13. Graph-Based Intelligent Cyber Threat Detection System

Julien Michel*, julien.michel2@etu.unistra.fr; Pierre Parrend, pierre.parrend@epita.fr
ICube, UMR 7357, Université de Strasbourg, CNRS, 67000 Strasbourg, France.
Laboratoire de recherche de L'EPITA (LRE), 94270 Le Kremlin-Bicêtre, France.

Abstract - Attackers are actively considering machine learning-based approaches to avoid being detected. Classification models for attack detection are primarily composed of feature-driven algorithms. Thus, primary features which are individual dimensions in the original attributes of data in the input space are a prime target to compromise an AI-driven model. Additionally, adversarial examples have shown that an attacker does not need to have knowledge of detection criteria to compromise a detection model, even in the case of a black box model. Attacks behavioural changes cause features from attacks datapoints to be altered and detection performances to drop. Thus, robust features must be engineered to prevent models from being compromised in such manner. Graph-based feature engineering has recently shown promising results considering robust threat detection. We offer an overview on methods for graph-based feature extraction and explain why they are relevant to robust feature engineering for threat detection purposes. We detail what we believe are properties for feature space to be sustainable and efficient for their prolonged exploitation in security operating centres. Specifically, we provide key criteria for the robustness of a feature space for attack detection. Finally, we summarize the characteristics for time robust feature selection, identify current limitations specific to the distinctive type of graph-based approaches in the purposes of threat detection in large internet networks.

Key words: Feature engineering – Graph representation – Features quality – Scalability – Time Robustness – Explainability

13.1 Introduction

As the atomic input element to Machine Learning (ML) algorithms, features are the foremost parameters when it comes to detection problems, both for supervised – like classification of known attacks – and unsupervised – like statistical anomaly detection of suspicious behaviours – models. Therefore, feature engineering is a crucial part of a detection system. This is especially true for threat detection purposes, considering big and critical Internet networks where data collection throughput can raise to Terabytes per minute. Operators in security centres monitoring such networks must ensure they take the right decision. Consequently, the information and model predictions must reach them as fast and as clearly as possible. Therefore, feature engineering techniques should be scalable. Features must be explainable and time robust.

The challenge is to find what make a feature engineering scalable while producing explainable feature for time robust classification of threat. This raises 2 main research questions: 1) Which are the characteristics to make features explainable and time robust in the context of large internet networks? As graph structures are very representative of the actual fine-grained network behaviour. 2) How are graph-based approaches efficient as a support for feature extraction and which type of graph approaches are more relevant and performant for threat detection purposes?

13.1.1. Threat detection

Threat detection is the detection of any element that could compromise and cause damage to an information system [52][25]. In our context, we consider the system to be a large network and consider as threat any elements in the data space that would hinder the operation process of machine and service in the network it does not have an authorisation to access to (attacks against availability and integrity of the systems), or that would unlawfully disclose information (attacks against confidentiality) [51]. The main objective of the threat detection system is therefore to identify, then stop, any behaviour that would hinder the "normal" operation process in the network while not itself hindering it [16].

However, threat behaviours are becoming more complex and continuously adapt to defender models [57]. Additionally, they do a better job at hiding themselves and it becomes increasingly time costly for defence operators to manually detect attacks. As such there is an intense pressure for scalable automatization of detection and classification of threats, which also implies a strong limitation of false positives alerts to limit inefficient manual verifications [36]. These classification algorithms must be efficient in their discrimination of threats while scalable, explainable and time robust.

13.1.2. Feature engineering

Feature engineering is the full process between data collection and the execution of detection algorithms [13]. It includes all transformations in the original feature space like normalisation, cleaning, categorisation or encoding, as well as the derivation of new features from original features, like relative values, distances. For example, typical derived features in network datasets would categorization of port or IP addresses, thresholding on size of packets, or ratio between number of packet and total message size [38]. The last step of feature selection is the ablation of feature from the feature space to reduce the search space, optimise analysis time and remove noisy features that lower detection capability. The feature space for detection is without

doubt the most crucial factor in any detection system as any properties or constraints that are not respected by the feature engineering process will not be respected by the detection system as a whole [62]. It is even more important considering that having an efficient feature engineering process will lead to the possibility of having more diverse choice in classification algorithms while retaining high detection performances [32]. Additionally, with regards to threat detection and more particularly in the presence adversarial actors, each feature is a potential vector of vulnerability and as such having an optimal feature space is crucial.

Graph based representation are closely related to network behaviour. As such it is expected that they could lead to explainable approaches with regards to threat detection in internet networks [49]. Graph based representations, especially unattributed connectivity graph that only rely on the topological aspects of the network are particularly relevant to the last raised point as they require and depend on a minimal amount of feature in the original feature space [46].

13.2 State of the art

13.2.1. Methodology

In this study, we opted for a methodology that would allow us to focus on the property for feature engineering in the context of threat detection concrete issues. As a starting point, we identify the key properties a detection system should strives for to be operable in a trustworthy and sustainable manner. From these identified properties, we analyse the current trend in recent research works, what challenges have been identified in the literature and what are the feature engineering approach based on graph that are trying to answer them. From those properties and challenges, we identify their research goals about specific detection problems and formulate criteria for feature engineering in their realisation. The scope of this study on threat detection includes works relative to the definition of the relation between features and the identified properties: Scalability, Explainability, Quality, Stability, Time Robustness. We than analyse graph-based feature engineering techniques and how they take, or do not take those properties in consideration. For each of those properties, we provide definitions from the literature and determines how the property is considered important to threat detection by the literature. We then make statistics on the number of studies addressing the issues of interest in the literature to the subjects of this paper, namely: threat detection, which is the main objective, feature engineering, which is the mean we use to attain the objective and graph representation, as a support of the feature engineering. We take interest in the papers that intersects those subjects between 2019 and June 2024. To better understand the place of feature engineering and graph

representation in threat detection, we used Google Scholar to search for paper including our keywords: explainability, scalability, feature stability, feature quality, concept drift, threat detection, feature engineering and graph representation. For each of the keywords we obtain the number of research paper including them. We repeat this process with intersection of the different properties with research paper on the threat detection topic and then compare general trends in the presence of the keywords in paper topics and how often they are discussed in a single research paper. Additionally, we consider the proportion on feature engineering and graph representation research paper in the topic of threat detection.

13.2.2. Prevalence of feature engineering and consecutive learning properties in the literature

We enounce in this sub-section a brief overview on the state of the art for the considered key property in the context of threat detection. We rely on google scholar search for an estimation of the number of papers on those topics and think the tendencies we can observe to be informative to have a general idea on the context on threat detection and its relation to feature engineering and graph representation.

