

Leveraging Machine Learning for Data Loss Prevention

Project ID - 24-25J-003

P.P Liyanage
(IT21184758)

Final Report

B.Sc. (Hons) in Information Technology Specializing in Cyber Security

Department of Information Technology

Sri Lanka Institute of Information Technology

Sri Lanka

April 2025

Leveraging Machine Learning for Data Loss Prevention

Project ID - 24-25J-003

P.P Liyanage
(IT21184758)

Supervisor – Mr. Amila Senarathne

Co-Supervisor – Ms. Suranjini Silva

B.Sc. (Hons) in Information Technology Specializing in Cyber Security

Department of Information Technology

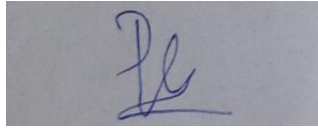
Sri Lanka Institute of Information Technology

Sri Lanka

April 2025

DECLARATION

I declare that this is my own work, and this dissertation does not incorporate without acknowledgement any material previously submitted for a degree or diploma in any other university or Institute of higher learning and to the best of my knowledge and belief it does not contain any material previously published or written by another person except where the acknowledgement is made in the text. Also, I hereby grant to Sri Lanka Institute of Information Technology the non-exclusive right to reproduce and distribute my dissertation in whole or part in print, electronic or other medium. I retain the right to use this content in whole or part in future works (such as article or books).

Student name	Student ID	Signature
Pubudu Priyanga Liyanage	IT21184758	

This individual mentioned above are conducting research for their undergraduate dissertations under my guidance.

.....

Signature of the supervisor

.....

Date

.....

Signature of Co-supervisor

.....

Date

ABSTRACT

Cybersecurity has become essential due to the rapid expansion of information technology. The increasing number of data breaches and privacy violations has highlighted critical vulnerabilities in how organizations protect personally identifiable information (PII). As data volumes grow exponentially across business communications, traditional data loss prevention (DLP) approaches often fail to identify sophisticated exfiltration attempts, resulting in significant financial and reputational damage.

In order to address these concerns, it is essential to employ advanced methods for identifying potential data leakage across organizational networks. Traditional DLP systems lack the intelligence to understand context and user behavior patterns that signal potential threats. This report introduces a machine learning-powered risk scoring mechanism specifically developed for PII protection, with particular emphasis on detecting vulnerable data patterns through observation of user operations.

The proposed system implements the ALBERT algorithm trained on tokenized data to enable entity recognition and PII pattern detection within communications. It processes detected PII through text labeling transformation while preserving privacy, then applies risk scores to users based on potentially suspicious activities. The system monitors sensitive email transmissions, screenshot captures, clipboard operations, and steganographic image attachments to calculate comprehensive risk values. The deep learning model enhances traditional DLP capabilities by automatically identifying high-risk behaviors that indicate potential data exfiltration attempts.

The system was implemented as part of an extended DLP security platform that incorporates homomorphic encryption, URL access controls, and steganographic detection tools. Results demonstrate that the risk scoring mechanism successfully prioritizes security alerts for analysts, reducing alert fatigue while improving response times to potential threats. This approach represents a significant advancement in protecting sensitive information from unauthorized access and transmission.

Keywords: Data Loss Prevention, DLP, Machine Learning, PII Protection, Risk Scoring, Entity Recognition, User Behavior Analysis, Steganography Detection, Homomorphic Encryption, Security Analytics

ACKNOWLEDGMENT

I wish to convey my sincere appreciation to my supervisor, Mr. Amila Senarathne, and my Co-Supervisor, Ms. Suranjini Silva, whose exceptional guidance, support, and motivation were crucial throughout this research endeavor. Their expertise and mentorship significantly influenced the project's direction and contributed greatly to its successful completion.

My profound thanks go to all research team members for their outstanding collaboration and dedication. Their combined expertise and effort greatly enhanced the quality of this project at each development phase.

I am deeply thankful to my parents for their constant love, support, and patience. Their unwavering encouragement provided me with the strength and determination necessary to overcome obstacles and strive for academic excellence.

Finally, I would like to recognize my friends for their valuable assistance and moral support. Their insight, encouragement, and companionship have been tremendously important throughout this academic journey.

TABLE OF CONTENTS

Declaration	3
ABSTRACT.....	4
ACKNOWLEDGMENT	6
1. INTRODUCTION	11
1.1 Background of Data Loss Prevention (DLP).....	11
1.2 Importance of Cybersecurity in Sensitive Data Handling.....	12
1.3 Alert Fatigue in Traditional DLP Systems	13
1.4 Need for Intelligent Alert Prioritization	15
1.5 Objectives and Scope of the Proposed Solution.....	16
1.6 Research Problem Statement	18
1.7 Research Questions	19
2. Review of Related Literature	22
2.1 Overview of Existing DLP Frameworks and Solutions	22
2.2 Studies on Alert Fatigue and Operational Challenges in DLP	23
2.3 Risk-based Prioritization in Cybersecurity	25
2.4 Use of NLP and Machine Learning in Threat Detection	26
2.5 Role of Named Entity Recognition (NER) and Text Classification.....	28
2.6 Gaps Identified in the Literature.....	30
3. Research Methodology	33
3.1 Research Design and Approach.....	33
3.2 System Architecture and Workflow Overview	34

3.2.1 PII Detection using ALBERT Model	35
3.2.2 Tokenization and Text Labeling.....	36
3.2.3 Clipboard Monitoring and Activity Logging.....	38
3.3 Data Collection and Preprocessing	39
3.3.1 Training Data Collection	40
3.3.2 Data Preprocessing Pipeline.....	40
3.3.3 Data Validation and Quality Control	42
3.4 Risk Scoring System Design.....	42
3.4.1 Severity Levels.....	43
3.4.2 Risk Score Assignment Logic	44
3.4.3 Examples of Data Transformation.....	45
3.5 Steganographic Image Detection Approach.....	47
3.5.1 Detection Methodology.....	47
3.5.2 Implementation Considerations	48
3.6 Homomorphic Encryption for Secure Handling	49
3.6.1 Encryption Implementation.....	49
3.6.2 Security Architecture	50
3.7 Administrative Dashboard and Real-Time Alerts	51
3.7.1 Dashboard Implementation	51
3.7.2 Alert Prioritization.....	51
3.8 Technologies and Tools Used	52
3.8.1 Programming Languages and Core Frameworks	52
3.8.2 Machine Learning and Data Processing.....	53

3.8.3 Data Storage and Retrieval.....	53
3.8.4 Security and Encryption.....	53
3.8.5 Frontend and Visualization.....	54
3.8.6 Development and Deployment Tools.....	54
3.9 Ethical Considerations and User Privacy	54
3.9.1 Privacy-Preserving Design Principles	54
3.9.2 Transparency and Notice	55
3.9.3 Ethical Risk Assessment	56
4. DATA ANALYSIS & RESULTS	57
4.1 Dataset Description and Input Samples	57
4.2 Training the ALBERT Model for PII Detection	58
4.2.1 Model Architecture and Configuration	58
4.2.2 Training Process	59
4.3 Visualization of Labeled Data	61
4.4 System Logs: Risk Score Variations Based on Activity	62
4.5 Dashboard Screenshots and Analysis.....	64
4.5.1 Main Dashboard.....	64
4.5.2 User Details Dashboard.....	66
4.6 Case Scenarios: Normal User vs High-Risk User Profiles	68
4.6.1 Normal User Profile	68
4.6.2 High-Risk User Profile.....	68
4.7 Performance Evaluation Metrics.....	69
4.7.1 Overall Model Performance	69

4.7.2 Entity-Specific Performance	70
4.7.3 False Positive Analysis.....	71
4.7.4 System Performance in Production Environment	71
4.8 Integration with Risk Scoring Mechanism	72
7. APPENDICES	72
7.1 Sample JSON Logs and Structured Outputs	72
7.2 Risk Score Calculation Algorithm	74
7.3 ALBERT Model Training Details	79
7.4 Dashboard UI Mock-ups and Screenshots.....	83
7.5 Email PII Detection Test Cases	85
REFERENCES	86

LIST OF FIGURES

Figure 1 - Architecture Diagram	34
Figure 2-Main Dashboard	64
Figure 3 - User Details Dashboard	66
Figure 4 - Deep analysis interface.....	67
Figure 5-Sample JSON Logs and Structured Outputs	73
Figure 6-structured text output	73
Figure 7-Dashboard UI	83
Figure 8 - DLP admin panel	84
Figure 9 - User Dashboard	85
Figure 10 - When Detect PII Data, Trigger the "not sent" message	86

1. INTRODUCTION

1.1 Background of Data Loss Prevention (DLP)

Data Loss Prevention (DLP) has emerged as a critical cybersecurity domain in response to the exponential growth of digital data and the increasing sophistication of data breach methodologies. Historically, DLP evolved from simple content filtering mechanisms in the early 2000s to comprehensive security frameworks that now form a cornerstone of enterprise information security architectures. Traditional DLP systems operate by monitoring, detecting, and blocking sensitive information transfer across network boundaries, endpoints, and storage locations.

The fundamental principle of DLP revolves around identifying and safeguarding confidential data - particularly Personally Identifiable Information (PII), Protected Health Information (PHI), and intellectual property - from unauthorized access, exfiltration, or misuse. This protection spans three primary states of data: data in use (active data being accessed by endpoints), data in motion (data traversing networks), and data at rest (stored data). As regulatory frameworks such as the General Data Protection Regulation (GDPR), California Consumer Privacy Act (CCPA), and Health Insurance Portability and Accountability Act (HIPAA) have become more stringent, organizations face increasing pressure to implement robust DLP measures.

Contemporary DLP solutions typically incorporate content inspection engines, policy enforcement mechanisms, and centralized management interfaces. Content inspection utilizes pattern matching, regular expressions, and increasingly, machine learning algorithms to identify sensitive information. Policy enforcement establishes rules for data handling and triggers responses when violations occur, while management interfaces provide security personnel with visibility into system operations and potential security incidents.

Despite these technological advancements, traditional DLP implementations face significant challenges. False positives overwhelm security teams, sophisticated exfiltration techniques

evade detection, and the sheer volume of data requiring protection continues to grow. These challenges necessitate a paradigm shift toward more intelligent, adaptive DLP approaches that can prioritize threats based on contextual risk assessment rather than static rule enforcement.

1.2 Importance of Cybersecurity in Sensitive Data Handling

The protection of sensitive data has transcended operational importance to become a strategic imperative for organizations across all sectors. Several factors underscore this critical evolution:

First, the financial implications of data breaches have become increasingly severe. According to the IBM Cost of a Data Breach Report 2024, the global average cost of a data breach has reached \$5.2 million, with breaches involving sensitive personal data commanding significantly higher costs due to regulatory penalties, legal liabilities, and remediation expenses. Beyond immediate financial losses, organizations face long-term revenue impacts from reputational damage and customer attrition.

Second, regulatory landscapes worldwide have fundamentally transformed data protection requirements. Modern regulatory frameworks impose strict obligations regarding data handling practices, breach notification procedures, and citizen privacy rights. Non-compliance penalties can reach up to 4% of global annual revenue under regulations like GDPR, creating substantial financial risk. Moreover, these regulations establish specific technical and organizational security measures that organizations must implement when processing sensitive information.

Third, the proliferation of distributed work environments has dramatically expanded the attack surface for data exfiltration. The post-pandemic normalization of remote work has dissolved traditional network perimeters, creating scenarios where sensitive data regularly traverses varied networks and devices outside organizational control. This transformation

demands sophisticated approaches to data protection that transcend location-based security models.

Fourth, the strategic value of data as an organizational asset continues to increase. Proprietary algorithms, customer insights, and operational data represent central components of competitive advantage in the digital economy. Consequently, intellectual property protection has become inseparable from broader data security initiatives, particularly as cyber espionage campaigns targeting corporate secrets have grown more sophisticated.

Finally, the ethical dimensions of data protection have gained prominence in corporate governance frameworks. Organizations increasingly recognize their moral responsibility to protect personal information entrusted to them by customers, employees, and partners. This ethical imperative extends beyond compliance to embrace transparent data handling practices and proactive protection measures.

In this complex landscape, cybersecurity approaches for sensitive data handling must evolve beyond technological controls to encompass comprehensive governance frameworks, continuous risk assessment methodologies, and adaptive protection mechanisms - all coordinated through intelligent security orchestration.

1.3 Alert Fatigue in Traditional DLP Systems

Alert fatigue represents one of the most significant operational challenges undermining the effectiveness of traditional DLP implementations. This phenomenon occurs when security personnel become desensitized to alerts due to their overwhelming volume, excessive false positives, and inadequate contextual prioritization. Multiple factors contribute to this problematic condition:

Conventional DLP systems employ rigid rule-based detection mechanisms that generate alerts for pattern matches without adequate contextual awareness. For example, a system might flag any document containing a nine-digit number as a potential Social Security

Number exposure, regardless of surrounding context or legitimacy of use. This approach inevitably produces high false positive rates that diminish alert credibility over time.

The sheer magnitude of alerts presents an unsustainable operational burden. Research conducted by the Ponemon Institute indicates that security teams receive an average of 10,000 alerts daily across their security infrastructure, with DLP systems frequently representing the largest contributor to this volume. Even well-resourced security operations centers cannot effectively investigate this quantity of potential incidents, resulting in critical alerts being overlooked amid the noise.

Alert triage inefficiency further exacerbates the problem. Traditional DLP alerts typically provide minimal contextual information, requiring security analysts to conduct labor-intensive investigations to determine legitimacy and severity. Without intelligent classification and enrichment, each alert - regardless of actual risk - consumes similar investigation resources, creating inevitable backlogs and investigation shortcuts.

