Hochschule für angewandte Wissenschaften
München
Fakultät für Informatik und Mathematik

Master's thesis
for obtaining the academic degree of
Master of Science
in course of studies IT-Security

Relation Extraction and Knowledge Graph Generation on MISP Event Reports

---

Author:	Leon Lukas
Matriculation number:	58771316
Submission Date:	13. April 2023
Supervisor:	Prof. Dr. Thomas Schreck
Advisor:	Thomas Geras

I confirm that this master’s thesis is my own work and I have documented all sources and material used.

München, 13. April 2023 Leon Lukas

Acknowledgments

I would like to express my sincere gratitude to my thesis supervisor, Thomas Schreck, for his valuable guidance, and insightful feedback throughout the research process.

I would also like to extend my appreciation to my advisor, Thomas Geras, for his valuable suggestions and constructive feedback that helped to shape the direction of my research.

Furthermore, I would like to thank my fellow student, Alexander Schwankner, for his efforts in the creation of the joint dataset. His collaboration and contribution have been crucial to the success of this project.

Finally, I would like to express my gratitude to all the people who have supported me throughout my academic journey. Your encouragement and support have been invaluable to me.

Thank you all for your help and support.

Abstract

The rapid growth of cyber-attacks requires organizations to be aware of procedures currently employed by malicious actors. This awareness can be gained through the sharing of cyber threat intelligence (CTI). As manually examining all CTI information is a time-intensive task for security analysts, this work investigates the possibilities of extracting relevant relations present within CTI-texts by using deep learning models to generate knowledge graphs for easier understanding. Because no fitting dataset for relation extraction in the CTI domain was publicly available, a dataset was created as part of this work. Using this dataset as well as the New-York-Times relation extraction dataset we fine-tuned Googles T5 language model to extract relations from CTI-texts. This combination of model, data, and framing relation extraction as a sequence-to-sequence task proved to be most effective for extracting relations from CTI-texts. To improve the model's performance on the highly nuanced texts present in CTI reports, a pre-processing pipeline, which replaces special words before passing them to the deep learning model, was developed. Results show that the use of the pre-processing pipeline was highly effective, increasing the model's performance by 27%. Using the system, sensible relations can be extracted from reports. However, due to limitations in the dataset, the developed model cannot achieve human-level performance at generating knowledge graphs from event reports. Contributions of this work to the field of natural language processing in the CTI-domain include: a publicly available dataset for training named entity recognition and relation extraction models, a model which can extract relations from CTI-texts with an F1-Score of 0.38 and a pre-processing pipeline that introduces the concept of using wordlists of important CTI-terms such as names of malware or threat actors.

Contents

1 Introduction **1**
- 1.1 Motivation . . . 1
- 1.2 Problem Statement . . . 1
- 1.3 Research Questions . . . 2
- 1.4 Structure . . . 2

2 Theoretical foundations **5**
- 2.1 Basics of Deep Learning . . . 5
- 2.2 Theory and History on Language Models . . . 6
- 2.3 Closed source language models . . . 7
- 2.4 Training NLP models using Huggingface . . . 7
- 2.5 T5 . . . 9
- 2.6 Relation Extraction . . . 10
  - 2.6.1 Approaches to Relation Extraction . . . 11
  - 2.6.2 Evaluation . . . 11
- 2.7 Relation Extraction in Literature . . . 13
  - 2.7.1 Utilization of Distant Supervision and Customized Loss Functions 14
  - 2.7.2 The Impact of Entity and Contextual Information . . . 14
  - 2.7.3 Leveraging Pretraining on Wikipedia Data . . . 15
  - 2.7.4 Work with similar approach . . . 16
  - 2.7.5 Zero-Shot Learning Approaches in Information Extraction . . . 16
- 2.8 Preprocessing of CTI-Texts . . . 17

3 Related Work **19**

4 Methodology **21**
- 4.1 Fundamental Methodological Choices . . . 21
- 4.2 Literature review . . . 21
- 4.3 Dataset . . . 21
- 4.4 Deep-Learning Experiments . . . 22
- 4.5 System Development . . . 24
- 4.6 Concluding Summary . . . 24

5 Datasets **27**
- 5.1 Relation Extraction Datasets . . . 27
  - 5.1.1 New York Times . . . 27
  - 5.1.2 CONLL04 . . . 27
  - 5.1.3 DocRED . . . 28
  - 5.1.4 Comparison of RE-Datasets . . . 28
- 5.2 Public CTI Datasets . . . 29

5.3 Creation of CTI Dataset . . . 30
5.3.1 Creation Process . . . 30
5.4 Ontology of the CTI-Dataset . . . 31
5.5 Dataset properties and versions . . . 31
5.5.1 Changes to the dataset split . . . 31
5.5.2 Changes to Entity Types . . . 33
5.5.3 Changes to Relations . . . 33
5.5.4 Properties of the Final Dataset . . . 33

6 Experiments **39**
6.1 Closed source language models . . . 39
6.2 DeepEx . . . 40
6.3 REBEL . . . 40
6.3.1 Baseline Experiments . . . 41
6.3.2 Comparison to other models . . . 41
6.3.3 Experiments on CTI-dataset . . . 42
6.4 Evaluation of previous studie . . . 43
6.5 Pre-Processing of CTI-Texts . . . 44
6.5.1 Technical Implementation . . . 45
6.5.2 Effect on Models Performance . . . 45
6.6 Experiments with Datasets . . . 45
6.6.1 Changes to Entity Types . . . 45
6.6.2 Changes to Relations . . . 46

7 Results and Evaluation **47**
7.1 Architecture of the final model . . . 47
7.2 Cross Validation . . . 48
7.3 Evaluation of the attention mechanism . . . 48
7.4 Graph Generation System . . . 49
7.4.1 Text Processing Component . . . 50
7.4.2 Graph Generation Component . . . 50
7.5 Interpretation and Error Analysis . . . 51

8 Discussion **53**
8.1 Availability of relation extraction datasets in the CTI-domain . . . 53
8.2 Choice of Model Architecture . . . 54
8.3 Effect of pre-processing CTI-texts . . . 54
8.4 Limitations . . . 55
8.5 Practical Applications . . . 55
8.6 Future Work . . . 56

9 Conclusion **59**

Bibliography **61**

A Appendix **69**
A.1 Regexes used for the pre-processing 69
A.2 Examples of Graphs 71
A.2.1 Example 1 71
A.2.2 Example 2 72
A.2.3 Example 3 73
A.2.4 Example 4 74
A.3 Error Analysis 75

1 Introduction

1.1 Motivation

Since the COVID-19 Pandemic, the scale and sophistication of cyberattacks by malicious actors have only further increased [2]. This poses a substantial threat to our increasingly digitized and interconnected working world. Examples like the Colonial Pipeline [11] and SolarWinds [10] illustrate the potential impact of attacks on critical infrastructure, as well as global cyber security. Consequences of cyber-attacks include out-of-pocket costs like remediation, investigation of the incident, and legal fees, but also reputational costs which potentially outweigh the former [35].

As threat actors employ similar attacks against different organizations, sharing information about attacks is important for organizations to mitigate the impact and occurrence probability of future incidents. Information concerning potential attacks is known as threat intelligence [47]. In the cyber security context, this information is called *cyber threat intelligence* (CTI) and provides evidence-based reports about existing or emerging threats, including their actions, context, mechanisms, indicators, implications, and actionable advice. This information is necessary to make proactive mitigation decisions. Such information can be gathered from several sharing communities on the internet [20, 57].

An important platform regarding the exchange of CTI is the community project MISP: Malware Information Sharing Project [50]. It can be used for sharing, storing, and correlating *Indicators of Compromise* (IOC) of targeted attacks, threat intelligence, financial fraud information, vulnerability information, or even counter-terrorism information. The MISP-tool supports several features to document information about IT-security events. The best way to textually describe a security incident is the *event report* [54]. In an event report, an analyst can share classical threat intelligence reports, malware analysis articles or even something completely different along with the actionable data in a structured way using the markdown format.

In addition to textual reports, MISP also provides functionality to visually describe events in the form of an event graph. This view helps readers to quickly get an understanding of an event by providing an overview of key entities of a report, as well as their relations. Due to the busy time schedule of security analysts, the authors of a report often lack the time to build the corresponding event graph. This is an issue, as the graph greatly improves the quality of the report.

1.2 Problem Statement

The MISP project is developed by the principles of automation, simplicity, and visualisation [50]. Following these principles, this work aims to contribute to the community project by developing an automatically generated event graph based on event reports written by security analysts.

This thesis tries to solve this task by employing methods from the field of *Natural Language Processing* (NLP). The specific fields of NLP necessary for solving this task are *Named-Entity Recognition* (NER), as well as *Relation Extraction* (RE).

The fields of NER and especially RE are still under development and don't offer an out of the box solution to extract entities and their relations from texts. Especially longer texts pose a challenge as they contain a lot of entities, which need to be checked for possible relations [58].

Recently, lots of research has gone into knowledge extraction form cyber security documents. Here systems are specifically designed for use on cyber security documents in contrast to general approaches mentioned above. Gao et al. [23] identified challenges to accurately extract threat knowledge from natural-language CTI text.

The primary challenge lies in the significant nuances that are specific to the security context, like the inclusion of special characters (such as dots and underscores) in IOCs. These nuances create confusion for most NLP modules, such as sentence segmentation and tokenization, which leads to the inefficacy of existing information extraction tools.

1.3 Research Questions

In this project several questions regarding the extraction of knowledge from MISP event reports will be addressed. This means identifying important entities and their relations in texts. For example: malware exploits vulnerability.

In the field of NER and RE different model types are being used. Some use classification based [18] *neural networks* (NN) others use sequence-to-sequence models [58]. Finding the right model or combination of models for the task of generating event reports will be one of the major questions of this project.

Additionally pre-processing of the CTI-texts has a big impact on the performance of NLP models. Gao et al. [23] developed a pipeline to handle the massive nuances which exist in CTI-texts (e.g., dots, underscores in IOCs). Finding a fitting pipeline for the use case of generating event graphs will be a question of interest for this project.

For later referencing the research questions will be split into the following questions:

RQ 1: Whats model architecture works best to extract relations from CTI texts?

RQ 2: How does the performance of NLP models for generating event graphs from MISP event reports improve with different pre-processing pipelines, and which pipeline is the most effective for handling the nuances of CTI-texts?

1.4 Structure

This thesis is structured into nine chapters. The first Chapter serves as the introduction to the research problem, research questions, and objectives of the study.

Chapter 2 provides a theoretical foundation of deep learning, natural language processing, and relation extraction. It explores the concepts, algorithms, and techniques used in these fields, with a particular focus on relation extraction.

Chapter 3 provides an overview of related work in the field of relation extraction. It reviews existing studies and datasets, as well as the strengths and limitations of different approaches compared to the techniques used in this work.

The fourth Chapter describes the methodology used in this study. It details the research design, data collection, and analysis techniques.

Chapter 5 provides an overview and analysis of existing datasets in the field of relation extraction. It also describes the creation of a new dataset as part of this work.

Chapter 6 describes the experiments conducted as part of this thesis. Here different existing models are adapted to the CTI-dataset and a new model is developed based on googles T5 model.

Chapter 7 presents the results of the experiments conducted in Chapter 6. It provides a detailed analysis and interpretation of the findings.

Chapter 8 discusses the availability of relation extraction datasets in the CTI-domain, the choices made in this study, and their practical applications. It also suggests avenues for future research.

Chapter 9 serves as the conclusion of this thesis. It provides a summary of the main findings and highlights the contributions, limitations, and implications of the study.

2 Theoretical foundations

Deep learning and especially NLP are highly complex fields. Fortunately, due to the development of many libraries working with language models is possible even without understanding all the underlying concepts. This chapter will provide the reader with the necessary information to understand the following chapters.

2.1 Basics of Deep Learning

This section explains the basics of deep learning necessary to understand the language models used as well as the experiments carried out in this work. To understand deep learning, first some basics about machine learning need to be established as deep learning is subtopic of machine learning.

Machine learning (ML) is a subfield of artificial intelligence that allows a computer to “learn” from data and make predictions based on that data. This is in contrast to classical programming, where the computer processes data using a pre-defined set of rules defined by a programmer.

The main idea behind ML is to provide the computer with input data and the expected answers to that data. Using this data, the model “learns” the rules that generate the expected answers and applies these rules to new data to make predictions.

To perform machine learning, three things are needed:

1. Input data points, in this case CTI-texts
2. Examples of the expected output, which in this thesis are the relations present in the input text.
3. A way to measure the algorithm’s performance and compare its output to the expected output. This performance measurement is typically referred to as the loss function.
   During the training process, the model’s parameters are updated to minimize the loss function, making the model’s predictions more accurate. The choice of a suitable loss function depends on the problem being addressed, and it is critical because it affects the learning process and the performance of the model. In summary, the loss function is an essential component of machine learning that is used to optimize the model’s performance by adjusting its parameters based on the discrepancy between predicted and expected outputs.

In deep learning, this mapping of input to output data is done via transforming the input data through multiple layers until reaching the output state. The name deep learning stems from these multiple layers of representations and not from any kind of “deeper” understanding of the model. Each successive “layer” represents an increasingly meaningful representation of the input data (compare Figure 2.1). In the realm of machine learning, deep learning is a subset of artificial neural networks that involves multiple layers of representation for the input data. The goal of deep learning

is to transform the input data through a series of successive layers until reaching the final output state.

The layers of representation can be thought of as the internal representation of the model for the input data. As the data flows through the multiple layers, it is transformed and abstracted, with each layer providing a higher abstraction level than the previous one. This abstraction process allows the model to learn and extract increasingly complex features from the input data, making it more powerful and capable of performing complex tasks. [9]

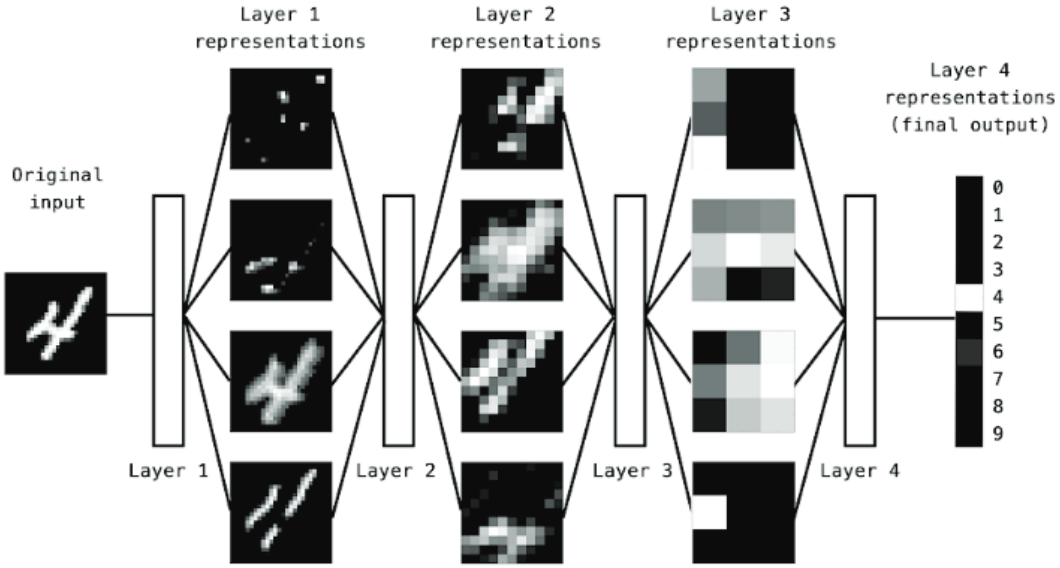


Figure 2.1: *Deep representations learned by a digit classification model [9]*

2.2 Theory and History on Language Models

At the heart of this work is the task of finding certain entities and relationships within a text. However, dealing with natural language is difficult for conventional computer programs, as it is a highly complex and nuanced form of communication. To address this challenge, the scientific field of Natural Language Processing (NLP) has emerged, with the goal of “accomplishing human-like language processing” [43].

Research in NLP has been ongoing for several decades, dating back to the late 1940s. This chapter will provide a brief overview of the past development of NLP techniques and approaches. However, the primary focus will be on recent progress in large language models and transformers, as these are the techniques that are most relevant to this work.

In the early 2000s, neural networks began to be used for language modeling tasks, such as predicting the next word in a text given the previous words [4]. This remains the basis for most NLP models to this day.

Two important developments in this period were the use of recurrent neural networks (RNNs) [49] and long short-term memory (LSTM) networks [28] for language modeling. These approaches posesse the ability to store information which allows for better processing of sequences (such as texts), which made them well-suited for NLP tasks. Another important development was the creation of the Word2Vec model [48], which enabled the large-scale training of word embeddings on huge corpora of unstructured text. These developments marked a significant step forward in the field of NLP.

In 2014, Sutskever et al. [64] proposed sequence-to-sequence learning, which involves mapping an input text to an output text. This process involves using an encoder neural network to process a sentence symbol by symbol and compress it into a vector representation, and a decoder neural network to predict the output sequence based on the encoder state and the previously predicted symbols.

Initially, encoders and decoders for sequences were typically based on RNNs. However, in recent years, transformer networks with self-attention have become the norm. The breakthrough paper “Attention is all you need” [67] introduced a new mechanism to help the model understand the context of the sentence, which has led to significant improvements in NLP tasks.

Pretraining of Large-Language models has become a popular approach in natural language processing in recent years. The idea is to first train a model on a large text corpus to predict masked words, and then fine-tune the model on a specific task. This approach was first proposed in 2015 by Dai and Le [13], but it has only recently started to show significant improvements over state-of-the-art methods. One of the main advantages of pretraining is that it allows the model to learn general language patterns and representations, which can then be fine-tuned for specific tasks such as relation extraction or language translation. This approach has been used to achieve state-of-the-art results on a variety of NLP tasks, and it is expected to continue to play a major role in the field in the future.

2.3 Closed source language models

Recent developments in natural language processing (NLP) have largely been driven by big technology companies, such as Meta, Google, and specialized firms such as OpenAI and Aleph Alpha. While some of these developments have been open source, others, such as Chat-GPT, GPT-3, and luminous, are closed source and can only be accessed through an API. This has made it difficult for researchers to use and study these models in their research, as the cost of using them through an API can be prohibitive and adoption of the models is not possible. However, it is still possible for researchers to build applied systems around the use of these APIs.

2.4 Training NLP models using Huggingface

The training of NLP models is a complex task. To overcome this complexity, the open-source library Transformers [74] has been developed. The library’s goal is to bring the

advances of the transformer architecture and pretrained models to the wider machine learning community. A wide range of pretrained models are included in the framework, which can be easily adapted to individual use cases. Using this library, a model can be deployed(Listing 2.1) or trained(Listing 2.2) in just a few lines of code, making the process of training NLP models easier and more accessible. To use the library a researcher needs to understand the following concepts:

Model In the Hugging Face library, a "Model" refers to a neural network architecture that has been pre-trained on large amounts of data and is capable of performing various NLP tasks such as text classification, named entity recognition, question answering, etc. The pre-trained models available in the library are designed to handle common NLP tasks and can be fine-tuned to adapt to specific use cases.

Tokenizer A "Tokenizer" is a component in the Hugging Face library that is responsible for splitting a sentence into individual tokens or words and converting them into numerical representations that can be fed into the model.

Trainer The "Trainer" is the component in the library that is responsible for training the model. It takes the model, the input data, and the desired output as inputs and iteratively adjusts the model's weights to minimize the difference between the predicted output and the actual output.

Epoch An "Epoch" refers to a complete iteration over the training data. During each epoch, the model sees all of the training data once, and the weights are updated based on the error from the predictions. The number of epochs determines how many times the model will see the entire training dataset during the training process.

Step A "Step" refers to one iteration of weight updates within an epoch. During each step, the model processes a mini-batch of data, and the weights are updated based on the error from the predictions for that mini-batch. The number of steps per epoch is determined by the number of mini-batches in the training dataset.

```
# load model and tokenizer from huggingface
tokenizer = AutoTokenizer.from_pretrained("Olec/cyber_rebel")
model =  T5ForConditionalGeneration.from_pretrained("Olec/cyber_rebel")

example_text = """LokiBot also known as Lokibot, Loki PWS, and Loki-bot employs
Trojan malware to steal sensitive information such as usernames, passwords,
 cryptocurrency wallets, and other credentials."""
# tokenize the input
input_ids = tokenizer(example_text, max_length=1024, truncation=True, padding='max_length',
                      return_tensors='pt')
# pass the input trough the model
generated_tokens = model.generate(input_ids=input_ids['input_ids'],
                            attention_mask=input_ids['attention_mask'],max_length=256)
#decode the output tokens
model_output = [tokenizer.decode(s, skip_special_tokens=False,
```

```
                                clean_up_tokenization_spaces=True) for s in generated_tokens][0]

print(model_output)
# <pad><triplet> Lokibot<malware> Trojan malware<malware> duplicate-of<triplet>
#Lokibot<malware> steal sensitive information<attack-pattern> uses
```

Listing 2.1: *Code to deploy a model using the huggingface transformers libary*

```
from datasets import load_dataset
from transformers import AutoTokenizer

#load a dataset; to keep this example simple a generic dataset was chosen
dataset = load_dataset("yelp_review_full")
#tokenize the dataset
tokenizer = AutoTokenizer.from_pretrained("bert-base-cased")
def tokenize_function(examples):
    return tokenizer(examples["text"], padding="max_length", truncation=True)

tokenized_datasets = dataset.map(tokenize_function, batched=True)

#load to model like in the deployment example
from transformers import AutoModelForSequenceClassification

model = AutoModelForSequenceClassification.from_pretrained("bert-base-cased", num_labels=5)

#train the model using the trainer class
from transformers import TrainingArguments, Trainer

training_args = TrainingArguments(output_dir="test_trainer", evaluation_strategy="epoch")

trainer = Trainer(
    model=model,
    args=training_args,
    train_dataset=tokenized_datasets["train"],
    eval_dataset=tokenized_datasets["test"],
)
trainer.train()
```

Listing 2.2: *Code to train a model using the huggingface transformers libary*

2.5 T5

In the paper “Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer” [59], the authors from Google propose T5, a transfer text-to-text transformer that is pre-trained on a large dataset before fine-tuning on a downstream task. T5 is inspired by the masked language modeling technique used for Googles previous model BERT [16], but with the added ability to mask multiple tokens at once (Compare Figure 2.2).

The authors pre-train T5 on C4, a dataset based on the Common Crawl dataset scraped from the web. The Common Crawl is a publicly available web archive that scrapes the internet every month and saves the resulting 20TB of text after removing markup. However, due to the nature of the data, many of the texts do not contain real text. To overcome this issue, the authors use heuristics to clean the dataset, such as

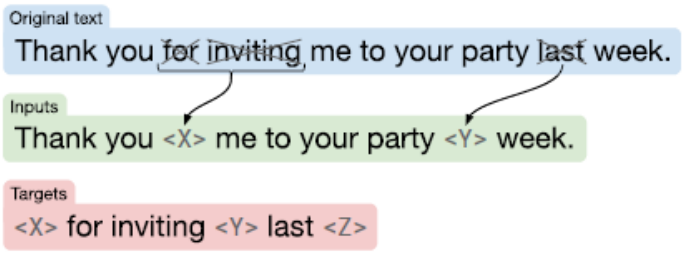


Figure 2.2: *Visualisation of the pre-training process used for the T5 model [59]*

removing lines that contain the word javascript or lines with less than 3 words. Only English data is used for pre-training, resulting in 750 GB of data.

The goal of T5 is to measure general language learning abilities and it is evaluated on a diverse set of downstream tasks. The authors use a task-specific prefix in the input text to indicate the task to the model. Tasks include text similarity, translation, and summarization(Figure 2.3). T5 sets new records for many of the downstream tasks, demonstrating its effectiveness as a general purpose language model.

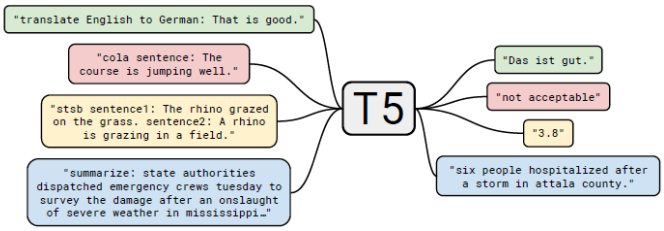


Figure 2.3: *Visualisation of the T5 model's features [59]*

2.6 Relation Extraction

Extracting relational facts from texts(Example in Figure 2.4) has long been a challenging problem in natural language processing (NLP). The ability to extract relationships from unstructured text is useful for a wide range of tasks, such as knowledge graph construction, question answering, and information retrieval.

The malware steals credentials through the use of a keylogger.

Head-Entity	Relation	Tail-Entity
The malware	uses	steals credentials
Type: malware		Type: attack-pattern

Figure 2.4: *Example of relation extraction in a CTI-text*

2.6.1 Approaches to Relation Extraction

However, the term “relation extraction” has been interpreted differently by researchers, as discussed in [65]. In this section, we will provide an overview of the various approaches that have been proposed for relation extraction, highlighting their strengths and limitations.

