The LeWiS Method: Target Variable Estimation using Cyber Security Intelligence

Leigh Chase[1][a], Alaa Mohasseb[1][b] and Benjamin Aziz[2][c]
[1]*School of Computing, University of Portsmouth, Portsmouth, UK*
up348663@myport.ac.uk, {alaa.mohasseb, benjamin.aziz}@port.ac.uk

Keywords: Cyber Security, Machine Learning, Threat Intelligence, Estimation Methods, STIX, TTPs

Abstract: Information Technology plays an increasingly important role in the provision of essential services. For these systems and networks to be reliable and trustworthy, we must defend them from those who would seek to compromise their Confidentiality, Integrity and Availability. Security intelligence tells us about the Tactics, Techniques and Procedures used by threat actors for these very purposes. In this paper, we introduce a novel method for learning malicious behaviours and then estimating how likely it is that a system has been compromised. One of the difficulties encountered when applying machine learning to cyber security, is the lack of ground truth on which to train supervised techniques. This is often compounded by the volume, variety and velocity of data which is far greater than can be processed using only human analyses. The technique, known as LeWiS, includes data preparation and processing phases that learn and later predict the presence of threat actors using a model of their behaviours. The method addresses the problems of scale and veracity, by learning Indicators of Attack via feature extraction from security intelligence that has been obtained through empirical methods. This approach shows promising classification performance for detecting learned malicious behaviours, within synthesised systems' event data.

1 INTRODUCTION

Information security is an enduring challenge and one whose importance is underscored simply by reading the popular media (BBC, 2021). The infection of the UK's National Health Service (NHS) by the WannaCry ransomware in 2017 highlights the widespread, real-world impact of sophisticated cyber attacks. It also suggests lessons we can learn about preparedness and the need for continuous evolution of our response efforts (Smart, 2018). Today's information systems are complex - they combine legacy with emerging technologies, are made-up of heterogeneous systems and provide more varied services than ever before. In parallel, we can see an upward trend in the frequency of cyber attacks as well as a diversification of the actors to whom they are attributed (DCMS, 2021). As identified by (Kumar, 2016), these factors demonstrate the need for creativity and novel approaches to how we construct and apply defensive measures. Machine learning for advanced threat detection is one example of innovation in the field.

There are many ways in which to apply machine learning to cyber security and a similar range of reasons as to why one may wish to do so. Whilst often portrayed as a single domain, security actually contains a far greater number of nested topics and important challenges. These are as diverse as software reverse engineering, Security Information and Event Management (SIEM), generative signature techniques and policy-based management - all of which require different methods, algorithms and processing techniques (Shaukat et al., 2020). In this paper we are concerned with the predictive power of machine learning and the use of Cyber Threat Intelligence (CTI) as a framework for knowledge representation. The literature highlights the increasing importance of CTI (Gupta, 2019), but research into intelligence-specific learning methods remains limited. Techniques such as those described by (Alghamdi, 2020) and (Amit et al., 2018) demonstrate applications using host- and network-sourced telemetry. However, they also highlight the risks of overfitting and the failure to generalise beyond a small number systems. In-line with (Shaukat et al., 2020), they also illustrate the impact that the lack of labelled

[a] https://orcid.org/0000-0002-1066-9125
[b] https://orcid.org/0000-0003-2671-2199
[c] https://orcid.org/0000-0002-9964-4134

data has on broadening the utility (and appeal) of learning schemes within this domain. These are critical issues and form the principal motivation for this work.

By way of a novel approach, we introduce a new technique named "Learning With the Structured Threat Information Expression language" (LeWiS). It is a method for estimating target variables within structured security event data, based on supervised machine learning. It involves training on intelligence material to learn the semantics for Indicators of Attack (IoAs), then construction of a behavioural model used to predict whether systems' event data indicates a compromise and if so, by whom. To represent CTI information during training and prediction, the Structured Threat Information Expression (STIX) language is used. STIX is an open-source standard maintained by the Organization for the Advancement of Structured Information Standards (OASIS, 2021b). The LeWiS method is designed to address the problems associated with (an absence of) labelled data and the tendency to over-fit. Experimentation using the MITRE ATT&CK framework (MITRE, 2021) (for training) and synthetic system events as estimation targets, indicates LeWiS achieves promising classification performance when provided with sufficient information about an attacker's Tactics, Techniques and Procedures (TTPs).

Section 2 of this paper discusses related work. Section 3 describes the LeWiS technique in detail and includes some worked-examples using real CTI. This section also provides some background on STIX, insofar that it is relevant to the main discussion. Section 4 provides the results of experiments conducted to validate and verify the technique during development. The final section (5) includes conclusions and a short discussion on further work.

2 RELATED WORK

Specialised research into learning the semantics of hostile actor TTPs remains limited. Much of the prior art focuses on the application of probabilistic, or more recently "deep learning" methods to problems such as anomaly detection, network event classification and behavioural analysis. These efforts include a very broad range of disciplines - examples include two-level Bayesian Networks and Markov models for use in unbalanced reporting environments (Zhou et al., 2018), clustering and classification within Internet Protocol (IP) packet analysis (Das and Morris, 2017) and the use of Random Forests in generalised network anomaly detection use-cases (Elmrabit et al., 2020). Within the wider literature we see the various parallels between general machine learning challenges (Scheau et al., 2018) and those more specific to cyber security (Shaukat et al., 2020). Outside of academic research, techniques are being used within commercial products by technology providers - such as Darktrace (Darktrace, 2017), Vectra (Vectra.ai, 2021) and Expanse (PaloAlto, 2021). Commercial implementations tend to focus more on methods that prioritise data for human analysts, underpinned by attempts to find similar and dissimilar behaviours within computer networks. More targeted examples consider the role of machine learning within specialised sub-domains, such as Software Defined Networking (Zhou et al., 2020). Contemporary research also includes an attacker's perspective on Artificial Intelligence (AI) - such as the potential for its misuse (KALOUDI and JINGYUE, 2020) and the evasion of defensive measures based upon it (Xu et al., 2020).

