Real Time Arabic Communities Attack Detection on Online Social Networks

Jalal S Alowibdi
Department of Computer Science and Artificial Intelligence, College of Computer Science and Engineering, University of Jeddah, Jeddah, Saudi Arabia
jalowibdi@uj.edu.sa

Abstract
The dynamic nature of Online Social Networks (OSNs), especially on platforms like Twitter, presents challenges in identifying and responding to community attacks, particularly within Arabic content. The proposed integrated system addresses these challenges by achieving 91% accuracy in detecting real-time community event attacks while efficiently managing computational costs. This is accomplished through the use of specialized integrated approach clustering to detect both major and minor attacks. Additionally, the system leverages clustering algorithms, temporal modules, and social network graphs to identify events, map communities, and analyze online dynamics. An extensive parameter sensitivity analysis was conducted to fine-tune the algorithm, and the system's effectiveness was validated using a benchmark dataset, demonstrating substantial improvements in event detection.

Keywords
Real-time Event Detection, Community Attacks, Inverted Indices, Incremental Clustering, Arabic-speaking Community

I. INTRODUCTION

Online Social Networks (OSNs) like Twitter have become integral parts of our daily lives, facilitating the rapid dissemination of information and the formation of online communities. These platforms allow users to share their thoughts, opinions, and experiences in real-time, creating a dynamic and constantly evolving digital landscape. The widespread use of OSNs has enabled users to engage in real-time discussions, share information, and form communities. However, while OSNs provide new opportunities for social interaction, they also introduce challenges, such as the spread of misinformation and the potential for coordinated attacks on individuals or groups

Twitter, in particular, stands out due to its immediacy and widespread use, making it a prime source for real-time data analysis. However, the rapid and dynamic nature of Twitter content presents unique challenges for monitoring and detecting malicious activities, such as community attacks, in real-time. Community attacks on OSNs can take various forms, including misinformation campaigns, coordinated harassment, and spam attacks, all of which can have significant societal impacts. The real-time detection of these attacks is crucial to mitigate their effects promptly and prevent further harm.

This study focuses on the Arabic-speaking community on OSNs, particularly Twitter, aiming to develop a robust system for detecting community attacks. Given the complexity of the task, the research explores the challenges of event detection in Twitter, particularly within Arabic content. The methodologies employed in our system are designed to address these challenges, and the experimental results highlight the system's robustness and efficiency in detecting and responding to community attacks.

This work contributes to the ongoing efforts to enhance real-time monitoring and intervention capabilities in OSNs, providing a valuable tool for mitigating the adverse effects of community attacks. By focusing on the Arabic-speaking community, this study also addresses the need for more inclusive tools that cater to diverse linguistic and cultural contexts.

The remainder of this article is organized as follows: Section II discusses related works, Section III lists the motivations and challenges, Section IV outlines the proposed methodology, and Section V presents the results and discussion.

II. RELATED WORKS

Analyzing and detecting events on OSNs, including Twitter, has attracted considerable attention from researchers [1—10]. The platform's capability to capture real-time public reactions and information has made event detection on OSNs, particularly Twitter, a significant area of study. Here, we are going to review various works that have explored event detection, especially in the context of both Arabic and English languages

Early research in event detection primarily focused on the English language, leveraging the large volume of available data and well-established natural

Manuscript received August 5, 2024
Manuscript revised August 20, 2024
https://doi.org/10.22937/IJCSNS.2024.24.8.7

language processing (NLP) tools. Techniques such as detecting spikes in keyword usage and conducting sentiment analysis have been widely employed to identify significant events from OSNs, including Twitter data. Sakaki et al. were pioneers in real-time event detection by utilizing Twitter as a social sensor for natural disasters, with a particular emphasis on earthquake detection [11]. Their study illustrated how Twitter's rapid information dissemination could be harnessed to detect events as they occur. Similarly, Petrovic et al. introduced the concept of First Story Detection (FSD), using a streaming model to detect breaking news on Twitter [12]. Their approach was centered on identifying the first mention of an event before it gained widespread attention. In recent years, advanced methods like topic modeling and wavelet analysis have been explored for event detection where Cordeiro combined wavelet analysis with topic inference summarization to improve event detection on Twitter by capturing the temporal dynamics of conversations [13].

On the other hand, research on event detection in Arabic is a more recent development, reflecting the unique challenges of working with a morphologically rich and less-resourced language. However, the importance of detecting events in Arabic-speaking regions, particularly during the well known events, has led to significant advancements in the field. Alharbi et al. introduced Kawarith, a multi-dialect Arabic Twitter corpus designed for crisis events, which includes over a million tweets from 22 crises that occurred between 2018 and 2020 [14]. Their work highlights the importance of developing resources for low-resource languages like Arabic to aid crisis-affected communities with real-time data. Additionally, Alsaedi et al. focused on Arabic event detection in social media, presenting methods to identify significant events in Arabic-speaking regions using Twitter data [15]. Their research highlights the challenges of processing Arabic text in social media and the necessity for specialized tools to effectively detect events. Moreover, Alkouz et al. concentrated on traffic event detection in Arabic Twitter, integrating data from multiple social media streams [16]. Their study emphasized the need for cross-platform event detection strategies that take both language and regional context into account.

Detecting events in Arabic presents challenges due to the language's complex morphology, diverse dialects, and lack of standardization. As the field of event detection on Twitter and social media continues to evolve rapidly, this research aims to further explore Arabic event detection, with a particular focus on attacks on communities within this linguistic context.

