

# EVOLVING FEATURES-ALGORITHMS KNOWLEDGE MAP TO SUPPORT NIDS DATA INTELLIGENCE AND LEARNING LOOP ARCHITECTING – A GENERALISED APPROACH TO NIDS PATTERN FEATURE SPACE KNOWLEDGE PROCESSING

**Atta Badii, Dhaval Patel**

Intelligent Media Systems & services Research Laboratory  
Department of Computer Science  
School of System Engineering, University of Reading,  
Reading RG6 6AY, United Kingdom  
{atta.badii, d.patel}@reading.ac.uk

## *Abstract*

*This paper describes a proposed new approach to the Computer Network Security Intrusion Detection Systems (NIDS) application domain knowledge processing focused on a topic map technology-enabled representation of features of the threat pattern space as well as the knowledge of situated efficacy of alternative candidate algorithms for pattern recognition within the NIDS domain. Thus an integrative knowledge representation framework for virtualisation, data intelligence and learning loop architecting in the NIDS domain is described together with specific aspects of its deployment.*

## *Keywords*

*Feature Map, Topic Map, Intrusion Detection System, Pattern recognition, Feature extraction, Algorithm, Algorithm Map, Intrusion Detection System (IDS), and Network Security.*

## **1. INTRODUCTION**

Pattern recognition has long been a topic of fundamental importance in a wide range of science and technology domains. In these days of rapidly growing information-oriented societies, improvement in its performance is an urgent technological issue. In particular, a mathematically proven, effective, and efficient method is desired for designing highly accurate pattern recognisers.

The subfield of computer science is concerned with the concepts and methods of symbolic inference by computer and symbolic knowledge representation for use in making inferences. Artificial Intelligence can be seen as an attempt to model aspects of human thought on computers. It is also sometimes defined as trying to solve by computer any problem that a human can solve faster. One branch of artificial intelligence is concerned with the classification or description of observations [A. Hoekstra at al].

Pattern recognition aims to classify data (patterns) based on either a priori knowledge or on statistical information extracted from the patterns. The patterns to be classified are usually groups of measurements or observations, defining points in an appropriate multidimensional space. Many theoretical studies in pattern recognition attempt to analyse the behaviour of

classifiers for all possible problems, i.e., classes defined on arbitrary combinations of points in a feature space [Tin Kam Ho et al].

A complete pattern recognition system consists of a sensor that gathers the observations to be classified or described; a feature extraction mechanism that computes numeric or symbolic information from the observations; and a classification or description scheme that does the actual job of classifying or describing observations, relying on the extracted features.

The classification or description scheme is usually based on the availability of a set of patterns that have already been classified or described. This set of patterns is termed the training set and the resulting learning strategy is characterised as supervised. Learning can also be unsupervised, in the sense that the system is not given an a priori labelling of patterns, instead it establishes the classes itself based on the statistical regularities of the patterns [M. Li et al].

The classification or description scheme usually uses one of the following approaches: statistical (or decision theoretic), syntactic (or structural), or neural. Statistical pattern recognition is based on statistical characterisations of patterns, assuming that the patterns are generated by a probabilistic system. Structural pattern recognition is based on the structural interrelationships of features. Neural pattern recognition employs the neural computing paradigm that has emerged with neural networks [M. Li et al].

A problem can be difficult to solve for various reasons. Certain problems are known to have a non-zero Bayes error. Others may have a complex decision boundary and/or subclass structures. Sometimes the high dimensionality of the feature space and sparseness of available samples lead to estimation difficulties.

Many pattern recognition problems can be reformulated as optimisation problems and then fit within the framework of evolutionary computation. Evolutionary algorithms (EA) use a metaphor of natural evolution, seen as the nature's answer to an optimisation problem, and exploit such a metaphor to tackle this class of problem. The implementation of such a metaphor requires general criteria to decide when EA can be conveniently used, some basic machinery, namely the evolutionary cycle and the genetic operators, and specific rules for setting the EA parameters in order to maximise the algorithm performance [Wikipedia].

When using EA in the framework of pattern recognition, there are two further issues that must be addressed: the reformulation of typical pattern recognition problems as optimisation ones and the crucial role played by both the encoding of the solution into the genotype and the definition of the fitness function on the attainable performance. In many cases of interest, moreover, this mapping naturally leads to a multimodal fitness function, and therefore the further issue of making an evolutionary algorithm able to search for many optima arises.

In reality, most practical classification problems arise from non chaotic processes many of which can be described by an underlying physical model. Though the models may contain a stochastic component, there should still exist a significant structure in the resulting data distributions that differ from a random combination of points.

Points with randomly assigned labels are difficult since not much can be learned from the training data about the unseen points. With real world recognition data such learning can often be done with various degrees of difficulty [T.Y. Kwok].

Data clustering is one of the fundamental techniques in scientific data analysis and data mining for pattern recognition. It partitions a data set into groups of similar items, as

measured by some distance metric. Over the years, data set sizes have grown rapidly with the exponential growth of computer storage and increasingly automated business and manufacturing processes [Tim hon et al].

In this paper we propose a novel method to develop an algorithm map for a generalised solution of NIDS domain pattern recognition problems. The algorithm map would be developed depending on the specific requirements of the pattern recognition problem, measures of problem complexity, related feature space and feature optimisation. This approach will consider and combine all available and best suitable algorithms to form a map, which eventually serve as a key hybrid multi-directional map to a particular pattern recognition problem and to reach the best efficient performance.

## **2. NIDS DOMAIN PATTERN SPACE**

The exploitation of a topic-map-enabled feature ontology revision approach has already been advocated in [A.Badii] which focused on the framework architecture for pattern space ontology engineering to support NIDS. This highlighted the advantages conferred by maintaining an integrated set of attack feature knowledge maps, and, attack detection rules-algorithms knowledge maps to offer a capability for dynamic update of attack reference signatures as well as of course responsive selection of applicable security policies, detection rules, and update of other specific behaviour models, attack context patterns, detection/correlation methods and algorithms. This is all so as to achieve the highest level of dynamic responsiveness and efficacy in NIDS by exploiting a virtuous learning loop to allow continuous evolutionary learning and enhancement of NIDS capability underpinned by both the efficacy of front-end real-time reconfigurable pattern matching hardware (FPGA) as well as optimal learning loop semantic integration and contextualising alerts thus enabling prioritisation in alerts processing and in particular filtering the false positive alerts. Malicious attacks directed at a computer network can of course impact the traffic volume of the network and various other aspects of its performance including the applications running on the computers. The attacks can also manifest their own characteristic attack patterns that may be open to integrative pattern data intelligence gathered from orchestrated pattern analysis in three network spaces namely at major network trunks or perimeter points, at gateways and at the desktops. Simultaneous deployment of both signature-based and anomaly-based approaches to packet stream scrutiny at the network perimeter with correlation mapping of all the available pattern intelligence can help improve Intrusion Detection and possible Pre-emption only if the challenges of semantic-integrative processing of the resulting intelligence across all sub-spaces of correlation can be overcome [Dominique Alessandri et al].

There are many challenges in architecting the feature map representation framework but once an open, dynamically evolvable framework has been established this can be exploited both for inferencing as well as virtualisation and Network Management Information including, if required, a cartographic attack pattern and alarm visualisation; particularisation of effects of semantic interventions in the feature map, network vulnerability fixes, and, security policy refinements.

The malware attack phenomena manifest themselves at a variety of levels ranging from socio-political, socio-technical and virtual-physical e.g. as depicted in Figure 1.

