Big Data and Semantics Management System for Computer Networks

Bassem Mokhtar*
Department of Electrical Engineering
Alexandria University, Egypt

Mohamed Eltoweissy*
Department of Computer and Information Sciences
Virginia Military Institute, USA

Abstract—We define “Big Networks” as those that generate big data and can benefit from big data management in their operations. Examples of big networks include the current Internet and the emerging Internet of things and social networks. The ever-increasing scale, complexity and heterogeneity of the Internet make it harder to discover emergent and anomalous behavior in the network traffic. We hypothesize that endowing the otherwise semantically-oblivious Internet with “memory” management mimicking the human memory functionalities would help advance the Internet capability to learn, conceptualize and effectively and efficiently store traffic data and behavior, and to more accurately predict future events. Inspired by the functionalities of human memory, we proposed a distributed network memory management system termed NetMem, to efficiently store Internet data and extract and utilize traffic semantics in matching and prediction processes. In particular, we explore Hidden Markov Models (HMM), Latent Dirichlet Allocation (LDA), and simple statistical analysis-based techniques for semantic reasoning in NetMem. Additionally, we propose a hybrid intelligence technique for semantic reasoning integrating LDA and HMM to extract network semantics based on learning patterns and features with syntax and semantic dependencies. We also utilize locality sensitive hashing for reducing dimensionality. Our simulation study using real network traffic demonstrates the benefits of NetMem and highlights the advantages and limitations of the aforementioned techniques.

Keywords- Network Management, Big Data, Bio-inspired Design, Semantics Reasoning, Pattern Learning, Hybrid Intelligence

1. Introduction

Due to semantically-oblivious protocol operations, the current Internet cannot effectively or efficiently cope with the explosion in services with different requirements, number of users, resource heterogeneity, and widely varied user, application and system dynamics [1]. This is leading to increasing complexity in Internet management and operations, thus multiplying the challenges to achieve better security, performance and QoS satisfaction. The current Internet largely lacks capabilities to extract network semantics to efficiently build runtime accessible dynamic behavior models of Internet elements at different levels of granularity to pervasively observe, analyze, predict and act upon network dynamics. For example, a network host might know the role of TCP, however, it might not know the behavior of TCP in mobile ad hoc networks. We refer to the limited utilization of Internet traffic semantics in networking operations as the “Internet Semantic Gap”.

The current and future internets (e.g., Internet of things [2]) support a massive number of Internet elements with extensive amounts of data. Fortunately these data generally exhibit multi-dimensional patterns (e.g., patterns with dimensions such as time, space, and users) that can be learned to extract network semantics [3]. These semantics can help in learning normal and anomalous behavior of the different elements (e.g., services, protocols, etc.) in the Internet and in building behavior models for those elements accordingly. Recognizing and maintaining semantics as accessible behavior models related to various Internet elements will aid in possessing intelligence thus helping elements in predicting future events (e.g., QoS degradation and attacks) that might occur and affect performance of networking operations. Furthermore, learning behavior of those elements will enhance their self-* properties such as awareness with unfamiliar services and also advance reasoning about their behavior. For instance, a router can classify a new

* Corresponding author. Tel.: (+2) 010-233-300-38
Email address: bmokhtar@alexu.edu.eg (B Mokhtar)

* The author is also affiliated with the ECE department at Virginia Tech and University of Arizona, USA
running service in a network as a specific type of TCP-based file transfer service when it finds similarity between behavior of that new service and that of an already known service.

There is a need to endow Internet operations and running services and applications with intelligence to mitigate the Internet semantic gap. In the literature, Internet (or network) intelligence (referred to here as InetIntel) is defined as the capability of Internet elements to understand network semantics to be able to make effective decisions and use resources efficiently [4]. InetIntel has to support Internet elements with the capability for learning normal and dynamic/emergent behavior of various elements and in turn building dynamic behavior models of those elements.

InetIntel can be achieved via employing intelligence techniques to efficiently reason about semantics from tons of Internet traffic raw data and provide runtime accessible valuable information at different levels of granularity. InetIntel should be achieved in a way that will not negatively impact Internet robustness or scalability. InetIntel systems may use either monolithic or hybrid intelligence techniques (HIT). Each implemented technique has its mechanisms for learning data patterns, extracting features and reasoning about data semantics. The Internet has tremendous and ever-growing scale. It is noisy and dynamic with dissimilar communicating networks and heterogeneous entities, running diverse services and resources. Accordingly, generated and transmitted Internet data have special characteristics such as massive volume with high- and multi-dimensionality that might negatively affect the performance of intelligence techniques. HIT are being investigated to better mitigate that challenge [5]. For example, in [6], comparison among various InetIntel techniques for intrusion detection systems showed HIT’s superiority in achieving higher detection accuracy. Some works (e.g. [7, 8]) proposed HIT for InetIntel using neural network (NN) with evolutionary algorithms. But, they did not provide solutions for extracting semantics from high data volume with large number of attributes mitigating NN’s challenges in their computed design and long processing time at large scale problems.

In [9-11], we proposed a preliminary design of a network memory system, termed NetMem, to support smarter networking and InetIntel. NetMem design is inspired by the functionalities of human memory [12], which maintains conceptual models that describe associative concepts according to learned multi-dimensional patterns of data captured from the outside world through human sensory system. Those models are updated continually and used for learning novel things and predicting future events achieving human intelligence. Analog with human memory’s functionalities, NetMem has a memory system structure comprising short-term memory (STM) and long-term memory (LTM), STM maintains highly dynamic network data or data semantics with lower levels of abstraction for short-time, while LTM keeps for long-time slower varying semantics with higher levels of abstraction. Maintained data in NetMem can be retrieved at runtime and on-demand to be used in matching and prediction processes within the various networking operations. From a system’s perspective, NetMem can be viewed as an overlay network of distributed “memory” agents, called NMemAgents, located at multiple levels targeting different levels of data abstraction and scalable operation.

In our preliminary design [9-11], NetMem utilized monolithic intelligence techniques, e.g., Hidden Markov Models (HMM) [13], for learning multi-dimensional data patterns to reason about network data semantics and to build conceptual models accordingly. HMM are computationally efficient and well-suited to handle raw raw data and extract inter-related network concepts at different levels of granularity. We classified concepts using the separation of network concerns (application, communication and resource concern) as presented in [14]. Concepts are represented using functional, behavioral, and structural (FBS) engineering design framework [15]. For example, NetMem can learn data semantics concerning TCP and UDP protocols (i.e., communication concerns) within file transfer services (i.e., an application concern) in wireless contexts (i.e., a communication concern). Those semantics can be used to construct a conceptual model, which describe: a) functions (i.e., functional aspect) of TCP and UDP protocols; b) the normal and abnormal behavior (i.e., behavioral aspect) of those protocols and the different behavior classes that can emerge (e.g., different attacks under the abnormal behavior class); c) relations (i.e., structural aspect) among concept classes such as overlapped TCP abnormal behavior classes that share common features.

Network data characteristics include massive volume, high- and multi-dimensionality, dynamicity and complexity (variety in representation models and languages). In [16], authors proposed a generative model based on Latent Dirichlet Allocation (LDA) [17] and HMM for learning words with short-range syntax and long-range semantics dependencies. Consequently, this aids in forming richer ontology with more
associated semantic topics and classes. We surmise that there are similarities between characteristics (e.g.,
huge volume, high dimensionality, and complexity) of datasets in networks and language modeling. In this
paper, we explore different reasoning models for NetMem to learn patterns and extract semantics from big
network data both with and without data dimensionality reduction via locality-sensitive hashing (LSH) algorithm
[18]. We show capabilities of each model in extracting features and learning semantics based on real captured
data by snort [19] and data attributes discovered and classified using associative rule learning (ARL) [20] and
fuzzy membership functions (FMF) [21]. Extracted semantics are represented and classified as concept classes
using the separation of network concerns (application, communication and resource (ACR) concern) as presented
in [14]. Concept classes are represented using functional, behavioral, and structural (FBS) engineering design
framework [15].

Our study for building reasoning models includes HMM, LDA and simple statistical analysis models.
Additionally, we present a HIT for NetMem integrating LDA and HMM for combining the advantages of
both algorithms in efficiently learning patterns of big network data with high- and multi-dimensionality. We
provide a semantic reasoning model for NetMem using the proposed HIT in order to have highly abstracted
and associated network traffic semantics on different levels of granularity and related to various network
concerns. We are motivated in our HIT design by the capability of LDA to discover latent and classified
high-level features from multi-dimensional network data patterns with long-range semantics dependencies.
These classified features will be sequenced to enable semantic reasoning via HMM with higher accuracy.
HMM are structured architectures that are able to predict sequences of semantic topics (related to different
network concerns) based on input sequences of extracted network features by LDA. Depending on input
sequences or pattern of highly-discriminative network data features, HMM with forward and backward
algorithms can learn semantics efficiently showing their FBS aspects.

Some related work (e.g.,[7, 8, 22, 23]) adopted monolithic and hybrid techniques for enhancing
networking operations such as intrusion detection and efficient routing. However, those works were
application-specific and they did not provide customizable application-agnostic way for building ontology of
concept classes. Also, unlike [24, 25], NetMem provides scalable memory storage for network data with
various levels of abstraction for data semantics matching and behavior predictions processes.

