ECTI Transactions on Computer and Information Technology
Journal homepage: https://ph01.tci-thaijo.org/index.php/ecticit/
Published by the ECTI Association, Thailand, ISSN: 2286-9131

# FEDIS: Graph-Based Social Media Evidence Collection with Correlation-Aware Forensic Analysis

Somchart Fugkeaw[1], Pratuck Khubpatiwitthayakul[2], Pongsapon Ateetanan[3] and Diyorbek Bakhtiyor Ugli Okyulov[4]

## ABSTRACT

Online Social Networks (OSNs) have emerged as a major source of digital evidence in cybercrime investigations, abuse detection, and online incident analysis. Publicly available data, such as posts, comments, reactions, and user interactions, provide critical insights into suspicious activities and behavioral patterns. However, extracting and analyzing such data in a forensic context remains challenging. Existing social media data acquisition approaches primarily rely on web scraping or API-based techniques that produce unstructured outputs (e.g., raw text and images) without preserving the relationships among entities such as users, posts, and interactions. This results in the loss of contextual information essential for forensic analysis. Furthermore, data collection and analysis are often performed separately, resulting in delayed investigations and limited real-time insight. To address these limitations, this paper focuses on two key forensic requirements: (i) efficient and structured acquisition of social media evidence and (ii) correlation-aware analysis of interactions. We propose FEDIS, a unified forensic data acquisition and analysis system that integrates a hybrid DOM-TAO data model, graph-based representation, and parallel keyword-based search. Experimental results demonstrate that FEDIS achieves complete relationship preservation, improves data collection completeness, and significantly reduces search latency compared to traditional approaches, making it practical for real-world social media forensic investigations.

*Article information:*

**Keywords:** Online Social Network, Data Forensics, Document Object Model, The Associations and Objects, Graph Database

**Article history:**
Received: September 30, 2025
Revised: March 26, 2026
Accepted: April 23, 2026
Published: April 30, 2026 (Online)

**DOI:** 10.37936/ecti-cit.2026202.264177

## 1. INTRODUCTION

Online Social Networks (OSNs) have become an important source of digital evidence in cybercrime investigations, abuse detection, and online incident analysis. Publicly available data, such as posts, comments, reactions, and user interactions, provide valuable insights into suspicious behaviors and relationships among entities. Recent studies demonstrate that social network data can be effectively leveraged for forensic analysis tasks such as cross-network identity linkage and structural behavior analysis [1], as well as community-level investigation and anomaly detection [2]. However, effectively acquiring and analyzing such data in a forensic context remains a challenging task.

Existing social media data acquisition approaches primarily rely on web scraping tools or platform APIs [16, 18, 20], which typically produce unstructured outputs such as raw text and images. While these methods are effective for collecting content, they fail to preserve the inherent relationships among entities, including users, posts, comments, and interactions. This limitation has also been highlighted in prior surveys on social network forensics, which emphasize the lack of structured and correlation-aware data-acquisition mechanisms [3]. Consequently, the loss of contextual and relational information significantly limits the effectiveness of forensic analysis, where understanding the connections between entities is often more important than analyzing isolated data points. Moreover, data collection and analysis are typically performed separately, resulting in delayed investigations and limited real-time insight.

Recent studies further highlight both the opportunities and risks of leveraging social media data for

[1,2,3,4]The authors are with the School of ICT, Sirindhorn International Institute of Technology, Thammasat University, Pathum Thani, Thailand, Email: somchart@siit.tu.ac.th, pratuck.khub@gmail.com, pongsapon.ateeta@gmail.com and diyorbekokyulov@gmail.com

forensic purposes. For example, social network data can be used to extract cyber threat intelligence and analyze attack patterns [33], while adversarial manipulation of social graphs can mislead analysis and compromise investigation accuracy [34]. These challenges underscore the need for a forensic data acquisition framework that not only efficiently collects data but also preserves structural and relational context for reliable analysis.

Several existing approaches attempt to support social media data collection and analysis through feature extraction, ontology-based modeling, and visualization techniques [4, 9, 28, 29, 30]. However, these methods either focus on specific aspects such as data extraction or visualization, or fail to provide integrated support for relationship-aware data modeling and efficient search. In particular, most existing solutions do not simultaneously support structured data acquisition, correlation-preserving modeling, and real-time analysis within a unified framework.

To address these limitations, this work focuses on two key forensic requirements: (i) efficient and structured acquisition of social media evidence, and (ii) correlation-aware analysis of interactions among entities. We propose FEDIS, a unified system for the acquisition and analysis of forensic social media data. The proposed system integrates a hybrid DOM-TAO data model, which combines the hierarchical structure of the Document Object Model (DOM) [27] with the relationship-oriented TAO model [13] to capture both content structure and entity relationships. The collected data are represented using a graph-based model to preserve correlations among users, posts, and interactions.

In addition, FEDIS incorporates a parallel data collection mechanism and a multi-keyword search algorithm over the graph database to enable efficient retrieval of relevant evidence along with its associated context. This unified design allows investigators to seamlessly perform data acquisition, visualization, and analysis within a single system, improving both efficiency and effectiveness in forensic investigations.

The main contributions of this paper are summarized as follows:

1. **A hybrid DOM-TAO data model for structured social media evidence acquisition.** We design a unified data model that integrates hierarchical content representation with relationship-aware modeling, enabling the preservation of both structural elements (e.g., posts, comments, multimedia) and interaction semantics among entities.
2. **A graph-based representation for correlation-aware forensic analysis.** We develop a graph data model that captures entities and their interactions as nodes and relationships, allowing efficient storage, traversal, and contextual exploration of social media evidence.
3. **A parallel multi-keyword search mechanism over graph data.** We propose a search algorithm that supports concurrent processing of query terms and retrieves not only matching entities but also their associated context, facilitating comprehensive forensic investigation.
4. **An integrated system for real-time data acquisition, visualization, and analysis.** We implement the FEDIS system, which unifies data collection, modeling, and querying within a single framework, and validate its effectiveness through experiments on real-world social media datasets, assessing data completeness, search efficiency, and system performance.

## 2. RELATED WORK

Existing digital forensics solutions for social networks include techniques such as online scanning, scraping, and keyword-based search queries to identify relevant content on social media platforms. This includes searching for posts, comments, messages, profiles, groups, devices, or hashtags related to the investigation. This section discusses literature in two research areas: data acquisition from social media and forensic data search approaches.

Typically, data acquisition from social media applications is performed through data extraction [4, 9] or scraping [16-22]. For example, in [4], the authors proposed an approach for categorizing Twitter authors that integrates feature extraction with the transformation of high-dimensional data into a kernel matrix. They have proposed a system that autonomously gathers Twitter data, extracts features relevant to author clustering on the platform, and uses an unsupervised learning module. This study focuses on authorship analysis to address the inherent challenges of anonymity in internet services, particularly on the Twitter platform, where users can manage multiple accounts.

In [9], the authors introduced an ontology-based framework for digitally profiling attributes that are interconnected across various social media platforms. The scheme also facilitates the identification of digital evidence to narrow down the relationship with potential suspect(s) involved in cybercrime on social media. In this context, user profiling relies on domain-specific digital artifacts defined by a set of digital attributes.

Regarding social media data collection using the scraping technique, most works used tweets, APIs, and programming libraries to scrape data from the given webpage and crawl associated pages.

In [16], the authors analyzed information flow in social media, examining how users search for information and how it is presented to them. The information extraction method uses web scraping, imple-

mented through the Facebook Developers API and Twitter Developers API. Extracted information is matched with user preferences using regular expressions (Regex), a language construct used for matching text patterns. The algorithm development phase includes designing snippet code for web scraping.