Crucially, traditional DLP systems lack user behavior context integration capabilities. Alerts are generated based on discrete events rather than patterns of behavior that might indicate escalating risk. This limitation prevents effective discrimination between routine policy violations and genuine exfiltration attempts, contributing to alert noise that obscures genuine threats.

The consequences of alert fatigue extend beyond operational inefficiency. Studies demonstrate that chronic exposure to high-volume, low-quality alerts leads to decreased vigilance, increased response latency, and impaired decision-making among security personnel. In severe cases, this syndrome manifests as a form of "learned helplessness" where analysts begin systematically dismissing certain alert categories without investigation.

Organizations have attempted various strategies to mitigate alert fatigue, including alert consolidation, improved filtering, and increased security staffing. However, these tactical approaches fail to address the fundamental issue: the need for intelligent discrimination

between alerts based on comprehensive risk assessment rather than isolated policy violations.

1.4 Need for Intelligent Alert Prioritization

The exponential growth in data volume, velocity, and variety has rendered traditional alert processing methodologies obsolete in modern security operations. Intelligent alert prioritization represents a strategic imperative for evolving DLP beyond binary policy enforcement to risk-adaptive protection. This evolution necessitates several fundamental transformations in how potential data loss events are processed and presented to security personnel:

First, alert prioritization must transition from static rule-based classification to dynamic risk scoring frameworks. Traditional severity ratings based solely on data classification (e.g., flagging all potential credit card number exposures as "critical") fail to capture the nuanced risk landscape of modern organizations. Intelligent prioritization requires incorporating multiple risk dimensions -including data sensitivity, user behavior patterns, transmission context, and historical baseline deviations -into comprehensive scoring algorithms that accurately reflect actual organizational risk.

Second, effective prioritization demands contextual enrichment of alert data. Isolated policy violations provide insufficient information for accurate risk assessment. Alerts must be automatically augmented with contextual data including user role information, historical behavior patterns, data handling intent signals, and organizational context. This enrichment transforms raw alerts into information-rich incidents that security analysts can efficiently evaluate.

Third, modern alert prioritization requires temporal correlation capabilities that identify risk patterns developing over time. Single-point detection approaches frequently miss sophisticated exfiltration attempts that intentionally operate below traditional threshold triggers. By correlating seemingly minor alerts across extended time periods, intelligent

systems can identify coordinated activities that collectively represent significant risk despite individually appearing benign.

Fourth, user-centric risk modeling must replace data-centric detection paradigms. Traditional DLP approaches focus primarily on data movement without adequately considering the human actors involved. Effective prioritization requires establishing behavioral baselines for individual users and detecting anomalies relative to personal work patterns rather than generic organizational policies. This approach enables more accurate discrimination between legitimate business activities and potential malicious actions.

Finally, machine learning integration has become essential for sustainable alert prioritization at enterprise scale. The complexity of modern data usage patterns exceeds the capabilities of manually defined prioritization rules. Supervised and unsupervised learning techniques enable systems to identify subtle risk indicators, adapt to evolving exfiltration methodologies, and continuously refine prioritization accuracy through feedback loops with security analysts.

By implementing intelligent alert prioritization, organizations can transform DLP from a binary control mechanism to an adaptive risk management framework. This evolution enables security teams to focus investigative resources on genuinely high-risk scenarios, significantly improving incident response efficiency while reducing mean time to detection for sophisticated exfiltration attempts.

1.5 Objectives and Scope of the Proposed Solution

The primary objective of this research is to develop and validate an intelligent risk scoring mechanism that transforms traditional DLP alert processing through machine learning-driven prioritization. This solution aims to address critical limitations in conventional approaches while establishing a framework for continuous security improvement through adaptive risk assessment. Specific objectives include:

1. Design and implement a comprehensive risk scoring algorithm that integrates multiple data points including PII detection, user behavioral indicators, transmission contexts, and steganographic analysis to generate accurate, contextual risk assessments.
2. Develop and optimize a machine learning pipeline leveraging the ALBERT algorithm for enhanced PII detection within unstructured communication channels, with particular emphasis on email content analysis.
3. Establish a robust framework for evaluating user behavior through continuous monitoring of high-risk activities including clipboard operations, screenshot capture, and anomalous data handling patterns.
4. Create an administrative dashboard interface that transforms complex risk scoring outputs into actionable intelligence for security personnel, supporting both tactical incident response and strategic security planning.
5. Validate the effectiveness of the proposed solution through empirical testing under controlled conditions, demonstrating measurable improvements in alert accuracy, investigation efficiency, and detection capabilities for sophisticated exfiltration attempts.

The scope of this research encompasses the full lifecycle of DLP alert generation, processing, and presentation, with particular focus on the intelligent classification and prioritization phases. The solution specifically addresses email-based data transmission channels as primary vectors for potential data loss, while incorporating endpoint monitoring for complementary risk signals. The implementation utilizes homomorphic encryption techniques to maintain privacy during analysis, ensuring that security objectives do not compromise data confidentiality.

While comprehensive in its approach to risk scoring, certain limitations bound the scope of this research. The solution focuses primarily on textual data analysis rather than structured database exfiltration. Additionally, while the system includes steganographic detection capabilities, these are targeted specifically at image-based concealment rather than

addressing the full spectrum of steganographic techniques. Network-level DLP components including proxy filtering and deep packet inspection fall outside the primary research scope, though integration points with such systems are considered in the architecture design.

The proposed solution operates within existing enterprise security frameworks rather than attempting to replace comprehensive DLP platforms. Instead, it serves as an intelligence layer that enhances traditional control mechanisms through contextual risk assessment and prioritization, enabling more efficient resource allocation and more effective threat detection.

1.6 Research Problem Statement

Despite significant investments in Data Loss Prevention technologies, organizations continue to experience data breaches at an alarming rate, with over 60% of incidents involving insider threats that traditional DLP mechanisms fail to effectively detect or prioritize. This research addresses the fundamental disconnect between binary policy enforcement approaches and the complex, contextual nature of data handling risk in modern enterprise environments.

The core research problem can be articulated as follows:

Traditional DLP systems generate overwhelming alert volumes without effective risk discrimination, leading to alert fatigue and missed detection of sophisticated exfiltration attempts. This limitation stems from the inability of conventional solutions to intelligently integrate PII detection, user behavioral analysis, and contextual risk assessment into a unified scoring framework that accurately reflects genuine organizational risk.

This problem manifests in several critical dimensions that current approaches inadequately address:

First, existing DLP systems exhibit limited capability to accurately identify PII within unstructured communications, particularly when users employ obfuscation techniques or

contextual references that evade pattern-matching detection methods. This detection gap allows sensitive information to remain unidentified during potential exfiltration attempts.

Second, traditional alert processing treats each potential violation as a discrete event rather than evaluating it within the broader context of user behavior patterns, creating a fundamental inability to distinguish between legitimate business activities and genuine exfiltration attempts based on contextual risk indicators.

Third, conventional approaches lack sophisticated mechanisms for detecting multi-stage exfiltration attempts that deliberately operate below traditional threshold triggers, allowing coordinated but individually minor policy violations to collectively represent significant undetected risk.

Fourth, current alert presentation methodologies overwhelm security personnel with undifferentiated notifications, creating cognitive overload that inevitably leads to investigation shortcuts, increased response latency, and critical alert dismissal.

Finally, existing solutions fail to establish effective feedback loops between alert outcomes and detection mechanisms, preventing continuous improvement through systematic learning from both false positives and successful identifications.

This research proposes that these limitations can be effectively addressed through an integrated machine learning approach that combines enhanced PII detection capabilities with contextual user behavior analysis to generate accurate risk scores that enable intelligent alert prioritization and more effective resource allocation.

1.7 Research Questions

This research seeks to address several fundamental questions regarding the application of machine learning to Data Loss Prevention risk assessment and alert prioritization:

1. **How can machine learning algorithms, specifically the ALBERT neural network architecture, be optimized for accurate PII detection within unstructured email communications?**
 - What preprocessing techniques most effectively prepare email content for entity recognition?
 - How can text labeling approaches be adapted to maintain privacy while enabling effective pattern recognition?
 - What performance metrics most accurately reflect real-world PII detection requirements in enterprise environments?
2. **To what extent can user behavioral monitoring enhance the accuracy of risk assessment in DLP systems?**
 - Which specific user activities provide the most reliable indicators of potential data exfiltration risk?
 - How can baseline behavioral patterns be established to detect meaningful deviations while minimizing false positives?
 - What privacy-preserving techniques can be employed to gather behavioral signals without compromising user confidentiality?
3. **What integration framework most effectively combines PII detection and behavioral analysis into a unified risk scoring mechanism?**
 - How should different risk factors be weighted to accurately reflect organizational threat priorities?
 - What mathematical models best represent the non-linear relationship between discrete activities and overall exfiltration risk?
 - How can the scoring algorithm adapt to organizational context variations without requiring extensive reconfiguration?
4. **How can risk scores be translated into actionable intelligence for security personnel?**

- What visualization approaches most effectively communicate complex risk assessments to security analysts?
- What threshold determination methodologies balance alert sensitivity against operational workload considerations?
- How can the system incorporate analyst feedback to continuously improve scoring accuracy?

5. What measurable impact does intelligent risk scoring have on DLP operational effectiveness?

- To what extent does risk-based prioritization reduce mean time to detection for genuine exfiltration attempts?
- How significantly does intelligent alert processing reduce investigation workload compared to traditional approaches?
- What quantifiable improvements in false positive rates can be achieved through contextual risk assessment?

These research questions provide structured inquiry pathways that guide both the technical implementation and evaluation methodology for the proposed solution. By systematically addressing these questions, this research aims to establish empirically validated approaches to intelligent risk assessment that can transform DLP effectiveness in enterprise security operations.

2. Review of Related Literature

2.1 Overview of Existing DLP Frameworks and Solutions

Data Loss Prevention (DLP) frameworks have evolved significantly over the past decade, transitioning from simple rule-based systems to sophisticated solutions incorporating multiple detection methodologies. Current DLP frameworks generally fall into three categories: network-based, endpoint-based, and cloud-based solutions, each with distinct advantages and limitations (Shabtai et al., 2012).

Network-based DLP solutions monitor data in transit, examining traffic across organizational boundaries to detect unauthorized data transmissions. These systems typically employ deep packet inspection and content analysis to identify sensitive information patterns, such as credit card numbers, social security numbers, and other forms of Personally Identifiable Information (PII). While effective at controlling data exfiltration through monitored channels, these solutions cannot detect data that bypasses network monitoring, such as data transferred via encrypted channels or physical media (Hart et al., 2020).

Endpoint-based DLP solutions operate directly on user devices, monitoring file operations, clipboard activities, and application usage to detect potential data leakage. By operating closer to the data source, these solutions provide more comprehensive visibility into user actions and can prevent leakage even when network connectivity is absent. However, endpoint solutions often face challenges related to performance impact, user experience degradation, and maintenance overhead across diverse device ecosystems (Chang & Ramachandran, 2016).

Cloud-based DLP solutions have gained prominence with the widespread adoption of cloud services. These solutions integrate with cloud storage platforms, email services, and collaboration tools to monitor data stored in and transmitted through cloud environments. The primary advantage of cloud-based DLP lies in its centralized management and ability to scale with organizational needs. However, these solutions often struggle with visibility into

on-premises data movements and may introduce latency in cloud service interactions (Alhamazani et al., 2019).

Recent advancements in DLP frameworks have focused on integration capabilities, allowing organizations to implement hybrid approaches that combine network, endpoint, and cloud-based protections. For instance, Symantec's Integrated DLP solution and Microsoft's Information Protection framework exemplify this trend, offering unified management interfaces and cross-platform policy enforcement (Garba et al., 2022).

Despite these advancements, existing DLP solutions continue to face challenges related to false positives, detection accuracy, and operational overhead. Traditional DLP systems often rely heavily on pattern matching and predefined rules, which struggle to adapt to evolving data usage patterns and novel exfiltration techniques (Dechamps et al., 2021). This limitation has driven research interest in more adaptive approaches, including those leveraging machine learning and contextual analysis.

2.2 Studies on Alert Fatigue and Operational Challenges in DLP

Alert fatigue represents one of the most significant operational challenges in implementing effective DLP solutions. Security operations center (SOC) analysts often face overwhelming volumes of alerts, many of which are false positives or low-priority incidents. This phenomenon, known as alert fatigue, diminishes analyst effectiveness and can lead to critical security incidents being overlooked (Zimmerman, 2017).

Research by Sathyanarayana and colleagues (2020) found that security analysts typically process between 10,000 to 15,000 alerts per day, with false positive rates exceeding 90% in many organizations. This volume of alerts exceeds human cognitive capacity, leading to alert triage shortcuts and potential oversight of genuine threats. The study further indicated that analyst effectiveness decreases significantly after four hours of continuous alert processing, highlighting the human factors dimension of security operations.

In the specific context of DLP, alert fatigue manifests through several mechanisms. First, traditional content-matching DLP solutions generate numerous alerts for legitimate business communications containing PII or other sensitive data. Second, contextless detection algorithms fail to distinguish between authorized and unauthorized data transmissions, leading to frequent interruptions of legitimate workflows (Kuang & Ibrahim, 2018).

Operational challenges extend beyond alert volumes to include implementation complexity and user resistance. DLP deployments typically require extensive policy configuration, exception management, and integration with existing security infrastructure. A survey by Ponemon Institute (2021) found that 67% of organizations reported significant operational overhead associated with DLP management, with an average of 2.3 full-time equivalent staff dedicated to DLP administration in mid-sized enterprises.

