Machine Learning of Semi-Autonomous Intelligent Mesh Networks Operation Expertise

Alex Bordetsky
Naval Postgraduate School
abordets@nps.edu

Carsten Glose
German Armed Forces (Bundeswehr) /
Exchange Scientist
carstenglose@bundeswehr.org

Steven Mullins
Naval Postgraduate School
sjmullin@nps.edu

Eugene Bourakov
Naval Postgraduate School
ebourako@nps.edu

Abstract

Operating networks in very dynamic environments makes network management both complex and difficult. It remains an open question how mesh or hastily formed networks with many nodes could be managed efficiently. Considering the various constraints such as limited communication channels on network management in dynamic environments, the need for semi-autonomous or autonomous networks is evident. Exploitation of machine learning techniques could be a way to solve this network management challenge. However, the need for large training datasets and the infrequency of network management events make it uncertain whether this approach is effective for highly dynamic networks and networks operating in unfriendly conditions, such as tactical military networks. This paper examines the feasibility of this approach by analyzing a recorded dataset of a mesh network experiment in a highly dynamic, austere military environment and derives conclusions for the design of future mesh networks and their network management systems.

1. Introduction

The management of tactical military networks is a much harder challenge than conventional network management, but the potential benefits to network centric warfare are enormous [3]. In addition to classical network management, it is important to consider difficulties associated with tactical network operations, such as adversaries trying to disturb or shut down the network, and various other constraints. Network management is especially difficult in dynamic environments and/or complex situations.

For tactical networks with a small number of entities, the management can be done manually by a Network Operations Center (NOC). However, it remains an open question how to manage a large-scale network. Classical network management decision support techniques like rule-based algorithms have not always succeeded in highly dynamic environments. Even some data center professionals who manage large companies or university networks, sometimes do not make use of decision-support components, as they are considered to be impractical and laborious to set up and maintain.

Although some progress has been made [5], further automation of network management is vital even in standard and non-military networks, as various efforts show [9].

Greater automation has been shown to be feasible in managing a decentralized network by using a distributed artificial intelligence [7]. In the field of tactical networks, the idea of distributed network management was adopted by Bordetsky and Hayes-Roth [2]. They propose the concept of hyper-nodes for command and control networks because the fundamental advantages could be demonstrated [1]. However, the details of network management and control systems of the hyper-nodes were not explained at that time.

Recently, Chen et al [4] developed an algorithm for cloud radio access networks based on echo state networks (ESN) that could predict several relevant parameters in a simulated environment, such as users' positions and content request distribution of users.

Although this result could lead to more automated network management, research regarding the automation of network management has tended to focus on standard or mobile networks and ignored networks in contested or austere conditions such as tactical military networks. For these networks, large

URI: https://hdl.handle.net/10125/59562
ISBN: 978-0-9981331-2-6

training datasets do not exist. Additionally, infrequent network management events are a special feature of these networks. Machine learning algorithms like artificial neuronal networks (see e.g. [8]) require large training datasets and sometimes have difficulties to learn infrequently occurring data points. Therefore, it is doubtful that the different approaches seen in commercial networks can be transferred and applied to the environment of tactical military networks. We assume that machine learning techniques cannot fully solve the problem of tactical network management because the limitations of machine learning techniques have a much higher impact in such environments.

We investigate whether and to which degree machine learning techniques could lead to a more automated network management in the field of tactical military networks. We do this by analyzing and applying machine learning algorithms to some real-world data.

2. Experiment Setup and Description

We analyzed the recorded network data of an experiment that employs a Network Control System of unmanned and manned nodes in support of a notional military mission. In different operational areas (offensive and defensive) the experiment examined technical solutions to autonomous vehicle support of tactical military tasks. The goal was to bring together unmanned aerial, surface and underwater vehicles as a collaborative networked system to support a difficult military objective.

The experiment focused on the littoral maritime domain. By using a range of different unmanned systems, it was proposed that military operations could be accomplished more rapidly, effectively, and with a reduced requirement for military personnel to be exposed to risk. One rationale for this was that more robustness through a higher diversity of sensors could be achieved. Another is that with this approach the ability to operate in all domains in and around the littorals could be enhanced.