The main contributions in this paper are:

- Design of a human-inspired memory management system for big networks with efficient processes for
  extraction of classified high-level features and reasoning about rich semantics;
- Hybrid intelligence technique for efficient and effective reasoning about network semantics; and
- Comparative study of semantic reasoning techniques suitable for big networks, with guidelines highlighting
  their advantages and limitations.

The remainder of this paper is organized as follows. Section 2 presents our proposed methodology for
semantics extraction from big data. Section 3 presents an overview of our human-memory inspired network
memory system. Sections 4, 5 and 6 describe three different techniques for semantics reasoning. Section 7
describes our proposed HIT for semantics management. Section 8 highlights the differences between the
proposed HIT and the monolithic techniques. Section 9 discusses evaluation and the obtained results for
performance of NetMem operations using a case study with real Internet data. Section 10 discuss related work.
Eventually, the conclusion is provided in Section 11 with an outline of future work.

2. Data Semantics Management Methodology

In this subsection, we show our proposed methodology for semantics management for big networks. Our
methodology is realized through the following concepts and operations as shown in Fig. 1:

- **Data virtualization**: Big data are generated with different formats and representation modes from various
  sources in big networks. To have efficient data collection, data virtualization (DV) techniques should be
  used. This would enable data abstraction and federation from different sources besides unified data
  representation. This drives us to functionalities of sensory system in human [12]. That system collects
  huge amount of data from five senses and sends it to the brain via nerve signals, i.e., unified data
  representation,

- **Data features selection**: Big data are high and multi-dimensional. This requires huge storage and
  computation capabilities to analyze patterns of these data. To have efficient big data processing,
  dimension reduction algorithms, with the capability of directive data features selection process, will be
used. Inspired by functionalities of human memory, not all captured data via the sensory system [12] at the moment are abstracted and stored at memory. There are some distinguished features that will be maintained. For example, some of the main features in each restaurant are the existence of tables, chairs, restrooms, etc. (not the locations of restrooms or number and color of chairs and tables).

- **Function-Behavior-Structure (FBS) Data modeling:** representing data uniformly, clarifying functional, behavioral, and structural aspects [24], will facilitate patterns learning and make it smoothly. This is done at human memory that data from various senses are represented as sequences of neurons in the nervous system.

- **Associative storage:** to learn patterns, data will be maintained in warehouses, which will be extensible. That storage would be enabled with capabilities of identifying storage locations by their contents or part of contents. This matches operations of short-term and long-term memories in human. Low or high levels of neurons fire in different regions in cortex when they capture data, which indicate to different types of information. These groups of neurons are connected as sequences. They are referring to lots of detailed abstracted data.

- **Pattern sensing:** states the ability to discover big data patterns based on data attributes (or concerns) extraction and classification processes. In human memory, sequences of neurons in certain cortex areas (e.g., visual and vocal areas) with certain connection pattern lead to identifying characteristics of that pattern. For instance, hearing and seeing a cat lead to a certain neurons pattern in our brain based on learned concepts.

- **Formal reasoning:** a well-founded artificial intelligence functionality based on integrated statistical reasoning models to perform semantics reasoning and matching. Those models will be used to extract semantics from learned data patterns and already known semantics. Constructed chains of neurons in different cortex locations (e.g., vocal and visual cortex areas) of human brain result in formation of high level neurons that will be fired along the human life. Capturing data, which infer to known information at brain, make sequences of neurons in various related cortex areas form.

![Data semantics management methodology](image)

**Fig. 1.** Data semantics management methodology

### 3. Overview of NetMem

NetMem is a shared distributed system that can be built separately on multiple autonomous entities with capability of inter-communication and semantics integration. NetMem can be attached with already shared, existing and interconnected networking entities (e.g., servers) in the current Internet (i.e., overlay networks). NetMem will not convert the Internet core to a state-awareness network that affects Internet scalability and robustness. Rather, NetMem will run as an application that will not overbearing networking operations and will not impede connection of the various entities. NetMem outputs semantics with lower levels of details based on monitoring and learning multi-dimensional patterns of huge amount of network data, which possess higher levels of details. NetMem provides capabilities for networking entities to store/discover/retrieve at runtime and on-demand raw data and semantics related to different network concerns. NetMem targets minimizing resource consumption at entities and enhancing scalability in data and algorithms by limiting their storage at enormous entities to perform NetMem functionalities. Maintained data (i.e., raw data and semantics) at NetMem are represented uniformly, associated in big relational tables [26], and classified into three networking concerns, namely application, communication
and resource concern as those concerns are defined in [14]. At the same time, data are abstracted via FBS engineering framework [15] to functional, behavioral, and structural (FBS) aspects.

NetMem structure adopts composable and cooperative building components forming connection patterns for data feedforward and semantics feedback. NetMem will be embedded with semantics reasoning models. NetMem comprises the following operations which show the bio-inspiration with human memory:

a) Data collection and acquisition operation, which mimics functionalities of sensory memory system in human [12]. That operation involves usage of data virtualization (DV) and function-behavior-structure (FBS) engineering framework [15, 27] as a data model. This model will be used for abstracting networking data to functional-behavior-structural attributes and representing data uniformly. Locality-sensitive hashing (LSH) algorithm [18] to reduce data dimension and to address data similarities to store data efficiently;

b) Semantics reasoning operation, which mimics neocortex functionality. The reasoning operation depends on using associative rule learning (ARL) [20] algorithm and Fuzzy membership functions (FMF) [21] for attributes discovery and classification, respectively, and semantics reasoning models to extract network semantics;

c) Cloud-like data storage operations which found short-term memory (STM) and long-term memory (LTM). STM and LTM mimic the hierarchal memory system of classified and sequenced patterns in human brain [12]. STM maintains for short-time highly dynamic raw data while LTM keeps less varying data semantics or concept classes for long-time. NetMem will adopt some mechanisms such as presented in [26] to store large scale data in extensible big tables and;

d) Control and interfacing operations which represents the capability of feedforward and feedback connectivity amongst the above mentioned operations.

The following subsections describe NetMem architecture and basic building blocks, and NetMem functions showing capabilities for data virtualization and semantics reasoning.

3.1 NetMem Architecture

NetMem utilizes relational big tables as a cloud data storage model to store data in one logical place. Patterns are learned based on these maintained data which are collected from diverse networking entities (e.g. users and servers) in different networks. The design of big tables is inspired by the work in [26]. There is a capability in [26] to store large datasets in big tables based on an open source implementation technique for big tables for massive scalability defined in Hbase [28] and built on top of the Java framework hadoop [29]. We apply DV techniques [30] with statistical methods for homogenizing data to represent and store data uniformly in the big tables. Additionally, we use associative rule learning algorithms [31, 32] and statistical analysis to provide capabilities of associative access and learning data patterns in NetMem big tables. Also, fuzzy set theory and decision trees are used to extract and classify features thus aids in learning patterns of these data.

NetMem architecture comprises the following components which can inter-communicate as shown in Fig. 2:

Data virtualization and access (DVA) which is a data collection and acquisition component inspired by sensory memory system in human. DVA provides capabilities for representing and storing networking data

![Fig. 2. Structure of NetMem](image-url)
from heterogeneous data sources in different networks, DVA implements DV techniques for data homogenization; and it possesses a data model for data structure and uniform representation in NetMem tables. DVA adopts LSH [18] for reducing data dimensionality.

Fig. 3 describes the proposed DVA component; DVA comprises the following units: a) Manager which handles raw data from various sources, receives data requests via SCI component, coordinates data storage, discovery and retrieval processes to/from SM; b) Cache which is the temporary warehouse where valuable data (i.e., frequently requested data) are stored by the manager to enhance data access time; c) Matching which is attached with data homogenization algorithms (e.g., LSH) for data dimensionality reduction, finding similarities, and categorizing data into groups based on ACR concern; d) Transformer which represents uniformly data as profiles at SM tables, using the defined data models, showing data attributes, classified due to ACR concern. The transformer adopts NetMem data model which defines the format and keywords for data structure, i.e., profile with attribute-value pairs; and e) Language Stubs represent plug-in drivers that are used by the transformer to map from various representation formats to the format adopted by NetMem.

1) Short-term memory (SM) and long-term memory (LiM) which are cloud-like data storage components. SM and LiM mimic the hierarchal memory system of classified and sequenced patterns in human brain [12]. SM, or working memory, mimics lower cortex areas in human brain which deal with multi-dimensional data. SM maintains raw data which possess higher levels of details and are related to diverse Internet elements (e.g., applications, protocols, etc.). LiM mimics higher cortex areas which deal with concepts, invariant patterns. LiM, auto-associative memory, maintains data semantics, i.e., concept classes and related conceptual and behavior models, based on reasoning processes for raw data in SM. SM and LiM consist of sets of Big extendable relational data tables.

2) Semantic manager (SM) which executes semantics reasoning processes, SM mimics neocortex functionality [12]. SM is responsible for discovering/generating/matching semantics in LiM based on monitoring and learning data patterns in SM and extracted features. Various semantics reasoning models are used in SM such as latent dirichlet allocation (LDA) [17], hidden Markov models (HMM) [13], and statistical models. SM relates syntax data of high levels of details in SM to semantics data in LiM. SM constructs relational big tables of concepts to build conceptual or behavior models, i.e., relations between extracted semantics. SM uses concept ontology based on the FBS engineering framework [19] as will be discussed later. SM provides NetMem with capabilities for representing/retrieving concepts sequentially and systematically with different patterns.