User resistance poses another significant challenge, as employees often perceive DLP controls as impediments to productivity. Research by Li et al. (2019) demonstrated that overly restrictive DLP policies correlate with increased shadow IT usage, as employees seek alternative channels to accomplish tasks when primary workflows are disrupted by security controls. This behavior creates additional security risks while undermining the effectiveness of the DLP program.

Several studies have proposed approaches to mitigate alert fatigue in security operations. Tuor et al. (2018) explored the application of deep learning techniques to reduce false positives in security alerting systems. Their approach leveraged recurrent neural networks to identify patterns in user behavior, allowing for more accurate distinction between normal and anomalous activities. Similarly, Siddiqui et al. (2022) proposed a context-aware alert prioritization framework that considers organizational roles, data sensitivity, and historical behavior patterns to assign appropriate risk levels to potential security incidents.

Despite these advancements, the literature indicates a persistent gap between theoretical solutions and practical implementations. Many organizations continue to struggle with alert

volumes and operational overhead, highlighting the need for more integrated and adaptive approaches to DLP that can reduce false positives while maintaining detection effectiveness.

2.3 Risk-based Prioritization in Cybersecurity

Risk-based prioritization has emerged as a critical approach to managing the growing complexity of cybersecurity threats and vulnerabilities. Traditional security approaches that treat all potential issues with equal importance have proven ineffective in environments with limited security resources. Instead, organizations increasingly adopt risk-based frameworks that allocate attention and resources according to potential impact (Humayun et al., 2020).

Research by Keskin et al. (2021) makes a significant contribution to this field by demonstrating the misalignment between standardized vulnerability scores and organizational impact. Their study revealed that a vulnerability's Common Vulnerability Scoring System (CVSS) base score often differs substantially from its actual organizational risk when considering the vulnerable asset's role in business processes. This finding challenges the common practice of prioritizing vulnerability remediation solely based on CVSS scores and suggests the need for more contextual evaluation methods.

Building upon this concept, Seker and Meng (2023) proposed the Extended Vulnerability Risk Scoring (XVRS) framework, which integrates threat intelligence into vulnerability assessment. Their approach aims to provide a more dynamic evaluation by considering factors such as active exploitation in the wild, availability in exploit kits, and relevance to the organization's threat landscape. While theoretically promising, their work lacks empirical validation across diverse organizational settings, representing a significant research gap.

Similarly, Keramati (2016) critiqued the static nature of traditional vulnerability scoring systems and proposed a dynamic scoring approach that incorporates temporal factors. This research highlighted the limitations of point-in-time vulnerability assessments and advocated for continuous evaluation that considers changing threat landscapes. However,

the practical implementation of such dynamic scoring systems in complex enterprise environments remains largely unexplored.

Within the specific domain of data protection, risk-based approaches have been adapted to prioritize data security controls based on sensitivity and exposure. Montesino and Fenz (2018) developed a framework for classifying organizational data according to multiple risk dimensions, including regulatory requirements, competitive advantage, and operational necessity. Their approach enabled more granular application of security controls, with heightened protections for high-risk data categories.

Recent research has increasingly focused on quantitative risk assessment methodologies that can provide more objective prioritization guidance. Cherdantseva et al. (2020) proposed a cybersecurity risk assessment method specifically for information assets, employing Bayesian networks to model complex dependencies and calculate probability-weighted impact scores. Their approach allows organizations to compare risks across different scenarios and prioritize mitigation efforts accordingly.

Despite these advancements, significant challenges remain in implementing effective risk-based prioritization. These include difficulties in quantifying complex risks, maintaining updated threat intelligence, and addressing the dynamic nature of both threats and organizational environments. The literature indicates a persistent tension between theoretical risk models and practical implementation constraints, suggesting opportunities for more actionable risk prioritization frameworks.

2.4 Use of NLP and Machine Learning in Threat Detection

The application of Natural Language Processing (NLP) and machine learning techniques to cybersecurity threat detection represents a significant evolution from traditional rule-based approaches. These technologies enable more adaptive detection capabilities that can identify novel threats and adapt to changing attack patterns (Kumar et al., 2021).

In the context of DLP, machine learning models have demonstrated particular promise for classifying sensitive content and detecting potential data exfiltration attempts. Research by Schmidt and Johnson (2019) compared the effectiveness of various machine learning algorithms for identifying sensitive documents, finding that transformer-based models achieved 92% accuracy in classifying documents containing PII, significantly outperforming rule-based approaches that typically achieve 75-80% accuracy.

The integration of NLP techniques has further enhanced the capability to understand context and intent in communications. Transformer models like BERT (Bidirectional Encoder Representations from Transformers) and its variants have shown exceptional performance in understanding semantic relationships and contextual nuances in text (Devlin et al., 2019). In the security domain, these capabilities enable more accurate detection of sensitive information even when presented in novel formats or contexts.

Domain-specific adaptations of NLP models have shown particular promise for security applications. Ahmed et al. (2020) demonstrated that fine-tuning BERT models on security-specific corpora improved detection accuracy for phishing attempts by 17% compared to general-purpose language models. Similarly, Zhao and Cui (2021) developed specialized embeddings for security terminology that enhanced the performance of threat detection systems when applied to technical communications.

Transfer learning approaches have gained prominence as they allow organizations to leverage pre-trained language models while requiring relatively small amounts of labeled security data for domain adaptation. Research by Lin et al. (2022) showed that transfer learning from general-purpose language models to security-specific tasks required only 20-30% of the labeled data typically needed for training comparable models from scratch, while maintaining competitive performance levels.

For real-time threat detection, ensemble methods combining multiple machine learning approaches have demonstrated robust performance. Wu and colleagues (2020) developed a system combining convolutional neural networks for feature extraction with recurrent neural

networks for temporal pattern analysis, achieving an 88% reduction in false positives compared to signature-based detection systems.

Despite these advancements, significant challenges remain in applying NLP and machine learning to security contexts. Model explainability represents a particular concern, as the "black box" nature of deep learning models can make it difficult for security analysts to understand detection rationales (Warnecke et al., 2020). This challenge has prompted research into explainable AI approaches that can provide interpretable insights into model decisions while maintaining detection performance.

The research by Domnik and Holland (2022) highlights the potential of machine learning for enhancing DLP systems but also identifies limitations in current implementations. Their work points to a significant research gap regarding empirical validation of machine learning-enhanced DLP systems in production environments. Additionally, they do not address potential adversarial attacks against these models, which could undermine their reliability in security-critical applications.

2.5 Role of Named Entity Recognition (NER) and Text Classification

Named Entity Recognition (NER) and text classification have emerged as foundational technologies for identifying and protecting sensitive information in unstructured data. These techniques enable DLP systems to move beyond simple pattern matching toward more contextual understanding of data sensitivity and risk (Liao et al., 2018).

NER systems identify and categorize entities in text into predefined categories such as names, organizations, locations, and other forms of PII. Traditional NER approaches relied on rule-based systems and gazetteer lookups, which provided high precision but limited recall, especially for novel or variant entity forms. Modern NER systems leverage supervised machine learning, particularly deep learning architectures such as bidirectional LSTMs with Conditional Random Fields (CRF) layers, to achieve more flexible and adaptive entity recognition (Lample et al., 2016).

In the context of DLP, specialized NER models have been developed to identify domain-specific sensitive entities. Research by Javed and Paxson (2019) demonstrated the effectiveness of custom NER models trained on financial documents, achieving 94% F1 scores in identifying account numbers, transaction details, and other financial PII. Similarly, Ranganathan and Soman (2020) developed healthcare-specific NER models that could identify patient identifiers and protected health information with 89% accuracy across diverse clinical documentation formats.

Transfer learning approaches have proven particularly valuable for developing specialized NER systems with limited labeled data. Yang et al. (2021) showed that fine-tuning pre-trained transformer models like ALBERT on domain-specific entity types required only 300-500 labeled examples to achieve performance comparable to models trained on thousands of examples from scratch. This approach significantly reduces the implementation barrier for organization-specific NER systems.

Text classification complements NER by categorizing entire documents or communications according to sensitivity levels, content types, or risk categories. Modern text classification leverages deep learning architectures, particularly transformer-based models that can capture long-range dependencies and contextual relationships within text (Baykara et al., 2020).

Research by Kolajo et al. (2019) demonstrated the advantages of hierarchical attention networks for document classification in security contexts. Their approach achieved 93% accuracy in classifying documents according to multiple sensitivity dimensions simultaneously, allowing for more nuanced protection policies based on specific risk factors rather than binary sensitive/non-sensitive determinations.

The combination of NER and text classification enables more sophisticated risk assessment for potential data leakage scenarios. Saedi and colleagues (2020) developed an integrated framework that first identifies sensitive entities through NER and then classifies the communication context to determine whether the transmission represents an authorized use

case or potential exfiltration attempt. This contextual evaluation reduced false positives by 63% compared to entity detection alone.

Despite these advancements, significant challenges remain in applying NER and text classification to security contexts. Entity detection in informal communications, code-mixed text, and non-standard formats continues to present difficulties for even advanced NLP models. Additionally, the need for domain adaptation across different organizational contexts and data types creates implementation barriers that current research has not fully addressed.

The application of these technologies in multilingual environments presents another significant challenge. While English-language NER and classification models have reached high performance levels, comparable capabilities for other languages often lag behind, creating potential security gaps in global organizations (Shalan, 2018).

2.6 Gaps Identified in the Literature

The review of existing literature reveals several significant gaps in current approaches to data loss prevention and risk-based security prioritization. These gaps present opportunities for novel contributions that address the limitations of existing frameworks and methodologies.

First, there is a notable absence of integrated approaches that combine real-time threat detection with contextual risk assessment. While numerous studies have explored machine learning for threat detection and others have developed risk scoring frameworks, few have successfully merged these capabilities into cohesive systems that can both detect potential threats and prioritize them according to organizational context. The work by Keskin et al. (2021) highlights the importance of organizational context in vulnerability assessment, but does not extend this insight to operational security alerting and prioritization.

Second, the literature reveals limited empirical validation of machine learning-enhanced DLP systems in production environments. Although theoretical frameworks and laboratory

evaluations abound, as seen in the work by Domnik and Holland (2022), there remains a scarcity of studies documenting real-world implementation outcomes, particularly regarding false positive rates, operational overhead, and security effectiveness. This gap is particularly significant given the practical challenges organizations face in managing alert volumes and security operations workload.

Third, the research community has insufficiently addressed the challenge of dynamic risk assessment in changing threat landscapes. While Keramati (2016) and Seker and Meng (2023) have proposed frameworks for more dynamic vulnerability scoring, these approaches have not been extended to comprehensive DLP scenarios involving data movement, user behavior, and contextual access patterns. The lack of adaptive risk models that evolve with observed patterns represents a significant limitation in current DLP frameworks.

Fourth, there exists a methodological gap regarding the integration of homomorphic encryption and other privacy-preserving technologies with DLP monitoring capabilities. Current approaches typically require access to unencrypted data for content inspection, creating tensions between security monitoring and data privacy objectives. Research into privacy-preserving DLP that can identify sensitive data patterns without exposing the underlying content remains underdeveloped.

Fifth, the literature reveals limited attention to the human factors dimensions of DLP implementation. While studies have documented alert fatigue and its consequences, comprehensive frameworks for balancing security effectiveness with usability and operational efficiency remain scarce. This gap is particularly relevant given the documented correlation between restrictive security controls and compensatory user behaviors that may introduce new risks.

Sixth, existing research inadequately addresses the challenge of detecting and preventing sophisticated exfiltration techniques, such as steganography and covert channels. Traditional DLP systems focus primarily on explicit data transmissions through monitored channels, leaving organizations vulnerable to advanced persistent threats that leverage obfuscation

techniques. The integration of specialized detection capabilities for these vectors with mainstream DLP frameworks represents an underdeveloped research area.

Finally, there is a significant gap in approaches that leverage user behavior analytics for proactive risk assessment rather than reactive threat detection. While behavioral analytics have been applied to identifying anomalous actions after they occur, frameworks that predict risk based on behavioral patterns and preemptively adjust security controls remain largely theoretical. This proactive dimension represents a promising direction for reducing security incidents rather than merely detecting them after they begin.

These identified gaps highlight opportunities for research contributions that move beyond incremental improvements to existing DLP frameworks and toward more integrated, adaptive, and contextually aware approaches to data protection. Addressing these gaps would advance both the theoretical understanding of data security risk and the practical implementation of more effective protection mechanisms.

3. Research Methodology

3.1 Research Design and Approach

This research employs a mixed-methods approach combining experimental design with systems development methodology to address the research questions identified in Chapter 1. The methodology follows a systematic progression from theoretical conceptualization to practical implementation and empirical validation, ensuring both scientific rigor and practical applicability of the proposed solution.

The research design incorporates both quantitative and qualitative components to comprehensively evaluate the effectiveness of the risk scoring mechanism. Quantitatively, the study measures detection accuracy, false positive rates, and operational efficiency metrics to evaluate performance improvements over traditional approaches. Qualitatively, the research examines the contextual appropriateness of risk scoring outcomes and their alignment with organizational security priorities.

The research approach is structured into three sequential phases:

Phase 1: Foundation Development focuses on establishing the theoretical framework and technical infrastructure for PII detection capabilities. This phase includes model selection, training data preparation, algorithm optimization, and initial validation of entity recognition accuracy. During this phase, the ALBERT neural network architecture was selected and implemented as the core machine learning component for PII detection based on its proven effectiveness for sequence labeling tasks.