The experiment was conducted from November 4-13, 2017 on an island off southern California, and consisted of:

- Two Scan Eagle Unmanned Aerial Vehicles
- Two SeaFox Unmanned Surface Vehicles
- Two Remus Autonomous Underwater Vehicles
- One Shield AI Quadrotor UAV

Persistent Systems Mesh radios comprised both the primary and the backup mesh ground networks.

3. Methodology

Mesh network performance data, including the

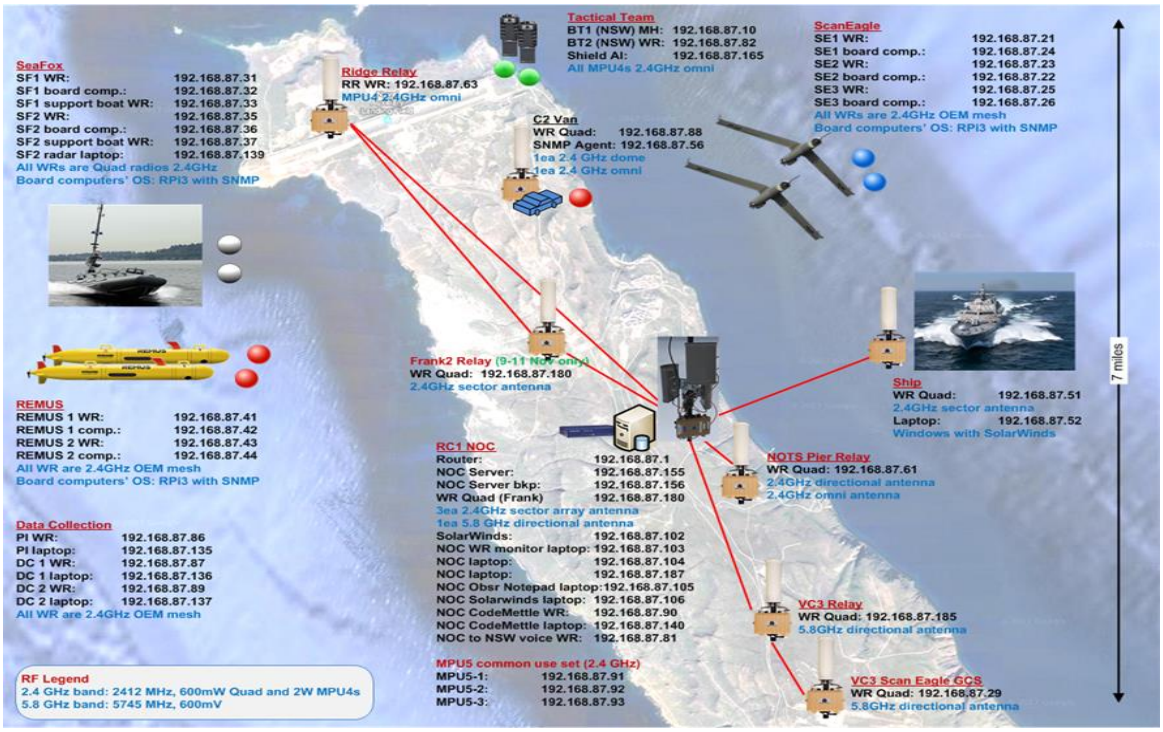

Figure 1. Overview of the experiment

behavior of unmanned nodes was captured and collected based on the Simple Network Management Protocol (SNMP) technique (see e.g. [10]).

To ensure a high-quality dataset, we post-processed the recorded data and filtered out invalid or faulty data items. After this, we performed some basic statistical analysis to learn more about the nature of the data, find interesting properties or even identify features and/or a feature set for the machine learning step. Then we applied several machine learning algorithms like rule-based, lazy learning, tree-based and support vector machines with this dataset to investigate whether machine learning can be used in such an environment.

3.1. Data Collection

Data was collected by an SNMP Agent specifically created for this project utilizing a Node.js framework. Since Node.js is a platform independent environment, the SNMP Agent was able to run on Windows and Linux OS. For this particular experiment, the SNMP Agent was running on Raspberry PI 3 and Odroid microcomputers (Linux OS) added to the unmanned aerial vehicle (UAV) and unmanned surface vehicle (USV) payloads. These agents recorded performance data and also device-specific non-SNMP data such as the GPS position. The data was uploaded to a central database after the experiment. This database consists of online and offline network performance data. In addition to the automated recorded data, significant events, interesting discoveries and relevant information that could not be collected automatically were recorded in textual form, manually entered into the central database. For our analysis, we used the automatically recorded data. The manually recorded data had no standard structure, and it would have been prohibitive to incorporate this into our machine learning analysis. Nevertheless, the manually recorded data was helpful meta-data for cleansing the dataset and to identify and filter out invalid values.