III. MOTIVATIONS AND CHALLENGES

Event detection in OSNs is the process of identifying and tracking significant occurrences or trends as they happen. This is crucial for understanding the impact of these events on public opinion and for providing timely responses to emerging situations. Traditional methods, which often rely on analyzing keywords and hashtags, face challenges due to the vast volume of data and the rapid spread of information on platforms like Twitter. To enhance the accuracy of event detection, it is essential to incorporate temporal modules that determine the start and end times of events. These modules analyze the timing of posts and interactions, helping to identify when an event gains momentum and when it begins to lose public interest. Temporal analysis is key to understanding the lifecycle of events, distinguishing between short-lived trends and more enduring occurrences.

Community detection plays a significant role in this process by identifying groups of users who either support or oppose an event. By mapping out these communities, we can gain insights into the diverse perspectives and motivations driving the conversation. Clustering algorithms are often used to group users based on their interactions and shared content, providing a clearer picture of the broader social context surrounding events. Communities can greatly influence how events are perceived: supportive groups may amplify positive aspects of an event, while opposing groups may work to discredit or disrupt it. Analyzing these community dynamics is crucial for understanding how events evolve and how public opinion is shaped.

To fully define an event, four key questions need to be addressed: What is the event? Who is involved? Where is it happening? When is it taking place? Answering these questions allows for a comprehensive understanding of the event and its context, enabling a more accurate assessment of its significance and potential impact. This study hypothesizes that real-time detection of community attacks on OSNs can provide valuable insights into the dynamics of events and the communities that drive them. By combining clustering algorithms, temporal modules, and social network graphs, we aim to create a robust framework for identifying and understanding online events.

However, several challenges must be overcome to improve the detection and analysis of events on OSNs. These challenges include the rapid pace of information dissemination, the massive volume of data generated, the complexity of social interactions, and the potential for misinformation and manipulation. Addressing these challenges is critical for ensuring that events are accurately identified and understood, regardless of their positive or negative implications.

One of the main challenges in real-time event detection on Twitter is the high computational cost. Traditional methods often struggle to keep up with the volume and velocity of data generated on such platforms, leading to delays and reduced accuracy in identifying critical events. To tackle this issue, we propose an event detection system that incorporates specialized inverted indices and an incremental clustering approach. This system is designed to offer a low computational cost solution capable of detecting both major and minor newsworthy events in real time, thus improving both the speed and accuracy of event detection on OSNs.

IV. Proposed Methodology

To comprehensively detect event communities on OSNs, we first need to employ a multifaceted approach incorporating various methods and techniques to ensure robust and accurate detection. Yet, we have detailed each component of our methodology.

Dataset Collection

Data collection is the first step, where we have gathered a large corpus of tweets related to specific events. This data is sourced from Twitter's API, using keywords, hashtags, and user handles to filter relevant tweets. Yet, we produced novel ideas in collecting the dataset since it is hard to build up the study without a relevant dataset. Thus, the process of collecting a dataset for events involved several methodical steps which were designed to ensure that the datasets were both relevant and comprehensive for analyzing user behavior and topic dynamics related to specific news events. The steps are outlined below:

1. Selecting the Organization

The first step in the dataset collection process was to select a specific organization as the study's focus. This organization needed to be one with a significant presence in public discourse, particularly on OSNs platforms like Twitter, where discussions are lively and varied. The choice of the organization was critical, as it would guide the subsequent steps in identifying and collecting relevant data. The organization selected was one that regularly generates newsworthy events, ensuring that there would be a sufficient volume of related discussions on Twitter.

2. Identifying Relevant News Events

Once the organization was selected, the next step involved identifying different news events associated with it. This required monitoring various news outlets and online sources to track announcements, controversies, policy changes, or other significant developments related to the organization. The goal was to find news events that sparked discussions among Twitter users, as these would form the basis of the dataset. Each news event was documented, along with key details such as the date, the nature of the event, and any specific hashtags or keywords that were being used in the discussions.

3. Collecting Twitter Data

With the news events identified, we moved to Twitter to collect relevant dataset. This involved searching for tweets that were directly related to the news events associated with the organization. Using Twitter's search functionality and APIs, we filtered tweets based on the identified keywords, hashtags, and user mentions that were linked to the news. The collected tweets were from various users, capturing a range of opinions, reactions, and discussions around the event. The time frame for dataset collection was typically aligned with the period when the news event was most actively discussed on the OSNs platform.

4. Sentiment Analysis

After gathering the dataset, the next step was to perform sentiment analysis to determine the tone of the discussions. Each tweet was analyzed to extract whether the sentiment expressed was positive, negative, or neutral. This analysis was crucial for understanding the general mood of the public towards the event and the organization. Sentiment analysis involved both automated tools and manual validation to ensure accuracy, particularly in identifying the subtleties of language use on OSNs.

5. Grouping and Event Categorization

Once the sentiment analysis was complete, the tweets were grouped into events based on the specific news they discussed. This involved categorizing the dataset into distinct events, ensuring that each group represented a coherent discussion about a particular news item. Grouping tweets by event allowed for a more structured analysis of how different news stories influenced public opinion and how users engaged with each topic.