As shown in Fig. 4, SM has the following components:

a) Coordinator which is responsible for handling and differentiating between enforcement signals from internal units, NCI and DVA to allow performing semantics derivation, retrieval and matching processes and to let pass networking data from SM to SM. It is the responsible for generating to NCI analytical reports based on results from semantics derivation and matching processes. Those reports might reveal the occurrence of abnormal services or attacks;

b) Patterns learning which executes algorithms and statistical analysis models for SM data attributes and features discovery (e.g., ARL [20]). This enables SM to learn patterns of data at SM;

c) Reasoning algorithms which represent semantics reasoning models such as HMM [13, 33, 34];

d) Semantics matching which is embedded with a case-based reasoning algorithm, using defined semantics in LiM, and a statistical analysis model for applying semantics fitting processes;
c) Semantics adaptation maintains algorithms for defining aspects and features of semantics considering changes happened in those semantics. FBS data based on results from semantics matching units and a control signal from coordinator;
d) Semantics definition and goals which shows experience of the SM, preserves semantic derivation models, and definitions and intentions of semantics;
e) Semantics placement which has data storage models and definition languages, such as extensible markup language (XML) document type definition, for maintaining semantics in LI/M tables;
f) Feature extraction and classification which maintains models for probabilistic and statistical analysis beside classification algorithms such as decision trees and FFMs [21] to extract and classify features of learned patterns;
g) Parse input which is responsible to handle inputs to SM whether those inputs are data (from LI/M and SM) or alerts and request signals from DVA and NCI, respectively; and
h) Data output is the gate where analytical reports and semantics are passed through to NCI and LI/M/NCI, respectively.

3) NetMem controller and interface (NCI) which represents the capability of feedforward and feedback connectivity among the above mentioned components besides allowing data/actions/alerts pass to requesting entities. NCI is responsible for handling syntax and semantics data requests from networking entities in various domains. In other words, NCI represents the gate for data exchange between NetMem components and entities in the networking world. It has the capability for differentiating the requests, and accordingly it sends tasks, i.e., data discovery, to DVA or SM. In addition, it receives responses from DVA and SM including required data that will enable requesting entities to access them. It might send SM additional tasks in case of the incapability of DVA to find required data. This is because SM has the ability to learn and derive semantic data based on concept ontology that shows different levels of details and information for networking data-based concepts. Furthermore, NCI is responsible for invoking actions/alerts based on analytical reports sent to it from DVA and SM. Analytical reports are generated based on data requests sent to DVA and SM and already data in their accessed memories where DVA has access to an internal cache memory and SM and SM has access to LI/M. NCI uses

---

Fig. 4. Semantic manager in NetMem

Fig. 5. The ontology of NetMem design
defined policies and Fuzzy rules to generate actions and alerts. For instance, NCI might generate attack alert based on a report sent to it from SM showing that current learned patterns in SM refers to an attack. NCI would have a policy or a Fuzzy rule which institutes that if an analytical report shows a prediction for attacks, hence, it will generate an attack alert.

Our ontology beyond the design of NetMem for keeping networking data with different levels of granularity is shown in Fig. 5.

3.2 NetMem Functions

3.2.1 Data Collection, Virtualization and Access
This task is executed at DVA component. Due to complexity and scale of operating computer networks and Internet, tons of high and multi-dimensional network data are generated. DVA will gather data via a sensory system (e.g., sensor networks with generic or specific type of sensors). DVA will adopt data models to represent uniformly at SM data from heterogeneous sources as profiles of attribute-value pairs normalizing range of the same type of attributes. DVA uses a data model defined by XML as a representation language clarifying type and attributes per each data profile.

DVA adopts locality sensitive hashing (LSH) algorithm [18] for reducing data dimension where it selects attributes or features using hash functions which are chosen randomly. LSH selects attributes or features using hash functions which are chosen randomly. LSH helps in annotating data and finding similarities amongst data that aids in minimizing storage space for data recollection and patterns learning. Additionally, by LSH, illustrated in Fig. 6, DVA can search for similarities especially when it faces new services that are not maintained before. Usage of LSH for data reduction would result in loss of some information which might affect NetMem operation. The degree of data loss depends on the value of hashing function length. However, DVA would utilize LSH due to its following features:

<table>
<thead>
<tr>
<th>Initialization</th>
<th>Input number of hash tables L and the width parameter K</th>
</tr>
</thead>
<tbody>
<tr>
<td>Preprocessing Phase</td>
<td>- Input a set of N data profiles P of length d which maintain group of features</td>
</tr>
<tr>
<td></td>
<td>- Given L and K, generate random hash functions g, each of length K</td>
</tr>
<tr>
<td></td>
<td>- Output L hash tables</td>
</tr>
<tr>
<td></td>
<td>for i=1 to L</td>
</tr>
<tr>
<td></td>
<td>store profiles P at bucket g(P) of hash table i</td>
</tr>
<tr>
<td></td>
<td>end</td>
</tr>
</tbody>
</table>

Query Phase

- Input a query data profile Pquery
- Generate hash value for Pquery
- Access L hash tables generated in the preprocessing phase
- Similarity test
  - for each i=1 to L
  - compare length (Pquery, Pi) and adjust if not equal
  - calculate hamming distance d = D (Pquery, Pi)
  - calculate probability of similarity = \( k \cdot (1 - \frac{1}{d+1}) \)
  - output most probable matching profiles (i.e., hamming distance is greater than threshold k) and probabilities of similarity of each one.

Fig. 6. LSH algorithm executed at NetMem

- Support high-dimensional network data by hashing and detection of approximate neighboring points;
- Its random sampling for data based on hashing (i.e., hash functions) which can enhance its complexity;
- Its similarity search and storage based on similarities (i.e., classification for features);
- Feature selection can be directed (i.e., specific features selection); and
- LSH can build indexed hash tables that speed up the operation of DVA to find out similarities amongst data.

3.2.2 Semantics Reasoning

This process is executed at SM. SM uses associative rule learning [20] to recognize attributes or features of data profiles maintained by DVA at SM. SM utilizes Fuzzy membership functions (FMF) [21] to classify extracted attributes. Different semantics reasoning models can be implemented at SM. Sections 3.4 and 5 describe three explored semantics reasoning models which are HMM-based, LDA-based and simple statistical-based reasoning models.

3.2.3 Cloud-like Data Storage

NetMem adopts large-scale data storage mechanisms as presented in [26] to maintain high-dimensional data and extracted semantics. DVA aids in reducing data dimension and extracting attributes. Those
attributes represent various network concerns as defined in [14]. The following two subsections represent two types of memory to store raw data with attributes and extracted data semantics.

### 3.2.3.1 Short-term (or working) memory (STM)
STM stores temporarily raw data of different Internet elements such as running services and applications in different networks. Communicating entities can access and learn at runtime data in STM to enhance their networking operations. SM learns patterns of data maintained in STM. Fig. 7 shows the structure of STM within our NetMem. We propose four dataset tables for STM, namely service data structured table (SDST), concerns data structured table (CDST), concerns index table (CIT), and composite concerns index table (CCIT). SDST contains entries for services and their related networking concerns. CDST extracts each concern’s data entry service in the SDST where it defines FBS attributes per each concern. The CIT and CCIT clarify in which entries the SDST, a single concern’s and group of concerns’ attributes can be found respectively.

![Fig. 7. The short-term memory in NetMem](image)

### 3.2.3.2 Long-term memory (LTM)
LTM is a permanent auto-associative memory of valuable networking data which are stored and abstracted as semantics. Extracted semantics are represented as sequences of concepts in LTM. Semantics are defined based on learning patterns of attributes at STM. Those attributes belong to three types of network concerns, i.e., application, communication, and resource concern as defined in [14], and functional, behavioral, and structural data in STM. Our concept ontology in LTM yields knowledge on different levels of abstraction where semantics are classified to application, communication, and resource concepts. Semantics are represented showing their functional, structural and behavioral aspects using FBS engineering framework as defined in [15, 27].

![Fig. 8. The long-term memory in NetMem](image)
Our L1M design, shown in Fig. 8, depends on using sets of extensible relational big tables which class and sub-class tables. Extracted semantics, or sequences of concepts, comprise three classes of features registered at class tables. Features per each class are stored at group of sub-class tables for that class. Entries, or constructed semantics, in class tables depend on categories of features found in sets of sub-class tables.

Concepts are defined as group of correlated features. Features are represented via numeric vectors and alphabetic symbols. Class and sub-class tables and their contents build relational graph (or conceptual pattern) for all registered concepts. A conceptual pattern is a composition of concepts; and shows a description of inter-relationship between those concepts where their relationship might be containment or imply or dependence.

4. HMM-based Reasoning Model

HMM [13] are widely used in learning processes in different fields as in image and speech recognition and robotics for gesture imitation. HMM are categorical sequences labeling supervised and unsupervised learning algorithms. Sequence labeling can be treated as a set of independent classification tasks. HMM depend on a mathematical model with parameters (i.e., initial state probability ($\pi$), state transition probability (A), and observation (B) probability) that can be adjusted for supporting different semantic topics in many contexts. With sets of training data, Baum-Welch’s forward-backward algorithms can be applied to HMM to discover unknown HMM parameters. HMM operation suits the operation of NetMem where HMM can deal with multi-dimensional big data [16]. Each input state in HMM can be specific to output semantic topics. Considering input states as Markovian processes might affect degree of accuracy for output semantics. This can be mitigated, to some extent, by adjusting parameters of HMM. For example, the forward and backward transition probabilities among specific states can have the same value. This will give equal weights for getting certain semantics if transition occurs among those states.

