Supplementary Document: Bibliometric Data Pipeline for Insider Threat Detection SLR

1. Overview

This document outlines the methodology used to generate the bibliometric visualizations for the Systematic Literature Review (SLR) titled "Insider Threat Detection Using Machine Learning." To ensure academic rigor and 100% representation of the 82-paper research corpus, the process utilized a hybrid data pipeline: combining the qualitative data extraction capabilities of NotebookLM for thematic auditing with the quantitative visualization power of Python (Google Colab) and MS Excel. This approach ensures that every figure (ranging from keyword frequency treemaps and word clouds to publisher distributions and JCR quartile rankings). To ensure 100% representation of the 82-paper research corpus and maintain academic rigor, a dual-stream data pipeline was utilized:

Stream A (Content-Based): Qualitative thematic extraction using NotebookLM and high-resolution visualization via Python (Google Colab).

Stream B (Structural-Based): Quantitative metadata mapping using Zotero and MS Excel, validated against the Web of Science (WoS) JCR database.

This dual approach ensures that every figure is grounded in manually audited metadata, providing a transparent and reproducible foundation for the study's bibliometric analysis.

2. Part I: Content-Based Bibliometric Analysis (Thematic)

This phase transformed the unstructured text of 82 full-text PDFs into structured thematic data.

2.1. AI-Driven Frequency Audit (NotebookLM)

To bypass the "Top-K" retrieval limitations of standard LLMs, all 82 PDFs were ingested into Google **NotebookLM** for a "Deep Audit."
To move beyond a simple summary, a " Comprehensive Audit" prompt was utilized. This stage converted unstructured text across 82 papers into structured frequency data.

The "Deep Audit" Prompting Strategy:

The analysis was executed using specific instructions to ensure statistical accuracy:
Prompt Snippet: *"Perform an exhaustive Frequency Audit of all 82 papers... Create a Table representing the global dataset. If a keyword appears in 40 papers, we expect to see the number 40... identify 'Minority Themes' found in outlier papers to ensure 100% representation."*

Statistical Calibration:

To ensure the data was ready for visualization, two specific metrics were calculated:

1. **Coverage % (Prevalence):** (f / 82) \times 100. This accounts for papers belonging to multiple categories (e.g., a paper using both *Deep Learning* and *Anomaly Detection*).
2. **Relative % (Distribution):** (f / \sum f) \times 100. This was the metric used for the **Treemap** and **Word Cloud** to ensure the total visual area equals 100%.

2.2 Python Visualization Logic (Figures 2 & 3)

The audit data was migrated to **Google Colab** to generate Figures 2 and 3.

- **Figure 2 (Treemap):** Size was mapped to Relative % using a customized discrete color palette for visual distinction.
- **Figure 3 (Word Cloud):** Configured with `collocations=False` to preserve technical phrases (e.g., "Deep Learning") as single units rather than fragmented words.

3. Technical Implementation: Python Source Code

The following scripts were executed to ensure high-resolution (300$DPI) output suitable for publication.

Code for Figure 2: Top 10 Keyword Treemap

```
import plotly.express as px
import pandas as pd
import textwrap
# 1. Dataset from NotebookLM Audit
data = {
   'Keyword': ['Anomaly Detection', 'Deep Learning', 'Behavioral Analysis', 'Graph Neural Networks',
            'Ensemble Learning', 'Data Imbalance Handling', 'Federated Learning',
            'NLP / Sentiment Analysis', 'Explainable AI (XAI)', 'IoT Security'],
   'Frequency': [42, 40, 24, 21, 19, 17, 12, 9, 8, 7],
   'Relative_%': [21.1, 20.1, 12.1, 10.6, 9.6, 8.5, 6.0, 4.5, 4.0, 3.5]
}
df = pd.DataFrame(data)

# 2. Styling and Rendering
def wrap_text(text, width=12): return "<br>".join(textwrap.wrap(text, width=width))
df['Label'] = "<b>" + df['Keyword'].apply(wrap_text) + "</b><br>" + df['Frequency'].astype(str) + " (" + df['Relative_%'].astype(str)
+ "%)"

My_colors = ['#8dd3c7', '#ffffb3', '#bebada', '#fb8072', '#80b1d3', '#fdb462', '#b3de69', '#fccde5', '#d9d9d9', '#bc80bd']
fig = px.treemap(df, path=['Label'], values='Relative_%', color='Keyword', color_discrete_sequence=My_colors)
fig.update_layout(title_text='<b>Figure 2. Top 10 Keyword Treemap Analysis (N=82)</b>', title_x=0.5)
fig.show()
```

