1.5em 0pt

Fake Document Generation for Cyber Deception

A project report submitted in partial fulfilment of the requirements for
SUMMER INTERNSHIP

by

N. Kathiravan

Department of Computer Science and Engineering

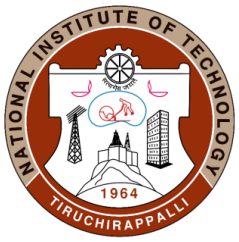
National Institute of Technology Puducherry

under the supervision of

Dr. Ghanshyam S. Bopche

Assistant Professor

Department of Computer Applications



National Institute of Technology

Tiruchirappalli India 620015

June - 2024

Bonafide Certificate

This is to certify that the project "**Fake Document Generation for Cyber Deception**" is a project work successfully done by *N. Kathiravan* (Department of Computer Science and Engineering National Institute of Technology Puducherry) in partial fulfillment of the requirements for the summer internship from National Institute of Technology, Tiruchirappalli, during the Summer Internship, June-2024.

Dr. Ghanshyam S. Bopche
(Project Guide)

Prof. Michael Arock
(Head of the Department)

Abstract

This project investigates the generation of fake documents by applying cyber deception techniques and Natural Language Processing (NLP). The primary objective is to transform text comprehensibility to create deceptive documents that can be utilized for various cybersecurity purposes, particularly in protecting intellectual property and sensitive technical information from theft.

With the increasing sophistication of cyber-attacks, it has become evident that traditional security measures alone are insufficient. Instead of solely relying on existing security protocols, this approach adds a layer of defense through the generation of fake documents. These deceptive documents serve as decoys to mislead attackers and protect genuine intellectual property, such as patents and proprietary technical documents like research papers.

The approach leverages three key metrics: sequentiality, connectivity, and dispersion. Sequentiality ensures that the generated text follows a logical order, mimicking the flow of genuine documents. Connectivity refers to the coherence and consistency of the text, ensuring that ideas and sentences are logically linked. Dispersion measures the distribution of deceptive elements across the document, balancing between believable and misleading content.

Our methodology includes advanced NLP techniques for text generation and manipulation, aiming to produce documents that are not easily distinguishable from legitimate ones to a casual observer, yet contain strategically placed inaccuracies. The main idea behind generating these documents lies in their potential to act as decoys in cybersecurity environments, thereby confusing and delaying attackers, and enhancing the training of security personnel to recognize and handle deceptive content. We can also add some kind of canary tokens to be triggered when the fake documents are opened since the correct user does not need to open the fake ones as he knows everything about his system.

The evaluation of the generated documents is found through the change in the values of the three key metrics. The results demonstrate that manipulating text comprehensibility using these metrics makes it possible to create convincing fake documents that significantly contribute to the field of cyber deception and cybersecurity defense, particularly in safeguarding intellectual property from theft.

Acknowledgments

I would like to express my deepest gratitude to several individuals who have contributed significantly to the successful completion of this project.

Firstly, I extend my heartfelt thanks to Dr. G. Aghila, Director of the National Institute of Technology, Tiruchirapalli, for providing an exceptional environment conducive to academic and research excellence during my internship.

I am also profoundly grateful to Dr. Michael Arock, Head of the Department of Computer Applications at the National Institute of Technology, Tiruchirapalli, for his unwavering support and encouragement throughout the project.

A special thanks goes to my guide, Dr.Ghanshyam S. Bopche, whose expert guidance, insightful feedback, and continuous support were instrumental in the successful execution of this project. His mentorship has been invaluable, and his extensive knowledge and expertise have profoundly influenced my approach to research and problem-solving. I deeply value his patience, encouragement, and the hours he devoted to helping me refine and improve my work.

Additionally, I would like to acknowledge Nilin Prabhaker, who is pursuing his PhD under the guidance of Dr.Ghanshyam S. Bopche, for his invaluable assistance and support. His help was crucial during the various stages of this project, and I deeply value his contributions. His willingness to share his knowledge and his readiness to assist with complex technical challenges greatly enhanced the quality of my work. His collaborative spirit and constructive feedback were invaluable assets throughout this journey.

Thank you all for your guidance, support, and encouragement, which have been fundamental to the completion of this project.

Table of Contents

List of Figures

List of Tables

List of Algorithms

Chapter 1

Introduction

The sophistication of cyber-attacks, especially within the modern digital environment, increasingly puts intellectual property and sensitive information at risk. With the rapid growth in technology and the spread of the internet, even greater avenues are opened up that cyber-criminals take advantage of to inflict damage and perpetrate theft of valuable data. One of the largest threats is the theft of technical documents like patents, private studies, and other kinds of intellectual property. More often than not, such documents represent years of research and enormous financial investment in their making, consequently, becoming very important to an organization.

Though found in firewalls, encryption, and intrusion detection systems, foundational steps have been made in securing digital assets. However, none of these can directly intertwine with security given the sophistication and persistence of cyber-attacks nowadays. More attackers now bypass traditional defenses in searching for innovative and proactive strategies for safeguarding sensitive information. In this way, cyber deception has thus been one promising approach toward improving an organization's security posture.