One of HMM problems is the floating-point underflow problem. This can affect the extraction process of correct semantics. It can be overcome by taking logarithm for values of probabilities and performing summation process instead of multiplication to calculate forward probabilities. HMM have limitations due to the assumption about data using the Markovian assumption and the usage of maximum likelihood estimator [35]. This limits HMM capability in discovering efficiently high-level features with long-range semantics dependencies. Another limitation for HMM can be found if it is required to design an HMM-based reasoning model with large input states. This means a lot of HMM parameters to be calculated. This might affect performance (e.g., timeliness) of a semantics reasoning model implemented with HMM.

HMM-based reasoning models can be combined to form a hybrid or multi-stage reasoning model. This can aid in enhancing HMM performance (e.g., reasoning accuracy). In case of designing large sets of multi-stage HMM-based models for network semantics reasoning, HMM might require large sets of training data. This can be offered easily through network data due to their characteristics (e.g., massive volume and complexity) and availability via diverse sources (e.g., offline datasets and Internet traffic). Since HMM-based reasoning models can be considered as static models or prototypes for semantic reasoning, HMM-based models face challenges at operation with reduced-dimensional data. This will affect accuracy of operations for obtaining right semantics.

Based on the above discussion on HMM, we implemented and tested HMM within NetMem SM [9]. SM runs HMM as models for semantics reasoning and extraction. Based on extracted features and maintained data profiles by DVA in SM, SM runs HMM-based models, described in Fig. 9, after executing the training phase using a sample of data profiles stored by DVA in SM. The input to HMM-based models is sequence of profiles’ attributes and the output is semantics (i.e., sequences of concept classes). Extracted and classified profiles’ attributes by ARL and FMF are fed as input states to HMM-based models. Based on sequence of input attributes, calculated HMM parameters and maximum likelihood estimation, HMM can extract semantics related to specific Internet elements (e.g., behavior of communication protocols in wireless networking contexts).
For example, NetMem adopts HMM-based reasoning model to learn behavior semantics (or concept classes) of storage memory in TCP hosts based on collected raw data from hosts [36]. HMM has five input states for estimating the behavior of those memories. Those states are “large memory size”, “long memory system uptime”, “abnormal allocation failures”, “normal running system processes”, “small stored session information size”. The most likelihood estimated output concept classes based on any sequence of the previously mentioned attributes are “host resources behavior” and “abnormal storage memory behavior”.

To have that feature (i.e., independence of input sequence forming in HMM), the state transition and observation probabilities should be designed carefully. For instance, group of input states might have equal transition probabilities to transfer from one of them to others. Also, the observation probability of a related semantic topic will be with a high value with respect to any of those attributes.

HMM parameters are trained and assigned using the unsupervised Baum–Welch learning algorithm [36] based on the training data sets. That algorithm depends on initially developed HMM for finding maximum likelihood HMM parameters through iteratively training the parameters of initial HMM and observing the output semantics sequence. The ability of feeding to HMM sequences of data attributes related to diverse network concerns enables getting output sequence with associated semantics or concept classes regarding those concerns. According to the above example of TCP memories behavior, an input sequence with equal initial state probability (i.e., \( \pi = 1/5 \)) to HMM might begin with one of the above aforementioned five classified attributes. We assume the designed HMM with equal state transition probabilities (i.e., \( A_k = 1/4 \) for \( i \neq j \) and \( A_k = 0 \) for \( i = j \)). To get the output concept classes, the observation probability \( B \) matrix, which relates each input state with an output, might equal, for instance, \((0.5, 0.1, 0.4), (0.5, 0.45), (0.1, 0.8, 0.1), (0.7, 0.3), (0.6, 0.35, 0.05)\). \( B \) matrix consists of \( r \) rows and \( c \) columns where \( r \) equals the number of input attributes and \( c \) equals the number of output concept classes. The three available output concept classes representing the three \( B \) columns are “host resources behavior”, “abnormal storage memory behavior” and “large storage resources”. Here, we remark according to the previous example that input classified attributes have different observation probabilities with the possible output concept classes.

5. LDA-based Reasoning Model

LDA-based reasoning models provide semantic topics models that enable discovery of hidden topic-based patterns through a supervised learning process. LDA algorithms [17] give a systematic and well-defined way for inferring latent semantic topics in large scale of multi-dimensional data sets. So, LDA operation can suit characteristics of network data. LDA samples weights for associations between semantic topics and data attributes based on adopting algorithms for sampling (e.g., Gibbs sampling [37]), multivariate distribution (e.g., dirichlet distribution) and training data sets. LDA-based reasoning models have the capability of extracting latent features and semantics based on prior probabilities for data attribute-profile and attribute-semantics associations. LDA assumes that attributes of data are related to randomly chosen semantic topics. The selection of semantic topics is based on random values of multivariate probability distribution parameter. LDA-based reasoning models have the ability to estimate semantics based on small scope of extracted features with long-range semantic dependencies. In LDA, selection parameters and prior association probabilities can be directed to relate to specific attributes to group of semantic topics (i.e.,

![Fig. 9. A probabilistic model using HMM for semantics extraction](image-url)
having specific distribution of semantic topics over group of attributes). This enables LDA-based reasoning models to work with reduced-dimensional data when applying dimensionality reduction algorithms such as LSH.

One of the main advantages for LDA-based reasoning models that these models are extendible [38] to support more associations amongst semantic topics and data attributes. LDA-based models can be used to enhance functionalities of other semantic reasoning models (e.g., HMM) to strengthen their capability of extracting latent features and semantics. Some limitations of LDA-based models can be found due to their randomization process for assigning parameters’ values of attribute-semantics associations. However, this can be mitigated according to the assigned initial prior association probabilities which can provide high association weights among groups of attributes and semantics. LDA-based models, at large number of training data sets, can face the overfitting problem.

According to the above discussion and characteristics of LDA, pseudo code is depicted in Fig. 10, we implemented for NetMem SM LDA-based models as generative probabilistic dynamic semantics reasoning models [11]. LDA-based reasoning models will aid in possessing NetMem the capability of efficient data recollection based on reduced-dimensional stored data in SM. SM assigns analyzed data profiles in SM to semantic topics based on prior assigned association weights for extracted and classified attributes (by ARL and FMF) with semantics and also association weights of data profiles with semantics. Semantic topics will be stored as sequences of concept classes by SM in LTM.

![Fig. 10. LDA algorithm executed at NetMem](image)

The operation of LDA to discover semantic topics in M analyzed profiles in SM is executed every defined reasoning window or through other criteria as triggering signal sent from DVA to SM. LDA samples a hidden semantic topic z for each m data profile through calculating sampled posterior probability vector $\theta$ of topic-data profile association which depends on prior association weight $\alpha$, number of the $m^{th}$ profile’s attributes related to a certain topic z and $N$ total number of attributes in the m profile. Also, LDA calculates sampled posterior probability $\phi$ of attribute-topic association based on prior attribute-topic association weight $\beta$, number of attribute instances assigned to topic z and total number of attributes in all M profiles assigned to topic z.

Through a certain number of iterations and using Gibbs sampling, LDA draws hidden topics for each m data profile and then draws semantic topics for comprised attributes in each analyzed data profile. For example, three semantic topics are defined in LDA: (“host resources behavior”, “abnormal storage memory behavior” and “huge storage resources”). Twenty data profiles (i.e., $M=20$) in SM has the same three extracted and classified attributes which are “large memory size”, “long memory system uptime”, “abnormal allocation failures”, “normal running system processes”, “small stored session information size”. Each attribute and profile has a prior topic association weight vector. Based on the overall prior weight vectors ($\alpha$ and $\beta$) and number of semantic topics, a sampled topic association probability vector $p_{assoc}$ of length equals the number of available semantic topics is calculated like $p_{assoc} = p(semantic\ topic_1) = 0.6, p(semantic\ topic_2) = 0.3, p(semantic\ topic_3) = 0.1$. In each LDA iteration, the current assigned topics for a data profile and comprised attributes are removed. Then, a random number $a$ is sampled based on $p_{assoc}$ and the summation of its contents. The higher $p$ topic association value will be chosen and the related topic is assigned. For example, if $a$ equals 0.4, number of attributes and related profiles assigned to the first semantic topic (i.e.,
the new topic (i.e., 0.6) increases. This is because the probability of the first semantic topic (i.e., 0.6) is larger than 0.4. Therefore, updates will be happened to posterior association weights \( \theta \) and \( \varphi \) according to changes in number of attributes and profiles that relate to first semantic topic. Hence, the posterior association weight of the first topic with data profiles and comprised topic-related attributes increases. In the example (stated in the HMM section) of semantic reasoning for the behavior of storage memory in TCP hosts, LDA assigns the classified attributes in analyzed data profiles to the following feature topics: “host resources behavior”, “abnormal storage memory behavior” and “huge storage resources”. For instance, LDA assigns a data profile to a specific semantic topic “abnormal storage memory behavior” based on assigned weights of that profile’s attributes regarding the assigned topic. More than one semantic topic can be assigned to data profiles based on the executed topic modeling process (which depends on parameters of the used multinomial distribution) and profiles’ attributes.