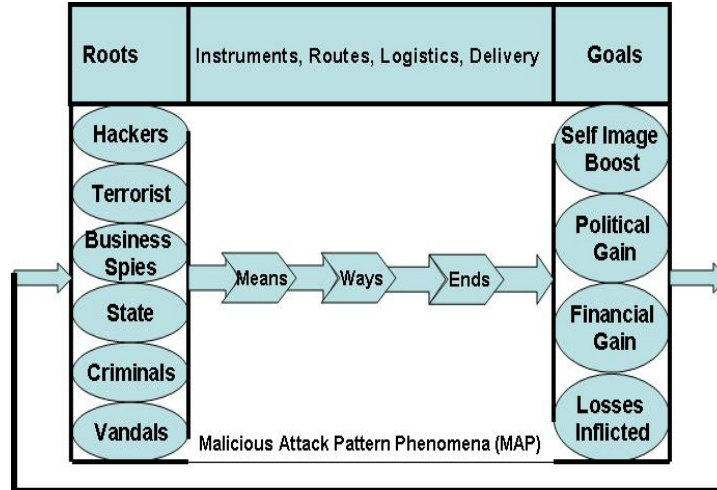


Figure 1: Malicious attack Pattern Phenomena (MAP).

However, although some socio-political and socio-technical attack pattern phenomena may be of interest to inform the security policy level, for the purpose of IDS-IPS functionality the analysis of causal, process and behavioural model of malware attacks will necessarily have to focus on the virtual-physical attack patterns (some of them listed below) and their consequences at various levels of abstraction:

- **Process-Specific Patterns**

- Causal e.g. relating to perpetrator type, initiation patterns
- Opportunistic e.g. relating to specific emergent vulnerabilities: old or recently announced.
- Roots: for example ancestry type, attacker type, country of origin etc.
- Routes: point of entry into the network, pattern of travel through the network and migration from host to host
- Contagion e.g. the infection model once unleashed, which is the pattern of mutation of the attacking code and its primary and secondary effects throughout the enterprise intranet and/or the internet.
- Containment: typical anti-dotes or patterns of containment whether it is by specific new fixes or including a self-limiting process.
- Spatio-temporal traffic patterns such as cartographic distribution of infected zones (domain clusters, countries) and global timing and duration of the attacks.

- **Episode-Specific Patterns**

Packet-specific, transaction/socket specific, connection-specific and session-specific features e.g.

- Protocol
- Immediate sender and receiver nodes in the network
- Source port addresses and TCP Flags
- Time-stamp
- Payload
- Secondary attributes of a network connection incident.
- Derived attributes of a network connection incident
- TCP Flags

- Traffic Type ( e.g. HTTP, FTP, Telnet, SMTP)
- Security Context ( security policy, security model)

Thus once the primary process of dynamic feature map optimisation is resolved then the FPGA technology and parallel programming approaches can be harnessed to achieve the necessary speed-up and re-adaptive operation that will be needed to realise the benefits of feature map -enabled semantic bridges and interventions that can enhance the attack pattern correlation and intrusion detection as well as infection localisation and alarm filtering [Stolfo J et al].

The NIDS Domain Feature Extraction Reviews and specifies a framework for representation, navigation and reasoning over the NIDS domain features to provide an integrative framework for representation and inference control over a multi-level, multi-model, multi-data view feature system hierarchy [J Li et al]. This will then be further elaborated to specify the virtual data model in a way that would allow multi-model semantic-associative and integrative inference to exploit:

- The semantic bridges that can operate horizontally across different signature models e.g. Snort-vulnerabilities-exploits, Hackers' cookbook reverse engineered signatures, packet stream observations (Live-scoping, Cumulative-Historical-Statistical) and dynamic responsive scoping of operative analysis windows and associated context ranges.
- A semantic scaffolding structure that can permit traversal up and down, and dynamic refinement of, the Feature Map abstraction hierarchy to enable Knowledge Extraction and Knowledge Reporting loops between the various levels to and from the off-line CPU-based inferencing environment to the online real-time FPGA-based processing arena. This will empower malware attack Detection & Identification (D&I) optimisation including dynamic learning as well as alarm filtering.

### **3. NIDS DOMAIN FEATURE MAP ONTOLOGY**

The need for non-linear modelling techniques has become more evident with our increasing capability to capture data. The Linearity assumption of traditional statistics methods is ill-suited to data intelligence processing based on information gathered by observing a human behaviour system because:

- i) Human behaviour seldom follows a linear model.
- ii) The number of variables in current data streams can be much larger than previously and this can outstrip the capabilities of old approaches to modelling

In attempting to overcome the above constraints we need to motivate those types of representations and algorithms which exploit semantically enriched linkages between pattern entities. Our representation of attack pattern feature maps supports feature map algorithms whereby the weights carry meaning within the empirically observed pattern space. In this way the weight vector connected to a particular node becomes a very informative indicator of the pattern class associated with the said node. This allows the automated extraction of discriminant features from weight vectors for different classes as represented by different nodes, for example using an ID3 type of algorithm to produce NIDS type of rules from such feature maps.

The crucial property we would motivate in any such Feature Map based representation and reasoning over pattern classes is that of incremental learning and evolution of the feature map

on the basis of a single pass training procedure which allows such approaches to yield useful results immediately whilst lending themselves to continual learning.

IDS Feature Map however it is represented (cartographically or symbolically) essentially determines the topological association of intra-segmental and inter-segmental features and their correlations for any given feature sets within the feature system. See Figure 2 for Feature Map Ontology.

In order to make clear exactly what element of the attack pattern-feature space is being referred to at any stage within any inferencing and learning process, it is best to begin specifying the evolving ontological framework that may be used in feature analysis as follows:

### **Feature System (FS)**

This is the conjunction of all features from all pattern spaces formalised as a ranked list ranging from the most significant to the least significant features for each class.

### **Meta Feature (MF)**

This is a composite or higher level feature which may be a primary, secondary or a derived feature but which in any case encapsulates or subsumes other features which are often at the lower level of granularity. A meta-feature essentially describes a feature which is a pattern of association of other features.

### **Atomic Feature (AF)**

This is the lowest level feature which may exist in any pattern space process-specific or packet-specific etc but essentially it resides at the lowest level of abstraction or granularity of the feature space.

### **Symbolic Feature (SF)**

This is a feature to which a label and therefore a symbolic annotation and thereby a meaning has been ascribed and can be semantically processed for example using RDF-based standards such as XML.

### **Sub-Symbolic Feature (SSF)**

This is a feature which exists as binary data (or pixel level information in the image/video domain) and is un-labelled as an entity and at this level primarily exhibits as a frequency distribution in the spatio-temporal pattern space e.g. a 1 or 0 or an un-labelled sequence of such elements in the packet stream.

### **Key Feature (KF)**

These are features which are dominant in the sense that once spotted in a pattern space they tend to reduce the search space branching factors significantly and essentially influence the expected truth-value of other features thus being significantly indicative of the expected co-occurrence or absence of other features. The key features are also sometimes referred to as primary features and it is expected that there is a direct mapping from key features to expected pattern and feature space context and scope and process .i.e. the value of a key feature, can determine:

- The existence/absence of an attack
- The sub-set of attack types or broad classificatory class of the attack

- connectors' orchestration scenario or plan/intention including intending-/preparing to attack, is -attacking, suspect, benign, idiosyncratic
- Analysis context (cache) scoping and serving/access required
- Type of applicable data analysis window(s) (e.g. discrete, sliding or dynamically-spanning in searching for co-occurring features to establish confirmatory evidence as to the attack type and thus to achieve convergence on the resolution process for detection and identification of attack type).
  - Applicable data analysis window range values.