Cyber deception refers to the process of creating deceptive artifacts and luring, confusing, and slowing threat actors. The ones specifically notable in this regard are fake networks and bogus databases, notably, counterfeit documents. An organization in that way, by seeding the digital environment with deceptive elements, can create traps and diversions covering the genuine assets while concurrently collecting intelligence about the ways and intentions of the attackers. The concept of cyber deceptions is based on changing the asymmetric power balance in a way that makes it harder and riskier for attackers to meet their targets. The current project deals with the generation of fake technical documents; using state-of-the-art techniques of NLP as a means for cyber deception. This deals with the creation of documents that might be convincing to the casual observer but have inserted inaccuracies designed to mislead and thwart an attacker. It gives a protective deception for real intellectual property from theft or misuse, using mainly

three metrics: sequentiality, connectivity, and dispersion.

Sequentiality in this context describes how logically the text flows in order within the document. Virtually all authentic technical documents contain a clean, clear chain of thoughts throughout the document. We would like to be able to lower sequentiality in our fake documents. In other words, this means that the text might logically fall apart upon closer inspection, and not quite coherent. These disturbances are designed to appear minor, perhaps insignificant at first glance, yet accumulate to make the document less usable and trustworthy.

Connectivity refers to how the text is coherent and consistent. Ideas and sentences in a real document are logically connected; otherwise, forming a cohesive narrative. In the fake documents, connectivity is reduced as well. This ends up with content that is in bits and pieces, where ideas and sentences do not quite seem to be running into each other as smoothly as possible. Lessened connectivity ensures that, while the document may appear credible, it works poorly as a credible source of information.

Dispersion refers to the act of spreading the conceptual elements throughout the document. The elements of deception are spread throughout the document, rather than compact in one area, for ease of identification since clustering inaccuracies yield easy identification. When we increase dispersion, we increase the number of contexts with real information contained in fake documents that appear convincing but have inaccuracies subtly built into them. This balancing act therefore maximizes the effectiveness of the deception since an attacker has to go through the additional effort of sifting through a document only at the end to find it unreliable.

These methods of fake document generation hence utilize advanced NLP techniques that allow manipulation and transforming text. A point that has been taken into consideration within this project is that in itself, NLP is a sub-domain of artificial intelligence that deals with the interaction between computers and human language. Within this project, NLP techniques are hence applied to generate and modify the text to cause the desired levels of decreased sequentiality, decreased connectivity, and increased dispersion.

The project can enhance cybersecurity at various levels. Layers of document-based deceptions can be used by organizations to further support their security posture. These would have been the first line of defense against such attacks, thereby warning organizations for real intellectual property. In addition, cyber deceptions may uncover the behaviors and strategies of attackers, thus helping an organization strengthen its security measures in all aspects.

The methodology developed within the project can be then applied to other classes of digital

artifacts, hence expanding the realm of cyber deception strategies. In that respect, the sophistication of cyber-attacks requires the search for innovative and proactive security strategies. Therefore, the generation of fake technical documents using cyber deception and NLP techniques will represent a promising approach to protecting intellectual property and enhancing overall cybersecurity. In this project, lower sequentiality, connectivity, and higher dispersion metrics will ensure that convincing decoys are generated to throw the attacker off the path to the real assets. It will be instrumental in enabling the whole cyber-security community to learn lessons from its findings on the concepts of deception and their practical applications in scenarios.

Chapter 2

Literature Review

2.1 Fake Document Generation for Cyber Deception by Manipulating Text Comprehensibility

This research paper is being used as the base paper for this project. This paper discusses the methodology of generating fake documents by manipulating the text comprehensibility using a genetic algorithm. This paper suggests three measures for a document to be evaluated as fit sequentiality, connectivity, and dispersion it also gives us three ways of modifying the document such as addition, deletion, and shuffling.

2.2 Generation of Believable Fake Logic Circuits for Cyber Deception

This paper focuses on technical, scientific documents or patents often containing logical circuits. this paper presents a Fake Boolean Logic Circuit Generation Engine (FBLCGE) to generate a fake version of logical design. Multiple believable decoy documents can be generated by replacing the original logic circuit with a believable fake version.

2.3 Three decades of deception techniques in active cyber defense - retrospect and outlook

This paper is a theoretical paper explaining cyber deception in detail. It explains about the cyber kill chain model, deception lifecycle, etc., This paper also speaks about moving target defense, honey pots, and honey tokens. As this is a theoretical paper it does not include any implementations.

2.4 Honeyfiles: deceptive files for intrusion detection

This paper introduces an intrusion-detection device named honeyfiles. Honeyfiles are bait files intended for hackers to access. The files reside on a file server, and the server sends an alarm when a honey file is accessed.

2.5 Controllable fake document infilling for cyber deception

This paper proposes a novel context-aware Document Infilling (FDI) model by converting the problem to a controllable mask-then-infill procedure. They use the FORGE framework and fine-tuning the language model for text infilling.

2.6 Deception techniques in computer security: A research perspective

This paper involves a theoretical approach to cyber deception techniques. It speaks about the modeling, mode of deployment, the interaction between attackers, and deception techniques.

2.7 Gait: A game-theoretic defense against intellectual property theft

The goal of this paper is to impose as much cost on the adversary (e.g. delay the attacker as much as possible, maximize mistaken identification of the real document), given the attacker's approach for finding the real document.

2.8 Generating fake documents using probabilistic logic graphs

This paper proposes the concept of a Probabilistic Logic Graph (PLG) and shows that PLGs provide a single, unified framework within which the different parts of a document can be expressed.