Early approaches to relation extraction typically followed a pipeline approach, first identifying entities and then classifying the relations or lack of relation between those entities, a task known as relation classification (RC). These methods used either convolutional neural networks (CNNs) or long short-term memory (LSTM) networks [79, 85]. However, later transformer models have replaced these techniques as the current state of the art [76].

Other pipeline systems that tackle both components of relation extraction: named entity recognition (NER) and relation classification. These systems jointly train the NER and RC components, as the two tasks share information. For example, some systems first identify named entities, and then use a classifier to extract the relations between those entities. Both models may share part of the encoder, allowing them to benefit from each other’s predictions. More recently however, sentence-level relation extraction models have been based on transformer models [17].

Despite their success, these pipeline and table methods have some issues. They assume that there is only one relation per entity pair and are also computationally expensive as they require computing all pairs.

More recently, newer approaches [55, 81] frame relation extraction as a sequence-to-sequence problem similar to translation. This solves the issues of the table and pipeline approach but has its own issues [84]. For example, triplets need to be linearized into somewhat arbitrary sequential order, such as alphabetically. The issue of linearization is explored by [80].

Recent seq2seq models like Google’s T5 [59] or BART made by Facebook [42] have been used in NLP tasks including relation extraction by reframing them as seq2seq tasks [6, 7, 32].

While REBEL [32] holds the first place in the “Joint Entities and Relation Extraction on DocRED” task on paperswithcode.com with F1 of 0.471, other models have performed better on the “Relation Extraction on DocRED” benchmark.

The main difference between these two tasks is that Joint Entities and Relation Extraction only takes text as input while in Relation Extraction, the entities between which relations should be extracted, as well as other additional information such as coreferences, are provided to the model. This additional information can be beneficial, as it allows the model to focus on the specific task of relation classification.

2.6.2 Evaluation

Evaluation of machine learning models is an important and error-prone step in their development. An essential step in this process is the division of the data into training

and test sets, where the test set is used to evaluate the model's performance and assess its generalization ability.

In some cases, the test set is further divided into a validation or development set, which is utilized to tune the model's hyperparameters. This technique helps prevent overfitting, where the model becomes too complex and memorizes the training data rather than learning the underlying patterns. Conversely, underfitting occurs when the model is too simple to capture the patterns and trends in the data. [83]

When data is limited, k-fold cross-validation can be used to train and test the model. This approach involves dividing the dataset into k folds, using k-1 folds for training and the remaining fold for evaluation. This process is repeated by switching the folds until all folds are utilized for testing, and an average is calculated to estimate the model's overall performance [14].

The selection of an appropriate metric is critical in the evaluation of machine learning models, as incorrect metric selection can lead to misleading results. In machine learning, the choice of metrics depends on the task at hand. If the task is treated as a classification problem, then precision, recall, and F1-score are sensible metrics to use. These metrics evaluate the performance of the model in terms of how well it is able to identify the correct relationships between entities in the input text. For relation extraction, the classification is considered correct only if the labels of the entities (e.g., malware, tool), their surface form (i.e., did the model find the right text span), as well as the linking relation (e.g., targets) are generated correctly. Therefore, the evaluation of the model needs to take into account all these 3 aspects, and not only the correct classification of the relation.

The basis for all classification metrics is the confusion matrix(Table 2.1). In a confusion matrix, the predictions of a model are divided into four categories: true positives (TP), false positives (FP), false negatives (FN), and true negatives (TN). These categories are important because in an imbalanced dataset, only differentiating between true and false predictions could lead to a bad representation of the model's performance.

Precision and recall are relative metrics that are derived from the absolute values of the confusion matrix. Precision is calculated as:

$$Precision = \frac{TP}{TP + FP}$$

It measures the proportion of correct predictions among all the positive predictions made by the model.

		Prediction outcome		
		p	**n**	**total**
actual value	**p′**	True Positive	False Negative	P′
	n′	False Positive	True Negative	N′
	total	P	N	

Table 2.1: *The confusion matrix*

Recall is calculated as:

$$Recall = \frac{TP}{TP + FN}$$

It measures the proportion of correct predictions among all the true positive instances in the test set.

A high precision value indicates that the model is good at avoiding false positives, while a high recall value indicates that the model is good at identifying all the true positive instances. In many cases, these two metrics are in trade-off with each other, and a high value of one often comes at the cost of a lower value of the other.

F1-score is the harmonic mean of precision and recall, and it is commonly used to balance the trade-off between precision and recall. It is calculated as :

$$F1 = 2 * \frac{Precision * Recall}{Precision + Recall}$$

2.7 Relation Extraction in Literature

Prior to adjusting models to CTI-texts, potential approaches identified as promising in the literature review were subjected to a comprehensive evaluation. In this stage, pertinent code was scrutinized, executed, and debugged to develop a profound technical comprehension of the systems outlined in the literature. This section elaborates on the discoveries made during this phase.

2.7.1 Utilization of Distant Supervision and Customized Loss Functions

The best model for the “Relation Extraction on DocRED” benchmark is the model proposed by Tan et al. in their paper “Document-Level Relation Extraction with Adaptive Focal Loss and Knowledge Distillation” [66]. Techniques used to achieve state-of-the-art performance will be explored in this section.

Research in this paper is grounded on the finding of previous research, that contextual information has proven to be helpful for relation classification tasks [75]. To incorporate this information, they use a pre-trained language model to extract the contextual representation for each entity pair. Entity mentions are marked by a special token “*” and a special attention mechanism is used to encode two-hop relations.

To further utilize the annotated data, the authors train a relation classification model trained on the human-annotated data as the teacher model. The final model is trained both on the labels from the distantly supervised data as well as the labels from the teacher model.

To overcome the imbalanced class distribution present in DocRED [77], the loss function is adapted to Adaptive Focal Loss to give more weight to infrequent classes.

Overall, the best model based on Roberta-large achieved an F1 score of 67.28 on the test set. Ablation studies showed that distant supervision had an impact of about 3 F1 percentage points, the special attention mechanism had an impact of 2 F1 percentage points, and Adaptive Focal Loss had an impact of 0.65 F1 percentage points.

2.7.2 The Impact of Entity and Contextual Information

The work “Entities Structure Within and Throughout: Modeling Mention Dependencies for Document-Level Relation Extraction” by Xu et al. (2021) [75] achieved second place on the “Relation Extraction on DocRED” leaderboard, with an F1 score of 65.92. The authors propose to provide the relation classification model with additional information encoded in a matrix, which captures the co-occurrence and coreference structures of mentions in the text.

- Co-occurrence structure: Whether or not two mentions reside in the same sentence.
- Coreference structure: Whether or not two mentions refer to the same entity.

Specifically, the matrix captures whether or not two mentions reside in the same sentence and if they refer to the same entity. A visual representation of this structure is provided in Table 2.2. The authors show that incorporating this structural information improves the performance of the model, as can be seen in Table 2.3. Performance effects of incorporating different types of structural information can be seen in Table 2.3, where it can be observed that especially inter sentence structural information leads to improvement in relation extraction performance.

		Coreference	
		True	False
Co-occurence	True	intra+coref	intra+relate
	False	inter+coref	inter+relate

Table 2.2: *The formulation of entity structure [75].*

Dependency	**F1**
SSANBiaffine (RoBERTa Large)	62.08
- intra+coref	61.57
- intra+relate	61.91
- inter+coref	61.74
- inter+relate	61.84
- intraNE	61.97
- all	60.58

Table 2.3: *Performance of the model [75].*

2.7.3 Leveraging Pretraining on Wikipedia Data

REBEL(Relation Extraction By End-to-end Language generation) [32] is a relation extraction model released in November 2021. It is a leader on the Relation Extraction on CoNLL04 and Joint Entities and Relation Extraction on DocRED tasks on paperswithcode.com. The model is based on the BART model [42] and uses two interesting techniques: pretraining on a large amount of data generated from Wikipedia and framing relation extraction as a sequence-to-sequence problem.

The dataset used to train REBEL was created by expanding the pipeline proposed by [19] to extract relations from Wikipedia abstracts. The final dataset contains the 220 most frequent relations from the extracted data, but contains no entity types. The dataset consists of almost 900,000 relation triplets.

BART is first trained on the large pretraining dataset to create the pretrained REBEL model. The pretrained model achieves a 0.74 F1 Score on the REBEL dataset and is capable of extracting 220 relations from texts. However, these relations have a Wikipedia character (e.g., located in the administrative territorial entity, place of birth) and may not be useful for the CTI domain in this form.

The BART model pre-trained on the Wikipedia dataset is referred to as REBEL by the authors and can be fine-tuned on specific RE datasets. The authors evaluated REBEL's performance on several known RE datasets. REBEL achieved state-of-the-art results on the CONLL04 dataset (F1 0.754) [62], DocRED (F1 0.471) [77], and the New York Times dataset (F1 0.920) [82]. Notably, pretraining has a greater impact on smaller datasets like CONLL04 (+4.2% F1 score) than on larger datasets like the NYT (+0.2% F1 score).

2.7.4 Work with similar approach

In this work, the author has focused on the extraction of relations from MISP event reports in order to create knowledge graphs. It is worth mentioning that a fellow student has conducted research in a similar area prior to and during the time of this thesis [21].

The author of this work and the author of [21] have developed a dataset together for relation extraction and named entity recognition in the cyber threat intelligence (CTI) domain. This section will analyze the models developed by the author of [21] on the CTI-dataset and their effectiveness in extracting information from threat reports for the creation of knowledge graphs.

First, the corpus dataset desribed in Chapter 5.3.1 was used to retrain the tokenizer and train the T5 base model using Masked Language Modeling. Then, the New York Times (NYT) dataset was used to train the model first for Named Entity Recognition followed by Relation Extraction on the NYT dataset. The task was framed as a sequence-to-sequence problem similar to [32]. Finally, the CTI dataset was used to train a relation extraction model on top of the pretrained model. Additionally, a CTI named entity recognition model was also trained on the T5 model, which was only pretrained for NER.

In order to evaluate the models, [21] developed a code to parse entities and relations from the model output and compare them to the expected relations. However, as the produced scores were relatively low, the Levenshtein [41] distance was used to compare string similarity instead of exact matches. This approach allows for predictions to be evaluated as true even if the model predicted a slightly different text as expected. It is important to keep in mind that this approach may lead to more lenient evaluation when comparing models. Additionally, as only a small portion of the CTI dataset contained relations, a second dataset was deduced from all sentences to sentences which contained labeled relations. The results of the evaluation can be found in Table 2.4.

The results of the evaluations on the NYT dataset are promising, with good performance both for relation extraction and named entity recognition tasks. The relation extraction F1 score is only 5.56 points lower than the state-of-the-art score reported in [32]. However, when it comes to the CTI-specific dataset, the performance is not as satisfactory. The F1 score for named entity recognition is 0.24, which is acceptable, but the relation extraction performance is poor, with an F1 score of 0. The results on the dataset containing only sentences with relations are slightly better, with an F1 score of 0.046. However, these results are not sufficient to build a knowledge graph for the CTI domain.

2.7.5 Zero-Shot Learning Approaches in Information Extraction

In their paper [69], the authors present a model called “DeepEX” which is capable of performing both relation classification and Open Information Extraction. Open Information Extraction refers to the task of extracting triplets (head, relation, tail)

	Precision	Recall	F1-Score
CTI-NER	0.3084	0.2178	0.2399
NYT-NER	0.8893	0.9298	0.898
NYT-RE	0.8708	0.8773	0.8644
CTI-RE	0.0464	0.0474	0.0462
CTI-RE-all	0.0002	0.0003	0.0002

Table 2.4: *Evaluation Results*

from text with no predefined relation classes. The authors claim that their model can perform these tasks in a "zero-shot" setting, meaning that it can perform these tasks without any task-specific fine-tuning, only using the latent knowledge present in pre-trained language models.

The architecture of the model is similar to the REBEL [32] model, where every information extraction problem is treated as a text-to-triple problem, that is, translating input text to output triples.

To enhance the performance of the model, the authors introduce a ranking model which chooses the best triplet from k generated triplets. The proposed model produces state-of-the-art results on multiple Open Information Extraction datasets, showing the effectiveness of the zero-shot approach in information extraction tasks.

2.8 Preprocessing of CTI-Texts

Extracting knowledge from CTI-texts is not trivial as these text posess nuances like "IP-adresses" or hash-values which confuse NLP modules [22, 23, 33]. A common approach [22, 23, 33] to address this issue is to pre-process the texts before extracting relations with NLP modules. This is mostly done by using regular expressions, as they are able to recognize the pattern of these CTI-specific text-parts. These CTI-specific words with are detected by the regular expression are then replaced by a natural language word. For example "192.168.4.2" would be replaced by "ip-address". This makes the task of the following model easier. While the system developed by Gao et al. [23] is build upon the existing library ioc-parser [24] the authors of TTPDrill [33] developed their own set of regular expressions. Data provided by Gao et al.[23] shows this process is highly effective as it increased performance in all compared approaches. It is important to note however, that the effect of the pre-processing was mostly visible in an increase of the recall values, while sometimes even reducing the precision slightly. This shows labeling of CTI-specific text-parts helped the model to not falsy label words as important entities.

3 Related Work

Sharan et al. [63] build a system to generate Knowledge Graphs from CTI reports. In this approach, instead of using RE, relation triplets were extracted by first generating Open Information Extraction (OIE) triples and subsequently labeling head and tail of the extracted triplet with NER. This way edges of the graph are not limited to a predefined set of relations. Additionally to NLP related tasks Sharan et al. also tackle Knowledge Graph related issues of data redundancy and ambiguity. To overcome this challenge a canonicalization technique is developed to fuse similar nodes of the graph.

Gao et al. [23] proposes a system to enable efficient cyber threat hunting via extracting threat behaviors from CTI text and querying the extracted behaviors from system audit logs. The system consists of multiple components. For this project the NLP pipeline build by the authors is especially interesting. By dividing CTI texts into blocks, dealing with IOCs and building trees out of the extracted entities EFFHUNTER [23] is able able to achieve a very high F1-Score of 96.64%. This score is substantially higher than general RE models.

Gao et al. [22] adopt a Conditional Random Field (CRF) model to extract security-related entities in unstructured texts. Additionally, methods developed by Gao et al. [23] are applied to protect IOCs to increase the performance of the NER. The extracted data is stored in a system build upon a Neo4j database, Elasticsearch. and React. In addition to providing automated data gathering and management the system allows analysts to search for index terms and explore the knowledge graph.

Paolini et al [58] developed a framework (TANL) to solve many structured prediction language tasks including joint entity and relation extraction, nested named entity recognition, relation classification, semantic role labeling, event extraction, coreference resolution, and dialogue state tracking. While other solutions [22, 23] are built for the particular domain of CTI texts, this [58] is a general purpose model. Even thought TANL can solve many different tasks on all domains it still has state of the art performance.

In 2017, Husari et al. [33] developed the TTP drill system, which uses a custom Threat Action Ontology to process articles about malware and summarize them in a structured way. Each article in the dataset is focused on a specific malware, and the system is designed to extract relevant information about the tactics, techniques, and procedures (TTPs) used by the malware. To pre-process the articles, the TTP drill system first replaces certain specialized terms with more general terms. This is similar to the approach taken by [23]. After pre-processing, the TTP drill system uses the Stanford typed dependency parser [46] to identify candidate actions in the text. These candidates are then mapped to the Threat Action Ontology using a similarity

score, allowing the system to extract structured information about the TTPs used by the malware. It is worth noting that the TTP drill system uses a Support Vector Machine (SVM) to pre-distinguish relevant articles from advertisements, to only process relevant documents.

CyNER is a Python library for named entity recognition (NER) developed by Alam et al. [1]. It combines transformer-based models for NER with a heuristic model for identifying threat indicators such as malware hashes and compromised IP addresses. To evaluate the performance of CyNER, the authors provide a manually labeled dataset consisting of 60 threat intelligence reports referenced on the MITRE att&ck website under the software category. They compare the performance of several transformer models on this dataset and find that XLM-RoBERTa-large achieves the best results with an F1 score of 76.66. It is worth noting that CyNER is focused solely on NER and does not address relation extraction (RE). To build knowledge graphs both NER and RE or a joint extraction is necessary.

Tan et al. [66] is the currently leader on the “Relation Extraction on DocRED” [77] task. This result is attained by using a semi-supervised approach where knowledge distillation is used to overcome the differences between human annotated data and distantly supervised data. A major difference to [58] is the annotation of entities with a special token. This trick makes it easier for the model to find relations, as part of the joint task is already solved.

In conclusion, while there is existing research on extracting information from CTI texts, much of it is not publicly available, making it difficult to validate and improve upon. Publicly available solutions, like CyNER, are limited to entity recognition and do not address relation extraction, which is essential for building knowledge graphs. This work builds upon state-of-the-art NLP techniques used in general text processing and adapts them to the CTI context, incorporating relevant techniques from other CTI papers like Gao et al. [23]. By doing so, it aims to provide a publicly available model that can extract both entities and relations from CTI texts, which can be used to construct a Security Knowledge Graph that enables efficient cyber threat hunting.

4 Methodology

The aim of this study is to investigate if and how well relations can be extracted from CTI (Cyber Threat Intelligence) texts. This chapter will present the methodology choices that were made throughout the thesis, including the methods used to create the dataset, develop the deep learning model, and the final system. Additionally, limitations will be discussed.

4.1 Fundamental Methodological Choices

The study is designed as an exploratory inductive study, as the goal is to develop a relation extraction system for CTI texts. This approach was chosen as it allows for different approaches to be tried out and observations to be compared. Using method is particularly suitable for the goal of this thesis as it provides the flexibility to experiment and make observations in order to improve the final system.

4.2 Literature review

A literature review was conducted to find relevant works for this thesis, as described in [38]. This step is crucial to ensure that all pertinent literature has been reviewed. As the topic of this thesis is interdisciplinary, the literature review was split between CTI-related NLP and general NLP. For both branches, queries consisting of one or multiple index terms were constructed and searched on Google Scholar, Google Websearch, and the library of Karlsruhe Institute of Technology. The literature discovered in this way was used as a theoretical basis for this thesis.

In addition, the website paperswithcode.com was used as it provides a good overview of state-of-the-art NLP implementations, as well as the corresponding code.

Research Queries The following queries were run trough the listed search engines:

1. cti nlp
2. cti relation extraction
3. cti knowledge graph
4. cti data relation extraction
5. relation extraction
6. knowledge graph
7. reproducibility deep learning

4.3 Dataset

The dataset used in this thesis is a crucial factor for the success of the deep-learning experiments. To create a representative and accurate dataset, it is important to

carefully consider how it will be generated. One common method to achieve this is by having multiple annotators [70].

For this thesis, the dataset was created by two IT-security students. The small number of annotators results in a biased dataset towards the way the two students interpret CTI-texts. While having more annotators and a two-stage review process would likely produce better results, due to time and resource limitations, a single-stage annotation process was chosen.

4.4 Deep-Learning Experiments

Deep learning experiments are a significant part of this thesis, as the author aims to train natural language processing (NLP) models to extract relations from CTI texts. This requires finding the optimal combination of data, base model, architecture, and configuration.

This task can be seen as a laboratory experiment, as defined in [73]. In such experiments, causal links are investigated in a controlled environment by manipulating a variable in a reproducible fashion and measuring its effect on the experiment. In the context of computer science, this laboratory consists of both hardware and software environment.

Reproducibility in Deep-Learning Reproducibility is a cornerstone of scientific research, as it allows other researchers to verify and build upon the work that has been done. However, reproducibility can be a challenge, even for the original researchers themselves, as evidenced by Begley et al. [3]. This problem is also prevalent in computer systems research, as confirmed by Collberg and Proebsting [12], who found that out of 402 experimental papers, they were only able to reproduce 32.3% without communicating with the authors, and 48.3% with communication. Interestingly, papers by authors with industry affiliation had a lower rate of reproducibility.

In their study on the reproducibility of AI experiments, [26] defined reproducibility in empirical AI research as “the ability of an independent research team to produce the same results using the same AI method based on the documentation made by the original research team”. This definition will be used as the basis for our understanding of reproducibility in this work.

Gundersen et al. [26] also identified a set of variables that are good indicators for reproducibility after reviewing the literature. These findings, which are presented in Table 4.1, will be used as guidelines for conducting experiments in this thesis. By following these guidelines, the author aims to ensure that the experiments are as reproducible as possible, which is crucial for the scientific validity of this work.

Technical implementation of the experiments The technical implementation of the experiments in this work was developed with reproducibility in mind, following the definition provided by [26]. The experiments in this thesis can be divided into two

Table 4.1: *Indicators for reproducibility by [26]*

Method	
Problem	The problem the research seeks to solve.
Objective/Goal	The objective of the research.
Research method	The research method used
Research questions	The research question asked.
Pseudo code	Method described using pseudo code.
Data	
Training data	Is the training set shared?
Validation data	Is the validation set shared?
Test data	Is the test set shared?
Results	Are the results shared?
Experiment	
Hypothesis	The hypothesis being investigated.
Prediction	Predictions related to the hypotheses.
Method source code	Is the method open sourced?
Hardware specifications	Hardware used.
Software dependencies	For method or experiment.
Experiment setup	Is the setup including hyperparameters described?
Experiment source code	Is the experiment code open sourced?

stages:

The first stage involves thoroughly analyzing approaches that were identified as promising during the literature review. This is done by downloading and reading the code and data provided by the authors, testing if the code can be run, debugging the code for better understanding, and adopting the code to the domain-specific data if the initial tests were successful. This stage is crucial for understanding the inner workings of the approaches and selecting approaches which can be applied to the CTI-domain. In the second stage of the experiments, the NLP architectures chosen in the first stage were evaluated in structured experiments. The goal of these experiments was to identify the best performing architecture for the use case of generating CTI knowledge graphs.

While the first stage of the experiments involved unstructured python scripts or jupyter notebooks, the second stage featured structured experiments to measure the effect of different parameters on the model's performance. This was done by implementing a parameterized base training script for each environment and configuring a .yaml file for each experiment. This structure was inspired by the implementation of [32] and helps to ensure that only the desired parameters of the training script are changed, while also keeping the code cleaner. All experiments were implemented using the transformers libary [74].
In this thesis, the cross-entropy loss function [83] is used to evaluate the output of

the language models and adjust the training process. This loss function is commonly used for language models, and it calculates the loss for each token in the sequence by comparing the expected token to the generated token.

The experiments in this work were tracked using Weights and Biases (W&B) [5]. This allowed for the progress of the experiments to be monitored continuously, which helped to identify any issues that may have arisen during the training process. W&B also provided a platform for comparing different variations of one experiment, which was useful for identifying the most promising approaches. Finally, W&B was used as a platform for storing the results of the experiments, which facilitated the analysis and interpretation of the results.

4.5 System Development

AI models are not standalone programs, they need a traditional program to allow users to interact with them. In order to evaluate the models not only in terms of metrics such as the F1 score, but also in an applied setting, prototypes of different versions were developed. This work uses prototyping as defined in [73], which involves creating a simplified version of the final product in order to test and refine its functionality

The prototypes were tested by the supervisor and fellow IT security students. This allowed for the models to be evaluated in a real-world setting, which provided valuable insights into their practical applicability. By iteratively refining the prototypes based on feedback from these tests, the final version of the system was able to meet the needs of the target users.

4.6 Concluding Summary

The methodology of this thesis is based on an experimental approach, with the aim of finding out if and how well relations can be extracted from CTI texts. The choice of an exploratory inductive study allows for different approaches to be tried and observations compared.

The foundation of the deep-learning experiments is the dataset, which was created by two IT-security students. Although having multiple annotators and a two-stage review process would produce the best results, due to time and resource limitations, a single-stage annotation process was chosen.

Throughout the experiments, various approaches were tried, with a selected few being analyzed in-depth. The technical implementation of the experiments was designed with reproducibility in mind, and promising models were developed into prototypes.

The steps taken as part of this work are visualized in Figure 4.1.

Figure 4.1: *Visualisation of the steps of this thesis*

5 Datasets

Deep learning models require to data to learn the desired task. This chapter examines available datasets for general relation extraction and relation extraction in the CTI domain. Additionally the ontology and statistics of the developed dataset are explained.

5.1 Relation Extraction Datasets

In order to evaluate existing systems and pre-train language models for the task of relation extraction, it is necessary to have access to large datasets specifically designed for this task. This section analyzes publicly available relation extraction datasets that can be used for this purpose.

5.1.1 New York Times

The New York Times (NYT) dataset is a widely used relation extraction dataset that was produced using the distant supervision method [61]. It consists of 1.18M sentences sampled from 294k New York Times news articles published between 1987 and 2007, and includes 24 valid relations.

This work uses an adopted version of the NYT dataset proposed by [82]. This version filtered out sentences with more than 100 words and sentences that contained no positive triplets, resulting in a dataset of 66195 sentences. From this dataset [82] randomly selected 5000 sentences as the test set, 5000 sentences as the validation set, and used the remaining 56195 sentences as the training set.

In addition to the 24 relations from the categories of location, people, business, and sports, the NYT dataset also includes entity labels for the entities in each sentence. These entity labels include “person,” “organization,” and “location”.

The inclusion of entity labels in the NYT dataset is useful for training models to recognize both entity and relation types. By learning to identify the entities in a sentence, models can more accurately extract the relations between them. This is especially important for tasks such as knowledge graph construction, where both the entities and their relationships need to be accurately identified in order to build a comprehensive and accurate representation of the domain.