In [18], J. You et al. proposed a scraping method called DeepScrap for collecting tweets data. The system performs high-speed, comprehensive scraping of recent tweets from a specific user's page, bypassing Twitter API rate limits. To enhance DeepScrap's crawling speed, they developed a multiprocessing architecture that assigns different IP addresses to processes while adhering to Twitter's robots.txt.

In [19], the authors collected tweet data using Twitter's Application Programming Interface (API). The scraping procedure was implemented using the Python Tweepy library. The collected data were categorized according to a social media framework, geographical regions, and user demographics. The findings indicate that Twitter user patterns are instrumental in both raising awareness and coordinating relief efforts during disasters. Different user groups engage with Twitter in ways that align with its conventional functions, highlighting the platform's utility in crises.

In [21], the authors present two approaches for accessing the information needed for the credibility model: web scraping and social media APIs. Users or developers can configure the system to gather information solely through web scraping or by combining it with available APIs. The proposed system can also automatically perform real-time credibility analysis.

Some tools [28, 29, 30, 32] were designed to collect and visualize data. However, they did not focus on the relationships among message posts and their associated contexts.

Recently, a few studies have addressed data collection from Facebook due to its more intricate data patterns, complex associations, and emotional context in posts. So far, the web scraping tool Apify [22] [15] has been used to gather data from Facebook posts. However, there are no existing tools or studies that provide comprehensive data collection capabilities, including visualized data pattern correlations and multi-keyword search functionality to display associated contexts, which are essential for forensic analysis.

## 3. BACKGROUND

This section introduces the concepts of The Associations and Objects (TAO), the Document Object Model (DOM) tree, and the graph-based model, which are the core constructs of our proposed data collection, visualization, and search.

The **Document Object Model (DOM)** [26] is an API for programming with HTML and XML documents. It outlines the logical structure of these documents and provides a method for accessing and manipulating them. In the context of the DOM specification, "document" is broadly defined; XML, in particular, is increasingly used to represent various types of information stored in different systems. Traditionally viewed as data, XML presents this information as documents, which the DOM can manage.

In the DOM, documents have a logical structure that resembles a tree; more accurately, it is akin to a "forest" or "grove" that can contain multiple trees. However, the DOM does not mandate that documents be implemented strictly as a tree or grove, nor does it dictate the specific implementation of relationships among objects. Essentially, the DOM specifies the logical model for the programming interface, and this logical model can be implemented in any manner that is convenient for a particular application. In this specification, the term "structure model" describes the tree-like representation of a document; terms like "tree" or "grove" are avoided to avoid implying a specific implementation method. A key feature of DOM structure models is structural isomorphism. If two different DOM implementations produce representations of the same document, they will yield identical structure models, with the same objects and relationships.

**The Associations and Objects (TAO)** [13] is a geographically distributed data store that provides efficient and timely access to the social graph for Facebook's demanding workload using a fixed set of queries. TAO provides fundamental access to the nodes and edges of a dynamically changing graph in data centers across various regions. It is heavily optimized for read operations and prioritizes efficiency and availability over consistency. In TAO, objects are typed nodes, and associations are typed directed edges between these objects. Each object is identified by a unique 64-bit integer (id). Associations are characterized by the source object (id1), the association type (atype), and the destination object (id2). Only one association of a given type can exist between any two objects. Both objects and associations can store data as key→value pairs, with a schema per type that lists possible keys, their value types, and default values. Each association also includes a 32-bit time field, which is crucial for query operations.

**Definition 1 (DOM-TAO Social Media Data Model)**
DOM-TAO (Document Object Model – The Associations and Objects) is a hybrid data modeling framework that combines (i) the hierarchical structure of the Document Object Model (DOM) and (ii) the relationship-centric design of the TAO model. The goal is to enable efficient, scalable, and query-friendly representation of social media data.

**1) DOM-based Hierarchical Structure.** Social media content is represented as a tree-like structure:
- $U_i$: user $i$
- $P_{ij}$: post $j$ created by user $i$
- $C_{ijk}$: comment $k$ on post $j$ by user $i$
- $M_{ijkl}$: multimedia element $l$ attached to comment $k$

The hierarchy follows:

$$U_i \rightarrow P_{ij} \rightarrow C_{ijk} \rightarrow M_{ijkl}$$

**2) TAO-based Relationships.** Entities are organized into sets:
$U = \{Ui\}, P = \{P_{ij}\}, C = \{C_{ijk}\}, M = \{M_{ijkl}\}$
Associations between entities are defined as:
- F $(U_i, U_j)$: friendship between users
- $R(U_i, P_{jk})$: user $i$ reacts to post $k$ of user $j$
- $S(U_i, P_{jk})$: user $i$ shares post $k$ of user $j$
- $A(P_{ij}, C_{ijk})$: comment belongs to a post

**3) Hybrid DOM-TAO Representation.** The unified model combines hierarchy and relationships:
- Nodes: represent hierarchical entities
  $Node(U_i) \rightarrow \{P_{i1}, P_{i2}, \ldots\}$
  $Node(P_{ij}) \rightarrow \{C_{ij1}, C_{ij2}, \ldots\}$
- Edges: represent TAO relationships
  $Edge(U_i, U_j) = F(U_i, U_j)$
  $Edge(U_i, P_{jk}) = \{R, S\}$

**4) Concrete Graph Mapping (Neo4j Example).** To improve practical understanding, the DOM-TAO model can be directly mapped into a property graph:
- *Node labels: User, Post, Comment, Media*
- Relationships:
  – $(:User) - [:FRIEND] - > (:User)$
  – $(:User) - [:POSTED] - > (:Post)$
  – $(:Post) - [:HAS_COMMENT] - > (:Comment)$
  – $(:Comment) - [:HAS_MEDIA] - > (:Media)$
  – $(:User) - [:REACT] - > (:Post)$
  – $(:User) - [:SHARE] - > (:Post)$
- Indexed fields: $user_id, post_id, timestamp$

**Example Query (Cypher):**
$MATCH(u:User) - [:POSTED] - > (p:Post)$
*[:HAS_COMMENT]-> WHERE u.user_id = "U1"*
$RETURN p, c$

**Summary**. DOM-TAO provides a unified abstraction where hierarchical content (DOM) and cross-entity interactions (TAO) are jointly modeled as a graph. This enables efficient data collection, scalable storage, and expressive querying for social media analytics.

Figure 1 shows an example of the DOM-TAO model presented in graph-based modeling.

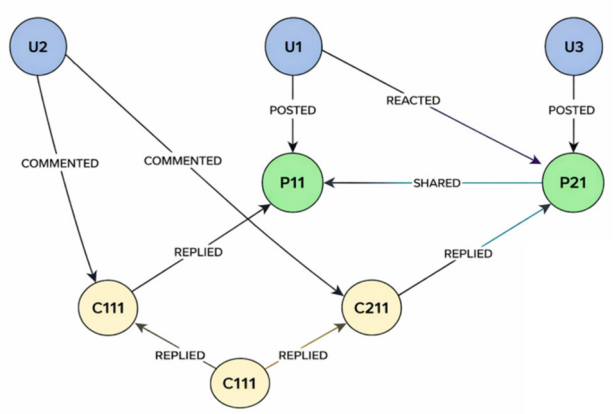



***Fig.1:*** *The example of a graph representation of the DOM-TAO Social Media Data.*

As shown in Figure 1, in this model, users, posts, comments, and multimedia elements are represented as nodes. In contrast, interactions such as friendships, likes, shares, and comment associations are represented as edges connecting these nodes. This graph-based representation enables the system to capture both static content and dynamic interactions on social networking platforms. The hierarchical structure of the DOM captures the nested relationships within social media content, including replies to comments, as well as images, videos, and hyperlinks embedded in posts. This ensures that the structural dependencies and contextual layering of content are preserved in the forensic model. At the same time, the TAO provides efficient management of the complex, large-scale associations between entities, including one-to-many and many-to-many relationships. For instance, a single user can be connected to thousands of posts and friends, and each post can accumulate likes, shares, and threaded discussions over time. TAO enables these relationships to be updated, retrieved, and queried in real time, thereby supporting both high-throughput data acquisition and low-latency forensic analysis. By combining DOM's fine-grained structural representation with TAO's scalable relational mapping, the proposed model ensures that forensic investigators can navigate seamlessly between micro-level details (e.g., the origin of a suspicious comment) and macro-level patterns (e.g., coordinated interactions across groups of users), making the system suitable for large-scale and real-time social media investigations.