Table 2.1: Literature summary

SR No	Reference	Technique	Limitations	Future Work
1	Karuna et al. [1]	Genetic algorithm	Modifies only the target concepts	identify semantically related rephrases of a target concept to manipulate their occurrences
2	Prabhaker et al. [2]	FBEGE Algorithm	FBEGE doesn't consider the cost of applying different edit operations to generate a believable fake Boolean equation	further experiments need to be conducted to test the efficacy and applicability of the FBEGE in generating believable fake equations for Boolean equation containing more input variables
3	Zhang and Thing [3]	Honeytokens, honeypots, etc.	No practical implementations	no future works
4	Yuill et al. [4]	Honey files	Honeyfiles are appropriate for file spaces that are accessible to one person or a small group	automatically updating a file's deceptive content
5	Hu et al. [5]	text infilling	less overlapping experiments between reviewers to calculate Kappa, FDI is not flawless and suffers from similar weaknesses as all LMs.	extend its applications to other critical domains, such as political science
6	Han et al.[6]	cyber deception research	no practical implementations	no future works
7	Zhang et al.[7]	GAIT	assumed a perfect adversary, one who makes no mistakes when examining a unit and declaring it to be real or fake	An important extension would build on ideas such as PLGs and conduct a game-theoretic analysis in the multimodal case.
8	Han et al.[8]	Probabilistic Logic Graphs	capability of PLG representing OCR not explained	exploring the uncertainty of ocr like tools

Chapter 3

Methodology

3.1 Data Collection and Preprocessing

3.1.1 Data Collection

The dataset of this project completely focuses on the technical documents. The first document set consists of 10 documents each from Wikipedia, CACM, and security encyclopedia. The next document set was created by using the introduction part of many research papers published in international conferences and journals.
The research papers cover a wide range of topics in the field of computer science. The topics include attack graph generation, bioinformatics, cybersecurity, fake document generation, machine learning and deep learning, and quantum computing. These document sets were used as input for the genetic algorithm for the generation of believable fake documents.

3.1.2 Preprocessing

The document sets were preprocessed and did not contain any uncertainty, so there was no necessity of preprocessing.

3.2 The Genetic Algorithm

The genetic algorithm comprises several components, which are depicted in the architecture shown in the fig.3.1.

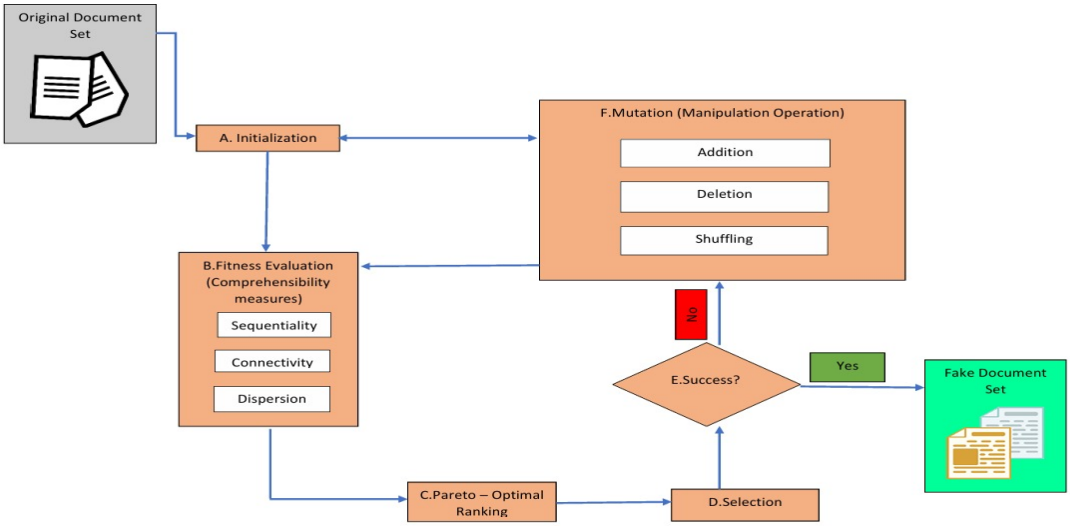


Figure 3.1: Genetic Algorithm working

The genetic algorithm has 6 main parts. They are:

1.Initialization

2.Fitness function

3.Ranking

4.Selection

5.Stopping criteria

6.Mutation

The genetic algorithm is given in algorithm 3.1

3.2.1 Initialization

The initialization function uses the three manipulation techniques for the initialization of the required document generation. The initialization function creates the number of required fake documents to be generated initially, then those are used as parent document set.