Figure 13.1: Proportion of Paper on Threat Detection Including Graph Representation or Feature Engineering between 2019 and June 2024

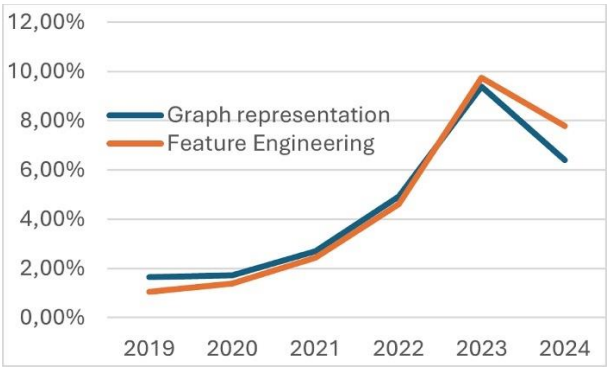


Figure 13.1 shows the proportion of papers in the threat detection domain that include graph representation or feature engineering as one of their topics. Both have seen a growth of more than 300% between 2019 and 2024, with a spike in 2023 where research paper on threat detection including feature engineering or graph representation represented respectively 9,74% and 9,37% of paper on threat detection. *W*e can observe that paper on threat detection have a similar evolution in their intersection with the topics of graph representation and feature engineering. The similarity in their growth could be related to the fact that they give tools for similar objectives of current landscape of the threat detection system.

Figure 13.2: Research Papers on Threat Detection Including Explainability and Scalability between January 2019 and June 2024

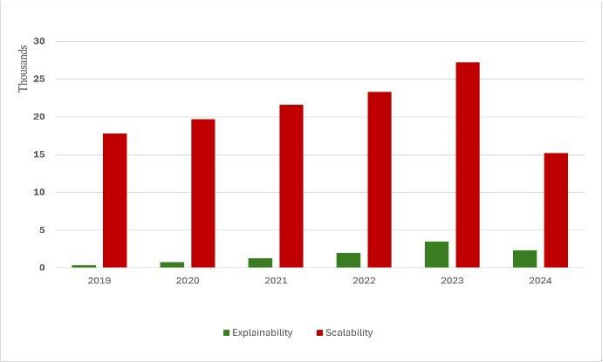
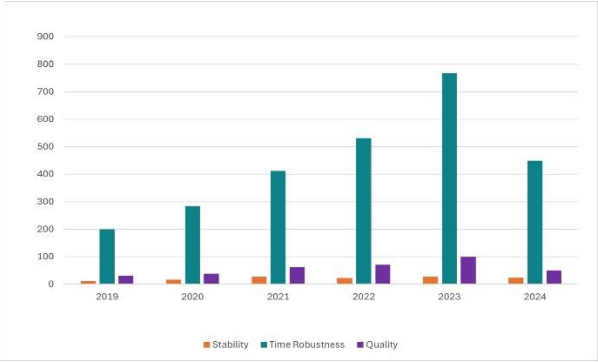


Figure 13.3: Research Papers on Threat Detection Including Feature Stability/Quality and Time Robustness between January 2019 and June 2024



As can be seen on *Figure 13.2* and *13.3*, all the properties considered have seen a growth in the number of papers between 2019 and June 2024 in the domain of threat detection. However, those properties are not equally spread, while in threat detection, scalability paper is considered in more than 27000 papers in 2023, explainability is considered in 3500 papers. Feature stability, feature quality and time robustness are respectively considered in 27, 99 and 768 papers in 2023. Over the five year all the properties have seen a growth in the number of paper and if the number of papers on those properties over 2024 remain constant we expect respectively 30400, 4720, 898, 100 and 46 papers for scalability, explainability, time robustness, feature quality and feature stability in the topic of threat detection. Such difference is not surprising as those properties have not the same scope, nor relation to end point objectives. Scalability is often a requirement for a sustainable solution, while explainability is a desired properties that can be observed in a system by the users. Time robustness is property of system to its sustainability over time, while feature quality and stability are property to attain time robustness and explainability while ensuring scalability.

Graph representation and feature engineering are both topics that are being more considered for the detection of threat in the recent years, both seems to be important for more robust threat detection: feature engineering as the mean to ensure scalability, explainability and time robustness and graph representation as the support of feature engineering.

13.3 Key properties for features in threat detection

In this section we introduce what we defined as five key-properties for feature engineering in threat detection. We divide those in two categories: the first category are the properties that not tied to features but should be objectives for a threat detection approach to thrive, scalability, explainability and time robustness. We think those properties to be especially important consideration for any trustworthy and sustainable threat detection system. The second category

are the properties which are only related to the feature in the detection system, the feature quality and stability. While not totally disjointed, those properties are still different from one to another, and are a factor of utmost importance in ensuring the three previous properties.

13.3.1. Scalability

Scalability of a system is its capacity to function properly with an expected computational workload and within expected margins with future workloads. In the context of detection, scalability is the capacity to produce a prediction or decision under a time constraint considering a potentially higher volume of data. Thus, the importance of scalability of a system is directly related to its task. When working with increasing numbers of objects, it is required for a system to be scalable. Sustainability of such systems requires their scalability to maintain the quality of service. Eventually, if scalability is not insured, sustainability can be compromised, and the system must be replaced, leading to miscontent from users, new production costs or security issues [23]. Scalability is mostly constrained by the time of computation and the memory space, but in some cases can depends on structural design, for example with the limited number of IPv4 addresses. A system scalability is tied to its worse sub-process scalability. For a detection pipeline, it will be the step in the pipeline with the worst scalability.

As such in a threat detection system, feature engineering must be scalable. Threat detection requires handling of volumes of communication data which are ever growing with passing time [48]. Moreover, data are becoming increasingly diverse with several types of structure and structural constraints. Thus, feature engineering techniques should consider time, space, and structural complexity to ensure operability of the threat detection system [1]. Threat detection environments data are often subjected to a high collection rate up to Terabytes of data per minutes for big network [65]. Therefore, the time constraint is especially strong when considering those type of networks and security operating centre scenario which require prediction in less than a minute. Therefore, scalability is a main concern to threat detection and by extension to feature engineering in this context. With regards to scalability, the main challenge identified by the literature for threat detection are scalable model that retain high detection performance, scalable feature engineering for high detection performance and scalable data structure for scalable feature engineering.

13.3.2. Explainability

Explainability for an AI-based system is defined as the capacity for each part of this system to provide an explanation for its prediction, all the parameters used in the detection system, and

the actions it has taken. It is a main concern in AI-driven solution as it is difficult to have a complete comprehension of machine learning model decision [8]. A system having a higher explainability level should be more trustworthy as the users would be reach an understanding of decision made by the system, and as such would have a better useability [50]. The main factor for this better useability is the capacity of the user to determine if using the AI-model results and his understanding will yield a better decision than his own. Thus, performances are a critical issue when considering explainability in AI-driven systems [15]. However, there is currently a lack in our capability of evaluation or quantification of explainability, therefore quantifying the relation between performances and the different layers of explainability an open issue [59]. Nevertheless, explainability has a major place in the current AI-driven detection landscape as it can be used to prove your detection system respect ethical concerns [30].