## 4. OUR PROPOSED SYSTEM

Our proposed system is designed to collect and visualize threads and their contents from social networking platforms. The system represents the correlation between the data and users for forensics purposes in the form of an interactive graph model having nodes as entities within the social network that users interact with (users, posts, or comments) and

relationships as interactions or associations (actions of POSTED, REACTED, or REPLIED) using Neo4j [11]. The search function is also implemented in this tool to provide a more efficient view of the data of interest. The search function is also implemented in this tool to provide a more efficient view of the data of interest. Figure 2 shows the system model of our FEDIS.

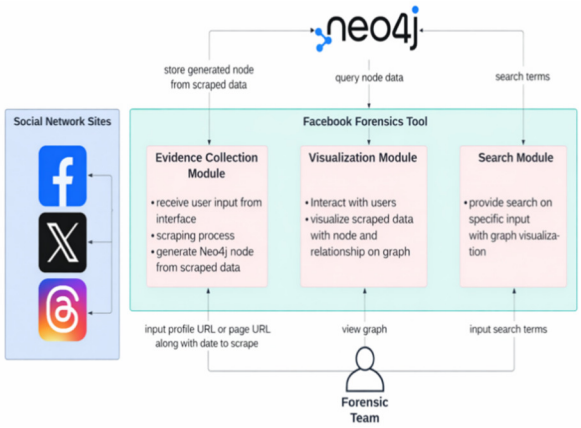



***Fig.2:*** *The proposed FEDIS System Model.*

As shown in Figure 2, each module can interact with users via a web interface. There are 5 major system entities as follows:

*A. Social Network Sites*

This component is the ideal target of the Evidence Collection module, where the evidence-collection process performs its tasks for every associated component. This contains all posts posted by the target user. On the sites, the structure is designed around a post as the main element, with sub-elements such as comments or likes. Facebook and Twitter (X) are examples of social networking applications that use this kind of structure.

For one showing window of the sites, there might be several posts. A post might include comments, likes, shares, captions, date and time, or multimedia (such as images or videos). These elements will be extracted from the sites during the evidence collection process to create a Neo4j node with properties and relationships.

*B. NEO4J Graph Database*

Neo4j, the graph database, is selected for preserving evidence and supporting search queries. Neo4j ensures low-latency and high-throughput database interactions through its native API when running in the Node.js environment we used. Neo4j also shows correlations between users and their actions (posts, likes, and comments). The graph model has achieved this goal by representing nodes and relationships. In this paper, we used full-text indexes to support our Search module, as they are suitable for precise keyword-based searches and offer higher scalability than traditional or other indexing approaches. Figure 3 illustrates how the Facebook posts and their related contexts, such as users and relationships, are analyzed and modeled in the Neo4J database.

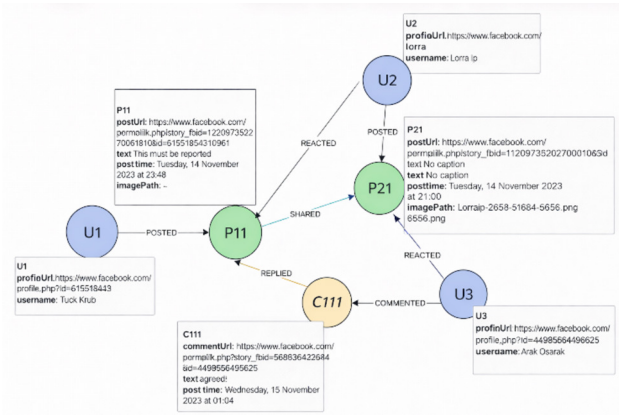



***Fig.3:*** *An Example of a Neo4j graph capturing Facebook Posts.*

Figure 3 illustrates an example graph constructed using the proposed DOM-TAO model. The graph consists of six nodes and seven relationships, representing interactions among three users, denoted as $U_1$, $U_2$, and $U_3$. The nodes include user, post, and comment entities, each enriched with associated attributes such as identifiers, content, timestamps, and metadata. The relationships capture various types of user activities, including *POSTED, COMMENTED, REPLIED, REACTED,* and *SHARED*, thereby reflecting the interaction semantics defined in the DOM-TAO model. Unlike traditional DOM-based representations, which primarily capture hierarchical content structure, the DOM-TAO model explicitly models cross-entity interactions, enabling richer representation of user behavior and social connections. This graph structure allows the system to preserve both content and interaction context, supporting efficient traversal, relationship discovery, and forensic analysis of user activities.

*C. The Evidence Collection Module*

This module accepts a public profile URL and a specific time range as input to narrow the scope of the scrape boundary. The scraped result will be ingested into the Neo4j graph database simultaneously. The Neo4j graph database will index necessary fields to support Boolean full-text search [12] and provide an API for the visualization module to display an interactive graph on the web interface.

The module is designed based on DOM-TAO, where the concept of object associations is based on TAO. In the context of social network applications, these objects might include users, pages, groups, and events, and associations represent relationships between them (e.g., friendships, likes, memberships).

According to the TAO data model, some object types and associations are not of interest to us or are outside our scope at this time. In TAO, objects such as CHECKIN and LOCATION are used, as are associations such as FRIEND and CHECKIN. To reduce the object and association dimensions, we focused on 3 objects (Post, Account, Comment) and 3 associations (POSTED, REACTED, REPLIED) for our proposed tool, as shown in Figure 4.

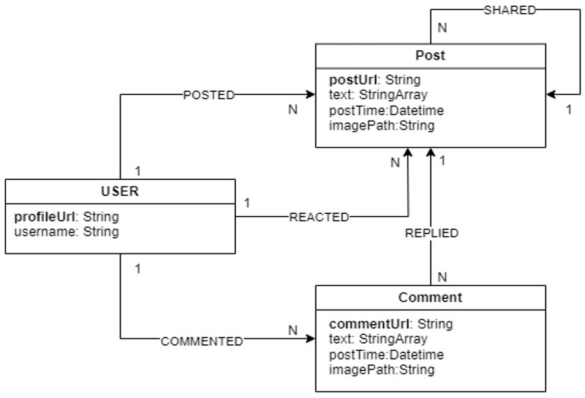



***Fig.4:*** *Proposed Graph model before being ingested into Neo4j.*

These objects and associations are used when creating a graph with nodes and relationships at Neo4j for each scraped social media data. The existing properties in Figure 4 are an example of Facebook data.

The process of evidence collection in this module is designed to extract the text/content we are interested in from the thread by using the search for unique element locators in the Document Object Model (DOM) tree. Since XPath is considered a peculiarity of modern site development with variable class names and identifiers, we use XPath to locate the specific element we are interested in [23].

In our system, there are two major algorithms designed to ingest the target social media URL and systematically collect data. We adopt a multi-threaded programming approach to streamline data scraping and profiling into Neo4j. Algorithm 1 illustrates the evidence-collection process for Facebook.