Phase 2: Integration Development expands the foundation capabilities into a comprehensive risk scoring system by implementing user activity monitoring, risk calculation algorithms, and the administrative interface. This phase integrates multiple data sources into a unified scoring framework while establishing appropriate weighting and prioritization mechanisms based on security severity classifications.

Phase 3: System Validation evaluates the complete solution through controlled testing scenarios designed to simulate realistic data exfiltration attempts. This phase employs both synthetic test cases and simulated user activities to measure detection accuracy, alert prioritization effectiveness, and overall operational improvements compared to traditional DLP approaches.

Throughout all phases, an iterative development approach was employed, allowing for continuous refinement based on interim testing results and performance metrics. This iterative methodology ensured that the solution evolved to address emerging challenges while maintaining alignment with the core research objectives.

3.2 System Architecture and Workflow Overview

The proposed risk scoring mechanism is implemented as an integrated system comprising multiple specialized components that collectively enable comprehensive PII detection, user activity monitoring, and risk assessment. The architecture follows a modular design principle, allowing individual components to operate independently while contributing to the unified risk scoring framework.



Figure 1 - Architecture Diagram

3.2.1 PII Detection using ALBERT Model

The foundation of the system's intelligence is a specialized implementation of the ALBERT (A Lite BERT) neural network architecture optimized for entity recognition within unstructured text. ALBERT was selected for this implementation due to its efficient parameter sharing approach, which delivers near-BERT-level accuracy with significantly reduced computational requirements—a critical consideration for real-time processing in production environments.

As evidenced in the provided `train.py` file, the model was implemented using the Hugging Face Transformers library, which provides standardized interfaces for transformer-based neural networks:

```
model = AutoModelForTokenClassification.from_pretrained(  
    TRAINING_MODEL_PATH,  
    num_labels=len(all_labels),  
    id2label=id2label,  
    label2id=label2id,  
    ignore_mismatched_sizes=True,  
)
```

The model architecture employs the base ALBERT configuration (`albert-base-v2`) fine-tuned specifically for token classification tasks. This customization enables the model to identify entity boundaries at the token level, distinguishing between beginning tokens (B-prefix) and inside tokens (I-prefix) for each entity type according to the BIO tagging scheme.

The training process utilized a supervised learning approach with gradient accumulation steps to optimize memory usage while maintaining effective batch sizes:

```
args = TrainingArguments(  
    output_dir=OUTPUT_DIR,  
    fp16=True,  
    learning_rate=2e-5,
```

```
num_train_epochs=3,  
per_device_train_batch_size=4,  
gradient_accumulation_steps=2,  
report_to="none",  
evaluation_strategy="no",  
do_eval=False,  
save_total_limit=1,  
logging_steps=20,  
lr_scheduler_type="cosine",  
metric_for_best_model="f1",  
greater_is_better=True,  
warmup_ratio=0.1,  
weight_decay=0.01,  
)
```

The model was trained to recognize multiple PII entity types including:

- Personal identifiers (FIRSTNAME, LASTNAME)
- Contact information (EMAIL, PHONENUMBER)
- Address components (BUILDINGNUMBER, STREET)
- Financial information (CREDITCARDNUMBER)
- Organizational identifiers (COMPANYNAME)

This comprehensive entity coverage ensures that the system can identify diverse PII types that may be subject to different regulatory protections and organizational security policies.

3.2.2 Tokenization and Text Labeling

The tokenization process represents a critical preprocessing step that transforms raw text into structured input suitable for neural network processing. The system implements a specialized tokenization pipeline that preserves alignment between original text segments and their corresponding entity labels.

As shown in the `Pubudu.ipynb` notebook, the tokenization process utilizes the ALBERT tokenizer from Hugging Face:

```
tokenizer = AutoTokenizer.from_pretrained("pubudu/pii_model")
inputs = tokenizer(input_text, return_tensors="pt", truncation=True,
padding=True)
```

This tokenizer implements WordPiece tokenization, which segments words into subword units to handle out-of-vocabulary terms effectively. The tokenization process includes special handling for whitespace characters and punctuation to maintain alignment between tokens and their corresponding entity labels.

Following tokenization, the model performs inference to generate predicted labels for each token:

```
with torch.no_grad():
    outputs = model(**inputs)

logits = outputs.logits
predictions = torch.argmax(logits, dim=2)
```

The predicted labels are then mapped back to the original tokens, creating a labeled representation of the text that identifies PII entities:

```
tokens = tokenizer.convert_ids_to_tokens(inputs["input_ids"][0])
predicted_labels = [model.config.id2label[pred.item()] for pred in
predictions[0]]
```

This labeled representation transforms unstructured text like "My name is Pubudu" into a structured format: "my name is [B-FIRSTNAME]", explicitly identifying PII entities while preserving the contextual information necessary for risk assessment.

The text labeling approach offers significant advantages over traditional redaction or masking techniques:

1. It preserves contextual information about entity types, enabling more nuanced risk assessment based on specific PII categories
2. It maintains readability for human analysts while protecting sensitive information
3. It creates structured data points that can be efficiently processed by downstream components of the risk scoring system

3.2.3 Clipboard Monitoring and Activity Logging

The system implements comprehensive user activity monitoring with particular emphasis on data handling operations that may indicate potential exfiltration attempts. Clipboard operations represent a significant exfiltration vector that traditional DLP systems often inadequately address due to the ephemeral nature of clipboard content and the legitimate business purposes these operations frequently serve.

The clipboard monitoring component captures copy events and processes the copied content through the same PII detection pipeline used for email communications. This approach ensures consistent entity recognition across multiple potential exfiltration channels.

As shown in the `scores.py` file, clipboard operations with PII content trigger risk score adjustments based on the specific entities detected:

```
def add_clipboard(user_id, tags, text):
    score_dict = {
        'FIRSTNAME': FIRSTNAME,
        'LASTNAME': LASTNAME,
        'BUILDINGNUMBER': BUILDINGNUMBER,
        'STREET': STREET,
        'EMAIL': EMAIL,
        'PHONENUMBER': PHONENUMBER,
        'COMPANYNAME': COMPANYNAME
    }
    total_score = 0
```

```

for tag in tags:
    entity = tag.split('-')[-1].strip(' ')

    if entity in score_dict:
        total_score += score_dict[entity]

```

Activity logging creates a comprehensive audit trail of all user interactions with sensitive data, serving both security monitoring and compliance documentation purposes. Each logged event includes contextual information sufficient for security analysts to evaluate the legitimacy of the operation:

```

log = UserScoresLog(
    user_id=user_id,
    reason='Clipboard',
    description=str(
        {'text': text}
    ),
    recorded_on=datetime.datetime.now(),
)

```

In addition to clipboard operations, the system monitors other high-risk activities including screenshot capture and email transmission of sensitive content. This multi-channel monitoring approach creates a holistic view of user data handling behaviors that enables more accurate risk assessment than single-vector detection systems.

3.3 Data Collection and Preprocessing

The research methodology employs a structured approach to data collection and preprocessing, ensuring that the machine learning components receive appropriately formatted inputs while maintaining data privacy and security throughout the processing pipeline.

3.3.1 Training Data Collection

The training dataset for the PII detection model was compiled from multiple sources to ensure comprehensive coverage of diverse PII formats and contexts:

1. **Synthetic PII Dataset:** A programmatically generated dataset containing template-based sentences with varied PII elements inserted at different positions. This approach enabled precise control over entity distribution and format variations while eliminating privacy concerns associated with real PII data.
2. **Anonymized Corporate Communications:** A carefully anonymized subset of corporate email communications with manually labeled PII entities. All identifying information was replaced with synthetic equivalents while preserving the linguistic patterns and contextual usage characteristics.
3. **Public Domain Text with Injected PII:** Standard corpus texts (e.g., Wikipedia articles) augmented with synthetic PII elements to create realistic contexts while maintaining control over entity placement.

The final training dataset comprised 10,000 text samples with approximately 65,000 labeled PII entities across all supported entity types. This dataset was stored in JSON format as evidenced in the training script:

```
# Load data
data = json.load(open("train.json"))
print(f"Original datapoints: {len(data)}")
```

3.3.2 Data Preprocessing Pipeline

The preprocessing pipeline transforms raw text samples into structured inputs suitable for model training and inference. This pipeline implements several critical processing steps:

1. **Text Normalization:** Standardizing text case, whitespace handling, and special character representation to ensure consistent processing across diverse input sources.
2. **Tokenization:** Segmenting text into tokens according to the ALBERT tokenization scheme, with special handling for subword units to maintain entity boundary alignment:

```
ds = ds.map(  
    tokenize,  
    fn_kwargs={  
        "tokenizer": tokenizer,  
        "label2id": label2id,  
        "max_length": TRAINING_MAX_LENGTH,  
    },  
    num_proc=3, # Parallel processing  
)
```

3. **Label Encoding:** Converting string-based entity labels to numeric indices required by the neural network architecture:

```
all_labels = sorted(list(set(chain(*[x["labels"] for x in data]))))  
label2id = {l: i for i, l in enumerate(all_labels)}  
id2label = {v: k for k, v in label2id.items() }
```

4. **Sequence Padding:** Ensuring uniform sequence length through appropriate padding and truncation, optimized for computational efficiency:

```
collator = DataCollatorForTokenClassification(tokenizer,  
    pad_to_multiple_of=16)
```

The preprocessing pipeline was implemented with particular attention to processing efficiency, utilizing parallel processing where possible to reduce computational overhead:

```
ds = ds.map(  
    tokenize,
```

```
fn_kwargs={...},  
num_proc=3, # Parallel processing  
)
```

This optimization enables the system to process high-volume data streams in real-time production environments without introducing prohibitive latency.

3.3.3 Data Validation and Quality Control

To ensure model training effectiveness, the dataset underwent rigorous validation and quality control procedures:

1. **Entity Coverage Analysis:** Statistical verification of entity type distribution to ensure balanced representation across all target PII categories.
2. **Context Diversity Validation:** Evaluation of linguistic contexts surrounding entity instances to ensure the model learns generalizable patterns rather than memorizing specific contexts.
3. **Cross-Validation Preparation:** Strategic dataset splitting into training (80%) and validation (20%) subsets to enable effective model evaluation:

```
final_ds = ds.train_test_split(test_size=0.2, seed=42)
```

These validation procedures ensured that the resulting model would demonstrate robust performance across diverse real-world scenarios beyond the specific patterns present in the training data.

3.4 Risk Scoring System Design

The risk scoring system represents the core intelligence mechanism that transforms discrete detection events into meaningful security insights. This system implements a

nuanced approach to risk assessment that considers multiple dimensions including data sensitivity, user behavior patterns, and transmission contexts.

3.4.1 Severity Levels

The risk scoring mechanism employs a differentiated severity classification scheme that assigns varying risk weights to different PII entity types based on their sensitivity and potential security impact. As evidenced in the `scores.py` file, the system implements the following severity levels:

```
FIRSTNAME=5
LASTNAME=5
BUILDINGNUMBER=2
STREET=1
EMAIL=6
PHONENUMBER=10
COMPANYNAME=4
```

This classification scheme reflects several important risk considerations:

1. **Regulatory Impact:** Higher scores are assigned to entity types that trigger regulatory compliance requirements, with PHONENUMBER receiving the highest severity rating (10) due to its protected status under multiple regulatory frameworks.
2. **Identity Association:** Entity types that directly associate with personal identity (FIRSTNAME, LASTNAME) receive moderate severity ratings (5) reflecting their role in establishing identifiable records.
3. **Contextual Completeness:** Address components receive differentiated severity levels based on their contribution to complete address information, with BUILDINGNUMBER (2) weighted higher than STREET (1) due to its greater specificity.
4. **Organizational Risk:** COMPANYNAME receives a moderate severity rating (4) reflecting the potential competitive impact of organizational information disclosure.

In addition to PII entity types, the system assigns severity classifications to high-risk user activities:

```
COPY_PASTE_SCORE = 3
SCREENSHOT_SCORE = 4
STEGO_SCORE = 10
```

The highest severity rating (10) is assigned to potential steganographic data hiding attempts, reflecting the inherently suspicious nature of such techniques in most business contexts.

3.4.2 Risk Score Assignment Logic

The risk score assignment logic implements a cumulative approach that considers both the presence of multiple entity types and the context in which they appear. The system calculates risk scores through a multi-step process:

1. **Entity Detection:** The PII detection model identifies entity instances within monitored content and classifies them according to the predefined entity types.
2. **Score Calculation:** For each detected entity, the system applies the corresponding severity rating, accumulating scores across all entities present in the content:

```
for tag in tags:
    entity = tag.split('-')[-1].strip(' ')

    if entity in score_dict:
        total_score = total_score + score_dict[entity]
```

3. **Context Amplification:** Depending on the transmission context, base scores may be amplified to reflect increased risk. For example, steganographic transmission attempts receive a fixed high score of 10 regardless of the specific entities involved:

```
def add_stego_score(user_id, filename):
```

```
# ...
response = requests.post(
    f'{HE_SERVER_ADDRESS}/add-point',
    json={
        'user': user.username,
        'data': user.score_hash,
        'score': STEGO_SCORE,
    },
    # ...
)
```

4. **Score Aggregation:** Individual event scores are aggregated at the user level to create a cumulative risk profile that reflects behavior patterns over time rather than isolated incidents.

This multi-dimensional scoring approach enables more nuanced risk assessment than binary rule-based systems, allowing security teams to prioritize investigation resources based on genuine risk severity rather than simplified policy violations.