3.2. Dataset Description

In total 135,546 items were recorded. Data items consist of the following attributes in Table 1. A database entry contains more attributes than are listed in Table 1; the additional attributes were not measured in this experiment and contain unknown or invalid values. Therefore, the following attributes were not used in our analysis: OriginType, DestID, Altitude, Speed, Course, EventType, EventDescription, EventStatus, fromGCU, coord_system, heave_vel, roll,

Table 1. Name and description of attributes in each data item

Attribute name	Description	Type
Log	Index of items. Not used by machine learning algorithms.	Numeric (Long)
WhenOccured	Time of event (timestamp), accurate to one second. Not used by ML algorithms.	Numeric, Format: YYYY-MM-DD HH:MM:SS
OriginID	Unique identifier of the sending entity	Numeric (Integer)
Lat	Latitude: decimal degrees.	Numeric (Float)
Longi	Longitude: decimal degrees.	“
platform	Describes if entity is an AUV, UAV, USV) or null (unknown)	Nominal, Values: {AUV, UAV, USV, None}
utm-zone, utm-northin, utm-easting	Position in UTM coordinates. Not used.	
depth	Altitude (meters above sea-level) (negative if under water). Not available for all entities.	Numeric (Float)
forward_vel	Velocity in forward direction in m/sec	“
sideslip_vel	Velocity in m/sec	“
yaw_rate	Rate of yaw in degrees	“
throughputin source: SNMP	Number of incoming packets (size = pktsize)	Numeric (Long)
throughputout source: SNMP	Number of outgoing packets (packets have size pktsize)	Numeric (Long)
rtt source: SNMP	Round-trip time in milliseconds.	Numeric (Float)
pktloss source: SNMP	Number of lost packets in last period	Numeric (Integer)
pktsize source: SNMP	Size of network packet in octets	Numeric (Integer)
Reachable	1, if the SNMP-poller could reach the entity within 30 seconds through the mesh network.0, otherwise. Null, if unknown/not measured.	Nominal Values: {1, 0, Null}
hasSnmpPoller	Flag, if an entity has an SNMP poller. Not used by ML algorithms.	Nominal Values: {1, 0, Null}
timestamp	timestamp in unix epoch (accurate to 1 minute). Not used.	Numeric
snrList	String containing observed SNR in dB to its neighbors. String format is: IP-Address SNR-value, IP-Address SNR-value	Nominal (String)

roll_rate, pitch_rate, IconColor.

Additionally, the attributes Universal Transverse Mercator (UTM) "utm-zone", utm-northing," "utm-easting" and "timestamp" were not used in the analysis, because they are redundant with the attributes "Lat," "Longi" and "WhenOccured" and were measured with similar or lower accuracy.

Values that were not measured were given a null value in the dataset. Some entities' radios contained more than one antenna. In these cases, an IP address can occur more than once (for every antenna) in the list of IP addresses in the attribute snrList.

In addition, a dataset with a map of an IP address to its object name is available (database name "mapObject"). The information in the mapObject database is irrelevant for machine learning purposes and therefore was not used.

3.3. Post-Processing

We cleansed the dataset of invalid or faulty items and removed unused or duplicate attributes that carry the same information, such as a position both in decimal degrees and in UTM format.

Even without the "Lat"/"Lon" attributes, the resulting dataset consisted of a total of 14 attributes (13 attributes and the target attribute "reachable"). To further minimize the dimensionality, we omitted the "snrList" attribute for the moment. Then, every numeric attribute from the resulting set was normalized (range [0-1]) and a principal component analysis was performed.

4. Analysis

In the first step of our analysis, we conducted a data exploration using a simple statistical analysis. This was done to learn more about the nature of the data and to find interesting properties of the underlying dataset. To apply machine learning techniques to the dataset, we sought to reduce its dimensionality.