Algorithm 3.1 Multi-Objective, Multi-Mutation Genetic Algorithm

Require: $d, N, ng, \alpha, \beta, \gamma$
Ensure: D

procedure MULTIOBJECTIVE_MULTIMUTATION_GA
    $P_{\text{initial}} \leftarrow \text{Initialization}(d)$
    $P_{\text{initial}} \leftarrow \text{Fitness\_function}(P_{\text{initial}})$
    $P_{\text{initial}} \leftarrow \text{Ranking}(P_{\text{initial}})$
    $P_{\text{parent}} \leftarrow \text{Selection}(P_{\text{initial}}, N)$
    $nGen \leftarrow 0$
    while not $\text{Check}(P_{\text{parent}}, \alpha, \beta, \gamma)$ and $nGen < ng$ **do**
        $P_{\text{children}} \leftarrow \emptyset$
        for $i \leftarrow 0$ to $|P_{\text{parent}}|-1$ **do**
            $P_{\text{children}} \leftarrow P_{\text{children}} + \text{Mutation}(P_{\text{parent}}.d[i], i \mod 3)$
        end for
        $P_{\text{children}} \leftarrow \text{Fitness\_function}(P_{\text{children}})$
        $P_{\text{parent}} \leftarrow P_{\text{parent}} + P_{\text{children}}$
        $P_{\text{parent}} \leftarrow \text{Ranking}(P_{\text{parent}})$
        $P_{\text{parent}} \leftarrow \text{Selection}(P_{\text{parent}}, N)$
        $nGen \leftarrow nGen + 1$
    end while
    $D \leftarrow P_{\text{parent}}$
end procedure

3.2.2 Fitness function

The Fitness function uses three metrics for finding fitness scores and appends it to the document. The fitness scores would later be used for ranking. These metrics involved in fitness function are

1.sequentiality

2.connectivity

3.dispersion

3.2.2.1 Sequentiality

When concepts in a document are not explained sequentially, i.e., if concept c is used to explain a related concept c' before c is explained adequately, this adds a comprehension burden on the reader to first understand c before understanding c'.

Sequentiality(d) $= -\sum_{c \in C} \Psi(c)$

where, $\Psi(c)$ represents the comprehension burden of the concept

3.2.2.2 Connectivity

When concepts in a document are well-connected to each other, i.e., when the document is written such that it relates to the different concepts in the document, the document is easier to comprehend.

Connectivity(d) = $\frac{\sum_{c \in C} \phi(c)}{|C|}$

where $\phi(c)$ is connectivity measure of each concept

3.2.2.3 Dispersion

In a well-written document, each discourse structure should have a specific focus. Lack of focus can make the discourse hard to comprehend as the reader has to remember more conceptual information.

Dispersion(d)=$\frac{\sum_{1 \leq k \leq K} \theta(p_k)}{K}$

where, $\theta(p_k)$ is the dispersion of concepts in p_k based on Shannon entropy

3.2.3 Ranking

The ranking function ranks the documents in the population using Pareto-optimality considering the sequentiality, connectivity, and dispersion simultaneously to provide the non-dominating solution. When two documents have the same Pareto-optimal ranking, the tie is broken based on their crowding distance which is a measure of the diversity of a population for GA. It is computed based on the density of solutions on the Pareto-frontier, where the solutions located in less dense regions are preferred.

3.2.4 Selection

The Selection function selects the top N documents from the population based on the ranking, where N is the number of required documents.

3.2.5 Stopping criteria

The GA stops after ensuring the following conditions. 1) A statistically significant change in the comprehensibility measures of a generated fake document d' in the current population P

compared to the legit d, 2)Preserving the similarity of the believability of d' to that of d, because too many mutations on d could render a generated d' unreadable and unbelievable, which leads to setting the limit of ng number of iterations for GA.

3.2.6 Mutation

The Mutation function uses the three manipulation operations where operation_id is 0 for Addition, 1 for Deletion, and 2 for Shuffling. It applies an operation on d in the current population Pparent where the operation_id is determined by the remainder of the modulo operator of 3 applied on the ranking index of d in Pparent.

The operations 1) Addition—identifies, duplicates, and strategically adds a selected set of hard-to-comprehend sentences (manufacturing reality). 2) Deletion—identifies and deletes sentences explain- ing a concept (hiding reality). 3) Shuffling—reorders sentences to manipulate the sequential ordering of concept explanations (altering reality).

The requirements of the mutated documents are

1. Sequentiality should decrease
2. Connectivity should decrease
3. Dispersion should increase

from that of the original document.

Chapter 4

Experimental Results

4.1 Setup

4.1.1 DataSet

The dataset of this project completely focuses on the technical documents. The first document set consists of 10 documents each from Wikipedia, CACM, and security encyclopedia. The next document set was created by using the introduction part of many research papers published in international conferences and journals.
The research papers cover a wide range of topics in the field of computer science. The topics include attack graph generation, bioinformatics, cybersecurity, fake document generation, machine learning and deep learning, and quantum computing. These document sets were used as input for the genetic algorithm for the generation of believable fake documents.

4.1.2 Parameter

The population size(N) is taken as 50 times for all the documents and run for two generation (ng) and until the criteria for successful believable fake generation is met(Sequentiality, Connectivity, and Dispersion).

4.1.3 Hardware/software

The Hardware used for the implementation is laptop or desktop with google colab runnable. The Hardware used is Intel i7 core. The software libraries used are

4.1.3.1 python-docx

The python-docx module helps to write and create document files and acces it for reading and taking paragraphs for creating fakes.

4.1.3.2 pandas

Pandas library is fast and flexible python library chiefly used for machine learning in the form of DataFrames. This library is used here for data manipulation and analysis.

4.1.3.3 numpy

The numpy library is also used here for the the creation of multi dimensional arrays and mathematical operations efficiently.

4.1.3.4 re

The re library is the short form of regex (Regular Expression). This library helps to match the patterns in the paragraph to separate the sentences and understand the concepts.