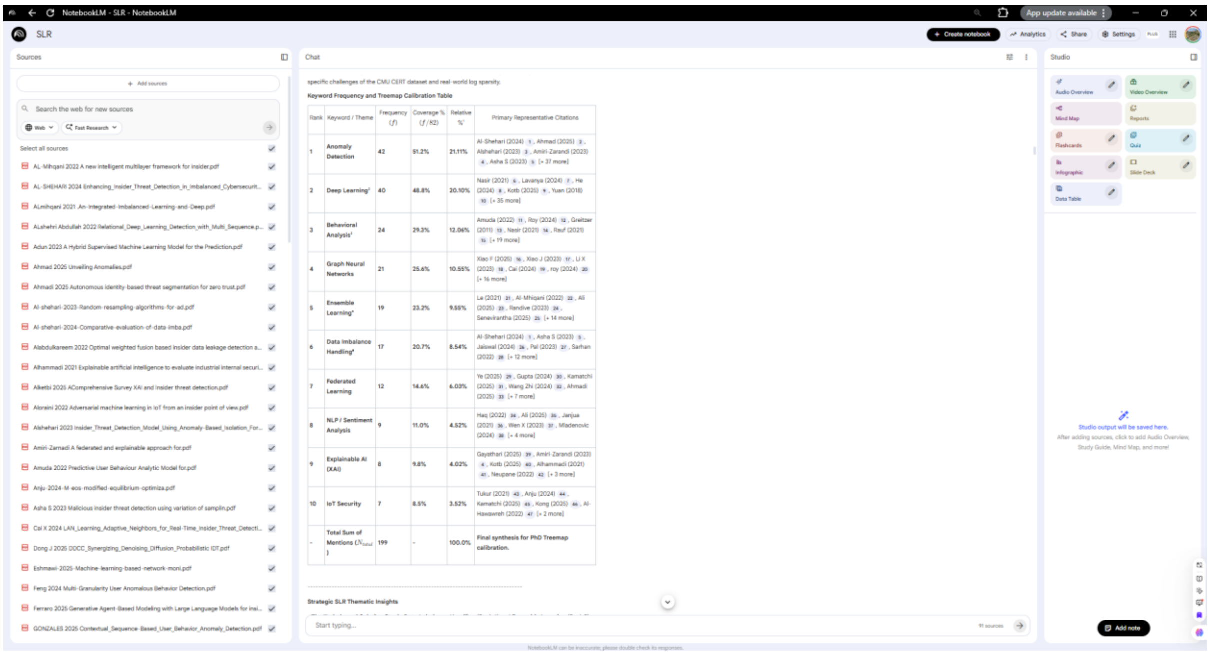
Code for Figure 3: Thematic Word Cloud

```
from wordcloud import WordCloud
import matplotlib.pyplot as plt

# 1. Thematic Dictionary (Top 10 + Minority Themes)
keywords_freq = {
   'Anomaly Detection': 42, 'Deep Learning': 40, 'Behavioral Analysis': 24,
   'Graph Neural Networks': 21, 'Ensemble Learning': 19, 'Data Imbalance Handling': 17,
   'Federated Learning': 12, 'NLP / Sentiment Analysis': 9, 'Explainable AI (XAI)': 8,
   'IoT Security': 7, 'Isolation Forest': 14, 'XGBoost': 13, 'Support Vector Machine': 12,
   'Hidden Markov Models': 11, 'Word2Vec': 9, 'Autoencoders': 9, 'Zero Trust': 6
}

# 2. Configure and Generate Word Cloud
wc = WordCloud(width=2000, height=1000, background_color='white', colormap='plasma',
          collocations=False, max_words=100, random_state=42).generate_from_frequencies(keywords_freq)

# 3. Render Visualization
plt.figure(figsize=(20, 10))
plt.imshow(wc, interpolation='bilinear')
plt.axis("off")
plt.title("Figure 3. Word Cloud of Insider Threat Detection Keywords", fontsize=30, pad=20)
plt.show()
```