After preprocessing, the size of the cleansed dataset was quite small for the use of machine learning algorithms. To avoid an overfit of the learned models we did not use some attributes that could give away too much information to the learning algorithm or would reduce possible generalizability. These attributes were "Log," "WhenOccured," "hasSnmpPoller," and in some cases "Lat"/"Lon." In addition to removing attributes to avoid overfitting, we tried to design a system where all decision-relevant attributes could be directly measured by the unmanned device. This is the case for the remaining attributes.

4.1. Principal Component Analysis

The Principal Component Analysis (PCA) for the attribute "reachable" and a variance of 95% covered, resulted in 12 remaining attributes. Although this result fell short of our expectations regarding the reduction of attributes, we found ourselves in the unusual situation where we were able to identify key factors and derive conclusions just by closely looking at the components.

Table 2. Excerpt of PCA result

#	Prop.	Component
1	0.217	0.56 platform=AUV-0.56platform=USV+0.506depth...
2	0.157	-0.541throughputout-0.536pktloss-0.47throughputin-0.313pktsize+0.199OriginID...
3	0.138	-0.664sideslip_vel-0.638forward_vel-0.366yaw-0.057platform=AUV+0.057pla...
4	0.118	-0.663OriginID+0.606pktsize-0.264throughputin-0.23pktloss-...
5	0.077	-0.983yaw_rate-0.098rtt+0.097throughputin-...
6	0.075	0.979rtt-0.119pktsize-0.091yaw_rate-0.063throughputout+0.06 depth...
7	0.056	0.868yaw-0.383forward_vel+0.16 platform+...
8	0.047	0.791throughputin-0.359throughputout-0.328pktloss-0.288OriginID-0.203pktsize...
9	0.034	0.709pktloss-0.514throughputout-0.303OriginID-0.3pktsize-0.146depth...
10	0.029	0.609pktsize+0.557OriginID-0.482throughputout+0.205thr...

4.2. PCA Findings

Platform-specific attributes (platform type) have the most influence (component 1 and 3). This is not surprising as the platforms possessed different capabilities and fulfilled different functions.

A bigger packet size and a larger throughput make reachability harder (component 2). We presume that this is caused by the priority algorithms in the device's network stack. With a higher workload, packets as the SNMP poll request could be dismissed. This part of the system could offer room for improvement.

We found that it matters which entities communicate (components 2, 4). This result is expected because it directly correlates to the "platform" attribute.

Velocity and yaw can have a positive or a negative impact on reachability (components 3, 5, 6, 7). This is an inconclusive result and requires further investigation.

The first four components account for approximately 60% of variance. The rest seems to be quite random and noisy, and without a direct interpretation. Figure 2 depicts this.

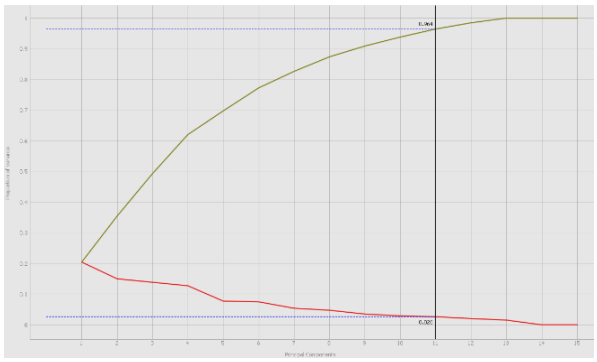

Figure 2. Number of PCA components vs. variance covered

4.3. Statistical Properties of some Informative Attributes

A closer examination of the statistical properties of the original attributes revealed some interesting insights. We found that the attribute "throughputin" seems to have an underlying Gaussian distribution.

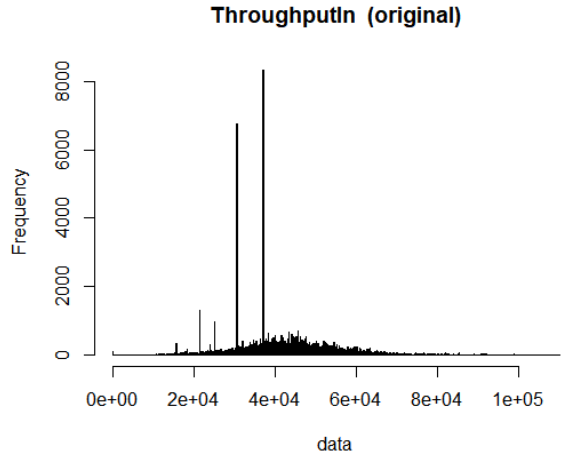

Figure 3. Histogram of "throughputin"

Having said that, it is notable that we found several outliers for certain frequencies (see Figure 3). Several protocols use fixed-size messages. It seems plausible that these outliers are a direct result of this. A similar situation exists for attribute "throughputout" where an underlying superimposition of two Gaussian distributions seems to take place.