The importance of both using and sharing CTI is reflected in government thinking. This is evidenced by Special Publication 800-150 from the National Institute for Standards and Technology, which identifies shared situational awareness, improved security posture, knowledge maturation and greater defensive agility as the principal benefits of sharing (Johnson, 2019). The UK's Cyber Security Information Sharing Partnership established by the National Cyber Security Centre, actively promotes government-industry sharing of threat intelligence materials for mutual benefit (NCSC, 2021). (Fransen et al., 2015) discusses the advantaged of using CTI when attempting to improve our understanding of malicious TTPs in enterprise environments. (Riesco et al., 2020) notes some of the difficulties associated with CTI-sharing communities - viz. the tendency to consume more than one provides - and introduces an innovative sharing mechanism using a smart contract-type structure, underpinned by Blockchain technology. The Authors do not explicitly cover machine learning, but the topics on modelling tactics and the need for ontology are familiar to this work. Prior art by two of the same authors introduces the idea of semantic inference using CTI - making explicit reference to the STIX standard, as well as to the Web Ontology Language (Riesco and Villagrá, 2019). Authors in (Jungsoo et al., 2020) outline a technique to automatically generate Malware Attribute Enumeration and Characterisation (MAEC) records using STIX-formatted CTI. This idea focuses more on automation and parsing within an intelligence 'workflow', but has clear parallels with the supervised estimation techniques introduced herein.

On the subject of ontology, (Blackwell, 2010) describes an original approach to the formulation and

expression of security incident data. This is paired to a three-level architecture for planning and preparing defensive measures. This work is especially notable because it predates what are now common standards in CTI data management. In (Ben-Asher et al., 2016) the authors introduce semantics as a useful abstraction built upon base data elements - such as the direction of flow, distribution of protocols, packets per hour and related metrics within IP networks. This includes a specific example on command and control channels that is especially relevant to the detection of malware. Importantly, the authors also introduce the idea of granularity and the resolution with which observations are made - showing some accordance with how transferable the resulting ontology is. Many works identify the problems associated with constructing a meaningful abstraction to model the behaviours of cyber actors - (Wali et al., 2013) sets out the problems and proposes a novel boot-strapping approach, that aims to reduce the burden placed on designers and engineers. This technique combines an existing ontology with a literal text book, demonstrating an approach that seeks to maintain the currency and relevance of the knowledge that has been modelled. Interestingly, this work is a contemporary of the very early release of STIX in 2012-2013.

Most closely related to this paper is (Zheng et al., 2018), within which the authors explicitly make the link between STIX and machine learning. Crucially, (Zheng et al., 2018) highlights the constrained nature of how machine learning has been applied in the prior art - namely that it uses limited, very specific data for similarly bounded purposes. The authors' experimentation combined Multi-layer Perceptron networks with Gradient Boosting Decision Trees, to address different facets of the problem domain. Verification found very good classification performance for certain types of attack against web services - as high as 96.2% under some conditions. Finally, (Mittal, 2017) describes the relevance of combining vector and graph-based representations to good effect. The authors introduce a structure they call the Vector Knowledge Graphs that seeks to blend the expressive quality of knowledge graphs with the crispness and functional nature of vectors.

The LeWiS method introduced in this paper offers an alternative to previous approaches. This centres on a new semantic model that combines subjects and object-predicates, with a supervised classification scheme to detect and attribute known-malicious TTPs. This is discussed in Section 3.

3 PROPOSED APPROACH

Modelling causation within cyber security data is difficult for a great many reasons, principal among them are the problems of data consistency and completeness (Mugan, 2013). LeWiS attempts to learn the semantics of how malicious actors compromise their targets. This information is articulated using STIX intrusion sets and the relationships they have to attack-pattern, malware and tool objects. Machine learning is applied via supervised techniques and trained using TTP information provided by ground-truth data. Predictions classify the behaviours within system telemetry using the pre-trained model - this data is also represented as STIX. The method is inherently repeatable and designed with automation in mind - similarly, it can be retrained whenever new or revised TTP data becomes available. LeWiS comprises three internal steps: Pre-processing, Processing and Learning, which are combined into a single 'pipeline' of operations.

- **Pre-processing** - concerned with data acquisition, normalization, feature extraction and constructing the subject-(object-predicate) data structure
- **Processing** - concerned with the constructing the vector representations using the subject-(object-predicate) data
- **Learning** - performs the fitting functions and predictions by applying the models to observable event data

These are discussed in Sub-sections 3.2, 3.3 and 3.4. They are also summarised in Figure 1.

3.1 DESCRIPTION OF THE DATASET

All input data (for training and prediction) are formatted as STIX (OASIS, 2021b). Formerly, this is a "schema that defines a taxonomy of cyber threat intelligence" (OASIS, 2021b) and employs a linked-data structure whose information architecture describes four main entities:

- STIX Domain Objects (SDOs);
- STIX Cyber Observable Objects (SCOs);
- STIX Relationship Objects (SROs); and
- STIX Meta Objects (SMOs).

Actor behaviours (TTPs) are articulated using the intrusion-set SDO, which in turn has relationships with other domain objects modelled using the SRO entity. Each relationship has a dedicated SRO whose source, target and type attributes describe the required

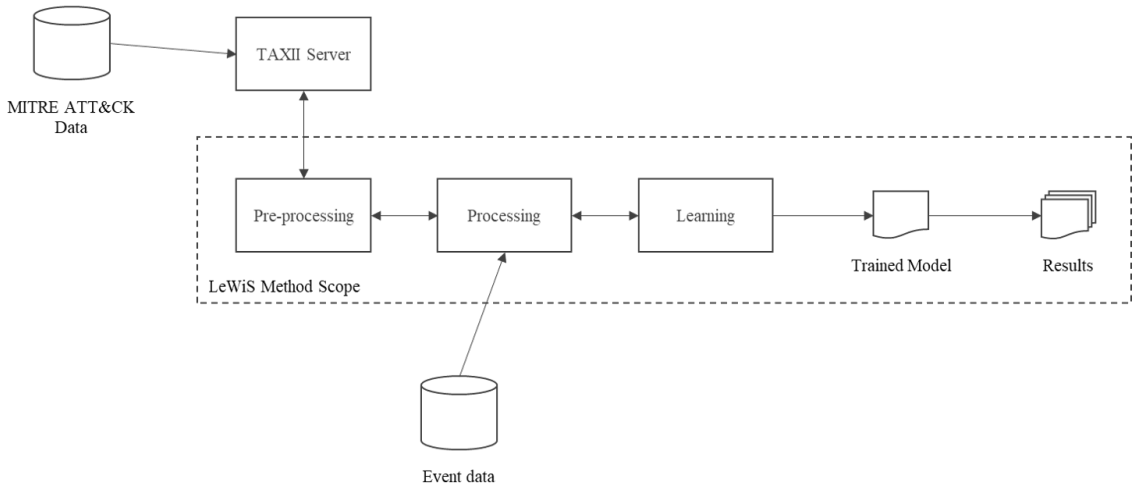

Figure 1: Steps of the LeWiS method.