6. Analyzing User-Topic Relations

With the events grouped and sentiment analyzed, the final step was to analyze the relationships between users and the topics they discussed. This involved mapping out which users were most active in discussing each event and identifying any patterns in their behavior. For instance, some users might consistently discuss news related to the organization, while others might only engage with specific types of events. By comparing user-topic relations across different events, it was possible to gain insights into the dynamics of online discourse related to the organization.

7. Repeating the Process for Multiple Events

To ensure a robust analysis, this process was repeated for multiple events associated with the same organization. At least three distinct events were selected, and the entire process, from news identification to user-topic relation analysis, was conducted for each one. This allowed for a comparative analysis, highlighting how different types of news might influence user behavior and sentiment on Twitter.

The systematic approach to data collection, encompassing organization selection, event identification, sentiment analysis, and user-topic relation mapping, provided a comprehensive dataset for analyzing how news events impact discussions on Twitter. By repeating this process across multiple events, the study generated valuable insights into the dynamics of online discourse and the role of different users in shaping public opinion.

Preprocessing the dataset

Preprocessing is a critical step in preparing the collected dataset for effective event detection. The primary goal of preprocessing is to clean and transform the raw data into a structured format that can be analyzed efficiently. Here, we are going to outline the various preprocessing techniques used, incorporating mathematical approaches where applicable. Thus, before diving into preprocessing, it is important to understand the nature of the data collected. The dataset consists of tweets gathered using Twitter's API, filtered by specific keywords, hashtags, and user handles related to the chosen organization and its associated events. This raw dataset contains a vast amount of information, including text content, user metadata, timestamps, and other relevant features. The preprocessing steps are designed to extract meaningful information from this raw data, focusing on preparing the text content for further analysis. Preprocessing steps include:

1. Tokenization

The first step in preprocessing is tokenization, which involves breaking down each tweet into individual words or tokens. Tokenization helps in converting the text into a format that can be easily analyzed by algorithms. Mathematically, if a tweet is represented as a string S tokenization can be defined as the process of mapping S into a set of tokens T as follows:

$$T = \{t_1, t_2, t_3, \dots, t_n\} \quad (1)$$

where t_i represents a single word or token derived from the tweet. Tokenization also involves handling punctuation and special characters, by removing them or treating them as separate tokens, depending on the requirements of the analysis.

2. Stopword Removal

Once the text is tokenized, the next step is stopword removal. Stopwords are frequently-used words that do not contribute significantly to the meaning of the text in the context of event detection. Let T be the set of tokens after tokenization, and let S_w be the set of stopwords. The process of stopword removal can be mathematically represented as:

$$T' = T \setminus S_w \quad (2)$$

where T' is the set of tokens after stopword removal and the set of stopwords is S_w . Then, the result of stopword removal would be stored in T' . This step significantly reduces the dimensionality of the dataset, making the subsequent analysis more efficient.

3. Stemming and Lemmatization

The next step in preprocessing is stemming which is a process that chops off the ends of words to arrive at the root form and lemmatization which is a more sophisticated process that takes into account the morphological structure of words. stemming and lemmatization involves reducing words to their base or root forms. This step is crucial for handling variations in word forms, ensuring that different forms of the same word are treated as a single entity. Let T'be the set of tokens after stopword removal. The process of stemming and lemmatization can be represented as a function f that applied to each token as follows:

$L = \{f(t'_1), f(t'_2), f(t'_3), \ldots, f(t'_m)\}$
(3)

where L is the set of lemmatized tokens.

4. Vectorization for Event Detection

The cleaned and transformed text data is typically converted into a numerical format that can be fed into machine learning models for event detection. One common approach is to use the Term Frequency (TF) and Inverse Document Frequency (IDF) method, which represents each document (tweet) as a vector in a high-dimensional space. The $(TF - IDF)$ value for a term t in a document d within a corpus D is calculated as follows:

$TF - IDF(t, d) = TF(t, d) \times IDF(t)$
(4)

where: $TF(t, d)$ is the term frequency, representing how frequently a term t appears in a document d and $IDF(t)$ is the inverse document frequency, calculated as follow:

$IDF(t) = log\, log\, \left(\frac{|D|}{|\{d \epsilon D : t \epsilon d\}|}\right)$
(5)

This approach emphasizes words important within a specific document while downplaying those common across all documents, making it effective for event detection.

Feature Extraction

Effective event detection on OSNs including Twitter relies heavily on a well-structured approach to feature extraction. The raw data collected from Twitter, typically in the form of tweets, must be transformed into meaningful features that can be used for analysis. An integrated approach to feature extraction combines textual, temporal, and network-based features to provide a comprehensive understanding of events as they unfold. Here, we will explore this combined approach, detailing the foundations to illustrate how each feature can be extracted and utilized.

