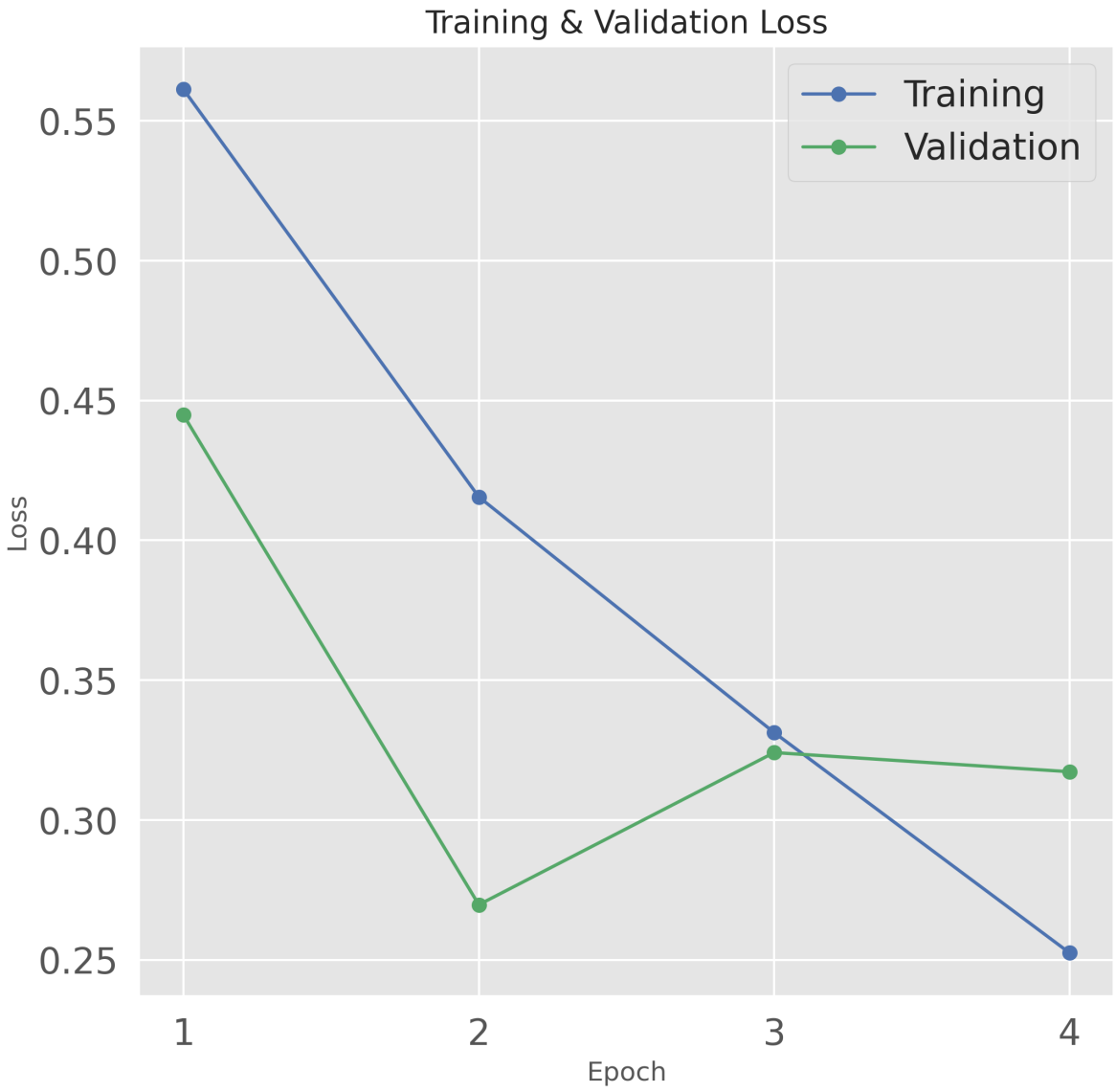
Label “targeted”

BERT train-validation loss for a different number of epochs:



Two epochs yielded the best results in the test set.