Equation (1) is used to calculate LDA perplexity where that equation was derived based on the one mentioned at [17]. Calculating perplexity helps in evaluating performance of the LDA-based models in detecting and categorizing features in data profiles based on learned parameters (e.g., prior profile-topic association weight or probabilities). So, based on \( M \) data profiles in SM and their \( N \) related features and number of defined semantic topics \( K \), we can get the value of perplexity.

\[
\text{Perplexity} = \exp^{-(\sum_{m=1}^{M} \log(p(feature_m)/\sum_{n=1}^{N} p(feature_n/\text{topic}_n)p(topic_n/\text{profile}_m))} \tag{1}
\]

Where \( p(feature_m) = \prod_{n=1}^{N} p(feature_n/\text{topic}_n)p(topic_n/\text{profile}_m) \) \( N_m \) is the number of features per each data profile \( m \).

6. Simple Statistical-Analysis-Based Reasoning Model

Characteristics of big data impede regular data monitoring and analysis tools to anticipate data contents and structure and to interpret patterns meaningfully. Construction of statistical models [39] can help in understanding patterns of big data. This would lead to a capability of extracting features and semantics. One of the problems with those models is that they are specific to certain semantic topics. Also, these models have limitations in extracting latent features. Inefficient design (e.g., inadequate algorithmic model) for those models can lead to incorrect extracted information. There is a need for a data training phase to test accuracy of models. Some rules (e.g., Fuzzy rules) can be constructed and used within statistical models to fit certain semantic topics. Adoption of classification techniques with rules can help statistical models to extract high level data features.

We provide a simple statistical-analysis-based reasoning model, illustrated in Fig. 11, for NetMem SM [10] to extract semantics related to specific semantic topics (e.g., behavior of TCP protocol or TCP hosts’ storage memory), SM will learn patterns of \( N \) different data profiles represented by DFA in SM to derive semantics and store them in ELM. There are \( K \) targeted attributes that can be extracted and classified from profiles in SM. Using data pattern learning algorithms (e.g., ARL [20]) and a classification technique (e.g., FMF [21]), SM will learn group of attributes \( \mathcal{A}_p \) per each data profile in SM that matches required \( \mathcal{K} \) attributes. An assumption is made that attributes per profiles are independent with equal probabilities of

![Fig 11. Simple statistical-based model for semantics reasoning](image)
existence at each analyzed profile (i.e., attributes have same weights per profile). SM searches every reasoning time period $T_b$ for similar data profiles $N_p$, of each $n$ profile of total $N$ profiles in SM. Equations (3) and (4) describe the initial attribute weight and data profile weight, respectively.

$I_{A_i,P_n}$ is defined as the initial attribute $i$ weight per data profile $P_n$ where:

$$I_{A_i,P_n} = 1/K$$

and $1 \leq n \leq N$, $N$ number of profiles in SM

$$W_{P_n} = \sum_i (I_{A_i,P_n} \times M_{A_i,P_n})$$

for each attribute $i$ per data profile, where $1 \leq i \leq K$

Where $M_{A_i,P_n}$ is the membership value of the attribute $i$ in a data profile $P_n$ calculated by defined FMEs. The previous simple equations calculate group of features that can be used to extract semantics. SM has definitions for sets of fuzzy rules to aid in extracting semantics. Those rules can be used, for example, in determining normal behavior of the storage memory in TCP hosts. With Fuzzy rules, SM adopts a vector $T$ of thresholds, which are determined by experts or via SM experience and history. Based on results from profiles analysis process and thresholds’ values, SM can abstract semantics. Here is an example of a rule that is applied for $N$ analyzed data profiles in SM through $T_b$ unit time:

IF ($W_{P_n} > T_a$) && $(N_p \leq T_b)$ && $((N_p/N) < T_b)$ && $(A_i = K)$

THEN normal behavior ELSE abnormal behavior

$T_a, T_b,$ and $T_c$ are thresholds defined in $T$ vector for profile weight, similar profile number, and similar to total profile number ratio, respectively. According to the above rule, SM will extract semantics for TCP hosts’ storage memory as follows:

- IF normal behavior THEN develop semantics; largeNumberOfProfiles, CompleteDataProfile, NormalProfileWeight.
- IF abnormal behavior THEN develop semantics; SmallNumberOfProfiles, InCompleteDataProfile, AbnormalProfileWeight.

largeNumberOfProfiles means that $N_p$ exceeds the threshold $T_b$. CompleteDataProfile means that the data profile maintains all interesting $K$ attributes such as “large memory size”, “long memory system uptime” and “abnormal memory allocation failures”. NormalProfileWeight means that $W_{P_n}$ is above threshold $T_a$. The semantics for the abnormal behavior will reveal that profiles do not satisfy the above conditions.

7. HIT-based Reasoning Model

7.1 HIT Overview

We propose and implement a HIT for NetMem to build efficient reasoning model to reason about network semantics based on learning patterns of full or reduced-dimensional data. The HIT integrates LDA and HMM algorithms as illustrated in Figs. 12 and 13 (HIT-based reasoning operations will be explained in the next subsection). Our HIT is designed to overcome limitations of the semantics reasoning operation with only adopting HMM, which was introduced in our preliminary NetMem design [40]. The hybridization of HMM and LDA enables latent features extraction with semantics dependencies not just based on learning features with syntax dependencies. On the one hand, LDA has the capability to discover high-level features of data profiles with full or reduced dimensionality. LDA possesses extendible

Fig. 12. The LDA-HMM-based reasoning model
Algorithm: Integrated LDA-HMM model for semantic reasoning

1. Repeat every $\alpha$,
2. Attribute generation: $a \sim \text{Dir}(\alpha)$
3. Data generation: $P = \text{captureData}(\text{Concept}(\text{Class}))(\text{Profile})$
4. for $m = 1$ to $M$
5. sample topic $z$ from $Z$ for the $m$th data profile based on Gibbs sampling prior probabilities $a^m$
6. end for
7. calculate $q = \text{Dir}(\beta)$ for attributes in each profile
8. for $n = 1$ to $N$
9. $z \sim \text{multinomial}(\theta^m)$
10. end for
11. for $m = 1$ to $M$
12. $z \sim \text{multinomial}(\theta^m)$
13. end for
14. for $m = 1$ to $M$
15. calculate observation probability $B_{m,n}(\text{semantic topic})$ for each input state of each implemented HMM
16. end for
17. get maximum likelihood semantic topic sequence of length $T$ with high probability for each implemented HMM
18. end for
19. end for
20. until operation time

Output: set of semantic topics that are related to various concepts and represented as an associated concept scheme.

Fig. 13. Pseudo code of the HIT-based reasoning model.

operation where it can support more associations amongst semantic topics and extracted high-level data features. LDA has a well-defined inference methodology to associate a data profile with groups of attributes with several semantic topics. However, adopting LDA alone in semantics reasoning might produce some shortcomings due to LDA’s limitations such as the bag of words assumption [41] that might result in semantic topics misclassification where there is a possibility of semantic topic sampling process to allow attributes, related to same semantic topics, to be assigned to other semantic topics. Furthermore, LDA’s probabilistic inference process for topics is computationally complex where that process has NP-hard complexity in case of inferring document’s topics using a large number of semantic topics. On the other hand, HMM can efficiently extract data semantics from variable-length input sequence of data features, extracted by LDA, using unsupervised learning algorithms. HMM’s statistical foundations are computationally efficient and well suited to handle new data. A single HMM can be built by combining a variety of knowledge sources with the consideration of their properties. This enables an efficient design of an HMM to reason about semantics related to various Internet elements. However, utilizing HMM singularly for semantic reasoning might result in some deficiencies due to HMM incapability to: a) discover high-level features with long-term semantics dependencies due to the assumption about data using the Markovian process and the usage of maximum likelihood estimator; and b) work efficiently with reduced-dimensional data since HMM needs large amounts of data for training and adjusting HMM’s unstructured parameters.

7.2 Operation of HIT-based Reasoner

In our HIT-based reasoning model, LDA [17] is able to extract latent features with long-range semantics dependencies based on adopting inference models. Those models define correlations of semantic topics among data attributes and related data profiles. Relying upon determined inference models, LDA can extract hidden semantic topics related to each data profile’s attribute and also assign an overall topic to every data profile. Based on extracted semantic topics or features by LDA, HMM [13] output depends on the sequence of input data features related to diverse network concerns. HMM can be designed and trained to get output based on sequences of data features, independent of features’ order. Table 1 shows the parameters used within the HIT.