1. Textual Feature Extraction

Textual features are the cornerstone of event detection, as they directly relate to the content of the tweets. Here we are going to list all the textual features used. First, Keyword Frequency and TF-IDF, which measure the importance of specific terms in the dataset. Keyword frequency can be calculated using the term frequency TF, which measures how often a term appears in a document relative to the total number of terms in that document as follows:

$TF(t, d) = \frac{f_{t,d}}{\sum_{t' \in d} f_{t',d}}$
(6)

where $f_{t,d}$ is the frequency of term t in document d. To account for the importance of a term across multiple documents, we use Inverse Document Frequency (IDF) as follows:

$IDF(t) = log\, log\, \left(\frac{N}{|\{d \in D : t \in d\}|}\right)$
(7)

where N is the total number of documents and $|\{d \in D : t \in d\}|$ is the number of documents containing the term t. The combined TF-IDF score is calculated as follows:

$TF - IDF(t, d) = TF(t, d) \times IDF(t)$
(8)

Second, we applied Sentiment analysis that assigns a polarity score to each tweet, categorizing it as positive, negative, or neutral. This helps in understanding the public sentiment surrounding an event. The process can be represented as a function $f(x)$ that assigns a sentiment y to a tweet x as follows:

$y = f(x)$
(9)

After that, Named Entity Recognition (NER) identifies and categorizes entities in tweets, such as names of people, organizations, or locations, which are often central to event-related discussions. Given a sequence of words $x = (x_1, x_2, x_3, \ldots, x_n)$, the goal is to assign a sequence of labels $y = (y_1, y_2, y_3, \ldots, y_n)$, where each y_i represents an entity type as follows:

$\hat{u} = arg\, arg\quad P(x)$
(10)

Finally, we applied Topic modeling, such as Latent Dirichlet Allocation (LDA), which uncovers hidden thematic structures within the dataset, grouping tweets into topics that represent underlying themes of discussion. The process involves choosing a distribution over topics θ_d for each document and generating words based on this distribution.

2. Temporal Feature Extraction

Temporal features are crucial for understanding when events occur and how they evolve. The posting time of tweets can reveal the dynamics of event-related discussions, while burst detection can highlight sudden spikes in activity that often correspond to significant developments. The number of tweets $f(\tau)$ posted at time τ can be analyzed to detect activity over time, and anomalies can be detected using the z-score as follows:

$$z(\tau) = \frac{f(\tau) - \mu}{\sigma}$$
(11)

where μ is the mean frequency and σ is the standard deviation. Then, we used the event lifecycle modeling of an event that involves tracking its four phases: initialization, growth, peak, and decline. Yet, the lifecycle function $L(\tau)$

$$L(\tau) = \begin{cases} \text{Initialization Phase} & \text{if } \tau < \tau_s \\ \text{Growth Phase} & \text{if } \tau_s \leq \tau < \tau_p \\ \text{Peak Phase} & \text{if } \tau = \tau_p \\ \text{Decline Phase} & \text{if } \tau_p < \tau \leq \tau_e \end{cases} \quad (12)$$

represents these phases based on tweet activity as follows:

3. Network Feature Extraction

Network features analyze the interactions between users, revealing influential users and the structure of communities involved in an event. A user interaction graph $G = (V, E)$ is constructed, where nodes V represent users, and edges E represent interactions such as retweets, mentions, and replies. Therefore, a user interaction graph here maps the relationships between users on Twitter based on interactions such as retweets, mentions, and replies. This network-based approach highlights how information spreads across the platform and identifies clusters of users who are actively engaging with each other. By analyzing the structure of these graphs, we can uncover the key players and the flow of information within the network. Then, we applied the Centrality measures to help identify influential users within the network. The Degree Centrality calculates the number of connections a user has, indicating their direct influence as follows:

$$C_D(u) = deg\, deg\, (u)$$
(13)

where $deg\, deg\, (u)$ is the number of connections user u has. Also, with the degree centrality, there are the Betweenness Centrality that assesses how often a user appears on the shortest paths between other users, highlighting their role as a bridge within the network, and the Closeness Centrality which measures how close a user is to all others, reflecting their overall accessibility and influence within the network. These metrics collectively help pinpoint key users who drive the flow of information and interactions during an event. Therefore, community detection identifies clusters of users who are more densely connected with each other than with the rest of the network, representing groups with shared interests and discussions. Also, Modularity Q is often used to measure the strength of division of a network into clusters as follows:

$$Q = \frac{1}{2m}\sum_{i,j} \left[A_{i,j} - \frac{k_i k_j}{2m}\right] \delta(C_i, C_j)$$
(14)

where $A_{i,j}$is the adjacency matrix and $\delta(C_i, C_j)$ is the Kronecker delta function. Then, the influence score evaluates the impact of a user based on their centrality in the network and their level of engagement (e.g., retweets, mentions and replies). The influence score $I(u)$ for user u is a weighted sum of their degree centrality, retweets, replies and mentions as follows:

$$I(u) = \alpha \times C_D(u) + \beta \times R(u) + \gamma \times M(u) + \delta \times P(u) \quad (15)$$

where $R(u)$ is the Retweets, M($M(u)$ is the Mentions and $P(u)$. Yet, the integrated approach to feature extraction combines textual, temporal, and network-based features to provide a comprehensive analysis framework for event detection on Twitter. By using these features together, we can gain deep insights into the dynamics of online discussions, identify key influencers, and understand how events evolve over time. The mathematical formulations provided offer a foundation for implementing these techniques in real-world applications, ensuring robust and effective event detection.

Clustering Algorithms Methods

Clustering algorithms play a pivotal role in the analysis of social networks, particularly in grouping users and detecting communities within large datasets. In the context of event detection on OSNs platforms like Twitter, these algorithms are essential for identifying clusters of users who share similar interests, opinions, or behaviors, which can significantly enhance the understanding of how events propagate and evolve over time. In this section, we are going to provide an in-depth look at the various clustering techniques utilized in our study, explaining their importance, methodology, and applicability to community detection in OSNs.