association. This is shown in Figure 2. SROs can be used to link any of the other three STIX entities together. This paper focuses only on the relationships from intrusion sets to attack-pattern, malware and tool types.

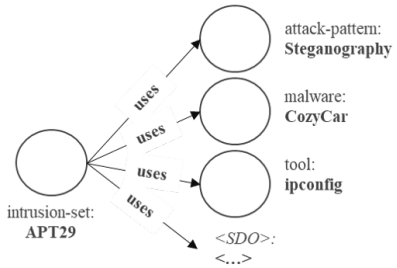

Figure 2: Summary of intelligence relationships

SDOs and SROs are used for training and SCOs / SDOs for variable estimation. Training was completed using the MITRE Enterprise ATT&CK dataset, which was acquired from the organization's official TAXII (Trusted Automated Exchange of Intelligence Information) server (MITRE, 2021). It was chosen for its rich, real-world (not synthetic) intelligence on threat actors' TTPs. Given the MITRE Corporation's standing within the field, it is considered a trusted source of ground-truth. The data used in this paper is a 'snapshot' whose contents reflect the latest information available at the time it was downloaded (May 2021). The ATT&CK data is packaged as a single STIX Bundle (OASIS, 2021a), within which each SDO has a unique identifier contained within its id attribute. ATT&CK contains 10990 SROs that model relationships between 1594 SDOs. The domain objects are distributed thus: 692 attack-patterns, 125 intrusion-sets, 424 malware types, 70 tools, 266 course-of-action records, 14 x-mitre-tactics, 1 x-mitre-matrix, 1 identity; and 1 marking definition. This is shown in Figure 3.

3.2 PRE-PROCESSING

Pre-processing is used to transform STIX data into a format suitable for the Processing functions. All data follow the same preparation techniques and the approach can be applied to any valid STIX-formatted CTI. This allows generalisation beyond single (or a small number of) systems. The source STIX datasets are parsed into a map of Subject-(Object-Predicate) (SOP) data structures, the design for which is original to LeWiS. The SOP map's keys are the intrusion-set descriptors extracted from the SDOs. Their corresponding values are a nested map that contains the relationship data. This is shown in Figure 4.

This structure offers a way to efficiently serialize the data used to train the supervised learning model. It is advantageous because it is simple yet enforces a formal structure. In turn, this ensures TTP data are expressed consistently and in a platform-agnostic fashion. The SOP map is stored and processed in JavaScript Object Notation (JSON) format:
"target-var": {"object-predicates": [{"predicate": string, "object": "name": string, "index": int, "type": string}], "vectors": {"attack-pattern-vector": [int], "malware-vector": [int], "tool-vector": [int], "raw-ap-vector": [bool], "raw-m-vector": [bool], "raw-t-vector": [bool], "raw-sample-vector": [bool] }, "label-value": int }

We can consider a simple example using the "APT29" intrusion-set. *{"predicate": "uses", "object": {"name": "Malicious Link", "index": 189}, "type": "attack-pattern" }* This indicates that this actor

