DOI: 10.15439/2023F8996
ISSN 2300-5963 ACSIS, Vol. 35

Online Learning Framework for Radio Link Failure Prediction in FANETs

Kiril Danilchenko
Department of Electrical and Computer Engineering
University of Waterloo
Waterloo, ON, Canada
kdanilch@uwaterloo.ca

Nir Lazmi
School of Electrical and Computer Engineering
Ben-Gurion University of the Negev
Beer-Sheva, Israel
nirlazm@post.bgu.ac.il

Michael Segal, *Senior Member*, IEEE
School of Electrical and Computer Engineering
Ben-Gurion University of the Negev
Beer-Sheva, Israel
segal@bgu.ac.il

***Abstract*—In this paper, we consider the problem of prediction of Radio Link Failures (RLF) in flying ad hoc networks (FANETs). Many environmental factors that influence the quality of radio wave propagation are dynamic, and thus, drones must continually learn and update their radio link quality prediction model while they operate online.**

Online machine learning algorithms can be used to build adaptive RLF predictors without requiring a pre-deployment effort. To predict the RLF, we use an online machine learning algorithm and information gathering by message-passing from the neighbors. We propose an algorithm called *ML-Net* (Machine Learning and Network algorithm) to predict RLF. To the best of our knowledge, the combination of online machine learning algorithms together with the message-passing algorithm has not been used before. The proposed methodology outperforms the state-of-the-art online machine learning algorithms.

***Index Terms*—Online learning, RLF prediction, UAV.**

I. Introduction

UNMANNED aerial vehicles (UAVs), also known as drones or flying robots, have gained significant attention in various real-life applications. The flying ad hoc network (FANET) is established to leverage high-speed communications. However, due to the high mobility of UAVs in FANETs, the network topology may continuously change, making it challenging to establish end-to-end connections. Radio Link Failure (RLF) prediction can help UAVs handle this issue and improve FANET performance to ensure continuous service availability. Accurate prediction of radio link failures (RLF) is critical for ensuring reliable communications in flying ad hoc networks (FANETs). However, link prediction remains challenging due to the dynamic topology and unpredictable mobility patterns in FANETs. Nodes can move in and out of communication range rapidly, leading to frequent link disruptions. The ability to accurately predict impending link failures can enable proactive mitigation strategies. For example, drones could switch to more reliable links ahead of time to prevent packet loss and service interruptions. Link prediction also allows optimizing routing by avoiding unstable links that are about to fail. Furthermore, timely knowledge of upcoming link losses enables adapting transmission parameters to maintain connectivity. The development of link prediction techniques tailored to the FANET environment is therefore essential for efficient network operation and robust aeronautical communications. Machine learning holds promise for developing adaptive, data-driven predictors that can operate in real-time based on local interactions. This motivates our exploration of online learning combined with message passing for high-accuracy RLF prediction in FANETs.

Two essential characteristics should be present in RLF predictors. First, adaptivity is critical since an RLF predictor must cope with quality fluctuations over time, especially for FANET. Second, plug-and-play is crucial since RLF predictors should be applied without requiring any predeployment effort, which might not be feasible for all deployment scenarios, even if the effort is reduced. Online machine learning algorithms can be used to build RLF predictors that are adaptive without requiring predeployment effort.

Most current machine learning approaches are limited to the traditional batch setting, where data is provided in advance to the training process. Model selection and meta-parameter optimization can rely on the full set of data, and training can assume that the data and its underlying structure are static. In contrast, online machine learning involves continuous model adaptation based on constantly arriving data. The underlying distribution of the data, which changes over time, presents some primary challenges and difficulties in the dynamic environment. Old data can become irrelevant or even detrimental to model the current concept. Online machine learning [3] exhibits great potential for performance improvement with the sequential arrival of data and superiority over offline learning, including real-time predictions and lower memory requirements.

This paper explores the possibility of using online machine learning algorithms together with local information gathering by messages to predict RLF. Each drone uses only its local information and information gathered from its neighbor. Specifically, the main contributions of this paper are as follows.

1) Motivated by the characteristics of RLF predictors mentioned above, we propose a link failure prediction model *ML-Net*, which combines the online machine learning algorithm with a message-passing algorithm.
2) To the best of our knowledge, we pioneer the use of the online machine learning method combined with the message-passing algorithm for RLF prediction.
3) We conduct simulations to analyze the performance of the *ML-Net*. We compare the proposed solution with state-of-the-art techniques and perform an analysis of simulation results. The analysis shows that the proposed approach *ML-Net* achieves better performance than the state-of-the-art methods.

The remainder of this paper is organized as follows. Section II discusses recent related studies. Section III describes the theoretical model used to define our problem. The details of the proposed approach are described in Section IV. In Section V, we describe all aspects of the evaluation setup and simulation results. Finally, we summarize our paper with conclusions and suggestions for future research in Section VI.

