SC4OSINT: A Story Clustering Approach to Optimize OSINT Analysis

Elisabeth Woisetschläger, Medina Andresel, Florian Skopik(✉), Benjamin Akhras, Peter Leitmann, Max Landauer, Markus Wurzenberger, and Alexander Schindler

AIT Austrian Institute of Technology GmbH, Vienna, Austria
{elisabeth.woisetschlaeger,medina.andresel,florian.skopik}@ait.ac.at

Abstract. Cyber Threat Intelligence (CTI) has become an indispensable element of cybersecurity operations and any mechanism or tool that alleviates the workload of security analysts is highly valuable. Natural Language Processing (NLP) supports efficient processing of news articles, and enables us to group articles that report about the same story. This allows Open Source Intelligence (OSINT) analysts to manage information overload and focus only on essential events. Therefore, the contributions of this paper are manyfold: (i) We identify the relevant requirements for designing an OSINT clustering tool, (ii) present a solution that can support such requirements, and (iii) evaluate the solution considering the needs of OSINT analysts. Our clustering approach, denoted as SC4OSINT, is inspired by an existing semi-supervised graph-based story clustering method and adapted to the OSINT requirements. Unlike the original method, SC4OSINT is a fully unsupervised two-layer approach, which handles multilingual streaming data and uses sentence transformers to create fine-grained clusters. We evaluate SC4OSINT's story clustering by letting security experts rate the clustering quality across various model configurations. The results show that the best hyper-parameter configuration achieves an average rating of 4.19/5, demonstrating the efficiency of our approach.

Keywords: OSINT · CTI · NLP · Story-based Clustering · Streaming Data · Analysis Time Optimization · NER · User-based Evaluation · Taranis AI

1 Introduction

Cyber Threat Intelligence (CTI) is vital to cybersecurity operations, supporting tasks from vulnerability management to intrusion detection and risk mitigation to respond to emerging threats [14]. Yet, the volume of data from diverse feeds overwhelms analysts, increasing the risk of missed threats and costly consequences. Tools that reduce this burden while preserving accuracy are therefore

B. Coppens et al. (Eds.): ARES 2025 Workshops, LNCS 15995, pp. 5–24, 2025.
https://doi.org/10.1007/978-3-032-00633-2_1

critical. The goal of our work is to alleviate the burden on analysts, enabling more efficient and timely ingestion of vital information. Story clustering aims to group related documents – represented as news items in this paper – into coherent stories, thereby significantly reducing redundancy in the analyst's workflow. For instance, major events like new software vulnerabilities or critical cloud outages are reported by many sources in varying styles and languages. Analysts face the cumbersome task of sorting through these repetitive reports to extract unique insights while maintaining an extensive understanding of the event.

By leveraging Natural Language Processing (NLP) techniques [4], story clustering methods group together related news items that report about the same event. This ensures that analysts can initially review a single representative cluster item instead of multiple redundant news items. Clustering cybersecurity is challenging, as articles often vary widely in length, style, and language (e.g., a blog article, a mailinglist entry, and a story of a tech magazine). Additionally, cybersecurity news is highly dynamic, with changing terminologies, such as novel advanced persistent threat (APT) group names, vulnerabilities, or software, emerging almost daily. In general, mistakenly clustering unrelated articles can lead to overlooking critical information, whereas treating related items separately is less ideal but tolerable. Moreover, certain text types, such as daily digests, research papers, or highly technical articles, may be less suitable for clustering due to their highly specialized content or the inclusion of multiple stories within a single news item. Furthermore, the solution must operate efficiently on standard CPUs under resource constraints, rather than relying on costly GPU infrastructures. To the best of our knowledge, there is no existing unsupervised clustering solution that can cope with all such requirements.

Our work draws inspiration from EventX [7], an existing story-based clustering approach, to better address the challenges inherent to Open Source Intelligence (OSINT) data, such as news feeds, blog posts, and cybersecurity articles. The existing approach employs keyword community detection [13] and a supervised classification method to determine whether two text passages describe the same event. An extension of this approach is essential due to the following limitations: (i) It employs a semi-supervised clustering method, making it unsuitable for the OSINT domain due to the lack of annotated data; (ii) it is incapable of handling multilingual data, which is critical for obtaining a broad range of security-relevant information necessary to efficiently respond to emerging threats; (iii) to handle streaming data, it compares each new event with all existing ones, which is time- and resource-intensive. This approach is unsuitable for OSINT due to the massive volume of streaming data. Additionally, the reliance on costly GPU clusters would further limit its adoption by Computer Emergency Response Teams; (iv) it considers all keywords as equally important, while we prioritize those most relevant to the OSINT domain.

Our approach SC4OSINT[1] (i.e., story clustering for OSINT) fulfills all above mentioned requirements, as it is a fully unsupervised approach, prioritizes the keywords that are relevant to the OSINT domain, handles multilingual and

[1] https://github.com/taranis-ai/story-clustering.

streaming data and is tailored for CPUs. Additionally, we present an OSINT evaluation dataset to address the lack of a pre-existing benchmark. This dataset also includes extracted keywords (tags in our use case). We apply SC4OSINT to this dataset and have security experts evaluate the clustering quality.

The main contributions of this paper are:

- **Motivation and Requirements:** We highlight the necessity of Open-Source Intelligence (OSINT) clustering in cybersecurity and outline specific requirements and design considerations.
- **Clustering Approach:** We propose a novel clustering method tailored to cybersecurity news items, designed for practical application with real-world OSINT data.
- **Dataset and Evaluation:** We introduce an OSINT dataset and demonstrate the efficacy of our approach through its application, followed by a detailed presentation and discussion of the evaluation results.

The remainder of this paper is structured as follows: Sect. 2 reviews related work, and Sect. 3 defines the problem and requirements. Section 4 presents SC4OSINT's components, including tagging, keyword community detection, and story-based clustering. Section 5 details the dataset, experiments, and user study, followed by the conclusion in Sect. 6.

2 Related Work

NLP [4] enables computers to process diverse forms of human language, from formal documents to informal messages, supporting applications like cyber situational awareness (CSA). Recent advances in deep learning and transformer models (e.g., BERT, GPT-3 [17]) have led to effective NLP-based CSA techniques for organizing and classifying text data [16]:

Named Entity Recognition (NER). NER, a subfield of NLP, extracts structured data by identifying key entities such as locations, persons, or cyber-specific terms like Advanced Persistent Threat (APT) groups [15]. It enhances clustering by providing domain-relevant features and is used in tools like TTPXHunter and TRAM to categorize Tactics, Techniques, and Procedures (TTPs) [11]. Our approach combines a multilingual NER model with word-matching to extract entities such as APTs, Indicators of Compromise (IoCs) and Common Vulnerabilities and Exposures (CVE) IDs (see Sect. 4.1).