In the threat detection domain, explainability is especially important as trust is a main issue in the detection of attack as any attack detected by a system with low trustworthiness will be as good as not [1]. Attacks must be ensured to be detected efficiently. False positives have a remarkably high impact because they will lose time to operator in security centre to analyse the alert or interfere with the quality of services for certain users if an action is mistakenly taken for the false alert [53]. However, there is an interesting concern about explainability for AI detection models and their interaction with adversarial models. While it is quite known that adversarial example can produced for black box model, it can be thought that having more explainability in a model or its features could yield more adversarial example. In fact, recent works have shown that using feature explainability it is possible to detect which feature could be a liability in consideration to adversarial models [24]. Main challenges considered by the literature for the explainability of model are on the evaluation of the explainability of a model and about the eventual trade-off between explainability and detection performances.

13.3.3. Time Robustness

A detection system is time robust if its detection performances are not compromised over time, therefore it is robust to concept drift. Concept drift is a problem by its unpredictable aspect and the diverse aspect of changes it encompasses. Additionally, changes in detection target and in the detection environment, i.e. the data that are not supposed to be a detection target, are both important concepts. A concept is defined as an event with a certain probability of appearance in an environment [61]. In this context, concept drift is not only the change in the behaviour of a concept, but additionally how the distribution of concepts evolves in the environment and how

some concepts disappear, or new concepts appear. This statistical definition of concept drift leads to the detection of concept drift properties as a mean to characterize it: its time of occurrences, its severity, and its distribution at a given time [42]. Concept drift as such is also problem to consider for unsupervised detection, but it also a mean to detect unexpected behaviours such as new potential threat [19].

In threat detection, concept drift is especially important as change in environment of detection can lead to an increase in the number of false alarms and leads to an unsustainable detection system for the detection of attacks in data streams [22]. Additionally, changes in detection target will lead to a decrease in the detection of previously seen threats, meaning the detection system will lose gradually its efficiency if nothing is done [37]. Features can have a varying robustness to concept drift for the detection of threats, thus feature engineering is vital in producing AI-based model time robust [18]. Feature engineering can even go a step further in its consideration of concept drift, with evolving feature set which react to concept drift in the data stream analysis [11]. A shift in the data profile is detected, feature previously selected that are submitted to shift are reevaluated for selection with the objective for the selection to better correspond to current data profile. The research challenges identified in the literature with regards to time robustness are the detection of shift in a data profile, the extraction of feature for a time robust feature space and the automatization of update for feature space in consideration to concept drift.

13.3.4. Feature Quality

Feature quality is a characteristic which is inherent to a specific detection purpose. Depending on the detection objectives such as false positive rate optimisation or overall performance, optimisation a single feature quality can vary greatly. To determine how qualitative a feature is, there is a strong need to understand all useful information it bears for the chosen purpose [40]. Additionally, a feature quality regarding a detection of a specific target on a can differ depending on the considered feature set, since information from distinctive features in the same set can overlap. Feature engineering is relevant for maximisation of the quality of a feature when considering feature quality because the quality of an engineered feature can be higher than cumulated quality of the source features [45]. For example, you can have distinctive features that have a range of value which are all evenly distributed when looking at them and the different classes to detect. As such the quality of those single feature would be low. But then you could notice that by crossing them the distribution is not even with the classes to detect, then resulting in a feature of higher quality. Having features that do not contribute to the detection can be very detrimental to AI-based detection models, adding them to the feature set

would make a drop in the detection performances. As such feature quality can be crucial in assembling a purposeful feature set [41], and to select the right features depending on specific objectives such as the detection of a particular target classes or lowering the false positive rate.

The relation of feature quality to threat detection is dual. Firstly, in term of detection performances, having a better quality of feature leads to better results as we would keep only features that are beneficial to the detection performances. This is supposedly due to having feature more closely related to physical or digital reality [3]. Secondly, having a better feature quality for dataset could lead to an overall better dataset quality [47]. While the quality of a dataset is not only tied to its features, and those features do not have a direct influence on the general behaviour of the data in the dataset, they are the main interface between the data and the threat detection tools [56]. The main challenges with regards to feature quality according to the literature are the evaluation of feature quality and the evaluation of the impact of feature quality to detection performances.

13.3.5. Feature Stability

Feature stability is a property of features which suffers from a reduced consideration in the literature being twice as less present in research paper in the 5 last year than feature quality. However, it is tightly linked to explainability, as having a feature not stable would mean the information it brings is not stable, feature quality, as it is constant if the feature is stable, and vary if it is not, and time robustness, as if the whole feature set is stable than you are not subjected to concept drift anymore [58]. It is defined as a measure of the robustness of the feature, i.e. considering the whole dataset, how relevant the feature is to the detection objectives, such as the optimisation of true positive rate for binary detection [4]. Depending on set conditions, it is possible to determines different values of stability, using different feature sets or aggregates which can be relevant for example to detect cyclo-stationarity. Empirically a more stable feature should be more qualitative, more explainable, and eventually more time robust. Additionally, when considering feature stability for feature selection, it should result in a more stable feature selection process as you would not need to reconsider stable features [28].
This last observation holds true for some threat detection issues like phishing detection [5]. Moreover, there is another form of information that can be extracted from feature stability and is especially important in threat detection which is how will the feature behave when there is a shift in the trend of the data. Threat detection environments are very subjected to change in the values of the feature space over time in both the general environment and the targets to detect in this environment, i.e. the concept drift [69]. In this context, what is of interest is not the