4.1.3.5 nltk

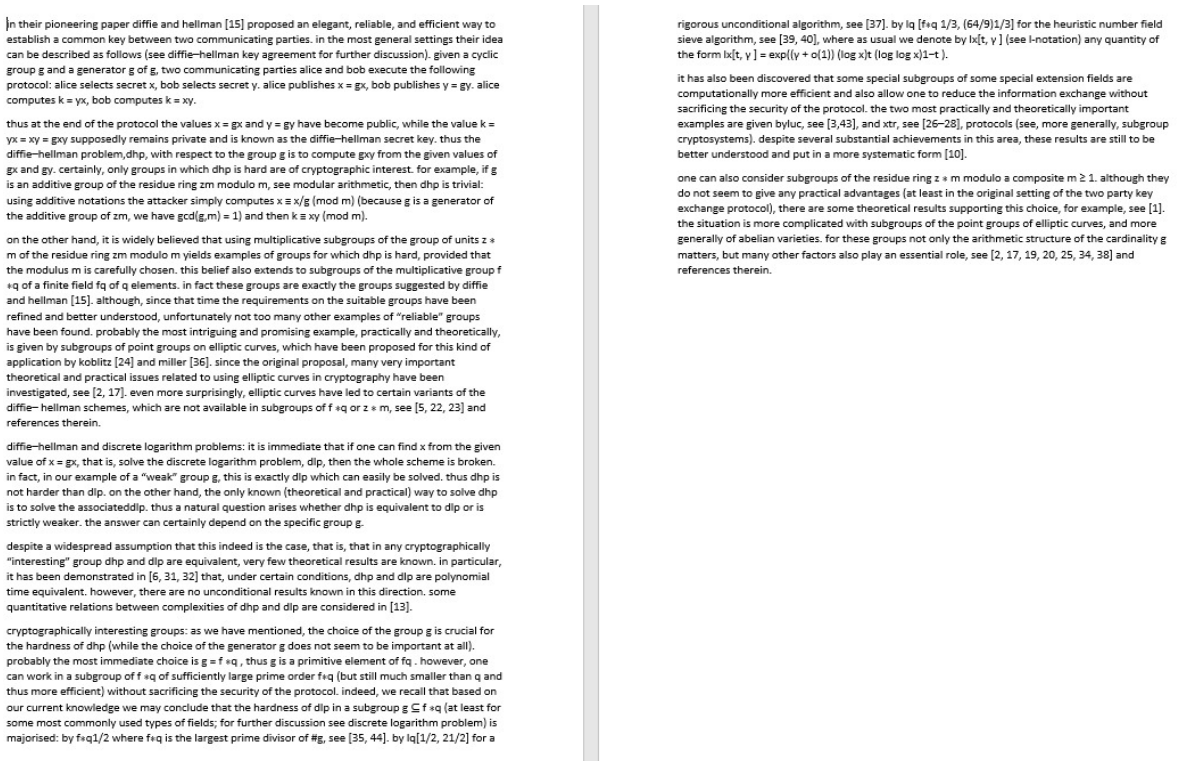
The NLTK stands for Natural Language Tool Kit. This library is used here for tokenizing the sentences as nouns, verb, adjectives, etc.,

4.1.3.6 math

This library of python is used here for accessing math functions.

4.1.3.7 sentence_transformers

This library loads pre-trained models to encode sentences/paragraphs into fixed-dimensional vectors (embeddings) for various NLP tasks. The sentence transformers used for this project is "bert-base-nli-mean-tokens".



In their pioneering paper diffie and hellman [15] proposed an elegant, reliable, and efficient way to establish a common key between two communicating parties. in the most general settings their idea can be described as follows (see diffie–hellman key agreement for further discussion). given a cyclic group g and a generator g of g, two communicating parties alice and bob execute the following protocol: alice selects secret x, bob selects secret y. alice publishes x = gx, bob publishes y = gy. alice computes k = yx, bob computes k = xy.

thus at the end of the protocol the values x = gx and y = gy have become public, while the value k = yx = xy = gxy supposedly remains private and is known as the diffie–hellman secret key. thus the diffie–hellman problem,dhp, with respect to the group g is to compute gxy from the given values of gx and gy. certainly, only groups in which dhp is hard are of cryptographic interest. for example, if g is an additive group of the residue ring zm modulo m, see modular arithmetic, then dhp is trivial: using additive notations the attacker simply computes x ≡ x/g (mod m) (because g is a generator of the additive group of zm, we have gcd(g,m) = 1) and then k ≡ xy (mod m).

on the other hand, it is widely believed that using multiplicative subgroups of the group of units z ∗ m of the residue ring zm modulo m yields examples of groups for which dhp is hard, provided that the modulus m is carefully chosen. this belief also extends to subgroups of the multiplicative group f ∗q of a finite field fq of q elements. in fact these groups are exactly the groups suggested by diffie and hellman [15]. although, since that time the requirements on the suitable groups have been refined and better understood, unfortunately not too many other examples of "reliable" groups have been found. probably the most intriguing and promising example, practically and theoretically, is given by subgroups of point groups on elliptic curves, which have been proposed for this kind of application by koblitz [24] and miller [36]. since the original proposal, many very important theoretical and practical issues related to using elliptic curves in cryptography have been investigated, see [2, 17]. even more surprisingly, elliptic curves have led to certain variants of the diffie– hellman schemes, which are not available in subgroups of f ∗q or z ∗ m, see [5, 22, 23] and references therein.