As shown in Algorithm 1, the initial-stage evidence collection process begins by receiving the target URL and the specified time range from the user via the web application interface, followed by initializing the required libraries and variables. The system then launches a browser session to access the target social media page within a permitted access context. Once the target URL is loaded, the system collects posts within the visible window; if none are found, the process terminates and returns no results. For each collected post, the corresponding post URL is accessed to perform a detailed extraction of relevant attributes, including timestamps, post content, and

**Algorithm 1** Initial Evidence Collection from a Facebook Target URL

**Require:** Target URL $url$, start date $StartDate$, end date $EndDate$
**Ensure:** Candidate post URLs within the specified time range are collected
$validStartDate \leftarrow StartDate$
$validEndDate \leftarrow EndDate$
Initialize empty sets $U, P, C, M, PostURLSet \leftarrow \emptyset$
Load required libraries and initialize the browser session
Authenticate to Facebook using permitted access context
Navigate to target URL $url$
**while** true **do**
  $PostURLSet \leftarrow GetShowingPostURL()$
  **if** $PostURLSet = \emptyset$ **then**
    **break**
  **end if**
  **for all** $postURL \in PostURLSet$ **do**
    $(fullHref, postTime) \leftarrow ExtractPostMetadata(postURL)$
    **if** $fullHref \neq \emptyset$ **and** $postTime \in [validStartDate, validEndDate]$ **then**
      $cleanURL \leftarrow CleanURL(fullHref)$
      **if** $cleanURL \neq \emptyset$ **then**
        Store $cleanURL$ for detailed evidence collection
        Create preliminary account/post nodes in Neo4j
      **else**
        Log partial post metadata to JSON
      **end if**
    **end if**
  **end for**
  **if** no additional posts are found **or** page bottom is reached **then**
    **break**
  **end if**
  Scroll down to load more posts
**end while**
**return** $PostURLSet$

associated interactions (e.g., comments). These extracted elements are subsequently used to construct nodes and relationships in the Neo4j graph database. Once the target URL is accessed, collect the available posts in the showing window from the target account; if none are available, end the process and return no collection result. For previously collected posts, each post's URL is accessed to obtain the post time and other entities (e.g., post content and comments), and then a node and relationship are created in Neo4j. The main-stage evidence-collection process will be started as depicted in Algorithm 2. In addition, when the collection process ends for each showing post on the window, the current window will scroll to the next post.

As shown in Algorithm 2, the algorithm first extracts the information in a 'post by post' manner, as seen from Algorithm 1, such that the uninterested posts outside the time range are avoided while ex-

tracting a post. The commentator, comment content, likers, and multimedia are then retrieved. Every time the information is extracted, it is promptly used to create a node and relationship in Neo4j.

**Algorithm 2** Post Evidence Extraction

**Require:** Post URL $targetURL$
**Ensure:** Structured evidence stored in Neo4j

Open $targetURL$ in browser
$(postContent, postTime, mediaSet, commentSet, likerSet, sharedPost) \leftarrow$ ExtractPostEntities($targetURL$)
$P \leftarrow$ CreatePostNode($postContent, postTime, mediaSet$)
$U \leftarrow$ ExtractOwnerProfile($targetURL$)
Create $(U) \xrightarrow{\text{POSTED}} (P)$
**if** $sharedPost \neq \emptyset$ **then**
    Create $(P) \xrightarrow{\text{SHARED}} (sharedPost)$
**end if**
**for all** $comment \in commentSet$ **do**
    $C \leftarrow$ CreateCommentNode($comment$)
    $U_c \leftarrow$ ExtractCommentator($comment$)
    Create $(U_c) \xrightarrow{\text{COMMENTED}} (C)$
    Create $(C) \xrightarrow{\text{REPLIED}} (P)$
**end for**
**for all** $liker \in likerSet$ **do**
    $U_l \leftarrow$ ExtractUser($liker$)
    Create $(U_l) \xrightarrow{\text{REACTED}} (P)$
**end for**
Save multimedia (if any)
Close browser tab

After discovering posts within the user-specified time range, the process automatically opens a new tab containing only that post. This is to ensure that the process extracts only the information from the specific post. Consequently, the post content, image, comments, and likes will be extracted when the target post URL is accessed. If multimedia is found within the post, it is saved locally and used as a property when creating a post node. In addition, if the target is a shared post, then the shared relationship is created. In both collecting comments and collecting likers, the process opens the like window, extracts all profiles, cleans the URLs, and creates a like relationship between each profile and the host post node. Once the first showing window's information is collected, if the window can still be scrolled down or still has some remaining comments/likes, it automatically scrolls down the window and repeats the collection process.

### *D. Data Visualization Module*

This module is designed to interactively visualize the correlation between nodes and the relationships inferred from the Search module's collected information. Before performing any search, this module visualizes all nodes and relationships in the Neo4j database. The module uses an API to access Neo4j from the application and retrieve the collected data from the Neo4j database. Afterward, customization on the graph model for more contrast visualization could be performed if needed. The customization includes adding colors to distinct nodes, indicating relationships with arrowheads, and labeling relationship types on the arrows. Figure 5 presents the layout of the graph visualization of the Facebook posts.

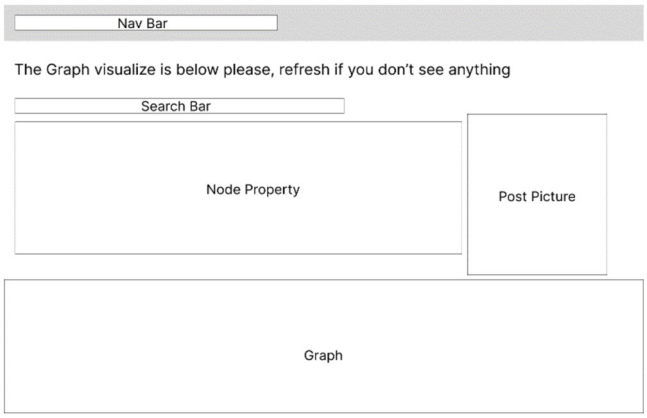



***Fig.5:*** *Visualization: Graph interface wireframe.*

In Figure 5, the Node property and Post Picture sections are initially empty since no node is selected. The graph initially displays random nodes and relationships, allowing the user to explore it freely by zooming in and out, dragging, and clicking on nodes to view more details. When a user clicks on a node, the Node Property section shows insights about the selected node, and the Post Picture section displays a screenshot of any media associated with it, if available. Figure 6 shows an example of the resulting graph and associated posts. The algorithmic process of visualization is demonstrated in Algorithm 3.




***Fig.6:*** *Visualization: Graph interface.*

Algorithm 3 outlines the visualization process in React. It primarily uses three functions that execute commands once the necessary libraries are imported and the graph has been initialized. The fetchData() function begins by connecting to Neo4j using the Neo4j driver, querying all nodes and relationships, and storing the retrieved information in local

variables. The renderNodeDetails() function displays the properties of a selected node when it is clicked. The searchNode() function searches the graph and displays the results in the interface. Additional details about this search function will be covered in the subsequent modules.