Text-Based Clustering. Unsupervised methods like k-means and DBSCAN help reduce information overload in Computer Emergency Response Teams (CERT) analysis tools [6], but the lack of cybersecurity-specific text benchmarks limits evaluation of such NLP techniques in this context. While we share the goal of easing analyst workload, our approach applies more fine-grained clustering criteria (see Sect. 3). Ma et al. [8] propose an unsupervised clustering approach using Twitter data to extract and summarize threat information for infrastructure security. Their pipeline applies NER and filters infrastructure-specific features before clustering with HDBSCAN. Liu et al. [7] propose EventX, a semi-supervised method using keyword communities and a trained classifier to link event-related text. Inspired by this, we adapt the approach, due to lack of annotated data, with pre-trained models for keyword extraction and similarity detection, enabling dynamic clustering on streaming data. We further evaluate its impact on reducing information overload in the cybersecurity domain. Regarding supervised clustering, Riebe et al. [12] use an event detection method based on similarity metrics applied to Twitter data to create an alert system for CSA. Another supervised approach relies upon a novel cyber threat unified taxonomy to classify OSINT feed text based on tags and remove information with low value [9]. None of these approaches fully meet our requirements, as Twitter data lack the depth of information found in OSINT data, and focusing only on classification provides limited threat information. Furthermore, our approach extracts keywords not only using NER but also through a word-matching algorithm, and the extraction of Indicators of Compromise (IoCs).

3 Problem Statement

Open-source threat information is continuously growing, challenging security analysts in identifying the most relevant events. The main objective of SC4OSINT is to minimize the analysis time, however, there are several requirements, inherent to OSINT domain, to be considered. These requirements, outlined below, were identified through three workshops with experts from national CERTs.

R1: Analysis time shall be optimized. To reduce analysts time, similar news items must be clustered so they can be evaluated collectively. These clusters support further analysis, like automated summaries for decision-making. However, if clustering ignores news heterogeneity, unrelated items may be grouped together, risking missed information or requiring manual splitting, which is inefficient.

R2: The method shall account for heterogeneity of news items.We collect news items with Taranis AI [1], an OSINT platform, as described in Sect. 5.1, which integrates a variety of German and English OSINT sources, thus requiring a multilingual clustering approach. The wide thematic range of content, including special items like daily summaries, duplicates, and arXiv entries[2], influences clustering, thus must be handled differently. Feedback from analysts has shown that clustering these special items reduces the system's usability and increases the analysis time. Therefore, arXiv sources, daily reports and summaries shall be treated differently from the majority of regular news items, as they would negatively impact the clustering performance.

[2] https://arxiv.org/.

R3: The approach shall be unsupervised and handle streaming data. Due to lack of OSINT annotated data, an unsupervised clustering approach is required. As threat information is continuously generated, new items must be matched to existing clusters or form new ones to help analysts quickly spot trends and respond to threats. Incorrectly merging unrelated items, however, can slow analysis and hinder decision-making.

R4: The model shall run on-premise and shall be deployed on CPUs. Since OSINT is often combined with closed-source information (such as internal reports, analysts' comments, and conclusions), information processing must be handled on-premise. Additionally, only a minority of CERTs operate their own GPU clusters or have access to one for processing (confidential) data.

R5: The approach shall be evaluated by security experts. As the proposed unsupervised clustering approach is novel to the OSINT domain, its performance must be evaluated by security experts. To this end, we introduce an OSINT dataset and apply SC4OSINT to it. The resulting clusters must then be presented to the experts in a meaningful and verifiable manner.

R6: The model shall consider the last seven days of data. Through our collaboration with CERT analysts, we learned that only the most recent seven days of data are relevant for their analysis. Consequently, the clustering approach must focus on the last week of data. Additionally, to handle streaming data, clusters are updated using the most recent seven days of data.

Meeting these requirements is challenging, as state-of-the-art methods fall short (see Sect. 2). SC4OSINT introduces a novel, CPU-efficient unsupervised clustering approach for heterogeneous OSINT data streams. Additionally we provide a benchmark dataset for evaluating clustering quality.

4 Story Clustering for OSINT (SC4OSINT) Method

Our approach, inspired by EventX [7], follows three phases: pre-processing, community detection, and fine-grained clustering. As shown in Fig. 1, SC4OSINT introduces key modifications (highlighted in blue) to fulfill the OSINT requirements. The next subsections and Alg. 1 detail each phase and our innovations.

4.1 Pre-processing Phase

To extract keywords from OSINT data, we rely upon Taranis AI, an open-source OSINT platform. A *keyword*, also known as a *tag*, in a text document is an important word or group of words which describes a main topic or theme in the text. Thus, in the initial phase, Taranis AI extracts keywords, from each provided document (news item) in the input corpus (the collection of input documents). Unlike EventX's supervised keyword extraction, SC4OSINT employs an unsupervised method due to the lack of annotated data, and tailors its keyword extraction process to the security domain. These keywords are the basis for constructing one input keywords graph for the entire corpus. A *keywords graph*,

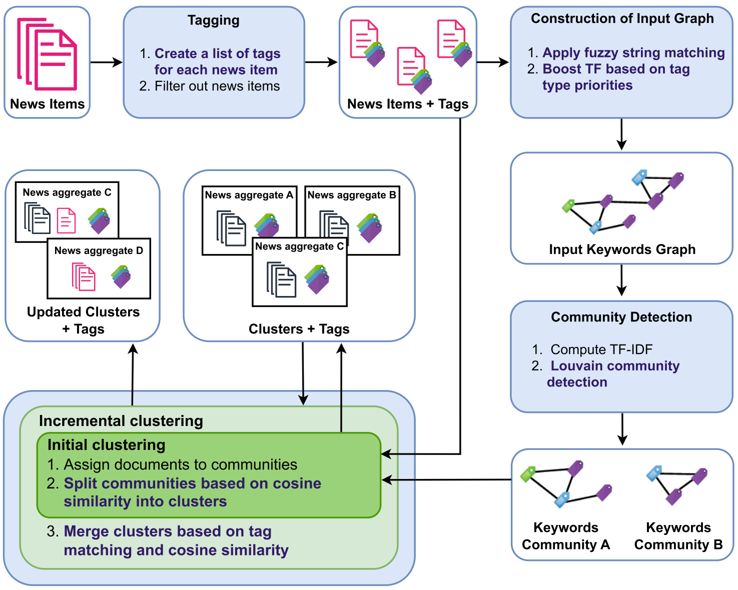
News Items
Tagging
1. Create a list of tags for each news item
2. Filter out news items
News Items + Tags
Construction of Input Graph
1. Apply fuzzy string matching
2. Boost TF based on tag type priorities
Input Keywords Graph
Community Detection
1. Compute TF-IDF
2. Louvain community detection
Keywords Community A
Keywords Community B
Incremental clustering
Initial clustering
1. Assign documents to communities
2. Split communities based on cosine similarity into clusters
3. Merge clusters based on tag matching and cosine similarity
News aggregate A
News aggregate B
News aggregate C
Clusters + Tags
News aggregate C
News aggregate D
Updated Clusters + Tags