diffie–hellman and discrete logarithm problems: it is immediate that if one can find x from the given value of x = gx, that is, solve the discrete logarithm problem, dlp, then the whole scheme is broken. in fact, in our example of a "weak" group g, this is exactly dlp which can easily be solved. thus dhp is not harder than dlp. on the other hand, the only known (theoretical and practical) way to solve dhp is to solve the associateddlp. thus a natural question arises whether dhp is equivalent to dlp or is strictly weaker. the answer can certainly depend on the specific group g.

despite a widespread assumption that this indeed is the case, that is, that in any cryptographically "interesting" group dhp and dlp are equivalent, very few theoretical results are known. in particular, it has been demonstrated in [6, 31, 32] that, under certain conditions, dhp and dlp are polynomial time equivalent. however, there are no unconditional results known in this direction. some quantitative relations between complexities of dhp and dlp are considered in [13].

cryptographically interesting groups: as we have mentioned, the choice of the group g is crucial for the hardness of dhp (while the choice of the generator g does not seem to be important at all). probably the most immediate choice is g = f ∗q , thus g is a primitive element of fq . however, one can work in a subgroup of f ∗q of sufficiently large prime order f∗q (but still much smaller than q and thus more efficient) without sacrificing the security of the protocol. indeed, we recall that based on our current knowledge we may conclude that the hardness of dlp in a subgroup g ⊆ f ∗q (at least for some most commonly used types of fields; for further discussion see discrete logarithm problem) is majorised: by f∗q1/2 where f∗q is the largest prime divisor of #g, see [35, 44]. by lq[1/2, 21/2] for a

rigorous unconditional algorithm, see [37]. by lq [f∗q 1/3, (64/9)1/3] for the heuristic number field sieve algorithm, see [39, 40], where as usual we denote by lx[t, γ] (see l-notation) any quantity of the form lx[t, γ] = exp((γ + o(1)) (log x)t (log log x)1−t).

it has also been discovered that some special subgroups of some special extension fields are computationally more efficient and also allow one to reduce the information exchange without sacrificing the security of the protocol. the two most practically and theoretically important examples are given byluc, see [3,43], and xtr, see [26–28], protocols (see, more generally, subgroup cryptosystems). despite several substantial achievements in this area, these results are still to be better understood and put in a more systematic form [10].

one can also consider subgroups of the residue ring z ∗ m modulo a composite m ≥ 1. although they do not seem to give any practical advantages (at least in the original setting of the two party key exchange protocol), there are some theoretical results supporting this choice, for example, see [1]. the situation is more complicated with subgroups of the point groups of elliptic curves, and more generally of abelian varieties. for these groups not only the arithmetic structure of the cardinality g matters, but many other factors also play an essential role, see [2, 17, 19, 20, 25, 34, 38] and references therein.

Figure 4.1: Original Document

4.1.3.8 sklearn

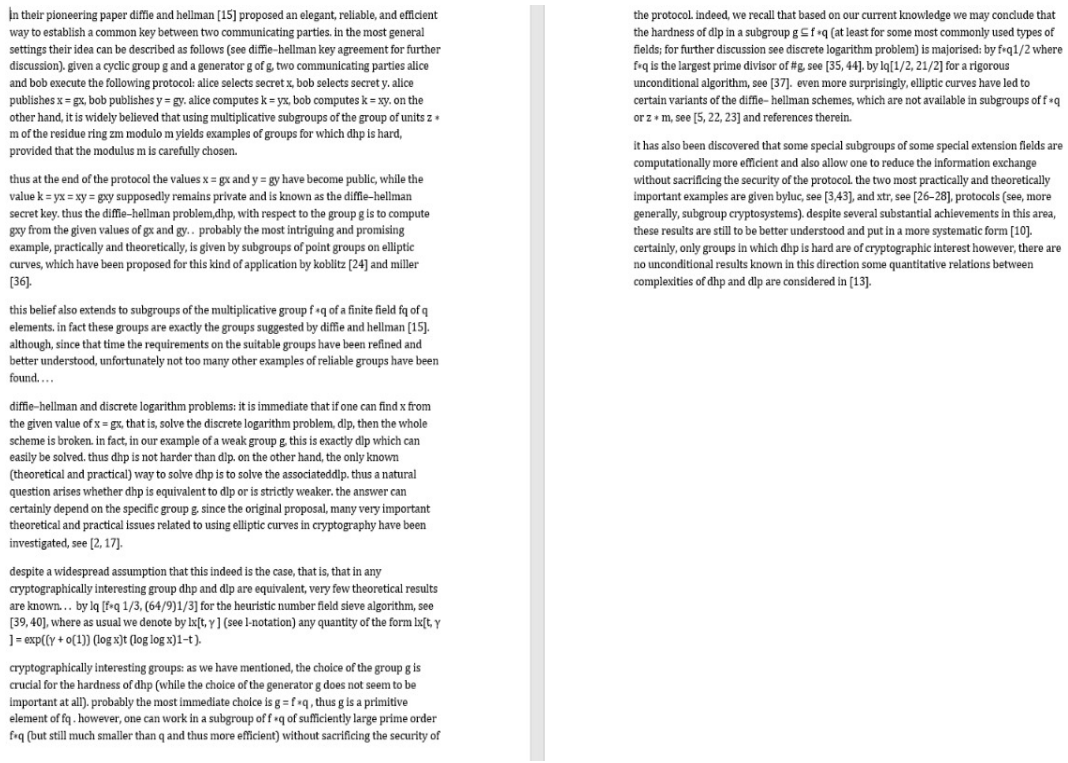
This Python library is used to import cosine similarity to find the similarity between two vectors.