**Algorithm 3** Graph Visualization Process

**Require:** Neo4j connection parameters, optional search query
**Ensure:** Interactive graph visualization rendered on the interface
Import visualization libraries and initialize the graph component
**function** FETCHDATA
    Open a Neo4j session
    Query all nodes and relationships
    Transform retrieved records into visualization format
    Update graph state with nodes and edges
    Close the Neo4j session
**end function**
**function** RENDERNODEDETAILS(*selectedNode*)
    **if** $selectedNode = \emptyset$ **then**
        **return** "No node selected"
    **end if**
    Retrieve node metadata
    Display node properties and associated media, if available
**end function**
**function** SEARCHNODE(*inputQuery*)
    **if** $inputQuery \neq \emptyset$ **then**
        Execute search query over Neo4j
        Update visualization with matched nodes and related edges
    **end if**
**end function**
Call FETCHDATA
Render the search bar, detail panel, media panel, and graph canvas

*E. Search Module*

This module is designed to return not only the matched result nodes but also their associated relationships and neighboring entities, thereby providing rich contextual information for forensic analysis. To achieve this, the proposed full-text search mechanism leverages Neo4j's native search APIs to support Boolean query capabilities (e.g., AND, OR, NOT). The search is executed over all indexed node attributes, as defined in the Evidence Collection module (ensuring comprehensive coverage of relevant data. The initial results obtained from the full-text search are subsequently expanded by identifying their structural and semantic associations within the graph database. Specifically, the retrieved entities are linked to related nodes through predefined relationships, enabling a context-aware exploration of the underlying social media data. Algorithm 4 presents a parallel multi-keyword search strategy over graph data. The input keyword set (searchTermSet) is partitioned into multiple batches and distributed across parallel threads to improve search efficiency and scalability. For each keyword, three types of index queries are executed, namely: *Post Index, Comment Index,* and *Account Index*. These queries retrieve entities that are directly or indirectly associated with the search terms. The results from these queries are aggregated at the batch level to produce intermediate summaries that capture both matched entities and their contextual relationships. Subsequently, the outputs from all parallel threads are merged to generate the final comprehensive result set. More concretely, for **post-centric queries**, the system retrieves the originating account, all accounts that reacted to the post, and all associated comments along with their authors. For **comment-centric queries**, the system identifies the commenting account, the corresponding parent post, and any reactions related to that post. For **account-centric queries**, the system gathers all posts created by the account, along with all reactions and comments made by that account.

**Algorithm 4** Parallel Multi-Keyword Search over Graph Data

**Require:** $searchTermSet = \{kw_1, kw_2, \ldots, kw_n\}$, batch size $b$
**Ensure:** $\mathcal{R}_{final}$
Partition $searchTermSet$ into batches $\mathcal{B} = \{B_1, \ldots, B_m\}$
$\mathcal{R}_{final} \leftarrow \emptyset$
**for all** $B_i \in \mathcal{B}$ **in parallel do**
    $\mathcal{R}_i \leftarrow \emptyset$
    **for all** $kw \in B_i$ **do**
        $\mathcal{R}_{post} \leftarrow$ QUERYPOSTINDEX($kw$)
        $\mathcal{R}_{comment} \leftarrow$ QUERYCOMMENTINDEX($kw$)
        $\mathcal{R}_{account} \leftarrow$ QUERYACCOUNTINDEX($kw$)
        $\mathcal{R}_{kw} \leftarrow \mathcal{R}_{post} \cup \mathcal{R}_{comment} \cup \mathcal{R}_{account}$
        $\mathcal{R}_i \leftarrow \mathcal{R}_i \cup \mathcal{R}_{kw}$
    **end for**
    Store $\mathcal{R}_i$
**end for**
$\mathcal{R}_{final} \leftarrow \bigcup_{i=1}^{m} \mathcal{R}_i$
**return** $\mathcal{R}_{final}$

By integrating and consolidating these multi-perspective query results, as illustrated in Fig. 5, the final output delivered to the Visualization module provides a semantically enriched, context-aware representation of the data. This significantly enhances the interpretability and usefulness of the retrieved evidence for forensic investigators. Finally, the outputs from all threads are merged to produce the complete search result for the input keyword set (searchTermSet). For posts, the queries include the account that posted it, every account that reacted to it, and all comments on it, along with the accounts that made them. For comments, the queries cover the account that commented, the post it replied to, and whether the account reacted to that post. For accounts, the queries include all posts, reactions, and comments made by the account. By integrating the results from these queries, as illustrated in Figure 5, the final result sent to the Visualization module will be more semantic.

## 5. EVALUATION

### A. Functionality Comparison

We compared our FEDIS system with other social network data collection schemes based on the features listed in Table 1.

***Table 1:*** *Comparison of the SN Data Collection Schemes.*

| Scheme | Social Network Types | Data Collection Method | Data Storage | Visualization | Keyword Search |
|---|---|---|---|---|---|
| [4] | Twitter | Feature Extraction with unsupervised learning | CSV file | No | No |
| [9] | Twitter, Facebook, Instagram | Ontology-based and domain-specific digital | Ontology DB | No | No |
| [31] | Flickr | Google Map API, Flask for Python, CSS, HTML5 | DB | Yes | No |
| Ours | Facebook, Twitter | Playwright API, DOM-TAO, Graph-based model | Graph DB | Yes | Yes |

As shown in Table 1, the comparative analysis of the social media data collection schemes highlights significant differences in their scope and capabilities. Scheme [4] focuses solely on Twitter using unsupervised learning for feature extraction, with data stored in CSV files but lacking visualization and search capabilities. Scheme [9] extends support to Twitter, Facebook, and Instagram, using ontology-based methods for structured data storage, but it also falls short in visualization and search features. Scheme [31] is specialized for Flickr, leveraging Google's Maps API and various web technologies, and offers visualization but no search functionality. In contrast, our scheme supports both Facebook and Twitter, employs advanced API and web data extraction methods, stores data in a graph database, and excels by providing both visualization and robust data search capabilities. This makes our scheme more versatile and powerful for comprehensive social media data analysis compared to the other schemes. Additionally, we developed multithreading programming specifically designed to support parallel data crawling and search across the target social network sites, helping streamline data collection and search processes.

### B. Parallel Processing Design and Implementation

To improve efficiency and scalability, the proposed FEDIS system employs multithreading in both the data collection and keyword search modules. This design enables concurrent execution of independent tasks while maintaining correctness and reproducibility.

**Thread Pool Configuration.**

A fixed-size thread pool is adopted to control concurrency. The number of available CPU cores bounds the number of threads T to balance performance and system stability. This prevents excessive context switching and avoids resource exhaustion during large-scale crawling and search operations.

**Work Partitioning.**

Tasks are decomposed into independent units and distributed across threads. During data collection, each thread processes a subset of URLs or target pages. In the keyword search phase, the input keyword set (searchTermSet) is partitioned into smaller batches, each assigned to a thread for parallel query execution. This ensures that the workload is evenly distributed and improves throughput.

**Deduplication and Consistency Control.**

To ensure correctness under concurrent execution, a deduplication mechanism is applied using unique identifiers (e.g., user ID, post ID, comment ID). A shared thread-safe data structure is maintained to track processed entities and relationships, ensuring that duplicate nodes or edges are not inserted into the graph database.

**Error Handling and Fault Tolerance.**

The system incorporates basic fault tolerance mechanisms. Failed tasks (e.g., network errors or incomplete responses) are automatically retried up to a predefined threshold. Exceptions within threads are isolated and logged without interrupting the overall execution. This ensures robustness and continuity of the data collection and search processes.

**Complexity and Scalability.**

Let $N$ denote the total number of tasks (e.g., URLs or keywords) and $T$ the number of threads. The parallel execution reduces the average workload per thread to approximately $O(N/T)$, subject to synchronization and I/O overhead. In practice, the system achieves near-linear speedup for moderate values of $T$, demonstrating effective scalability for large datasets.

### C. Data Collection Performance

The experiments were conducted on a Windows 10 laptop equipped with an AMD Ryzen 5 2500U processor (2.00 GHz) and 8 GB RAM. The proposed FEDIS system was implemented in JavaScript using Visual Studio Code, together with relevant supporting libraries, including web automation and graph database connectors. All experiments were conducted under controlled, reproducible conditions.