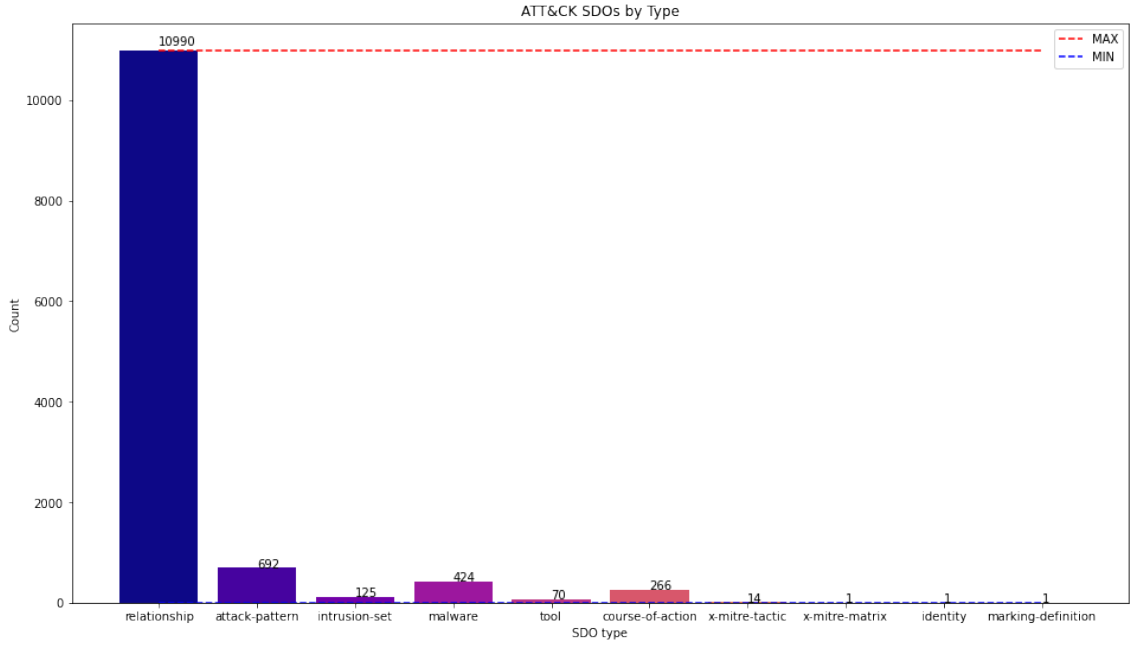

Figure 3: SROs use ID and type fields to relate SDOs

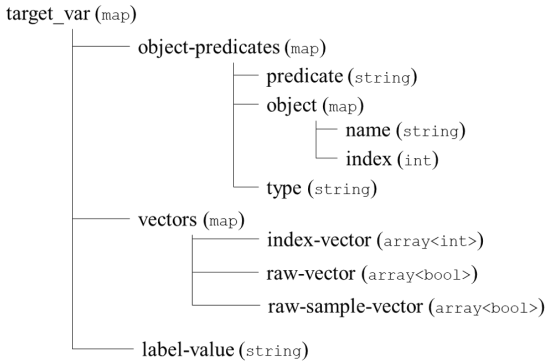

Figure 4: Example of the SOP map structure

"uses" the "Malicious Link" attack-pattern and that this pattern can be found at offset 189 in the set of all attack-patterns included in the STIX bundle. For APT29, the training set contains 109 Object-Predicate structures like this. These records act as qualifiers for the behavioural associations of the target intrusion-set – they are extracted during Pre-processing and parsed into this format. The Object-Predicates represent the qualities by which we can distinguish the classification targets – viz. they define the separability criteria that give the entire process meaning. Ultimately, this is a 'template' for learning features extracted from STIX data sets and provides a scaffold for preparing and fitting samples for classification. For later reference, the Subject (root key of the map) is what the learning algorithms are attempting to classify - this is discussed in Section 3.4.

3.3 PROCESSING

Processing generates vector representations for use in the Learning phase. Vectors are required because classifiers expect array-like input with consistent types - neither 'native' STIX nor the SOP structure conform to this. Processing has two inputs: SOP data for training and SCO data for prediction. Outputs are n-dimension vectors which are then passed to the Learning phase. These are stored under the vectors key in the SOP map data structure - meaning that eventually both the semantic relations and the vectors used to train the model are contained within a single data structure - extending the principles outlined by (Mittal, 2017). The intrusion-set relationship attributes are stored in the Vectors key of the SOP data structure. It contains another nested map within which are the vector representations of the Object-Predicates. LeWiS defines two types of vector: Index Vectors and Raw Vectors. The former contains the unique indices (integers) of all SDOs with which the target variable has an outbound relationship. For example, the APT29 has the following relationships with tool SDOs (non-exhaustive):
{'predicate': 'uses', 'object': {'name': 'AdFind', 'index': 5}, 'type': 'tool'},
{'predicate': 'uses', 'object': {'name': 'Tor', 'index': 39}, 'type': 'tool'},

{'predicate': 'uses', 'object': {'name': 'Mimikatz', 'index': 69}, 'type': 'tool'}
For each intrusion-set, Index Vectors for the associated attack-pattern (Equation 1), malware (Equation 2) and tool (Equation 3) objects are created. For APT29. The complete tool vector, for instance, is (Equation 4).

$$apt29_{attack-pattern-vector} \tag{1}$$

$$apt29_{malware-vector} \tag{2}$$

$$apt29_{tool-vector} \tag{3}$$

$$apt29_{tool-vector} = \langle 5,56,59,64,62,27,37,39,69,65 \rangle \tag{4}$$

Note (Equation 5) is the set of object-predicates in the SOP map for this intrusion-set.

$$|apt29_{x-vector}| = |op_x| \tag{5}$$

The Raw Vectors are a Boolean representation of each intrusion set's relationships - based on a simple "has" or "has not" evaluation. The total length of each vector is equal to the cardinality of the corresponding set of SDOs within the STIX bundle. Because the ID of each SDO is unique, if a target variable has an association with an SDO a 1 is stored in the vector at the location defined by the index. Therefore, each intrusion-set entry in the SOP map contains a Boolean vector named "raw_$<xxx>$_vector", where *xxx* is the short name for the SDO type. Continuing the example started above, there are 70 tool SDOs in the ATT&CK data and "APT29" has an association with 10 of them. The $apt29_{raw\_1\_vector}$ for this intrusion-set has 70 components of which only 10 take the value 1 (at the indices specified in $apt29_{tool-vector}$) - all other values are 0. The same approach is taken to populate the $apt29_{raw\_ap\_vector}$ (attack-patterns) and $apt29_{raw\_m\_vector}$ (malware).

In reality, threat actors are not characterised by the sum of all of their behaviours. Figure 2 shows APT29 is linked to the "Steganography" attack-pattern, the "CozyCar" malware and the "ipconfig" tool. The ATT&CK data contains 109 SROs for which the source reference is the ID of the APT29 intrusion-set. STIX-formatted intelligence does not includes occurrence information for TTPs (only for observations). That is to say, it does not specify how frequently a given intrusion-set uses any given attack-pattern, malware or tool. If only the entire feature vector for each intrusion-set was used to train the supervised model, then the learning algorithm could only make classification decisions when all of the attack-pattern, malware and tool relationships were presented. In plain terms, when all of the target threat actor's TTPs were contained in a single piece of telemetry! Instead, LeWiS includes a technique referred to as "α-β re-sampling", which creates additional vectors (Equation 6) whose components are subsets of the original 'full' feature vector (Equation 7).

$$|f(v_x, \alpha, \beta)| \tag{6}$$

$$f(v_x, \alpha, \beta) \subset v : v = \{x_0, x_1, ..., x_n\} \tag{7}$$

Where

- f is the re-sampling function,
- v_x is the intrusion-set vector
- $_x$ is the class label
- α is the re-sampling rate
- β is the re-sampling mask
- v is the set containing the original vector components

The Processing phase summarises all of this information in a single array-like structure. It contains the attack-pattern, malware and tool vectors (Equation 8).

$$v_x = \langle \langle ap_0, ap_1, ..., ap_n \rangle, \langle m_0, m_1, ..., m_n \rangle, \langle t_0, t_1, ..., t_n \rangle \rangle \tag{8}$$

APT29 has relationships with 75 attack-pattern SDOs, 24 malware SDOs and 10 tool SDOs. A naïve way to approach re-sampling is to simply create sets from the feature vectors, then calculate the Power Set (Equation 9) to create component vectors representing each combination of elements. This resolves to Equation 10 or as the binomial Equation 11, which does not scale well if the size of the feature vectors (the value of n) becomes very large.