5.1.2 CONLL04

The CONLL04 dataset [62] is another widely used relation extraction dataset. It is composed of sentences from news articles and is annotated with four entity types: “person”, “organization”, “location” and “other”. The dataset includes five relation types: kill, work for, organization based in, live in and located in.

5.1.3 DocRED

DocRED [77] is a relatively new relation extraction dataset, created in 2019. It was created in a distantly supervised fashion using Wikipedia and Wikidata, and also includes a manually annotated portion. One notable aspect of DocRED is that it not only contains inter-sentence relations, but also relations that span across multiple sentences. This is particularly interesting for knowledge graph construction, as many relations in real-world datasets are likely to span multiple sentences.

The DocRED dataset includes a human-annotated test and dev set, as well as a portion of the training set that is manually annotated. In total, the human-annotated portion of the dataset consists of 5,053 documents with 40,276 sentences. The distantly supervised portion of the dataset includes 101,873 documents containing 828,115 sentences.

Overall, the DocRED dataset is a valuable resource for training and evaluating models for relation extraction, particularly for tasks that involve extracting relations that span multiple sentences. Its combination of manually annotated and distantly supervised data allows for a more comprehensive evaluation of model performance.

5.1.4 Comparison of RE-Datasets

Most of the relation extraction datasets analyzed in this work are built upon news articles, which may be an advantage as CTI texts are a type of news articles. However, it is challenging to compare these datasets directly, as they can only be evaluated in the context of their own specific tasks and data.

One way to compare relation extraction datasets is to consider their difficulty and label noise. Difficulty refers to how difficult it is for a model (or a human) to identify the relations that the dataset defines as correct. For example, extracting the relation “Peter <person> Berlin <location> resident” from the sentence “After spending his childhood in France, Peter now moved to Berlin where his parents grew up.” would be more difficult than extracting it from the sentence “Peter lives in Berlin.”. Additionally, extracting relations that span across multiple sentences is especially difficult.

Label noise, on the other hand, refers to the accuracy of the labels in the dataset. Distantly supervised data, in particular, can sometimes contain noise, which can affect the quality of the dataset. It is also important to consider the size of the dataset and whether it is sufficient for the intended task.

On the Papers With Code leaderboards for relation extraction, F1 scores for the NYT, CoNLL04, and DocRED datasets are 93.9, 76.65, and 67.28, respectively. This suggests that the DocRED dataset is the most difficult, due to its inclusion of inter-sentence relations, while the NYT dataset is either easier or of higher quality(meaning less label noise) than the CoNLL04 dataset (which may be too small or have incorrect annotations). It is worth noting, however, that these scores should be interpreted with caution, as they do not necessarily reflect the true difficulty or quality of the datasets, but rather the performance of specific models on these datasets.

Due to its simplicity and big size the NYT dataset was chosen as a pre-training dataset to teach the model the concept of relation extraction before fine tuning it on the CTI-domain.

5.2 Public CTI Datasets

The choice of CTI dataset is fundamental in the development of relation extraction systems for this thesis. It plays a key role in adapting general-purpose NLP models to the specific task of extracting relations and entities from CTI texts, as it defines the possible entity and relation types that the final model's output will include.

In the paper "Recent Progress of Using Knowledge Graph for Cybersecurity" [44], a thorough analysis of knowledge graphs in the CTI domain is presented, including a comparison of publicly available datasets. The authors found that while there are many datasets available for named entity recognition (NER) in the CTI domain, there is only one dataset specifically designed for relation extraction (RE). This highlights the need for more public datasets in this domain, as they are essential for the development and evaluation of relation extraction systems. The following section will compare existing publicly available datasets as well as describe the dataset created as part of this thesis.

APTNER [70] is a CTI dataset for named entity recognition (NER). It includes 21 categories and is formatted in IOB/BIOES format. However, it is only useful for NER tasks and is not suitable for relation extraction. The dataset is available on GitHub.

CTI-Reports-dataset [36] is a CTI dataset formatted in IOB format. It includes 9 entity classes and a total of 15,720 entities. The authors of the dataset also developed a model for NER and achieved an F1 score of 75.05% on the NER task using this dataset.

CyEnts [8] is a CTI dataset that includes texts from companies such as MacAfee, Mandiant, FireEye, and Juniper Networks. It includes 781 entities in 29 entity classes.

CyNER [1] is another CTI dataset formatted in IOB format. It includes 5 entity classes and a total of 4530 entities. The dataset is sourced from the MITRE ATT&CK Website Software section.

DNRTI [71] is a CTI dataset created by researchers from China formatted in IOB format. It includes 13 entity types and a total of 36,412 entities.

MalKG [60] is a CTI dataset created for the purpose of evaluating the cybersecurity graph framework of the same name. The dataset consists of two subsets: MT3K and MT40K. MT3K is a small, hand-annotated dataset containing 80 documents and 3,000 triplets. MT40K is a larger, automatically annotated dataset containing 1,100

documents and 40,000 triplets. Both datasets are available on GitHub.

One notable aspect of the MalKG dataset is that it only includes triplets, with no context or additional information. This makes it more suitable for relation classification tasks, rather than relation extraction.

Overall, the majority of the CTI datasets analyzed in this work were created with the goal of named entity recognition (NER) in mind. This makes them less suitable for relation extraction (RE) tasks, as they do not include the necessary annotations for extracting relationships between entities.

5.3 Creation of CTI Dataset

Since no fitting datasets for relation extraction are publicly available in the CTI domain, a dataset was created in collaboration with a fellow student. The first step in this process was to create a corpus dataset containing sentences from APTnotes [37], a collection of blogs related to malware campaigns and APTs. This corpus dataset, created by [21] can be used for domain adaptation of models, allowing them to better handle the specific language and terminology used in CTI texts.

To fine-tune the model to the task of relation extraction, the author and the fellow student created a custom dataset based on a selection of articles from various sources. This dataset includes 34 articles from the Google Threat Analysis Group blog, 100 articles from the TrendMicro Research blog filtered for “APT & Targeted Attacks,” and 85 articles from Unit 42. The process of creating this dataset, as well as its statistical properties, will be explained in more detail in the following section.

5.3.1 Creation Process

To create the relation extraction (RE) dataset, the first step was to split the articles into smaller chunks of text that could be processed by NLP models. One sentence was chosen as the length of these chunks, similar to the approach used in the NYT dataset [82]. This keeps the task relatively simple, while still allowing the model to learn contextual information. Alternatively, using two or more sentences, as in the DocRED dataset [77], would make the task more complicated but enable the model to learn inter-sentence relations.

The splitting of the articles was done using spaCy [29], which offers advanced NLP functionality. However, the process was challenging due to the peculiarities of CTI texts, such as the inclusion of IP addresses and code snippets. After splitting, a total of 14,421 sentences were available for annotation. Out of those 14,421 sentences the students managed to annotate 9732. Unfortunately only 480 of them contained relations relevant , while the other sentences just contained NER tags or no labels at all.

5.4 Ontology of the CTI-Dataset

The selected ontology defines the possible entities and their relationships that the model can produce. Choosing a meaningful ontology is crucial for both the model's ability to learn the task and the usability of the developed system to users. In the context of this thesis, this means defining the types of security-related entities and relationships in the security knowledge graph.

While the work of the fellow student focused on the Structured Threat Information Expression (STIX) Standard by the OASIS Open Community, this work focused on the Malware Information Sharing Platform (MISP). However, there is no conflict between these two approaches, as the STIX format is a subset of MISP [52]. The set of entity types and possible relations will be explained in the following section.

Entities The connections that can exist between the entities previously defined in this thesis are referred to as relations. These relationships will be illustrated as edges in the knowledge graph, allowing for a clear visual depiction of how the entities are connected. Tables 5.1 and 5.2 present the relations utilized in this study.

Relations Relations are defined as the possible connections between entities, which were defined in the previous section of my thesis. These relations will be represented as edges in the knowledge graph, providing a visual representation of the connections between entities. The relations used in this work are presented in Tables 5.3 and 5.4.

5.5 Dataset properties and versions

In this thesis, the previously described ontology is the one used for the final model. Throughout the thesis, several modifications to the dataset were made and their effect on the model's performance were evaluated. An overview of all CTI-datasets used in this work can be found in Table 5.5.

5.5.1 Changes to the dataset split

The initial dataset [30] was split randomly into train, test and validation subsets. However, this led to an imbalanced distribution of classes in the three sets. An imbalanced distribution in the training set can lead to the model performing better on the overrepresented classes, while an imbalanced distribution in the test and validation sets can lead to an inaccurate evaluation of the model's performance. To make the best use of the dataset, it was split using a special Python script that aimed to achieve a uniform distribution of all relation classes known as stratified sampling across the different datasets (compare Figure 5.1). However, this could not be achieved perfectly as multiple relations can occur in one sentence. The effect of changing the dataset split can be found in Table 6.3.

Table 5.1: *Entity types chosen for this work(Part 1)*

Attack Pattern	An attack pattern refers to a specific method or tactic used by cyber attackers to gain unauthorized access to a system or network. These patterns can take many forms, such as phishing emails, malware infections or exploiting vulnerabilities in software. They are typically described in terms of the techniques and tools used, as well as the objectives and motives of the attackers.
Campaign	A campaign refers to a coordinated series of malicious activities Orchestrated by an attacker or group of attackers. These activities are often aimed at achieving a specific goal, such as stealing sensitive information, disrupting business operations, or spreading malware.
E-Mail	An E-Mail address present in the CTI-text.
Filepath	A filepath in STIX context refers to the location of a file on a Computer or network. It is used to identify the specific file that is being targeted or used in an attack.
Identity	Identity refers to the identification of an individual or organization that is involved in a cyber threat. This can include information such as names, email addresses, IP addresses, and other identifying details.
Infrastructure	Infrastructure refers to the physical and logical components that support the operation of an organization's information systems. This can include things like servers, routers, switches, and other network devices, as well as the software and applications that run on them.
IP-Adress	The IP-Adress of a server or client computer involved in a cybersecurity event.
Location	A geographical location such as Germany, London or Asia relevant to an CTI-text.
Malware	Malware refers to software specifically designed to harm or exploit computer systems. This can include viruses, worms, trojans, and other malicious programs that can steal sensitive information, disrupt operations, or even cause physical damage.
Hash	A hash value is generated by applying a mathematical algorithm to the contents of a file, resulting in a fixed-length string of characters that can be used to confirm the integrity of the file or to detect if it has been modified.
Registry key	A registry key is a specific entry in the Windows Registry, a hierarchical database that stores configuration settings for the operating system and installed applications. Registry keys can be used by malware to establish persistence on a compromised system and maintain control over it, even after a reboot.
Software	Software refers to the programs and applications that run on computer systems. This can include operating systems, web browsers, and other types of software that are commonly used by organizations.

Table 5.2: *Entity types chosen for this work(Part 2)*

Threat actor	A threat actor refers to an individual or group that is actively trying to exploit vulnerabilities in computer systems or networks. This can include nationstate actors, criminal organizations, hacktivists, and other malicious actors who use malware, phishing, social engineering, and other techniques to compromise systems and steal sensitive information.
Time	In a STIX (Structured Threat Information eXpression) context, time refers to the temporal information associated with a cyber threat event or incident. This can include the date and time when the event occurred, the duration of the event, and the time period during which the threat is considered active or relevant.
Tools	Tools refer to the software or technology used by malicious actors to carry out cyber attacks. A URL (Uniform Resource Locator) is a specific type of identifier used to reference a webpage or other online resource. URLs are commonly used in cyber threat intelligence (CTI) to identify and track the locations of malicious websites and web-based attacks.
Vulnerability	A vulnerability refers to a weakness in a system or software that can be exploited by an attacker to gain unauthorized access or cause damage. These vulnerabilities can exist in operating systems, applications, or even hardware components and are typically tracked using the Common Vulnerabilities and Exposures (CVE) is a standardized system (E.g.CVE- 2013-0640)

5.5.2 Changes to Entity Types

The initial dataset [30] contained separate entity classes for hash algorithms like MD5, SHA1, and SHA2. However, this information is not necessary for the CTI knowledge graph, and it only makes the task harder for the models to learn to distinguish between the different hash types. To simplify the task, the three entity classes were merged into the single class of hash. Additionally, the classes of domain and URL were merged as they represent similar entities. The effects of these changes on the model performance are explained in Section 6.6.1.

5.5.3 Changes to Relations

The initial dataset contained several relations that occur very rarely, which could have a negative impact on the model's performance. To reduce the number of occurring relations and improve the model's performance, two different approaches were evaluated. First, the author consulted with his supervisors to determine which relations could be omitted. Then, one dataset was produced by deleting sentences with dispensable relations. Another dataset was produced by mapping the dispensable relations to the closest remaining relation. The results of these experiments can be found in Section 6.6.2.

5.5.4 Properties of the Final Dataset

Being aware of the distribution of classes in a dataset is important for several reasons. It helps to better understand metrics calculated on it and to develop an understanding

Table 5.3: *Realtion types chosen for this work(Part 1)*

Attributed to	"Attributed to" refers to the identification of the actor or group responsible for a cyber-attack or other malicious activity. This provides valuable information about the origin and motivation of an attack, and can be used to identify patterns of behavior and known tactics, techniques, and procedures (TTPs) used by the actors.
Authored by:	"Authored by" refers to the information about the originator of the malware. This can include information about the individual or group responsible for creating and distributing the malware, as well as any known affiliations or motivations.
Based on	"Based on" describes the relation of an indicator with observed data. This helps analysts to understand the rationale behind a given indicator.
Beacons to	The relation "beacons to" refers to a type of malicious activity where an attacker establishes a communication channel between a compromised system anda command and control server. This communication channel, known as a "beacon", allows the attacker to remotely control the compromised system and exfiltrate sensitive data.
Communicates with	"Communicates with" refers to the relationship between two entities, such as systems or devices, that are able to exchange information or data over a network. This relation is important in identifying patterns of communication between entities and can be used to identify malicious activity.
Compromises	The "Compromises" relation is used to describe that an attacker gained access to a specific system or network component.
Consists of	The "Consists of" relation refers to the relationship between a parent object and its child components. This can include things like a network consisting of various devices or a software package consisting of multiple files.
Controls	"Controls" refers to the state where malware has successfully gained control over an organization's infrastructure, including servers, routers, switches, and other network devices, as well as the software and applications that run on them.
Delivers	The relation "delivers" refers to the process by which malware is transmitted to a target system. This can be achieved through various means, such as email attachments, malicious websites, or infected software downloads.
Downloads	The relation "downloads" refers to the process by which malware infects a system and then proceeds to download additional malware, tools, or files. This is a common tactic used by attackers to gain a foothold on a system and then propagate their malware to other parts of the network.

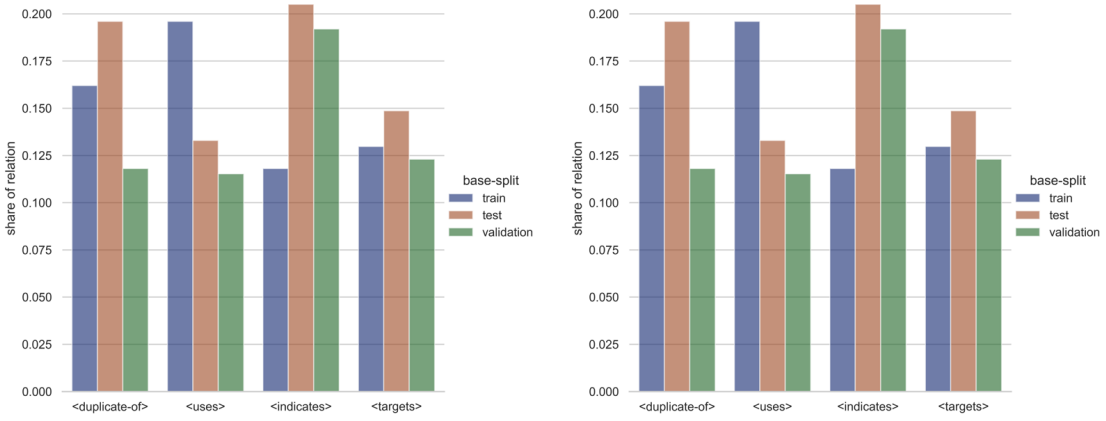
Table 5.4: *Realtion types chosen for this work(Part 2)*

Drops	“Drops” describes the action of malware using a tool, known as a “dropper,” to install and distribute itself on a target system. A dropper is typically a legitimate tool or file that has been compromised or repurposed by an attacker to deliver malware payloads.
Duplicate of	The fact that two entities refer to the same thing is described by the “duplicate of” relationship.
Exfiltrates to	The relation “exfiltrates to” refers to the process by which malware
	steals sensitive information from a compromised computer system and then sends it to a specific infrastructure, such as a command and control server, for further use.
Exploits	“Exploits” describes a connection between a specific type of malware and a vulnerability in infrastructure or software. This type of relation is important in the context of cyber threat intelligence (CTI) as it helps to identify specific ways in which malware can be used to target and exploit vulnerabilities in an organization’s systems.
Has	“Has” refers to the association between a particular infrastructure or tool and a vulnerability it possesses.
Hosts	To describe an environment where infrastructure or a threat actor hosts tools or malware, the relationship “hosts” is used.

Table 5.5: *Overview of CTI-datasets used in this work*

Name	Description	Download
CTI-Base-Split	The initial dataset created by the author and a fellow student.	[30]
CTI-New-Split	Initial dataset split to achieve stratified sampling.	-
CTI-Reduced-Rel	CTI-New-Split with merged relations.	-
CTI-Reduced-Ent Final Dataset	CTI-New-Split with merged entities.	[31]

Figure 5.1: *Comparison of the distribution of the 4 most frequent relation classes*



of the real-world performance of the model. The following section will describe the statistical properties of the final dataset used to train the model.

The final dataset used in this thesis is highly imbalanced with regards to the distribution of both entity classes and relation classes, as seen in Tables 5.7 and 5.6. This is a result of the nature of the underlying CTI articles.

The imbalanced nature of the data leads to an imbalanced performance of the model. While this is not necessarily a bad thing, as a better performance on frequent classes can help the generation of knowledge graphs, it is important to consider that this is only true if the distribution of classes in the input texts is similarly skewed to the distribution of classes in the training data.

Table 5.6: *Absolute and relative frequencys of relations in the CTI-dataset*

Relation \ Split	train	test	validation	total
targets	105 (13,7%)	18 (12,7%)	22 (11,8%)	145
uses	141 (18,4%)	23 (16,2%)	28 (15,1%)	192
beacons-to	7 (0.9%)	1 (0,7%)	2 (1,1%)	10
duplicate-of	121 (15.8%)	22 (15,5%)	38 (20,4%)	181
related-to	70 (9,2%)	14 (8,5%)	12 (6,5%)	96
communicates-with	9 (1,2%)	2 (1,4%)	2 (1,1%)	13
delivers	8 (1%)	2 (1,4%)	3 (1,6%)	13
located-at	50 (6.5%)	13 (9,2%)	13 (7%)	76
attributed-to	17 (2,2%)	3 (2,1%)	3 (1,6%)	23
exploits	24 (3,1%)	5 (3,5%)	6 (3,2%)	35
impersonates	4 (0,5%)	1 (0,7%)	1 (1,6%)	6
downloads	12 (1,6%)	3 (2,1%)	2 (1,1%)	17
exfiltrates-to	3 (0.3%)	0	1 (0,7%)	4
indicates	109 (14,2%)	18 (12,7%)	29 (15,6%)	156
has	46 (6%)	7 (4,9%)	9 (6,3%)	62
variant-of	10 (1,3%)	2(1,4%)	2 (1,1%)	14
authored-by	11 (1,4%)	5 (3,5%)	7 (3,8%)	23
consists-of	7 (0,9%)	1 (0,7%)	2 (1,1%)	10
controls	1 (0,1%)	0	1 (1,6%)	2
drops	2 (0,3%)	1 (0,7%)	1 (1,6%)	4
hosts	5 (0,7%)	1 (0,7%)	2 (1,1%)	8
total	762	142	186	1090

Table 5.7: *Absolute and relative frequencys of entities in the CTI-dataset*

Entity \ Split	train	test	validation	total
attack-pattern	113 (8,4%)	21 (8,3%)	33 (9,8%)	167
url	26 (1,9%)	3 (1,2%)	4 (1,2%)	33
identity	135 (10%)	28 (11,0%)	45 (13,3%)	208
software	110 (8,1%)	17 (6,7%)	18 (5,3%)	145
malware	291 (21,5%)	58 (22,8%)	71 (21%)	420
infrastucture	16 (1,2%)	3 (1,2%)	3 (0,9%)	22
threat-actor	150 (11,1%)	35 (13,8%)	34 (10,1%)	219
time	47 (3,5%)	13 (5,1%)	12 (3,6%)	72
IP-address	10 (0,7%)	0	0	10
location	101 (7,5%)	15 (5,9%)	36 (10,7%)	152
tools	62 (4,6%)	12 (4,7%)	14 (4,1%)	88
vulnerability	91 (6,7%)	18 (7,1%)	17 (5%)	126
campaign	32 (2,4%)	6 (2,4%)	8 (2,4%)	46
hash	79 (5,8%)	9 (3,5%)	21 (6,2%)	109
filepath	87 (6,4%)	16 (4,3%)	22 (6,5%)	125
registrykey	2 (0,1%)	0	0	2
email	1 (0,1%)	0	0	1
total	1353	254	338	1945

6 Experiments

This Chapter describes experiments carried out as part of this work. First several models are explored followed by experiments focused on enhancing the most promising model.

6.1 Closed source language models

To create a baseline for the following experiments the capabilities of models discussed in Section 2.3 were analysed regarding their performance on CTI texts. The text used to exemplary test these models was taken from a MISP training website [51].

The first model evaluated in this work was Luminous, developed by Heidelberg start-up Aleph Alpha. Upon initial testing in the "playground" provided by Aleph Alpha, it was found that no useful output could be produced. However, the question answering module did produce somewhat sensible output when asked "What are relevant entities and their relationships in this text?". Despite this, it was ultimately determined that Luminous was not suitable for the task of relation extraction in the CTI domain.

Chat-GPT was released on November 30, 2022, and it represents a significant change compared to previous models due to its use of large language models combined with reinforcement learning with human feedback (RLHF) [39, 56].

The results of Chat-GPT were impressive, considering that it is a general-purpose model. When given the input text and prompt "Extract entities and their relationships from this text as triplets (head, relationship, tail):", Chat-GPT produced the following output:

- (CEO, received an email from, john.doe@luxembourg.edu)
- (john.doe@luxembourg.edu, sent an email from, throwaway-email-provider.com)
- (email, contained a malicious file, attached file)
- (malicious file, tried to download a secondary payload from, https://evilprovider.com /this-is-not-malicious.exe)
- (secondary payload, has a hardcoded C2 at, https://another.evil.provider.com:57666)

Usability of closed source language models The evaluations previously discussed show that closed source language models perform better than comparable publicly available models like Google's T5 model [59]. The main reason for this is the large amounts of money that firms invest in labeling data and staffing. However, there are also downsides to using closed source language models. One of the main downsides is that in order to be able to sell the model to many people, the models tend to remain general and not specialized to a specific use case.

Despite these downsides, using closed source language models can still be a viable option depending on the use case. For example, closed source models may have higher

inference costs but lower development costs, making them more suitable for systems which are not used often or on small amounts of data.

6.2 DeepEx

The concept of extracting relational triplets without the need for annotated data is of interest to this work. After reading the paper [69], the corresponding code was downloaded from Github and the scripts for the experiments described in the paper were rerun and debugged. The scores reported by the authors could be replicated.

However, when analyzing how the scores were calculated, the author noticed that the model uses long spans of the input text as relations of the triplets (see Table 6.1). These answers are still considered correct due to the nature of the scoring function used in DeepEx.

However, this makes the model not suitable for constructing a CTI-Knowledge-Graph for this work, as it requires an ontology similar to the one defined by MISP. Therefore, we will not use DeepEx in our approach, and we will focus on other models that better suit the requirements of our task.

Table 6.1: *Triplets produced by DeepEx compared to the Gold Answer provided by the authors [69]*

Text	Gold-Answer	Model-Output
Cathleen Colella , the owner of the Hazardous Elimination Corporation , a hazardous substance remediation company based in Farmingdale , N.Y. , agreed .	based Hazardous Elimination Corporation Farmingdale	“Cathleen Colella” “owner of the Hazardous Elimination Corporation , a hazardous substance remediation company based in Farmingdale ,” “N.Y.”
What fun , what with the costumes , the music , the pageantry , the pride , starting last evening with Italian night as the ethnically diverse Mets met the equally diverse Dodgers .	met Mets Dodgers	“the equally diverse Dodgers” “, the pageantry , the pride , starting last evening with Italian night as the ethnically” “the music”

6.3 REBEL

The REBEL model [32] has shown to produce state-of-the-art results in relation extraction tasks. One of the major advantages of REBEL is that the code is easily understandable and adoptable. In this work, REBEL was thoroughly analyzed in a series of experiments.

The code provided by the authors already contained all the necessary scoring and monitoring functionality, making it easy to evaluate the model's performance. The only required modification was the adoption of the data loading modules to the CTI-dataset.