Fig. 1. Overview of the SC4OSINT approach.

is an undirected graph in which nodes represent keywords and edges represent co-occurrence of two keywords in the same document. EventX uses a supervised method to extract representative keywords [7], but this isn't applicable to our case due to the lack of a public model and annotated security data. Instead, we use fuzzy string matching to identify keyword co-occurrences. As in EventX, the resulting keyword graph represents key topics and supports keyword community detection.

Tagging. Taranis AI automatically assigns tags to each news item using a multilingual NER model, a word-matching algorithm, and/or the extraction of Indicators of Compromise (IoCs) and Common Vulnerabilities and Exposures (CVE) IDs. The tagging is tailored to the security domain, and the heterogeneity of news items (R2) is addressed.

In detail, Taranis AI performs NER tagging using the 'Flair' framework [2], with the standard 4-class multilingual NER model, covering English, German, Dutch, and Spanish. The four tag classes are person name (PER), location name (LOC), organization name (ORG), and miscellaneous name (MISC). Taranis AI only utilizes NER tags that are longer than two characters and have a confidence score[3] greater than 0.97. Both values were derived empirically by the developers of Taranis AI. Additionally, tags are extracted using a word-matching

[3] The confidence score in the 'Flair' framework is an optional value that specifies the label's confidence level, ranging from 0 to 1, with a default of 1.

Algorithm 1. SC4OSINT Algorithm

```
procedure InitialClustering($\mathcal{C} = \{\langle\mathcal{D}_1, \mathcal{T}_1\rangle, \ldots, \langle\mathcal{D}_n, \mathcal{T}_n\rangle\}$)
    for $i \leftarrow 1$ to $n$ do
        if $size(\mathcal{T}_i) \leq$ MT then
            $\mathcal{C} \leftarrow \mathcal{C} \setminus \{\langle\mathcal{D}_i, \mathcal{T}_i\rangle\}$        ▷ Filter out documents based on MT
    $\mathcal{G} \leftarrow CreateKeywordsGraph(\mathcal{C})$        ▷ Create co-occurrence graph
    Compute TF-IDF for $\mathcal{D}_1, \ldots, \mathcal{D}_n$
    $\mathcal{G}_1, \ldots \mathcal{G}_k \leftarrow LouvainCommunityDetection(\mathcal{G},\mathcal{C})$
    for $\langle\mathcal{D}, \mathcal{T}\rangle \in \mathcal{C}$ do        ▷ Doc to keywords community
        $\ell \leftarrow argmax(\{s_i \mid s_i = KeywordsSim(\mathcal{G}_i, \mathcal{T}), 1 \leq i \leq k\})$
        Add $\mathcal{D}$ to the set of documents of community $\mathcal{G}_\ell$
    Let $\mathcal{S}_i \leftarrow \{\mathcal{D} \mid \mathcal{D}$ assigned to $\mathcal{G}_i\}$, for $1 \leq i \leq k$
    $\mathcal{A} = \emptyset$
    for $i \leftarrow 1$ to $k$ do
        for unprocessed $\mathcal{D} \in \mathcal{S}_i$ do        ▷ Initially all unprocessed
            $\mathcal{S}_\mathcal{D} \leftarrow \emptyset$
            for unprocessed $\mathcal{D}' \neq \mathcal{D} \in \mathcal{S}_i$ do
                Let $\mathbf{D}$, $\mathbf{D}'$ denote the sentence embeddings of $\mathcal{D}, \mathcal{D}'$
                if CosineSimilarity$(\mathbf{D}, \mathbf{D}') \geq$ ST then
                    Add $\mathcal{D}, \mathcal{D}'$ to $\mathcal{S}_\mathcal{D}$ and remove them from $\mathcal{S}_i$
                Mark $\mathcal{D}'$ as processed
            Mark $\mathcal{D}$ as processed
            Add $\mathcal{S}_i$ and non-empty $\mathcal{S}_\mathcal{D}$ to $\mathcal{A}$
    Return $\mathcal{A} = \{\mathcal{S}_1, \ldots, \mathcal{S}_m\}$        ▷ Returning the story-based clusters

procedure IncremClustering($\mathcal{C} = \{\langle\mathcal{D}_1, \mathcal{T}_1\rangle, \ldots, \langle\mathcal{D}_n, \mathcal{T}_n\rangle\}$, $\mathcal{A} = \{\mathcal{S}_1, \ldots, \mathcal{S}_m\}$)
    $\mathcal{A}' \leftarrow InitialClustering(\mathcal{C})$        ▷ New clusters
    for $\mathcal{S}'$ in $\mathcal{A}'$ do
        Let $maxScore = -1$, $superClsID = None$
        for $\mathcal{S}$ in $\mathcal{A}$ do        ▷ Iterate over existing clusters
            Let $\mathcal{U}'$, resp. $\mathcal{U}$, be the union of the tags of all documents in $\mathcal{S}'$, resp. $\mathcal{S}$
            if $FuzzyStringMatching(\mathcal{U}',\mathcal{U}) \geq 3$ then
                Pick documents $\mathcal{D}, \mathcal{D}'$ from $\mathcal{S}'$, and $\mathcal{D}''$ from $\mathcal{S}$, and let $\mathbf{D}$, $\mathbf{D}', \mathbf{D}''$ denote their
                sentence embeddings
                $score_1 \leftarrow CosineSimilarity(\mathbf{D}, \mathbf{D}')$
                $score_2 \leftarrow CosineSimilarity(\mathbf{D}, \mathbf{D}'')$
                if $score_2 - score_1 \geq$ DST then
                    if $score_2 \geq$ ST and $score_2 \geq maxScore$ then
                        $maxScore \leftarrow score_2$ and $superClsID \leftarrow$ ID of $\mathcal{S}$
        if $superClsID \neq None$ then        ▷ Merge clusters
            Add each $\mathcal{D}'$ in $\mathcal{S}'$ to the cluster with $superClsID$ in $\mathcal{A}$
        else Add $\mathcal{S}'$ to $\mathcal{A}$
    Return $\mathcal{A}$        ▷ Returning the new and updated story-based clusters
```

algorithm by means of word lists provided by Taranis AI[4], which include vendors and products, APT groups and operations, countries, and Austrian municipals[5] Vendors and products are extracted based on CVE lists containing all current CVE records[6] and the MITRE ATT&CK framework[7] is used to gather APT group names. Countries and Austrian municipals are obtained with public lists provided by country.io[8] and Umweltbundesamt[9], respectively. Tags are then cre-