To evaluate the data collection module's performance, we primarily used a synthetic dataset that emulates Facebook-like interactions, including user profiles, posts, comments, multimedia content, and engagement behaviors (e.g., reactions and shares). The synthetic data generation process follows real-

istic structural and behavioral patterns observed in social networks, such as hierarchical post-comment relationships, temporal activity distribution, and heterogeneous user interactions. This ensures that the dataset preserves both structural fidelity and practical relevance for performance evaluation. To improve external validity, we also conducted validation using publicly available social network datasets (e.g., SNAP and Twitter/X), which were transformed and mapped to the proposed DOM-TAO representation. This enables us to verify that the FEDIS data collection and processing mechanisms generalize effectively beyond simulated environments.

The data collection process is strictly restricted to publicly accessible or simulated data and operates within authorized, controlled environments, without bypassing platform access controls or violating the terms of service. Furthermore, the system incorporates practical safeguards, including rate limiting to prevent excessive requests, controlled crawling depth, and selective data acquisition to ensure that only relevant information required for evaluation is collected. These measures ensure compliance with ethical standards while maintaining realistic experimental conditions.

The experiments were designed to assess data collection performance in terms of speed and scalability. Table II demonstrates the data-collection speed and scalability of our approach across varying data sizes.

***Table 2:*** *Scalability (Performance Metrics with Increasing Data Size).*

| Data size (Nodes) | Data size (Relationships) | Data Collection Time (sec) | Data Collection Speed (node/sec) | Data Collection Speed (relationship /sec) | Data Collection Speed (node+ relationship /sec) |
|---|---|---|---|---|---|
| 104 | 400 | 131.191 | 0.793 | 3.049 | 3.842 |
| 504 | 2000 | 474.977 | 1.061 | 4.210 | 5.272 |
| 1015 | 4032 | 1020.545 | 0.995 | 3.950 | 4.945 |

As illustrated in Table II, the results were derived from scraping various datasets: one set of posts with 100 comments each, five sets of posts with 100 comments each, and ten sets of posts with 100 comments each. Each experiment specifies the graph size, denoted by the number of nodes and relationships, enabling the calculation of data-collection speeds using three distinct metrics: by node, by relationship, and by both node and relationship combined. Notably, the findings indicate that the scraping speed remains consistent irrespective of the number of posts subjected to scraping.

### *D. Data Collection Accuracy*

To evaluate the accuracy and completeness of the proposed FEDIS system, we compare it with a baseline DOM-only approach. The goal of this experiment is to assess how effectively each model captures entities and their associated relationships from social media data.

**Relationship Definition**

In this evaluation, a *relationship* is defined as a valid interaction or association between entities, represented as a directed edge in the data model. Specifically, we consider the following relationship types: *POSTED (user-to-post), REACTED (user-to-post), SHARED (post-to-post), COMMENTED (user-to-comment),* and *REPLIED (comment-to-post).* Each occurrence of these interactions is counted as one relationship instance.

To ensure a transparent evaluation, we construct a ground-truth dataset during data collection. All extracted entities and interactions are first recorded in structured logs before being stored in the database. This reference dataset captures the complete set of nodes and relationships present in the simulated Facebook feeds. A subset of the data is manually validated, while the entire dataset is programmatically verified for consistency (e.g., matching identifiers and timestamps). The total number of valid relationships in this reference dataset is treated as the ground truth.

**Evaluation Metric**

The completeness of each model is computed as:

$$\text{Completeness} = \frac{\text{Number of correctly retrieved relationships}}{\text{Total relationships in ground truth}}$$

Table III presents the comparison of data completeness between the DOM model and the proposed DOM-TAO model. Both approaches achieve 100% completeness in terms of node extraction. However, a significant difference is observed in relationship extraction.

***Table 3:*** *Comparison of the Completeness of Both Models.*

| Scraper | Nodes | Relationships |
|---|---|---|
| DOM | 100% | 49.6% |
| DOM-TAO (Our FEDIS) | 100% | 100% |

As shown in Table III, the DOM model retrieves only 2,000 of the 4,032 relationships in the ground truth, achieving 49.6% completeness. In contrast, the proposed DOM-TAO model successfully captures all relationships, achieving 100% completeness.

This discrepancy arises from the inherent limitations of the traditional DOM-based approach, which primarily focuses on hierarchical structures (e.g., user-to-post, post-to-comment, and post-to-multimedia) and does not explicitly model interactions between entities. As a result, relationships such as user reactions, shares, and cross-entity interactions are either partially captured or omitted. In contrast, the DOM-TAO model integrates hierarchical (DOM)

and relational (TAO) representations, enabling explicit modeling of interactions among users, posts, and comments. For example, it can accurately capture which users react to or share a post, as well as the association between comments and their authors. Furthermore, multimedia elements are efficiently represented as node properties, reducing structural redundancy while preserving semantic relationships.

These results demonstrate that the proposed DOM-TAO model provides a more comprehensive and semantically rich representation of social media data, significantly improving relationship completeness compared to the DOM-only approach.

*E. Keyword Search Performance Evaluation and Baseline Configuration*

To ensure a fair and reproducible comparison, we evaluate the proposed graph-based approach (Neo4j) against a relational baseline (MySQL) under equivalent query workloads and indexing strategies.

- **Index Configuration**

For Neo4j, we define full-text and property indexes on key node attributes (e.g., User.id, Post.content, Comment.content) and relationship properties. For MySQL, we construct equivalent B-tree and full-text indexes over normalized relational tables (e.g., Users, Posts, Comments, and Reactions). All indexes are created before query execution to ensure consistent conditions across systems.

- **Query Equivalence.**

We design semantically equivalent queries for both platforms. In Neo4j, Cypher queries traverse graph relationships to retrieve associated entities, whereas in MySQL, equivalent results are obtained using multi-table joins. Care is taken to ensure that both queries return identical result sets, including users, posts, comments, and their relationships.

- **Caching Conditions.**

We evaluate performance under two scenarios: (i) cold cache, where system caches are cleared before execution, and (ii) warm cache, where queries are executed repeatedly to benefit from caching mechanisms. This distinction allows us to assess both worst-case and practical performance.

- **Experimental Protocol.**

Each experiment is repeated multiple times (e.g., 20 runs) under identical conditions. The reported results correspond to the mean execution time, along with the standard deviation to capture variability. Where appropriate, we also report confidence intervals to provide statistical reliability.

- **Evaluation Metrics.**

We measure (i) query latency, (ii) throughput (queries per second), and (iii) scalability with respect to dataset size and number of keywords. These metrics provide a comprehensive view of system performance.

- **Additional Baselines.**

Beyond MySQL, we discuss the applicability of alternative baselines, including optimized relational configurations and graph-aware indexing strategies, to contextualize the advantages of the proposed approach. Overall, this evaluation setup ensures that the comparison between Neo4j and MySQL is conducted under controlled, equivalent, and statistically rigorous conditions.

*Keyword Search Performance*

Based on the above configuration, we compare the search performance of Neo4j and MySQL in terms of query execution efficiency. The dataset used for this evaluation contains 510,045 nodes and 2,900,600 relationships, generated using the Faker library to simulate large-scale social media interactions. To ensure comparability, the same dataset is deployed on both Neo4j and MySQL, and semantically equivalent queries are executed using their respective query languages. In particular, keyword-matching semantics are aligned by configuring both systems to perform OR-based retrieval on search terms, ensuring consistent query behavior across platforms. We conduct experiments using both single-keyword and multi-keyword queries to evaluate search performance under varying query complexity. Each reported result corresponds to the average execution time over multiple runs (e.g., 10 query instances per setting), with variability captured through the statistical measures described earlier.