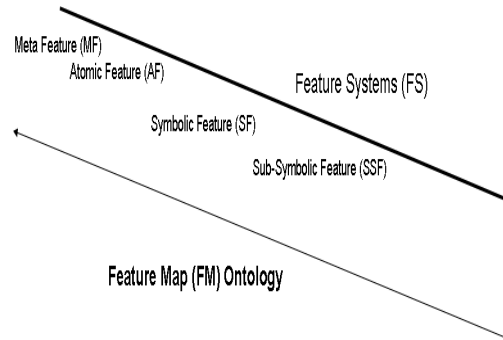


Figure 2: Feature Map Ontology Representation.

Accordingly Key Features can often include Processual features such as (e.g. TCP state-flag, source port, destination port, timestamp, payload (existential), Linearly Independent or Orthogonal Features/ feature sets (LIF), Linearly Dependent Features (LDF), Most Significant Features (MSF), Least Significant Features (LSF), Pattern or Feature-Space Segment (PFSS), Intra-Segmental Feature (ISF) and Inter-Segmental Feature (XSF).

#### **Linearly Independent or Orthogonal Features/ feature sets (LIF)**

These are those features or feature sets whose values are weakly-coupled or de-coupled from each other so that the presence, absence and/or value of one is expected to have little or no influence on the same properties for the other.

#### **Linearly Dependent Features (LDF)**

These are those features or feature sets whose values are strongly-coupled to each other so that the presence, absence and/or value of one is expected to influence the properties for the other.

#### **Most Significant Features (MSF)**

These are the highest ranking features for a given class in the sense that their value, once established, (i.e. once their absence or presence is spotted) significantly reduces the branching factor in the search space. This means that these are the most indicative features i.e. they are the most effective and the most robust and reliable features for a given class of attack. Whilst a Key Feature (KF) may be common to more than one class of attack, an MSF is so distinguished because it is most closely indicative of a particular class and typically it is the most reliable feature i.e. most resilient; that is to say least vulnerable to noise-effects such as attack masking, traffic-embedding etc.

**Least Significant Features (LSF)**

These are the lowest ranking and the more highly variable/unstable features for a given class in the sense that their value, once established, (i.e. once their absence or presence is spotted) is not expected to reliably and significantly reduce the branching factor in the search space. This means that these are the least indicative features for a given class i.e. these tend to occur commonly in several attack types and/or be also the least robust and least reliable features for a given class of attack in the sense that they can exhibit widely variable or noisy values and be prone to particular noise-effects such as attack masking, traffic-embedding etc.

**Pattern or Feature-Space Segment (PFSS)**

This is any single entity such as binary data or blocks of such data which may or may not be contiguous or co-located within the same packet or block or analysis level but which for the purpose of feature extraction can be usefully grouped together as an actual or virtual segment in the spatio-temporal pattern space (e.g. certain packet-stream and/or snort elements).

**Intra-Segmental Feature (ISF)**

This is a feature which resides within an actual or virtual segment in the spatio-temporal pattern space (e.g. within a packet stream or a snort element).

**Inter-Segmental Feature (XSF)**

This is a feature, often a meta-feature or derived feature, which signifies a certain association and correlation e.g. co-occurrences across the feature space which may involve one or more segments.

**4. NIDS DOMAIN FEATURE EXTRACTION**

Feature selection from the available data is vital to the effectiveness of the methods employed. Traditional feature selection techniques cannot be directly applied here since they fail to consider (across record boundaries) the sequential correlation of features selected across dynamically adaptive ranges, so the techniques used have to include heuristics to provide for a capability. Connection records provide numerous features that are intrinsic to each connection. These features are summarised in Table 1 below. The timestamp, source address and port, destination address and port, and protocol, uniquely identify a connection, making them essential attributes.

| <b>Essential Attributes of Network Connections</b> |
|--|
| Protocol   |
| Destination port                                   |
| Source port  |
| Destination IP                                     |
| Source IP  |
| Timestamp  |

*Table 1: Essential Attributes of Network Connections.*

They maintain that “*association rules should describe patterns related to the essential attributes*”. Specifically, at least one of those attributes must be present in the antecedent of a rule in order for that rule to be useful. They call this the axis attribute for the rule. For

example, a rule that is based solely on the number of bytes transferred really does not convey any useful information. Likewise, if the value of some feature must be kept constant through the processing of a set of records (for instance, the destination host), that feature is called a reference attribute.

Other researchers also had success with this approach. They found that their best results were achieved when they limited their rules to only using a key consisting of the source IP, destination IP, and the destination port, which they called the sdp Key.

Researchers used the same approach, although they assigned all connections with unassigned privileged ports to one service group, and all connections with unassigned non-privileged ports to another group.

While essential attributes as the Most Significant Features (MSFs) provide vital information about connections and transactions, most research uses some of the secondary attributes, such as connection duration, TCP flags and the volume of data passed in each direction, as shown in Table 2 below.

| <b>Secondary Attributes of Network Connections</b>          |
|---|
| Percentage of control packets                               |
| Percentage of data packets                                  |
| wrong data packet size rate                                 |
| Hole rate   |
| Duplicate ACK rate  |
| Wrong re-sent rate  |
| Re-sent rate  |
| Number of urgent  |
| Number of wrong flag  |
| Land packet   |
| TCP Flags   |
| Destination bytes   |
| Source bytes  |
| Duration  |
| Number of same connection attempts in given temporal window |

*Table 2: Secondary Attributes of Network Connections.*

Unfortunately, the intrinsic attributes of a connection are insufficient to provide an adequate detector performance against most attacks. It is interesting that including temporal information with each data point significantly increased accuracy and most research since has corroborated this finding.

Temporal information is captured in the form of calculated or Derived Attributes (DA). For example a DA may be one that provides the average value of an attribute, or the count or percentage of connections fulfilling some criteria over the last  $w$  seconds, or  $n$  connections. Some example DAs are listed in Table 3.

| <b>Calculated Features (Derived Attributes) of Network Connections</b> |
|--|
| Percentage of connections involving the same service to the same host  |

|   |
|---|
| Percentage of connections involving the same host to the same service |
| Bytes transferred with current host                                   |
| Bytes transferred with current service                                |
| Bytes transferred on all services                                     |
| Average duration with current host                                    |
| Average duration with current service                                 |
| Average duration on all services                                      |
| Variance of packet count to keys                                      |
| Number of keys with outside hosts                                     |
| Number of new keys  |
| Number of unique keys   |
| Number of ICMP packets  |
| Number of packets to all services                                     |
| Number of different services accessed                                 |
| Number of connections involving the same host                         |
| Number of connections involving the same service                      |
| Number of connections involving specific services                     |
| Number of connections to unprivileged services                        |
| Number of connections to privileged services                          |
| Number of FIN flags   |
| Number of RST flags   |
| Number of other errors  |
| Number of SYN flags   |
| Number of establishment errors  |
| Number of total connections   |

*Table 3: Derived Attributes of Network Connections.*

For instance, if the time window is too long, the algorithms may miss some rules that can be profitably fired (are hot) only in a short period of time. On the other hand, if the time window is too small, they may miss the rules that span a long time period in a slow manner. Researchers have used two time windows: a three second window to examine features that are highly correlated across a shorter time span and a one day window to examine features that are correlated over a longer time period.

There are numerous techniques to identify which of the secondary or calculated attributes provide the best feature set for a given method. Researchers used the following techniques to rank these types of attributes:

- Backwards sequential search.
- Beam search.

- Random generation plus sequential selection.

The backward sequential search begins with the full set of features. At each stage of the search, each feature in the remaining set is removed. The best feature to eliminate from the set is determined by comparing the error rates of the classifiers created using the resulting feature set. This algorithm which amounts to a kind of brute force ablation of features runs in polynomial time.