The semantic reasoning process in each NMemAgent is executed every defined reasoning window size or through other criteria as triggering signal sent from DVA to SM. The HIT operation depends on learning patterns from profiles and comprised attributes in storage memory. Groups of attributes are discovered using the applied ARL algorithm and a defined set of FMF in SM. For instance and through a defined reasoning window, SM aggregates information, initially, via learning patterns of TCP data profiles in the storage memory through recognizing and classifying attributes of each profile. Analyzing TCP data profiles in storage memory by ARL and FMF might give for LDA the following classified attributes: 10000 profile’s instances, large_TCP_packet_size, TCP-SYN_packet_type, file_transfer_service_type.
Table 1
LDA-HMM-based model parameters

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>K</td>
<td>Number of hidden feature or semantic topics</td>
</tr>
<tr>
<td>Z</td>
<td>Identity of hidden semantic topics</td>
</tr>
<tr>
<td>V</td>
<td>Number of attributes in all data profiles (or profiles)</td>
</tr>
<tr>
<td>N</td>
<td>Number of attributes per each data profile, ( N \leq V )</td>
</tr>
<tr>
<td>M</td>
<td>Number of data profiles</td>
</tr>
<tr>
<td>F</td>
<td>Identity of feature topics of all attributes</td>
</tr>
<tr>
<td>( \alpha )</td>
<td>Prior featureTopics/profile weight for feature topic-profile association</td>
</tr>
<tr>
<td>( \beta )</td>
<td>Prior attribute/topic weight for attribute-feature topic association</td>
</tr>
<tr>
<td>( \theta )</td>
<td>Sampled posterior feature topics/profile weight vector of length Z for feature topic-profile association</td>
</tr>
<tr>
<td>( \phi )</td>
<td>Sampled posterior attribute/profile weight vector of length V for profile attribute-feature topic association</td>
</tr>
<tr>
<td>( \pi )</td>
<td>Initial state probability vector of length N</td>
</tr>
<tr>
<td>A</td>
<td>State transition probability matrix of size N ( \times ) N</td>
</tr>
<tr>
<td>B</td>
<td>Observation (or output semantics) probability matrix of size N ( \times ) N</td>
</tr>
<tr>
<td>( ST_i )</td>
<td>Input HMM state i (or feature topic), ( i \leq N )</td>
</tr>
<tr>
<td>( O_i )</td>
<td>Output semantics based on input state i</td>
</tr>
</tbody>
</table>

**LDA operation in HIT**

LDA [17] gives a systematic and well-defined way for inferring latent semantic topics in large scale of multi-dimensional data sets. LDA assigns random values for prior probabilities of semantic topics and related attributes based on data sampling (e.g., Gibbs sampling), multivariate distribution (e.g., directory distribution) and training data sets. LDA has the capability of extracting latent features and semantics based on prior probabilities, using directory distribution, for data-attribute data profile and data-attribute semantic topic associations. LDA assumes that attributes of data are related to randomly chosen semantic topics. The selection of semantic topics is based on random values of multivariate probability distribution parameters. LDA executes semantic topic sampling for each data attribute in every data profile for all profiles, and the updating processes for profile-semantic topic and attribute-topic associations, which are iterated many times. LDA looks at each data attribute and generates its related latent feature within each data profile (i.e., LDA makes semantic topic modeling for each data profile based on its comprised attributes). LDA randomly assigns a semantic topic for each attribute based on initially defined weights of topic-attribute associations. Accordingly, the weight of the assigned topic with respect to related attributes is increased. For certain number of iterations, the process repeats and LDA will provide posterior weights of topic-attributes association and accordingly profile-topic association.

For example, three feature or semantic topics (i.e., \( K = 3 \)) are defined in LDA: (“normal TCP packet”, “normal comm-flow”, “TCP comm-protocol”). Ten data profiles (\( M = 10 \)) in the storage memory have the same three attributes (i.e., \( N = 10 \)). Each attribute and profile has a prior topic association weight vector. Based on the overall prior weight vectors (\( \alpha \) and \( \beta \)) and number of semantic topics, a sampled topic association probability vector \( p_{\text{topic}} \) of length equals the number of available semantic topics is calculated like: \( p_{\text{topic}} \sim p(\text{semantic topic } _1) = 0.75 \), \( p(s_2) = 0.2 \), \( p(s_3) = 0.05 \). In each LDA iteration, the current assigned topics for a data profile and comprised attributes are removed. Then, a random number \( u \) is sampled based on \( p_{\text{topic}} \), and the summation of its contents. The higher \( p \) topic association value will be chosen and the related topic is assigned. For example, if \( u \) equals 0.6, number of attributes and related profiles assigned to the first semantic topic (i.e., the new topic) increases since \( p(s_1) \) which equals 0.75 is greater than 0.6. Thereafter, updates will be happened to posterior association weights \( \theta \) and \( \phi \) according to changes in number of attributes and profiles that relate to first semantic topic. Hence, the posterior association weight of the first topic with data profiles and comprised topic-related attributes increases.
HMM operation in HIT

Extracted and classified features, output from LDA, form a sequence and convey to parameters of HMM to generate semantics. The estimation process for HMM observations relies on continuous input sequence with different Gaussian distributions. Then, HMM performs distribution mixture for obtaining the most-likely output sequence. HMM looks at the group and the sequence of data attributes or features. The order of states in an input sequence might change the output observations. In other words, the existence of the same data features, however, with different order might result in different outputs adopting the same HMM. However, the HMM can be designed to have outputs based on having specific group of input states (or features) without considering their sequence order.

For example, an input sequence to HMM might be (“normal TCP packet size”, “normal comm-flow”, “TCP comm-protocol”) with equal initial state probability \( \pi \) (i.e., \( \pi = 1/3 \)) and state transition probabilities \( A \) (i.e., \( A_{ij} = 1/2 \) for \( i \neq j \) and \( A_{ij} = 0 \) for \( i = j \)) where \( A_{ij} \) is the transition probability form state \( i \) to state \( j \). The first feature can be classified as an application concern and the other two features as communication concerns. Accordingly, the expected HMM output observation based on the previous sequence with any feature order might be “normal TCP-based service”. To get the previous output, the observation probability B matrix, which relates each input state with an output, regarding that concept class (i.e., output) will be high. For instance, B matrix might consist of three rows \( r \) and three columns \( c \); and it might equal

\[
((0.3, 0, 5, 0, 2), (0.2, 0.8, 0, 0), (0.4, 0.45, 0.05))
\]

where the number of \( r \) equals the number of input features and the number of \( c \) equals the number of output concept classes. According to the previous example, all input features have high observation probability with the “normal TCP-based service” concept.

8. HIT-based versus Monolithic Reasoners

We highlight in this section the differences between NetMem semantics reasoning process using monolithic intelligence techniques, adopting HMM or LDA algorithms, and HIT (i.e., the hybrid LDA-HMM algorithm). As an example, NetMem adopts different reasoning algorithms to reason about semantics of TCP-based file transfer service. Through a defined reasoning window, SM learns patterns of reduced-dimensional data profiles formed by DVA processes and maintained in the storage memory using the implemented ARL algorithm and FMF for recognizing and classifying attributes, respectively. SM recognizes the frequency of each data profile in the storage memory and its comprised attributes. According to the discussed example in the section of data handling in NetMem, analyzing the reduced dimensional data profile \( P_{LSH} \) by the adopted ARL and defined FMF gives the following: 10000 profile’s instances with the attributes vector \( Attr: \{ a^1 = \text{large_TCP-SYN\_packet\_size}, a^2 = \text{TCP-SYN\_packets}, a^3 = \text{file\_transfer\_service}, a^4 = \text{abnormal\_port\_number} \} \).

We show in the next paragraphs the operation technique for three implemented algorithms (HMM, LDA and HIT) for semantics reasoning based on processes of data representation and dimensionality reduction which are discussed in the last paragraph. Due to adopting different algorithms for semantics reasoning, various output semantics will be formed and kept.

Firstly, HMM-based reasoning model is used to extract semantics regarding the behavior of TCP-based file transfer services. The input states to HMM are described as sequences. Each input state represents one learned and classified data-attribute. For example, “large TCP-SYN packet size” is a learned and classified attribute based on captured values in the TCP packet size and packet type fields in a data profile. For HMM operation, we assume that we have four possible input states \( ST_1, ST_2, ST_3 \) and \( ST_4 \) to HMM for estimating the behavior of those services. Those input states represent the extracted and classified data attributes per each data profile. Those states are data attributes defined in the \( Attr \) vector. The estimated observation (or concept class) based on any sequence of the previously mentioned attributes is “O: Abnormal TCP-based service”. HMM extracts that concept class based on its defined operation parameters. For instance, the observation probabilities \( B \), using the maximum likelihood estimator, for the concept class \( O \) and the other concepts are high (e.g., 0.9) based on having any input attribute (i.e., state \( ST_i \), and \( i \leq 4 \)) of the defined \( Attr \) attributes vector \( \{ a^1, a^2, a^3, a^4 \} \). Also, the transition probability \( A \) between any two input features (\( ST_i \) and \( ST_j \), and \( i \neq j \)) has the same value. This increases the probability of having the targeted \( O \) based on using the trained HMM with the forward algorithm over sequences of interesting attributes.

Secondly, LDA-based reasoning model is used to reason about semantics for TCP-based file transfer services, LDA after 1000 iterations assigns attributes in \( P_{LSH} \) to the following feature topics: “Abnormal
TCP-SYN packet size”, “TCP-based File transfer service”, “Abnormal TCP control packet”, “Abnormal TCP-based file transfer service”. LDA samples the feature topics as the following example. The classified attribute \( a \), large_TCP-SYN_packet_size, has high prior association weight \( \beta \) (e.g. \( \beta > 0.75 \)) with the semantic topic, Abnormal_TCP-SYN_packet_size. So, the updated attribute-topic posterior association weight \( \phi \) for \( a \) after 1000 iterations will be high with respect to the assigned feature topic. In addition, the prior association weight \( (a) \) of P_LSH with the feature topic “Abnormal_TCP-based_file transfer service” is high (e.g., \( \alpha > 0.8 \)). Then, LDA samples P_LSH that specific topic based on the updated topic-profile posterior association weight \( \theta \) taking into consideration the number of profile's attributes assigned to that topic (e.g., two attributes) and the total number of profiles' attributes (e.g., P_LSH has four attributes).