The queries retrieve not only matching entities (e.g., usernames, post content, and comment text) but also their associated relationships and interacting profiles. This enables a comprehensive forensic analysis by identifying both direct matches and contextual connections among entities. Figure 7 compares search performance between Neo4j and MySQL. The results demonstrate that the proposed graph-based approach consistently outperforms the relational baseline, particularly as the number of keywords increases. This improvement is attributed to Neo4j's efficient graph traversal and relationship-centric query processing, whereas MySQL incurs additional overhead from complex join operations across relational tables.

As shown in Fig. 7, the proposed search algorithm implemented in Neo4j consistently outperforms the MySQL-based approach across all query settings, from single-keyword to multi-keyword (up to 20 terms) searches. The performance gap becomes increasingly pronounced as the number of keywords grows. This trend highlights the superior scalability of the graph-based approach. In Neo4j, queries are

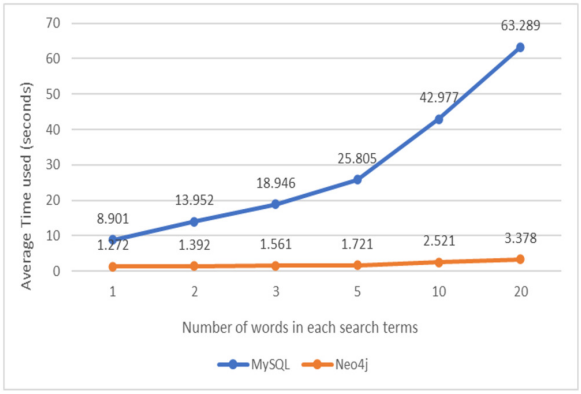



***Fig.7:*** *Search Speed Performance Comparison Between Neo4j Database and MySQL Database.*

executed through native graph traversal, which efficiently explores relationships between entities without requiring costly data reconstruction. As a result, the increase in query complexity (i.e., additional keywords) leads to only a moderate increase in execution time. In contrast, MySQL relies on multi-table join operations to reconstruct relationships across normalized tables. As the number of keywords increases, the query plan becomes more complex, involving additional join conditions, intermediate result materialization, and larger search spaces. This leads to a super-linear growth in execution time, particularly for multi-keyword queries. Furthermore, Neo4j benefits from index-assisted node lookups followed by localized traversals, which reduce unnecessary data scanning. In comparison, MySQL must repeatedly evaluate join conditions over larger candidate sets, resulting in higher computational overhead. Overall, the results demonstrate that the proposed graph-based model is better suited to relationship-intensive forensic queries, where efficiently traversing and correlating connected data is critical.

## 6. LIMITATION

Despite the effectiveness of the proposed FEDIS framework, several limitations remain. First, part of the evaluation relies on simulated social media data, which may not fully capture the complexity and variability of real-world platforms. Second, the system is currently designed and evaluated primarily for Facebook-like structures, which may limit direct applicability to other platforms with different data schemas and access constraints. Third, while multithreading improves performance, scalability in large-scale distributed environments has not been fully explored.

Additionally, several factors may affect the validity of the experimental results. The synthetic dataset, although designed to emulate realistic interaction patterns, may not perfectly reflect real-world user behavior and network dynamics. In addition, performance measurements may be influenced by system-specific factors such as hardware configuration, caching effects, and workload distribution. While efforts have been made to ensure fair comparison and reproducibility, these factors should be considered when interpreting the results.

## 7. CONCLUSION AND FUTURE WORK

In this paper, we present an efficient online social network (OSN) forensics framework, FEDIS, to address the challenges of manual, time-consuming social media investigations. Existing open-source tools often provide incomplete or poorly structured raw data, making analysis difficult, and lack support for evolving platform structures, such as recent versions of Facebook. To overcome these limitations, we proposed the DOM-TAO model, which integrates hierarchical and relational representations to enhance evidence collection and enable richer data modeling. In addition, we developed a parallel search algorithm and leveraged the Neo4j graph database to support efficient, relationship-aware querying and interactive visualization. A prototype system was implemented as a proof-of-concept and evaluated using simulated social media datasets. The experimental results demonstrate that the proposed framework achieves higher functional coverage while maintaining complete data extraction. Furthermore, the graph-based approach significantly improves search efficiency and enables more effective forensic analysis by enabling the exploration of contextual relationships.

Future work will focus on several directions. First, we plan to validate the proposed system using real-world datasets and authorized case studies to demonstrate its practical applicability further. Second, we aim to extend the DOM-TAO model to support multiple social media platforms with heterogeneous data structures. Third, we will investigate distributed and cloud-based deployment to enhance scalability for large-scale forensic investigations. Finally, we intend to integrate advanced analytics techniques, including automated anomaly detection and AI-assisted investigation, to improve the effectiveness of forensic analysis further. These directions will further enhance the robustness, scalability, and practical applicability of the proposed FEDIS framework in real-world digital forensic investigations.

## AUTHOR CONTRIBUTIONS

Conceptualization, S.F., P.K., P.A., and D.O.; methodology, S.F. and P.K. software, P.K., P.A., and D.O; validation, P.K., P.A., and D.O; formal analysis, S.F.; investigation, S.F. and P.K. data curation, S.F. and P.K.; writing—original draft preparation, S.F., P.K., P.A., and D.O.; writing—review and editing, S.F.; visualization, P.K., P.A., and D.O.; supervision, S.F.; funding acquisition, S.F. All authors

have read and agreed to the published version of the manuscript.

## References


[1] H. Wang *et al.*, "Anchor Link Prediction for Cross-Network Digital Forensics From Local and Global Perspectives," in *IEEE Transactions on Information Forensics and Security*, vol. 19, pp. 3620-3635, 2024.

[2] A. Shen, K. P. Chow and Q. Zhou, "Unsupervised Community Detection Framework for Social Network Forensics," *2024 IEEE/ACIS 24th International Conference on Computer and Information Science (ICIS)*, Shanghai, China, pp. 162-168, 2024.

[3] A. Umair, P. Nanda and X. He, "Online Social Network Information Forensics: A Survey on Use of Various Tools and Determining How Cautious Facebook Users are?," *2017 IEEE Trustcom/BigDataSE/ICESS*, Sydney, NSW, Australia, pp. 1139-1144, 2017.

[4] S. Shao, C. Tunc, A. Al-Shawi and S. Hariri, "Automated Twitter Author Clustering with Unsupervised Learning for Social Media Forensics," *2019 IEEE/ACS 16th International Conference on Computer Systems and Applications (AICCSA)*, Abu Dhabi, United Arab Emirates, pp. 1-8, 2019.

[5] H. Arshad, A. Jantan, G. K. Hoon and I. O. Abiodun, "Formal knowledge model for online social network forensics," *Computers & Security*, vol. 89, p. 101675, Feb. 2020.

[6] O. Mayer and M. C. Stamm, "Exposing Fake Images With Forensic Similarity Graphs," in *IEEE Journal of Selected Topics in Signal Processing*, vol. 14, no. 5, pp. 1049-1064, Aug. 2020.

[7] O. Elezaj, S. Y. Yayilgan and E. Kalemi, "Criminal network community detection in social media forensics," in *Proc. Int. Conf. Intell. Technol. Appl. Cham*, Switzerland: Springer, pp. 371–383, 2021.

[8] X. Zhang, Z. H. Sun, S. Karaman and S. -F. Chang, "Discovering Image Manipulation History by Pairwise Relation and Forensics Tools," in *IEEE Journal of Selected Topics in Signal Processing*, vol. 14, no. 5, pp. 1012-1023, Aug. 2020.