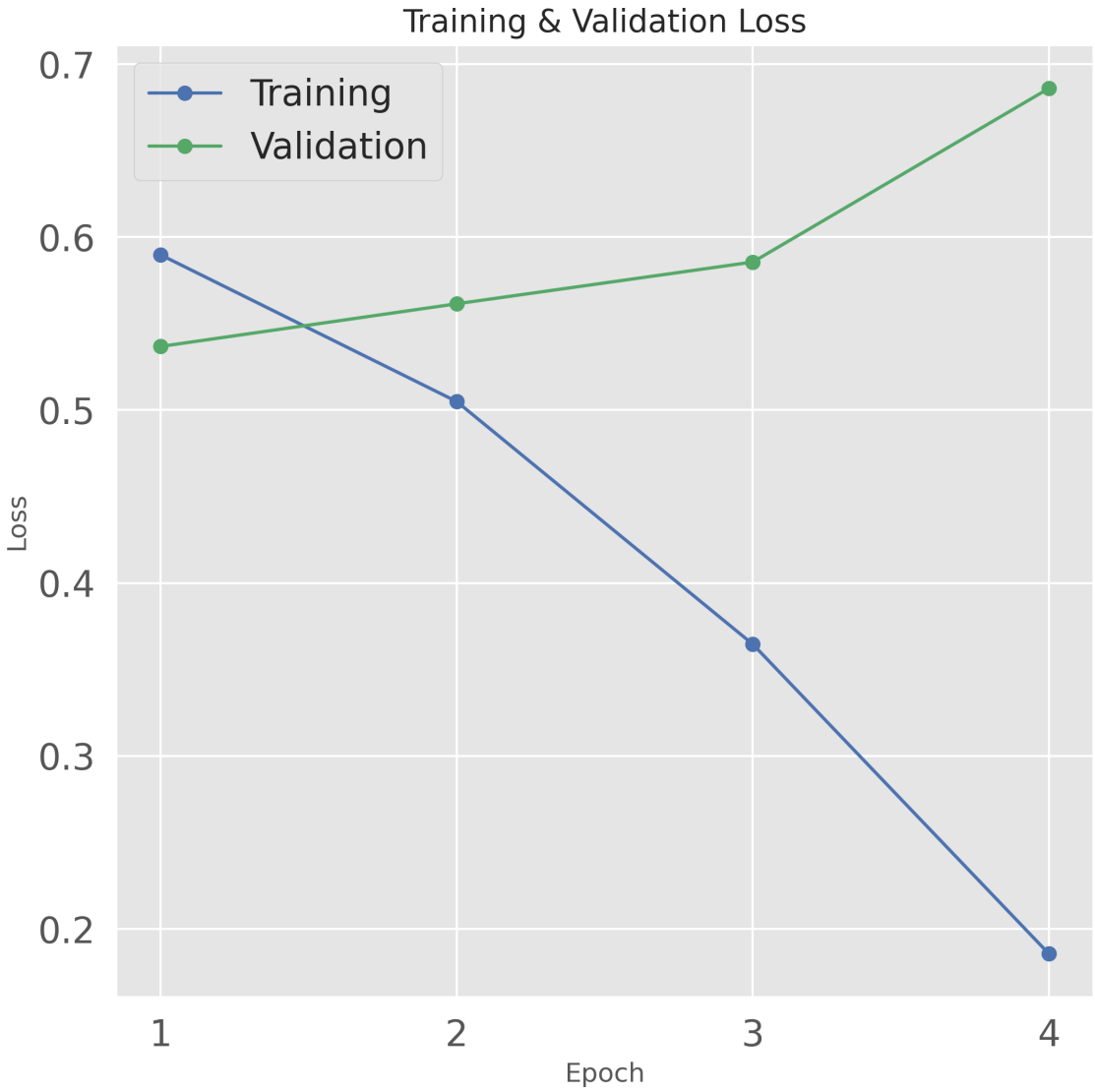
RoBERTa train-validation loss for a different number of epochs:



Two epochs yielded the best results in the test set.

Label “refers to previous attack”

BERT train-validation loss for a different number of epochs:



One epoch yielded the best results in the test set.