Select all sources

- AL-Mihqani 2022 A new intelligent multilayer framework for insider.pdf
- AL-SHEHARI 2024 Enhancing_Insider_Threat_Detection_in_Imbalanced_Cybersecurit...
- ALmihqani 2021 .An-Integrated-Imbalanced-Learning-and-Deep.pdf
- ALshehri Abdullah 2022 Relational_Deep_Learning_Detection_with_Multi_Sequence.p...
- Adun 2023 A Hybrid Supervised Machine Learning Model for the Prediction.pdf
- Ahmad 2025 Unveiling Anomalies.pdf
- Ahmadi 2025 Autonomous identity-based threat segmentation for zero trust.pdf
- Al-shehari-2023-Random-resampling-algorithms-for-ad.pdf
- Al-shehari-2024-Comparative-evaluation-of-data-imba.pdf
- Alabdulkareem 2022 Optimal weighted fusion based insider data leakage detection a...
- Alhammadi 2021 Explainable artificial intelligence to evaluate industrial internal securi...
- Alketbi 2025 AComprehensive Survey XAI and Insider threat detection.pdf
- Aloraini 2022 Adversarial machine learning in IoT from an insider point of view.pdf
- Alshehari 2023 Insider_Threat_Detection_Model_Using_Anomaly-Based_Isolation_For...
- Amiri-Zamadi A federated and explainable approach for.pdf
- Amuda 2022 Predictive User Behaviour Analytic Model for.pdf
- Anju-2024-M-eos-modified-equilibrium-optimiza.pdf
- Asha S 2023 Malicious insider threat detection using variation of samplin.pdf
- Cai X 2024 LAN_Learning_Adaptive_Neighbors_for_Real-Time_Insider_Threat_Detecti...
- Dong J 2025 DDCC_Synergizing_Denoising_Diffusion_Probabilistic IDT.pdf
- Eshmawi-2025-Machine-learning-based-network-moni.pdf
- Feng 2024 Multi-Granularity User Anomalous Behavior Detection.pdf
- Ferraro 2025 Generative Agent-Based Modeling with Large Language Models for insi...
- GONZALES 2025 Contextual_Sequence-Based_User_Behavior_Anomaly_Detection.pdf

specific challenges of the CMU CERT dataset and real-world log sparsity.