$$\mathcal{P}(v_x) \tag{9}$$

$${}_nC_k = \frac{n!}{r!(n-r)!} \tag{10}$$

$$\binom{N}{k} \tag{11}$$

In CTI it is practical to assume that the size of the attack-pattern, malware and tool vectors will increase over time. The α and β terms introduced above are used to avoid the "curse of dimensionality"(Bellman, 1957). The α value is array-like and contains integers

used to define the maximum number of samples to be taken from each sub-vector (example, Equation 12).

$$\alpha = \langle max_a p, max_m, max_t \rangle \tag{12}$$

The β value is also array-like and contains three vectors of integers (example, Equation 13). It is used to mask values that are not of interest to the re-sampler. The three component vectors represent the attack-pattern, malware and tool SDOs thus allow users to be selective over what is not used during re-sampling. This is of particular use to those building models to learn TTP patterns used against specific systems. For instance, to ignore all malware and tool SDOs relating only to Microsoft Windows operating systems where the user is only concerned with Linux targets.

$$\beta = \langle \langle 12, 34 \rangle, \langle 7 \rangle, \langle 28 \rangle \rangle \tag{13}$$

System telemetry is noisy and the detection methods for actor TTPs must combine events that happen over time. These events are parsed into SCOs containing the corresponding instance data (one per event), which are then aggregated to populate an Observed Data SDO.

3.4 LEARNING

LeWiS functions in a 'de-coupled' fashion and does not prescribe any particular learning methods. What LeWiS is really providing is a semantic approach to learning actor TTPs, using the SOP data structure. This avoids the need to develop highly specialized logic that applies only on a system-by-system basis. The bulk of the work done by this technique is representational - viz. building a domain model that is consistent, platform-agnostic and can be used for both training and prediction. LeWiS has been tested using Support Vector Machine (SVM), Decision Tree Classifier (DTC) and Logistic Regression (Logit) algorithms. The normalized confusion matrix was used to measure classification performance. The results are discussed in Section 4.

These techniques were chosen because they all support large class registries, handle multiple data types and are interpretable. We avoid the deep learning methods used by (Zheng et al., 2018) and (Wali et al., 2013), to preserve transparency and 'explainability' within the Learning phase. Furthermore each represents a different approach to classification and offers a high-degree of configuration potential. As with other aspects of the technique, the process for choosing the 'best' classifier is domain-specific and not something that can specified without particular uses in mind. All development, training, testing and configuration activities were completed using the Python "sklearn" (Sci-kit Learn) module - these techniques were not implemented from scratch. Currently, the approach affords limited options for re-training and does not include reinforcement techniques. The ATT&CK data used to train with LeWiS was imported into the local work space - it was not 'online' in any sense and so the models reflect the "as-was" view of the data, according to when it was acquired.

4 RESULTS OF EXPERIMENTATION

Whilst this is an exploratory technique, the approach has shown promising results when classifying intrusion-set objects. Applying LeWiS to the latest ATT&CK data (version 9, at the time of writing) yields 125 unique class labels, which includes the null-actor. The ATT&CK training data includes intelligence on 124 intrusion sets and 1186 objects of types attack-pattern, malware and tool. Sparse matrices within SOP structures were common simply because ATT&CK contains intelligence on a broad range of TTPs. This variety means intrusion-sets or attack-patterns can make good discriminators. The SOP generation counts are show in Figure 6.

The component feature vectors are defined in Equation 14, Equation 15 and Equation 16. The resulting 'full' feature vector is Equation 17. Re-sampled vectors each have the same dimensions. When applying LeWiS to the ATT&CK data, it generates 3138 SOP structures within the map that span all intrusion-set SDOs.

$$|v_a p| = 692 \tag{14}$$

$$|v_m| = 424 \tag{15}$$

$$|v_a p| = 70 \tag{16}$$

$$|v_x| = 1186 \tag{17}$$

Testing was completed using SVM, DTC and Logit learning algorithms. These were evaluated and the best models chosen according to their average classification accuracy. Specimen vectors where synthesised to test the classifiers, however as these are not created from live telemetry the output is considered advisory. Initial exploration suggests the SVM, DTC and Logit. techniques produce similar performance - maximum average accuracy was 59.1%, minimum was 43.7% using a common set of synthesized vectors. A logical extension to this research is greater