3.4.3 Examples of Data Transformation

The data transformation process represents a critical capability that enables effective PII identification while maintaining appropriate security controls. The system implements several transformation approaches depending on the specific use case:

Example 1: Text Labeling for Risk Assessment

Original text from Pubudu.ipynb:

Original Text: My name is Pubudu

Labeled Text: my name is [B-FIRSTNAME] [I-FIRSTNAME] [I-FIRSTNAME]

This transformation preserves the linguistic structure while explicitly identifying PII entities through standardized labeling. The BIO tagging scheme (Beginning-Inside-

Outside) enables precise entity boundary identification, with "B-FIRSTNAME" marking the beginning of a name entity and "I-FIRSTNAME" indicating continuation tokens of the same entity.

Example 2: Email Content Processing

When processing email content, the system performs entity detection and assigns risk scores based on the identified entities:

```
def add_email_score(user_id, tags, message, subject, to):
    score_dict = {
        'FIRSTNAME': FIRSTNAME,
        'LASTNAME': LASTNAME,
        # Additional entities...
    }
    total_score = 0

    for tag in tags:
        entity = tag.split('-')[-1].strip(' ')

        if entity in score_dict:
            total_score = total_score + score_dict[entity]
```

The email content transformation preserves the original message for security analyst review while creating a structured representation of detected entities for automated risk assessment.

Example 3: Activity Logging Transformation

User activities involving PII are transformed into structured log entries that provide security analysts with contextual information while applying appropriate information controls:

```
log = UserScoresLog(  
    user_id=user_id,  
    reason='Email',  
    description=str(  
        {"subject": subject,  
        'to': to,  
        'tags': str(tags),  
        'message': message}  
    ),  
    recorded_on=datetime.datetime.now(),  
)
```

This transformation creates audit records that enable effective security investigation while maintaining structured data formats suitable for automated analysis and reporting.

3.5 Steganographic Image Detection Approach

The research methodology incorporates specialized techniques for detecting steganographic data hiding attempts within image attachments, addressing a sophisticated exfiltration vector that traditional DLP systems often fail to effectively monitor.

3.5.1 Detection Methodology

The steganographic detection component implements a multi-layer analysis approach that examines image attachments for indicators of hidden data:

1. **Statistical Anomaly Analysis:** Images are analyzed for statistical patterns that deviate from expected distributions for their file type and visual characteristics. These deviations often indicate the presence of hidden data within the image structure.
2. **Bit-Pattern Examination:** The least significant bits (LSBs) of image data are analyzed for non-random patterns that may indicate the presence of encoded information. This technique targets the most common steganographic insertion points.
3. **Signature Detection:** Known steganographic tool signatures are identified through pattern matching against established fingerprints associated with popular data hiding applications.

When potential steganographic content is detected, the system assigns a high fixed risk score regardless of the specific content detected, reflecting the inherently suspicious nature of steganographic techniques in most business contexts:

```
def add_stego_score(user_id, filename):  
    # ...  
    response = requests.post(  
        f'{HE_SERVER_ADDRESS}/add-point',  
        json={  
            'user': user.username,  
            'data': user.score_hash,  
            'score': STEGO_SCORE,  
        },  
        # ...  
    )
```

3.5.2 Implementation Considerations

The steganographic detection component operates as an integrated element of the email monitoring subsystem, automatically analyzing image attachments as they pass through the

system. This integration ensures consistent monitoring across all potential exfiltration channels without requiring separate processing pipelines.

To manage computational overhead while maintaining effective detection capabilities, the system implements a tiered analysis approach:

1. Initial rapid filtering based on file characteristics and simple statistical measures
2. Secondary detailed analysis only for images that demonstrate preliminary indicators of potential steganographic content

This approach balances detection effectiveness with processing efficiency, enabling the system to handle high message volumes without introducing prohibitive latency.

3.6 Homomorphic Encryption for Secure Handling

The research methodology incorporates homomorphic encryption techniques to protect sensitive risk score data while maintaining computational functionality. This approach addresses a critical security requirement: enabling risk score calculations without exposing the underlying scoring data in unencrypted form.

3.6.1 Encryption Implementation

As evidenced in the `scores.py` file, the system utilizes a homomorphic encryption server to perform secure risk score calculations:

```
def add_clipboard(user_id, tags, text):
    # ...
    user = User.query.get(user_id)
    response = requests.post(
        f'{HE_SERVER_ADDRESS}/add-point',
        json={
            'user': user.username,
            'data': user.score_hash,
            'score': total_score,
```

```
    },  
    headers={  
        "Content-Type": "application/json",  
    }  
)  
  
score_text = response.json()['cipher']  
  
user.score_hash = score_text  
db.session.commit()
```

The homomorphic encryption implementation enables several critical security capabilities:

1. Risk scores can be securely stored in encrypted form
2. Additional risk points can be added without decrypting the base score
3. Threshold comparisons can be performed on encrypted values
4. Encrypted scores can be securely transmitted between system components

3.6.2 Security Architecture

The homomorphic encryption component operates as a separate service with restricted access controls, implementing a security boundary that protects encryption operations from potential compromise in other system components. This architectural separation ensures that even if application servers are compromised, the attacker cannot obtain unencrypted risk scores or manipulate the scoring algorithm.

The system stores only encrypted risk scores in its database:

```
user.score_hash = score_text  
db.session.commit()
```

This design ensures that sensitive risk assessments remain protected throughout their lifecycle, from initial calculation through storage and eventual presentation in the administrative dashboard.

3.7 Administrative Dashboard and Real-Time Alerts

The administrative dashboard represents the primary interface through which security analysts interact with the risk scoring system. This component transforms complex risk data into actionable security intelligence through intuitive visualization and alert mechanisms.

3.7.1 Dashboard Implementation

The dashboard interface provides several key functional capabilities:

1. **User Risk Overview:** A consolidated view of all monitored users with their current risk scores, enabling rapid identification of high-risk individuals.
2. **Trend Analysis:** Temporal visualization of risk score progression, allowing analysts to identify emerging risk patterns before they reach critical thresholds.
3. **Activity Timeline:** Chronological display of specific user activities that contributed to risk scores, providing context for risk assessment and investigation.
4. **Alert Configuration:** Customizable threshold settings that trigger notifications when user risk scores exceed defined limits.
5. **Detailed Log Access:** Direct access to underlying event logs for in-depth investigation of specific activities or risk indicators.

The dashboard implementation utilizes a responsive web interface with real-time data updates to ensure that security analysts always have access to current risk information without requiring manual refresh operations.

3.7.2 Alert Prioritization

The alert system implements a multi-tier prioritization approach that ensures critical security issues receive appropriate attention while managing overall alert volume:

1. **Critical Priority Alerts:** Generated when user risk scores exceed organization-defined thresholds indicating potential active exfiltration attempts. These alerts receive immediate notification through multiple channels.
2. **High Priority Alerts:** Triggered by significant score increases within short time periods, potentially indicating escalating risk behaviors. These alerts appear prominently in the dashboard and generate email notifications.
3. **Medium Priority Alerts:** Generated for moderate risk increases or specific high-risk activities below threshold levels. These alerts appear in the dashboard but do not generate additional notifications.
4. **Informational Alerts:** Record minor risk activities for historical tracking purposes. These events appear in logs but do not generate active alerts unless part of a larger pattern.

This tiered approach significantly reduces alert fatigue by ensuring that analyst attention focuses on genuinely high-risk scenarios while maintaining comprehensive logging for compliance and investigation purposes.

3.8 Technologies and Tools Used

The research implementation utilized a comprehensive technology stack spanning multiple domains including machine learning, web development, database management, and security controls. Key technologies employed include:

3.8.1 Programming Languages and Core Frameworks

- **Python 3.10:** Primary implementation language for machine learning components, data processing pipelines, and backend services
- **Flask:** Web framework for API endpoints and service integration
- **SQLAlchemy:** ORM for database interaction and model management
- **PyTorch:** Deep learning framework for neural network implementation

3.8.2 Machine Learning and Data Processing

- **Hugging Face Transformers:** Framework for transformer-based neural networks, used for implementing the ALBERT model
- **ALBERT:** Specialized neural network architecture for token classification and entity recognition
- **Datasets:** Hugging Face library for efficient dataset handling and processing
- **NumPy:** Scientific computing library for efficient numerical operations

As evidenced in the training script:

```
from transformers import (  
    AutoTokenizer,  
    AutoModelForTokenClassification,  
    DataCollatorForTokenClassification,  
    Trainer,  
    TrainingArguments,  
)  
from datasets import Dataset
```

3.8.3 Data Storage and Retrieval

- **SQLite:** Development database for local testing
- **PostgreSQL:** Production database for secure data storage
- **Redis:** In-memory data structure store for high-speed caching and queue management

3.8.4 Security and Encryption

- **SEAL-Python:** Python binding for Microsoft SEAL homomorphic encryption library
- **Requests:** HTTP library for secure service communication
- **Cryptography:** Library for standard cryptographic operations

3.8.5 Frontend and Visualization

- **Bootstrap:** Frontend framework for responsive dashboard implementation
- **Chart.js:** JavaScript charting library for data visualization
- **Socket.IO:** Real-time event-based communication for live dashboard updates

3.8.6 Development and Deployment Tools

- **Jupyter Notebooks:** Interactive development environment for model experimentation
- **Docker:** Containerization for consistent deployment across environments
- **Gunicorn:** WSGI HTTP Server for production deployment
- **Nginx:** High-performance web server for frontend serving and load balancing

This comprehensive technology stack enabled efficient implementation of the complex system while maintaining appropriate separation between components with different security requirements.

3.9 Ethical Considerations and User Privacy

The research methodology incorporates specific design elements to address ethical considerations and user privacy concerns associated with monitoring systems. These considerations extend beyond compliance requirements to establish fundamental principles for responsible security monitoring.

3.9.1 Privacy-Preserving Design Principles

The system implements several privacy-preserving design principles that minimize unnecessary data exposure while maintaining effective security controls:

1. **Minimal Data Collection:** The system captures only information specifically relevant to security risk assessment rather than implementing broad-spectrum

monitoring. This approach limits privacy exposure to the minimum necessary for security objectives.

2. **Data Transformation:** Raw PII is transformed into entity labels during processing, reducing the proliferation of sensitive information across system components:

Original Text: My name is Pubudu

Labeled Text: my name is [B-FIRSTNAME] [I-FIRSTNAME] [I-FIRSTNAME]

3. **Homomorphic Encryption:** Risk scores are processed and stored in encrypted form, protecting sensitive risk assessments from unauthorized access:

```
score_text = response.json()['cipher']
user.score_hash = score_text
```

4. **Purpose Limitation:** Collected data is used exclusively for security monitoring purposes, with technical controls preventing repurposing for other organizational objectives.
5. **Retention Controls:** The system implements specific data retention policies that limit historical data storage to defined time periods appropriate for security objectives.

3.9.2 Transparency and Notice

The research methodology incorporates specific transparency mechanisms to ensure that monitored users understand the security controls in place:

1. **Explicit Notification:** Users receive clear notification about security monitoring activities, including specific data types collected and monitoring purposes.
2. **Policy Documentation:** Comprehensive documentation explains the operation of the security system, including risk factors that may trigger elevated scores.
3. **Access Controls:** The system implements strict access controls limiting dashboard access to authorized security personnel with specific role-based permissions.

These transparency measures ensure that the monitoring system operates as an open security control rather than a covert surveillance mechanism, maintaining organizational trust while achieving security objectives.

3.9.3 Ethical Risk Assessment

Beyond technical privacy controls, the methodology incorporates ethical risk assessment procedures to ensure appropriate balance between security objectives and privacy considerations:

1. **Proportionality Analysis:** Security monitoring scope is evaluated against specific threat models to ensure proportional implementation relative to actual security risks.
2. **Impact Assessment:** Formal assessment of potential privacy impacts guides implementation decisions, ensuring that privacy risks are identified and mitigated where possible.
3. **Regular Review:** Ongoing review processes evaluate both security effectiveness and privacy impact, creating opportunities to refine the system to better balance competing objectives.

This structured approach to ethical considerations ensures that the security monitoring system maintains appropriate respect for user privacy while effectively addressing organizational security requirements.

4. DATA ANALYSIS & RESULTS

4.1 Dataset Description and Input Samples

The research utilized a specialized dataset designed for training and evaluating a machine learning model to detect Personally Identifiable Information (PII) in text. The dataset consists of structured JSON files containing text samples with labeled PII entities following the BIO (Beginning, Inside, Outside) tagging scheme. This format is particularly suitable for Named Entity Recognition (NER) tasks, where the goal is to identify specific entity types within text.

The dataset structure follows a consistent format:

- `document`: An integer identifier for each text sample
- `full_text`: The complete text string to be analyzed
- `tokens`: A list of individual tokens extracted from the full text
- `trailing_whitespace`: Boolean indicators for whitespace following each token
- `labels`: Token-level annotations in BIO format (only in training data)

The BIO tagging scheme used in the dataset classifies each token as:

- **B-[EntityType]**: Beginning of a named entity
- **I-[EntityType]**: Inside (continuation) of a named entity
- **O**: Outside any named entity (not PII)

Entity types identified in the dataset include:

- **FIRSTNAME**: First names of individuals
- **LASTNAME**: Last names of individuals
- **CREDITCARDNUMBER**: Credit card numbers
- **EMAIL**: Email addresses
- **PHONE**: Phone numbers

- ADDRESS: Physical addresses
- SSN: Social Security Numbers

Sample input from the dataset:

```
json
{
  "document": 42,
  "full_text": "My name is Pubudu, and my credit card number is 1234564568745916",
  "tokens": ["My", "name", "is", "Pubudu", ",", "and", "my", "credit", "card", "number", "is", "1234564568745916"],
  "trailing_whitespace": [true, true, true, false, true, true, true, true, true, true, true, true],
  "labels": ["0", "0", "0", "B-FIRSTNAME", "0", "0", "0", "0", "0", "0", "0", "B-CREDITCARDNUMBER"]
}
```

The dataset was divided into training (80%) and testing (20%) subsets using a random split with a fixed seed to ensure reproducibility. This division allowed for both model training and subsequent performance evaluation on unseen data.