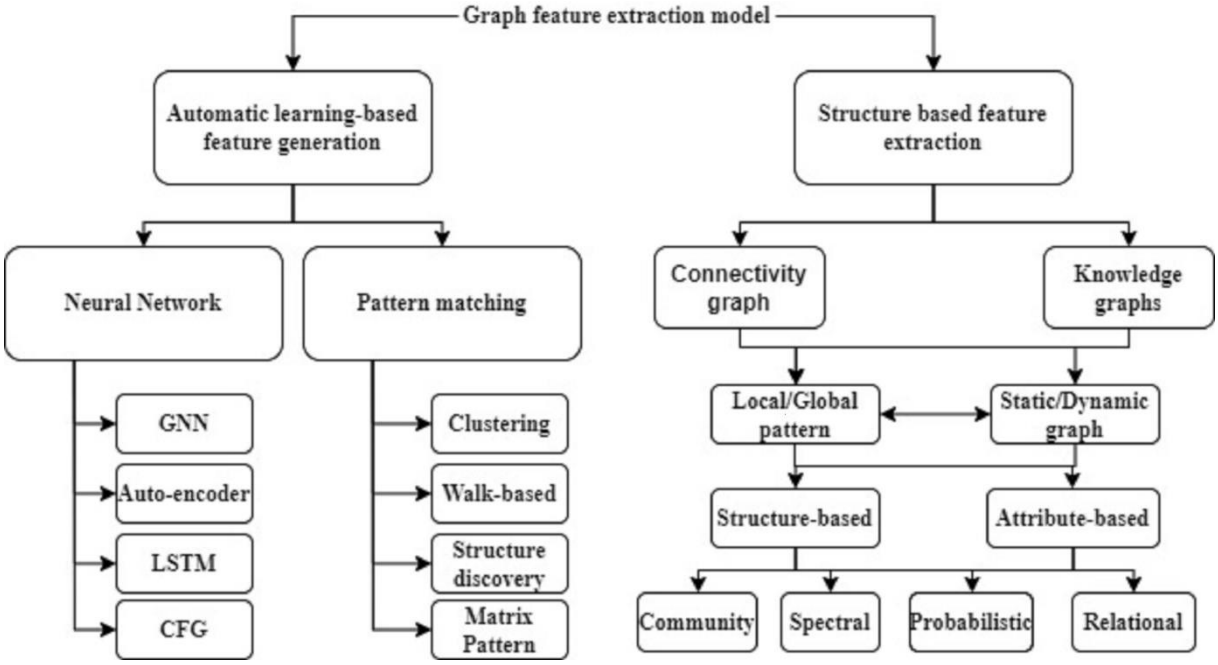
general stability of the feature, but how a robust or stable feature can maximise stability for the environment to differentiate threat from normal behaviours. Generating or extracting features which are stable in consideration to evolution of data with passing time is a key property for a time robust threat detection and the concept drift of the data [6]. The main challenges considering feature stability for threat detection identified in the literature are how correlated is a feature space stability to time robustness and how to ensure feature stability with passing time.

13.4 Graph-based Approaches

In this section, the different kinds of graph-based approaches for feature engineering we have considered are detailed. We give a general idea of their exploitation for threat detection purposes and point out their advantages and limitations.

13.4.1. Landscape of graph-based feature engineering

Figure 13.4: Graph Based Feature Generation Taxonomy



The main objective in feature engineering is to render new angles of information accessible and operable for a specific purpose. Graph-based feature engineering techniques have thus evolved to open the access to information that was not available beforehand. The main point of graph representation is the capacity of showing behaviour of interactions between distinctive objects in the data space [12]. Additionally, it leads to a multitude of graph representations, themselves leading to distinct types of structures on different scales, temporality, information layers and means of accessing that information. These graphs representations type and different possibles processing operations are interchangeable in their association. We propose in our taxonomy of

feature generation techniques (*Figure 13.4*) to make a distinction between automatic feature generation, i.e. feature generated by a learning model, and more classical feature extraction methods as they usually are both significantly different in the resultant features. They do however have a similitude in their purpose to benefit from data structure closely related to real-world structure like social networks or transportation networks for examples [44].

13.4.2. Types of metrics

In this section the different kinds of graph metrics, i.e. the parameter from graph that can be extracted as new features, are detailed. There are various means to produce feature from graph structured data [54], which depend on a range of factors that are detailed in the next sub-sections, where we explain advantages and limitations for each of them.

13.4.2.1 Global and local metrics

The first important parameter in graph feature extraction for threat detection is the locality of the metrics. Indeed, depending on their locality, metrics can be tied to vastly different threat behaviours [20]. The locality of a metrics depends on the objects needed to compute it. For example, a global metric is a metric that need to consider elements in the whole graph to be computed, at most it would refer to all the nodes and edges contained in the graph. Invertedly, the most local metrics would be information tied to a single node or edge. In general, either node or edge corresponds to a single data point in a tabular view, meaning such information are not related to graph topological behaviours. Thus, we in general a metric is local when it considers a single node or edge and its direct neighbours. There are multiple locality levels between local and global including connected component level and or any partition-based sub-graph level. Each locality has its importance when trying to detect specific threat behaviours as some threats could have a visible impact only considering a smaller part of the graph, while others could only be detected while looking at a bigger scale. Those graph topological behaviours are linked to spatial behaviours of the data, and actively link attack behaviours with specific elements in the graph. Thus, there is a need for any graph feature engineering for threat detection to determine which locality is relevant to its various threat detection purposes.

13.4.2.2 Dynamicity and temporality

While locality is important for its relation to the spatial behaviour of threats, attack behaviours are temporal events as well. Dynamicity is the parameter representing how behaviours evolve in a data structure. Graph structure can be adapted to show the appearance, disappearance, or transformation of events inside the structure. There are dynamic graph representations and there

exist multiple representations which can represent different behaviours. For example, a complete dynamic graph can be represented as a full spatial graph with additional temporal edges between nodes having the same identifier at different temporality. The temporality is the given time of an event for a given dynamic graph representation. In the case of a complete dynamic graph, it is adapted for representing the evolution of the behaviour of a single node. However, it would show severe drawbacks in term of scalability for the representation of the dynamicity of global metrics. For the same data we can represent dynamic graphs with different parameters for temporality. Absolute time can be the parameter for temporality, although usually data are divided into slices of time and the temporality is affected by time windows. Another mean of representing a dynamic graph is to make series of graphs divided by those time windows. Data are assigned to the graph which corresponds to their time window. In the dynamic graph, locality of the behaviour and evolution of those behaviour is considered to better represent and detect the spatiotemporal events. Different combinations of locality and dynamicity will lead to different metrics more representative of specifics threats.

13.4.2.3 Attributed graph

Nodes and edges in a graph can contain properties apart from their identifier. Those properties are named attributes. If a graph is made of nodes and edges without any attribute, it is said to be unattributed, otherwise this is an attributed graph. In the case of an unattributed graph, we are only able to compute metrics based on the topological aspect of the graph. The purpose of an attributed graph is to be able to build relations between objects in the graph based on their attributes. To compute the graph metrics, we add another constraint based on graph elements attributes. This way, we can add a bias correlated to the attributes to the topological metrics. However, while biases are necessary for detection, it can also be detrimental and can lead to overfitting for example. Thus, attribution in graphs should be thought carefully, as one of the purposes of graph representation is to be free of some data bias. We want graph representation to give detection criteria related to mandatory behaviours of specific threats while avoiding criteria related to behaviours of a specific threat on a specific period, but which could be easily modified at later times. Relying on more attributes lead to more leeway in the compromission of the time robustness of a model.