We opine that this kind of outlier and the huge variance that we have discovered are a special feature of a tactical mesh networks.

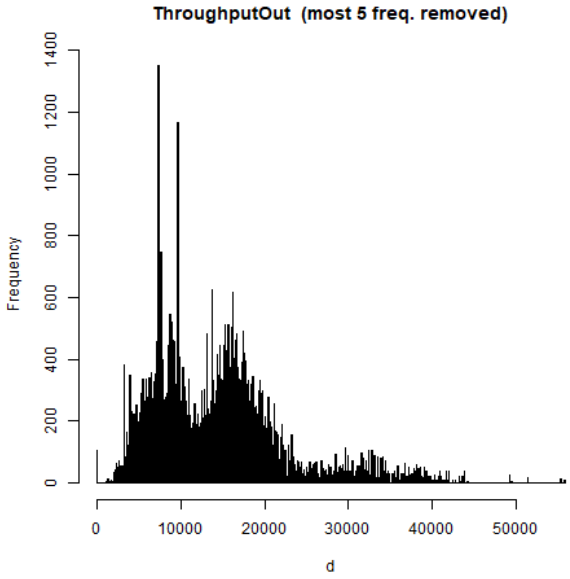

Figure 4. Distribution of "throughputout"

Figure 4 shows the frequency of the values of the attribute "throughputout." The figure was restricted to values under 60,000, and the five most frequent values were removed. Values over 60,000 occurred relatively rarely in the dataset and the Gaussian distribution of the data is hard to see in the full picture (compare e.g. to Figure 3).

We did find a linear correlation between the attributes "throughputin" and "throughputout."

Figure 5 shows the identified linear model for the attributes "throughputin" related to "throughputout". The plot was restricted to values under 30,000 for "throughputout" and values under 60,000 for "throughputin" to clear the clutter of a lot of outliers.

We think that this finding can be explained as a feature of the mesh network. Many incoming messages are forwarded to neighbor nodes and as such, output traffic correlates to input traffic. This indicates that our network design and setup for a mesh network is sound, as there are no "supernodes" which receive and put all the data to the network. In addition, this also means that communication devices used in mesh networks could be designed with symmetrical up- and downlink channels.

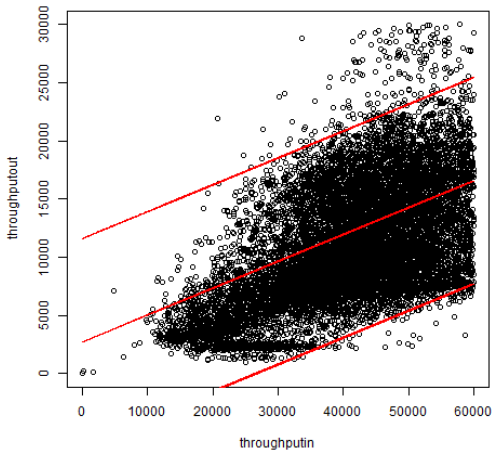

Figure 5. Linear fit for throughputin vs. throughputout

Additionally, we found that a positive value of "yaw" leads to unreachability in higher altitudes (Figure 6).

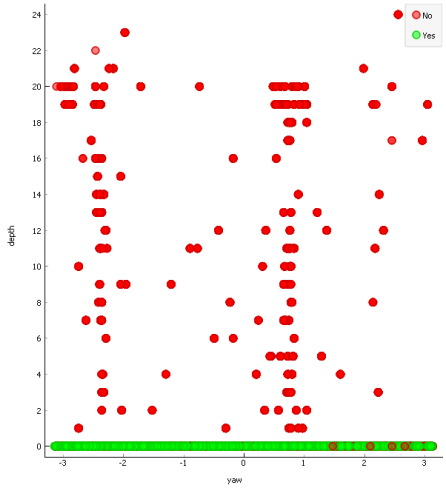

Figure 6. Yaw vs depth (altitude)

Our assumption is that features of the antenna characteristics and subsequently characteristics in the beam pattern lead to a link loss if the device moves or rotates.