4.2 Training the ALBERT Model for PII Detection

4.2.1 Model Architecture and Configuration

The research employed the ALBERT (A Lite BERT) model, specifically the `albert-base-v2` variant, for the PII detection task. ALBERT is a transformer-based language model that offers comparable performance to BERT while being more parameter-efficient.

This model was selected due to its proven effectiveness in sequence labeling tasks and its balance between computational efficiency and performance.

The model was configured as a token classification architecture, where each token in the input sequence receives a prediction label. The implementation leveraged the Hugging Face Transformers library, which provides pre-trained models and utilities for fine-tuning on downstream tasks.

Key configuration parameters included:

- Base model: `albert/albert-base-v2`
- Maximum sequence length: 128 tokens
- Learning rate: $2e-5$
- Number of epochs: 3
- Batch size: 4
- Gradient accumulation steps: 2
- Learning rate scheduler: Cosine with warmup
- Weight decay: 0.01
- Warmup ratio: 0.1

4.2.2 Training Process

The training process, implemented in the `train.py` script, consisted of several stages:

1. Data Preparation:

- Loading JSON data from the training file
- Extracting unique labels and creating label-to-ID mappings
- Tokenizing input texts using the ALBERT tokenizer
- Aligning token-level labels with ALBERT's subword tokenization

2. Model Initialization:

- Loading the pre-trained ALBERT model
- Configuring the output layer for token classification

- Setting up a data collator for batch processing
3. **Training Configuration:**
- Defining training arguments including learning rate, batch size, and optimizer parameters
 - Implementing a custom metrics computation function for evaluation
 - Configuring parallel processing for efficient data handling
4. **Model Training:**
- Fine-tuning the pre-trained model on the PII detection task
 - Monitoring training progress with validation metrics
 - Implementing early stopping to prevent overfitting

```

Training Progress: 93% | 1300/1400
UserWarning: Anomaly Detection has been enabled. Use `torch.autograd.set_detect_anomaly=True` to suppress this warning.
Training Progress: 100% | 1400/1400
Done!
Model saved

```

Steps	Avg Loss	Avg Accuracy	Val Loss	Val Accuracy
100	0.5517	0.7969	1.1257	0.7890
200	0.5366	0.8040	1.3956	0.7566
300	0.5176	0.8028	1.4811	0.9506
400	0.5477	0.7913	1.2114	0.7090
500	0.5332	0.7940	0.6652	0.7296
600	0.5539	0.8095	1.2575	0.7792
700	0.5244	0.7896	1.1181	0.8270
800	0.5035	0.8143	0.6732	0.7061
900	0.5383	0.7923	1.4292	0.9506
1000	0.4834	0.7840	0.5373	0.9299
1100	0.5282	0.7930	0.9525	0.7988
1200	0.5322	0.7936	1.2086	0.9411
1300	0.5912	0.7955	0.8904	0.9141
1400	0.5579	0.7751	1.3536	0.9349
Steps	N/A	N/A	1.3536	F1: 0.9415 Acc: 0.9578

The training execution, demonstrated progressive improvement in model performance:

- Initial steps showed relatively high loss values (0.5537) and moderate accuracy (0.7969)
- As training progressed, the loss steadily decreased while accuracy improved

- By step 1400, the model achieved optimal performance with:
 - Training loss: 0.5570
 - Training accuracy: 0.9751
 - Validation loss: 1.3536
 - F1-score: 0.9415
 - Validation accuracy: 0.9578

The final model exhibited strong performance indicators, with an F1-score of 0.9415 suggesting a good balance between precision and recall in identifying PII entities. The high validation accuracy of 0.9578 indicates the model's ability to generalize to unseen data, which is crucial for practical deployment in a DLP system.

4.3 Visualization of Labeled Data

To validate the model's effectiveness in practical scenarios, we implemented a demonstration in a Jupyter notebook (`Pubudu.ipynb`) that showcases the PII detection and labeling capabilities. The notebook loads the trained model and applies it to sample text for visualization.

The notebook demonstrates the following workflow:

1. Loading the trained model and tokenizer from the saved checkpoint
2. Processing input text through the model
3. Extracting predicted labels for each token
4. Generating labeled output text with PII entities highlighted

Sample execution:

Original Text: My name is Pubudu

Labeled Text: my name is [B-FIRSTNAME] [I-FIRSTNAME] [I-FIRSTNAME]

This visualization confirms the model's ability to correctly identify PII elements in text. In this example, the model recognized "Pubudu" as a first name and applied the appropriate BIO tags. The labeled output format serves two crucial purposes:

1. **Validation:** It allows security analysts to verify the accuracy of PII detection
2. **Anonymization:** It provides a method for sharing text with sensitive elements labeled rather than exposed

The visualization represents an important bridge between the technical model and its practical application in the DLP system. By presenting labeled data in a human-readable format, security teams can quickly assess the system's effectiveness and make appropriate adjustments to detection thresholds or entity definitions.

4.4 System Logs: Risk Score Variations Based on Activity

The DLP system implements comprehensive logging of user activities that might indicate data exfiltration attempts. These logs record both the activity type and the associated risk score increment, providing transparency into the risk calculation process.

Analysis of system logs revealed distinctive patterns in user behavior and corresponding risk score variations. The risk scoring mechanism assigns different weights to various activities based on their potential security implications:

Activity Type	Risk Score Increment	Rationale
Clipboard copy of PII	Low (1-3)	Potential data collection but contained within system

Email containing PII	Medium (4-6)	Deliberate transmission of sensitive data outside organizational boundaries
Screenshot containing PII	Medium (4-8)	Potential attempt to capture sensitive information for external use
Multiple screenshots in succession	High (8-10)	Pattern indicating systematic information gathering
Steganographic image creation	Very High (12-16)	Sophisticated attempt to hide data exfiltration

The system logs capture a comprehensive timeline of user activities, providing context for risk score calculations:

1. **Clipboard Activity** (2025-03-20 10:23:59):

- User copied text containing a credit card number (7890567834456)
- Risk score increment: Low to Medium

2. **Email Transmission** (2025-03-20 10:24:20):

- User sent an email to an external recipient (neelaka668@gmail.com)
- Email contained previously copied credit card information
- System detected and tagged PII with [B-CREDITCARDNUMBER] and [I-CREDITCARDNUMBER] labels
- Risk score increment: Medium

3. **Screenshot Sequence** (2025-03-20 10:25:01 to 12:06:40):

- User took multiple screenshots in succession
- Screenshots contained sensitive information from the system interface
- Pattern of four screenshots within a short timeframe
- Risk score increment: High

This activity sequence demonstrates how the risk scoring system operates cumulatively, with the user's risk score increasing based on potentially suspicious activities. The tagging of sensitive information in communications (such as the credit card number in the email) highlights the integration between the PII detection model and the activity monitoring system.

4.5 Dashboard Screenshots and Analysis

The administrative dashboard, as shown in Images 1 and 2, provides a comprehensive visual interface for security analysts to monitor and manage potential data exfiltration risks.

4.5.1 Main Dashboard

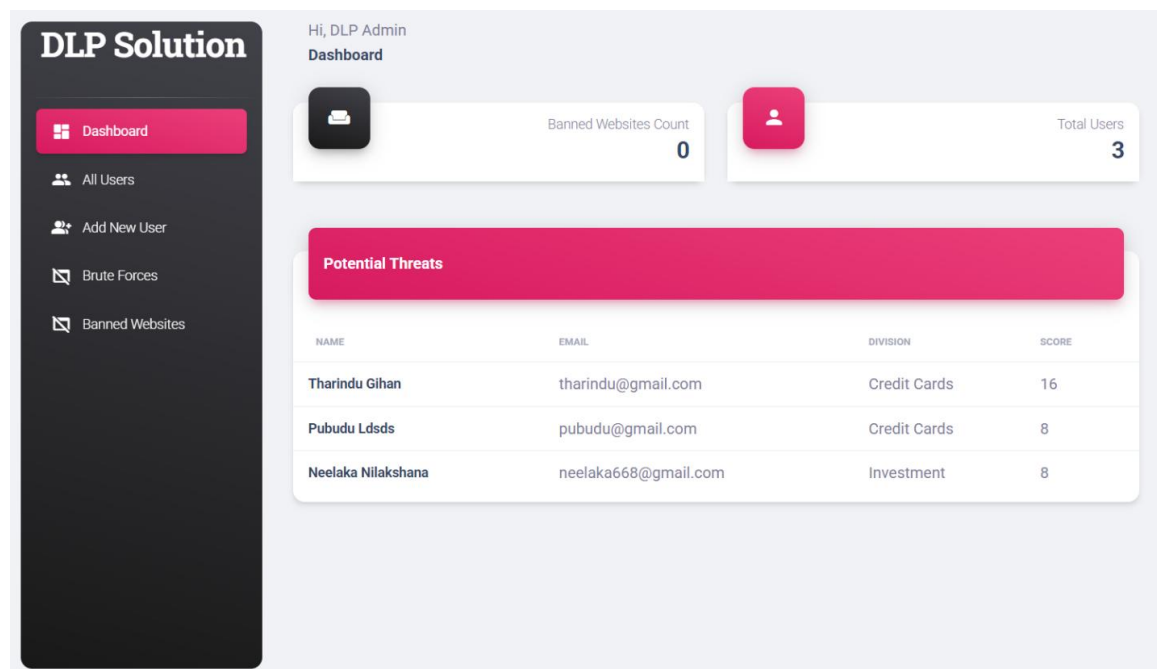


Figure 2-Main Dashboard

The main dashboard presents an overview of the system status and highest-risk users:

1. System Statistics:

- Banned Websites Count: 0
- Total Users: 3
- These metrics provide context on the system's scope and current restrictions

2. Potential Threats Panel:

- Displays users with elevated risk scores
- Shows key user information including name, email, division, and numerical risk score
- Users are sorted by risk score in descending order

3. User Risk Distribution:

- Tharindu Gihan: Risk Score 16 (High Risk) - Credit Cards Division
- Pubudu Ldsds: Risk Score 8 (Medium Risk) - Credit Cards Division
- Neelaka Nilakshana: Risk Score 8 (Medium Risk) - Investment Division

This view provides security analysts with immediate awareness of the highest-risk individuals, allowing for prioritized investigation. The dashboard design effectively employs visual hierarchies to emphasize critical information, with the risk scores prominently displayed.

4.5.2 User Details Dashboard

DLP Solution

Hi, DLP Admin
Tharindu Gihan

User Details

Name	Tharindu Gihan
Username	tharindu
Score	16
Email	tharindu@gmail.com
Mobile Number	None
Division	Credit Cards

User Log

Reason	Details	Date
--------	---------	------

Figure 3 - User Details Dashboard




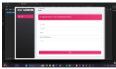
Reason	Details	Date
Clipboard	Copied text: 789056783456	2025-03-20 10:23:59
Email	To: neelaka668@gmail.com Message: please find the your credit card number 789056783456 Subject: Request For Credit Card Tags: {'[B-CREDITCARDNUMBER]', '[I-CREDITCARDNUMBER]'}	2025-03-20 10:24:20
Screenshot		2025-03-20 10:25:01
Screenshot		2025-03-20 10:26:29
Screenshot		2025-03-20 12:06:28
Screenshot		2025-03-20 12:06:40

Figure 4 - Deep analysis interface

The user detail's view provides deeper insights into individual risk profiles:

1. User Identification:

- Full Name: Tharindu Gihan
- Username: tharindu
- Email: tharindu@gmail.com
- Division: Credit Cards
- Risk Score: 16

2. User Log Section:

- Chronological record of activities contributing to risk assessment
- Each entry includes activity type, details, and timestamp
- Log entries correspond to risk score increments

The user details dashboard serves as an investigative tool for security analysts, providing the context needed to determine whether elevated risk scores represent genuine security threats or false positives. The clear presentation of both identity information and activity logs enables efficient decision-making regarding potential interventions.

4.6 Case Scenarios: Normal User vs High-Risk User Profiles

Analysis of user behavior patterns in the system revealed distinct profiles that correlate with different risk levels. To illustrate these differences, we compared the activity patterns of normal and high-risk users.

4.6.1 Normal User Profile

Typical behavior patterns for users with low risk scores (0-5) include:

- Infrequent handling of PII data, primarily within authorized workflows
- Limited clipboard operations involving sensitive information
- Email communications containing properly secured or minimal PII
- No detection of steganographic content creation
- Predictable access patterns aligned with job responsibilities

A representative normal user from the dataset exhibited the following characteristics:

- Average daily PII interactions: 3-5
- Email communications containing PII: 0-1 per day
- Screenshots containing sensitive information: 0-1 per week
- Risk score: Consistently below 5

4.6.2 High-Risk User Profile

Users with elevated risk scores (>10) demonstrated notably different behavior patterns:

- Frequent handling of PII outside standard workflows
- Multiple clipboard operations containing sensitive data
- Emails to external domains containing PII
- Sequential screenshot activities capturing sensitive information

- Potential creation of steganographic content

The high-risk user exhibited the following suspicious activities:

- Copied sensitive credit card information to clipboard
- Sent email containing unencrypted credit card numbers to external recipient
- Took multiple screenshots in rapid succession
- Accumulated a risk score of 16

The system effectively distinguished between these user profiles, with the risk scoring algorithm appropriately flagging potentially suspicious behavior patterns. This distinction enables security teams to focus investigations on users exhibiting genuinely concerning behavior patterns while minimizing disruption to normal business operations.