Beam Search is a type of best-first search which uses a bounded queue to limit the scope of the search. The queue is ordered from best-state to worst-state, with the best state placed at the front of the queue. The algorithm operates by taking the first state in the queue, the most promising state, and extending the search from that state as in Backward Sequential Search. Each new state visited is placed in the queue in order of the goodness of the state. If there is no limit on the length of the queue, then Beam Search takes exponential time to complete but if the queue length is 1, then Beam Search takes polynomial time. Thus accuracy can be increased if the queue length is increased.

In the third approach, namely Random Generation plus Sequential Selection, several sequential selections from different places in the search space are performed. The goal is to avoid picking the first good feature set seen on the assumption that other good feature sets are also available. To do so, a random feature set is generated, then backward and forward sequential selection on the state are performed. Random Generation runs in polynomial time but is more expensive than a Backward Sequential Search.

Both Dynamic Mining and Domain Level Mining algorithms use a single size time window, and it is non-trivial to compute an optimum window size that can exploit all kinds of rules that may be invocable at different frequencies in a given data (feature) context. See figure 3 for Dynamic Feature Optimization for Feature Map development.

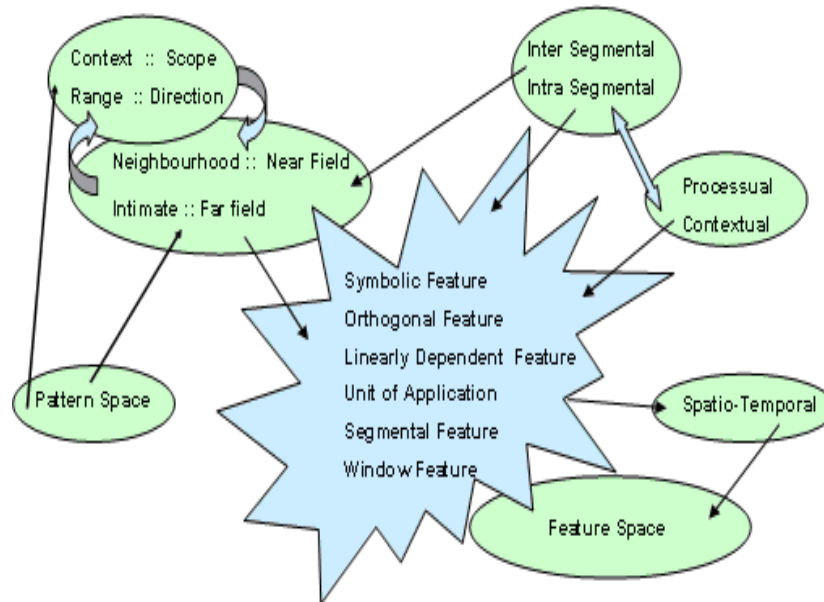


Figure 3: Dynamic Optimal Feature Map development.



### 5. MULTI-MODEL/DATA VIEW/LEVEL PACKET PATTERN SPACE PROCESSING

It is clear that there is an opportunity to innovate additional mechanisms to build on the existing NIDS technology by way of establishing a capability for maintaining and refining dynamic feature maps and exploiting the adaptive topological refinement of such maps to supply the (near) real-time processing arena with the most optimal feature templates at each refresh cycle. This will have to be predicated on an ability to efficiently traverse the feature map hierarchy and to be able to build semantic bridges between the various feature spaces at different levels of abstraction above and below the symbolic – sub-symbolic boundaries of the pattern space i.e. at the packet stream level or at the signature and snort attributes level and across the two combined.

It is therefore proposed that the open standard of XML-based processing as mobilised by the rich representation of Topic Maps [ISO/IEC 13250] is exploited at all levels as far as possible in order to represent, traverse and reason over feature map hierarchies within the process of recognition and dynamic learning and refinement of the feature space templates and therefore dynamic enhancement of the detection and identification of attacks as well as for enhanced adaptive alarm filtering. See figure 5 for Multi-model/data view/level Packet Pattern Space Processing on Topic Maps [ISO/IEC 13250] concept

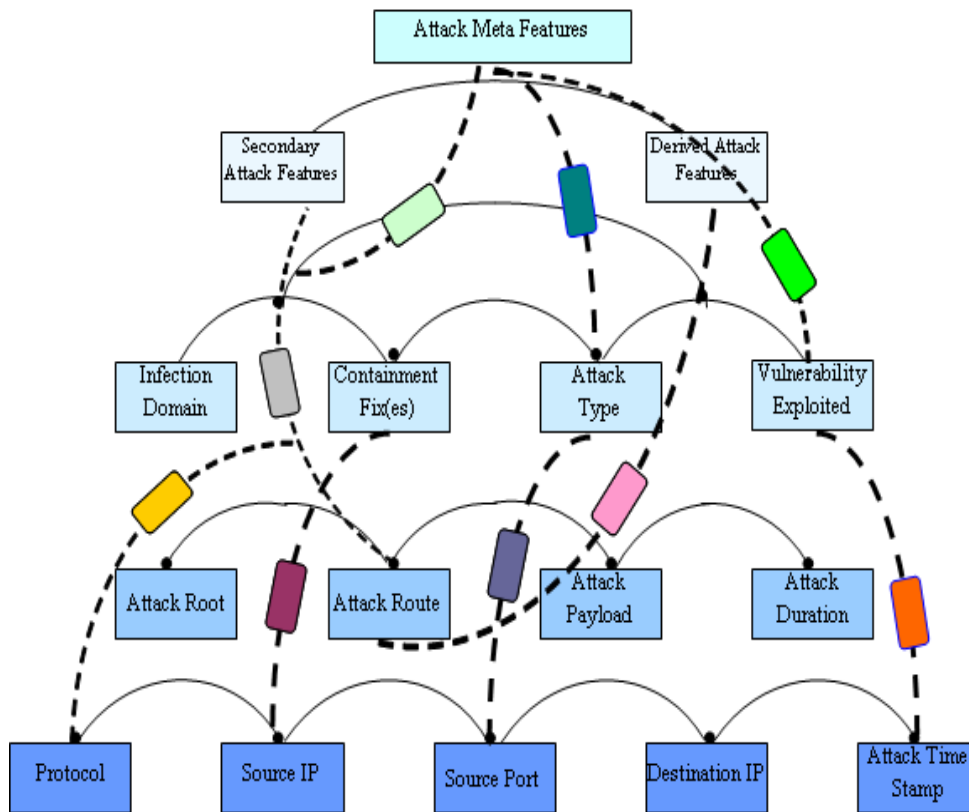


Figure 5: Mapping NIDS FS Ontology onto Topic Map.

Accordingly we shall review the essential concepts of the Topic Map Technology which we will use in our approach to the formalisation of the NIDS domain symbolic feature maps to be exploited as topic map-enabled, revisable and dynamically (data driven) refined feature maps

As already mentioned we envisage the framework architecture to exhibit the potential that will lend itself to incrementally more powerful detection and deliberation capability as may be conferred by an integration of the Bayesian Belief Network (BBN). This can be used to endorse the feature map probabilistically thereby permitting advanced off-line inferences to inform the periodic refinement process. However, this enhancement falls outside the scope of the present specification.

## **6. Mapping NIDS Feature System Ontology onto Topic Map Technology Constructs**

According to [ISO/IEC 13250], a topic Map is an SGML or XML document (or set of documents) in which different element types, derived from a base set of architectural forms, are used to represent topics, occurrences of topics, and relationships (or “associations”) between topics.