4.2 Results Overview

The documents in the dataset had been given as input for the Multi-Objective, Multi-Mutation Genetic Algorithm topicwise, and the initialized documents were stored in a folder inside the input folder and the successful documents only are stored in a folder inside the initialization folder named "success".

This "success" folder is stored inside the "fakes" folder which contains the initialized documents in it. This "fakes" folder is stored inside the main input Original Documents folder. Also the metrics of each Original, newly initialized documents, and the correctly mutated documents with the suffix "successful" is stored in the name "metrics.docx" in the input folder.

The original document and the sample output for a document created from the data from the encyclopedia are given in fig 4.1 and fig 4.2 respectively



In their pioneering paper diffie and hellman [15] proposed an elegant, reliable, and efficient way to establish a common key between two communicating parties. In the most general settings their idea can be described as follows (see diffie–hellman key agreement for further discussion). given a cyclic group g and a generator g of g, two communicating parties alice and bob execute the following protocol: alice selects secret x, bob selects secret y. alice publishes x = gx, bob publishes y = gy. alice computes k = yx, bob computes k = xy. on the other hand, it is widely believed that using multiplicative subgroups of the group of units z ∗ m of the residue ring zm modulo m yields examples of groups for which dhp is hard, provided that the modulus m is carefully chosen.

thus at the end of the protocol the values x = gx and y = gy have become public, while the value k = yx = xy = gxy supposedly remains private and is known as the diffie–hellman secret key. thus the diffie–hellman problem,dhp, with respect to the group g is to compute gxy from the given values of gx and gy. . probably the most intriguing and promising example, practically and theoretically, is given by subgroups of point groups on elliptic curves, which have been proposed for this kind of application by koblitz [24] and miller [36].

this belief also extends to subgroups of the multiplicative group f ∗q of a finite field fq of q elements. in fact these groups are exactly the groups suggested by diffie and hellman [15]. although, since that time the requirements on the suitable groups have been refined and better understood, unfortunately not too many other examples of reliable groups have been found. . . .

diffie–hellman and discrete logarithm problems: it is immediate that if one can find x from the given value of x = gx, that is, solve the discrete logarithm problem, dlp, then the whole scheme is broken. in fact, in our example of a weak group g, this is exactly dlp which can easily be solved. thus dhp is not harder than dlp. on the other hand, the only known (theoretical and practical) way to solve dhp is to solve the associateddlp. thus a natural question arises whether dhp is equivalent to dlp or is strictly weaker. the answer can certainly depend on the specific group g. since the original proposal, many very important theoretical and practical issues related to using elliptic curves in cryptography have been investigated, see [2, 17].

despite a widespread assumption that this indeed is the case, that is, that in any cryptographically interesting group dhp and dlp are equivalent, very few theoretical results are known. . . by lq [f∗q 1/3, (64/9)1/3] for the heuristic number field sieve algorithm, see [39, 40], where as usual we denote by lx[t, γ] (see l-notation) any quantity of the form lx[t, γ] = exp((γ + o(1)) (log x)t (log log x)1−t).

cryptographically interesting groups: as we have mentioned, the choice of the group g is crucial for the hardness of dhp (while the choice of the generator g does not seem to be important at all). probably the most immediate choice is g = f ∗q , thus g is a primitive element of fq . however, one can work in a subgroup of f ∗q of sufficiently large prime order f∗q (but still much smaller than q and thus more efficient) without sacrificing the security of the protocol. indeed, we recall that based on our current knowledge we may conclude that the hardness of dlp in a subgroup g ⊆ f ∗q (at least for some most commonly used types of fields; for further discussion see discrete logarithm problem) is majorised: by f∗q1/2 where f∗q is the largest prime divisor of #g, see [35, 44]. by lq[1/2, 21/2] for a rigorous unconditional algorithm, see [37]. even more surprisingly, elliptic curves have led to certain variants of the diffie– hellman schemes, which are not available in subgroups of f ∗q or z ∗ m, see [5, 22, 23] and references therein.

it has also been discovered that some special subgroups of some special extension fields are computationally more efficient and also allow one to reduce the information exchange without sacrificing the security of the protocol. the two most practically and theoretically important examples are given byluc, see [3,43], and xtr, see [26–28], protocols (see, more generally, subgroup cryptosystems). despite several substantial achievements in this area, these results are still to be better understood and put in a more systematic form [10]. certainly, only groups in which dhp is hard are of cryptographic interest however, there are no unconditional results known in this direction some quantitative relations between complexities of dhp and dlp are considered in [13].

Figure 4.2: Generated fake document

4.3 Detailed Analysis

The detailed analysis of the output of this project is as follows

4.3.1 Performance metrics

The Performance metrics are analyzed based on the fitness of the generated believable fake documents. The fitness of the documents is measured by the comparison of the sequentiality, connectivity, and dispersion values with that of the original documents.