6.3.1 Baseline Experiments

Before further investigating the performance of the REBEL model on the CTI-dataset, a series of baseline experiments were conducted to verify the scores reported by the authors and compare the model to other publications.

In these experiments, the scores reported by the authors for the New York Times (NYT) and DocRed datasets could be replicated. These results provide a strong indication that REBEL is a reliable and effective model for relation extraction tasks.

The experiments also highlight the effect of the pretraining on the Wikipedia dataset, as can be seen in Figure 6.1. This figure compares the performance of REBEL to the BART model over the course of the training process, demonstrating the benefits of the extensive pretraining on the Wikipedia dataset used in REBEL.

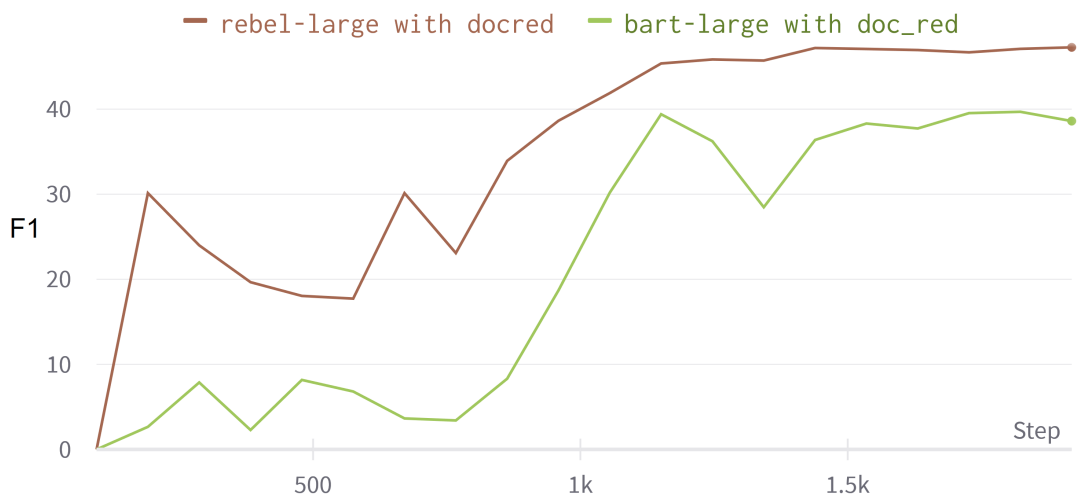


Figure 6.1: *Comparison of the rebel-large and bart-large models on the DocRED validation dataset. It highlights the effectiveness of pretraining on the REBEL dataset done with the rebel model.*

6.3.2 Comparison to other models

To further analyze the performance of REBEL, an experiment was conducted to compare it to the model proposed by Tan et al. [66] which uses a special token "*" to mark entities, as described in subsection 2.7.1. This modification makes the task easier for the model.

To conduct this experiment, the DocRed dataset was modified by adding the “*” token to mark entities and the experiments were rerun. The results showed that the model started off with a higher performance and reached slightly better overall performance figures, as can be seen in Figures 6.2 and 6.3. However, this does not explain all the differences to the model proposed by Tan et al. [66]. Additional performance gains can be attributed to the use of contextual information in their model.

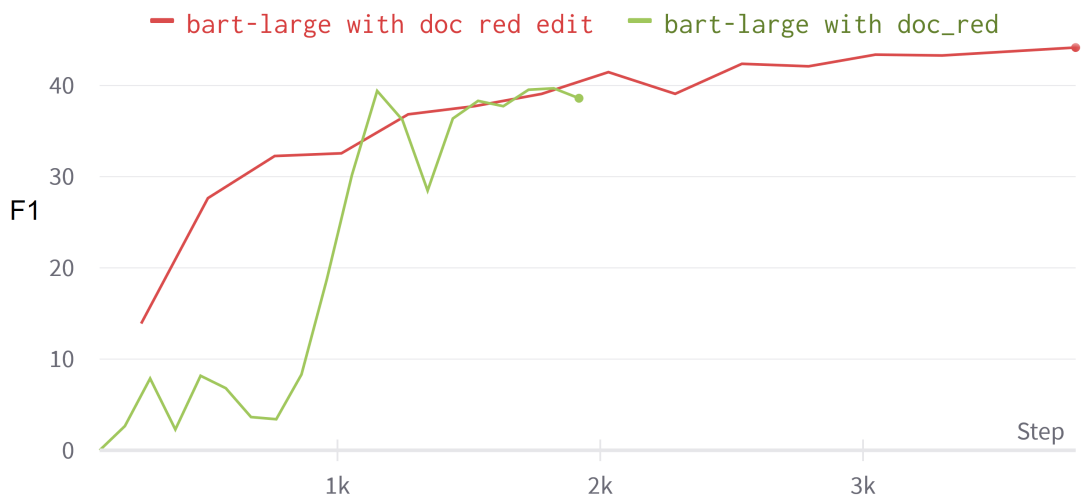


Figure 6.2: *Effect of adding an entity marking token to REBEL(bart-large). The extra token helps the model to learn the task faster and achieve slightly better results.*

6.3.3 Experiments on CTI-dataset

After verifying the performance of the REBEL model and understanding the differences with other implementations, experiments on the CTI-dataset were carried out. Two experiments were conducted:

1. Adopting the CTI-Dataset to the REBEL format: In this experiment, the CTI-dataset was adapted to the format used by REBEL. However, information about classes of entities was lost as it is not part of the REBEL format.
2. Creation of a new dataloader where entity types are considered similar to the NYT dataset: In this experiment, a new dataloader was created that takes into account entity types, similar to the format used in the NYT dataset.

Both experiments used the REBEL-large model, which was first pretrained on the DocRed dataset. The results of these experiments are provided in table 6.2. The results for the REBEL format is better, as the task of just predicting the relations is easier than predicting the entity types as well.

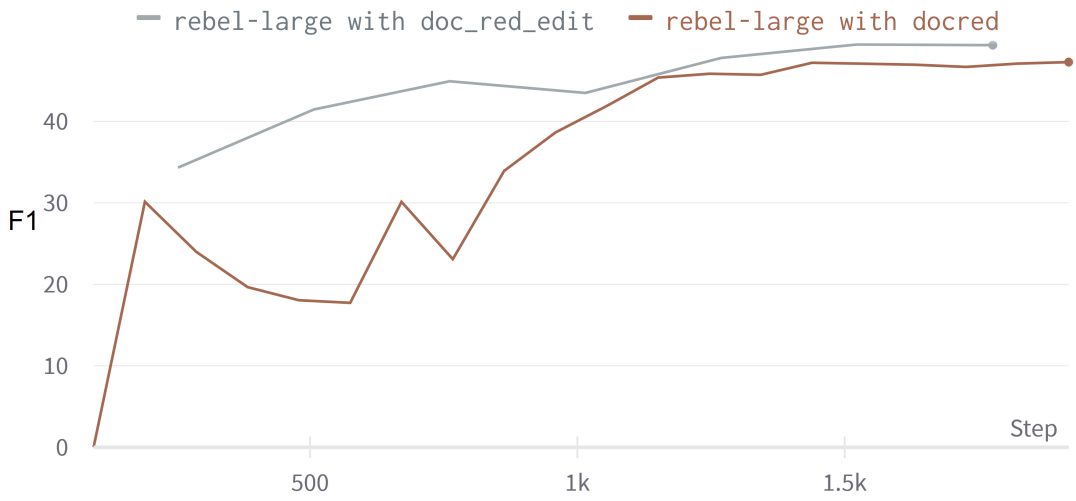


Figure 6.3: *Effect of adding an entity marking token to REBEL(rebel-large). With the extra token additionaly to the pretraining the model can identify relations from the beginning and achieve slightly better results than rebel-large without the token.*

Dataset format	F1-Score
REBEL	0.33
NYT	0.23

Table 6.2: *Results of experiments with the CTI dataset using the REBEL model*

6.4 Evaluation of previous studie

The thesis of a fellow student, Alex Schwankner [21], had a similar goal to this work and was therefore studied carefully. Both works use the same dataset, which allowed some code to be reused. However, the results reported in Schwankner's thesis were not sufficient to build a CTI-graph (Chapter 2.7.4). To determine if better performance could be attained, several experiments were conducted. By monitoring train and validation loss as well as validation precision, recall and F1-score, it was noticed that Schwankner trained the model too short. By training the model longer, an F1-score of 26.9 was achieved on the validation set.

After achieving a decent baseline performance, the dataset provided by Schwankner [21] was modified as described in Chapter 5.5.1. This modification resulted in an improved performance, with an F1-score of 28.4, which will be taken as the baseline for further experiments carried out in this work (see Table 6.3).
Performance metrics in Sections 6.4 and 6.5 are calculated using the same lenient function as used by the author of [21] to keep results comparable. In this lenient evaluation approach the Levenshtein [41] distance was used to calculate metrics. In Chapter 7 both strict and lenient metrics will be calculated and compared.

Table 6.3: *Performance of the NLP-model on the validation set*

Dataset	Precision	Recall	F1
CTI-Base-Split	0.281	0.25.8	0.269
CTI-New-Split	0.272	0.297	0.284

6.5 Pre-Processing of CTI-Texts

As the pre-processing of CTI texts has been shown to be effective, a similar approach to that presented in 2.8 was used in this work. In addition to the regular expressions used by [22, 23, 33], this work also used word lists to identify words that cannot be found by regular expressions but can be scraped from databases on the internet. The categories of regular expressions used in this work are listed in the Table 6.4, the corresponding expression can be found in the Appendix A.1.

Table 6.4: *Regular expressions used to preprocess CTI-texts*

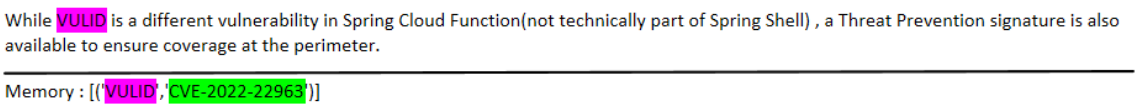
Regular Expression	**Description**
APT	Advanced Persistent Threat
DOM	Domain Name
IPV4	Internet Protocol version 4
IPV6	Internet Protocol version 6
EMAIL	Email addresses
URL	Uniform Resource Locator
VULID	Vulnerability Identifier
SHA2	Secure Hash Algorithm 2
SHA1	Secure Hash Algorithm 1
MD5	Message-Digest Algorithm 5
REGISTRYKEY	Windows Registry key

Wordlists were scraped from the internet using Python scripts from websites such as MITRE ATT&CK [53] or Malpedia [45]. The categories of wordlists used to preprocess the texts are shown in the table 6.5, a full version of the lists and scripts to create them are available in the GitHub repository [25].

Table 6.5: *Wordlists used to preprocess CTI-texts*

attack-pattern
identity
malware
threat-actor
tool

Figure 6.4: *Visualisation of the substitution process*



6.5.1 Technical Implementation

The technical implementation of integrating the information present in the regular expressions and wordlists into the Relation Extraction process involved developing a set of Python functions. These functions were designed to be easily added into the process at training or inference time. Python's built-in regular expression library and spaCy were utilized to develop the necessary functionality. The developed functions are capable of identifying special words present in the defined wordlists and regular expressions and substituting them with the name of their class. The information about which words have been substituted is stored, so that it can be resubstituted after the processing by the NLP model is done, ensuring that no information is lost. A visualization of the process can be seen in Figure 6.4.

6.5.2 Effect on Models Performance

In order to analyse the effect of different preprocessing techniques, a series of experiments were conducted. In each experiment, only one category of wordlist or regular expression was used to preprocess the CTI texts. The results of each experiment can be found in table 6.6.

It is important to note that the results are heavily influenced by the number of occurrences of each class in the relatively small dataset. The results are therefore not representative.

Having established the performance of each individual class, combinations of different classes were tried. Interestingly, combining all the classes did not give the best results. The best combination was found to be a combination of the regular expressions and the word list classes malware, attack pattern and tool with an F1 score of 0.362. This is an significant improvement over the base-models performance of 0.276.

6.6 Experiments with Datasets

Several changes were made to the dataset to improve the model performance as explained in Section 5.5. The effects of these changes will be explained in this section.

6.6.1 Changes to Entity Types

Merging of similar entity types improved the model's performance from 0.284 F1-score to 0.311, as shown in 6.7. This is a significant improvement, especially since the

Table 6.6: *Performance of the NLP-model depending the on pre-processing technique*

Class	Model F1-score
Regular expressions	0.352
Malware	0.346
Attack-pattern	0.309
Tool	0.305
Threat-actor	0.294
Identity	0.292
Selected	0.362
Base-model	0.284

Table 6.7: *Effect of Changes to the dataset to the performance of the NLP-model on the validation set*

Dataset	Precision	Recall	F1
CTI-New-Split	27.2	29.7	28.4
Mapped-Entities	30.2	32.6	31.3
Mapped-Relations	24.2	28.5	26.2
Mapped-Relations(delete)	27.8	24.6	26.0

changes only affected a small subset of all entities. This shows that the model had difficulties distinguishing between hash-types and domains from URLs.

6.6.2 Changes to Relations

Both approaches to reduce the relations did not improve the model's performance. Deleting sentences with dispensable relations led to a slightly worse performance than mapping them to other relations. Both approaches deteriorated the performance , as shown in Table 6.7.

Keeping in mind that deleting sentences reduced the dataset size from 480 to 334, this approach might have worked if more training data was available. However, mapping the relations proved to be impractical, likely because the mapping labeled different means with the same relation, confusing the NLP model.

7 Results and Evaluation

The experiments conducted in the previous chapter formed the foundation of the model's development, and this chapter focuses on presenting the outcomes of those experiments. In this chapter, we will examine the generated graphs and JSON outputs, showcasing the model's performance on applied examples. We will also evaluate the errors that were encountered during the experimentation process and analyse the attention mechanism used in the model.

7.1 Architecture of the final model

The final model developed in this work represents the culmination of a series of experiments and evaluations as described in Chapter 6. The best-performing combination of model, pipeline, and dataset was selected (System overview in Figure 7.1). As the base model, the T5 model developed by Google and pre-trained on the CTI corpus dataset, as well as the NYT dataset, was chosen (as described in Chapter 2.7.4). This base model, trained on the CTI dataset, performed better than the REBEL model on the CTI data. For preprocessing, regular expressions and the word list classes malware, attack pattern, and tool were employed, as detailed in Chapter 6.5. This architecture was trained on the dataset with mapped entity types, which showed the best results in Chapter 6.6.1. Finally, both the lenient Levenshtein algorithm-based and the strict metrics were calculated on all three parts of the dataset, as shown in Table 7.1.

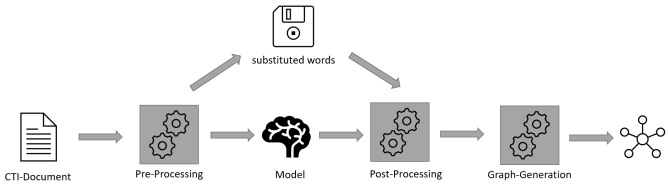


Figure 7.1: *Graphical system overview*

Table 7.1: *Metrics for the final model calculated using the Levenshtein metrics as well as strict metrics*

	Precision	Recall	F1-Score
Lev:train	0.9624	0.9652	0.9638
Lev:test	0.3731	0.4463	0.4064
Lev:val	0.3473	0.3831	0.3643
Strict:train	0.9583	0.9615	0.9599
Strict:test	0.3557	0.4290	0.3889
Strict:val	0.3447	0.3700	0.3569

Table 7.2: *Results of the four cross validation runs(strict evaluation)*

Run	Precision	Recall	F1
0	0.4182	0.3809	0.4003
1	0.3100	0.2966	0.3031
2	0.3978	0.3837	0.3906
3	0.3827	0.4290	0.4045

Table 7.3: *Mean and standard deviation of the computed metrics*

	Precision	Recall	F1
mean	0.3772	0.37256	0.3746
std	0.0471	0.05522	0.0480

7.2 Cross Validation

As explained in Section 2.6.2, calculating metrics on small datasets is not optimal as the scores will be highly dependent on the dataset split. Therefore, the final model was evaluated using 4-fold cross validation. The dataset was randomly split into 4 folds, and the model was trained and evaluated 4 times, each time using a different fold as the test set and the remaining 3 folds as the training set. The F1 score was used as the evaluation metric. The average F1 score over the 4 runs was 0.3746 with a standard deviation of 0.048 (see Table 7.3). The results of all runs can be seen in Table 7.2 and show a relatively low standard deviation. This can be attributed to the balance of relations across the three split datasets.

7.3 Evaluation of the attention mechanism

Deep learning models are often considered as black boxes that map input to output without providing insight into the internal workings of the model. However, researchers have developed tools to help explain some of the inner workings of these models.

Vig et al. [68] developed a tool to visualize the attention mechanism of transformers, which was described in Section 2.2. The Bertviz library is used to show which other tokens the model “attends” to when processing the inputs. This tool was used to visualize the model’s understanding of of the CTI texts. An example input was analyzed and the results are shown in Figure 7.2. This visualization helped to provide insight into how the model was processing the input and which concepts it was focusing on.

By visualizing the models’ attention(compare Figure 7.2) analysis showed how the model understands basic cybersecurity concepts. The first two examples show the self-attention of the encoder and the cross-attention between input and output, while the third example shows how the model generates the triplets based on the input.

The all three examples highlighted the model’s attention to punctuation and special tokens for structural information about the sentence. In the first example, the attention was focused on the connections between “security flaws”, “trojanised documents” and

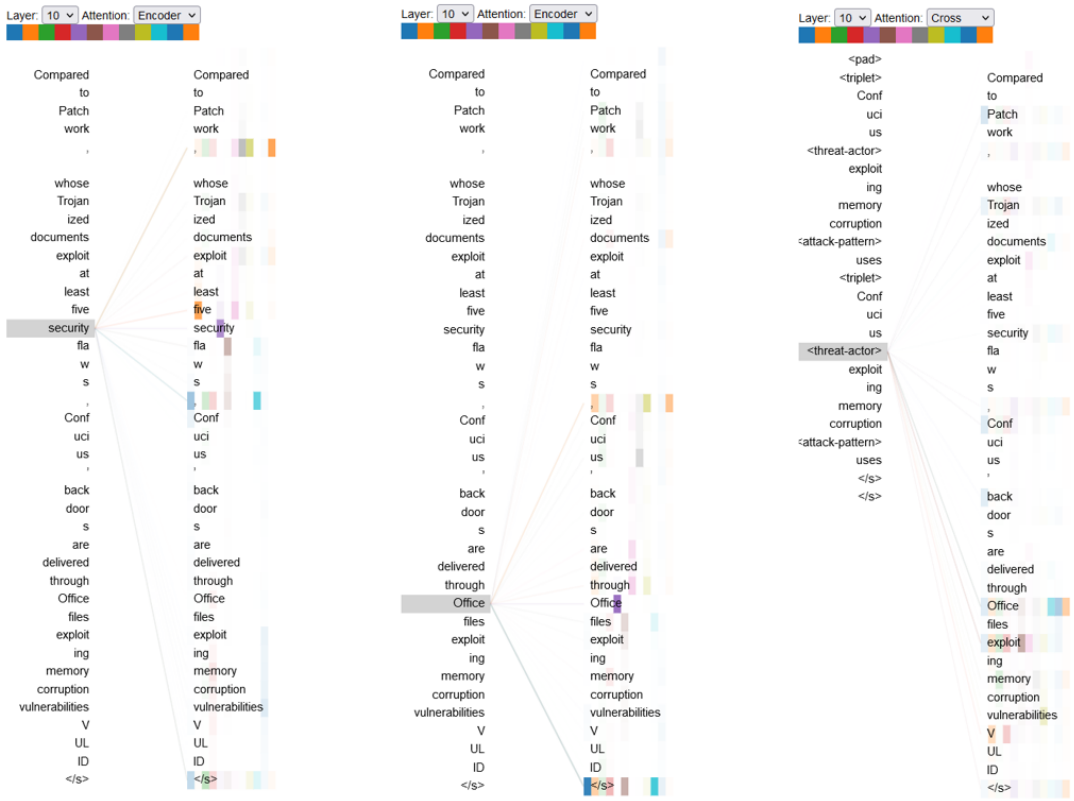


Figure 7.2: *Example visualizations of the attention mechanism. The left and middle example show encoder self attention while the right example visualized cross attention between input and output of the model.*

"memory corruption vulnerabilities". While in the second example, attention was drawn to the links between "Office files", the associated threat actor "Confucius" and "delivered by". The third example shows the links between the input on the right and the generated triplets on the left. Here it can be seen that the model classified "Confucius" as a threat actor based on the tokens "Office files exploiting memory corruption vulnerabilities".

7.4 Graph Generation System

The developed NLP model cannot be used standalone and requires a program to interact with CTI texts and graph libraries. This section describes the different components and their functionality.

The system can be implemented using most programming languages that support a machine learning inference framework like ONNX [15]. As there are no requirements in the scope of this thesis regarding the choice of programming language, Python was chosen due to its ease of use with deep learning models and the availability of powerful graph libraries.

7.4.1 Text Processing Component

The system takes TXT files as input(Compare Listing 7.1), while parsing texts from websites or PDFs is not part of this work. The article text was first split into sentences using the Python library spaCy [29], as also described in Section 5.3.1. The resulting sentences were then passed through the pre- pipeline and NLP model to extract the relevant relations, resulting in a list of relations for each article. This list can either be exported as a JSON file (see Listing 7.2) or further processed by the graph component.

7.4.2 Graph Generation Component

The graphs should be similar to the graphs integrated in the MISP tool (see Figure 7.3). The NetworkX library [27] was chosen to generate the graphs, as it provides all the functionality for network graphs with labeled edges. Subjects and objects of the relation were added as nodes to the graph, while the class of the relation was used as the edge label. Relations with the same head and tail entity were discarded as they rarely made any sense. The generated graphs (see Figure 7.4) are exported as PNG files.

```
LokiBot also known as Lokibot, Loki PWS, and Loki-bot employs Trojan malware to
steal sensitive information such as usernames, passwords, cryptocurrency wallets,
and other credentials.
The malware steals credentials through the use of a keylogger to monitor browser and
desktop activity (Credentials from Password Stores [T1555]).
(Credentials from Password Stores: Credentials from Web Browsers [T1555.003])
(Input Capture: Keylogging [T1056.001])
LokiBot can also create a backdoor into infected systems to allow an attacker to
install additional payloads (Event Triggered Execution: Accessibility Features).
```

Listing 7.1: *Example of a CTI input text*

```
[{'head': {'label': '<malware>', 'text': 'Lokibot'},
  'subj': {'label': '<malware>', 'text': 'Trojan malware'},
  'rel': 'duplicate-of'},
 {'head': {'label': '<malware>', 'text': 'Lokibot'},
  'subj': {'label': '<attack-pattern>', 'text': 'steal sensitive information'},
  'rel': 'uses'},
 {'head': {'label': '<malware>', 'text': 'malware'},
  'subj': {'label': '<attack-pattern>', 'text': 'steals credentials'},
  'rel': 'uses'},
 {'head': {'label': '<malware>', 'text': 'malware'},
  'subj': {'label': '<attack-pattern>',
   'text': 'monitor browser and desktop activity'},
  'rel': 'uses'},
 {'head': {'label': '<malware>', 'text': 'LokiBot'},
  'subj': {'label': '<attack-pattern>', 'text': 'Input Capture'},
  'rel': 'uses'},
 {'head': {'label': '<malware>', 'text': 'LokiBot'},
  'subj': {'label': '<attack-pattern>', 'text': 'infected systems'},
  'rel': 'uses'}]
```

Listing 7.2: *Example of relations in JSON format*

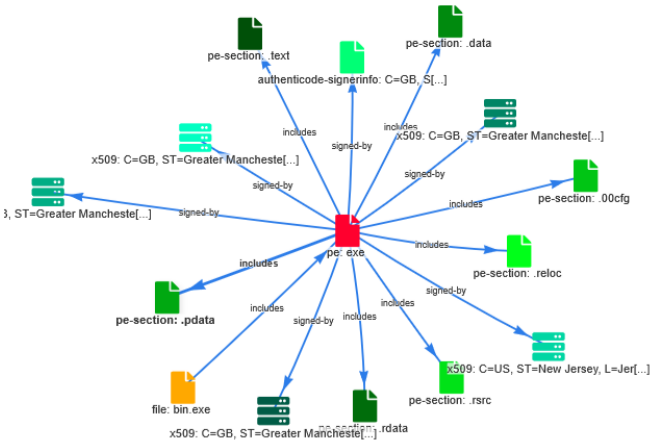


Figure 7.3: *Graph visualisation in the MISP tool*

7.5 Interpretation and Error Analysis

While for some applications of machine learning, for example regression or image classification metrics can be calculated easily, this is not the case for NLP especially relation extraction. The sentence “The centre point of the sun has a distance of X from the centre point of the earth.” serves as a good example how ambiguous relation extraction can be. While the obvious relation is: centre point of the sun <distance to earth> X many other relations could be extracted from this sentence. For example: sun <has> centre point. This is especially true for generating CTI-knowledge graphs as here the real task is extracting the relevant relations. These relevant relations are defined by the annotator when creating the dataset.

Of course, metrics still need to be calculated, else comparing different approaches would be very difficult. When interpreting these metrics, it should be kept in mind that unlike in other fields, the labeled truth is not necessarily the only truth.