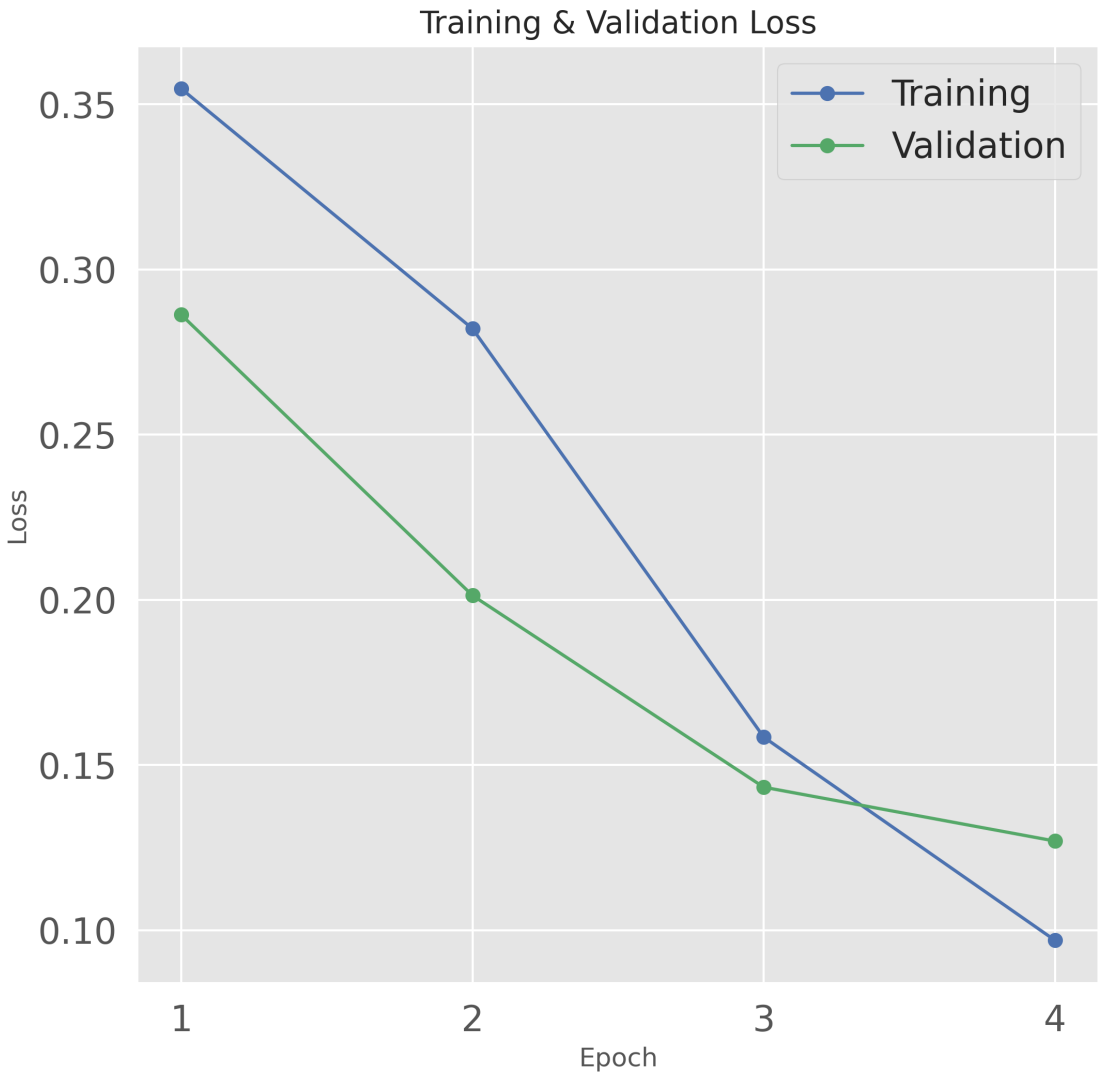
RoBERTa train-validation loss for a different number of epochs:

Four epochs yielded the best results in the test set.