[4] https://github.com/taranis-ai/wordlists (last accessed: 2024-11-19).

[5] Taranis AI uses a word list containing Austrian municipalities, as the validation of the tool took place in Austria, involving experts from the Austrian national CERT.

[6] https://github.com/CVEProject/cvelistV5 (last accessed: 2024-11-19).

[7] https://attack.mitre.org/groups/ (last accessed: 2024-11-19).

[8] https://country.io/names.json.

[9] https://secure.umweltbundesamt.at/edm_portal/redaListOptimization.do?seqCode=8yc33c74k8xcc2&6578706f7274=1&display=plain&d-49520-e=1.

ated by matching words (case-insensitive) from the word list with the content of the provided news item, using the regex word boundary '\b'. Further tagging is performed using the Python IOC finder library[10]. The following categories are extracted from news items: Bitcoin addresses (P2PKH, P2SH, and Bech32), CVEs, registry key paths, SSDeep hashes, file hashes (MD5, SHA1, SHA256, and SHA512), and IPv4 Classless Inter-Domain Routing (CIDR) addresses.

News items and their tags serve as input to the clustering algorithm. Items with fewer tags than the threshold `MT` are filtered out early, as they lack sufficient relevant information (lines 2–4 in Algorithm 1).

Construction of Input Keywords Graph. In the next step (line 5, Algorithm 1), we build the input keyword graph by using unique tags as nodes and connecting them via edges based on their co-occurrence in documents, edge weights are determined by the total number of co-occurrences in the corpus. To determine tag co-occurrence, we use fuzzy string matching (PolyFuzz[11]) with a 0.65 threshold to detect identical or slightly varied forms within a document. The input keyword graph also stores the term frequency (TF) and document frequency (DF) for each tag in the corpus, computed using again fuzzy string matching. These metrics are used in the community detection phase to compute the term frequency–inverse document frequency metric (TF-IDF) based on which the first clustering layer is created. Unlike EventX, the TF of certain security related keywords, such as CVEs and APTs, is boosted by applying a predefined priority order to the tag types (see Table 1). This is important as they do not appear as frequently as other tag types and might not be considered in the community detection phase as explain in the following section.

4.2 Keywords Community Detection Phase

This section describes SC4OSINT's community detection phase, the first clustering layer. In contrast to EventX, we use the Louvain method [3] instead of betweenness centrality due to its lower computational complexity (fulfilling R4) and ability to incorporate edge weights. SC4OSINT leverages this by weighting edges with keyword co-occurrence frequency, grouping frequently co-occurring terms into the same community.

Computing Document TF-IDF. We compute the TF-IDF [10] for each document (line 6 in Algorithm 1), which is a numerical representation of the relevant information in the document, used to compute cosine similarity between documents and keywords communities. Since TF-IDF metric favors tags with high TF and low DF, we boosted security relevant TF values, as previously described. Therefore, tags such as CVEs (which typically have low TF and low DF) obtain higher TF-IDF values. This metric is then used for the document-keyword community assignment, as described below.

[10] https://hightower.space/ioc-finder/ (last accessed: 2024-11-19).
[11] https://maartengr.github.io/PolyFuzz/.

Community Detection Method. The input keywords graph is used to construct keywords communities – densely connected sub-graphs, which are loosely connected to other sub-graphs. Given that keywords are nodes and edges represent co-occurrence, with frequency of co-occurrences as weights, a keywords community denotes a topic. To detect communities, we use the Louvain community detection method [3] (line 7 in Algorithm 1) since it is computationally more efficient than the approach used by EventX and it is based on modularity optimization. In this algorithm each node is initially considered its own community. Nodes are then moved to the neighboring community that yields the highest increase in modularity, computed by taking into account edge weights. This step is repeated until no further modularity improvement is possible.

Assign Documents to Keywords Communities. Once the keywords communities are constructed, documents can be assigned to them. This step is essential in grouping documents that share the same topic. The TF-IDF metric is computed for each community, and based on this, the TF-IDF cosine similarity between documents and communities. We then assign a community to each document in the corpus by selecting the community with the highest similarity score (lines 8–11 in Algorithm 1). This step results in news stories of documents that were assigned to the same community.

4.3 Fine-Grained Clustering Phase

SC4OSINT handles streaming data, essential for continuously generated threat information (R3). Unlike EventX, which uses a supervised classifier on document pairs, SC4OSINT leverages pre-trained embedding models for semantic-based clustering. Moreover, to update existing clusters with incoming news items, EventX's costly comparison of new and existing clusters limits its suitability for OSINT, while SC4OSINT offers a more efficient alternative (R4). SC4OSINT uses a two-layered approach: an *initial* clustering step that groups news items without modifying existing clusters, followed by *incremental* clustering, where new items are added to or form clusters, potentially merging with existing ones.

Initial Clustering. In this step, we split the created communities into clusters, representing more fine-grained news stories (lines 12–22 in Algorithm 1). For this, and to meet requirement R2, we use a pre-trained multilingual sentence transformer model[12] to create embeddings of the first five sentences of each news article's text. We then compute the cosine similarity between the embeddings of each document in a community. If the similarity is above a certain threshold – `ST` denoting *similarity threshold*, a new cluster is created containing the documents which have higher similarity compared to the others in the community. We use sentence transformers instead of bag-of-words approaches as it preserves the

[12] https://huggingface.co/sentence-transformers/paraphrase-multilingual-MiniLM-L12-v2.

semantical meaning of the text. Moreover, we only use the first five sentences of each document due to requirement R4 from Sect. 3, considering that news articles typically follow the same structure where the most critical information is presented at the beginning. The fine-grained clusters are then post-processed to create *news aggregates*, which represent news articles that report the same story. Finally, each cluster is assigned a set of tags by taking the union of the tag sets from all documents within the cluster.