To get a better feeling for the quality of the results produced by the final model, 21 relations classified as wrong were evaluated again and rated as:

- **Good error**: The model did not produce the labeled relation but the generated output was sensible.
- **Bad error**: The generated output did not make sense.

Examples of good and bad errors can be found in Table 7.4, and the complete comparison of generated output and expected output is available in Appendix A.3. Out of the 21 relations, 13 relations made sense, while just not being the relation expected from the dataset. 8 relations did not provide factual information.

In light of the manual error analysis, it appears that the sentence-level relation extraction model developed in this work may perform better than the measured F1 score of 0.3746 suggests, possibly achieving an F1 score of around 0.6. While metrics such as F1 score are useful for comparing different training and preprocessing

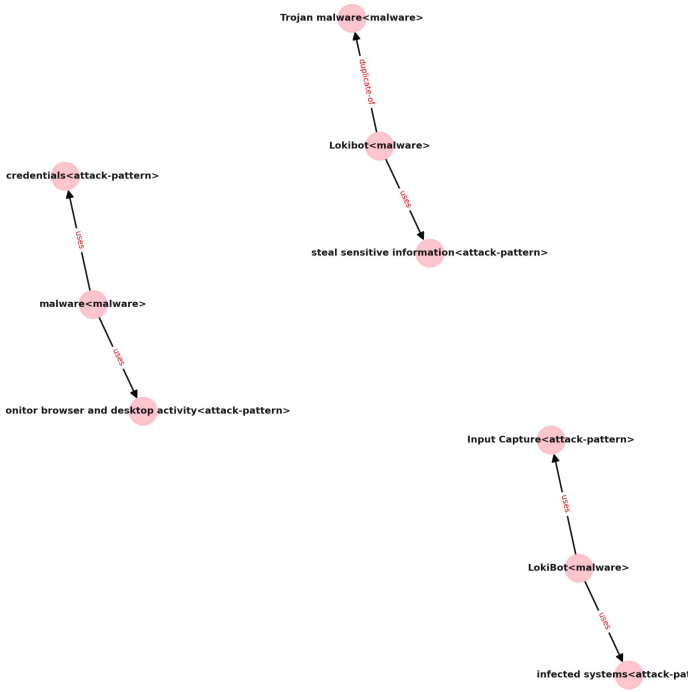


Figure 7.4: *Graph visualisation with NetworkX*

techniques, they only serve as an indicator of the system's real-world performance. The generated graph shown in Section 7.4, as well as more examples in Appendix A.2, demonstrate that the system's overall performance is far from perfect. This limitation can be attributed to the fact that the model was only trained on sentences that contain relations. In a setting where all sentences of a text are processed by the model, relation are identified where there are none, as the model is biased toward finding at least one relation.

Table 7.4: *Example of "good" and "bad" errors*

Example of "good errors"	
Expected output:	<triplet>Apache Struts versions 2.3 to 2.3.34 and 2.5 to 2.5.16<software>VULID<vulnerability>has
Model output:	<pad><triplet>Apache Struts<software>VULID<vulnerability>has
Example of "bad errors"	
Expected output:	<triplet>Conti<threat-actor>U.S.<location>targets
Model output:	<pad><triplet>Conti<threat-actor>U.S.<location>located-at</s>

8 Discussion

This chapter presents the discussion of the research findings that address the main aim of the study, which was to determine the effectiveness of extracting knowledge from MISP event reports. To achieve this aim, the study was further divided into three research questions that focused on identifying publicly available datasets for training relation extraction models in the CTI domain, determining the best model architecture for extracting relations from CTI texts, and evaluating the performance of NLP models for generating event graphs from MISP event reports using different pre-processing pipelines.

The study found that while many datasets exist for training NLP models for the task of named entity recognition, there are no publicly available relation extraction datasets in the domain of cyber threat intelligence. Consequently, the study developed an own dataset based on reports scraped from the internet. Additionally, the study found that framing relation extraction as a sequence-to-sequence problem using transformer models proved to be an effective way of extracting relations from both general texts and CTI-texts. Finally, the study developed a pre-processing pipeline that improved the model's performance by 27%.

8.1 Availability of relation extraction datasets in the CTI-domain

An important question and pre-condition for the whole thesis was to determine whether publicly available datasets exist for training relation extraction models in the CTI domain. To answer this question, a thorough analysis of available datasets was conducted, which led to the same conclusion as Liu et al. [44] that no publicly available relation extraction datasets are available in the CTI domain. Although the MalKG provides relational triplets, they lack context, making them more suitable for relation classification tasks rather than relation extraction.

To overcome this limitation and train a relation extraction model in the CTI domain, the study developed its own dataset. As labeling data is a laborious task, only a relatively small dataset of 9732 sentences was created, out of which only 480 contained relations. Despite its limited size and annotation by only two annotators, the dataset was sufficient to adopt a relation extraction model to the CTI domain. This was possible because the general domain NYT dataset was used to teach the model the concept of relation extraction before being applied to the CTI domain.

The development of the own dataset for relation extraction in the CTI domain provides a significant contribution to the field, as it addresses the lack of publicly available datasets in this area. However, further research is required to develop larger datasets that include more varied types of CTI-texts and to improve the quality and consistency of the annotations.

8.2 Choice of Model Architecture

The first research question aimed to identify the best model architecture for extracting relations from CTI texts. Throughout this thesis, various approaches to extracting knowledge from CTI texts were evaluated. Experiments showed that closed-source general models like Chat-GPT [56] have the potential to extract knowledge from CTI texts. However, due to restricted access to such models, they were not further analyzed.

Different types of open-source language models were analyzed, including the zero-shot Open Information Extraction model DeepEx [69]. It was found to be unsuitable for extracting relations for CTI knowledge graphs due to its unstructured output. On the other hand, the sequence-to-sequence relation extraction approaches by Cabot et al. [32] and Schwankner [21] both worked well on both general relation extraction datasets like NYT as well as the CTI-dataset created as part of this work. This finding highlights the effectiveness of adopting pre-trained large language models like BART or T5 to downstream tasks.

Furthermore, the effectiveness of first training on a larger task-specific dataset like NYT and later adapting it to a specific domain using a smaller dataset was demonstrated. This approach can significantly improve the performance of relation extraction models in the CTI domain, where labeled datasets are limited.

Overall, the findings of this study provide valuable insights into the best model architectures for relation extraction in the CTI domain. These insights can guide future research in developing more effective models for extracting relations from CTI texts.

8.3 Effect of pre-processing CTI-texts

The second research question aimed to investigate how the performance of NLP models for generating event graphs from MISP event reports could be improved with different pre-processing pipelines and to identify the most effective pipeline for handling the nuances of CTI-texts. Extracting knowledge from CTI-texts is challenging as these texts possess nuances such as IP-addresses or hash-values, which may confuse NLP modules [22, 23, 33]. Previous research [22, 23, 33] has used pre-processing of texts with regular expressions to address this issue. In this study, pre-processing with regular expressions was also used and validated as an effective approach. Moreover, this work proposed the use of wordlists containing names of CTI-related entities in addition to regular expressions. The use of wordlists further improved the performance of NLP models. Increasing Interestingly, the positive effects of merging entity types such as different hashes were smaller than in the comparison of models without the pipeline. This could be due to the fact that the model does not need to learn to distinguish between them if they are labeled by the pre-processing pipeline. Overall, the findings suggest that using pre-processing pipelines, particularly with the use of regular expressions and wordlists, can improve the performance of NLP models(in the case of this work by up to 27%) for generating event graphs from MISP event reports,

and that careful consideration of the nuances of CTI-texts is essential when developing these pipelines.

8.4 Limitations

In discussing the limitations of this work, it is important to note that while a new, functional approach for extracting relations from CTI texts has been developed, there are still some limitations. Firstly, the models used in this thesis are derived from the base version of the T5 model, as bigger versions of the model could not be used due to hardware limitations. The use of bigger models could potentially lead to increased performance, particularly if a larger dataset was available. However, by demonstrating how Relation Extraction can be performed with the base model, this work enables researchers with better hardware to reproduce the experiments of this work with bigger models.

Secondly, the dataset used in this thesis is not perfect as it had to be annotated before any training could be performed. Findings about the dataset that were achieved during later stages of the thesis would have required redoing the laborious task of annotating texts, which was not possible due to the scope of this thesis. Despite this limitation, the dataset used in this work was still sufficient to achieve the research goals.

Thirdly, the graphs drawn by the graph module are just static images. Displaying the data on an interactive website would provide a better user experience. However, the user experience was not a focus of this work, and therefore the static images were deemed sufficient for the research objectives.

Overall, despite the limitations, the findings of this work demonstrate the potential for improving the extraction of relations from CTI texts and provide a foundation for future research in this area.

8.5 Practical Applications

The practical applications of the developed model in this work have been explored, and several potential use cases have been identified. Despite the fact that the model is capable of extracting sensible relations from CTI texts, its performance is still limited, with an F1 score of 0.37. This means that the extracted relations cannot provide a full representation of the analyzed report, and this limitation should be kept in mind when using the model.

One potential use case of the model is to summarize a report for an analyst. By using the extracted relations, a summary of the report can be created, which can be useful for analysts who want to get an overview of the report quickly.

Another potential use case is to create a template graph that can be improved manually by adding relations that the model did not find. This could help analysts to create graphs faster.

The model could also be used to fill databases with relations about which malware exploits which CVE or which identities are targeted by hacker groups. This information can be used to inform decision-making processes and improve the overall security posture of an organization.

Finally, it should be noted that the model could be easily improved by labeling additional data and retraining the model. Therefore, organizations that plan on using this model could improve its performance by labeling more data that is relevant to their specific use case and retraining the model. This would improve the accuracy of the model and make it more useful for practical applications.

8.6 Future Work

The limitations discussed in the previous section provide opportunities for future work in improving the dataset and the model used in this thesis. Concrete examples of how they could be improved are provided in this section.

Dataset In terms of future work, there are several opportunities for improvement in the dataset used in this thesis. The evaluations performed in this thesis showed that the limited dataset available was a clear limitation of the model's performance. Therefore, future work could focus on improving the dataset in several ways.

Firstly, a dataset containing cross-sentence relations similar to the one used in DocRed [77] could be created. This would enable the model to find such relations in reports and improve its overall performance. Additionally, more relations could be added to the dataset to increase its size and diversity.

Secondly, the selection of training texts could be improved to better represent the distribution of input texts. This could involve using a more diverse set of reports or selecting reports that are more representative of the types of reports that the model is intended to analyze.

Thirdly, an additional field could be added to the dataset that provides the whole report that the sentence was taken from as context. This would enable the model to better understand the overall context of the report and improve its ability to extract relations.

Lastly, a two-stage labeling process with multiple annotators could be implemented, where labeled relations need to be confirmed by another annotator. This would ensure that the dataset is of high quality and that the labeled relations are accurate. This way, a more stringent dataset could be produced, which would greatly improve the performance of NLP models trained on this dataset.

Model As one core limitation of this work is the processing of single sentences, processing documents as a whole promises significant room for improvement. Various techniques to provide the model with more context for the task of relation extraction are conceivable. These techniques were not evaluated as part of this thesis due to the big engineering effort required to incorporate. Current state-of-the-art models

such as BERT [16] or T5 [59] normally can process 512 tokens, which corresponds to roughly 400 words. Therefore, providing additional context can be split into two categories. Using the same model as in this work, up to 512 tokens can be used as input. A length of about 400 words should be sufficient to provide the model with one paragraph of text from the report. Experiments in this work showed this size to be a sensible choice to generate graphs. If longer texts are necessary as input, specialized model architectures need to be used. Different approaches like a sliding window [72] or mechanisms selecting a smaller subset of relevant contexts to feed in the transformer [34, 40] have been researched. The most prominent work in this field is Big Bird [78] by Google, where a sparse attention matrix is used. All these approaches make the task significantly harder to develop.

9 Conclusion

The concluding chapter of this study will provide a summary of the research findings that are relevant to the research aim of extracting knowledge from MISP event reports. Additionally the value and contribution of this work will be discussed while limitation, practical applications and areas of future research have been discussed in the previous chapter.

This work aimed to address the problem of automatically extracting relevant knowledge from MISP event reports using deep learning models. By doing so, it aims to reduce the workload of security analysts. Through this study, it was analyzed whether suitable datasets exist to train models to extract relevant CTI relations and what model architecture works best to extract relations from CTI texts. It was also studied how much the use of pre-processing can improve the model's performance.

Analysis of available datasets showed that only datasets for Named Entity Recognition are publicly available. To overcome this issue, an dataset was developed, containing 9732 sentences annotated with NER tags and relations, out of which 480 contained relations. This dataset is a significant contribution to the field of NLP in the CTI domain, a field in which datasets are scarce, as discussed before.

Using the dataset developed as part of this work, a model was developed to extract relations from CTI texts by framing the problem as a sequence-to-sequence task. Moreover, the NYT dataset was used as task-specific pre-training to teach the T5 model to extract relations from texts. While this technique is commonly used in the NLP field, none of the reviewed related works from the CTI field used task-specific pre-training.

This work also verified the efficacy of pre-processing texts to deal with nuances that exist in CTI texts, such as dots, underscores in IOCs, similar to multiple other researchers. In addition to pre-processing texts with regular expressions, this work proposes the novel idea of using word lists of CTI entities, such as names of malware or tools, which can be scraped from the internet. The use of pre-processing increased the model's performance by 27% to a total of 0.374 F1 score on the dataset created as part of this work.

Overall, this work demonstrated that deep learning models can be effectively used to extract relevant information from CTI texts, and pre-processing is an essential step in achieving high performance. The developed dataset and model are publicly available and can be used by security analysts and researchers to analyze CTI texts and extract relevant information efficiently. Furthermore, the proposed pre-processing technique using word lists of CTI entities can be explored further to improve the performance of models in the CTI domain. It is hoped that this work will contribute to the development of more accurate and efficient methods for analyzing CTI texts and extracting relevant information, ultimately enhancing the security posture of organizations.

Bibliography

[1] M. T. Alam, D. Bhusal, Y. Park, and N. Rastogi. *CyNER: A Python Library for Cybersecurity Named Entity Recognition*. 2022. doi: 10.48550/ARXIV.2204.05754. url: https://arxiv.org/abs/2204.05754.

[2] M. Alawida, O. Omolara, O. Abiodun, and M. Al-Rajab. "A deeper look into cybersecurity issues in the wake of Covid-19: A survey". In: *Journal of King Saud University - Computer and Information Sciences* (Aug. 2022). doi: 10.1016/j.jksuci.2022.08.003.

[3] C. Begley and J. Ioannidis. "Reproducibility in Science Improving the Standard for Basic and Preclinical Research". In: *Circulation research* (Jan. 2015). doi: 10.1161/CIRCRESAHA.114.303819.

[4] Y. Bengio, R. Ducharme, P. Vincent, and C. Janvin. "A Neural Probabilistic Language Model". In: *Journal of machine learning research*. 2003.

[5] L. Biewald. *Experiment Tracking with Weights and Biases*. 2020. url: https://www.wandb.com/.

[6] R. Blloshmi, S. Conia, R. Tripodi, and R. Navigli. "Generating Senses and RoLes: An End-to-End Model for Dependency- and Span-based Semantic Role Labeling". In: *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*. Ed. by Z.-H. Zhou. International Joint Conferences on Artificial Intelligence Organization, Aug. 2021. doi: 10.24963/ijcai.2021/521. url: https://doi.org/10.24963/ijcai.2021/521.

[7] N. D. Cao, G. Izacard, S. Riedel, and F. Petroni. "Autoregressive Entity Retrieval". In: *International Conference on Learning Representations*. 2021. url: https://openreview.net/forum?id=5k8F6UU39V.

[8] M. M. Casey Hanks and A. Joshi. "Recognizing and Extracting Cybersecurity Entities from Text". In: *Workshop on Machine Learning for Cybersecurity, International Conference on Machine Learning*. PLMR, July 2022.

[9] F. Chollet. *Deep Learning with Python*. Manning, Nov. 2017. isbn: 9781617294433.

[10] CISA. *Alert (AA20-352A) Advanced Persistent Threat Compromise of Government Agencies, Critical Infrastructure, and Private Sector Organizations*. 2021. url: https://www.cisa.gov/uscert/ncas/alerts/aa20-352a.

[11] CISA. *Alert (AA21-131A) DarkSide Ransomware: Best Practices for Preventing Business Disruption from Ransomware Attacks*. 2020. url: https://www.cisa.gov/uscert/ncas/alerts/aa21-131a.

[12] C. Collberg and T. A. Proebsting. "Repeatability in Computer Systems Research". In: *Commun. ACM* 3 (Feb. 2016). issn: 0001-0782. doi: 10.1145/2812803.

[13] A. M. Dai and Q. V. Le. *Semi-supervised Sequence Learning*. 2015. doi: 10.48550/ARXIV.1511.01432. url: https://arxiv.org/abs/1511.01432.

[14] H. Dalianis. *Clinical Text Mining : Secondary Use of Electronic Patient Records*. 2018. isbn: 978-3-319-78502-8. doi: 10.1007/978-3-319-78503-5.

[15] O. R. developers. *ONNX Runtime*. https://onnxruntime.ai/. 2021.

[16] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*. 2018. doi: 10.48550/ARXIV.1810.04805. url: https://arxiv.org/abs/1810.04805.

[17] M. Eberts and A. Ulges. "An End-to-end Model for Entity-level Relation Extraction using Multi-instance Learning". In: *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*. Online: Association for Computational Linguistics, Apr. 2021. doi: 10.18653/v1/2021.eacl-main.319. url: https://aclanthology.org/2021.eacl-main.319.

[18] M. Eberts and A. Ulges. "Span-based Joint Entity and Relation Extraction with Transformer Pre-training". In: *CoRR* (2019). arXiv: 1909.07755.

[19] H. Elsahar, P. Vougiouklis, A. Remaci, C. Gravier, J. Hare, et al. "T-REx: A Large Scale Alignment of Natural Language with Knowledge Base Triples". In: *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*. Miyazaki, Japan: European Language Resources Association (ELRA), May 2018. url: https://aclanthology.org/L18-1544.

[20] S. T. I. eXpression. *Structured Threat Information eXpression*. url: http://stixproject.github.io/.

[21] Fris. "Extraktion von Informationen/TTPs aus Bedrohungsberichten zur Erstellung von Wissensgraphen durch den Einsatz von Machine Learning". Master's Thesis. XXX: Munich University of Applied Sciences, 2022.

[22] P. Gao, X. Liu, E. Choi, B. Soman, C. Mishra, et al. "A System for Automated Open-Source Threat Intelligence Gathering and Management". In: *Proceedings of the 2021 International Conference on Management of Data*. SIGMOD '21. Virtual Event, China: Association for Computing Machinery, 2021. isbn: 9781450383431. doi: 10.1145/3448016.3452745. url: https://doi.org/10.1145/3448016.3452745.

[23] P. Gao, F. Shao, X. Liu, X. Xiao, Z. Qin, et al. "Enabling Efficient Cyber Threat Hunting With Cyber Threat Intelligence". In: *CoRR* (2020). arXiv: 2010.13637.

[24] *Github ioc-parser*. https://github.com/armbues/ioc_parser.

[25] *Github Thesis Repo*. https://github.com/l0renor/Relation-Extraction-and-Knowledge-Graph-Generation-on-MISP-Event-Reports.

[26] O. E. Gundersen and S. Kjensmo. "State of the Art: Reproducibility in Artificial Intelligence". In: *Proceedings of the AAAI Conference on Artificial Intelligence* 1 (Apr. 2018). doi: 10.1609/aaai.v32i1.11503.

[27] A. A. Hagberg, D. A. Schult, and P. J. Swart. “Exploring Network Structure, Dynamics, and Function using NetworkX”. In: *Proceedings of the 7th Python in Science Conference*. Ed. by G. Varoquaux, T. Vaught, and J. Millman. Pasadena, CA USA, 2008.

[28] S. Hochreiter and J. Schmidhuber. “Long Short-Term Memory”. In: *Neural Comput.* 8 (Nov. 1997). issn: 0899-7667. doi: 10.1162/neco.1997.9.8.1735.

[29] M. Honnibal and I. Montani. “spaCy: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing”. 2017.

[30] *Huggingface DatasetV1*. https://huggingface.co/datasets/mrmoor/cyber-threat-intelligence.

[31] *Huggingface DatasetV2*. https://huggingface.co/datasets/Olec/cyber-threat-intelligence_v2.

[32] P.-L. Huguet Cabot and R. Navigli. “REBEL: Relation Extraction By End-to-end Language generation”. In: *Findings of the Association for Computational Linguistics: EMNLP 2021*. Punta Cana, Dominican Republic: Association for Computational Linguistics, Nov. 2021. url: https://aclanthology.org/2021.findings-emnlp.204.

[33] G. Husari, E. Al-Shaer, M. Ahmed, B. Chu, and X. Niu. “TTPDrill: Automatic and Accurate Extraction of Threat Actions from Unstructured Text of CTI Sources”. In: *Proceedings of the 33rd Annual Computer Security Applications Conference*. ACSAC ’17. Orlando, FL, USA: Association for Computing Machinery, 2017. isbn: 9781450353458. doi: 10.1145/3134600.3134646. url: https://doi.org/10.1145/3134600.3134646.

[34] M. Joshi, D. Chen, Y. Liu, D. S. Weld, L. Zettlemoyer, et al. “SpanBERT: Improving Pre-training by Representing and Predicting Spans”. In: *Transactions of the Association for Computational Linguistics* (2020). doi: 10.1162/tacl_a_00300.

[35] S. Kamiya, J.-K. Kang, J. Kim, A. Milidonis, and R. M. Stulz. “Risk management, firm reputation, and the impact of successful cyberattacks on target firms”. In: *Journal of Financial Economics* 3 (2021). issn: 0304-405X. doi: https://doi.org/10.1016/j.jfineco.2019.05.019.

[36] G. Kim, C. Lee, J. Jo, and H. Lim. “Automatic extraction of named entities of cyber threats using a deep Bi-LSTM-CRF network”. In: *International Journal of Machine Learning and Cybernetics* 10 (Oct. 2020). issn: 1868-808X. doi: 10.1007/s13042-020-01122-6.

[37] B. Kiran and C. Santiago. url: https://github.com/aptnotes/data.

[38] B. Kitchenham and S. Charters. “Guidelines for performing Systematic Literature Reviews in Software Engineering”. In: (Jan. 2007).

[39] N. Lambert, L. Castricato, L. von Werra, and A. Havrilla. *Illustrating Reinforcement Learning from Human Feedback (RLHF)*. Online-Blog. Dec. 2022. url: https://openai.com/blog/chatgpt/.

[40] K. Lee, M.-W. Chang, and K. Toutanova. *Latent Retrieval for Weakly Supervised Open Domain Question Answering*. 2019. doi: 10.48550/ARXIV.1906.00300. url: https://arxiv.org/abs/1906.00300.

[41] V. I. Levenshtein. "Binary Codes Capable of Correcting Deletions, Insertions and Reversals". In: *Soviet Physics Doklady* (Feb. 1966).

[42] M. Lewis, Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, et al. *BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension*. 2019. doi: 10.48550/ARXIV.1910.13461. url: https://arxiv.org/abs/1910.13461.

[43] E. D. Liddy. "Natural language processing." In: (2001).

[44] K. Liu, F. Wang, Z. Ding, S. Liang, Z. Yu, et al. "Recent Progress of Using Knowledge Graph for Cybersecurity". In: *Electronics* 15 (2022). issn: 2079-9292. doi: 10.3390/electronics11152287.

[45] *Malpedia*. https://malpedia.caad.fkie.fraunhofer.de/.

[46] M.-C. de Marneffe and C. D. Manning. "The Stanford Typed Dependencies Representation". In: *Coling 2008: Proceedings of the workshop on Cross-Framework and Cross-Domain Parser Evaluation*. Manchester, UK: Coling 2008 Organizing Committee, Aug. 2008. url: https://aclanthology.org/W08-1301.

[47] R. McMillan. *Definition: Threat intelligence*. url: https://www.gartner.com/en/documents/2487216.

[48] T. Mikolov, K. Chen, G. Corrado, and J. Dean. *Efficient Estimation of Word Representations in Vector Space*. 2013. doi: 10.48550/ARXIV.1301.3781. url: https://arxiv.org/abs/1301.3781.

[49] T. Mikolov, M. Karafiát, L. Burget, and J. Cernocký. "Recurrent neural network based language model." In: *INTERSPEECH*. Ed. by T. Kobayashi, K. Hirose, and S. Nakamura. ISCA, 2010. url: http://dblp.uni-trier.de/db/conf/interspeech/interspeech2010.html#MikolovKBCK10.

[50] MISP. *MISP, the open source threat sharing platform*. url: https://www.misp-project.org.

[51] *MISP Examples*. https://hdoc.csirt-tooling.org/o57tRHppR5ezWhZqrwFoFQ#Interesting-MISP-eventsexamples.

[52] *MISP Objects*. https://www.misp-project.org/objects.html.

[53] *MITRE ATTCK*. https://attack.mitre.org/.

[54] MSIP. *MISP Event report*. url: https://www.misp-project.org/2020/10/08/Event-Reports.html/.