13.4.3. Learning-based approaches

As can be seen on *Figure 4*, learning-based approaches are detached from other types of approaches in our taxonomy. The principal reasons for this separation are the fact that learning

based approaches are mostly automatised and that the feature generation is directly tied to the model performances. We define by automatised the fact that before the computation of such feature generation model, the user has no prior knowledge of the features that will be extracted. While in more classical feature generation models, features to be extracted are defined and purposefully extracted, in learning-based approaches features happen to be extracted. There is a radical paradigm shift, leading to the second difference: instead of choosing features to be extracted in expectation to optimize model performances, we optimize the model in expectation of meaningful features. Thus, we make a clear distinction between those approaches in the taxonomy. Neural networks have shown to be particularly efficient for the genereration of feature for optimisation problems [10] and have been applied to threat detection models [17].

13.4.3.1 Graph Neural Network (GNN)

GNN are neural networks processing graph structured data. They are primarily used in the detection of elements or groups of elements in a graph, i.e. nodes, edges, sub-graphs or connected components. They are mostly applied to attributed graphs as a mean to extract information from the graph attributes. This capacity of association of graph attributes and topology has led to various works for the generation of features using graphs. The main advantage of the use of GNN is their capacity to aggregate the data from a graph structure automatically and efficiently [60]. They may however need more classical feature engineering in prior for specific use-cases. As most neural networks-based models, they have a high specificity leading to approaches of feature generation not extensible to any use beside the specific case they have been modelled for [31]. GNN have recently been applied to feature engineering for threat detection purposes [63].

13.4.3.2 Other Neural Networks (NN)

While less common for the extraction of features based on graph, other NN techniques have been applied to this purpose. For example, to manage dynamic graph, LSTM which is an autoencoder model based on recurrent neural networks (RNN) has been applied to generate features on temporal information in the graph and to assign them to static nodes [33]. The choice of LSTM compared to GNN is justified in the literature by a characteristic of GNN to over-focus on the topological aspect of the graph [27]. For threat detection purposes as well, like in the detection of specific messages in social networks we have seen use of NN as a mean to produce highly qualitative features [68]. Other types of RNN have proven to be efficient in engineering of features, notably applied on control flow graphs for unsupervised detection [43].

13.4.4. Topological based approaches

While not inherently unattributed approaches, topological based approaches primarily focus on the connectivity inside the graph structure. They are interesting for feature extraction purposes as they tend to be scalable for pre-defined tasks and shows prominent level of interpretability [21]. Since many threat detection problems involve structures that can be precisely represented by graph like social networks, end-to-end network or internet of things devices networks, their topological aspect can be highlighted in the graph metrics. Additionally, some approaches produce high quality feature representing global behaviours [21], while others can better represent local behaviours in the graph structure. Thus, a broad range of behaviours can be tailored for the detection of specific threats.

13.4.4.1 Probabilistic models

Probabilistic models may be the topological approach most closely related to learning based models as they produce feature spaces based on rules. They differ from learning based models from the sources of the rules as they are human made mathematical rules [44]. Some models make use of hierarchical structures in networks to compute probabilities of existence of edges between nodes in the graph. This is the link prediction. Instead of using this information as for link prediction, the probabilities are mapped to data in the feature space and can then be used by various learning algorithms. The major drawback of most of these models it that they are often based on Bayes rules and suffer from a serious lack of scalability. Another branch of probabilistic models are the stochastic models. The main advantage of stochastic models is that while they are automatable, they are highly parametrizable as well [29]. By their stochastic nature, the computation time is malleable and can provide an adaptable framework for the dynamicity of the data as they can update the feature space automatically.

13.4.4.2 Community models

Graph community structures are a mean to partition a graph into clusters using the graph topology itself as the only parameter. A node is part of community if it is more closely related to the node in its community compared to other nodes in the graph structure depending on the community partitioning criteria. The most common partition criterion is the maximization of the modularity. The appearance of community-based partition as another form of clustering comes from the realisation that for large networks, the clustering techniques were failing in distinguishing communities compared to the ground truth of the network-based datasets. Large networks tend to have a lot of noise, i.e. behaviour sufficiently different for being outlier, but

not significative while considering a threat detection purpose. As such classical clustering approaches tend to make small clusters out of all those small-scale outlier behaviours. On the other hand, those uninteresting behaviours are statistically over-present when compared to threat behaviours, rendering more the detection of threats more difficult in this context. However, specific threat behaviours have shown to be closely related in a graph structure. Hence, community structures have emerged to highlight those behaviours. Additionally, small-scaled attacks have a lessened impact while looking at whole graph metrics, whereas they are more impactful on a community structure. Community-based approaches also prove to be quite time efficient and more explainable as they can tie behaviour to areas in the graph structure [19].

13.4.4.3 Spectral models

Spectral models use the Laplacian matrix representation of a graph to extract features. Matrix representation for graphs can be very costly both in space and time complexity. Thus, spectral models are mostly applied on dynamic graphs, where the number of node and edge for each time window tend to be lower as activity on short period are more concentrated in the graph structure. Spectral models specifically access topological information inside the graph through the eigenvalues of the Laplacian Matrix. These values are the main interest of spectral models because they provide information on the graph structure in an instantaneous manner, such as the number of connected components corresponding to zero in the eigenvalues. Moreover, similarly to the community models, spectral models are independent from the original feature set of the dataset, i.e. they work well with unattributed graph. Thus, they are not as affected by noise and bias in the original feature space [67].

13.4.5. Relation-based approaches

Relational based approaches, while still relying on topological aspects of the graph structure are not dissociable from the attributed aspect. Indeed, those approaches are tied to heterogeneous graph structures, where nodes can be objects of several types and edges are the relation between those nodes. They can represent a relational database where any row in the database is a node and edge are foreign keys. However, a relational database is not required as input data [66]. As such, it is possible for relational based approaches to be quite scalable using relational database for data storage [9]. Additionally, by the nature of the relation between the different objects, features and rules generated are inherently explainable [64][14]. Relational graph structures can represent variety of types of data and prove to be efficient at modelling data with multitude of objects classes as videos or natural language texts [34]. This proves to

be particularly interesting in the detection of threats in a social network environment as they can make an efficient use of posts content [26]. Additionally, providing more explainable features give a more trustworthy base on approaches for threat detection [55].

13.4.6. Knowledge graphs

Similarly to approaches based on relational graphs, approaches based on knowledge graphs are indissociable from the knowledge graph structure. They present similarity to relational graphs, notably by the facts that they are heterogeneous graphs where nodes are from different classes of objects or concepts in this case, and the edges represent relations between the objects. The main difference between relational and knowledge graphs is that a knowledge graph can be refined. More precisely, generated features from the knowledge inside the graph will lead to further analysis which in return will feed the knowledge graph resulting in a new knowledge graph [7]. Not all approaches using knowledge graphs for feature engineering have a process to update the knowledge graph. However, this raises the critical point about knowledge graphs: elements in the knowledge graph do not need to exist in the original data to be part of the knowledge graph. While the nodes can represent existing objects, they can represent more abstract concepts. Additionally, knowledge graph-based approaches intend to be highly explainable representation and to perform efficiently together with tree-based learning models like random forest or XgBoost [39]. Knowledge graphs being structurally like relational graphs, feature engineering approaches are scalable, and this hold true for tree-based models.