Label “espionage”

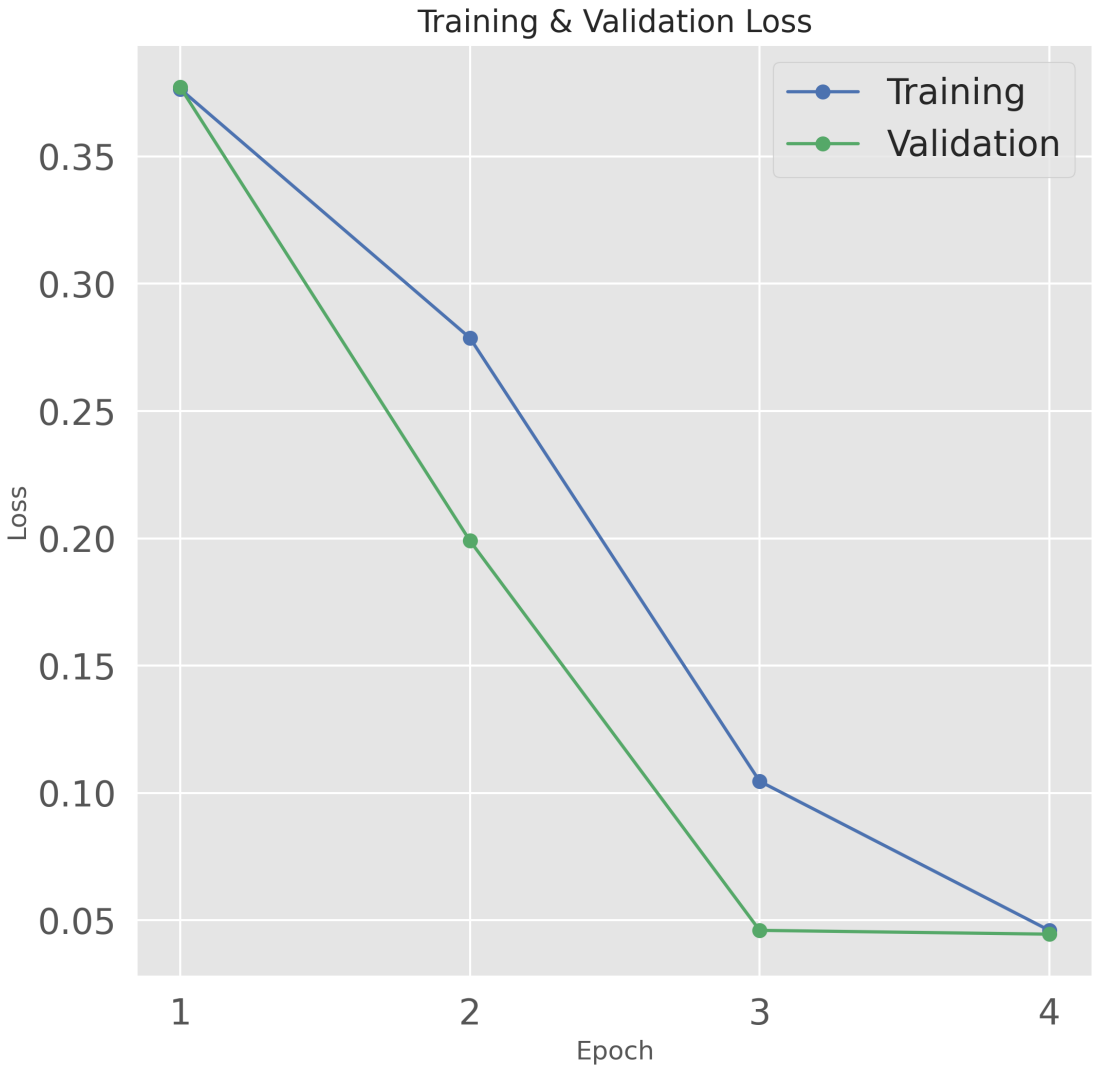
Original training subset

BERT train-validation loss for a different number of epochs:



Four epochs yielded the best results in the test set.