Incremental Clustering. In the incremental clustering step, we first perform initial clustering on the new news items, followed by merging the newly created clusters with the existing ones (line 25 in Algorithm 1). In our novel approach we first verify whether the keywords community of each new cluster and the set of tags of the pre-existing cluster share at least three[13] keywords, using the fuzzy string matching approach (line 29–30 in Algorithm 1). If so, the title and the first 5 sentences of one randomly selected document in the old and new cluster are used to compute the cosine similarity using the above mentioned sentence-transformer model (lines 31–33 in Algorithm 1). If the difference in cosine similarity between the documents of each cluster (new and existing) exceeds a threshold – `DST` denoting *difference-similarity threshold*, the existing cluster is extended to include all news items from the new cluster (lines 34–36 in Algorithm 1). As shown in Fig. 1, the output of the incremental clustering can include both novel and updated news aggregates (lines 37-40 in Algorithm 1).

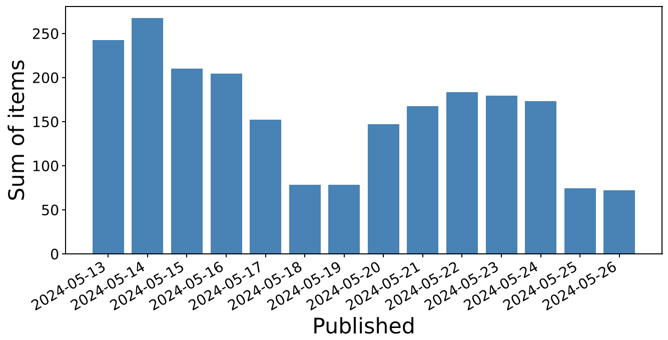

Fig. 2. Number of data items per day.

5 Evaluation

This section outlines the evaluation of our story clustering approach. Due to the lack of benchmarks, we created a dedicated evaluation dataset (R5) and tested

[13] A minimum of three keywords was chosen to ensure that similar clusters are not overlooked during comparison.

various hyper-parameter configurations. Feedback from security experts helped identify the optimal setup and assess its value for OSINT analysts.

5.1 Dataset

Creation of a Benchmark. Taranis AI collects news items from various publicly available sources. The admin user of the platform selects and configures these sources. Examples of relevant sources include Heise's security feed[14] and Help Net Security[15]. The platform automatically collects news items every eight hours and processes them for further use. This process involves extracting key information such as content, author, publication date, and title. The content is either directly sourced from the RSS feed or derived from the collected news item. When such extraction from the news item is needed, either the admin user provides an XPath expression specific to the source, or the extraction is performed using Trafilatura[16], a Python package for text processing. Additionally, tags are assigned to reflect the content of each news item (as described in Sect. 4).

In this work, we present a new OSINT dataset – AIT-OSINT-Summer2024, publicly available at Zenodo[17] consisting of news items stored in daily JSON files. To create the dataset, we utilized a Taranis AI instance to collect news items from 158 different open sources over a period of 12 weeks. Each day, the news items were gathered and exported by the platform over the previous two days into a JSON file. This overlapping content ensures that no information is lost due to time delays in the collection or export process. During post-processing, we merged all files, extracted unique news items, and organized them into files sorted by their creation dates. We filtered out items labeled with future dates or containing obvious errors from the retrieval phase, such as technical errors due to unreachable sources. Additionally, we removed attributes such as the user's relevance, ratings, and comments, as these pertain solely to the platform's usability and are not essential for the described algorithm. To fulfill requirement R6 we used only a subset of the entire dataset, applying incremental and initial clustering to one week of data at a time. Consequently, the first two weeks of data serve as the basis for all the experiments discussed in this paper.

Data Analysis. For the evaluation, we use 2,226 news items collected between May 13 and May 26, 2024, in German and English. Figure 2 shows the distribution of all news items; fewer items are published on weekends and a publication peak is recorded on May 14. Table 1 lists all tag types that occur within these two weeks, along with their priority and quantity. The priority of each tag type is used to weight the tags in the clustering algorithm (see Sect. 4). The total quantity of tags for each type is divided into quantities for the first and second week, respectively. The quantities per week is of importance, as we use the first

[14] https://www.heise.de/security/rss/news-atom.xml (last accessed: 2024-11-19).
[15] https://www.helpnetsecurity.com/feed (last accessed: 2024-11-19).
[16] https://trafilatura.readthedocs.io (last accessed: 2024-11-19).
[17] https://doi.org/10.5281/zenodo.14228995.

Table 1. Tag type frequencies

Tag Type	Priority	Quantity		
		Total	Week1	Week2
CVE	High	976	732	244
APT	High	133	68	65
SHA1	Mid-High	49	4	45
MD5	Mid-High	46	46	0
SHA256	Mid-High	60	51	9
Company	Mid-High	21	14	7
Registry key path	Mid-High	13	11	2
Bitcoin address	Mid-High	2	2	0
Country	Medium	2,461	1,844	617
ORG	Mid-Low	6,206	3,399	2,807
PER	Mid-Low	3,445	1,853	1,592
LOC	Mid-Low	1,586	902	684
MISC	Low	4,842	2,711	2,131
Cybersecurity	Low	760	442	318
IPv4 CIDR	Low	9	7	2
SSDeep hash	Low	4	4	0
SHA512	Low	1	1	0

week for initial clustering and the second for incremental clustering. Additionally, Table 1 indicates that tag types with medium priority or below are primarily assigned to the news items. Notably, tags of type ‘ORG’ are especially prevalent, representing organizations’ names such as ‘Apple’. Tag types of high and mid-high priority mostly occur in the first week, with CVEs being the most frequent. In general, more tags are assigned during the first week than during the second.

Figure 3 displays the ten most common tags in the dataset. ‘Microsoft’, ‘Google’ and ‘Apple’ are categorized as ORG tags. ‘U.S.’ represents a location (LOC), while ‘ransomware’ and ‘phishing’ fall under the tag type ‘Cybersecurity’, representing a word list. ‘China’, ‘United States’ represent Countries, and ‘Windows’ and ‘Linux’ are classified as MISC tags. Hereby, the tags ‘U.S.’, and ‘United States’ refer to the same country but are tagged with different techniques. Additionally, as stated above, the collected news items are in German or English, and thus tags can appear in multiple languages. Both effects (synonymity and multilingualism) must be considered in the clustering algorithm.