13.5 Discussions

In this section, we detail our reflection on the observations while analysing the literature and shed light to the lack of consideration about certain areas of feature engineering for threat detection. We try to propose leads on points we think should be improved in the future.

13.5.1. Limitations of GNN and other NN approaches

Studies on GNN and other NN approaches tend to focus on the performance optimisation of specific models. As such, key-properties for an efficient feature engineering, independent of the learning model itself are rarely considered in this domain. While explainable GNN models exist, they are still exceedingly rare [35]. And since NN-based models are inherently less explainable, as a feature generation tool they produce features that are hardly interpretable. While for a specific detection process, it may be acceptable, it is completely unreliable for threat detection purposes as it is impossible to gauge the trustworthiness of a system based on those approaches. This is specially the case for prolonged detection over networks that are subject to concept drift.

NN models being extremely specific, they are particularly sensitive to concept drift and therefore are not time robust. If in addition they are not explainable, it is hard to detect the breaking point where the detection system will stop to function properly. NN models additionally suffer from problems linked to their exploitation. They are not scalable and as such cannot handle a big volume of data under time constraint. Additionally, they are not adapted to handle heterogeneous graphs or dynamic graphs. Moreover, they suffer from imbalance in the classes, which is inevitable as threats are in most cases a minority in the data space.

13.5.2. Use of attributes in graph feature engineering

Different approaches based on graphs for feature engineering use attributed graphs. While attributed graphs add another layer of information compared to unattributed graphs, the graph attributes represent either features from the original feature space or are derived from them. As such they suffer from part of the original feature space biases. While in a static context, the impact of this matter of fact could be minimal. In a detection environment subject to concept drift, this is crucial. This is a main argument for using graph representation for feature engineering. We want the features we extract from the graph to be representative of threat behaviours throughout the system life cycle and not of the behaviour the threat had at a specific time point. Behaviours issued from the original feature space are frequently those that could be easily modified by an attacker, and thus having a model that can avoid relying on these features produce more time robust predictions relying on more stable features. Additionally, relying on a lower number of features tends to make more time robust models.

13.5.3. Consideration of key-properties in current landscape

The number of research papers addressing properties of features are scarce, especially in the threat detection domain. While scalability, explainability and time robustness properties that are not directly tied to features are discussed, feature quality and stability are hardly considered, even in feature engineering focused works. These can be explained, as we could observe that in most works that claim to address feature engineering, feature engineering was in fact not the focus of the work. Feature engineering is a mean to obtain better detection performance. Current landscape of feature engineering for threat detection lacks means for the evaluation of the feature engineering and feature spaces. Decision systems, powered by AI or not, are feature-driven and ultimately the main parameter of those system are the features. They are the prime target for adversarial behaviours as well. Therefore, the quality of feature spaces should be ensured, and we should ensure their conformity to the threat detection purpose.

13.6 Conclusion and future works

In this paper, prominent graph representations and approaches for feature engineering purposes have been detailed with regards to threat detection and classification of attacks in large network environment. We synthetised definition for what we identify as key properties for feature engineering and robust threat detection, and analysed how they are considered in the current landscape of threat detection. We elicited the limitations of the current approaches for graph-based feature engineering while highlighting the relevant behaviour they display for time robust, scalable, and explainable detection, such as the minimization of original feature space as parameter for derived features, the assignment of local behaviour from graph structures to the data and smoothing of statistical anomalies in large network data environments.

For future works, we would like to evaluate concept drift robustness, while primarily looking for clues in identifying and evaluating criteria for having a time robust feature space. To this end we expect graph representation to produce stable and qualitative features.

13.7 Bibliography

1. W. Abdelghani, C. A. Zayani, I. Amous and F. Sèdes. 2019. *Trust Evaluation Model for Attack Detection in Social Internet of Things* (pp. 48–64).
2. M. Alhanahnah, C. Stevens and H. Bagheri. 2020. Scalable analysis of interaction threats in IoT systems. *ISSTA 2020 - Proceedings of the 29th ACM SIGSOFT International Symposium on Software Testing and Analysis*: 272–285.
3. A. Alhowaide, I. Alsmadi and J. Tang. 2019. Features Quality Impact on Cyber Physical Security Systems. *2019 IEEE 10th Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON)*: 0332–0339.
4. L. Al-Shalabi. 2022. New Feature Selection Algorithm Based on Feature Stability and Correlation. *IEEE Access*, *10*: 4699–4713.
5. L. Al-Shalabi and Y. Hasan Jazyah. 2024. Phishing Detection Using Hybrid Algorithm Based on Clustering and Machine Learning. *International Journal of Computing and Digital Systems*, *15*(1): 1–13.
6. D. Angioni, L. Demetrio, M. Pintor and B. Biggio. 2022. *Robust Machine Learning for Malware Detection over Time*.
7. M. Atzmueller and E. Sternberg. 2017. Mixed-initiative feature engineering using knowledge graphs. *Proceedings of the Knowledge Capture Conference, K-CAP 2017*.
8. A. Barredo Arrieta, N. Díaz-Rodríguez, J. Del Ser, A. Bennetot, S. Tabik, A. Barbado, S. Garcia, S. Gil-Lopez, D. Molina, R. Benjamins, R. Chatila and F. Herrera. 2020. Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI. *Information Fusion*, *58*: 82–115.
9. M. O. Çakıroğlu, H. Kurban, P. Sharma, O. Kulekci, E. K. Buxton, M. Raeeszadeh-Sarmazdeh and M. Dalkilic. 2024. An extended de Bruijn graph for feature engineering over biological sequential data. *Machine Learning: Science and Technology*.