A topic, in its most generic sense, can be any “thing” e.g. a malware attack incident, /exploit, a vulnerability, a fix, a packet or pattern space segment thereof, a feature space/map/system etc; i.e. any entity, any concept, anything – regardless of whether it exists or has any other specific characteristics, about which anything whatsoever may be asserted from any viewpoint whatsoever.

As a general rule of thumb, the topics should be anything that may require re-visiting and possible modification. Normally the topics are automatically detected using an indexing tool. Appropriate metadata should be already available for the topics. Normally, such metadata will be descriptive or content-metadata and its details will vary according to the type of topic. The topics can be considered as the feature names, or any aspect of any feature, its associations with any other feature/context, its occurrences, roles, etc. This will form the ontology network for the NSPES and can additionally be exploited to provide attack phenomenology/ cartographic or contagion/infection maps of various formats to enable powerful SNMP level visualisations.

TopicMaps.Org is an independent consortium of parties developing the applicability of the topic map paradigm [ISO13250] to the World Wide Web by leveraging the XML family of specifications.

The specification describes version 1.0 of XML Topic Maps (XTM) 1.0 [XTM], an abstract model and XML grammar for interchanging Web-based topic maps, written by the members of the TopicMaps.Org Authoring Group. More information on XTM and TopicMaps.Org is available at <http://www.topicmaps.org/about.html>. All versions of the XTM Specification are permanently licensed to the public, as provided by the Charter of TopicMaps.Org [XML Topic Maps (XTM) 1.0].

The TopicMap element type is declared as in the figure 6:

```

<!ELEMENT topicMap
 ( topic | association | mergeMap )*
>
<!ATTLIST topicMap
  id          ID          #IMPLIED
  xmlns       CDATA       #FIXED 'http://www.topicmaps.org/xtm/1.0'
  xmlns:xlink CDATA       #FIXED 'http://www.w3.org/1999/xlink'
  xml:base    CDATA       #IMPLIED
>

```

Figure 6: The topic Map element type.

The scope element type is used throughout XTM to indicate the scope of a topic characteristic assignment. The scope element type is declared as in figure 7:

```

<!ELEMENT scope
 ( topicRef | resourceRef | subjectIndicatorRef )+
>
<!ATTLIST scope
  id          ID          #IMPLIED
>

```

Figure 7: The topic Map element type.

The association element type is used to express associations between topics. The member child elements provide the association role players of the association.

The association element type is declared as in figure 8:

```

<!ELEMENT association
 ( instanceOf?, scope?, member+ )
>
<!ATTLIST association
  id          ID          #IMPLIED
>

```

Figure 8: The topic Map element type.

An XTM topic map is a topic map serialised in XTM syntax as a Topic Map element with descendants. The topic Map element is the root element of all XTM topic maps. It acts as a container for the topic map, and can be either the document element of an XML document, or it may be the root of a sub tree inside an XML document that contains more than just a single topic map. In both cases, the input to the XTM deserialisation process is the subtree contained by the Topic Map element.

An XTM document is an XML document that contains one or more XTM topic maps. In a process known as deserialisation, the XTM topic map is read by a topic map processor, which produces from it some representation of the Standard Application Model, by following a procedure equivalent to the one defined in this specification.

## **7. Classification NIDS attacks on the topic map**

It is important to form a topic map example case which will help us to understand how we can use topic map for this paper. Following the topic map standard and design specification, a fundamental topic map for the core classification of different types of NIDS attacks has been generated. This topic map will help us to understand the basic classes and types of NIDS attacks, subclasses of any type of attack, features related to a specific type of attack or a specific attack, and finally associations among all the elements of the topic map. In figure 9, we have shown the basic concept of the transformation of the “feature Table” to a feature based topic map.

The topic map shown in the figure 9 is based on different types of attack classification and sub-classifications. Blue arrows are indications of the hierarchy of the classes and specific association between an attack class and its sub-classes. As from the definition, in a topic map each element is a topic.

The figure 10 below presents one possible generalisation hierarchy of the network security attack typology and feature systems. So each block in that figure is a topic with specific features, whereas arrows are also topics with specific directional and associative information as key features. Our next task is to populate this topic map to the deep lower level with more classes as the whole example become easily understandable.

Clearly in order to devise an effective real-time system for detecting, locating and recognising network security attacks as well as to learn from the dynamically evolving pattern of attacks in order to proactively inform detection strategy choices and responsive security policy, a prerequisite is the study of the typology of network security attacks and their pattern spaces. As we can see in figure 1 attack phenomena present a wide spectrum of pattern spaces which admit multiple data-views which must remain accessible for inferencing by detection algorithms in order to allow efficient and accurate recognition.

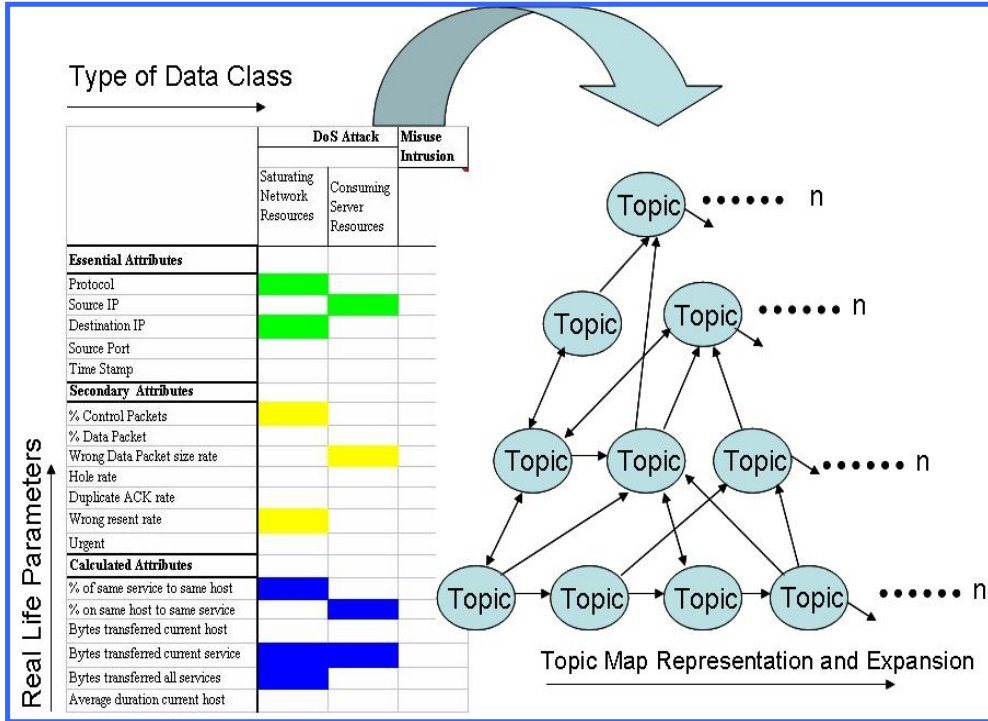


Figure 9: Conversion of Feature Matrix to Topic Map Representation.

In the next stage of this topic map we have incorporated the features relevant to a specific attack and populated it with more specific lower level information. Importance of the topic map for our application could be realised from figure 11 in conjunction with figure 10.