5.2 Experiments

The main goal of the experiments is to measure the quality of the clustering solution and to identify the optimal configuration based on the hyper-parameters

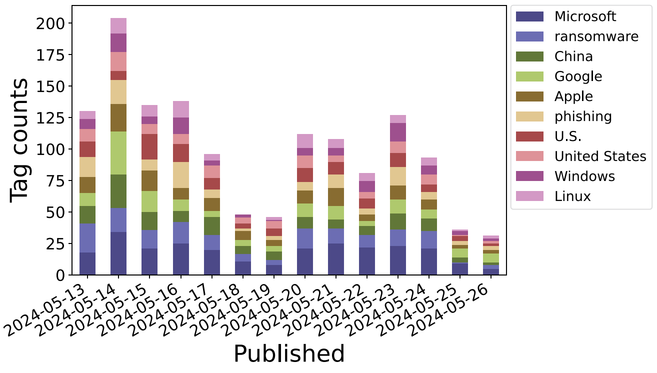

Fig. 3. Distribution of the 10 most common tags.

ST, MT, DST (see Sect. 4). For this, we use the first two weeks of the provided dataset as described in Sect. 5.1. Furthermore, the provided dataset does not include special sources such as arXiv or daily summaries (R2). Due to the lack of clustering benchmarks for OSINT, we perform a user-based evaluation, which allows for the assessment of the optimal hyper-parameter configuration as well as the overall quality of the clusters.

To identify the best configuration, we ran the initial clustering and incremental clustering for each $\texttt{Conf}_{\texttt{ST\_DST\_MT}}$, where $\texttt{ST} \in \{0.4, 0.45, 0.5, 0.55\}$, $\texttt{DST} \in \{0.2, 0.25, 0.3, 0.35\}$, $\texttt{MT} \in \{3, 5, 7, 9\}$. We initially observed that varying the hyper-parameter DST did not affect clustering results, indicating that ST (line 42 in Algorithm 1) plays a more critical role on this dataset. Thus, DST can be fixed, a random value 0.25 is used across all configurations, reducing the configuration space to 16 combinations of ST and MT. The next sections detail the configuration selection, clustering results, and user study design.

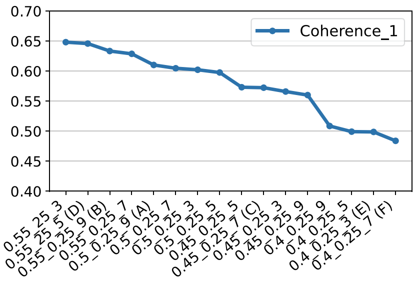

Fig. 4. Intra-cluster based coherence per configuration of the form ST_DST_MT.

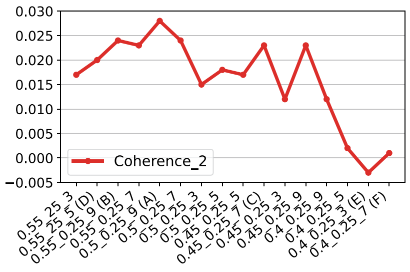

Fig. 5. Intra and inter-cluster coherence per configuration of the form ST_DST_MT.

Design of the User Study. In this section we present the analysis of the configurations and clusters obtained and the selection process for the user-based evaluation.

Selection of Configurations. Expert evaluation of all 16 remaining configurations is impractical due to the high number of resulting clusters. Therefore, we selected configurations for the user study based on coherence metrics. These metrics are calculated using cosine similarity between document title embeddings, applying the same sentence transformer model described in Sect. 4. We describe below the selection process:

(1) *Intra-cluster based coherence metric.* For each $\texttt{Conf}_{\texttt{ST\_DST\_MT}}$ we compute:

$$Coherence_1^{\texttt{ST\_DST\_MT}} = \frac{\sum_{i=1}^{n} intra\_sim_i}{n},$$

where n is the number of clusters obtained in the configuration[18], and $intra\_sim_i$ denotes the average pairwise cosine similarity of the titles of the documents in the cluster i.
The issue with this metric is that it does not account for inter-cluster relationships; therefore, a more comprehensive coherence metric is also used.

(2) *Intra and inter-cluster based coherence metric.* For each $\texttt{Conf}_{\texttt{ST\_DST\_MT}}$ we compute:

$$Coherence_2^{\texttt{ST\_DST\_MT}} = \frac{\sum_{i=1}^{n} intra\_sim_i - inter\_sim_i^k}{n},$$

where n is the number of clusters in the configuration[19], $intra\_sim_i$ is defined as above, and $inter\_sim_i^k$ is defined as the average cosine similarity between each item in cluster i and the nearest k news items from other clusters, averaged by the number of items in cluster i. In our experiments, we used $k = 5$[20].

Figure 4 and Fig. 5 display the results of these two metrics. Based on these results, we ranked all configurations using both metrics and selected configurations that performed well, moderately, and poorly across all configurations. Additionally, we considered the importance of selecting different values for hyperparameters `ST` and `MT` to ensure diverse samples and enable meaningful comparisons during evaluation. The resulting list of configurations selected in the evaluation is presented in Table 2.

Selection of News Aggregates. For the selected configurations, we take into account the *Coherence₁* metric value for each cluster in the configuration. Therefore, for each configuration, based on this metric we split the clusters into those

[18] We discarded here the clusters with only one item.
[19] We allow here also clusters with only one item.
[20] Commonly used neighboring size value in clustering approaches.

Table 2. Configurations selected for evaluation based on performance across both coherence metrics.

Performance	Configuration	Value of `ST`	Value of `MT`
Good	Conf A	0.5	9
	Conf B	0.55	9
Mixed	Conf C	0.45	7
	Conf D	0.55	5
Poor	Conf E	0.4	3
	Conf F	0.4	7

with high coherence (≥ 0.6), middle coherence (≥ 0.4 and < 0.6) and low coherence (< 0.4). The goal was to select two non-trivial representatives per category in each configuration. Given the large number of resulting clusters in each category, the next selection criteria involved identifying clusters that were updated during incremental clustering, multilingual clusters (R2) and clusters with more than two items. Furthermore, to identify semantically overlapping clusters, we plotted all clusters for each configuration using Uniform Manifold Approximation and Projection (UMAP) [5] to reduce the dimensionality of the title embeddings. These embeddings were obtained using the same sentence embedding model as applied to the titles in the clustering process (see Sect. 5.2). Based on the visualization, we could identify overlapping (semantically close) clusters which were prioritized in the selection to assess the quality of non-trivial clusters. After this specialized selection, we choose two representative clusters per category, (i.e., 6 per configuration containing a balanced sample w.r.t. coherence). Due to updates during incremental clustering, some clusters appear in both initial and updated forms. As a result, users can evaluate 8 news aggregates for Conf A and Conf F, 7 for Conf B, and 6 each for Conf C, Conf D, and Conf E.