1. K-Means Clustering

K-means clustering is one of the most widely used clustering algorithms due to its simplicity and efficiency. The algorithm partitions a dataset into K distinct clusters, where each data point belongs to the cluster with the nearest mean. In the context of Twitter data, each user or tweet can be represented as a feature vector, and K-means is applied to group these vectors into clusters that represent different communities and topics of discussion. The K-means algorithm aims to minimize the within-cluster variance, which is defined as the sum of squared distances between each point and its corresponding cluster centroid. Mathematically, the objective function to minimize is as follow:

$$\sum_{i=1}^{K} \sum_{x_j \in C_i} \left| x_j - \mu_i \right|^2 \quad (16)$$

where K is the number of clusters, C_i represents the i -th cluster, x_j is the data point for the user and the tweet vector and μ_i is the centroid of the i -th cluster. Therefore, in our dataset, the features used to represent users and tweets include the frequency of specific keywords, hashtags, sentiment scores, and user interaction patterns. K-means is effective for identifying well-defined groups within this high-dimensional space, such as communities discussing a particular event or supporting a specific viewpoint. However, one limitation is that the number of clusters K must be predefined, which may not always be straightforward in complex datasets.

2. DBSCAN

Density-Based Spatial Clustering of Applications with Noise (DBSCAN) is a density-based clustering algorithm that identifies clusters based on the density of data points. Unlike K-means, DBSCAN does not require the number of clusters to be specified beforehand. It is particularly effective in identifying clusters of arbitrary shapes and handling noise in the data, making it well-suited for Twitter data, which often contains outliers and irregular patterns. Moreover, DBSCAN operates by identifying core points, which are data points with at least a specified number of neighboring points within a certain radius ϵ . The algorithm expands clusters from these core points by adding all points that are reachable within the ϵ neighborhood. Mathematically, a point ρ is considered a core point if:

$$|N_\epsilon(\rho)| \geq MinPts \quad (17)$$

where $N_\epsilon(\rho)$ is the neighborhood of point ρ with radius ϵ and $MinPts$ is the minimum number of points required to form a dense region. Thus, in our dataset, DBSCAN is used to identify dense clusters of tweets and users that represent active communities. These might be groups of users discussing a specific event and those frequently interacting with each other. The ability of DBSCAN to handle noise is particularly valuable for filtering out irrelevant and spam content, which is common in OSNs datasets

3. Hierarchical Clustering

Hierarchical clustering is a method that builds a hierarchy of clusters through a tree-like structure called a dendrogram. This approach is useful for understanding the nested relationships between users or tweets, where smaller clusters can be merged into larger ones, revealing the structure of communities at different levels of granularity. Hierarchical clustering can be either agglomerative (bottom-up) or divisive (top-down). In agglomerative clustering, each data point starts as its own cluster, and pairs of clusters are merged iteratively based on a similarity criterion until all points are in a single cluster. The similarity or distance between clusters can be measured in several ways. The distance between two clusters is the minimum distance between any two points in the clusters and can be represented as follows:

$$D_{Single}\left(C_i, C_j\right) = d(x, y) \quad (18)$$

Also, the distance between two clusters is the maximum distance between any two points in the clusters and can be represented as follows:

$$D_{Complete}\left(C_i, C_j\right) = d(x, y) \quad (19)$$

In addition, the distance between two clusters is the average distance between all pairs of points in the clusters and can be represented as follows:

$$D_{Average}\left(C_i, C_j\right) = \frac{1}{|C_i| \cdot |C_j|} \sum_{x \in C_i, x \in C_{i,}}^{d} (x, y) \quad (20)$$

Therefore, in our dataset, Hierarchical clustering is particularly useful when we are interested in understanding the hierarchical nature of discussions. Yet, a large community discussing a broad topic might have sub-communities focusing on specific aspects of the event. The dendrogram produced by hierarchical clustering can visually represent these nested relationships, providing insights into the structure and dynamics of online conversations.

4. Spectral Clustering

Spectral clustering is a technique that uses the eigenvalues of a similarity matrix derived from the data to perform dimensionality reduction before clustering in fewer dimensions. This approach is particularly effective when the data has a complex structure not well captured by traditional clustering methods. Spectral clustering involves constructing a similarity graph$G = (V, E)$, where each node represents a data point, and edges between nodes represent the similarity between those points. The similarity matrix W is then used to compute the graph Laplacian L as follow:

$$L = D - W$$
(21)

where D is the degree matrix and be a diagonal matrix where each entry D_{ii} is the sum of the corresponding row of W . The algorithm then computes the eigenvectors of L , which are used to project the data into a lower-dimensional space where traditional clustering methods like K-means can be applied. Therefore, in our dataset, Spectral clustering is used to identify communities that are not linearly separable in the original feature space. Thus, it can detect overlapping communities where users may belong to multiple clusters based on their diverse interests and interactions. This method is particularly useful when the goal is to uncover complex patterns of user behavior and topic engagement that might be missed by other clustering techniques.