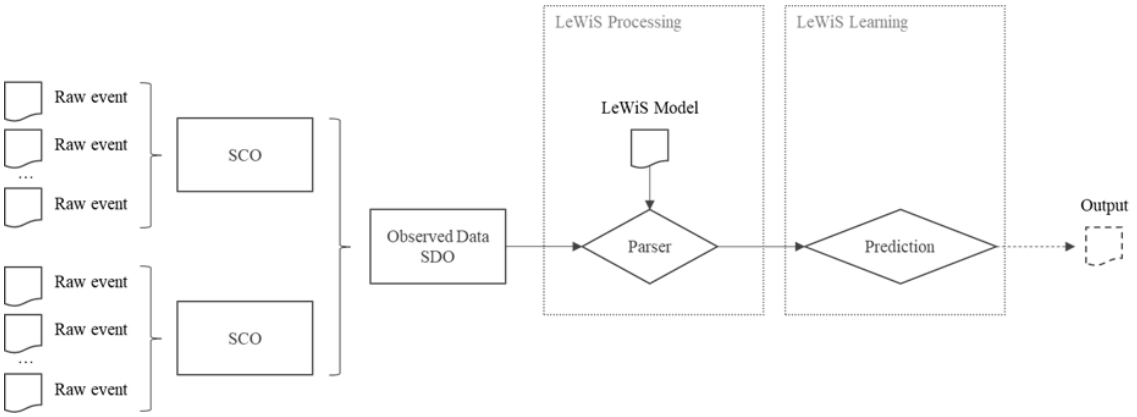

Figure 5: Telemetry processing using SCOs and SDOs

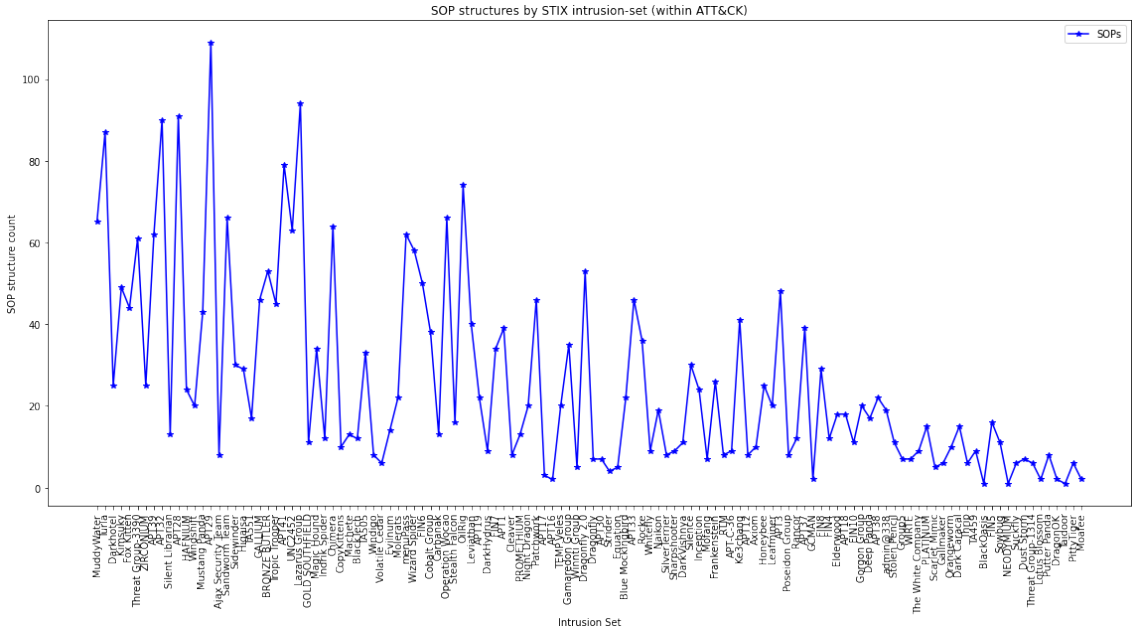

Figure 6: SOP object counts by intrusion-set

evaluation of how the re-sampling techniques can improve overall performance. The technique proved predictably sensitive to 'hedging' its classification decisions where intrusion-sets exhibit common features from across the attack-pattern, malware and tool objects. This offers some insight into the practical separability of actor TTPs using these SDOs, but it also suggests further analysis is required to understand these sensitivities more fully. The α-β re-sampling technique provides some mitigation by compensating the 'reporting bias' within the training data (where the distribution of TTP information is unbalanced). Using the ATT&CK data, the depth of information on attack-pattern and malware SDOs mean the classification logic is typically biased towards these types. As stated in the Sections 1 and 2, the abstraction and knowledge representation layers built atop STIX are a vital part of the technique's portability and generalisation. Performance across the three algorithms varied by small degrees and the general trend confirms that actors with more ranging TTP information yield the highest classification performance. The training set contained a notable imbalance in the number the attack-pattern, malware and tool SDOs. The first two are far more populous than the last, however the combination of attack-pattern and malware relationships appears to be more indicative of specific actors when all three SDOs are present. Where tool types were dominant these proved a positive discriminator. Interestingly, this suggests the technique might be effective in detecting actors involved in 'living off the land' attacks. Generally, in the case of actors for whom the

training data was sparse, training performance was far lower than is desirable. The performance scores are shown in Figure 7.

Peak performance (not averaged) was produced for the APT29, Dragonfly 2.0 and BRONZE BUTLER intrusion sets - reaching the 89^{th} percentile under optimum conditions. With little-or-no re-sampling the overall classification performance was poor - giving a mean of 26.8%. This was caused by the sparse data in evidence for certain intrusion sets or where there was considerable duplication of TTPs between actors. When suppressing sets that gave very poor performance and performing basic re-sampling, the mean performance rose to 59.1%, with 15 intrusion sets performing well above this. Performance was markedly worse for intrusion sets about which little is known - this is as one might expect For these actors the classifiers scored poorly because the lack of TTP data meant there was little to discriminate these actors from others. DragonOK and Taidoor classification was especially poor - effectively yielding zero. This could likely be addressed with weights that compensate for these tendencies but this was not attempted by this research. The lack of TTP data also meant α-β re-sampling was impractical. The principle difficulty is that when re-sampling is applied on actors for whom a large amounts of TTP data was available, the re-sampled vectors may contain the same information as the full (raw) feature vector for these 'lesser known' intrusion sets. This is problematic because if a mask vector is provided for β when re-sampling vectors for the 'greater known' sets, it is possible that this may inadvertently remove relationships that are statistically significant to the classification decision. In practice, there is no way to know this without relying on input from expert users.

5 CONCLUSION AND FURTHER WORK

The overall classification performance is notable for intrusion-sets on whom the training corpus contains a suitable volume of information. Overall performance is also by the re-sampling process applied to the SOP map and so the more varied and voluminous the training data are, the more re-sampling can be effectively performed to 'tune' performance. The utility of this approach is also demonstrated through the ability to create specimen vectors for prediction against models trained on real CTI data sets. This find broader applications in intrusion detection system and firewall testing, or when simulating security incidents for the purposes of personnel development. The multi-class implementations for each of the classifiers tested set the weights for each class to 1.0. This was done to avoid introducing skew or bias that was not inferred within the training data, but also because predictions were made using only specimen vectors. Real-world scenarios would introduce greater context given by the type of system being monitored and business-level information about the threats faced. Operators applying LeWiS to actual systems may wish to bias the classification decision depending on factors, such as:

- '*Guilty knowledge*' held by a user that would inform proper classification decisions, but cannot be (or is not) encoded within the learning methods;
- Trustworthiness or known-accuracy of the intelligence on which the LeWiS model was trained;
- To reflect the quality, or some other attribute, of the data provided by the system under scrutiny that affects the classification results; and
- To use the classification to scale, or become a coefficient of, a value external to LeWiS process - such as a calculated risk score.