Evaluation User Interface. For the analyst, the distinction between initial and incremental clusters is irrelevant, as the focus is on optimizing analysis time regardless of the cluster type. Therefore, we present a list of all the selected configurations. These configurations along with their containing news aggregates are presented in a user interface (UI). The UI is implemented using Streamlit[21] and visualizes the news articles (title and link) and the tags for each cluster. Additionally, each user can rate the news aggregates based on their semantic similarity. A news aggregate is rated as good (5) if all its items are closely related (e.g., describing the same vulnerability). Conversely, a news aggregate is rated as poor (1) if its items are totally unrelated. Furthermore, the users can rate, on a scale from 1 (poor) to 5 (good), how accurately the tags provided for each news aggregate represent the news items within it.

[21] https://streamlit.io/.

Evaluation Setup. Our goal is to evaluate the impact of our clustering solution on improving the efficiency of OSINT analyst in processing recently gathered news articles (R1). For that, we asked in total 15 OSINT specialists to evaluate the selected news aggregates in terms of semantic relatedness and tags coverage. To minimize the workload for each user, we split them into two groups: one group evaluating Conf A-Conf C, and the other evaluating Conf D-Conf F. The groups are balanced in terms of their level of expertise. Furthermore, the users were not informed about the coherence metric results to avoid bias. In addition to the rating options described above, we allow users to rate the entire configuration and provide comments that can help interpret the results and suggest improvements.

5.3 Results

Table 3 summarizes the evaluation results obtained from security experts. In the following, we analyze these results to evaluate the quality of the tagging and the clustering approach, and to examine the impact of hyper-parameter values on the clustering outcome. Furthermore, we compare the user-based results with those based on coherence metrics.

Tagging Quality. The results in Table 3 indicate that, according to security experts, the tags were considered, on average, to be moderately effective in representing the articles within nearly all news aggregates across all configurations. As a result, the tags were rated as not particularly informative in fully describing the story conveyed by the items in the cluster. However, users noted in comments that aggregate tags generally cover most articles, suggesting usefulness for search but limited topic representation. Since Taranis AI assigns tags immediately after collecting them and SC4OSINT merges them when aggregates are created, maintaining a robust tag set is essential for keyword community detection—removing tags would harm incremental clustering accuracy. The roughly 1-point standard deviation in ratings aligns with evaluators comments. Given that most tags denote broad categories (e.g., countries, organizations, persons or “MISC”), SC4OSINT’s strategy of prioritizing tag types and applying cosine similarity to form sub-clusters is crucial for sharper story-based clustering.

Semantics Quality. In terms of semantical relatedness, which refers to how well our solution SC4OSINT clusters semantically related articles, the overall semantics rating across all configurations is above ≥ 3.2 and thus *fair* to *good* (see Table 3). When examining the standard deviations of the average cluster semantics, it is evident that the range of ratings varies in each configuration. This indicates that the experts’ understanding of a good cluster differs. Based on the size of the news aggregates, some users commented that they had more difficulties in evaluating larger news aggregates, stating that the items are only loosely related. As a result, these larger clusters were generally rated lower. This indicates that security experts are more focused on specific topics represented by smaller clusters rather than more general stories, which aligns with the comments

Table 3. Overall evaluation results. For each configuration, the number of evaluators, number of news aggregates (NAs), the average overall rating for the configuration, the average rating for news aggregates semantics and the average rating for news aggregates tags are presented. In parenthesis, the sample standard deviation average is presented for each metric.

Config.	#Evaluators	#NAs	Avg. overall rating	Avg. NAs semantics	Avg. NAs tags
Conf A	6	8	3.16 ($\pm$0.75)	3.29 ($\pm$1.13)	3.00 ($\pm$1.08)
Conf B	6	7	4.00 ($\pm$0.00)	4.19 ($\pm$0.61)	3.80 ($\pm$1.16)
Conf C	6	6	3.33 ($\pm$0.44)	3.44 ($\pm$0.77)	3.02 ($\pm$0.88)
Conf D	6	6	3.83 ($\pm$0.75)	3.88 ($\pm$0.87)	2.97 ($\pm$0.97)
Conf E	6	6	3.00 ($\pm$0.63)	3.30 ($\pm$1.11)	2.80 ($\pm$0.75)
Conf F	6	8	2.83 ($\pm$1.22)	3.14 ($\pm$0.91)	2.43 ($\pm$0.95)

provided by the users. Furthermore, this is supported by the fact that Conf B and Conf D, with the highest `ST` ratings, outperform Conf E and Conf F, which have the lowest `ST` ratings (i.e., higher `ST` leads to more fine-grained clusters).

Parameters Analysis. Examining each configuration represented in Table 3 individually, Conf B is evaluated as *good*, compared to the other configurations, in terms of overall rating and the average ratings for news aggregates semantics. In depth, all clusters with Conf B had an average semantic relatedness rating of over 3, with 5 out of 7 clusters scoring above 4.3. When examining the standard deviations of the average overall rating, it is evident that all experts rated Conf B identically. This is noteworthy, as Conf B is evaluated as the best overall configuration. In contrast, the range of ratings for the other configurations varies.

As outlined in Sect. 5.2, the configurations include different values for the `ST` and `MT` hyper-parameters. When observing the influence of the `ST` hyper-parameter, higher values (i.e., 0.55) result in better outcomes compared to lower values (i.e., 0.4), as evidenced by the better ratings of Conf B and Conf D over Conf E and Conf F. The overall rating of Conf A is lowered by two poorly rated news aggregates containing daily summaries. As noted by users and in line with R2, such articles should not be clustered, as they increase analysis time. This underscores the importance of expert evaluation and careful configuration selection to meet the requirements in Sect. 3. Notably, excluding summary clusters raises Conf A's average semantic rating to 3.66, making it the third-best configuration and further supporting the relevance of the `ST` hyper-parameter. The `MT` hyper-parameter primarily filters out documents with few tags, reducing the number of clustered items, but seems to have less influence on clustering quality than `ST`; its optimal value range remains inconclusive.