[55] T. Nayak and H. T. Ng. "Effective Modeling of Encoder-Decoder Architecture for Joint Entity and Relation Extraction". In: *AAAI Conference on Artificial Intelligence*. 2019.

[56] Open$_AI$. *ChatGPT: OptimizingLanguage Modelsfor Dialogue*. Online-Blog. url: https://openai.com/blog/chatgpt/.

[57] openphish.com. *OSCTI feed: openphish.com*. url: https://openphish.com.

[58] G. Paolini, B. Athiwaratkun, J. Krone, J. Ma, A. Achille, et al. *Structured Prediction as Translation between Augmented Natural Languages*. Jan. 2021.

[59] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, et al. *Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer*. 2019. doi: 10.48550/ARXIV.1910.10683. url: https://arxiv.org/abs/1910.10683.

[60] N. Rastogi, S. Dutta, R. Christian, M. Zaki, A. Gittens, et al. *Information Prediction using Knowledge Graphs for Contextual Malware Threat Intelligence*. Feb. 2021. doi: 10.13140/RG.2.2.12526.54083.

[61] S. Riedel, L. Yao, and A. Mccallum. "Modeling Relations and Their Mentions without Labeled Text". In: Sept. 2010. isbn: 978-3-642-15938-1. doi: 10.1007/978-3-642-15939-8_10.

[62] D. Roth and W.-t. Yih. "A Linear Programming Formulation for Global Inference in Natural Language Tasks". In: *Proceedings of the Eighth Conference on Computational Natural Language Learning (CoNLL-2004) at HLT-NAACL 2004*. Boston, Massachusetts, USA: Association for Computational Linguistics, May 2004. url: https://aclanthology.org/W04-2401.

[63] I. Sarhan and M. Spruit. "Open-CyKG: An Open Cyber Threat Intelligence Knowledge Graph". In: *Knowledge-Based Systems* (2021). issn: 0950-7051. doi: https://doi.org/10.1016/j.knosys.2021.107524.

[64] I. Sutskever, O. Vinyals, and Q. V. Le. *Sequence to Sequence Learning with Neural Networks*. 2014. doi: 10.48550/ARXIV.1409.3215. url: https://arxiv.org/abs/1409.3215.

[65] B. Taillé, V. Guigue, G. Scoutheeten, and P. Gallinari. "Let's Stop Incorrect Comparisons in End-to-end Relation Extraction!" In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Online: Association for Computational Linguistics, Nov. 2020. doi: 10.18653/v1/2020.emnlp-main.301. url: https://aclanthology.org/2020.emnlp-main.301.

[66] Q. Tan, R. He, L. Bing, and H. T. Ng. *Document-Level Relation Extraction with Adaptive Focal Loss and Knowledge Distillation*. 2022. doi: 10.48550/ARXIV.2203.10900. url: https://arxiv.org/abs/2203.10900.

[67] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, et al. *Attention Is All You Need*. 2017. doi: 10.48550/ARXIV.1706.03762. url: https://arxiv.org/abs/1706.03762.

[68] J. Vig. "BertViz: A Tool for Visualizing Multi-Head Self-Attention in the BERT Model". In: (May 2019).

[69] C. Wang, X. Liu, Z. Chen, H. Hong, J. Tang, et al. “Zero-Shot Information Extraction as a Unified Text-to-Triple Translation”. In: *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2021.

[70] X. Wang, S. He, Z. Xiong, X. Wei, Z. Jiang, et al. “APTNER: A Specific Dataset for NER Missions in Cyber Threat Intelligence Field”. In: *2022 IEEE 25th International Conference on Computer Supported Cooperative Work in Design (CSCWD)*. 2022. doi: 10.1109/CSCWD54268.2022.9776031.

[71] X. Wang, X. Liu, S. Ao, N. Li, Z. Jiang, et al. “DNRTI: A Large-Scale Dataset for Named Entity Recognition in Threat Intelligence”. In: Dec. 2020. doi: 10.1109/TrustCom50675.2020.00252.

[72] Z. Wang, P. Ng, X. Ma, R. Nallapati, and B. Xiang. “Multi-passage BERT: A Globally Normalized BERT Model for Open-domain Question Answering”. In: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Hong Kong, China: Association for Computational Linguistics, Nov. 2019. doi: 10.18653/v1/D19-1599. url: https://aclanthology.org/D19-1599.

[73] T. Wilde and T. Hess. “Forschungsmethoden der Wirtschaftsinformatik”. In: *WIRTSCHAFTSINFORMATIK* 4 (Aug. 2007). issn: 1861-8936. doi: 10.1007/s11576-007-0064-z.

[74] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, et al. *HuggingFace’s Transformers: State-of-the-art Natural Language Processing*. 2019. doi: 10.48550/ARXIV.1910.03771. url: https://arxiv.org/abs/1910.03771.

[75] B. Xu, Q. Wang, Y. Lyu, Y. Zhu, and Z. Mao. “Entity Structure Within and Throughout: Modeling Mention Dependencies for Document-Level Relation Extraction”. In: *Proceedings of the AAAI Conference on Artificial Intelligence* 16 (May 2021). doi: 10.1609/aaai.v35i16.17665.

[76] I. Yamada, A. Asai, H. Shindo, H. Takeda, and Y. Matsumoto. “LUKE: Deep Contextualized Entity Representations with Entity-aware Self-attention”. In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Online: Association for Computational Linguistics, Nov. 2020. doi: 10.18653/v1/2020.emnlp-main.523. url: https://aclanthology.org/2020.emnlp-main.523.

[77] Y. Yao, D. Ye, P. Li, X. Han, Y. Lin, et al. “DocRED: A Large-Scale Document-Level Relation Extraction Dataset”. In: *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Florence, Italy: Association for Computational Linguistics, July 2019. doi: 10.18653/v1/P19-1074. url: https://aclanthology.org/P19-1074.

[78] M. Zaheer, G. Guruganesh, A. Dubey, J. Ainslie, C. Alberti, et al. “Big Bird: Transformers for Longer Sequences”. In: (2020). doi: 10.48550/ARXIV.2007.14062.

[79] D. Zeng, K. Liu, S. Lai, G. Zhou, and J. Zhao. “Relation Classification via Convolutional Deep Neural Network”. In: *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*. Dublin, Ireland: Dublin City University and Association for Computational Linguistics, Aug. 2014. url: https://aclanthology.org/C14-1220.

[80] X. Zeng, S. He, D. Zeng, K. Liu, S. Liu, et al. “Learning the Extraction Order of Multiple Relational Facts in a Sentence with Reinforcement Learning”. In: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Hong Kong, China: Association for Computational Linguistics, Nov. 2019. doi: 10.18653/v1/D19-1035. url: https://aclanthology.org/D19-1035.

[81] X. Zeng, D. Zeng, S. He, K. Liu, and J. Zhao. “Extracting Relational Facts by an End-to-End Neural Model with Copy Mechanism”. In: *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Melbourne, Australia: Association for Computational Linguistics, July 2018. doi: 10.18653/v1/P18-1047. url: https://aclanthology.org/P18-1047.

[82] X. Zeng, D. Zeng, S. He, K. Liu, and J. Zhao. “Extracting Relational Facts by an End-to-End Neural Model with Copy Mechanism”. In: *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Melbourne, Australia: Association for Computational Linguistics, July 2018. doi: 10.18653/v1/P18-1047. url: https://aclanthology.org/P18-1047.

[83] A. Zhang, Z. C. Lipton, M. Li, and A. J. Smola. *Dive into Deep Learning*. 2023. arXiv: 2106.11342 [cs.LG].

[84] R. H. Zhang, Q. Liu, A. X. Fan, H. Ji, D. Zeng, et al. “Minimize Exposure Bias of Seq2Seq Models in Joint Entity and Relation Extraction”. In: *Findings of the Association for Computational Linguistics: EMNLP 2020*. Online: Association for Computational Linguistics, Nov. 2020. doi: 10.18653/v1/2020.findings-emnlp.23. url: https://aclanthology.org/2020.findings-emnlp.23.

[85] P. Zhou, W. Shi, J. Tian, Z. Qi, B. Li, et al. “Attention-Based Bidirectional Long Short-Term Memory Networks for Relation Classification”. In: *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. Berlin, Germany: Association for Computational Linguistics, Aug. 2016. doi: 10.18653/v1/P16-2034. url: https://aclanthology.org/P16-2034.

A Appendix

A.1 Regexes used for the pre-processing

```
'APT': ['[Aa][Pp][Tt]\s?\d\d\d?', '[Tt][Aa]\d\d\d?'],
'DOM': [
    r"[a-z0-9.\-]+[.](?:com|net|org|edu|gov|mil|aero|asia|biz|cat|coop|info|int|jobs|mobi|
        museum|name|post|pro|tel|travel|xxx|ac|ad|ae|af|ag|ai|al|am|an|ao|aq|ar|as|at|au|aw|ax
        |az|ba|bb|bd|be|bf|bg|bh|bi|bj|bm|bn|bo|br|bs|bt|bv|bw|by|bz|ca|cc|cd|cf|cg|ch|ci|ck|
        cl|cm|cn|co|cr|cs|cu|cv|cx|cy|cz|dd|de|dj|dk|dm|do|dz|ec|ee|eg|eh|er|es|et|eu|fi|fj|fk
        |fm|fo|fr|ga|gb|gd|ge|gf|gg|gh|gi|gl|gm|gn|gp|gq|gr|gs|gt|gu|gw|gy|hk|hm|hn|hr|ht|hu|
        id|ie|il|im|in|io|iq|ir|is|it|je|jm|jo|jp|ke|kg|kh|ki|km|kn|kp|kr|kw|ky|kz|la|lb|lc|li
        |lk|lr|ls|lt|lu|lv|ly|ma|mc|md|me|mg|mh|mk|ml|mm|mn|mo|mp|mq|mr|ms|mt|mu|mv|mw|mx|my|
        mz|na|nc|ne|nf|ng|ni|nl|no|np|nr|nu|nz|om|pa|pe|pf|pg|ph|pk|pl|pm|pn|pr|ps|pt|pw|py|qa
        |re|ro|rs|ru|rw|sa|sb|sc|sd|se|sg|sh|si|sj|Ja|sk|sl|sm|sn|so|sr|ss|st|su|sv|sx|sy|sz|
        tc|td|tf|tg|th|tj|tk|tl|tm|tn|to|tp|tr|tt|tv|tw|tz|ua|ug|uk|us|uy|uz|va|vc|ve|vg|vi|vn
        |vu|wf|ws|ye|yt|yu|za|zm|zw)(?!\w)"],
'IPV4': ['\b(?:[0-9]{1,3}\.){3}[0-9]{1,3}\b', '^(?:[0-9]{1,3}\.){3}[0-9]{1,3}$'],
'IPV6': [
    '(([0-9a-fA-F]{1,4}:){7,7}[0-9a-fA-F]{1,4}|([0-9a-fA-F]{1,4}:){1,7}:|([0-9a-fA-F]{1,4}:)
        {1,6}:[0-9a-fA-F]{1,4}|([0-9a-fA-F]{1,4}:){1,5}(:[0-9a-fA-F]{1,4}){1,2}|([0-9a-fA-F
        ]{1,4}:){1,4}(:[0-9a-fA-F]{1,4}){1,3}|([0-9a-fA-F]{1,4}:){1,3}(:[0-9a-fA-F]{1,4})
        {1,4}|([0-9a-fA-F]{1,4}:){1,2}(:[0-9a-fA-F]{1,4}){1,5}|[0-9a-fA-F]{1,4}:((:[0-9a-fA-F
        ]{1,4}){1,6})|:((:[0-9a-fA-F]{1,4}){1,7}|:)|fe80:(:[0-9a-fA-F]{0,4}){0,4}%[0-9a-zA-Z
        ]{1,}|::(ffff(:0{1,4}){0,1}:){0,1}((25[0-5]|(2[0-4]|1{0,1}[0-9]){0,1}[0-9])\.)
        {3,3}(25[0-5]|(2[0-4]|1{0,1}[0-9]){0,1}[0-9])|([0-9a-fA-F]{1,4}:)
        {1,4}:((25[0-5]|(2[0-4]|1{0,1}[0-9]){0,1}[0-9])\.){3,3}(25[0-5]|(2[0-4]|1{0,1}[0-9])
        {0,1}[0-9]))',
'EMAIL': ['[a-z0-9]+[\._]?[a-z0-9]+[@]\w+[.]\w{2,3}$',
          '(?:[a-z0-9!#$%&\'*+/=?^_`{|}~-]+(?:\.[a-z0-9!#$%&\'*+/=?^_`{|}~-]+)*|"(?:[\x01-\x08\
              x0b\x0c\x0e-\x1f\x21\x23-\x5b\x5d-\x7f]|\\[\x01-\x09\x0b\x0c\x0e-\x7f])*")@
              (?:(?:[a-z0-9](?:[a-z0-9-]*[a-z0-9])?\.)+[a-z0-9](?:[a-z0-9-]*[a-z0-9])
              ?|\[(?:(?:(2(5[0-5]|[0-4][0-9])|1[0-9][0-9]|[1-9]?[0-9]))\.)
              {3}(?:(2(5[0-5]|[0-4][0-9])|1[0-9][0-9]|[1-9]?[0-9])|[a-z0-9-]*[a-z0-9]:(?:[\x01
              -\x08\x0b\x0c\x0e-\x1f\x21-\x5a\x53-\x7f]|\\[\x01-\x09\x0b\x0c\x0e-\x7f])+)\])'
              ],
'URL': [
    '((?:https?|ftp:(?:/{1,3}|[a-z0-9%])|[a-z0-9.\-]+[.](?:com|net|org|edu|gov|mil|aero|asia|
        biz|cat|coop|info|int|jobs|mobi|museum|name|post|pro|tel|travel|xxx|ac|ad|ae|af|ag|ai|
        al|am|an|ao|aq|ar|as|at|au|aw|ax|az|ba|bb|bd|be|bf|bg|bh|bi|bj|bm|bn|bo|br|bs|bt|bv|bw
        |by|bz|ca|cc|cd|cf|cg|ch|ci|ck|cl|cm|cn|co|cr|cs|cu|cv|cx|cy|cz|dd|de|dj|dk|dm|do|dz|
        ec|ee|eg|eh|er|es|et|eu|fi|fj|fk|fm|fo|fr|ga|gb|gd|ge|gf|gg|gh|gi|gl|gm|gn|gp|gq|gr|gs
        |gt|gu|gw|gy|hk|hm|hn|hr|ht|hu|id|ie|il|im|in|io|iq|ir|is|it|je|jm|jo|jp|ke|kg|kh|ki|
        km|kn|kp|kr|kw|ky|kz|la|lb|lc|li|lk|lr|ls|lt|lu|lv|ly|ma|mc|md|me|mg|mh|mk|ml|mm|mn|mo
        |mp|mq|mr|ms|mt|mu|mv|mw|mx|my|mz|na|nc|ne|nf|ng|ni|nl|no|np|nr|nu|nz|om|pa|pe|pf|pg|
        ph|pk|pl|pm|pn|pr|ps|pt|pw|py|qa|re|ro|rs|ru|rw|sa|sb|sc|sd|se|sg|sh|si|sj|Ja|sk|sl|sm
        |sn|so|sr|ss|st|su|sv|sx|sy|sz|tc|td|tf|tg|th|tj|tk|tl|tm|tn|to|tp|tr|tt|tv|tw|tz|ua|
        ug|uk|us|uy|uz|va|vc|ve|vg|vi|vn|vu|wf|ws|ye|yt|yu|za|zm|zw)/)(?:[^\s()
        <>{}\[\]]+|\([^\s()]*?\([^\s()]+\)[^\s()]*?\)|\([^\s]+?\))+(?:\([^\s()]*?\([^\s()]+\)
        [^\s()]*?\)|\([^\s]+?\)|[^\s`!()\[\]{};:\'".,<>?«»“”‘’])|(?:(?<!@)[a-z0
        -9]+(?:[.\-][a-z0-9]+)*[.](?:com|net|org|edu|gov|mil|aero|asia|biz|cat|coop|info|int|
        jobs|mobi|museum|name|post|pro|tel|travel|xxx|ac|ad|ae|af|ag|ai|al|am|an|ao|aq|ar|as|
        at|au|aw|ax|az|ba|bb|bd|be|bf|bg|bh|bi|bj|bm|bn|bo|br|bs|bt|bv|bw|by|bz|ca|cc|cd|cf|cg
        |ch|ci|ck|cl|cm|cn|co|cr|cs|cu|cv|cx|cy|cz|dd|de|dj|dk|dm|do|dz|ec|ee|eg|eh|er|es|et|
        eu|fi|fj|fk|fm|fo|fr|ga|gb|gd|ge|gf|gg|gh|gi|gl|gm|gn|gp|gq|gr|gs|gt|gu|gw|gy|hk|hm|hn
        |hr|ht|hu|id|ie|il|im|in|io|iq|ir|is|it|je|jm|jo|jp|ke|kg|kh|ki|km|kn|kp|kr|kw|ky|kz|
        la|lb|lc|li|lk|lr|ls|lt|lu|lv|ly|ma|mc|md|me|mg|mh|mk|ml|mm|mn|mo|mp|mq|mr|ms|mt|mu|mv
```

```
            |mw|mx|my|mz|na|nc|ne|nf|ng|ni|nl|no|np|nr|nu|nz|om|pa|pe|pf|pg|ph|pk|pl|pm|pn|pr|ps|
            pt|pw|py|qa|re|ro|rs|ru|rw|sa|sb|sc|sd|se|sg|sh|si|sj|Ja|sk|sl|sm|sn|so|sr|ss|st|su|sv
            |sx|sy|sz|tc|td|tf|tg|th|tj|tk|tl|tm|tn|to|tp|tr|tt|tv|tw|tz|ua|ug|uk|us|uy|uz|va|vc|
            ve|vg|vi|vn|vu|wf|ws|ye|yt|yu|za|zm|zw)\b/?(?!@))'],
    'VULID': ['CVE-\d{4}-\d{4,7}'],
    'SHA2': [r"[A-Fa-f0-9]{64}"],
    'SHA1': [r"[0-9a-f]{40}"],
    'MD5': [r"[a-f0-9]{32}"],
    'REGISTRYKEY': [
        r"(HKEY_LOCAL_MACHINE\\|HKLM\\|HKEY_CURRENT_USER\\|HKCU\\|HKLM\\|HKEY_USERS\\|HKU\\)([a-zA-
            Z0-9\s_@\-\^!#.\:\/\$%&+={}\[\]\\*])+"],
```

Listing A.1: *List of Regexes used for the pre-processing of CTI-texts*

A.2 Examples of Graphs

A.2.1 Example 1

The group frequently uses publicly available exploits to conduct widespread scanning and exploitation against vulnerable systems, likely in an effort to obtain authentication credentials to allow further access. This broad targeting potentially gives the group access to a large number of systems globally, many of which are unlikely to be of immediate intelligence value. The group may maintain a store of stolen credentials in order to access these systems in the event that they become more relevant to their requirements in the future.
In recent attacks targeting COVID-19 vaccine research and development, the group conducted basic vulnerability scanning against specific external IP addresses owned by the organisations. The group then deployed public exploits against the vulnerable services identified.

Listing A.2: *Text example 1*

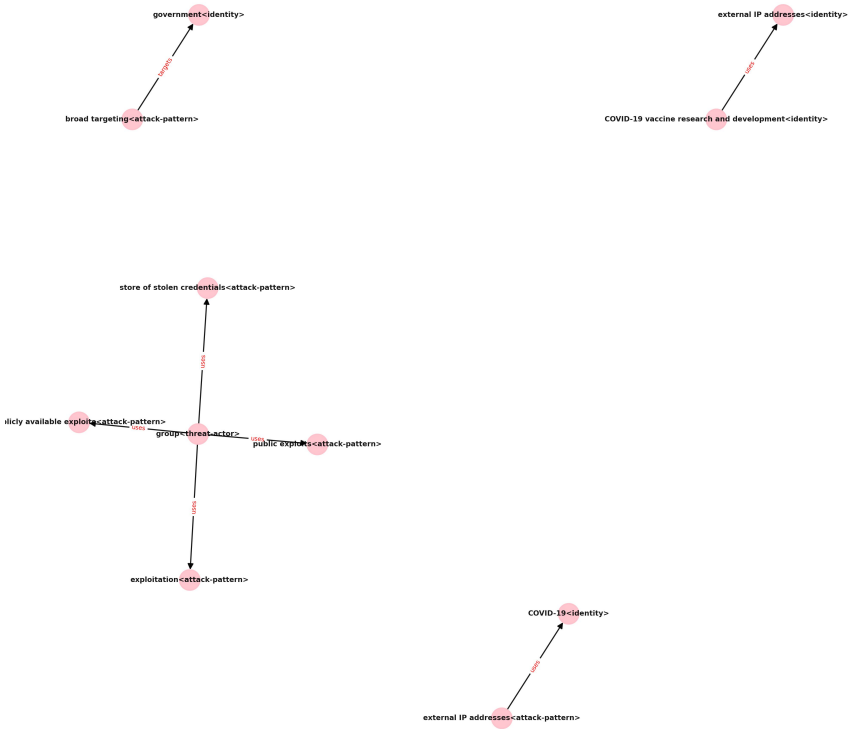


Figure A.1: *Graph example 2*

A.2.2 Example 2

```
In some cases, APT29 also deploys custom malware known as WellMess or WellMail
to conduct further operations on the victims system.
WellMess is malware written in either Golang or .NET and has been in use since at
least 2018. WellMess was first reported on by |PCERT and LAC researchers in July
2018[4][5]. It is named after one of the function names in the malware - wellmess .
WellMess is a lightweight malware designed to execute arbitrary shell commands,
upload and download files. The malware supports HTTP, TLS and DNS
communications methods.
Indicators of compromise (lOCs) for WellMess are available in the appendix.
```

Listing A.3: *Text example 2*

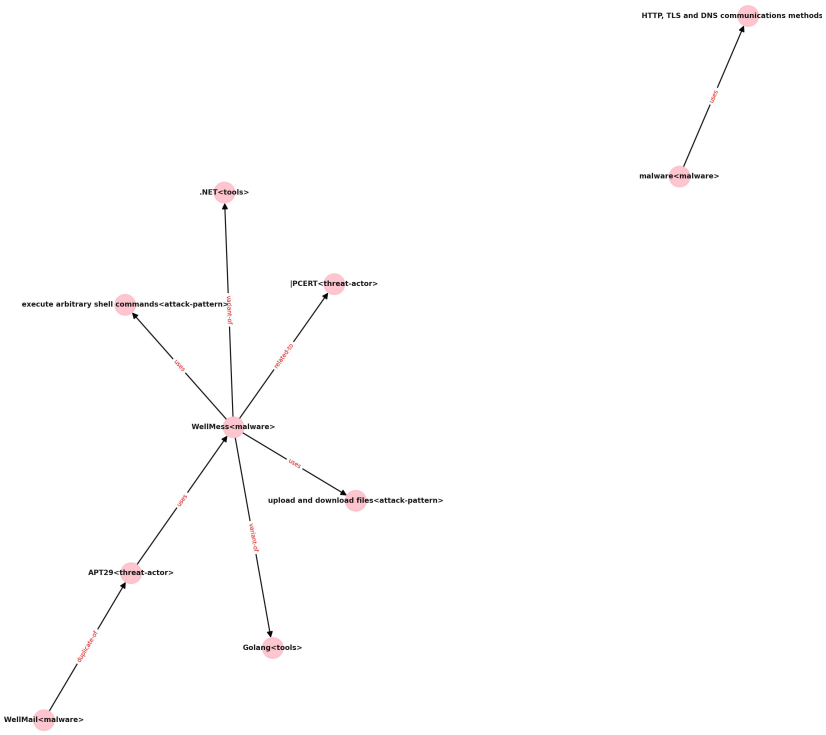


Figure A.2: *Graph example 2*

A.2.3 Example 3

The FBI and CISA have observed Russian state−sponsored APT actor activity targeting U.S. SLTT government networks, as well as aviation networks. The APT actor is using Turkish IP addresses 213.74.101[.]65, 213. 74.139[.]196, and 212.252. 30 .]17@ to connect to victim web servers (Exploit Public Facing Application [11190]).
The actor is using 213.74.101[.]65 and 213.74.139[.]196 to attempt brute force logins and, in several instances, attempted Structured Query Language (SQL) injections on victim websites (Brute
Force [11110]; Exploit Public Facing Application [11190]). The APT actor also hosted malicious domains, including possible aviation sector target columbusairports.microsoftonline[.]host, which resolved to 108.177.235[.]92 and [cityname] .westus2.cloudapp.azure.com; these domains are U.S. registered and are likely SLTT government targets (Drive−By Compromise [11189]).
The APT actor scanned for vulnerable Citrix and Microsoft Exchange services and identified vulnerable systems, likely for future exploitation. This actor continues to exploit a Citrix Directory
Traversal Bug (CVE−2019−19781) and a Microsoft Exchange remote code execution flaw (CVE−2020−0688).