Whilst the Boolean vectors bring relatively large dimensionality, they are simple-valued and have comparatively low storage complexity. A more elegant solution may be preferable in future iterations however, since the vector sizes scale linearly with the growth of intelligence material and they will likely become unwieldy. Improvements can also be made to the re-sampling function by applying masking operations (such as exclusive-OR logic) to create the combinations required. It is interesting to consider whether additional semantics might improve overall performance - for example, the introduction of second-order logic and conditionals that do not treat all relationships as equal. The ideas that underpin the SOP data structure could be extended to include statefulness with respect to the actor. This might further qualify their presence within a network / system and may also give some insight into what further actions they might perform. This may be especially useful in real-world scenarios, where an actor has already compromised some part of a system and those charged with its defence seek to understand how the threat could move laterally or gain a toehold in other systems. This initial, exploratory version of LeWiS is attempting to simply determine the presence of an actor - in reality this resolves to a binary classification of the system under review being in one of two states: compromised or not-compromised. Further development of LeWiS could see it applied in a more differentiated fashion such that it can work within the 'degrees of compromise' that exist in the real world. In so doing, it might offer insights into post-compromise de-

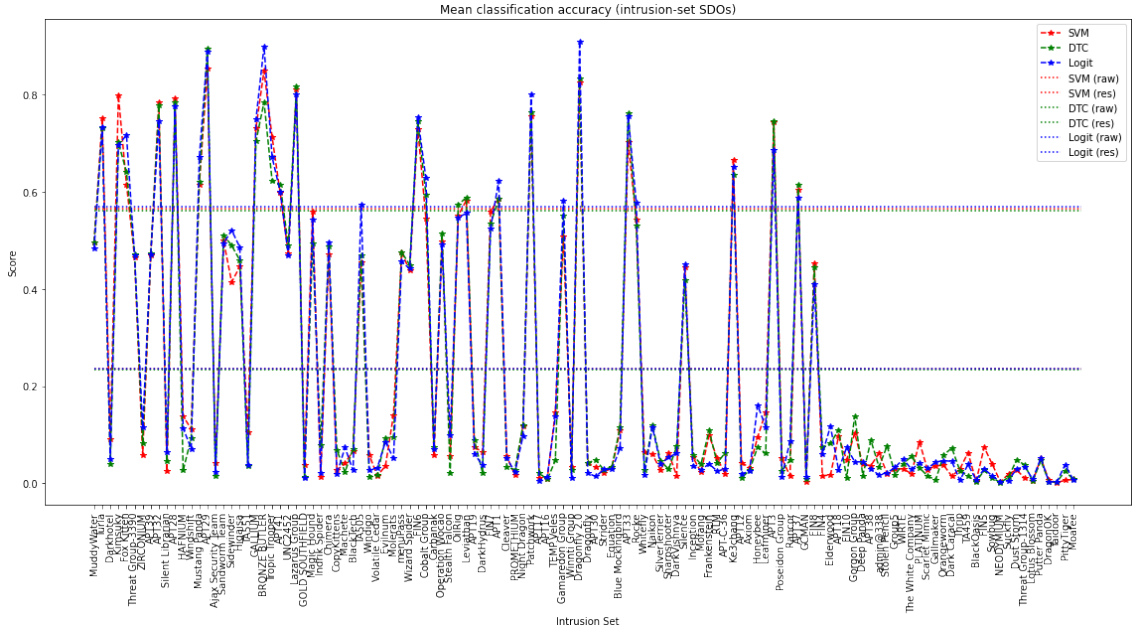

Figure 7: SOP Classification Scores

fensive techniques to isolate, mitigate and manage the effect of hostile actors already operating within a network or system.

Perhaps the most compelling extension to the LeWiS method is generalising it to predict other target variables. This would entail training models whose classification targets are not only intrusion sets, but broaden to include Infrastructure, Malware and Vulnerability SDOS. Infrastructure estimation is an example of finding new TTPs through generalisation of known data. Similarly for Malware SDOs, telemetry would be used to estimate the presence of a particular implant within a system (rather than whom may be responsible for it). This may offer opportunities for detection outside of more more conventional means (such as intrusion detection and endpoint security technologies). Estimating Vulnerability SDOs could establish the presence of a specific vulnerability, or set of vulnerabilities within the monitored infrastructure purely thought intelligence processing.

Finally, STIX does not include a mechanism to state how common is any particular relationship between SDOs. This could be of real significance in machine learning and be used to improve the resolution of the models; avoiding the need to train only on binary relationships (i.e. one exists, or it does not) and allowing a more comprehensive scheme to be defined that uses the degree to which a relationship is present. The re-sampling technique described herein provides a partial solution to the problem, but greater improvements could likely be made by adding 'strengths' to the underlying data model. This is of course, not a trivial undertaking and it is necessary to remember that this additional attribute would require greater empirical information that might otherwise be used to construct attack sets. Furthermore, one has to assume imperfect knowledge of the TTPs for any threat actor and because the 'strength' attribute is a function of other observable data, it may be difficult to manage bias when working in real-world settings.

REFERENCES

Alghamdi, M. I. (2020). Survey on applications of deep learning and machine learning techniques for cyber security. *International Journal of Interactive Mobile Technologies*, 14:210 – 224.

Amit, I., Matherly, J., Hewlett, W., Xu, Z., Meshi, Y., and Weinberger, Y. (2018). Machine learning in cyber-security - problems, challenges and data sets.

BBC (2021). Cyber attack 'most significant on irish state'. https://www.bbc.co.uk/news/world-europe-57111615.

Bellman, R. (1957). *Dynamic Programming*. Princeton University Press.

Ben-Asher, N., Hutchinson, S., and Oltramari, A. (2016). Characterizing network behavior features using a cyber-security ontology. *MILCOM 2016 - 2016 IEEE Military Communications*

Conference, Military Communications Conference, MILCOM 2016 - 2016 IEEE, pages 758 – 763.

Blackwell, C. (2010). A security ontology for incident analysis. In *CSIIRW 10*.

Darktrace (2017). Darktrace industrial uses machine learning to identify cyber campaigns targeting critical infrastructure. https://www.darktrace.com/en/press/2017/204/.