10. X. Chen, B. Qiao, W. Zhang, W. Wu, M. Chintalapati, D. Zhang, Q. Lin, C. Luo, X. Li, H. Zhang, Y. Xu, Y. Dang, K. Sui and X. Zhang. 2019. Neural Feature Search: A Neural Architecture for Automated Feature Engineering. *2019 IEEE International Conference on Data Mining (ICDM)*: 71–80.
11. Z. Chen, Z. Zhang, Z. Kan, L. Yang, J. Cortellazzi, F. Pendlebury, F. Pierazzi, L. Cavallaro and G. Wang. 2023. Is It Overkill? Analyzing Feature-Space Concept Drift in Malware Detectors. *IEEE Security and Privacy Workshops (SPW)*: 21–28.
12. J. Cheng. 2023. Graph Feature Management: Impact, Challenges and Opportunities. *Proceedings of the 6th Joint Workshop on Graph Data Management Experiences & Systems (GRADES) and Network Data Analytics (NDA)*.
13. D. Chicco, L. Oneto and E. Tavazzi. 2022. Eleven quick tips for data cleaning and feature engineering. *PLoS Computational Biology*, *18*(12).
14. J. Clearman, R. R. Fayzrakhmanov, G. Gottlob, Y. Nenov, S. Reissfelder, E. Sallinger and E. Sherkhonov. 2019. Feature Engineering and Explainability with Vadalog: A Recommender Systems Application. *Datalog*, *2*: 39–43.
15. B. Crook, M. Schlüter and T. Speith. 2023. Revisiting the Performance-Explainability Trade-Off in Explainable Artificial Intelligence (XAI). *2023 IEEE 31st International Requirements Engineering Conference Workshops (REW)*: 316–324.
16. Z. Deng, K. Chen, G. Meng, X. Zhang, K. Xu and Y. Cheng. 2022. Understanding Real-world Threats to Deep Learning Models in Android Apps. *Proceedings of the ACM Conference on Computer and Communications Security*: 785–799.
17. E. Esenogho, I. D. Mienye, T. G. Swart, K. Aruleba and G. Obaido. 2022. A Neural Network Ensemble with Feature Engineering for Improved Credit Card Fraud Detection. *IEEE Access*, *10*: 16400–16407.
18. D. W. Fernando and N. Komninos. 2022. FeSA: Feature selection architecture for ransomware detection under concept drift. *Computers and Security*, *116*.
19. B. Friedrich, T. Sawabe and A. Hein. 2023. Unsupervised statistical concept drift detection for behaviour abnormality detection. *Applied Intelligence*, *53*(3): 2527–2537.
20. G. Giakkoupis, A.-M. Kermarrec, O. Ruas and F. Taiani. 2021. Cluster-and-Conquer: When Randomness Meets Graph Locality. *2021 IEEE 37th International Conference on Data Engineering (ICDE)*: 2027–2032.
21. A. Grover and J. Leskovec. 2016. Node2vec: Scalable feature learning for networks. *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, *13-17-August-2016*: 855–864.
22. A. Guerra-Manzanares, M. Luckner and H. Bahsi. 2022. Android malware concept drift using system calls: Detection, characterization and challenges. *Expert Systems with Applications*, *206*.
23. A. Gupta, R. Christie and R. Manjula. 2017. Scalability in Internet of Things: Features, Techniques and Research Challenges. In *International Journal of Computational Intelligence Research* (Vol. 13, Issue 7).
24. A. Hartl, M. Bachl, J. Fabini and T. Zseby. 2020. Explainability and Adversarial Robustness for RNNs. *2020 IEEE Sixth International Conference on Big Data Computing Service and Applications (BigDataService)*: 148–156.

25. S. Hemalatha, M. Mahalakshmi, V. Vignesh, M. Geethalakshmi, D. Balasubramanian and J. Anand A. 2023. Deep Learning Approaches for Intrusion Detection with Emerging Cybersecurity Challenges. *2023 International Conference on Sustainable Communication Networks and Application (ICSCNA)*: 1522–1529.
26. W. Herzallah, H. Faris and O. Adwan. 2018. Feature engineering for detecting spammers on Twitter: Modelling and analysis. *Journal of Information Science*, *44*(2): 230–247.
27. W. Hong, J. Yin, M. You, H. Wang, J. Cao, J. Li, M. Liu and C. Man. 2023. A graph empowered insider threat detection framework based on daily activities. *ISA Transactions*, *141*: 84–92.
28. C. Huang. 2021. Feature Selection and Feature Stability Measurement Method for High-Dimensional Small Sample Data Based on Big Data Technology. *Computational Intelligence and Neuroscience*, *2021*.
29. Y. Huang, Y. Zhou, M. Hefenbrock, T. Riedel, L. Fang and M. Beigl. 2023. Automatic Feature Engineering Through Monte Carlo Tree Search. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *13715 LNAI*: 581–598.
30. R. Islam, W. Eberle and K. Ghafoor. 2020. Towards Quantification of Explainability in Explainable Artificial Intelligence Methods. *The Thirty-Third International Flairs Conference*.
31. S. O. Kayode and K. Sheriffdeen. 2024. *Enhancing Graph Neural Network Performance through Advanced Node and Edge Feature Engineering Techniques*.
32. P. W. Khan and Y. C. Byun. 2020. Genetic algorithm based optimized feature engineering and hybrid machine learning for effective energy consumption prediction. *IEEE Access*, *8*: 196274–196286.
33. S. Khoshraftar, S. Mahdavi, A. An, Y. Hu and J. Liu. 2019. Dynamic Graph Embedding via LSTM History Tracking. *2019 IEEE International Conference on Data Science and Advanced Analytics (DSAA)*: 119–127.
34. U. Khurana, H. Samulowitz and D. Turaga. 2018. Feature Engineering for Predictive Modeling Using Reinforcement Learning. *Proceedings of the AAAI Conference on Artificial Intelligence*, *32*(1).
35. H. Kim, B. S. Lee, W. Y. Shin and S. Lim. 2022. Graph Anomaly Detection With Graph Neural Networks: Current Status and Challenges. *IEEE Access*, *10*: 111820–111829.
36. T. Kim, N. Park, J. Hong and S. W. Kim. 2022. Phishing URL Detection: A Network-based Approach Robust to Evasion. *Proceedings of the ACM Conference on Computer and Communications Security*: 1769–1782.
37. S. Kumar, A. Viinikainen and T. Hamalainen. 2018. Evaluation of ensemble machine learning methods in mobile threat detection. *2017 12th International Conference for Internet Technology and Secured Transactions, ICITST 2017*: 261–268.
38. X. Larriva-Novo, V. A. Villagrá, M. Vega-Barbas, D. Rivera and M. Sanz Rodrigo. 2021. An IoT-focused intrusion detection system approach based on preprocessing characterization for cybersecurity datasets. *Sensors (Switzerland)*, *21*(2): 1–15.
39. L. Li, H. Yang, Y. Jiao and K. Y. Lin. 2020. Feature Generation Based on Knowledge Graph. *IFAC-PapersOnLine*, *53*(5): 774–779.