5. Inverted Indices and Incremental Clustering

To address computational challenges, we implement specialized inverted indices and incremental clustering. In Inverted Indices, we facilitate efficient data retrieval by mapping terms to their locations within the dataset as follows:

$$index = \{t: [d_1, d_2, d_3, \ldots., d_k]\}$$
(22)

where t is the term and d_i are the document IDs containing t . Yet, in Incremental Clustering, we continuously update clusters as new data arrives, reducing computational load and maintaining real-time detection capabilities as follows:

$$\mu_i^{new} = \frac{n_i \mu_i + x_j}{n_i + 1}$$
(23)

where we update C_i with new data point x_j and μ_i^{new} is the updated centroid and n_i is the number of points in cluster i

V. Results and Discussion

A comprehensive parameter sensitivity analysis ensures the optimal performance of our algorithms. Thus, we adjust parameters such as the number of clusters k , epsilon ϵ and minimum points $MinPts$ in DBSCAN, and linkage criteria in hierarchical clustering, the number of components in spectral clustering, and configurations in inverted indices and incremental clustering. The goal was to maximize performance by systematically adjusting parameters using techniques like grid search and cross-validation. This fine-tuning process involves first the Grid search which to identify the best parameter settings, a grid search was employed, systematically exploring different combinations of parameters. The optimal parameters θ_{opt} were selected by maximizing the performance function as follows:

$$\theta_{opt} = arg\ arg \quad performance(\theta)$$
(24)

This method ensured that the best possible configurations were chosen for each algorithm based on performance metrics. The second one is Cross-Validation, which was used to evaluate the robustness of parameter settings by dividing the dataset into multiple subsets $(folds)$ and testing the model on different combinations of these subsets. The cross-validation score $(CV - Score)$ was calculated as the average performance across all folds as follows:

$$CV - Score = \frac{1}{k} \sum_{i=1}^{k} Performance(Fold_i)$$
(25)

This approach helped ensure that the selected parameters performed well across different parts of the dataset, reducing the risk of overfitting. To evaluate the effectiveness of our proposed system, we compared its performance against five state-of-the-art baseline models commonly used for event detection and community analysis on OSNs platforms. The metrics used for comparison were precision, recall, and F1-score. These metrics provide a comprehensive assessment of the models' ability to accurately detect events and identify relevant communities. We have built up different settings and we found specific findings from Parameter Sensitivity Analysis. In K-Means Clustering, the parameter is the number of clusters (k) and the optimal value is $k = 5$. Thus, the finding is revealed that K=5 provided the best balance between cluster separation and computational efficiency. In DBSCAN, the parameters are Epsilon (ε) and $MinPts$, yet the optimal values are $\varepsilon = 0.5$, $MinPts = 5$. Thus, the finding reveals that smaller ε values increased noise, while larger values merged distinct clusters. Also, a $MinPts$ value of 5 was optimal for reducing noise without losing valid points.

Moreover, in Hierarchical Clustering**,** the parameter is Linkage Criteria and the optimal choice is the average linkage. Thus, the finding is the average linkage that provided the best balance between cluster separation and robustness to outliers. In Spectral Clustering, the parameter is the number of components, and the optimal value is 10 components. Thus, the finding reveals that the analysis showed that using 10 components provided a good representation of the data structure without introducing noise. In Inverted Indices and Incremental Clustering, the optimal configuration is the term frequency, yet, threshold is 0.01 and the cluster update frequency is every 100 new data points. Thus, the finding reveals that lowering the term frequency threshold improves detection accuracy but also increases computational cost, while an update frequency of 100 data points are maintained efficiency without sacrificing accuracy.

The key component of our integrated approach is using the strength of each clustering algorithm. In the K-Means clustering, we have used it for initial partitioning of data, providing a baseline structure for more complex analysis. In the DBSCAN, we have Applied it to refine clusters by identifying dense regions and filtering out noise, ensuring robustness to outliers. In Hierarchical clustering, we have used it for exploring multi-level clusters, capturing both broad and specific event categories. In Spectral clustering we have used it to help in detecting complex social structures and overlapping communities, adding depth to the analysis. Finally, in Inverted indices and incremental clustering, we have used it to facilitate real-time event detection by allowing the system to handle dynamic data and continuously update clusters as new information becomes available. Therefore, our proposed integrated approach incorporates the strengths of the above clustering methods to form a robust system for event detection and community analysis on OSNs platforms. By leveraging multiple algorithms, the system adapts to different data structures and types, handling noise, detecting complex social structures, and maintaining real-time responsiveness. Then, the use of inverted indices and incremental clustering further enhances the system's ability to process large datasets efficiently and update event detection models in real time, ensuring more accurate event identification and community detection.

The accuracy and performance of the proposed system, that illustrated in Figure 1 and Figure 2, were evaluated across multiple epochs and compared with several baseline models, including K-Means Clustering, DBSCAN, Hierarchical Clustering, Spectral Clustering, and Inverted Indices & Incremental Clustering. The goal, here, was to assess how well each model performs in terms of accuracy and overall clustering effectiveness over time.

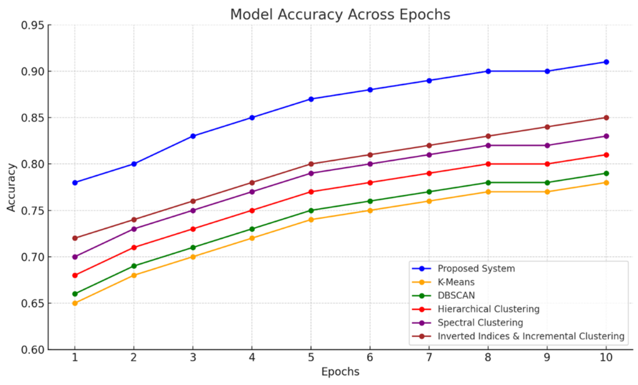

Figure 1. Accuracy of all models.