Listing A.4: *Text example 3*

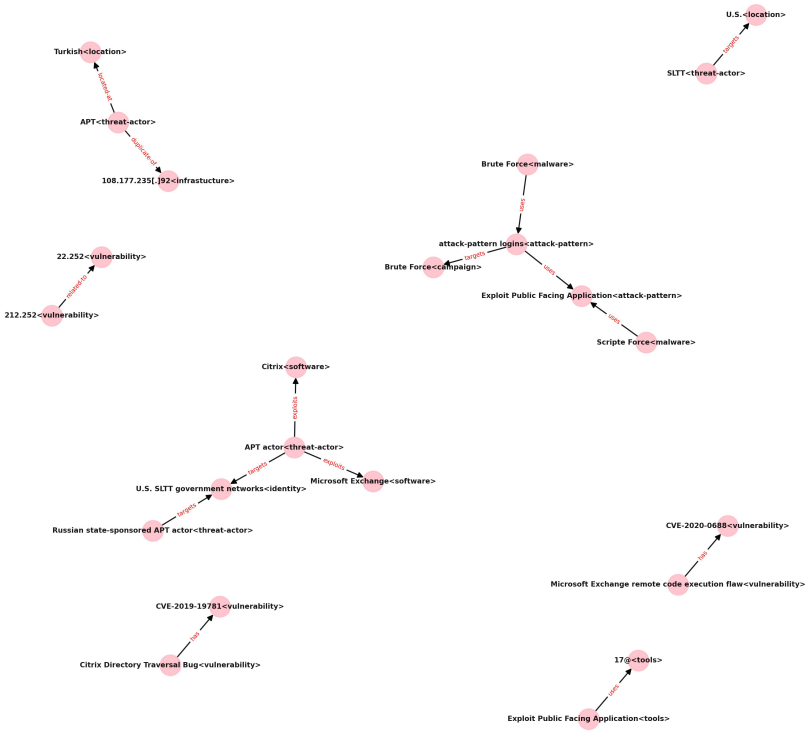


Figure A.3: *Graph example 3*

A.2.4 Example 4

```
CISA and FBI are aware of a widespread campaign from an Iran-based malicious cyber actor
targeting several industries mainly associated with information technology, government,
healthcare, financial, insurance, and media sectors across the United States. The threat
actor conducts mass-scanning and uses tools, such as Nmap, to identify open ports. Once
the open ports are identified, the threat actor exploits CVEs related to VPN infrastructure to
gain initial access to a targeted network. CISA and the FBI have observed the threat actor
exploiting multiple CVEs, including CVE-2019-11510, CVE-2019-11539, CVE-2019-19781, and
CVE-2020-5902.
After gaining initial access to a targeted network, the threat actor obtains administratorlevel
credentials and installs web shells allowing further entrenchment. After establishing a
foothold, the threat actors goals appear to be maintaining persistence and exfiltrating data
    .
This threat actor has been observed selling access to compromised network infrastructure
in an online hacker forum. Industry reporting indicates that the threat actor operates as a
contractor supporting Iranian government interests, but the malicious activity appears to
also serve the threat actors own financial interests. The FBI notes this threat actor has
    the
capability, and likely the intent, to deploy ransomware on victim networks.
CISA and FBI have observed this Iran-based threat actor relying on exploits of remote
external services on internet-facing assets to gain initial access to victim networks. The
threat actor also relies heavily on open-source and operating system (OS) tooling to conduct
operations, such as ngrok; fast reverse proxy (FRP); Lightweight Directory Access Protocol
(LDAP) directory browser; as well as web shells known as ChunkyTuna, Tiny, and China
Chopper.
```

Listing A.5: *Text example 4*

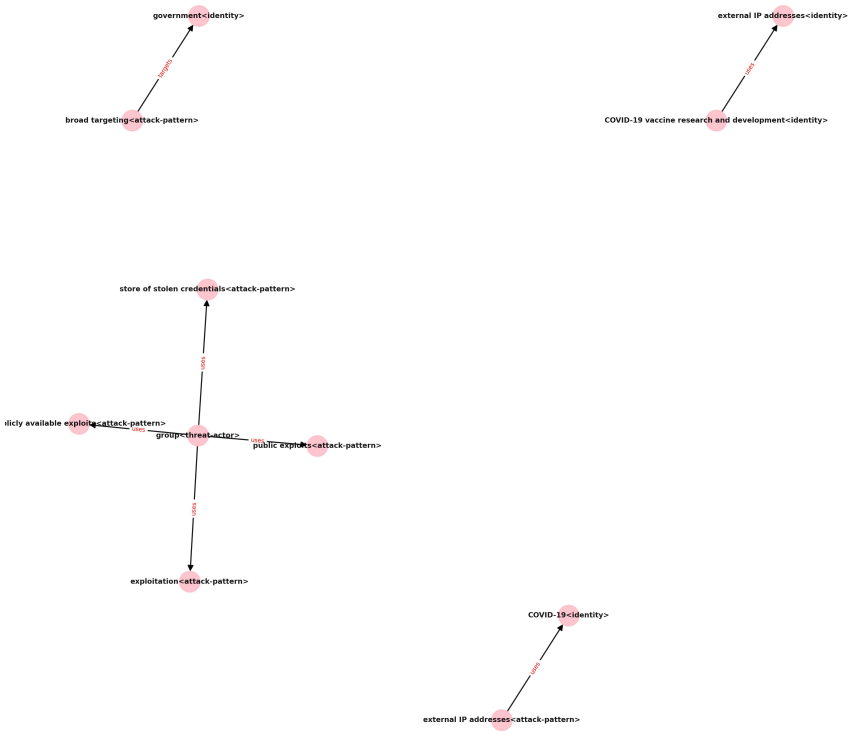


Figure A.4: *Graph example 4*

A.3 Error Analysis

On August 22, 2018, the Apache Foundation released a critical security update for VULID, a remote code execution vulnerability affecting Apache Struts versions 2.3 to 2.3.34 and 2.5 to 2.5.16.
Expected:
<triplet>Apache Struts versions 2.3 to 2.3.34 and 2.5 to 2.5.16<software>VULID<vulnerability> has<triplet>Apache Struts versions 2.3 to 2.3.34 and 2.5 to 2.5.16<software>remote code execution vulnerability<vulnerability>has
model out:
<pad><triplet> Apache Struts<software> VULID<vulnerability> has<triplet> VULID<vulnerability> remote code execution<attack−pattern> uses</s>
{'precision': 0, 'recall': 0, 'f1': 0}

For example, in the âĂIJproofâĂİ of the Okta breach posted on the Lapsus$ GroupâĂŹs Telegram channel, the actor states: âĂIJâĂę our focus was ONLY on okta customersâĂİ (Figure 3)
Expected:
<triplet>Lapsus$<threat−actor>Okta<identity>targets
model out:
<pad><triplet> Lapsus$<threat−actor> Okta<malware> targets</s>
{'precision': 0, 'recall': 0, 'f1': 0}

```
Compared to Patchwork, whose Trojanized documents exploit at least five security flaws,
    ConfuciusâĂŹ backdoors are delivered through Office files exploiting memory corruption
    vulnerabilities VULID and VULID.
Expected:
<triplet>Confucius<threat-actor>exploiting memory corruption<attack-pattern>uses
model out:
<pad><triplet> Patchwork<threat-actor> VULID<vulnerability> exploits<triplet> Confucius<threat-
    actor> Office files<software> exploits<triplet> Confucius<threat-actor> VULID<
    vulnerability> exploits</s>
{'precision': 0, 'recall': 0, 'f1': 0}

 DDoS Attacks Impacting Ukrainian Government and Banking Institutions malware Malware Website
     Defacement Rise in attack-pattern and Scam Attacks Increase in Cybersquatting Trends Fake
     Donation Websites DoS Attacks on Ukrainian News Sites Distribution of Apps How Palo Alto
     Networks Is Working to Keep You malware How You Should Prepare for an Increase in
     Cyberthreats Such as Wipers, DDoS, Website Defacement and Other Related Attacks How Unit
     42 Threat Intelligence and Security Consulting Can Help Additional Cybersecurity
     Resources Indicators of Compromise
Expected:
<triplet>DDoS Attacks<attack-pattern>Ukrainian Government and Banking Institutions<identity>
    targets
model out:
<pad><triplet> DDoS attacks<attack-pattern> Ukrainian Government and Banking Institutions<
    identity> targets<triplet> DDoS attacks<attack-pattern> Ukrainian Government and Banking
    Institutions<identity> targets<triplet> Cybersquatting<attack-pattern> Ukrainian News
    Sites<identity> targets<triplet> Cybersquatting<attack-pattern> Ukrainian Government and
    Banking Institutions<identity> targets<triplet> DDoS attacks<attack-pattern> Ukrainian
    News Sites<identity> targets</s>
{'precision': 0, 'recall': 0, 'f1': 0}

Both StealthMutant and StealthVector contain a payload of either the Cobalt Strike beacon or
    ScrambleCross, a newly discovered backdoor.
Expected:
<triplet>StealthVector<malware>Cobalt Strike<tools>uses<triplet>StealthVector<malware>
    ScrambleCross<malware>uses<triplet>StealthMutant<malware>Cobalt Strike<tools>uses<triplet>
    StealthMutant<malware>ScrambleCross<malware>uses
model out:
<pad><triplet> StealthVector<malware> Cobalt Strike<tools> uses<triplet> StealthMutant<malware>
     ScrambleCross<malware> uses<triplet> StealthVector<malware> Cobalt Strike<tools> uses</s>
{'precision': 1.0, 'recall': 0.5}

Virlock has the largest number of variants due to its file-infector-like behavior.
Expected:
<triplet>Virlock<malware>file-infector-like behavior<attack-pattern>uses
model out:
<pad><triplet> Virlock<malware> file-infector-like behavior<attack-pattern> uses</s>
{'precision': 1.0, 'recall': 1.0}

 The FBI has connected Conti to more than 400 cyberattacks against organizations worldwide,
     three-quarters of which are based in the U.S., with demands as high as $25 million.
Expected:
<triplet>Conti<threat-actor>U.S.<location>targets
model out:
<pad><triplet> Conti<threat-actor> U.S.<location> located-at</s>
{'precision': 0, 'recall': 0, 'f1': 0}

Indicators of Compromise (IoCs)      SHA256 Filename Note Analysis SHA2 big.txt malware
    Python Script (Version 3) TrojanSpy.Python.BIOPASS.A SHA2 c1222.txt malware Python
    Script (C1222 module) Trojan.Python.BIOPASS.A SHA2 c1222.txt malware Python Script (
    C1222 module) Trojan.Python.BIOPASS.A SHA2 c1222.txt malware Python Script (C1222
    module)
Expected:
```

```
<triplet>SHA2<hash>malware<malware>indicates<triplet>big.txt<filepath>malware<malware>indicates
      <triplet>big.txt<filepath>SHA2<hash>duplicate−of<triplet>malware<malware>TrojanSpy.Python.
      BIOPASS.A<malware>duplicate−of<triplet>SHA2<hash>malware<malware>indicates<triplet>c1222.
      txt<filepath>malware<malware>indicates<triplet>c1222.txt<filepath>SHA2<hash>duplicate−of<
      triplet>Trojan.Python.BIOPASS.A<malware>malware<malware>duplicate−of<triplet>SHA2<hash>
      malware<malware>indicates<triplet>c1222.txt<filepath>malware<malware>indicates<triplet>
      Trojan.Python.BIOPASS.A<malware>malware<malware>duplicate−of<triplet>c1222.txt<filepath>
      SHA2<hash>duplicate−of<triplet>SHA2<hash>malware<malware>indicates<triplet>c1222.txt<
      filepath>malware<malware>indicates<triplet>c1222.txt<filepath>SHA2<hash>duplicate−of
model out:
<pad><triplet> SHA2<hash> malware<malware> indicates<triplet> c1222.txt<filepath> malware<
      malware> indicates<triplet> malware<malware> C1222 module<filepath> duplicate−of<triplet>
      Python Script (C1222 module)<malware> Trojan.Python.BIOPASS.A<malware> duplicate−of<
      triplet> malware<malware> c1222.txt<filepath> duplicate−of<triplet> Python Script (C1222
      module)<malware> Trojan.Python.BIOPASS.A<malware> duplicate−of<triplet> SHA2<hash> Trojan.
      Python.BIOPASS.A<malware> indicates<triplet> Trojan.Python.Python.BIOPASS.A<malware>
      indicates<triplet> SHA2<hash> Trojan.Python.Python.BIOPASS.A<malware> indicates<triplet>
      SHA2<hash> Trojan.Python.BIOPASS.A<malware> indicates<triplet> SHA2<hash> Trojan.Python.
      Python.BIOPASS.A<malware> indicates<triplet> c1222.
      txtxtxtxtxtxtxtxtxtxtxtxtxtxtxtxtxtxtxtxt
{'precision': 0.2, 'recall': 0.13333333333333333}

The campaign initially compromised e−commerce online shops in Japan, Australia, and European
      countries via spam emails with malicious attachments.
Expected:
<triplet>campaign<campaign>e−commerce online shops<identity>targets<triplet>e−commerce online
      shops<identity>Japan<location>located−at<triplet>e−commerce online shops<identity>
      Australia<location>located−at<triplet>e−commerce online shops<identity>European<location>
      located−at<triplet>campaign<campaign>spam emails<attack−pattern>uses
model out:
<pad><triplet> campaign<campaign> e−commerce online shops<identity> targets<triplet> campaign<
      campaign> Japan<location> targets<triplet> e−commerce online shops<identity> European
      countries<location> targets</s>
{'precision': 0.3333333333333333, 'recall': 0.2}

malware, a known backdoor, is executed by abusing the attack−pattern dynamic−link library (DLL)
      .
Expected:
<triplet>malware<malware>abusing the attack−pattern dynamic−link library (DLL)<attack−pattern>
      uses
model out:
<pad><triplet> malware<malware> abusing the attack−pattern dynamic−link library (DLL)<attack−
      pattern> uses</s>
{'precision': 1.0, 'recall': 1.0}
```

```
 Geographic Distribution of Koluoz Spam Nodes in North America  Thus far weâĂŹve detected the
      following command and control servers in use.
Expected:
<triplet>Koluoz<malware>Spam Nodes<infrastucture>communicates−with<triplet>Spam Nodes<
      infrastucture>North America<location>located−at
model out:
<pad><triplet> Koluoz Spam Nodes<malware> North America<location> located−at</s>
{'precision': 0, 'recall': 0, 'f1': 0}

During our investigation into the Turla CyberAzov apps, we identified another Android app first
      seen in the wild in March 2022 that also claimed to conduct DoS attacks against Russian
      websites.
Expected:
<triplet>Turla<threat−actor>CyberAzov<malware>uses
model out:
<pad><triplet> Turla CyberAzov apps<malware> Android<software> targets</s>
{'precision': 0, 'recall': 0, 'f1': 0}
```

While VULID is a different vulnerability in Spring Cloud Function (not technically part of SpringShell), a Threat Prevention signature is also available to ensure coverage at the perimeter.
Expected:
<triplet>Spring Cloud<software>VULID<vulnerability>has
model out:
<pad><triplet> Spring Cloud Function<software> VULID<vulnerability> has<triplet> Spring Cloud Function<software> VULID<vulnerability> has</s>
{'precision': 0, 'recall': 0, 'f1': 0}

In tracking this threat, early reports show that the malware has been deployed against a financial institution in Ukraine as well as two contractors in Latvia and Lithuania that provide services to the Ukrainian Government.
Expected:
<triplet>malware<malware>inancial institution<identity>targets<triplet>malware<malware>two contractors in Latvia and Lithuania<identity>targets
model out:
<pad><triplet> malware<malware> Ukrainian Government<identity> targets<triplet> malware<malware> Latvia<location> targets<triplet> malware<malware> Lithuania<location> targets</s>
{'precision': 0, 'recall': 0, 'f1': 0}

In total weâĂŹve found 18 different applications in the Google Play store that contain the Funtasy Trojan.
Expected:
<triplet>Google Play<software>Funtasy<malware>hosts
model out:
<pad><triplet> Funtasy Trojan<malware> Google Play<software> targets</s>
{'precision': 0, 'recall': 0, 'f1': 0}

While not all SMS-based malware applications steal user data, we recently identified that the Chinese Taomike SDK has begun capturing copies of all messages received by the phone and sending them to a Taomike controlled server.
Expected:
<triplet>Taomike SDK<malware>Taomike controlled server<infrastucture>exfiltrates-to<triplet>Taomike SDK<malware>Taomike controlled server<infrastucture>communicates-with<triplet>Taomike SDK<malware>Chinese<location>located-at<triplet>Taomike SDK<malware>steal user data<attack-pattern>uses
model out:
<pad><triplet> Taomike controlled server<infrastucture> steal user data<attack-pattern> uses</s>
{'precision': 0, 'recall': 0, 'f1': 0}

We found an Avast anti-attack-pattern driver installed as service 'asWarPot.sys' using the command sc.exe create aswSP_ArPot2 binPath= C:\windows\aswArPot.sys type= kernel.
Expected:
<triplet>Avast anti-attack-pattern driver<software>asWarPot.sys<filepath>consists-of
model out:
<pad><triplet> sc.exe<filepath> aswSP_ArPot.sys<filepath> indicates<triplet> Avast anti-attack-pattern driver<tools> aswSP_ArPot.sys<filepath> consists-of</s>
{'precision': 0, 'recall': 0, 'f1': 0}

The Cobalt Strike loader, which has a PDB string âĂIJC:\Users\test\Desktop\fishmaster\x64\Release\fishmaster.pdbâĂİ, connects to the C&C server âĂIJdownload[.]google-images[.]mlâĂİ.
Expected:
<triplet>C:\Users\test\Desktop\fishmaster\x64\Release\fishmaster.pdb<filepath>Cobalt Strike<tools>indicates<triplet>Cobalt Strike<tools>download[.]google-images[.]ml<url>beacons-to
model out:
<pad><triplet> Cobalt Strike<tools> C:<unk>Users<unk>test<unk>Desktop<unk>fishmaster<unk>x64<unk>Release<unk>fishmaster.pdb<filepath> consists-of<triplet> C&C server<infrastucture> download[.]google-images[.]ml<url> consists-of</s>
{'precision': 0, 'recall': 0, 'f1': 0}

During the operation, 25.1 BTC from 538 wallets was sent to 1NDyJtNTjmwk5xPNhjgAMu4HDHigtobu1s (SHA256: SHA2), which is a wallet owned by Binance cryptocurrency exchange.
Expected:
<triplet>Binance cryptocurrency exchange<identity>SHA2<hash>owns
model out:
<pad><triplet> SHA2<hash> Binance cryptocurrency exchange<identity> indicates</s>
{'precision': 0, 'recall': 0, 'f1': 0}

This post is also available in: æŮčæIJňèld (Japanese) A recent vulnerability in the Kerberos authentication protocol, VULID (dubbed Bronze Bit), has been disclosed by Microsoft.
Expected:
<triplet>Bronze Bit<vulnerability>VULID<vulnerability>duplicate-of
model out:
<pad><triplet> Bronze Bit<vulnerability> VULID<vulnerability> duplicate-of<triplet> Kerberos<tools> VULID<vulnerability> has</s>
{'precision': 0.5, 'recall': 1.0}

In November 2019, we published a blog analyzing an exploit kit we named Capesand that exploited Adobe Flash and Microsoft Internet Explorer flaws.
Expected:
<triplet>Capesand<malware>Adobe Flash<software>exploits<triplet>Capesand<malware>Microsoft Internet Explorer<software>exploits
model out:
<pad><triplet> Capesand<malware> Adobe Flash<software> targets<triplet> Capesand<malware> Microsoft Internet Explorer<software> targets</s>
{'precision': 0, 'recall': 0, 'f1': 0}

One of the RTF files was found in a server related to Confucius (SHA2, detected as TROJ_CVE201711882.AG), while the other one (SHA2, detected as Mal_CVE20170199-2) downloaded a VB Backdoor that pings back to twitck[.]com, a domain name belonging to Urpage.
Expected:
<triplet>SHA2<hash>Confucius<threat-actor>authored-by<triplet>TROJ_CVE201711882.AG<malware>SHA2<hash>related-to<triplet>TROJ_CVE201711882.AG<malware>Confucius<threat-actor>authored-by<triplet>SHA2<hash>Confucius<threat-actor>authored-by<triplet>Mal_CVE20170199-2<malware>Confucius<threat-actor>authored-by<triplet>Mal_CVE20170199-2<malware>twitck[.]com<url>beacons-to<triplet>Urpage<threat-actor>twitck[.]com<url>controls
model out:
<pad><triplet> SHA2<hash> TROJ_CVE201711882.AG<malware> indicates<triplet> Confucius<threat-actor> TROJ_CVE20170199-2<malware> uses<triplet> TROJ_CVE20170199-2<malware> Urpage<threat-actor> related-to</s>
{'precision': 0, 'recall': 0, 'f1': 0}

malware Loader Backdoor.Win64.BIOPASS.A SHA2 Silverlight_ins.exe
Expected:
<triplet>malware<malware>Backdoor.Win64.BIOPASS.A<malware>duplicate-of<triplet>SHA2<hash>Backdoor.Win64.BIOPASS.A<malware>indicates<triplet>Silverlight_ins.exe<filepath>Backdoor.Win64.BIOPASS.A<malware>indicates<triplet>Silverlight_ins.exe<filepath>SHA2<hash>duplicate-of
model out:
<pad><triplet> malware<malware> Backdoor.Win64.BIOPASS.A<malware> duplicate-of<triplet> Silverlight_ins.exe<filepath> SHA2<hash> duplicate-of<triplet> Silverlight_ins.exe<filepath> Backdoor.Win64.BIOPASS.A<malware> indicates<triplet> SHA2<hash> Backdoor.Win64.BIOPASS.A<malware> indicates</s>
{'precision': 1.0, 'recall': 1.0}

malware Loader Backdoor.Win64.BIOPASS.A SHA2 flash_installer.exe
Expected:
<triplet>flash_installer.exe<filepath>Backdoor.Win64.BIOPASS.A<malware>indicates<triplet>SHA2<hash>Backdoor.Win64.BIOPASS.A<malware>indicates<triplet>malware<malware>Backdoor.Win64.BIOPASS.A<malware>duplicate-of<triplet>SHA2<hash>flash_installer.exe<filepath>duplicate-of

model out:
<pad><triplet> malware<malware> Backdoor.Win64.BIOPASS.A<malware> duplicate-of<triplet> SHA2<hash> Backdoor.Win64.BIOPASS.A<malware> indicates<triplet> flash_installer.exe<filepath> SHA2<hash> duplicate-of<triplet> flash_installer.exe<filepath> Backdoor.Win64.BIOPASS.A<malware> indicates</s>
{'precision': 0.75, 'recall': 0.75}

One recent campaign from a group tracked by CERT-UA as UAC-0098 delivered malicious documents with the Follina exploit in password-protected archives, impersonating the State Tax Service of Ukraine.
Expected:
<triplet>UAC-0098<threat-actor>State Tax Service of Ukraine<identity>impersonates<triplet>UAC-0098<threat-actor>Follina exploit<vulnerability>uses
model out:
<pad><triplet> UAC-0098<threat-actor> Follina exploit<vulnerability> exploits<triplet> UAC-0098<threat-actor> State Tax Service of Ukraine<identity> impersonates</s>
{'precision': 0.5, 'recall': 0.5}

They can be used to compromise an endpoint in the following way: This functionality was leveraged in exploitation of VULID − a Microsoft Office/WordPad remote code execution (RCE) vulnerability patched by Microsoft in September 2017, and was used in multiple campaigns, like this OilRig campaign.
Expected:
<triplet>Microsoft Office<tools>VULID<vulnerability>has<triplet>WordPad<tools>VULID<vulnerability>has<triplet>OilRig campaig<campaign>VULID<vulnerability>exploits
model out:
<pad><triplet> Microsoft Office<software> VULID<vulnerability> has<triplet> VULID<vulnerability> remote code execution (RCE)<attack-pattern> uses</s>
{'precision': 0, 'recall': 0, 'f1': 0}

This post is also available in: æÛčæIJňèld (Japanese) malware 2.0 is ransomware as a service (RaaS) that first emerged in June 2021 as an upgrade to its predecessor malware (aka ABCD Ransomware), which was first observed in September 2019.
Expected:
<triplet>malware 2.0<malware>malware<malware>variant-of<triplet>ABCD Ransomware<malware>malware<malware>duplicate-of
model out:
<pad><triplet> malware 2.0<malware> ABCD Ransomware<malware> duplicate-of<triplet> malware 2.0<malware> September 2019<time> related-to</s>
{'precision': 0, 'recall': 0, 'f1': 0}

Kuluoz is a descendant of the Asprox malware and spreads by sending copies of itself as an e-mail attachment.
Expected:
<triplet>Kuluoz<malware>Asprox<malware>variant-of<triplet>Asprox<malware>sending copies of itself as an e-mail attachment<attack-pattern>uses<triplet>Kuluoz<malware>sending copies of itself as an e-mail attachment<attack-pattern>uses
model out:
<pad><triplet> Kuluoz<malware> Asprox<malware> variant-of</s>
{'precision': 1.0, 'recall': 0.3333333333333333}