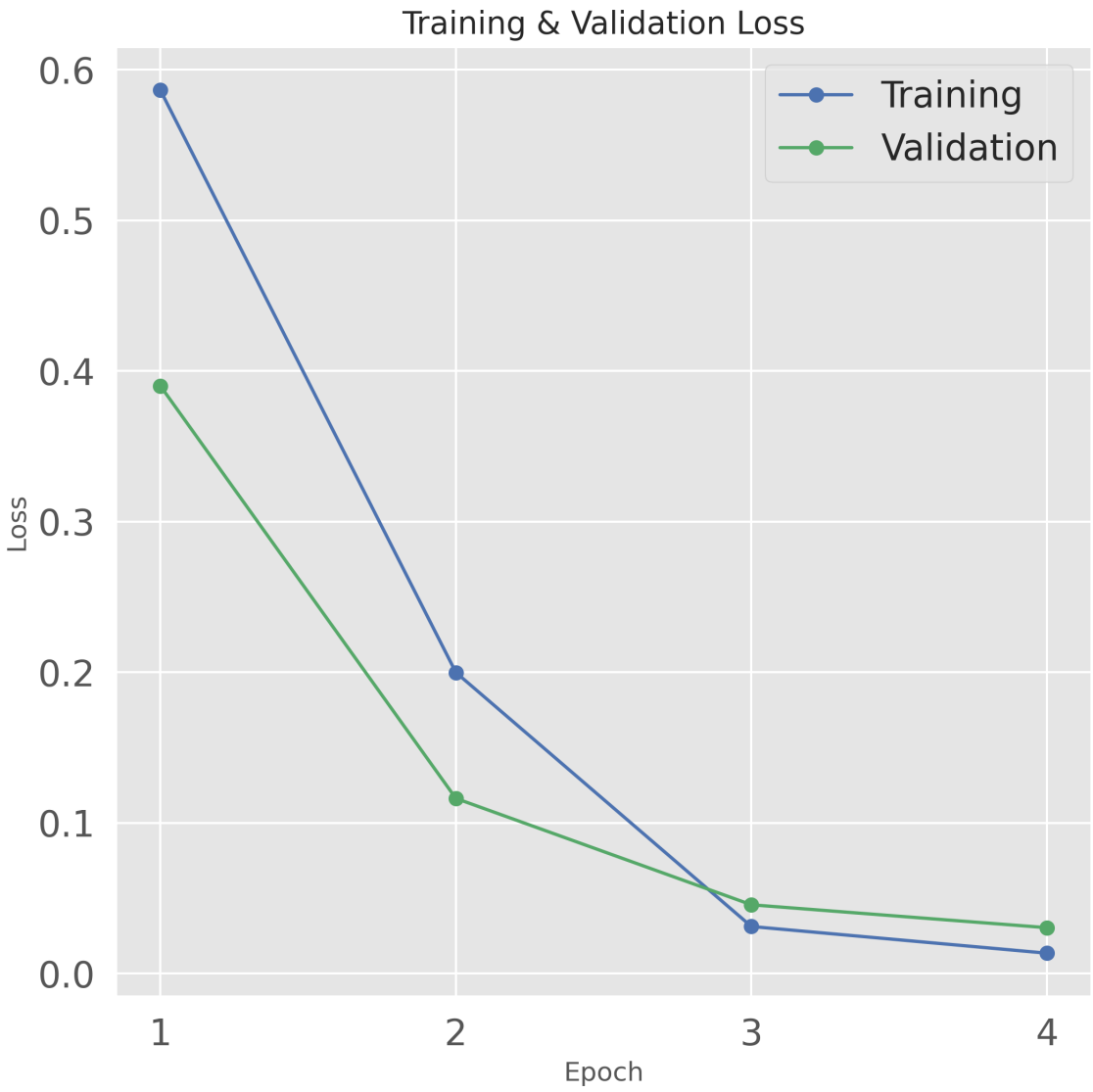
RoBERTa train-validation loss for a different number of epochs:



Four epochs yielded the best results in the test set.