[9] Š. Grigaliūnas, R. Brūzgienė and A. Venčkauskas, "Ontology-Driven Digital Profiling for Identification and Linking Evidence Across Social Media Platform," in *IEEE Access*, vol. 11, pp. 111672-111691, 2023.

[10] T. Fernando, C. Fookes, S. Denman and S. Sridharan, "Detection of Fake and Fraudulent Faces via Neural Memory Networks," in *IEEE Transactions on Information Forensics and Security*, vol. 16, pp. 1973-1988, 2021.

[11] S. Thorpe and M. Bernard, "Graph mining for forensic databases," *SoutheastCon 2017*, Concord, NC, USA, pp. 1-10, 2017.

[12] "Full-Text Search Index," Neo4j, [Online]. Available: `https://neo4j.com/docs/cypher-manual/current/indexes-for-full-text-search/`. [Accessed: Jan. 15, 2024].

[13] N. Bronson *et al.*, "TAO: Facebook's distributed data store for the social graph," in *Proceedings of the 2013 USENIX Conference on Annual Technical Conference (USENIX ATC'13)*. USENIX Association, USA, 49–60, 2013.

[14] I. Mysiuk and R. Shuvar, "Automating Web Scraping of User Comments for Sentiment Analysis in Social Networks," *2023 IEEE 13th International Conference on Electronics and Information Technologies (ELIT)*, Lviv, Ukraine, pp. 77-81, 2023.

[15] "What Is an Actor?," Apify, [Online]. Available: `https://docs.apify.com/platform/actors`. [Accessed: Jan. 15, 2024].

[16] L. Citra Dewi and A. Chandra, "Social Media Web Scraping using Social Media Developers API and Regex," in *Proc. of 4th International Conference on Computer Science and Computational Intelligence 2019 (ICCSCI)*, pp. 444-449, Sep. 2019.

[17] A. Hernandez-Suarez, G. Sanchez-Perez, K. Toscano-Medina, R. Toscano-Medina, V. Martinez-Hernandez, J. Olivares-Mercado, H. Pérez-Meana and V. Sanchez, "Can Twitter API be bypassed? A new methodology for collecting chronological information without restrictions," in *Proc. SoMeT*, 2018, pp. 453–462.

[18] J. You, J. Lee and H. -Y. Kwon, "A Complete and Fast Scraping Method for Collecting Tweets," *2021 IEEE International Conference on Big Data and Smart Computing (BigComp)*, Jeju Island, Korea (South), pp. 24-27, 2021.

[19] B. Kusumasari and N. P. A. Prabowo, "Scraping social media data for disaster communication: How the pattern of Twitter users affects disasters in Asia and the pacific," *Natural Hazards*, vol. 103, no. 3, pp. 3415–3435, Sep. 2020.

[20] I. Dongo, Y., Cadinale, A. Aguilera, F. Martínez, Y. Quintero and S. Barrios, "Web scraping versus Twitter API: A comparison for a credibility analysis," in *Proceedings of the 22nd International Conference on Information Integration and Web-based Applications & Services*, pp. 1–11, Nov. 2020.

[21] Y. Cardinale, I. Dongo, G. Robayo, D. Cabeza, A. Aguilera and S. Medina, "T-CREo: A Twitter Credibility Analysis Framework," in *IEEE Access*, vol. 9, pp. 32498-32516, 2021.

[22] Apify, "Apify," [Online]. Available: `https://apify.com`. [Accessed: Jan. 15, 2024].

[23] M. Mitchell, S. Wu, A. Zaldivar, P. Barnes, L.

Vasserman, B. Hutchinson, E. Spitzer, I. D. Raji and T. Gebru, "Model cards for model reporting," in *Proc. Conf. Fairness, Accountability, Transparency*, pp. 220–229, Jan. 2019.

[24] H. Arshad, A. Jantan and E. Omolara, "Evidence collection and forensics on social networks: Research challenges and directions," *Digital Investigation*, vol. 28, pp. 126–138, Mar. 2019.

[25] P. V. Srimukh and S. Shridevi, "Ontology-based crime investigation process," in *Advances in Smart Grid Technology*, P. Siano and K. Jamuna, Eds. Singapore: Springer, vol. 687, pp. 497–509, 2020.

[26] M. Woodward, "Twitter User Statistics 2024: What Happened After 'X' Rebranding," Search Logistics, [Online]. Available: `https://www.searchlogistics.com/learn/statistics/twitter-user-statistics/`.[Accessed: Jul. 1, 2024].

[27] J. Robie, "What Is the Document Object Model?," W3C, [Online]. Available: `https://www.w3.org/TR/WD-DOM/introduction.html`. [Accessed: Jul. 1, 2024].

[28] Y. Wu, N. Cao, D. Gotz, Y. -P. Tan and D. A. Keim, "A Survey on Visual Analytics of Social Media Data," in *IEEE Transactions on Multimedia*, vol. 18, no. 11, pp. 2135-2148, Nov. 2016.

[29] S. Chen, L. Lin, and X. Yuan, "Social media visual analytics," *Computer Graphics Forum*, vol. 36, no. 3, pp. 563–587, Jun. 2017.

[30] L. Anthony, "Introducing Fireant: A Freeware, Multiplatform Social Media Data-Analysis Tool," in *IEEE Transactions on Professional Communication*, vol. 61, no. 4, pp. 428-442, Dec. 2018.

[31] T. Isenberg, Z. Salazar, R. Blanco and C. Plaisant, "Do You Believe Your (Social Media) Data? A Personal Story on Location Data Biases, Errors, and Plausibility as Well as Their Visualization," in *IEEE Transactions on Visualization and Computer Graphics*, vol. 28, no. 9, pp. 3277-3291, 1 Sept. 2022.

[32] X. Bao and S. Jin, "Knowledge Graph Analysis of Social Media in the Context of Big Data," *2024 9th International Conference on Cloud Computing and Big Data Analytics (ICCCBDA)*, Chengdu, China, pp. 295-300, 2024.

[33] S. Lee, A. A. H. Mujammami and K. Kim, "Leveraging Social Networks for Cyber Threat Intelligence: Analyzing Attack Trends and TTPs in the Arab World," in *IEEE Access*, vol. 13, pp. 5679-5693, 2025.

[34] S. Jiang, Y. Qiu, X. Mo, R. Tang and W. Wang, "An Effective Node Injection Approach for Attacking Social Network Alignment," in *IEEE Transactions on Information Forensics and Security*, vol. 20, pp. 589-604, 2025.



**Somchart Fugkeaw** (Member, IEEE) received the B.B.A. degree in Management Information Systems from Thammasat University, Bangkok, Thailand, the M.Sc. degree in Computer Science from Mahidol University, Thailand, and the Ph.D. degree in Electrical Engineering and Information Systems from The University of Tokyo, Japan, in 2017.He is currently an Associate Professor with the Sirindhorn International Institute of Technology, Thammasat University. His research interests include information security, access control, cloud security, blockchain technologies, big data analytics, and high-performance computing. Dr. Fugkeaw has authored and co-authored more than 100 publications in leading refereed journals and international conferences. His research interests include information security, cloud computing, database systems, and blockchain.



**Pratuck Khubpatiwitthayakul** completed a bachelor's degree in computer engineering from Sirindhorn International Institute of Technology, Thammasat University. His research interests include Network Security, programming language, and Cloud computing.



**Pongsapon Ateetanan** completed a bachelor's degree in computer engineering from Sirindhorn International Institute of Technology, Thammasat University. His research interests include Information security, Artificial Intelligence, and Cloud computing.



**Diyorbek Bakhtiyor Ugli Okyulov** completed a bachelor's degree in computer engineering from Sirindhorn International Institute of Technology, Thammasat University. His research interests include Network Security, Digital Forensics, and Cloud computing.