In conclusion, our analysis found that there are strong statistical regularities and that all attributes seem to be important. Based on this assessment, we decided to use all remaining 12 PCA attributes for the machine learning step.

4.3. Application of Machine Learning Techniques to the Recorded Dataset

We used several supervised learning methods with the target attribute "reachable" to examine whether learning could be done in this environment. The prior probability of the target attribute is 71.2%. The analysis was conducted with Weka [11] and Orange [6]. We used cross-validation with a 10-fold for each run.

Many classic machine learning algorithms master this particular learning problem (Table 3). Except for Naïve Bayes, Logistic Regression, SVM and Ripper, performance does not differ significantly between the learning algorithms. As 5% of the variance is lost via the PCA transformation, we were surprised that the best learning algorithms have a higher classification rate and were curious whether we could obtain better results by using the original dataset. As it turns out, a very similar performance result is achieved with the original dataset. Interestingly, the kNN and Naïve

Table 3. Result for different machine learning algorithms regarding target attribute "reachable"

Algorithm	Impl.	Correctly Classifica.	F-Score	Remarks
Random Forest	Weka	97.09 %	0.97	Number of trees: 10, No split subsets smaller than 5
kNN	Weka iBK	96.59 %	0.96	5-NN
C 4.5	Weka J48 (prune)	96.45 %	0.96	Size: 3175 Number of Leaves: 1588
Neuronal Network	Orange	95 %	0.95	Hidden Layers: 50,150 Activation: ReLu, Solver:Adam
RIPPER	Weka JRIP	94.86 %	0.94	17 Rules
SVM	Weka (SMO)	92.52 %	0.91	Poly-kernel
Log.Reg.	Weka	91.56 %	0.90	Regularization Ridge (L2), C=1
Naïve Bayes	Weka	56.39 %	0.62	

Bayes algorithms perform very differently between the transformed and untransformed datasets. Whereas kNN benefited enormously (performance of 57.28% correct classification on the untransformed dataset compared to 96.59% on the transformed dataset) from the

transformation, Naïve Bayes suffered (from 84.76% correct classification to 56.39% on the transformed dataset) from the transformation. Closer examination of the learned models (original and transformed datasets) indicates that the models seem to be overfitted. One example of this overfitting is the tree built by the J48 algorithm with a size of 3175 and 1588 leaves. As we do not have a dataset of a different operation available, we have not yet been able to investigate whether and to what extent the models generalize to different scenarios.

5. Conclusions

Our research as it is presented in this paper indicates that the initial assumption (automation of network management in tactical networks is much harder than automation of classical networks and therefore machine learning techniques may be not applicable) was overstated.

We found strong statistical regularities in the recorded network data of the observed mesh network designed to support a tactical military mission. These regular patterns are sufficient to predict relevant network management decision features related to unmanned system operation, subject to changing network performance and configuration conditions. Our analysis is based on one recorded dataset of the performance data of one tactical network and therefore, the results are limited.

Nevertheless, we believe that our findings give some cause to expect that distributed autonomous network management systems for unmanned systems in tactical networks are in the realm of feasibility. On the contrary, the data also shows clearly a much higher degree of variance than is seen in other network data. We assume that these irregularities are the special feature of tactical networks.

Nevertheless, it seems that in the big picture, tactical mesh networks are not so different from classical networks with regard to the question of network management automation. However, it is different when the details of tactical networks are taken into account.

In conclusion, it still seems plausible to us that it is infeasible to fully automate management of tactical military networks. It is unclear how machine learning algorithms could meet the challenge of unprecedented forms of attacks to a tactical network. Having said that, we propose the concept of semi-automation of network management for tactical military networks. This means that autonomous nodes perform the easy and regular parts of network management (hyper-node concept). In this refined concept, machine learning techniques are used to enrich the decision support systems of the hyper-nodes. A network operations center remains in charge of the main network operation but the task shifts from monitoring and controlling the network to dealing with unprecedented or very exceptional situations. The hyper-nodes help to quickly identify irregularities in network behavior using their autonomous intelligence and report this to the NOC decision makers. The NOC crew analyzes the situation and takes appropriate action.