Coherence Metrics. Table 3 shows that security experts rated configurations Conf B and Conf D with the highest scores, while configurations Conf E and Conf

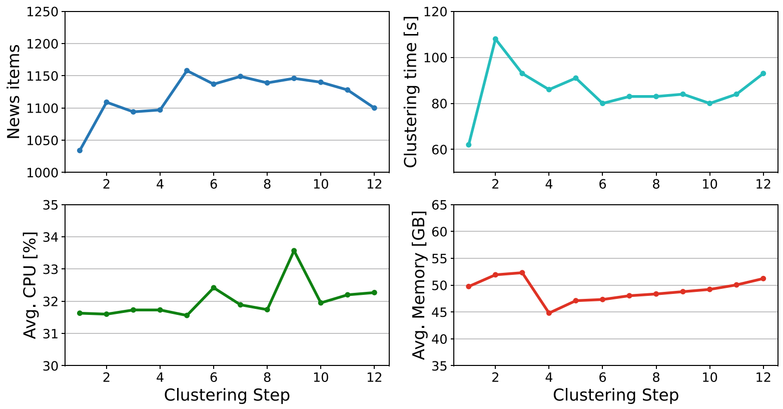

Fig. 6. Clustering performance analysis.

F received the lowest. When examining both coherence metrics shown in Fig. 4 and Fig. 5, it is evident that Conf B and Conf D also achieve higher scores than Conf E and Conf F. Based on this observation, it seems that $Coherence_2$ metric better distinguishes between *good* and *fair* configurations by considering inter-cluster similarity, revealing whether items in a cluster may belong elsewhere—a key factor in evaluating OSINT clusters. In future work, such metric can support expert evaluations due to the scarcity of available security experts.

Performance Analysis. Taranis AI was deployed using Docker (version 27.5.1) on a virtual machine (VM) with 64GB of RAM, an Intel© Xeon© Gold 6342 CPU (4 cores), and Ubuntu 24.04 LTS as the operating system. SC4OSINT was executed every 8 h on the deployed Taranis AI instance. SC4OSINT took into account the last 7 days of data (R6) in each clustering step regardless of whether initial or incremental clustering is applied. Figure 6 displays various performance metrics of 12 clustering steps. The first step represents initial clustering, followed by eleven incremental clustering steps. Incremental clustering requires more time than initial clustering, which is, however, not reflected in average CPU and memory usage. Further, no significant correlation is observed between the number of news items clustered and the other metrics.

6 Conclusions

Supporting security analysts in handling the volume of OSINT data is a major challenge. Automatically clustering news by story can boost efficiency, but existing NLP-based methods often rely on domain-specific labeled data, which are unavailable in the OSINT field. This paper presents SC4OSINT, an unsupervised story clustering approach tailored for OSINT streaming data. Inspired by existing methods but adapted to meet specific OSINT requirements, SC4OSINT also introduces a novel evaluation dataset for future research. A user study with security experts shows promising results, with most news clusters rated fair to good, and the best configuration scoring 4/5 in semantic similarity.

Acknowledgment. Funded by the European Union (EU) under the European Defence Fund (GA no. 101121403 - NEWSROOM and GA no. 101121418 - EUCINF). Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the EU or the European Commission. Neither the EU nor the granting authority can be held responsible for them. This work is co-funded by the Austrian FFG Kiras project ASOC (GA no. FO999905301).

References

1. Taranis AI (2024). https://taranis.ai/
2. Akbik, A., Bergmann, T., Blythe, D., Rasul, K., Schweter, S., Vollgraf, R.: FLAIR: an easy-to-use framework for state-of-the-art NLP. In: NAACL 2019, pp. 54–59 (2019)
3. Blondel, V.D., Guillaume, J.L., Lambiotte, R., Lefebvre, E.: Fast unfolding of communities in large networks. J. Stat. Mech. Theory Exp. **2008**(10), P10008 (2008)
4. Chowdhary, K.R.: Natural Language Processing, pp. 603–649. Springer, New Delhi (2020)
5. Härdle, W.K., Simar, L., Fengler, M.R.: Uniform manifold approximation and projection, pp. 581–595. Springer, Cham (2024)
6. Kuehn, P., Kerk, M., Wendelborn, M., Reuter, C.: Clustering of threat information to mitigate information overload for computer emergency response teams. CoRR arxiv:2210.14067 (2022)
7. Liu, B., Han, F.X., Niu, D., Kong, L., Lai, K., Xu, Y.: Story forest: extracting events and telling stories from breaking news. ACM Trans. Knowl. Discov. Data **14**(3), 31:1–31:28 (2020)
8. Ma, C., et al.: FineCTI: a framework for mining fine-grained cyber threat information from twitter using NER model. In: 2023 IEEE 22nd International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom), pp. 531–538 (2023)
9. Martins, C., Medeiros, I.: Generating quality threat intelligence leveraging OSINT and a cyber threat unified taxonomy. ACM Trans. Priv. Secur. **25**(3), 1–39 (2022)
10. Rajaraman, A., Ullman, J.D.: Data Mining, pp. 1–17. Cambridge University Press, Cambridge (2011)
11. Rani, N., Saha, B., Maurya, V., Shukla, S.: Ttpxhunter: actionable threat intelligence extraction as TTPS from finished cyber threat reports. Digit. Threats: Res. Pract. (2024)
12. Riebe, T., et al.: CySecAlert: an alert generation system for cyber security events using open source intelligence data. In: Gao, D., Li, Q., Guan, X., Liao, X. (eds.) ICICS 2021. LNCS, vol. 12918, pp. 429–446. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-86890-1_24
13. Sayyadi, H., Raschid, L.: A graph analytical approach for topic detection. ACM Trans. Internet Technol. **13**(2) (2013)

14. Skopik, F.: Collaborative Cyber Threat Intelligence: Detecting and Responding to Advanced Cyber Attacks at the National Level. CRC Press, Boca Raton (2017)
15. Skopik, F., Akhras, B., Woisetschläger, E., Andresel, M., Wurzenberger, M., Landauer, M.: On the application of natural language processing for advanced OSINT analysis in cyber defence. In: Proceedings of the 19th International Conference on Availability, Reliability and Security. ARES 2024. Association for Computing Machinery, New York (2024)
16. Sun, N., et al.: Cyber threat intelligence mining for proactive cybersecurity defense: a survey and new perspectives. IEEE Commun. Surv. Tutorials **25**(3), 1748–1774 (2023)
17. Trummer, I.: From BERT to GPT-3 codex: harnessing the potential of very large language models for data management. Proc. VLDB Endow. **15**(12), 3770–3773 (2022)