In the blog describing Operation Earth Kitsune, we noted sites using the GNUBoard Content Management System (CMS) had been compromised and were used to host malware.
Expected:
<triplet>malware<malware>GNUBoard Content<software>compromises<triplet>GNUBoard Content<software>malware<malware>hosts
model out:
<pad><triplet> Operation Earth Kitsune<campaign> GNUBoard Content Management System (CMS)<software> hosts</s>
{'precision': 0, 'recall': 0, 'f1': 0}

```
The group was reportedly using spear-attack-pattern emails with weaponized attachments to
    exploit known vulnerabilities.
Expected:
<triplet>group<threat-actor>spear-attack-pattern<attack-pattern>uses
model out:
<pad><triplet> spear-attack-pattern<attack-pattern> known vulnerabilities<vulnerability>
    exploits</s>
{'precision': 0, 'recall': 0, 'f1': 0}

The latest development in the ransomware world is malware 2.0, a new version of this malware
    family that uses the Tor network for command and control.
Expected:
<triplet>malware 2.0<malware>Tor<tools>uses
model out:
<pad><triplet> malware 2.0<malware> Tor<tools> uses<triplet> malware 2.0<malware> command and
    control<attack-pattern> uses</s>
{'precision': 0.5, 'recall': 1.0}

 Technically speaking, we have observed LockBit 2.0 affiliates leveraging the following tactics
     , techniques and procedures: Windows SysInternals PsExec has been utilized for both
     persistence and execution purposes.
Expected:
<triplet>LockBit 2.0<threat-actor>Windows SysInternals PsExec<tools>uses
model out:
<pad><triplet> LockBit 2.0<threat-actor> Windows SysInternals PsExec<tools> uses</s>
{'precision': 1.0, 'recall': 1.0}

The injected commands worth mentioning that intended to further download payloads to a
    vulnerable machine can be categorized into the following broad categories: We observed
    several instances of VULID being exploited to drop variants of the Mirai malware.
Expected:
<triplet>Mirai malware<malware>VULID<vulnerability>exploits
model out:
<pad><triplet> Mirai malware<malware> VULID<vulnerability> exploits<triplet> Mirai malware<
    malware> VULID<vulnerability> exploits</s>
{'precision': 0.5, 'recall': 1.0}

It is important to note that previous analysis of suspected APT activities from different
    security vendors date from 2016, and the captured samples for the Operation Earth Kitsune
    have been developed recently.
Expected:
<triplet>APT<threat-actor>2016<time>related-to
model out:
<pad><triplet> Operation Earth Kitsune<campaign> APT<attack-pattern> uses</s>
{'precision': 0, 'recall': 0, 'f1': 0}

Backdoor.Win64.BIOPASS.A SHA2 flash_installer.exe malware Loader Backdoor.Win64.BIOPASS.A
    SHA2 System.exe
Expected:
<triplet>SHA2<hash>malware<malware>indicates<triplet>flash_installer.exe<filepath>malware<
    malware>indicates<triplet>SHA2<hash>Backdoor.Win64.BIOPASS.A<malware>indicates<triplet>
    System.exe<filepath>Backdoor.Win64.BIOPASS.A<malware>indicates<triplet>Backdoor.Win64.
    BIOPASS.A<malware>malware<malware>duplicate-of<triplet>Backdoor.Win64.BIOPASS.A<malware>
    malware<malware>duplicate-of<triplet>SHA2<hash>flash_installer.exe<filepath>duplicate-of<
    triplet>SHA2<hash>System.exe<filepath>duplicate-of
model out:
<pad><triplet> malware<malware> Backdoor.Win64.BIOPASS.A<malware> duplicate-of<triplet>
    flash_installer.exe<filepath> SHA2<hash> duplicate-of<triplet> SHA2<hash> Backdoor.Win64.
    BIOPASS.A<malware> indicates<triplet> flash_installer.exe<filepath> Backdoor.Win64.BIOPASS
    .A<malware> indicates<triplet> SHA2<hash> Backdoor.Win64.BIOPASS.A<malware> indicates<
    triplet> flash_installer.exe<filepath> Backdoor.Win64.BIOPASS.A<malware> indicates<triplet
```

> SHA2<hash> Backdoor.Win64.BIOPASS.A<malware> indicates<triplet> SHA2<hash> Backdoor.Win64.BIOPASS.A<malware> indicates</s>
{'precision': 0.16666666666666666, 'recall': 0.125}

malware Loader Backdoor.Win64.BIOPASS.A SHA2 test-flash.exe malware Loader Backdoor.Win32.BIOPASS.A SHA2 test.exe malware Loader Backdoor.Win64.BIOPASS.A SHA2 flashplayerpp_install_cn.exe malware Loader Backdoor.Win64.BIOPASS.A SHA2 flash2.exe malware Loader Backdoor.
Expected:
<triplet>SHA2<hash>test.exe<filepath>duplicate-of<triplet>flashplayerpp_install_cn.exe<filepath>SHA2<hash>duplicate-of<triplet>flash2.exe<filepath>SHA2<hash>duplicate-of<triplet>SHA2<hash>Backdoor.Win64.BIOPASS.A<malware>indicates<triplet>test-flash.exe<filepath>Backdoor.Win64.BIOPASS.A<malware>indicates<triplet>SHA2<hash>Backdoor.Win32.BIOPASS.A<malware>indicates<triplet>test.exe<filepath>Backdoor.Win32.BIOPASS.A<malware>indicates<triplet>malware<malware>Backdoor.Win64.BIOPASS.A<malware>duplicate-of<triplet>test-flash.exe<filepath>SHA2<hash>duplicate-of<triplet>malware<malware>Backdoor.Win32.BIOPASS.A<malware>duplicate-of<triplet>malware<malware>Backdoor.Win64.BIOPASS.A<malware>duplicate-of<triplet>malware<malware>Backdoor.Win64.BIOPASS.A<malware>duplicate-of<triplet>flashplayerpp_install_cn.exe<filepath>Backdoor.Win64.BIOPASS.A<malware>indicates<triplet>SHA2<hash>Backdoor.Win64.BIOPASS.A<malware>indicates<triplet>SHA2<hash>Backdoor.Win64.BIOPASS.A<malware>indicates<triplet>flash2.exe<filepath>Backdoor.Win64.BIOPASS.A<malware>indicates
model out:
<pad><triplet> malware<malware> Backdoor.Win64.BIOPASS.A<malware> duplicate-of<triplet> malware<malware> Backdoor.Win32.BIOPASS.A<malware> duplicate-of<triplet> malware<malware> Backdoor.Win64.BIOPASS.A<malware> duplicate-of<triplet> malware<malware> Backdoor.Win64.BIOPASS.A<malware> duplicate-of<triplet> malware<malware> Backdoor.Win64.BIOPASS.A<malware> duplicate-of<triplet> malware<malware> Backdoor.Win64.BIOPASS.A<malware> duplicate-of<triplet> test-flash.exe<filepath> SHA2<hash> duplicate-of<triplet> test-flash.exe<filepath> SHA2<hash> duplicate-of<triplet> flashplayerpp_install_install_install_install_install_install_install_install_install_install_install_install_i

{'precision': 0.8333333333333334, 'recall': 0.3125}

Figure 6: Decompiled Form structure of ConfuciusâĂŹ sample (SHA2, Detected as TSPY_CONFSTEAL.A)
Figure 7:
Expected:
<triplet>SHA2<hash>TSPY_CONFSTEAL.A<malware>indicates
model out:
<pad><triplet> SHA2<hash> TSPY_CONFSTEAL.A<malware> indicates<triplet> Confucius<threat-actor> TSPY_CONFSTEAL.A<malware> uses</s>
{'precision': 0.5, 'recall': 1.0}

Similarly, Russian observed disinformation efforts are also focused on the war in Ukraine and TAG has disrupted coordinated influence operations from several actors including the Internet Research Agency and a Russian consulting firm as detailed in the TAG Bulletin.
Expected:
<triplet>Internet Research Agency<threat-actor>disinformation<attack-pattern>uses<triplet>Russian consulting firm<threat-actor>disinformation<attack-pattern>uses<triplet>Internet Research Agency<threat-actor>Russian<location>located-at<triplet>Russian consulting firm<threat-actor>Russian<location>located-at<triplet>Internet Research Agency<threat-actor>Ukraine<location>targets<triplet>Russian consulting firm<threat-actor>Ukraine<location>targets
model out:
<pad><triplet> Russian<location> Ukraine<location> targets<triplet> Internet Research Agency<identity> Ukraine<location> targets</s>
{'precision': 0, 'recall': 0, 'f1': 0}

ChessMaster can utilize any of these methods to download the next malware in the chain, the open attack-pattern post-exploitation tool known as âĂIJKoadic,âĂİ which the previous campaign also used.
Expected:

<triplet>ChessMaster<malware>Koadic<malware>downloads
model out:
<pad><triplet> ChessMaster<malware> Koadic<malware> uses</s>
{'precision': 0, 'recall': 0, 'f1': 0}

At the collection phase, Carbanak and FIN7 campaigns harvest attack-pattern sources and through input and attack-pattern (as performed in a related campaign using the Tirion malware).
Expected:
<triplet>Carbanak<threat-actor>FIN7<threat-actor>related-to<triplet>Carbanak<threat-actor> harvest attack-pattern sources<attack-pattern>uses<triplet>Carbanak<threat-actor>attack-pattern<attack-pattern>uses
model out:
<pad><triplet> Carbanak<threat-actor> FIN7<threat-actor> related-to<triplet> FIN7<threat-actor> harvest attack-pattern<attack-pattern> uses<triplet> FIN7<threat-actor> Tirion malware<malware> uses</s>
{'precision': 0.3333333333333333, 'recall': 0.3333333333333333}

Recently observed IPs used in Curious Gorge campaigns: COLDRIVER, a Russian-based threat actor sometimes referred to as Calisto, has launched credential attack-pattern campaigns, targeting several US based NGOs and think tanks, the military of a Balkans country, and a Ukraine based defense contractor.
Expected:
<triplet>military<identity>Balkans country<location>located-at<triplet>NGOs<identity>US<location>located-at<triplet>think tanks<identity>US<location>located-at<triplet>defense contractor<identity>Ukraine<location>located-at<triplet>Calisto<threat-actor>credential attack-pattern<attack-pattern>uses<triplet>Calisto<threat-actor>NGOs<identity>duplicate-of<triplet>Calisto<threat-actor>think tanks<identity>duplicate-of<triplet>Calisto<threat-actor>military<identity>duplicate-of<triplet>Calisto<threat-actor>defense contractor<identity>duplicate-of<triplet>Calisto<threat-actor>COLDRIVER<campaign>related-to
model out:
<pad><triplet> COLDRIVER<threat-actor> Russian<location> located-at<triplet> Calisto<threat-actor> COLDRIVER<threat-actor> duplicate-of<triplet> COLDRIVER<threat-actor> credential attack-pattern campaigns<attack-pattern> uses<triplet> COLDRIVER<threat-actor> US based NGOs and think tanks<identity> targets<triplet> COLDRIVER<threat-actor> military of a Balkans country<identity> targets<triplet> COLDRIVER<threat-actor> Ukraine based defense contractor<identity> targets</s>
{'precision': 0, 'recall': 0, 'f1': 0}

Overview Palo Alto Networks Unit 42 threat research team has just uncovered a new set of attacks by the Sofacy group using malicious emails targeting foreign affairs agencies and ministries in North America and Europe, including a European embassy in Moscow.
Expected:
<triplet>Sofacy group<threat-actor>malicious emails<attack-pattern>uses<triplet>European embassy<identity>Moscow<location>located-at<triplet>Sofacy group<threat-actor>European embassy<identity>targets<triplet>Sofacy group<threat-actor>North America<location>targets<triplet>Sofacy group<threat-actor>Europe<location>targets
model out:
<pad><triplet> Sofacy group<threat-actor> malicious emails<attack-pattern> uses<triplet> Sofacy group<threat-actor> North America<location> targets<triplet> Sofacy group<threat-actor> European embassy in Moscow<identity> targets</s>
{'precision': 0.6666666666666666, 'recall': 0.4}

If the storeâĂŹs e-commerce system is vulnerable to attack-pattern attacks, the malicious script will be loaded and executed on the merchantâĂŹs management panel once someone (like a system administrator or store employee) opens the said order.
Expected:
<triplet>malicious script<malware>management panel<software>targets<triplet>malicious script<malware>attack-pattern<attack-pattern>uses
model out:
<pad><triplet> malicious script<malware> merchantâĂŹs management panel<software> targets<triplet> merchantâĂŹs e-commerce system<software> attack-pattern<attack-pattern> uses</s>
{'precision': 0, 'recall': 0, 'f1': 0}

```
Recently observed Ghostwriter credential attack-pattern domains and emails: Curious Gorge, a
    group TAG attributes to China's PLA SSF, has remained active against government, military,
     logistics and manufacturing organizations in Ukraine, Russia and Central Asia.
Expected:
<triplet>Curious Gorge<threat-actor>PLA SSF<identity>attributed-to
model out:
<pad><triplet> Curious Gorge<threat-actor> China's PLA SSF<identity> attributed-to<triplet>
    Curious Gorge<threat-actor> Ukraine<location> targets<triplet> Curious Gorge<threat-actor>
     Russia<location> targets<triplet> Curious Gorge<threat-actor> Central Asia<location>
    targets</s>
{'precision': 0, 'recall': 0, 'f1': 0}

 Threat Brief: Ongoing Russia and Ukraine Cyber Conflict (Jan. 20) RussiaâĂŹs Gamaredon aka
     Primitive Bear APT Group Actively Targeting Ukraine (Updated Feb. 16)
Expected:
<triplet>Gamaredon<threat-actor>Primitive Bear<threat-actor>duplicate-of<triplet>Primitive Bear
    <threat-actor>RussiaâĂŹ<location>originates-from<triplet>Gamaredon<threat-actor>RussiaâĂŹ<
    location>originates-from
model out:
<pad><triplet> Gamaredon<threat-actor> Russia<location> located-at<triplet> Gamaredon<threat-
    actor> Ukraine<location> targets</s>
{'precision': 0, 'recall': 0, 'f1': 0}

 malware Loader  Backdoor.Win64.BIOPASS.A  SHA2
Expected:
<triplet>malware<malware>Backdoor.Win64.BIOPASS.A<malware>duplicate-of<triplet>SHA2<hash>
    Backdoor.Win64.BIOPASS.A<malware>indicates
model out:
<pad><triplet> malware<malware> Backdoor.Win64.BIOPASS.A<malware> duplicate-of<triplet> SHA2<
    hash> Backdoor.Win64.BIOPASS.A<malware> indicates<triplet> malware<malware> Backdoor.Win64
    .BIOPASS.A<malware> duplicate-of</s>
{'precision': 1.0, 'recall': 1.0}

After analyzing the ChiserClient, we found that it shares a similar style of network
    configuration to the malware mentioned in our previous research on Earth Centaur.
Expected:
<triplet>malware<malware>Earth Centaur.<threat-actor>authored-by<triplet>ChiserClient<malware>
    Earth Centaur.<threat-actor>authored-by
model out:
<pad><triplet> ChiserClient<malware> Earth Centaur<threat-actor> related-to</s>
{'precision': 0, 'recall': 0, 'f1': 0}

The PowerShell downloads, installs, and allows the remote desktop tool AnyDeskMSI through the
    firewall.
Expected:
<triplet>PowerShell<tools>AnyDeskMSI<tools>downloads
model out:
<pad><triplet> PowerShell<tools> AnyDeskMSI<tools> downloads</s>
{'precision': 1.0, 'recall': 1.0}

Similarly, Russian observed disinformation efforts are also focused on the war in Ukraine and
    TAG has disrupted coordinated influence operations from several actors including the
    Internet Research Agency and a Russian consulting firm as detailed in the TAG Bulletin.
Expected:
<triplet>Internet Research Agency<threat-actor>Ukraine<location>targets<triplet>Russian
    consulting firm<threat-actor>Ukraine<location>targets
model out:
<pad><triplet> Russian<location> Ukraine<location> targets<triplet> Internet Research Agency<
    identity> Ukraine<location> targets</s>
{'precision': 0, 'recall': 0, 'f1': 0}
```

Analyzing the Marade downloader (detected by Trend Micro as TSPY_MARADE.ZTBC) and the version of Protux (BKDR_PROTUX.ZTBC) employed by BlackgearâĂŹs latest campaigns, we found their encrypted configurations on blog and social media posts (see Figure 1).
Expected:
<triplet>TSPY_MARADE.ZTBC<malware>Marade downloader<malware>duplicate−of<triplet>Protux (BKDR_PROTUX.ZTBC)<malware>Marade downloader<malware>based−on<triplet>Marade downloader<malware>Blackgear<threat−actor>authored−by
model out:
<pad><triplet> Blackgear<threat−actor> TSPY_MARADE.ZTBC<malware> uses<triplet> TSPY_MARADE.ZTBC<malware> Protux<malware> variant−of</s>
{'precision': 0, 'recall': 0, 'f1': 0}

malware isnâĂŹt the only threat that communicates over Tor and if your network doesnâĂŹt have an explicit reason to allow anonymization networks, you should consider blocking the application altogether with your firewall.
Expected:
<triplet>malware<malware>Tor<tools>uses
model out:
<pad><triplet> malware<malware> Tor<tools> uses</s>
{'precision': 1.0, 'recall': 1.0}

While the original campaign was comprehensive and used remote access Trojans (RATs) such as ChChes and RedLeaves, this new campaign used a new backdoor (Detected by Trend Micro as BKDR_ANEL.ZKEI) that leverages the VULID vulnerability for its cyberespionage activities.
Expected:
<triplet>BKDR_ANEL.ZKEI<malware>VULID<vulnerability>exploits
model out:
<pad><triplet> BKDR_ANEL.ZKEI<malware> VULID<vulnerability> exploits<triplet> Trend Micro<threat−actor> BKDR_ANEL.ZKEI<malware> uses</s>
{'precision': 0.5, 'recall': 1.0}

Here is a deeper look at the campaign activity TAG has observed over the past two weeks: Curious Gorge, a group TAG attributes to China's PLA SSF, has conducted campaigns against government and military organizations in Ukraine, Russia, Kazakhstan, and Mongolia.
Expected:
<triplet>Curious Gorge<threat−actor>China's PLA SSF<identity>attributed−to<triplet>Curious Gorge<threat−actor>Ukraine<location>duplicate−of<triplet>Curious Gorge<threat−actor>Russia<location>duplicate−of<triplet>Curious Gorge<threat−actor>Kazakhstan<location>duplicate−of<triplet>Curious Gorge<threat−actor>Mongolia<location>duplicate−of
model out:
<pad><triplet> Curious Gorge<threat−actor> China's PLA SSF<identity> attributed−to<triplet> Curious Gorge<threat−actor> Ukraine<location> targets<triplet> Curious Gorge<threat−actor> Russia<location> targets<triplet> Curious Gorge<threat−actor> Kazakhstan<location> targets</s>
{'precision': 0.25, 'recall': 0.2}

Instead, they were either non−specific Mirai variants or contained previously known exploits such as VULID.
Expected:
<triplet>Mirai<malware>VULID<vulnerability>exploits
model out:
<pad><triplet> Mirai<malware> VULID<vulnerability> exploits</s>
{'precision': 1.0, 'recall': 1.0}

This particular webshell is based on AntSwordâĂŹs shell.jsp, which was modified to use a parameter of k3rwin instead of ant to load the class.
Expected:
<triplet>AntSword<tools>shell.jsp<filepath>consists−of
model out:
<pad><triplet> shell.jsp<filepath> k3rwin<tools> uses</s>
{'precision': 0, 'recall': 0, 'f1': 0}

```
 The Atlassian products vulnerable to VULID are those using the following versions of
      Confluence Server and Data Center:  Confluence Cloud customers are not affected by this
      vulnerability.
Expected:
<triplet>Confluence Server and Data Center<software>VULID<vulnerability>has
model out:
<pad><triplet> Confluence Server and Data Center<software> VULID<vulnerability> has<triplet>
     Atlassian<software> VULID<vulnerability> has</s>
{'precision': 0.5, 'recall': 1.0}

malware Loader  Backdoor.Win64.BIOPASS.A  SHA2  flash-64.exe
Expected:
<triplet>flash-64.exe<filepath>SHA2<hash>duplicate-of<triplet>malware<malware>Backdoor.Win64.
     BIOPASS.A<malware>duplicate-of<triplet>flash-64.exe<filepath>Backdoor.Win64.BIOPASS.A<
     malware>indicates<triplet>SHA2<hash>Backdoor.Win64.BIOPASS.A<malware>indicates
model out:
<pad><triplet> malware<malware> Backdoor.Win64.BIOPASS.A<malware> duplicate-of<triplet> flash
     -64.exe<filepath> SHA2<hash> duplicate-of<triplet> flash-64.exe<filepath> Backdoor.Win64.
     BIOPASS.A<malware> indicates<triplet> SHA2<hash> Backdoor.Win64.BIOPASS.A<malware>
     indicates</s>
{'precision': 1.0, 'recall': 1.0}

Despite these measures, users still choose to open these files and enable their content, thus
     allowing macros to continue be a common attack vector âĂŞ both in wide and simple attacks
     to deliver ransomware such as malware, as well as for sophisticated attacks like this
     Sofacy campaign.
Expected:
<triplet>Sofacy campaign<campaign>Sofacy<threat-actor>related-to
model out:
<pad><triplet> Sofacy campaign<campaign> malware<malware> delivers<triplet> Sofacy campaign<
     campaign> Sofacy campaign<campaign> delivers</s>
{'precision': 0, 'recall': 0, 'f1': 0}

An old sample of the module (SHA2) is likely a watchdog to check the status of another module
     that is known as âĂIJc1222âĂİ.  The malicious actors can change this behavior by replacing
     the content of the cdaemon.txt service in the cloud so that when combined with the regular
      execution of the attack-pattern, the cdaemon task can behave like a backdoor.
Expected:
<triplet>SHA2<hash>c1222<malware>related-to
model out:
<pad><triplet> SHA2<hash> c1222<filepath> related-to</s>
{'precision': 0, 'recall': 0, 'f1': 0}

The name of the attack-pattern file is hardcoded to be either âĂIJec_ver.php,âĂİ âĂIJlog3.php,âĂİ
      or âĂIJtemp.php.âĂİ
Expected:
<triplet>attack-pattern<attack-pattern>ec_ver.php,<filepath>related-to<triplet>attack-pattern<
     attack-pattern>log3.php<filepath>related-to<triplet>attack-pattern<attack-pattern>temp.php
     <filepath>related-to
model out:
<pad><triplet> attack-pattern file<filepath> ec_ver.php<filepath> related-to<triplet> attack-
     pattern file<filepath> âĂIJlog3.php<filepath> related-to</s>
{'precision': 0, 'recall': 0, 'f1': 0}

 Recently Palo Alto Networks discovered a backdoor program ( md5: MD5, sha256: SHA2 ) that is
      used to control a linux operating system.
Expected:
<triplet>MD5<hash>SHA2<hash>related-to
model out:
<pad><triplet> SHA2<hash> linux operating system<software> indicates</s>
{'precision': 0, 'recall': 0, 'f1': 0}
```

```
Here is a deeper look at the campaign activity TAG has observed over the past two weeks:
    Curious Gorge, a group TAG attributes to China's PLA SSF, has conducted campaigns against
    government and military organizations in Ukraine, Russia, Kazakhstan, and Mongolia.
Expected:
<triplet>Curious Gorge<threat-actor>China's PLA SSF<identity>attributed-to<triplet>Curious
    Gorge<threat-actor>government and military organizations<identity>targets<triplet>Curious
    Gorge<threat-actor>Ukraine<location>targets<triplet>Curious Gorge<threat-actor>Russia<
    location>targets<triplet>Curious Gorge<threat-actor>Kazakhstan<location>targets<triplet>
    Curious Gorge<threat-actor>Mongolia<location>targets
model out:
<pad><triplet> Curious Gorge<threat-actor> China's PLA SSF<identity> attributed-to<triplet>
    Curious Gorge<threat-actor> Ukraine<location> targets<triplet> Curious Gorge<threat-actor>
     Russia<location> targets<triplet> Curious Gorge<threat-actor> Kazakhstan<location>
    targets</s>
{'precision': 1.0, 'recall': 0.6666666666666666}

Mimikatz components were also copied to the affected machine via AnyDeskMSI.
Expected:
<triplet>AnyDeskMSI<tools>Mimikatz<tools>drops
model out:
<pad><triplet> AnyDeskMSI<tools> Mimikatz<malware> downloads</s>
{'precision': 0, 'recall': 0, 'f1': 0}

This new campaign used updated instances of the Tdrop malware family discovered in the
    Operation malware campaign.
Expected:
<triplet>Operation malware campaign<campaign>Tdrop<malware>delivers
model out:
<pad><triplet> Operation malware campaign<campaign> Tdrop<malware> delivers</s>
{'precision': 1.0, 'recall': 1.0}

This post is also available in:      æŮĕæIJňèld (Japanese)   On Aug. 25, 2021, Atlassian
    released a security advisory for an injection vulnerability in Confluence Server and Data
    Center, VULID.
Expected:
<triplet>Confluence Server and Data Center<software>VULID<vulnerability>has
model out:
<pad><triplet> Confluence Server and Data Center<software> VULID<vulnerability> has</s>
{'precision': 1.0, 'recall': 1.0}

We already mentioned that Confucius had possible links to other groups in our previous blog
    post, which mentioned code sharing between Patchwork and Confucius.
Expected:
<triplet>Patchwork<threat-actor>Confucius<threat-actor>related-to
```

Listing A.6: *Error analisis over the whole validation dataset*