Finally, the hybrid LDA-HMM algorithm is designed to produce more meaningful information from analyzed data profiles and to form richer associations amongst extracted semantics. The hybrid algorithm extracts latent features of each data attribute in analyzed data profiles relying on learned and classified attributes in analyzed data profiles in the storage memory. Then, the hybrid algorithm outputs data profiles for each data profile based on a sequence of extracted latent features of the whole data profile. The LDA-HMM algorithm looks at both (a) data attributes and related latent features within data profiles and b) the sequence and set of extracted latent features within each data profile. The implemented LDA algorithm performs semantic of feature topics modeling for analyzed data profiles and their comprised attributes. The output of that modeling process is a set of syntax states per each profile that is fed to HMM to reason about semantics concerning each profile. Utilizing simple feature classification techniques, e.g., using FMF, enables assignment of membership degrees for some extracted feature topics which have related designed FMF. These degrees enable SM to build and update dynamic network-concept ontology. In the example of reasoning about TCP-based file transfer service semantics and using the same operation parameters of LDA in the previous semantics reasoning case, the latent feature topics \( T \) extracted by the LDA algorithm from P_LSH and its comprised attributes might be \( T = \{ t = \text{Abnormal TCP-SYN packet size}, t = \text{TCP-based File transfer service}, t = \text{Abnormal TCP control packet} \} \). An input sequence comprised the previous latent feature topics (i.e., any sequence order of \( (t', t', t', t') \)) to a designed HMM might yield the following semantics or observations: \( O = \{ O_1=\text{File Transfer Service Behavior}, O_2=\text{TCP-based Service Operation}, O_3=\text{TCP SYN Flood Attack} \} \). Accordingly, a simple dynamic network-concept ontology can be built via the obtained observations or concept classes. For instance, the top parent class will be \( O_1 \) and it will have three sub classes \( O_2, O_3 \) and \( O_4 \). The concept class \( O_2 \) will be a child class for the parent class \( O_3 \) and so on. In addition, the four classified attributes in the Atr vector will be registered as FBS aspects for the lower child concept class \( O_4 \).

9. Evaluation

We evaluated NetMem effectiveness with real network data and Internet traffic to learn behavior and extract semantics of attacks and abnormal data profiles. This real traffic was collected via snort [19], open-source software for intrusion detection systems. Snort represented in our case study the sensory system for data acquisition. We compared NetMem effectiveness with snort in detecting attacks like TCP-SYN flood attack, UDP-flood attack, ICMP ping flood, and abnormal data flow due to irregular port numbers. NetMem models for data representation depended on data profiles constructed by snort.

A main goal in this paper is to show the capability of NetMem for understanding at runtime the networking context via learning and deriving semantics related to normal/abnormal data flows and attacks. A test bed network of five entities was installed. There were two hosts and two servers. FTP and Web servers were implemented over two static laptop machines running Windows 7. Two hosts were built over other two static laptop machines to handle data from servers. Furthermore, hosts were connected to the Internet. NetMem was implemented over a Windows 7 laptop, an entity with routing functionalities that can capture data/control packets go from/to hosts to/from servers and Internet. NetMem was implemented as an application, whose code was written in Java and clarified operations of NetMem comprised entities: NCI, DVA, SM, LfM, and SM. NetMem operation was tested without and with LSH for reducing data dimensionality at different values of hashing function length. In addition, NetMem utilized ARL algorithm and FMFs for discovery and classification of attributes and features from data. NetMem performance was evaluated with 3 different semantics reasoning models, namely HMM, LDA, and statistical models, respectively.
In our scenario, hosts run file transfer services on top of TCP protocol to get files and access pages from FTP and Web servers, respectively. Also, hosts utilized UDP protocol to transfer data packets through Internet. One of the two hosts was malicious and it generated ICMP ping flood to the other legitimate host. In addition, the malicious user sent the web server successive of TCP SYN requests to form TCP-SYN flood attack. Also, the flow generated from web server was abnormal due to the usage of irregular TCP port numbers. Moreover, the non-privileged host connected to Internet tried at time periods to overwhelm network bandwidth by generating UDP Flood attack through sending many worked/unworked web servers.

The other host is a legitimate user which required to get file from FTP server. However, it faced challenges to access its service according to the behavior of the malicious host. Tables 2 and 3 show configured parameters of the test bed scenario and performance evaluation metrics, respectively.

### Table 2
Testbed configuration parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of attributes per data profile</td>
<td>23</td>
</tr>
<tr>
<td>Rate of Using Sensory Functions in Hosts for</td>
<td>Every 11 seconds</td>
</tr>
<tr>
<td>Recognition Attack Alerts (frequently)</td>
<td></td>
</tr>
<tr>
<td>Rate of NetMem Access and Data Patterns</td>
<td></td>
</tr>
<tr>
<td>Detection in StM by SM (frequently)</td>
<td>Every 50 seconds</td>
</tr>
<tr>
<td>Rate of Change for Contents (i.e. Data Patterns) in StM</td>
<td>Every time Hosts</td>
</tr>
<tr>
<td>LSH (L hash tables, K hash function length)</td>
<td>(L=8, K=3,6,9)</td>
</tr>
<tr>
<td>(variable)</td>
<td></td>
</tr>
<tr>
<td>LDA (# of iterations, topics, prior probabilities for topic-profile &amp; topic-concerns)</td>
<td>20000, 4 topics, variable</td>
</tr>
<tr>
<td>Operation Time</td>
<td>100 seconds</td>
</tr>
</tbody>
</table>

### Table 3
Semantics reasoning process analysis metrics and equations

<table>
<thead>
<tr>
<th>Metric</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Prediction Accuracy</td>
<td>Where $T_p$, $F_p$, $T_n$ and $F_n$ are true and false positive and negative, respectively</td>
</tr>
<tr>
<td>False Positive (Fp) Rate</td>
<td>$F_p(T_p+T_n)$</td>
</tr>
<tr>
<td>False Negative (Fn) Rate</td>
<td>$F_n(F_n+T_p)$</td>
</tr>
<tr>
<td>Recall (R) Rate</td>
<td>$R = \frac{T_p}{F_p+T_p}$</td>
</tr>
</tbody>
</table>

NetMem DVA adopted a data model to represent raw data (collected by snort) of running TCP/UDP services as profiles of attribute-value pairs (i.e., DVA homogenizes representation of services data as profiles of attributes and values). For example, a TCP data profile contained attributes such as packet type, port number, and service type. DVA applied an LSH algorithm, written in Java, to reduce snort data profiles of dimension $d$ ($d=23$). LSH algorithm generated random hash function of length $K$. DVA used that algorithm to find similarities among those profiles for efficient storage in StM. SM analyzed data profiles in StM and it extracted and classified features, e.g., identifying the used transport protocol and calculating number of profiles repetitions. SM studied profiles adopting our defined ARL algorithm and FMFs for discovery of and classifying attributes per profiles. SM run our designed HMM, LDA, statistical, HIT models to extract features and define a semantic-topic for each data profile. If features of a profile result in estimation of attack/abnormal profile for a defined number of successive times, then SM would register semantics for attack/abnormal flow at LtM. For example, SM looks for specific features in profiles as number of occurrence in StM, type of packets as TCP-SYN packets, type of service such as TCP of UDP and packet size. Based on those features and calculated and posterior probabilities for topic-profile association, SM decides normal or abnormal profiles and predicts attacks.

In our experiments, we proposed different cases of networking operations. We tested performance (e.g., accuracy, recall, false negative (Fn) and false positive (Fp) rates) of operations without using NetMem (i.e., with snort) and with NetMem based on data captured and modeled by snort. Snort used data capturing tool, called winpcap [42], to represent data as profiles which comprised attributes such as source/destination IPs and ports. We enabled snort rules (e.g., FTP rules) concerning interesting attacks and also designed rules that can detect abnormality such as traffic with irregular TCP port numbers. Snort performance measurement depended on fired rules and its generated statistical files. We evaluated NetMem performance without LSH.
and with LSH at various $K$ values (3, 4 and 5). Note that $K$ value refers to length of or attributes number in abstracted profiles.

In Fig. 14, we show the impact of adopting LSH algorithm in minimizing space required for storing data profiles in StM compared with different operation cases; NetMem without LSH and with LSH at different values of $K$. In operation without LSH, NetMem data models are used to represent all attributes constructed from snort data. So, each data profile will be of full-dimension and StM size in case of no LSH will be larger than case of using LSH. In the case with LSH, StM maintains dimensionality-reduced profiles. But, increasing the $K$ value means more represented attributes (i.e., decreasing in the loss of represented information) by profiles stored in StM and high probability to have similar profiles in StM. So, the storage space saving ratio as illustrated in Fig. 15 improves at larger $K$ values where the probability to have profiles with same group of attributes increases.

Fig. 16 shows the processing time overhead of NetMem system and snort for detecting attacks and abnormal profiles. NetMem overhead is examined at using different semantics reasoning algorithms.
Longer processing overhead is faced in case of using NetMem without adopting LSH. On the other hand, reducing data dimensionality with LSH enhances NetMem operation where features discovery and semantics extraction processes were performed in a timely manner. Although, the usage of LSH results in losing some information, however, NetMem succeeds in learning normal and abnormal traffic behavior with high accuracy.