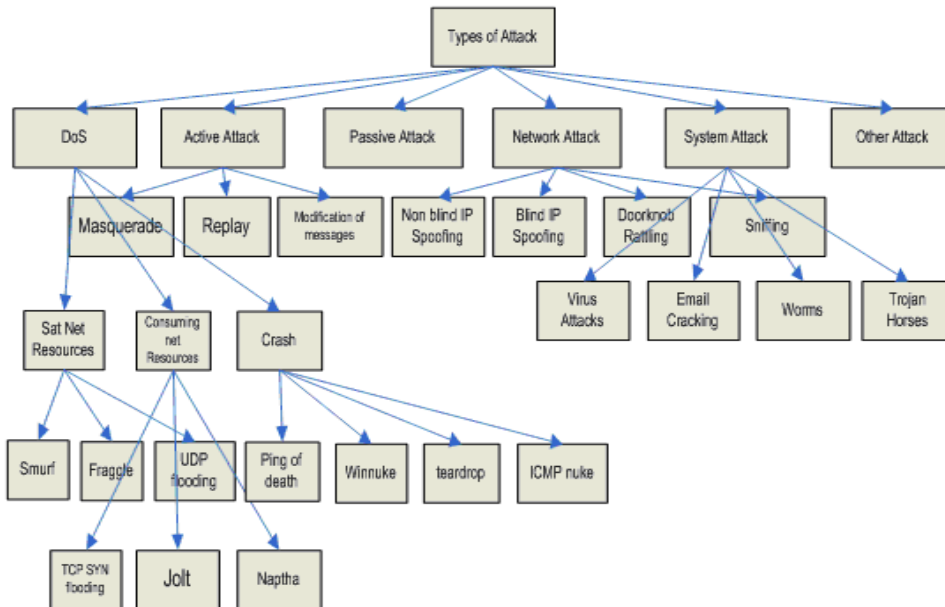


Figure 10: One possible generalisation hierarchy of the network security attack typology and feature systems.

In figure 11, UML based topic 'Type of Attack' has a class association with the topic 'DoS'; where 'Consuming Net Resource' and 'Crash' are two topics which are associated with 'DoS' as sub-classes. If we populate the 'Consuming Net Resource' topic, one of the sub-categories under this topic would be 'TCP SYN Flooding' topic, whereas 'Ping of Death' is a sub-topic of the 'Crash'. Finally, from 'TCP SYN Flooding' and 'Ping of Death' topics we can draw associations with specific attributes relevant to the type of attack.

### **Representing Security Attacks Feature System:**

As we have seen attacks are phenomena involving variable formats, protocols, payload type, roots, routes, processes, contagion trajectories, targets, execution modes and mechanisms, consequences etc.

For example the wide spectrum of attack types and features that are dynamically evolving with new attacks can have their features pattern space represented from an application-centric or network-centric or other viewpoint. The required multi-dimensional feature system requires a pattern representation that is sufficiently rich to enable the various sub-spaces of the attacks feature systems to be represented from different feature-views thus allowing all the significant features of the pattern space to be encoded, updated and reasoned over in efficient computation for real-time or offline attack detection inferencing and learning. The following UML feature Map is just one example of how this can be achieved using the topic map enabled feature representation to be developed.

This section provides a model and grammar for representing the structure of information resources used to define UML topics, and the associations (relationships) between UML topics. Names, resources, and relationships are said to be characteristics of abstract subjects, which are called topics. Topics have their characteristics within scopes: i.e. the limited contexts within which the names and resources are regarded as their name, resource, and relationship characteristics. One or more interrelated documents employing this grammar are called a "topic map."

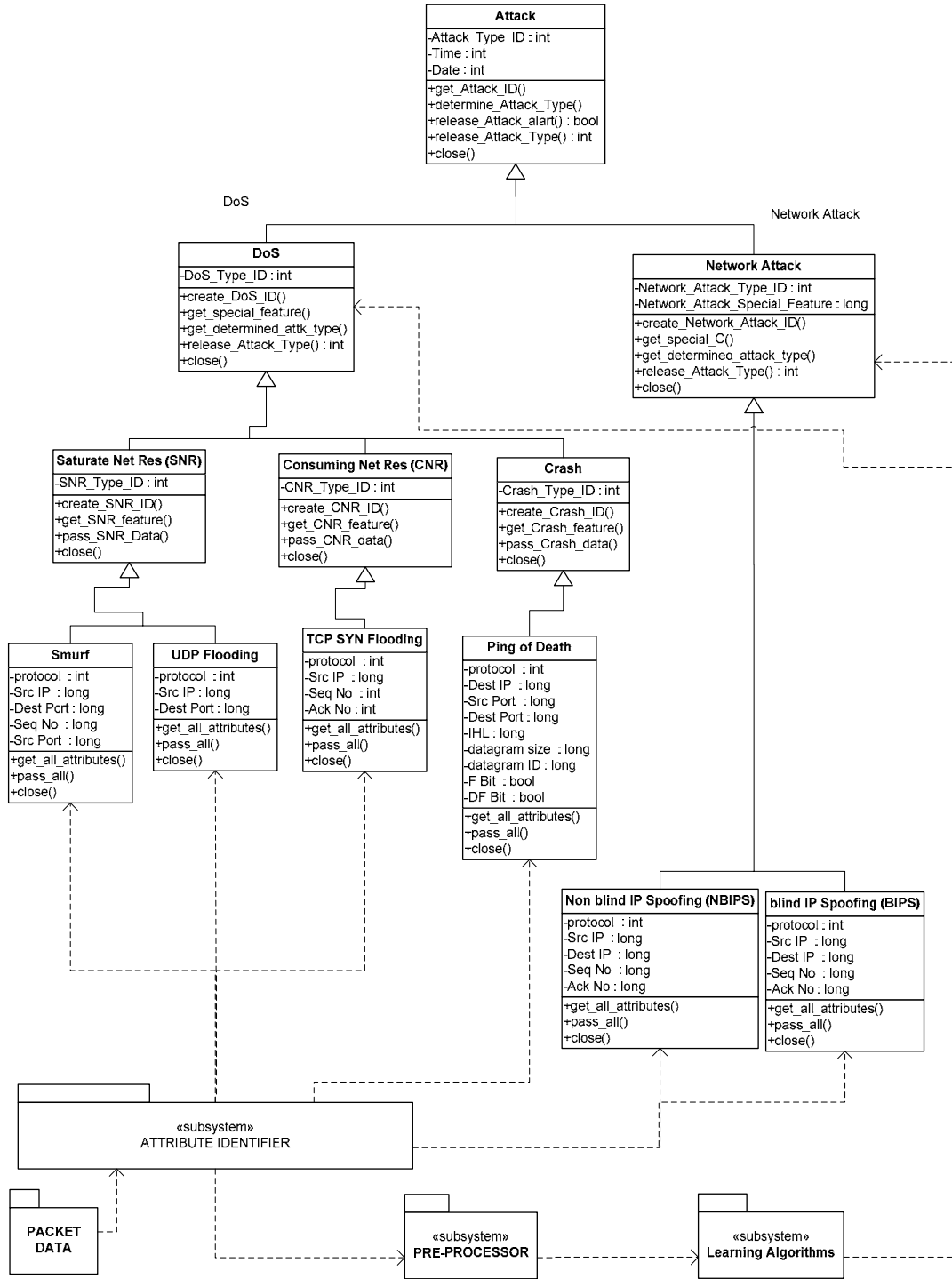


Figure 11: Populating NIDS attacks on the UML based topic map based on features.

## 8. Requirement and Development of an Algorithm Map

After developing a ‘Feature Map’ mapped on a Topic Map architecture it is now important that the algorithm map would be developed depending on the specific requirements of the pattern recognition problem, measures of problem complexity, related feature space and

feature optimisation. This approach will consider and combine all available and best suitable algorithms to form a map, which eventually serve as a key hybrid multi-directional map to a particular NIDS pattern recognition problem and to reach the best efficient performance. In mathematics and computing, an algorithm is a procedure (a finite set of well-defined instructions) for accomplishing some task which, given an initial state, will terminate in a defined end-state. The computational complexity and efficient implementation of the algorithm are important in computing, and this depends on suitable data structures. Informally, the concept of an algorithm is often illustrated by the example of a recipe, although many algorithms are much more complex; algorithms often have steps that repeat (iterate) or require decisions (such as logic or comparison). Algorithms can be composed to create more complex algorithms.