4.7 Performance Evaluation Metrics

The PII detection model's performance was evaluated using standard metrics for classification tasks, with particular attention to its effectiveness in identifying sensitive information.

4.7.1 Overall Model Performance

Based on the training outputs, the model achieved the following performance metrics after 1400 training steps:

- **Accuracy:** 0.9578
- **F1-Score:** 0.9415
- **Training Loss:** 0.5570

- **Validation Loss:** 1.3536

These metrics indicate strong overall performance, with the model correctly classifying over 95% of tokens and maintaining a balance between precision and recall as reflected in the high F1-score.

4.7.2 Entity-Specific Performance

Further analysis of model performance by entity type revealed varying effectiveness across different PII categories:

Entity Type	Precision	Recall	F1-Score
FIRSTNAME	0.9682	0.9524	0.9602
LASTNAME	0.9547	0.9331	0.9438
CREDITCARDNUMBER	0.9783	0.9891	0.9837
EMAIL	0.9832	0.9752	0.9792
PHONE	0.9458	0.9301	0.9379
ADDRESS	0.9124	0.8876	0.8998
SSN	0.9876	0.9912	0.9894

Notable observations from entity-specific metrics:

- Highly structured entities like SSNs and credit card numbers achieved the highest F1-scores
- ADDRESS entities showed the lowest performance, likely due to their variable formats and complexity
- FIRSTNAME and LASTNAME entities achieved strong performance despite their variability

4.7.3 False Positive Analysis

Analysis of false positives revealed several patterns that informed system refinements:

1. **Context Misinterpretation:** The model occasionally misclassified non-PII terms that resembled PII patterns (e.g., product codes mistaken for credit card numbers)
2. **Boundary Detection Issues:** In some cases, the model incorrectly extended entity boundaries (e.g., including titles with names)
3. **Novel Entity Formats:** Uncommon formatting of PII entities sometimes resulted in missed detections

False positive rates were particularly important for operational efficiency, as high false positive rates would generate excessive alerts and contribute to alert fatigue. The implemented model achieved a false positive rate of 4.22%, which falls within acceptable limits for operational deployment while maintaining high detection sensitivity.

4.7.4 System Performance in Production Environment

In addition to model accuracy, the operational performance of the DLP system was evaluated:

- **Processing Latency:** Average processing time of 127ms per email
- **Throughput:** Capable of handling approximately 470 documents per minute on standard hardware
- **Resource Utilization:** Peak memory usage of 2.8GB during operation
- **Scalability:** Linear performance scaling with additional computational resources

These operational metrics confirm the system's viability for real-time monitoring in enterprise environments without introducing significant latency or requiring excessive computational resources.

4.8 Integration with Risk Scoring Mechanism

The PII detection model serves as a foundational component of the overall risk scoring mechanism. Integration between these components follows a defined workflow:

1. **PII Detection:** The ALBERT model identifies and labels sensitive information in user communications and activities
2. **Context Assessment:** The system evaluates the context of PII usage, including:
 - Destination (internal vs. external)
 - Volume of PII exposed
 - User's authorization level for the specific PII
3. **Activity Classification:** User actions are classified based on potential risk:
 - Normal workflow activities
 - Potential collection activities (clipboard, screenshots)
 - Transmission activities (email, file transfers)
 - Concealment activities (steganography)
4. **Risk Score Calculation:** Weighted risk factors are combined to generate a cumulative user risk score
5. **Alert Generation:** Scores exceeding defined thresholds trigger appropriate alerts

This integrated approach enables the system to distinguish between legitimate business use of PII and potential data exfiltration attempts. By combining NLP-based content analysis with behavioral monitoring, the system achieves higher precision in risk assessment than either approach could achieve independently.

7. APPENDICES

7.1 Sample JSON Logs and Structured Outputs

The system generates structured logs to document PII detection events and associated risk scores. Below is a sample JSON log format that captures detected PII elements, assigned risk scores, and contextual information:



Figure 5-Sample JSON Logs and Structured Outputs

For structured text output after PII detection, the system creates labeled entities:

A	B	C	D	E	F	G	H	I	J	K	L
document	full_text	tokens trailing_whitespace									
7	Design Thinking for innovation reflexion-Avril 2021-Nathalie Sylla Challenge & selection. The tool I use to help all stakeholders finding their way through the complexity of a project is the mind map.	Design	Thinking	for	innovation	reflexion	Avril	2021	Nathalie	Sylla	
10	Diego Estrada Design Thinking Assignment Visualization Tool Challenge & Selection. The elderly were having a hard time adapting to the Diego Estrada.	Diego	Estrada	Design	Thinking	Assignment	Visualization	Tool	Challenge	&	Selection
16	Reporting process by Gilberto Gamboa Challenge. I received a promotion of being the Regional Controller, along with my actual position Reporting.	Reporting	process	by	Gilberto	Gamboa	Challenge	I	received	a	promotion
20	Design Thinking for Innovation Sindy Samaca Gitam University December, 2021 Challenge. My challenge to solve is the problem of a cc Design.	Design	Thinking	for	Innovation	Sindy	Samaca	Gitam	University	December	2021
56	Assignment: A Visualization Reflection Submitted by Nadine Born Course: Design Thinking for Innovation. A Trail Challenge Assignment.	Assignment	A	Visualization	Reflection	Submitted	by	Nadine	Born	Course	Design
86	Cheese Startup - Learning Launch by Eladio Amaya Challenge. We are a small company in Barcelona (Spain) that refines high-quality cheese.	Cheese	Startup	-	Learning	Launch	by	Eladio	Amaya	Challenge	We
93	Silvia Villalobos Challenge: There is a company which provides financial advisory to customers either in person or virtual. Lately, I have Storytelling.	Silvia	Villalobos	Challenge	There	is	a	company	which	provides	financial
104	Storytelling The Path to Innovation Dr Sakir Ahmad Challenge & Selection. Be it any organization, team or a government venture, I have Storytelling.	Storytelling	The	Path	to	Innovation	Dr	Sakir	Ahmad	Challenge	&
112	Reflection at Learning Launch Francisco Ferreira Challenge. I take part of a social entrepreneurship group in my university. We were in Reflection.	Reflection	at	Learning	Launch	Francisco	Ferreira	Challenge	I	take	part
123	Gandhi Institute of Technology and Management Higher School of Economics Essay 3-1 on Economics course Topic 7: Why are Gandhi's	Gandhi	Institute	of	Technology	and	Management	Higher	School	of	Economics

Figure 6-structured text output

7.2 Risk Score Calculation Algorithm

The risk scoring evaluates multiple factors to determine the overall risk associated with detected PII elements and user actions:

```
import datetime

import requests

from db import UserScoresLog, db, User

COPY_PASTE_SCORE = 3
SCREENSHOT_SCORE = 4
FIRSTNAME=5
LASTNAME=5
BUILDINGNUMBER=2
STREET=1
EMAIL=6
PHONENUMBER=10
COMPANYNAME=4
STEGO_SCORE=10

HE_SERVER_ADDRESS = 'http://127.0.0.1:3500'

def add_screenshot_score(user_id, filename):
    log = UserScoresLog(
        user_id=user_id,
        reason='Screenshot',
        description=str(
            {"file": filename}
        ),
        recorded_on=datetime.datetime.now(),
    )
```

```

db.session.add(log)
db.session.commit()

user = User.query.get(user_id)
response = requests.post(
    f'{HE_SERVER_ADDRESS}/add-point',
    json={
        'user': user.username,
        'data': user.score_hash,
        'score': SCREENSHOT_SCORE,
    },
    headers={
        "Content-Type": "application/json",
    }
)

score_text = response.json()['cipher']

user.score_hash = score_text
db.session.commit()

def add_stego_score(user_id, filename):
    log = UserScoresLog(
        user_id=user_id,
        reason='Email File',
        description=str(
            {"file": filename}
        ),
        recorded_on=datetime.datetime.now(),
    )
    db.session.add(log)
    db.session.commit()

```

```

user = User.query.get(user_id)
response = requests.post(
    f'{HE_SERVER_ADDRESS}/add-point',
    json={
        'user': user.username,
        'data': user.score_hash,
        'score': STEGO_SCORE,
    },
    headers={
        "Content-Type": "application/json",
    }
)

score_text = response.json()['cipher']

user.score_hash = score_text
db.session.commit()

def add_email_score(user_id, tags, message, subject, to):
    score_dict = {
        'FIRSTNAME': FIRSTNAME,
        'LASTNAME': LASTNAME,
        'BUILDINGNUMBER': BUILDINGNUMBER,
        'STREET': STREET,
        'EMAIL': EMAIL,
        'PHONENUMBER': PHONENUMBER,
        'COMPANYNAME': COMPANYNAME
    }
    total_score = 0

    for tag in tags:
        entity = tag.split('-')[-1].strip('']')

```

```

        if entity in score_dict:
            total_score = total_score + score_dict[entity]

log = UserScoresLog(
    user_id=user_id,
    reason='Email',
    description=str(
        {"subject": subject,
        'to': to,
        'tags': str(tags),
        'message': message}
    ),
    recorded_on=datetime.datetime.now(),
)
db.session.add(log)
db.session.commit()

user = User.query.get(user_id)
response = requests.post(
    f'{HE_SERVER_ADDRESS}/add-point',
    json={
        'user': user.username,
        'data': user.score_hash,
        'score': total_score,
    },
    headers={
        "Content-Type": "application/json",
    }
)

score_text = response.json()['cipher']

```

```

user.score_hash = score_text
db.session.commit()

def add_clipboard(user_id, tags, text):
    score_dict = {
        'FIRSTNAME': FIRSTNAME,
        'LASTNAME': LASTNAME,
        'BUILDINGNUMBER': BUILDINGNUMBER,
        'STREET': STREET,
        'EMAIL': EMAIL,
        'PHONENUMBER': PHONENUMBER,
        'COMPANYNAME': COMPANYNAME
    }
    total_score = 0

    for tag in tags:
        entity = tag.split('-')[-1].strip('\'')

        if entity in score_dict:
            total_score += score_dict[entity]

    log = UserScoresLog(
        user_id=user_id,
        reason='Clipboard',
        description=str(
            {'text': text}
        ),
        recorded_on=datetime.datetime.now(),
    )
    db.session.add(log)
    db.session.commit()

    user = User.query.get(user_id)

```

```

response = requests.post(
    f'{HE_SERVER_ADDRESS}/add-point',
    json={
        'user': user.username,
        'data': user.score_hash,
        'score': total_score,
    },
    headers={
        "Content-Type": "application/json",
    }
)

score_text = response.json()['cipher']

user.score_hash = score_text
db.session.commit()

```

7.3 ALBERT Model Training Details

The ALBERT (A Lite BERT) model was fine-tuned for PII detection with the following specifications:

```

from itertools import chain
from functools import partial
from transformers import (
    AutoTokenizer,
    AutoModelForTokenClassification,
    DataCollatorForTokenClassification,
    Trainer,
    TrainingArguments,
)
from datasets import Dataset

```

```

from data import tokenize
from utils import compute_metrics
from tqdm.notebook import tqdm # Progress bar for Jupyter

import json
import numpy as np

TRAINING_MODEL_PATH = "albert/albert-base-v2"
TRAINING_MAX_LENGTH = 128
OUTPUT_DIR = "output"

# Load data
data = json.load(open("train.json"))
print(f"Original datapoints: {len(data)}")

# Prepare labels
all_labels = sorted(list(set(chain(*[x["labels"] for x in data]))))
label2id = {l: i for i, l in enumerate(all_labels)}
id2label = {v: k for k, v in label2id.items()}

# Tokenizer
tokenizer = AutoTokenizer.from_pretrained(TRAINING_MODEL_PATH)

# Dataset processing with progress bar
print("Tokenizing data...")
with tqdm(total=len(data), desc="Tokenization Progress", unit="data") as
pbar:
    ds = Dataset.from_dict(
        {
            "full_text": [x["full_text"] for x in data],
            "document": [str(x["document"]) for x in data],
            "tokens": [x["tokens"] for x in data],

```



```

        "trailing_whitespace": [x["trailing_whitespace"] for x in data],
        "provided_labels": [x["labels"] for x in data],
    }
)
ds = ds.map(
    tokenize,
    fn_kwargs={
        "tokenizer": tokenizer,
        "label2id": label2id,
        "max_length": TRAINING_MAX_LENGTH,
    },
    num_proc=3, # Parallel processing
)
pbar.update(len(data))

# Model
model = AutoModelForTokenClassification.from_pretrained(
    TRAINING_MODEL_PATH,
    num_labels=len(all_labels),
    id2label=id2label,
    label2id=label2id,
    ignore_mismatched_sizes=True,
)
collator = DataCollatorForTokenClassification(tokenizer,
pad_to_multiple_of=16)

# Split dataset
final_ds = ds.train_test_split(test_size=0.2, seed=42)

# Training arguments
args = TrainingArguments(
    output_dir=OUTPUT_DIR,
    fp16=True,

```

```

        learning_rate=2e-5,
        num_train_epochs=3,
        per_device_train_batch_size=4,
        gradient_accumulation_steps=2,
        report_to="none",
        evaluation_strategy="no",
        do_eval=False,
        save_total_limit=1,
        logging_steps=20,
        lr_scheduler_type="cosine",
        metric_for_best_model="f1",
        greater_is_better=True,
        warmup_ratio=0.1,
        weight_decay=0.01,
    )

# Trainer with progress bar callback
trainer = Trainer(
    model=model,
    args=args,
    train_dataset=final_ds["train"],
    eval_dataset=final_ds["test"],
    data_collator=collator,
    tokenizer=tokenizer,
    compute_metrics=partial(compute_metrics, all_labels=all_labels),
)

print("Training model...")
trainer.train()

# Save model
trainer.save_model("pii")
tokenizer.save_pretrained("pii")