The fitness of the generated fake documents is almost one document for the 30 -50 documents generated. This means if we use a population size of 50 then we almost have a probability of successfully creating a new believable fake document with the required standards (i.e.) less sequentiality, less connectivity, and increased dispersion.

4.3.2 Comparative analysis

The comparative analysis of the output is conducted over the successful files created in the "success" folder. This analysis was done using the "metrics.docx" created in the input folder which stores the information about the newly generated and original documents. This folder contains the information about which document after which operation had been successful whether it was addition, deletion, or shuffling.

After the detailed analysis of metrics from all the outputs generated the inferred outputs are:

1. This genetic algorithm prefers the shuffling operation and almost 80% of the generated documents which are successful are generated by shuffling operation and almost the remaining successful documents are created by the deletion operation. The success of the addition operation was very minimal.

2. As we increase the number of times in the population size the number of successful documents created increases and the certainty of successful believable fake generation for each document also increases. When the population size was set to 201 times there was a certainty of almost 100% that atleast one new successful document following all the metrics would be generated.

3. Shuffling and deletion operations in the successful documents are found effective when manually checked. Also, the shuffled document made it hard to comprehend while reading.

Chapter 5

Conclusions

In this project, we explored the development and implementation of a sophisticated framework for generating fake technical documents using cyber deception and Natural Language Processing (NLP). The primary goal was to protect intellectual property by creating convincing decoy documents that can mislead potential attackers.

The results of this project have significant implications for cybersecurity, particularly in protecting intellectual property such as patents and proprietary research. By deploying these fake documents within an organization's network, we can create an additional layer of security that complements existing measures, making it more challenging for attackers to steal valuable information.

5.1 Limitations

The Limitations of this algorithm are:

1. This algorithm takes a lot of time for generating new fake documents so , it might require a lot of time and resources when used for practical applications which may have a large input dataset.

2. This algorithm does not generate successful believable fake documents for addition operation following all the fitness evaluation necessities.

5.2 Future work

To build upon the current work, several future research directions are proposed:

5.2.1 Enhanced NLP Models

Explore the use of more advanced and specialized NLP models to improve the authenticity and variability of the generated documents. In this project, we used bert-base-nli-mean-tokens which did not create addition but there might be a possibility that when we use different models we could obtain a different result since each model has their own way of understanding the concepts.

5.2.2 Real-World Testing

Conduct extensive real-world testing to evaluate the effectiveness of the fake documents against actual cyber attacks and adapt the strategies accordingly. We can actually use this method to prevent real-world attacks.

5.2.3 Use of canarytokens

One future work could be the addition of the canary tokens to the fake generated files and this canary token will alarm whenever opened.

5.2.4 Dataset automation

In future instead of testing and training the model using the manually created dataset we could possibly use the Wikipedia API library and other web scraping tools to take the paragraphs and generate the dataset automatically. We can also try this for a dataset of documents from other fields than the technical field, like political science, finance, etc.,

5.2.5 Use of machine learning

We can use some machine learning algorithms to test the generated fake documents and analyze whether those algorithms could differentiate the original from the generated believable fake documents. If that were possible then it would become easier for attackers to differentiate the originals and their time would be saved.

5.2.6 Use of other algorithms

We can try to use other algorithms such as bird flocking/bird swarm algorithm, and ant colony algorithm to overcome the disadvantages of the genetic algorithm.

Bibliography

[1] P. Karuna, H. Purohit, S. Jajodia, R. Ganesan, and O. Uzuner, "Fake document generation for cyber deception by manipulating text comprehensibility," *IEEE Systems Journal*, vol. 15, no. 1, pp. 835–845, 2020.

[2] N. Prabhaker, G. S. Bopche, A. Mishra, and M. Arock, "Generation of believable fake logic circuits for cyber deception," in *2024 16th International Conference on COMmunication Systems & NETworkS (COMSNETS)*. IEEE, 2024, pp. 13–18.

[3] L. Zhang and V. Thing, "Three decades of deception techniques in active cyber defense - retrospect and outlook," *Computers Security*, vol. 106, p. 102288, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167404821001127

[4] J. Yuill, M. Zappe, D. Denning, and F. Feer, "Honeyfiles: deceptive files for intrusion detection," in *Proceedings from the Fifth Annual IEEE SMC Information Assurance Workshop, 2004.* IEEE, 2004, pp. 116–122.

[5] Y. Hu, Y. Lin, E. S. Parolin, L. Khan, and K. Hamlen, "Controllable fake document infilling for cyber deception," 2022.

[6] X. Han, N. Kheir, and D. Balzarotti, "Deception techniques in computer security: A research perspective," *ACM Comput. Surv.*, vol. 51, no. 4, jul 2018. [Online]. Available: https://doi.org/10.1145/3214305

[7] Y. Zhang, D. Chen, S. Jajodia, A. Pugliese, V. Subrahmanian, and Y. Xiong, "Gait: A game-theoretic defense against intellectual property theft," *IEEE Transactions on Dependable and Secure Computing*, pp. 1–12, 2023.

[8] Q. Han, C. Molinaro, A. Picariello, G. Sperlì, V. S. Subrahmanian, and Y. Xiong, "Generating fake documents using probabilistic logic graphs," *IEEE Transactions on Dependable and Secure Computing*, vol. 19, no. 4, pp. 2428–2441, 2022.