II. Related Work

In this study, we propose the use of online machine learning algorithms combined with message-passing approach for predicting Radio Link Failure (RLF) in communication networks. Several recent studies on RLF suggest using online machine learning algorithms for prediction.

In [1], the authors studied the link quality prediction for wireless mesh networks. They performed a performance analysis of four state-of-the-art algorithms for link quality prediction and proposed a new hybrid online algorithm for link quality prediction based on this analysis.

The authors of [7] presented an adaptive link estimator (TALENT) that uses online learning. They argued that TALENT adapts to network dynamics better than statically trained models without the need for advance data collection for training the model before deployment in a real system. The solutions follows the performance of the autoencoders very closely with a tiny margin on very bad, very good and intermediate link quality classes.

The study [8] introduced a framework to reduce energy and network capacity overhead expenses by incorporating active learning to selectively label only a portion of the samples from the data stream. The framework also uses incremental training batches to conserve labeling resources and updates the batches using change detection and forgetting mechanisms to mitigate concept drift. Experimental results showed that the framework reduces label queries by up to 21.5% and prediction error by up to 9% after periods of concept drift.

As a part of their work in [12], the authors looked at the problem of predicting channel quality between vehicles in terms of path loss, which shows strong fluctuations over time due to the highly dynamic nature of vehicular environments. They proposed a framework for a data-driven path loss prediction model that combines the changepoint detection method and online learning. The evaluation of the proposed framework was done using real-world datasets.

Machine learning methods were used in [10] to predict the short-term evolution of link quality for switching to a better link for data transmission. The problem was modeled as a game of prediction based on experts' advice, using the Link-Quality Indicator (LQI) metric. A decision-maker predicts the LQI values, called a forecaster, who receives advice from several experts. To predict values close to actual LQI values, the forecaster can learn how to adapt its strategy. As a general model, the proposed learning and prediction model can be easily adapted to different link-quality metrics or prediction methods.

In summary, some recent works have used online machine-learning algorithms to predict RLF and took into consideration the adaptability issue, see [2]. Based on these previous studies, we propose the use of a novel method based on online machine learning with a combination of neighbor information gathering by a message-passing solution.

III. System Model and Problem Formulation

We consider a set of n UAVs (denoted as S) that are deployed arbitrarily in a given area and may be located in any position in $\mathbb{R}^3$. The network imposed on S is connected.

We assume that the transmission power of all nodes is fixed. We denote the transmission power of node i at time t as P_i^t. The set of nodes i's neighbors at time t depends on the current positions and channel gain. Based on [13], we can approximate the path loss on link ij as follows:

$$L_{ij}^t = -10\log_{10} G_l \left(\frac{\lambda}{4\pi d^t}\right)^2, \tag{1}$$

where G_l is the product of the transmitter and receiver antenna field radiation patterns of LOS transmissions, d^t is the distance between a transmitter and its corresponding receiver at time t, and λ is the operating wavelength. Thus, the path loss is related to the transmission distance when transmitting radio signals over a specific channel through a specific antenna.

Let N_i^t be the set of neighbors of node i at time t. Formally, N_i^t is defined as:

$$N_i^t = \{j \in S \setminus i | L_{ij}^t \geq \theta\}, \tag{2}$$

where θ is a predefined parameter of the network.

A network is defined as a directed graph $G^t = (S, E^t)$, where node $i \in S$ has a directed edge (link) to each of its neighbors according to Equation 2. The edge set E^t is the union of all directed edges among the nodes in time t, $E^t = \bigcup_{i\in S} E_i^t$.

At each timestamp t, when node i receives a message from node j, node i collects a set x_t of metrics (features) that describe the communication with node j at time t. Node i adds x_t to a multivariate time series that represents features of communication with node j, denoting this multivariate time series as $X_{ij}^t = \{x_1, \ldots, x_t\}$.

Thus, each edge $e_{ij}^t \in E^t$ has a multivariate time series associated with it, X_{ij}^t. In addition, e_{ij}^t has an associated binary class variable $v_{ij}^t \in \{1, -1\}$, representing the existence (1) or failure (-1) of the edge.

Now we can formally define our problem. Given the current X_{ij}^t, we wish to determine whether the edge e_{ij} will fail or not in the following timestamp (namely, at $t+1$).