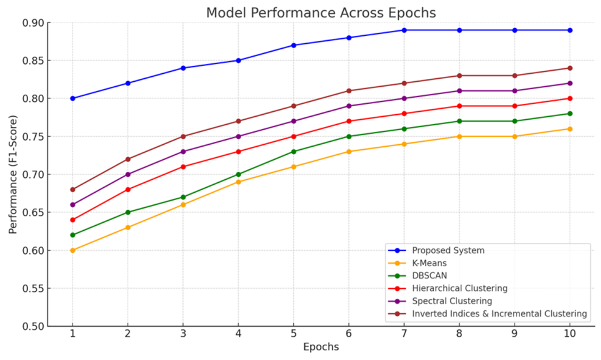

Figure 2. Performance of all models.

In terms of a model's accuracy, it is a critical metric that measures its ability to correctly classify events and communities within the dataset. After successfully applying our experiment with different values of epochs, the accuracy of each model was tracked to observe how it improved with continued training and parameter adjustments. Figure 1 illustrates the accuracy trends across the different models. Thus, the proposed system shows a strong and consistent increase in accuracy, reaching a high of 0.91 by the final epoch. This indicates that the integrated approach is highly effective in accurately identifying relevant events and communities as it continues to learn from the data.

The proposed system's ability to combine multiple clustering techniques allows it to handle the complexity and noise typical of OSNs data better than the baseline models. In addition, the Inverted Indices & Incremental Clustering performs well by achieving an accuracy of 0.85. Its real-time adaptability, which allows for continuous updates and efficient data retrieval, makes it a strong performer, especially in dynamic environments like Twitter. Also, the Spectral clustering achieves an accuracy of 0.83, showing that it is effective at handling complex and non-spherical clusters. However, it slightly lags behind the proposed system and inverted indices due to its reliance on eigenvalues for dimensionality reduction, which may not capture all nuances in the data. Moreover, with an accuracy of 0.81 the hierarchical clustering demonstrates steady improvement.

This method's ability to explore multi-level clusters is beneficial, but it is still less effective than the proposed system in capturing the full complexity of social media interactions. Finally, the DBSCAN and K-Means clustering methods show lower accuracy, with DBSCAN reaching 0.79 and K-Means 0.78. These models struggle with the inherent noise and variability in OSNs data, which limits their overall performance compared to more advanced approaches. In addition to accuracy, the F1-score was used to measure the performance of each model across epochs as shown in Figure 2. The F1-score balances precision and recall, providing a more comprehensive view of how well each model identifies relevant events without misclassifying irrelevant data. The performance trends across epochs are depicted in Figure 2. The proposed system achieves the highest F1-score, reaching 0.89. Its rapid convergence to a high-performance level highlights its robustness and efficiency in handling event detection tasks. Yet, the Inverted Indices & Incremental Clustering performs strongly, reaching an F1-score of 0.84. Its ability to adapt to incoming data in real-time contributes to its strong performance, particularly in fast-moving environments. Also, the Spectral clustering shows steady improvement, reaching an F1-score of 0.82 While effective, it does not achieve the same level of performance as the proposed system due to its limitations in handling overlapping communities and non-linear structures.

In addition, the Hierarchical clustering achieves an F1-score of 0.80, showing that it is capable of effective clustering but is less adaptable to the dynamic nature of OSNs data compared to more advanced models. Finally, the DBSCAN and K-Means models show slower improvement and lower overall F1-scores, with DBSCAN reaching 0.78 and K-Means 0.76. Their inability to handle noisy data and complex social structures limits their performance in event detection. Conclusion The proposed system consistently outperforms all baseline models in terms of both accuracy and overall performance (F1-score). The combination of multiple clustering techniques and feature extraction methods enables it to handle the challenges of social media data more effectively than traditional methods. The charts below provide a visual comparison of model accuracy and performance across epochs, emphasizing the superiority of the proposed system in real-time event detection and community analysis.

Figure 3 shows the confusion matrix visualizes the performance of the proposed system for event detection. Figure 4 shows the relations betweens events that's populated from one event to another.

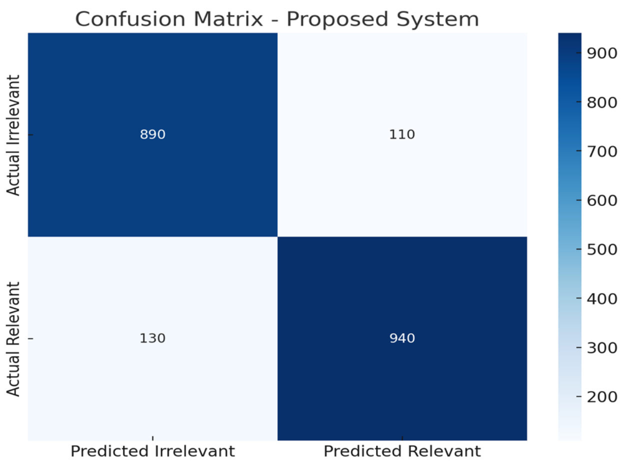

Figure 3. Confusion Matrix.