40. J. Liu, Y. Lin, M. Lin, S. Wu and J. Zhang. 2017. Feature selection based on quality of information. *Neurocomputing*, *225*: 11–22.
41. M. Lorbach, R. Poppe, E. A. van Dam, L. P. J. J. Noldus and R. C. Veltkamp. 2015. Automated Recognition of Social Behavior in Rats: The Role of Feature Quality. In V. Murino & E. Puppo (Eds.), *Image Analysis and Processing* (Vol. 9280, pp. 565–574). Springer International Publishing.
42. J. Lu, A. Liu, F. Dong, F. Gu, J. Gama and G. Zhang. 2018. Learning under Concept Drift: A Review. *IEEE Transactions on Knowledge and Data Engineering*, *31*(12): 2346–2363.
43. L. Massarelli, G. A. Di, L. Cini, F. Petroni, L. Querzoni and R. Baldoni. 2019. Investigating Graph Embedding Neural Networks with Unsupervised Features Extraction for Binary Analysis. *Proceedings of the 2nd Workshop on Binary Analysis Research.*
44. E. C. Mutlu, T. Oghaz, A. Rajabi and I. Garibay. 2020. Review on Learning and Extracting Graph Features for Link Prediction. In *Machine Learning and Knowledge Extraction* (Vol. 2, Issue 4, pp. 672–704). MDPI.
45. S. S. Naqvi, W. N. Browne and C. Hollitt. 2016. Feature quality-based dynamic feature selection for improving salient object detection. *IEEE Transactions on Image Processing*, *25*(9): 4298–4313.
46. E. Navruzov and A. Kabulov. 2022. Detection and analysis types of DDoS attack. *2022 IEEE International IOT, Electronics and Mechatronics Conference, IEMTRONICS 2022*.
47. S. Picard, C. Chapdelaine, C. Cappi, L. Gardes, E. Jenn, B. Lefevre and T. Soumarmon. 2020. Ensuring Dataset Quality for Machine Learning Certification. *2020 IEEE International Symposium on Software Reliability Engineering Workshops*: 275–282.
48. F. Pierazzi, G. Apruzzese, M. Colajanni, A. Guido and M. Marchetti. 2017. Scalable architecture for online prioritisation of cyber threats. *International Conference on Cyber Conflict, CYCON*, *2017-June*: 1–18.
49. T. Pourhabibi, K. L. Ong, B. H. Kam and Y. L. Boo. 2020. Fraud detection: A systematic literature review of graph-based anomaly detection approaches. *Decision Support Systems*, *133*.
50. J. Qu, J. Arguello and Y. Wang. 2021. A Study of Explainability Features to Scrutinize Faceted Filtering Results. *International Conference on Information and Knowledge Management, Proceedings*: 1498–1507.
51. C. Regan, M. Nasajpour, R. M. Parizi, S. Pouriyeh, A. Dehghantanha and K.-K. R. Choo. 2022. Federated IoT attack detection using decentralized edge data. *Machine Learning with Applications*, *8*: 100263.
52. R. Reshma and A. Jose Anand. 2023. Predictive and Comparative Analysis of LENET, ALEXNET and VGG-16 Network Architecture in Smart Behavior Monitoring. *2023 Seventh International Conference on Image Information Processing (ICIIP)*: 450–453.
53. A. A. Rida, R. Amhaz and P. Parrend. 2023. Metrics for Evaluating Interface Explainability Models for Cyberattack Detection in IoT Data. *International Conference on Complex Computational Ecosystems*: 180–192.
54. T. Siameh. 2017. *Graph Analytics Methods In Feature Engineering*.
55. J. S. Tharani, Z. Hou, E. Y. A. Charles, P. Rathore, M. Palaniswami and V. Muthukkumarasamy. 2024. Unified Feature Engineering for Detection of Malicious

Entities in Blockchain Networks. *IEEE Transactions on Information Forensics and Security*.
56. M. Torres, R. Alvarez and M. Cazorla. 2023. A Malware Detection Approach Based on Feature Engineering and Behavior Analysis. *IEEE Access*, *11*: 105355–105367.
57. F. Tramèr, R. Shokri, A. San Joaquin, H. Le, M. Jagielski, S. Hong and N. Carlini. 2022. Truth Serum: Poisoning Machine Learning Models to Reveal Their Secrets. *Proceedings of the ACM Conference on Computer and Communications Security*: 2779–2792.
58. J. E. van Timmeren, R. T. H. Leijenaar, W. van Elmpt, J. Wang, Z. Zhang, A. Dekker and P. Lambin. 2016. Test–Retest Data for Radiomics Feature Stability Analysis: Generalizable or Study-Specific? *Tomography*, *2*(4): 361–365.
59. G. Vilone and L. Longo. 2021. Notions of explainability and evaluation approaches for explainable artificial intelligence. *Information Fusion*, *76*: 89–106.
60. Z. Wang, F. Yang, Q. Xu, Y. Wang, H. Yan and M. Xie. 2023. Capacity estimation of lithium-ion batteries based on data aggregation and feature fusion via graph neural network. *Applied Energy*, *336*.
61. G. I. Webb, R. Hyde, H. Cao, H. L. Nguyen and F. Petitjean. 2016. Characterizing Concept Drift. *Data Mining and Knowledge Discovery*, *30*(4): 964–994.
62. Xin XIA and D. Lo. 2018. Feature Generation and Engineering for Software Analytics. In *Feature engineering for machine learning and data analytics* (pp. 335–358).
63. C. Xue, X. Wang, Y. Zhou, P. Palangappa, R. Brugarolas Brufau, A. D. Kakne, R. Motwani, K. Ding and J. Zhang. 2023. Graph Enhanced Feature Engineering for Privacy Preserving Recommendation Systems. *ACM International Conference Proceeding Series*: 44–51.
64. L. Xue, C. F. Lynch and M. Chi. 2016. Unnatural Feature Engineering: Evolving Augmented Graph Grammars for Argument Diagrams. *International Educational Data Mining Society, Paper Presented at the International Conference on Educational Data Mining (EDM)*.
65. W. Yao, H. Shi and H. Zhao. 2023. Scalable anomaly-based intrusion detection for secure Internet of Things using generative adversarial networks in fog environment. *Journal of Network and Computer Applications*, *214*.
66. H. Zhang, Q. Gan, D. Wipf and W. Zhang. 2023. *GFS: Graph-based Feature Synthesis for Prediction over Relational Databases*.
67. W. Zheng, X. Zhu, Y. Zhu, R. Hu and C. Lei. 2018. Dynamic graph learning for spectral feature selection. *Multimedia Tools and Applications*, *77*(22): 29739–29755.
68. D. Zimbra, M. Ghiassi and S. Lee. 2016. Brand-related twitter sentiment analysis using feature engineering and the dynamic architecture for artificial neural networks. *Proceedings of the Annual Hawaii International Conference on System Sciences*, *2016-March*: 1930–1938.
69. R. Zuech and T. M. Khoshgoftaar. 2015. A survey on feature selection for intrusion detection. *Proceedings of the 21st Issat International Conference on Reliability and Quality in Design*: 150–155.