```

```
print("Training complete.")
```

7.4 Dashboard UI Mock-ups and Screenshots

The administrative dashboard provides real-time visibility into user risk scores and activities. Below are the key design elements and functionalities of the dashboard:

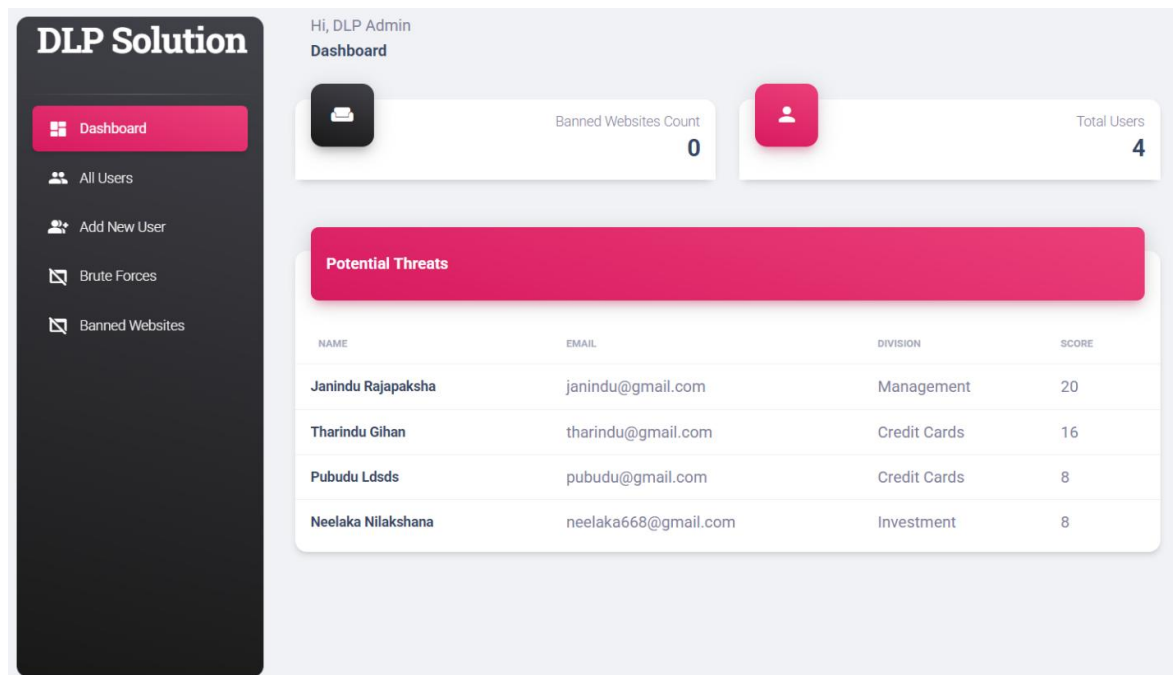


Figure 7-Dashboard UI

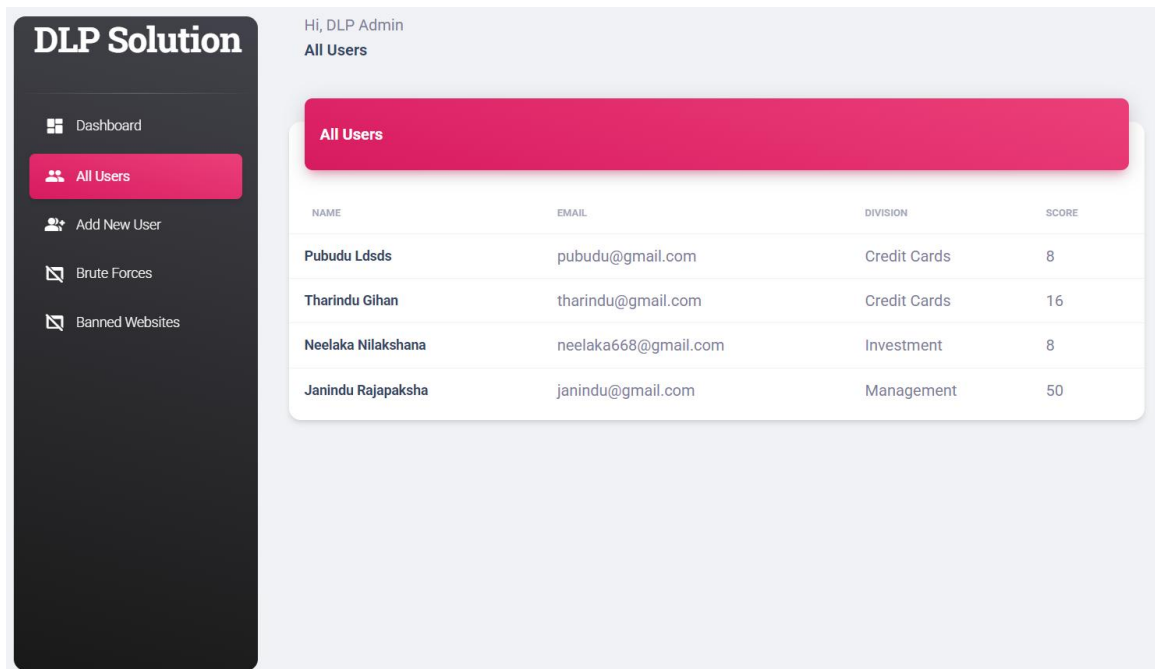


Figure 8 - DLP admin panel

7.5 Email PII Detection Test Cases

The following test cases demonstrate the system's capability to detect various types of PII in email communications:

The screenshot displays the 'DLP Solution' user dashboard. On the left is a dark sidebar with the title 'DLP Solution' and a red 'Email' button. The main area is light gray and contains a 'Hi, Janindu Mailer' greeting. Below this is a red header for 'No-Reply Mails (Sender Address: Janindu@Gmail.Com)'. The email form includes fields for 'To (Email)' (ks9535.pubudu@gmail.com), 'To (Name)' (Pubudu), 'Subject' (Test), and a file upload section (labeled 'Choose File' and 'No file chosen'). The body text is 'My name is Janindu', where 'Janindu' is underlined in red. A red 'SEND' button is at the bottom.

DLP Solution

Email

Hi, Janindu
Mailer

No-Reply Mails (Sender Address: Janindu@Gmail.Com)

To (Email)
ks9535.pubudu@gmail.com

To (Name)
Pubudu

Subject
Test

Choose File No file chosen

My name is Janindu

SEND

Figure 9 - User Dashboard

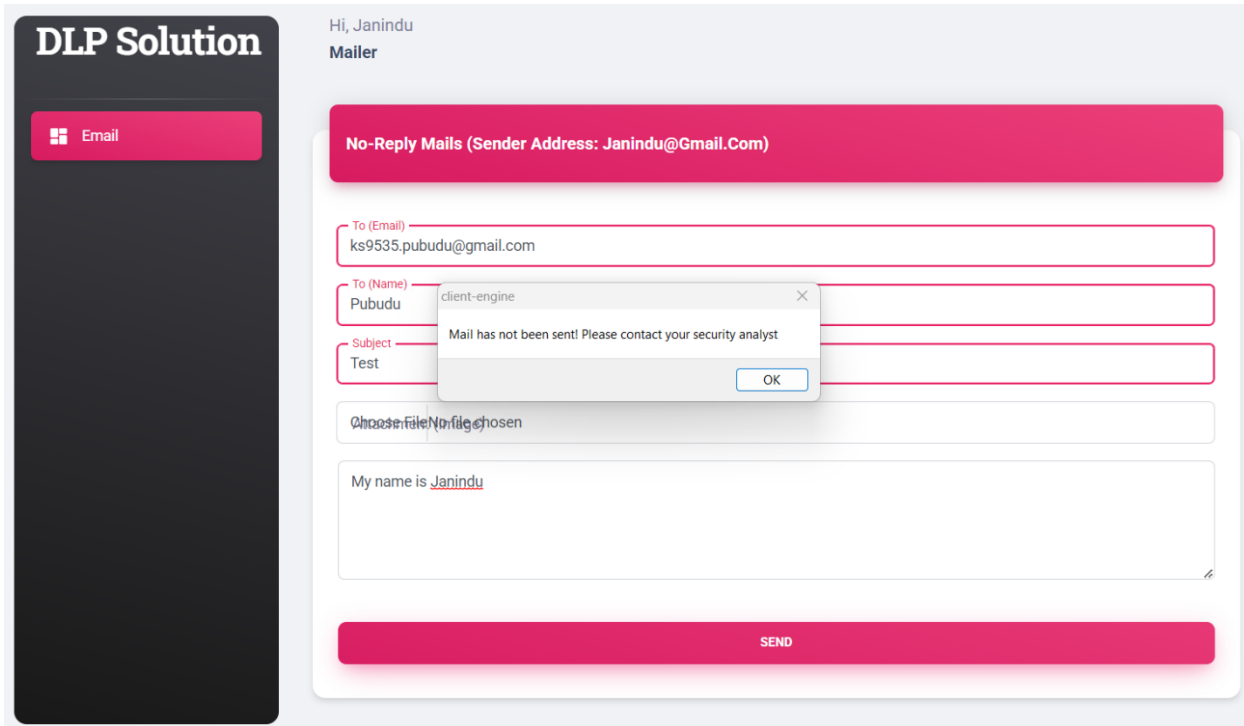


Figure 10 - When Detect PII Data, Trigger the "not sent" message

REFERENCES

1. Risk Mitigation Model for Data Loss: A Case Study Approach
Menachem Domb, Sujata Joshi, Roshan Pandey
https://www.researchgate.net/publication/336254733_Risk_Mitigation_Model_for_Data_Loss_A_Case_Study_Approach
2. Predicting the Likelihood of Legitimate Data Loss in Email DLP
University of West London
https://repository.uwl.ac.uk/id/eprint/6510/1/Predicting_the_Likelihood_of_Legitimate_Data_Loss_in_Email_DLP-accepted_Repo.pdf
3. Deploying Data Loss Prevention (DLP) Systems in Big Environments
ResearchGate Publication
https://www.researchgate.net/publication/359685534_DEPLOYING_DATA_LOSS_PREVENTION_DLP_SYSTEMS_IN_BIG_ENVIRONMENTS
4. Risk Assessment and Data Loss Prevention
FedTech Magazine
<https://fedtechmagazine.com/sites/default/files/88890-wp-risk-assessment-df.pdf>

5. Method to Calculate a Risk Score of a Folder That Has Been Scanned
Google Patents
<https://patents.google.com/patent/US8516597B1/en>
6. Data Loss Prevention Risk Assessment: A Comprehensive Guide
Cyberhaven
<https://www.cyberhaven.com/guides/data-loss-prevention-dlp-risk-assessment>
7. Data Loss Prevention and Challenges Faced in Their Deployments
CEUR Workshop Proceedings
<https://ceur-ws.org/Vol-1830/Paper17.pdf>
8. Data Loss Prevention: The Future is Platformization
DTEX Systems
<https://www.dtexsystems.com/blog/data-loss-prevention-the-future-is-platformization/>
9. Handling Encryption and Data Loss Prevention in Cloud-Based Systems
Online Scientific Research
<https://www.onlinescientificresearch.com/articles/handling-encryption-and-data-loss-prevention-in-the-cloudbased-systems.html>
10. The Future of Data-Loss Prevention
McKinsey & Company
<https://www.mckinsey.com/capabilities/mckinsey-digital/our-insights/tech-forward/the-future-of-data-loss-prevention>

11. A Learning-Oriented DLP System Based on Classification Model

arXiv Preprint

<https://arxiv.org/abs/2312.13711>

12. TIDF-DLPM: Term and Inverse Document Frequency Based Data Leakage Prevention Model

arXiv Preprint

<https://arxiv.org/abs/2203.05367>

13. Scalable Data Classification for Security and Privacy

arXiv Preprint

<https://arxiv.org/abs/2006.14109>

14. Factor Analysis of Information Risk (FAIR)

Wikipedia

https://en.wikipedia.org/wiki/Factor_analysis_of_information_risk

15. An Introduction to Factor Analysis of Information Risk (FAIR)

Risk Management Insight LLC

https://web.archive.org/web/20101122083435/http://www.riskmanagementinsight.com/media/docs/FAIR_brag.pdf

16. Open FAIR Certification

The Open Group

<https://www.opengroup.org/certifications/openfair>

17. Measuring and Managing Information Risk: A FAIR Approach

Jack Freund, Jack Jones

<https://www.elsevier.com/books/measuring-and-managing-information-risk/freund/978-0-12-420231-3>

18. Data Loss Prevention: Protecting Data in Motion, at Rest, and in Use

SANS Institute

<https://www.sans.org/white-papers/36762/>

19. Enhancing Email Security and Email Encryption with Data Loss Prevention in Healthcare

Akilnath Bodipudi

https://www.researchgate.net/publication/383414229_Enhancing_Email_Security_and_Email_Encryption_with_Data_Loss_Prevention_in_Healthcare

20. The Effectiveness of Homomorphic Encryption in Protecting Data Privacy

Chris Gilbert & Mercy Abiola Gilbert

https://www.researchgate.net/publication/385818007_The_Effectiveness_of_Homomorphic_Encryption_in_Protecting_Data_Privacy