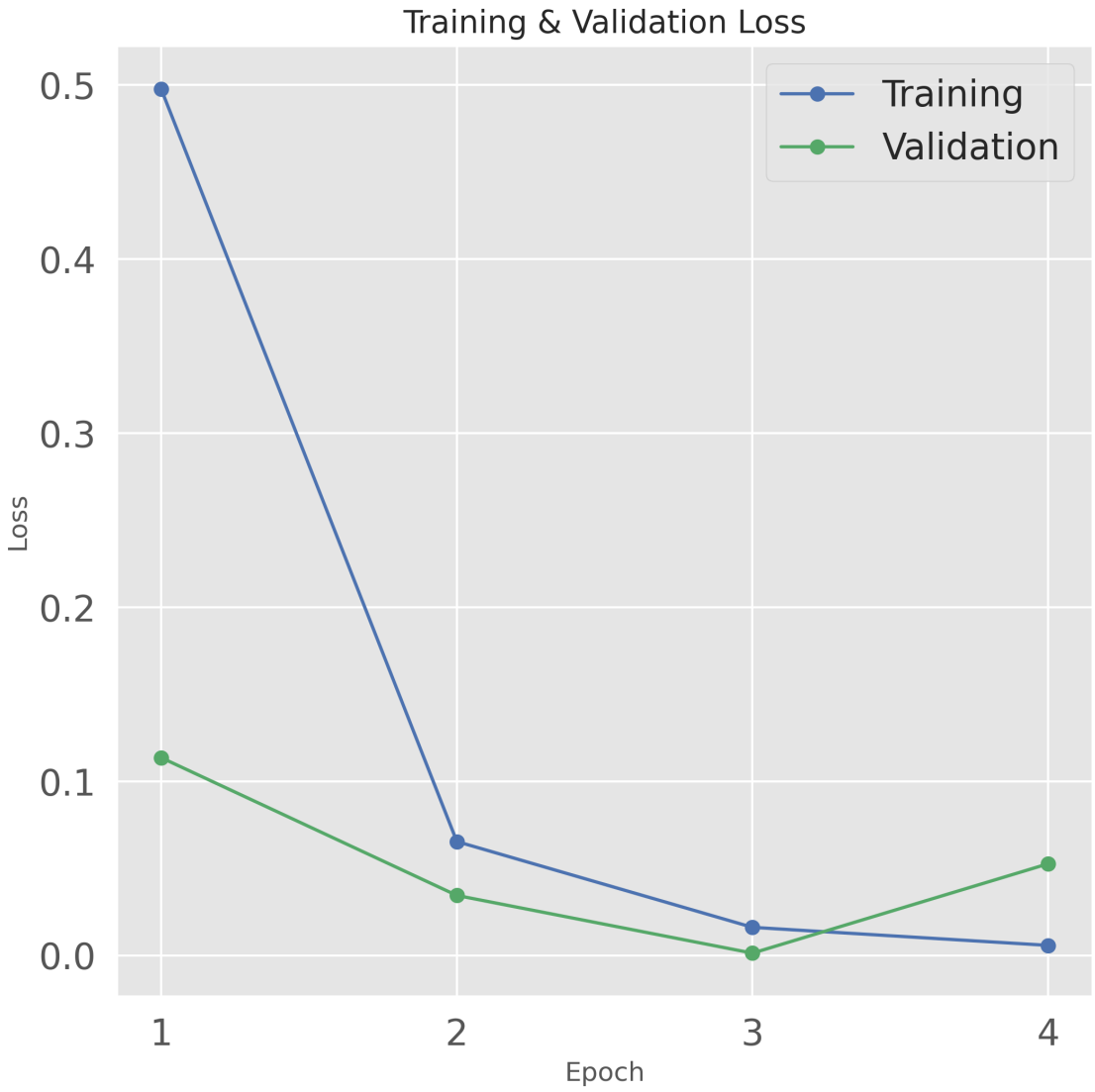
Balanced training subset

BERT train-validation loss for a different number of epochs:



Four epochs yielded the best results in the test set.

RoBERTa train-validation loss for a different number of epochs:



Three epochs yielded the best results in the test set.