Automating the learning process requires certain intellectual information to initialise it. This information falls naturally into three basic elements:

- A suitable problem representation
- A search strategy performance measures

A representation includes the basic functions, operators, and data structures that effectively characterise the solution space or possible range of attainable solutions. Search strategies define the rules or techniques for manipulating the representation and navigating the search space.

The many algorithms for data clustering developed in recent decades all face a major challenge in scaling up to very large database sizes. Clustering algorithms with quadratic (or higher order) computational complexity, such as agglomerative algorithms, do not scale up. Even for more efficient algorithms, such as K-Means and Expectation-Maximization (EM), which have linear cost per iteration, research is needed to improve their scalability for ever growing data sets.

The final choice of algorithms is heavily dependent on their optimisation capabilities. To find the most efficient and best suitable sets algorithms an estimation methodology based on ten parametric factor measurements is proposed. See figure 12 for the concept of the formation of the algorithm map. These factors are listed below:

1. Total Throughput
2. Scalability
3. Specialisation
4. Parallelisability
5. Serial Processing
6. Speed Accuracy
7. Bi-directionality
8. Re-configurability
9. Data Dependency
10. Re-trainability

One particular NIDS attack classification problem could not solved by using a particular algorithm, as all algorithms have got advantages and disadvantages. Algorithmic limitations would restrict the whole process to find an optimal solution. So it is necessary to have hybrid algorithmic approaches in the system modelling where system can use multiple algorithms simultaneously (and in parallel) to overcome the limitation due to single algorithm.

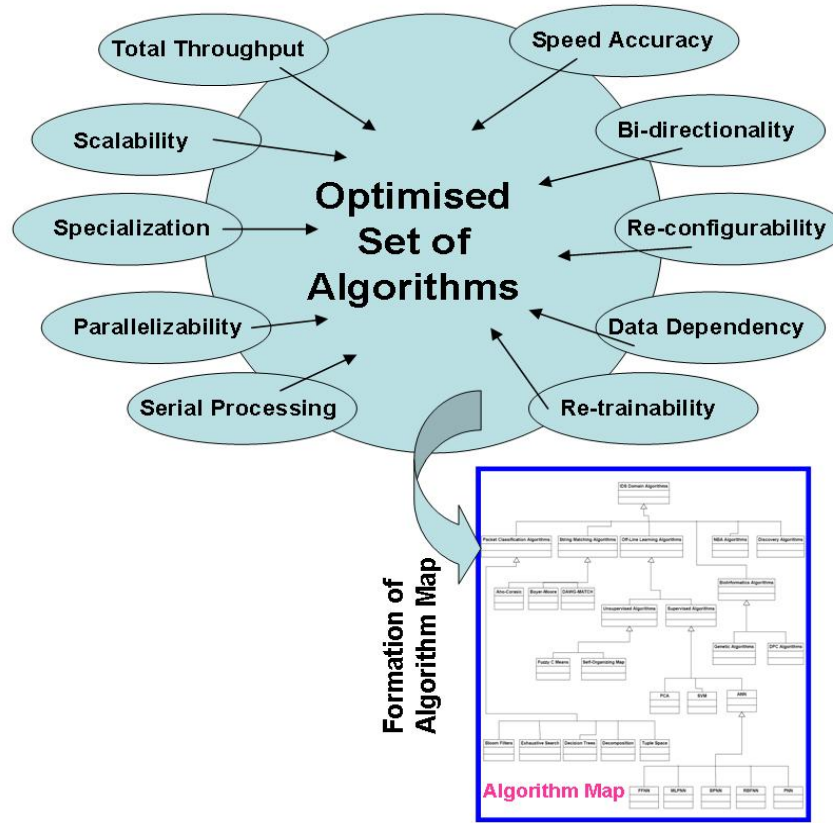


Figure 12: Formation of Algorithm Map.

Depending on these ten optimisation factors an algorithm map (based on topic map technology) would be developed for NIDS pattern recognition problems where all advantages of all possible algorithms would be considered. This Algorithm map would be able to find a suitable set of algorithms, combination of which can deal with the dynamic feature maps more efficiently.

For this modelling work in this paper six different types of algorithms have been identified depending on the sequential strategic algorithmic tasks of the NIDS pattern recognition problems. They are as follows:

1. Packet classification algorithms
2. String matching algorithms
3. Correlation and network behaviour analysis algorithms
4. Learning algorithms
5. Bio-informatically inspired or so-called natural algorithms
6. Discovery algorithms

In the algorithm map these six types of algorithms would form the generalised algorithm class for the NIDS domain. In the next stage sub-classes of a type of algorithm would form that particular type class of algorithm. In figure 13 branches and sub-branches of algorithm map for NIDS domain are depicted. For example 'supervised algorithms' and 'unsupervised algorithms' form the 'Off-line algorithms' class. 'PCA', 'SVM' and 'ANN' form the 'supervised algorithms' class and finally 'FFNN', 'MLPNN', 'BPNN', 'RBFNN' and 'PNN'

will form the ‘ANN’ class. Based on the feature map and pattern recognition requirement an optimal algorithmic path would be highlighted in the algorithm map. The whole process of NIDS pattern recognition would follow that optimal algorithmic path. The whole algorithm map can be used to visualise the segmental algorithmic processing requirements for a particular pattern recognition problem.

In this section we examine a range of algorithm classes which constitute the space of candidate algorithms from which certain specific algorithms have been foregrounded as primary candidates for optimisation studies against the testbed and final choice making and fine tuning which can only be done once the initial prototype of the framework architecture has been implemented leaving degrees of freedom open in its architecting such that it will be capable of being fine tuned and particularised for the most optimal execution of specific algorithms.