Figs. 17 and 18 illustrate detection accuracy, recall, Fp and Fn rates for NetMem with and without using LSH. Due to the capability of LDA to discover high level features with long-range semantics dependencies, NetMem with LDA-based reasoning model achieved better performance; higher accuracy and recall ratio and low Fp and Fn rates. We notice that the less information-loss (i.e., larger K values) in data represented in
SiM, the more accuracy NetMem can achieve at learning patterns and extracting semantics. Performance of NetMem with LSH improved as K length increased. NetMem with LSH could achieve better utilization of resources (e.g., minimize SiM storage space) and low processing overhead for detecting attacks and learning their semantics as illustrated in Figs. 14 and 16.

Fig. 18 illustrates Fp and Fn rates of NetMem and comparing other rates obtained by snort in detecting abnormal/normal profiles and attacks. Better results for Fp ratios were obtained with NetMem compared with snort with or without using LSH. Snort was not able to learn correctly normal behavior classes of running flows because snort is considered a stateless firewall in general [43]. As an example, a stateless firewall will prevent legitimate packets concerning a FTP session from passing since it has no knowledge that those packets destined to a protected network adopt a certain host's destination port such as 4970. Also, NetMem, especially at using the LDA-based reasoning model, achieved good levels of Fp ratios relative to snort with/without using LSH. In other words, NetMem with LDA-based reasoning model succeeded in recognizing most of the abnormal traffic behavior and attack classes.

Fig. 19 shows the working data storage space of the implemented reasoning models versus the obtained prediction accuracy using those models. As in the figure, significantly better accuracy was obtained when using the LDA-based model whether using LSH or not. Without using LSH, the implemented semantic reasoning models worked over large-scale of full dimensional data with large number of data attributes which reach 23 attributes. In the case of using LSH, the models worked over reduced dimensional data profiles of lengths 3, 4, and 5 attributes.

![Fig. 19. Space complexity versus accuracy of the implemented reasoning algorithms](image)

From the results above, we can conclude the following:

1. NetMem can provide an effective and efficient capability for learning behavior classes of normal/abnormal network data traffic relying on learning patterns of full- or reduced-dimensionality profiles of traffic data.
2. Implementing a LDA-based semantics reasoning model with capabilities of extracting high-level features improves NetMem performance (e.g., accuracy and false negatives) compared with other two reasoning models;
3. HMM-based reasoning model integrated with unsupervised learning algorithm succeeds in extracting group of associated concept classes with accuracy and low computation overhead;
4. Adopting light-weight simple statistical-based reasoning model achieves good levels of prediction accuracy with low overhead, however, lower than the LDA-based and HMM-based models;
5. The LDA-HMM-based reasoning model can learn latent features and reason about semantics whether using full- or reduced data dimensionality achieving higher NetMem effectiveness and efficiency; and
6. NetMem can be attached with networking tools as intrusion detection and prevention systems (e.g., snort [19]) to enhance their effectiveness and efficiency.

10. Related Work

Researchers have investigated mechanisms and systems for storing Internet and measurement data based on
various attributes. For example, CAIDA in [44] presented the Internet measurement data catalogue (IMDC) as metadata repositories of measurement data to achieve smooth accessibility of that data for comparative analysis purposes. IMDC provided detailed information about stored data such as its source and location and time of its occurrence. Also, authors in [45] proposed a scalable Internet measurement repository to track Internet measurements where large databases provide information about measurements, tools, users, experiments, and datasets. These databases can be accessed easily for analyzing obtained measurements at various contexts and times.

Some previous works have presented systems for managing and processing large sets of data, as in [46], to extract information that will be used to enhance decision making targeting specific goals. In [46], authors presented Apolo, a system to support sense-making for large network data (e.g., data from social networks). Apolo depended on integrating machine learning algorithm, called belief propagation, with user interaction and graph visualization technique. Unlike work at [46], reasoning models for extracting valuable information in NetMem operate autonomously and can support high-dimensional data. NetMem reasoning models (e.g., HMM) can be constructed as prototypes to support specific semantic topics based on interests of network data analyzers.

Some models for semantics and behavior extraction are also found in the literature. Authors in [38] proposed a unified LDA-based model for categorization of web pages’ semantics. They build that model through designing a separate LDA-based model for each category with specific term of topic. Semantics inference process for unknown documents depends on results obtained each LDA-based model and using Gibbs sampling. In [33], Jit et al. proposed an HMM model to support multi-dimensional data in image contexts and extraction of low level features. The proposed HMM model depended on Baum-Welch algorithm and maximum likelihood training to adjust HMM parameters and to find most likelihood semantic topics or features. Authors in [47] adopted HMM models and k-means clustering method for extracting human behavior. It constructs HMM from time series data using the Viterbi algorithm. It builds probabilistic density for frequency of occurrence “discrete” and successive time “continuous” using sequential discounting Laplace estimation and expectation and maximization [48]. Khan et al. [24] provided HMM for discovering abnormal temporal events in sensor networks. The assumption was that unusual events are rare and not enough data are found for training. Different detection models were implemented where abnormal activities were detected if their likelihood were below defined thresholds. Another system uses support vector machine (SVM) for data analysis, classification, and pattern recognition plus Fuzzy rules [49] for discovering human behavior. It performs abstraction of data using SVM-based classification into sequence of actions. It constructs Fuzzy rules for each behavior, defined by a sequence of actions. Others use neural network (NN) and statistical approaches [50] to discover application behavior. It forms a stochastic model to represent application behavior variations using Markov chains. It builds a time series to represent application execution state changes then apply to a NN for predicting future behavior. The authors in [39] discussed two types of statistical models that enable prediction of future events and extracting information from large sets of complex data.

Researchers have also investigated mechanisms for data dimension reduction and semantics extraction in different fields as image processing and pattern recognition. Some works have used NN for extracting semantics [51] and reducing data dimension [52]. In [51], authors adopted NN to extract semantics or interesting events based on learning temporal video features of syntactic data segments. Multi-layer NN was used in [52] to form short-length codes for high dimensional image and document data based on weights assigned by network for extracted binary features.

Differences in NetMem

According to the aforementioned works for Internet data management like [44] and network semantics reasoning, there was no or limited ability: a) to reason about semantics on different levels of abstractions; and b) maintain extracted semantics as concept classes forming runtime accessible ontology of concepts related to various Internet elements (e.g., services and protocols). Additionally, NetMem provides a storage memory structure which comprise STM and LTM to facilitate data patterns learning and semantics reasoning/retrieval for matching and prediction processes.

Compared with related work such as [24, 51, 52], NetMem can provide behavior models related to
specific network concerns through the formed ontology and the discovered FBS aspects per each concept class. NetMem enables discovery of latent features from Internet traffic patterns with reduced data dimensionality. Moreover, NetMem enables extraction of data semantics at different levels of abstraction based on known classified features and related to various network concerns. Accordingly, NetMem forms dynamic runtime accessible ontology of associated network concept classes where classes are maintained updated by clarifying their FBS aspects. Derived concept classes are accessed and learned by Internet elements at runtime for learning novel things (e.g., unfamiliar services’ QoS requirements) and predicting future events (e.g., attacks and anomalies).

11. Conclusion and Future Work

In this paper, we addressed the “Internet Semantic Gap”. Learning and utilizing network data semantics of large-scale and complex networks would lead to smarter networking. Accordingly, networking entities would be able to better recognize dynamic and emergent behavior, which would help enhance QoS satisfaction, behavior discovery, and future event prediction. Inspired by functionalities of human memory, we presented NetMem for managing, at real-time, big data to extract network semantics. We explored hidden Markov models, latent dirichlet allocation, and simple statistical models for efficient semantics extraction. We proposed a HLT integrating LDA and HMM for efficient semantic reasoning processes. We also utilized locality-sensitive hashing for reducing data dimensionality for efficient storage and collection and to have low time-overhead for learning and matching Internet behavior classes. Evaluation of networking operations using real-time Internet traffic data showed the efficacy of NetMem for learning behavior classes of normal/anomalous flows and attacks. Also, NetMem with hybrid intelligence provided better effectiveness and efficiency compared with monolithic intelligence techniques. Future work includes (i) leveraging NetMem’s hybrid intelligence capability of learning dynamic and abnormal behavior to enhance behavior prediction and self-and situation-awareness by the various Internet elements for better performance and resource utilization, (ii) formalizing a methodology for optimizing NetMem system configuration, and (iii) expanding our evaluation to study NetMem as a distributed intelligent multi-agent system with dynamic adaptation using heterogeneous reasoners.

References


**Bassem M. Mokhtar** received his BS and MS degrees in Electrical Engineering from University of Alexandria, Egypt, in 2004 and 2006, respectively. He received his PhD degree in Computer Engineering from Virginia Tech, USA, in 2014. His research interests include bio-inspired network data management, semantic reasoning, network intelligence, semantic-driven networking operations, information management and knowledge-based network management.

**Mohamed Eltoweissy** received his BS and MS degrees in Computer Science and Automatic Control from University of Alexandria, Egypt, in 1986 and 1989 respectively. He received his PhD degree in Computer Science from University of Old Dominion, USA, in 1993. His research interests crosscuts the areas of trustworthy engineering, networking architecture and protocols, and distributed systems for large-scale ubiquitous cyber-physical systems. Eltoweissy is a Professor of Electrical and Computer Engineering and Computer Science. From March 2010 to March 2012, Eltoweissy served as Chief Scientist for Secure Cyber Systems at Pacific Northwest National Laboratory. Eltoweissy is on the editorial board of IEEE Transactions on Computers as well as other notable journals.