Keyword Frequency and Treemap Calibration Table

Rank	Keyword / Theme	Frequency (f)	Coverage % ($f/82$)	Relative %[1]	Primary Representative Citations
1	**Anomaly Detection**	**42**	**51.2%**	**21.11%**	Al-Shehari (2024) [1], Ahmad (2025) [2], Alshehari (2023) [3], Amiri-Zarandi (2023) [4], Asha S (2023) [5] [+ 37 more]
2	**Deep Learning[2]**	**40**	**48.8%**	**20.10%**	Nasir (2021) [6], Lavanya (2024) [7], He (2024) [8], Kotb (2025) [9], Yuan (2018) [10] [+ 35 more]
3	**Behavioral Analysis[1]**	**24**	**29.3%**	**12.06%**	Amuda (2022) [11], Roy (2024) [12], Greitzer (2011) [13], Nasir (2021) [14], Rauf (2021) [15] [+ 19 more]
4	**Graph Neural Networks**	**21**	**25.6%**	**10.55%**	Xiao F (2025) [16], Xiao J (2023) [17], Li X (2023) [18], Cai (2024) [19], roy (2024) [20] [+ 16 more]
5	**Ensemble Learning***	**19**	**23.2%**	**9.55%**	Le (2021) [21], Al-Mhiqani (2022) [22], Ali (2025) [23], Randive (2023) [24], Senevirantha (2025) [25] [+ 14 more]
6	**Data Imbalance Handling***	**17**	**20.7%**	**8.54%**	Al-Shehari (2024) [1], Asha S (2023) [5], Jaiswal (2024) [26], Pal (2023) [27], Sarhan (2022) [28] [+ 12 more]
7	**Federated Learning**	**12**	**14.6%**	**6.03%**	Ye (2025) [29], Gupta (2024) [30], Kamatchi (2025) [31], Wang Zhi (2024) [32], Ahmadi (2025) [33] [+ 7 more]
8	**NLP / Sentiment Analysis**	**9**	**11.0%**	**4.52%**	Haq (2022) [34], Ali (2025) [35], Janjua (2021) [36], Wen X (2023) [37], Mladenovic (2024) [38] [+ 4 more]
9	**Explainable AI (XAI)**	**8**	**9.8%**	**4.02%**	Gayathari (2025) [39], Amiri-Zarandi (2023) [4], Kotb (2025) [40], Alhammadi (2021) [41], Neupane (2022) [42] [+ 3 more]
10	**IoT Security**	**7**	**8.5%**	**3.52%**	Tukur (2021) [43], Anju (2024) [44], Kamatchi (2025) [45], Kong (2025) [46], Al-Hawawreh (2022) [47] [+ 2 more]
-	**Total Sum of Mentions (N_{total})**	**199**	-	**100.0%**	**Final synthesis for PhD Treemap calibration.**

Strategic SLR Thematic Insights

NotebookLM can be inaccurate; please double check its responses.

Figure 1. Interface of the Deep Audit Process in NotebookLM.

The above snapshot demonstrates the ingestion of the 82-paper research corpus (left pane) and the generation of the Keyword Frequency and Treemap Calibration Table (center pane) used as the primary data source for subsequent Python-based visualizations.

4. Part II: Structural Bibliometric Analysis (Metadata)

This phase utilized the bibliographic metadata of the 82 papers to map the research landscape's quality and trends.

4.1. Metadata Extraction and Data Cleaning (Zotero)

The primary metadata was managed via **Zotero**. To ensure 100% accuracy, a "Clean-Room" approach was adopted:

1. **Export:** A CSV file was generated from the Zotero library containing the 82 studies.
2. **Manual Verification:** Every entry was manually cross-referenced against the original PDF metadata to correct any errors in **Publication Year**, **Journal Name**, and **Publisher**.

4.2. Quantitative Mapping via MS Excel (Figures 4–7)

Using the verified CSV, MS Excel was utilized to generate the structural visualizations:

- **Figure 4 (Publication by Year):** A column chart illustrating publication volume between 2021–2025.
- **Figure 5 & 6 (Publisher/Journal Distribution):** Mapping the corpus to major academic houses (IEEE, Elsevier, Springer).

- **Figure 7 (JCR Quality Audit):** Each journal was manually cross-referenced with the **Web of Science (WoS)** database to identify its **JCR Quartile (Q1–Q4)**.

5. Summary of the Data Pipeline

The following data served as the input for the final visualizations:

Stage	**Process**	**Primary Tool**	**Output**
Collection	Search & Ingestion	Zotero	Master Bibliography
Thematic Audit	Content Deep-Scan	NotebookLM	Frequency Table
Thematic Viz	Visual Generation	Python (Colab)	Figure 2 & 3
Metadata Audit	Validation	Manual Review	Verified Master CSV
Structural Viz	Trend Mapping	MS Excel	Figure 4, 5, & 6
Quality Audit	JCR Ranking	Web of Science	Figure 7 (Pie Chart)