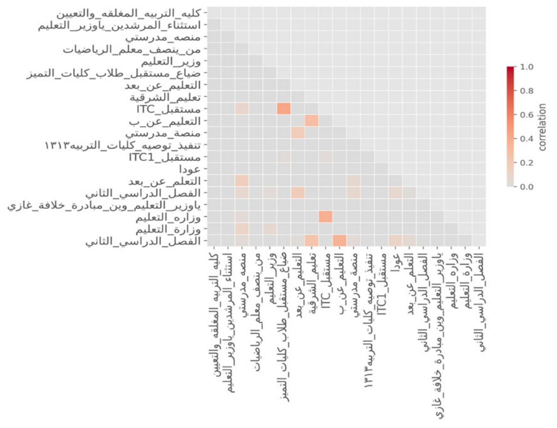

Figure 3. events relations..

VI. Conclusion

This study presents a comprehensive approach to real-time event detection and community analysis on Twitter. By integrating clustering algorithms, temporal modules, and social network graphs, we developed a robust framework capable of identifying and understanding online events with high accuracy. Overcoming the challenges of data volume, velocity, and complexity was essential for ensuring accurate and timely detection of events. Our proposed system demonstrated significant improvements over existing methods, offering a valuable tool for monitoring and mitigating the impact of community attacks on Online Social Networks (OSNs).

The parameter sensitivity analysis highlighted the importance of fine-tuning key parameters across various algorithms. By carefully adjusting parameters such as the number of clusters in K-Means, epsilon and MinPts in DBSCAN, linkage criteria in hierarchical clustering, components in spectral clustering, and configurations in inverted indices and incremental clustering, we were able to enhance the overall performance of the system. This process ensured that our models operated at their best, providing reliable and effective event detection capabilities across different scenarios.

References

[1] Alowibdi, Jalal S., Ugo A. Buy, and Philip Yu. "Language independent gender classification on Twitter." Proceedings of the 2013 IEEE/ACM international conference on advances in social networks analysis and mining. 2013.

[2] Khan, H. U., Daud, A., Ishfaq, U., Amjad, T., Aljohani, N., Abbasi, R. A., & Alowibdi, J. S. (2017). Modelling to identify influential bloggers in the blogosphere: A survey. Computers in Human Behavior, 68, 64-82.

[3] Alowibdi, J. S., Buy, U. A., & Yu, P. (2013, December). Empirical evaluation of profile characteristics for gender classification on twitter. In 2013 12th international conference on machine learning and applications (Vol. 1, pp. 365-369). IEEE.

[4] Saving lives using social media: analysis of the role of twitter for personal blood donation requests and dissemination.

[5] Abbasi, R. A., Maqbool, O., Mushtaq, M., Aljohani, N. R., Daud, A., Alowibdi, J. S., & Shahzad, B. (2018). Saving lives using social media: analysis of the role of twitter for personal blood donation requests and dissemination. Telematics and Informatics, 35(4), 892-912.

[6] Alowibdi, J. S., Buy, U. A., Philip, S. Y., & Stenneth, L. (2014, August). Detecting deception in online social networks. In 2014 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM 2014) (pp. 383-390). IEEE.

[7] Alowibdi, J. S., Buy, U. A., Yu, P. S., Ghani, S., & Mokbel, M. (2015). Deception detection in Twitter. Social network analysis and mining, 5, 1-16.

[8] Jarwar, M. A., Abbasi, R. A., Mushtaq, M., Maqbool, O., Aljohani, N. R., Daud, A., ... & Chong, I. (2017). CommuniMents: A framework for detecting community based sentiments for events. International Journal on Semantic Web and Information Systems (IJSWIS), 13(2), 87-108.

[9] Alowibdi, J. S., Alshdadi, A. A., Daud, A., Dessouky, M. M., & Alhazmi, E. A. (2021). Coronavirus pandemic (COVID-19): Emotional toll analysis on Twitter. International Journal on Semantic Web and Information Systems (IJSWIS), 17(2), 1-21.

[10] Gillani, M., Ilyas, M. U., Saleh, S., Alowibdi, J. S., Aljohani, N., & Alotaibi, F. S. (2017, April). Post summarization of microblogs of sporting events. In Proceedings of the 26th international conference on World Wide Web companion (pp. 59-68).

[11] Sakaki, Takeshi, Makoto Okazaki, and Yutaka Matsuo. "Earthquake shakes twitter users: real-time event detection by social sensors." Proceedings of the 19th international conference on World wide web. 2010.

[12] Petrovic, Sasa, Miles Osborne, and Victor Lavrenko. "Streaming first story detection with application to twitter." Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics (HLT'10).

[13] Cordeiro, M. Twitter event detection: combining wavelet analysis and topic inference summarization. In Doctoral symposium on informatics engineering (Vol. 1, pp. 11-16).

[14] Alharbi, A., & Lee, M. (2021, April). Kawarith: an Arabic Twitter corpus for crisis events. In Proceedings of the Sixth Arabic Natural Language Processing Workshop (pp. 42-52).

[15] Alsaedi, Nasser, and Pete Burnap. "Arabic event detection in social media." Computational Linguistics and Intelligent Text Processing: 16th International Conference, CICLing 2015, Cairo, Egypt, April 14-20, 2015, Proceedings, Part I 16. Springer International Publishing, 2015.

[16] Alkouz, Balsam, and Zaher Al Aghbari. "Fusion of Multiple Arabic Social Media Streams for Traffic Events Detection." 2022 7th International Conference on Big Data Analytics (ICBDA). IEEE, 2022.