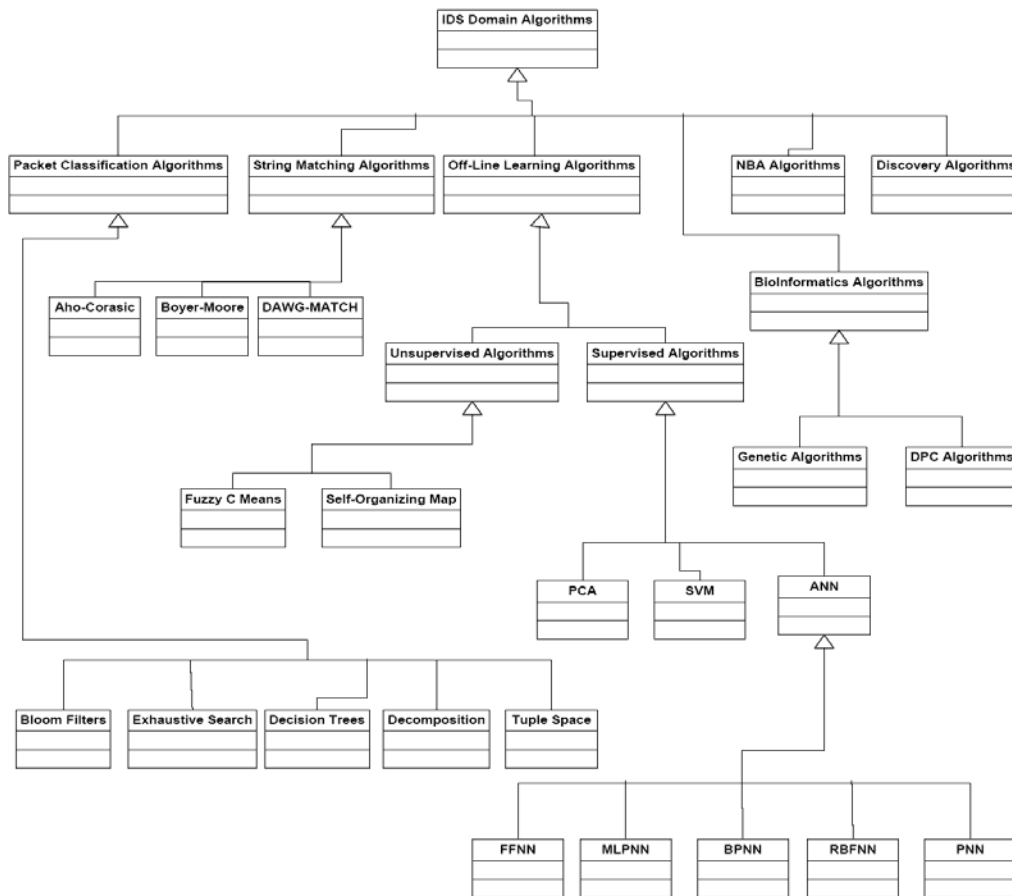


Figure 13: Algorithm Map: Populating NIDS Domain algorithms on the UML based Algorithm Map.

### 7. Conclusion

We have described a proposed new approach to NIDS domain knowledge processing focused on a topic map technology-enabled representation of features of the threat pattern space as well as the knowledge of situated efficacy of alternative candidate algorithms for pattern recognition within the NIDS domain. Thus an integrative knowledge representation framework for virtualisation, data intelligence and learning loop architecting in the NIDS

domain has been described together with specific aspects of its deployment. We believe this to offer a powerful scalable framework for evolving, sharing and thus continuous refinement of the critical knowledge that can be brought to bear on the NIDS domain with its highly dynamic pattern space.

## 8. Acknowledgments

This work has been supported by the European Community Research Programme under the FastMatch project which is partially supported by the European Community under the Information Society Technologies (IST) priority of the 6th Framework Programme for R&D (IST- IST-27095 ,[www.fastmatch.org](http://www.fastmatch.org)) Thanks are due to all FastMatch project partners and participants, for their collaboration. We acknowledge Rittaban Datta's assistance in production of some of the illustrations on this paper.

## 9. Reference

A. Hoekstra, R.P.W. Duin, "On the nonlinearity of pattern classifiers", Proc. of the 13th ICPR, Vienna, August 1996, D271-275.

S. Mitra, S.K. Pal and P. Mitra, "Data mining in soft computing framework: a survey", IEEE Transactions on Neural Networks, vol. 13, no. 1, 2002, 3-14.

George Forman and Bin Zhang, "Linear Speed-Up for a Parallel Non-Approximate Recasting of Center-Based Clustering Algorithms, including K-Means, K-Harmonic Means, and EM", ACM SIGKDD Workshop on Distributed and Parallel Knowledge Discovery, KDD-2000, Boston, MA, August 20, 2000.

Tin Kam Ho and Mitra Basu, "Measuring the Complexity of Classification Problems", 0-7695-0750-6/00, 2000 IEEE.

M. Li, P. Vitanyi, "An Introduction to Kolmogorov Complexity and Its Applications", Springer-Verlag, 1993.

Louis A. Tamburino, Mithael A. Zmuda, and Mateen M. Rizki, Generating, "Pattern Recognition Systems Using Evolutionary Learning", Evolutionary Processing, 2000 IEEE.

[Wikipedia] [http://en.wikipedia.org/wiki/Pattern\\_recognition](http://en.wikipedia.org/wiki/Pattern_recognition) (December 2006).

A.K. Jain, R.P.W. Duin and J. Mao, "Statistical pattern recognition: a review," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 22, no. 1, 2000, 4-37.

T.Y. Kwok and D.Y. Yeung, "Constructive algorithms for structure learning in feedforward neural networks for regression problems", IEEE Transactions on Neural Networks, vol. 8, no. 3, 1997, 630-645.

A.Badii, "From Malware Pattern-State-Spaces to NIDS Attacks Feature Map" MONAM06 Panel Presentation, Tübingen, 27 September 2006.

Dominique Alessandri, "Using Rule-Based Activity Descriptions to Evaluate Intrusion-Detection Systems", IBM Research Laboratory Zurich, Switzerland, October 2000.

M. Esposito, C. Mazzariello, F. Oliviero<sup>1</sup>, S.P. Romano and C. Sansone, "Evaluating Pattern Recognition Techniques in Intrusion Detection Systems", Research outlined in this paper is partially funded by the Italian "Ministero dell'Istruzione, dell'Universit`a e della Ricerca (MIUR)" in the framework of the FIRB Project "Middleware for advanced services over large-scale, wired-wireless distributed systems (WEB-MINDS)"

Stolfo J., Fan W., Lee W., Prodromidis A. and Chan P.K., "Cost-based Modeling and Evaluation for Data Mining with Application to Fraud and Intrusion Detection", DARPA Information Survivability Conference, 2000.

J. Luo, S. M. Bridges, "Mining Fuzzy Association Rules and Fuzzy Frequency Episodes for Intrusion Detection", International Journal of Intelligent Systems, John Wiley & Sons, pp. 687-703, 2000.

J Li, Guo-yin Zhang, Guo-chang Gu, "The research and implementation of intelligent intrusion detection system based on artificial neural network", Proceedings of the Third International Conference on Machine Learning and Cybernetics, Shanghai, 26-29, August 2004.

J. Allen, A. Christie, W. Fithen, J. McHugh, J. Pickel, and E. Stoner, "State of the Practice of Intrusion Detection Technologies", Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 1999.

Aikaterini Mitrokotsa and Christos Douligeris, "Detecting Denial of Service Attacks Using Emergent Self-Organizing Maps", IEEE International Symposium on Signal Processing and Information Technology, 2005.

[idealliance.org] <http://www.idealliance.org/europe/04/call/xmlpapers/03-03-03.91/.03-03-03.html> (December 2006).

Lars Kulik, Matt Duckham, and Max Egenhofer, "Ontology-Driven Map Generalization", Journal of Visual Languages and Computing 16 (3): 245-267, 2005.

[networkedplanet] <http://www.networkedplanet.com/products/episerver-module.html> (December 2006).

Jianqiang Xin, John E. Dickerson, and Julie A. Dickerson, "Fuzzy Feature Extraction and Visualization for Intrusion Detection", IEEE 0-7803-7810-5/03, The IEEE international Conference on Fuzzy Systems, 2003.

Jeremy Frank, "Artificial Intelligence and Intrusion Detection: Current and Future Directions", IEEE 1994.

D. Denning, "An Intrusion Detection Model", IEEE Transactions on Software Engineering, Vol. SE 31, No. 2, 1987.

[TopicMaps] <http://www.topicmaps.org/> (December 2006).

T. Trautmann and T. Deïceux, "Comparison of dynamic feature map models for environmental monitoring", 0-7803-2768-3/95/, IEEE International Conference on Neural Networks, 1995.

R. Kamimura and T. Kamimura, “Dynamic Feature Extraction by Greedy Information Acquisition Algorithm”, Proceeding (362) Artificial Intelligence and Applications – 2002, ACTA Press.

[XML Topic Maps (XTM) 1.0] <http://www.topicmaps.org/xtm/> (December 2006).