Due to the fact that few datasets of tactical mesh networks are available, we call upon others to conduct similar experiments and collect more data in this area of research. It is our understanding that the continuing process of experimentation and data collection generates a much-needed and valuable network knowledge base. This helps to develop machine learning systems in operating semi-autonomous tactical networks. Experimentation and data collection is one of the major tasks on which our team is planning to concentrate our future efforts.

In particular we would like to conduct a series of experiments that is similar to the one described above in terms of scale, type of manned-unmanned nodes, and their mobility. We would want to see whether the ML algorithm would be able to generalize the rules of network performance management and nodes' mutual adaptation. Based on a series of experiments with a similar tactical scale, node types and tactical scenarios, we would explore whether other, different patterns of node performance adaptation emerge, which we could capture in an ML algorithm.

Based on more data captured during similar tactical scenarios, we would develop an adaptive network management simulator to be integrated in the hyper-nodes' mutual adaptation in real-time. This would create an element of data analytics for use by human operators in conjunction with the ML actions executed by mutually adapting machines.

Note: A Machine Learning algorithm might accidentally learn the features of the simulator. Everything must be validated by real data. For example, our criticism to Chen et al [4] is that they have used their simulator most of the time to create the algorithm. The simulator would be fed real-data; however it might be the case that their algorithm learns the features of the simulator and didn't generalize.

Our data can be found at the following website: https://nps.box.com/s/hx3djmibiz8mot48y37aelncqwfi5lbm

6. Acknowledgements

The authors would like to thank Dr. Ray Buettner for the opportunity to conduct this study and Dr. Doug

Horner for his invaluable contributions in putting together and conducting the experiment.

7. References

[1] Bordetsky, A. and Hayes-Roth, R., 2007. Extending the OSI model for wireless battlefield networks: a design approach to the 8th Layer for tactical hyper-nodes, In *International Journal of Mobile Network Design and Innovation*. Inderscience Publishers

[2] Bordetsky, A. and Hayes-Roth, R., 2006. Hyper-Nodes for Emerging Command and Control Networks: The 8th Layer, In *11th International Command and Control Research and Technology Symposium (ICCRTS)*

[3] Burbank, J. L. et al., 2006. Key Challenges of Military Tactical Networking and the Elusive Promise of MANET Technology, In *IEEE Communications Magazine, vol. 44, no. 11, pp. 39-45, November 2006.*

[4] Chen, Mingzhe, et al., 2016. Caching in the Sky: Proactive Deployment of Cache-Enabled Unmanned Aerial Vehicles for Optimized Quality-of-Experience. In *IEEE Journal on Selected Areas in Communications 2016, 10.*

[5] Chiang, C. y. J., et al., 2006. Towards Automation of Management and Planning for Future Military Tactical Networks. In *MILCOM 2006 - 2006 IEEE Military Communications conference, Washington, DC, 2006, pp. 1-7*.

[6] Demsar J., et al., 2013. Orange: Data Mining Toolbox in Python. In *Journal of Machine Learning Research 14, pp. 2349-2353.*

[7] Koch, F., et al., 2004, Distributed Artificial Intelligence for Network Management Systems --- New Approaches. In *Service Assurance with Partial and Intermittent Resources*. Springer Berlin Heidelberg.

[8] Kotsiantis, S. B., 2007. Supervised Machine Learning: A Review of Classification Techniques. In *Proceedings of the 2007 Conference on Emerging Artificial Intelligence Applications in Computer Engineering: Real Word AI Systems with Applications in eHealth, HCI, Information Retrieval and Pervasive Technologies*. IOS Press.

[9] Sohail, S., 2010. Automation of Network Management with Multidisciplinary Concepts. In *International Journal of Computer Technology and Applications, 2010, 11*.

[10] Subramanian, M., 2010. *Network Management: Principles and Practice*. Prentice Hall; 2nd edition.

[11] Witten, Ian H., et al, 2016. *The WEKA Workbench. Online Appendix for "Data Mining: Practical Machine Learning Tools and Techniques"*. Morgan Kaufmann, 4th edition, 2016.