Das, R. and Morris, T. H. (2017). Machine learning and cyber security. *2017 International Conference on Computer, Electrical Communication Engineering (ICCECE), Computer, Electrical Communication Engineering (ICCECE), 2017 International Conference on*, pages 1 – 7.

DCMS (2021). Cyber security breaches survey 2021. Technical report, UK Government. https://www.gov.uk/government/statistics/cyber-security-breaches-survey-2021/cyber-security-breaches-survey-2021.

Elmrabit, N., Zhou, F., Li, F., and Zhou, H. (2020). Evaluation of machine learning algorithms for anomaly detection. *2020 International Conference on Cyber Security and Protection of Digital Services (Cyber Security), Cyber Security and Protection of Digital Services (Cyber Security), 2020 International Conference on*, pages 1 – 8.

Fransen, F. . . ., Kerkdijk, R. . . ., and Smulders, A. . . . (2015). Cyber security information exchange to gain insight into the effects of cyber threats and incidents. *Elektrotechnik und Informationstechnik*, 132(2):106–112.

Gupta, B., S. Q. (2019). *Machine learning for computer and cyber security : principles, algorithms, and practices.* CRC Press.

Johnson, C., B. L. W. D. S. J. S. C. (2019). *NIST Special Publication 800-150: Guide to Cyber Threat Information Sharing*. National Institute for Standards and Technology.

Jungsoo, P., Long Nguyen, V., Bencivengo, G., and Souhwan, J. (2020). Automatic generation of maec and stix standards for android malware threat intelligence. *KSII Transactions on Internet Information Systems*, 14(8):3420 – 3436.

KALOUDI, N. and JINGYUE, L. (2020). The ai-based cyber threat landscape: A survey. *ACM Computing Surveys*, 53(1):1 – 34.

Kumar, S.R, Y. S. S. S. S. A. (2016). Recommendations for effective cyber security execution. In *2016 International Conference on Innovation and Challenges in Cyber Security (ICICCS-INBUSH)*, pages 342–346.

MITRE (2021). Mitre att&ck. https://cti-taxii.mitre.org/stix/collections/95ecc380-afe9-11e4-9b6c-751b66dd541e/objects/.

Mittal, S., J. A. F. T. (2017). Thinking, fast and slow: Combining vector spaces and knowledge graphs. *CoRR*, abs/1708.03310. http://arxiv.org/abs/1708.03310.

Mugan, J. (2013). A developmental approach to learning causal models for cyber security. volume 8751, page 87510A.

NCSC (2021). Cyber information sharing partnership. https://www.ncsc.gov.uk/section/keep-up-to-date/cisp.

OASIS (2021a). Stix 2.1 bundle specification. https://docs.oasis-open.org/cti/stix/v2.1/cs02/stix-v2.1-cs02.html$_g ms872 kuzdmg$.

OASIS (2021b). *STIX Version 2.1*. https://docs.oasis-open.org/cti/stix/v2.1/cs02/stix-v2.1-cs02.html.

PaloAlto (2021). Expanse - attack surface reduction. https://expanse.co/attack-surface-reduction/.

Riesco, R., Larriva-Novo, X., and Villagra, V. A. (2020). Cybersecurity threat intelligence knowledge exchange based on blockchain: Proposal of a new incentive model based on blockchain and smart contracts to foster the cyber threat and risk intelligence exchange of information. *Telecommunication Systems*, 73(2):259 – 288.

Riesco, R. and Villagrá, V. A. (2019). Leveraging cyber threat intelligence for a dynamic risk framework: Automation by using a semantic reasoner and a new combination of standards (stix™, swrl and owl). *International Journal of Information Security*, 18(6):715 – 739.

Scheau, M., Arsene, A.-L., and Popescu, G. (2018). Artificial intelligence / machine learning challenges and evolution. 7:11–22.

Shaukat, K., Luo, S., Varadharajan, V., Hameed, I., and Xu, M. (2020). A survey on machine learning techniques for cyber security in the last decade. *IEEE Access*, 8:222310–222354.

Smart, W. (2018). Lessons learned review of the wannacry ransomware cyber attack. Technical report, Department for Health and Social Care. https://www.england.nhs.uk/wp-content/uploads/2018/02/lessons-learned-review-wannacry-ransomware-cyber-attack-cio-review.pdf.

Vectra.ai (2021). Vectra.ai - how we do it. https://www.vectra.ai/products/how-we-do-it.

Wali, A., Soon Ae, C., and Geller, J. (2013). A bootstrapping approach for developing a cyber-security ontology using textbook index terms. *2013 International*

Conference on Availability, Reliability and Security, Availability, Reliability and Security (ARES), 2013 Eighth International Conference on, Availability, Reliability and Security (ARES), 2012 Seventh International Conference on, pages 569 – 576.

Xu, J., Wen, Y., Yang, C., and Meng, D. (2020). An approach for poisoning attacks against rnn-based cyber anomaly detection. *2020 IEEE 19th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom), Trust, Security and Privacy in Computing and Communications (TrustCom), 2020 IEEE 19th International Conference on, TRUSTCOM*, pages 1680 – 1687.

Zheng, H., Wang, Y., Han, C., Le, F., He, R., and Lu, J. (2018). Learning and applying ontology for machine learning in cyber attack detection. *2018 17th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/ 12th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE), Trust, Security And Privacy In Computing And Communications/ 12th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE), 2018 17th IEEE International Conference On, TRUSTCOM-BIGDATASE*, pages 1309 – 1315.

Zhou, L., Shu, J., and Jia, X. (2020). Collaborative anomaly detection in distributed sdn. *GLOBECOM 2020 - 2020 IEEE Global Communications Conference, Global Communications Conference (GLOBECOM), 2020 IEEE*, pages 1 – 6.

Zhou, Y., Zhu, C., Tang, L., Zhang, W., and Wang, P. (2018). Cyber security inference based on a two-level bayesian network framework. *2018 IEEE International Conference on Systems, Man, and Cybernetics (SMC), Systems, Man, and Cybernetics (SMC), 2018 IEEE International Conference on, SMC*, pages 3932 – 3937.