IV. *ML-Net*

In this section, we describe the proposed *ML-Net* approach. We combine the machine online learning algorithm with the network structure algorithm to cope with RLF prediction in the next timestamp. We suggest using the expression $S(X_{ij}^t, m_{i\leftarrow j}^t)$ which approximates the probability that the edge e_{ij}^t will fall in the next timestamp.

$$S(X_{ij}^t, m_{i\leftarrow j}^t) = \begin{cases} \alpha \hat{Pr}(v_{ij}^{t+1} = -1 | X_{ij}^t) + (1-\alpha) m_{i\leftarrow j}^t, & \text{Message from node } j \text{ was received} \\ \hat{Pr}(v_{ij}^{t+1} = -1 | X_{ij}^t), & \text{otherwise} \end{cases} \tag{3}$$

where $0 \leq \alpha \leq 1$, $m_{i\leftarrow j}^t$ is the message sent by j to i including the belief of node j about v_{ji}^t, and $\hat{Pr}(v_{ij}^{t+1} = -1 | X_{ij}^t)$ is the probability that edge e_{ij} will fail in timestamp $t+1$ given multivariate time series X_{ij}^t. We use on the shelf online machine learning algorithm to calculate $\hat{Pr}(v_{ij}^{t+1} = -1 | X_{ij}^t)$ (in Section V-A we will broadly describe the proposed algorithms).

$$m_{i\leftarrow j}^t = \begin{cases} \alpha \hat{Pr}(v_{ji}^{t+1} = -1 | X_{ji}^t) + (1-\alpha) m_{j\leftarrow i}^t, & \text{Message from node } j \text{ was received} \\ \hat{Pr}(v_{ji}^{t+1} = -1 | X_{ji}^t), & \text{otherwise} \end{cases} \tag{4}$$

Due to the fact that the $G^t = (S, E^t)$ is a directed graph, then the following scenario may occur: the edge $e_{i,j}^t$ exists but the edge $e_{j,i}^t$ does not exist. In this scenario, $m_{i\leftarrow j}^t$ may arrive but $m_{j\leftarrow i}^t$ cannot arrive. In Figure 1 we can see an example of this scenario.

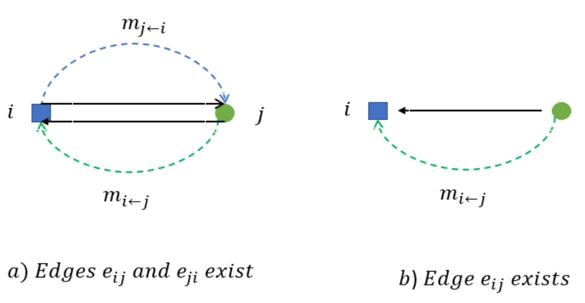

Fig. 1: Directed edges.

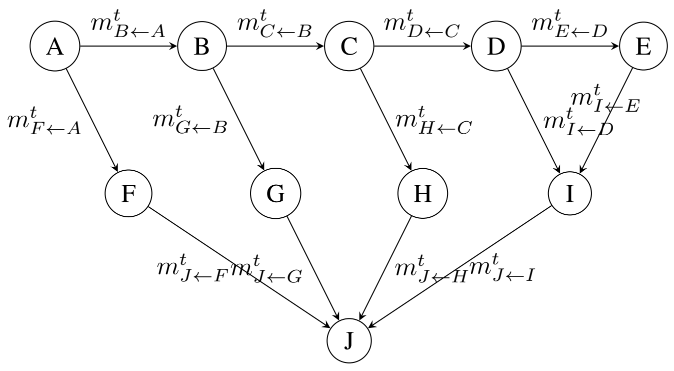

Fig. 2: Illustration of Message-Passing Algorithm

Figure 2 showcases a random graph with 10 nodes (A-J), representing entities in a system. The directed edges between nodes symbolize the flow of messages, demonstrating the operation of the message-passing algorithm. The labels on the arrows, m_{ij}^t, represent the messages passed from node i to node j at time t. This graph serves as a visual representation of the algorithm's functioning in a system with multiple interacting components.

Algorithm 1 ML-Net Algorithm

```
1: Input: Node $i$, current time $t$, set of neighbors $N_i^t$, link features $X_{ij}^t$
2: Output: Link failure prediction $S(X_{ij}^t, m_i^{t\leftarrow j})$
3: Hyperparameters: $\alpha$
4: for $j \in N_i^t$ do
5:     if $msg_received(j, t)$ then
6:         $m_i^{t\leftarrow j} = \alpha \hat{P}(v_{ij}^{t+1} = -1 | X_{ij}^t) + (1-\alpha) m_j^{t\leftarrow i}$
7:     else
8:         $m_i^{t\leftarrow j} = \hat{P}(v_{ij}^{t+1} = -1 | X_{ij}^t)$
9:     end if
10:    $S(X_{ij}^t, m_i^{t\leftarrow j}) = \alpha \hat{P}(v_{ij}^{t+1} = -1 | X_{ij}^t) + (1-\alpha) m_i^{t\leftarrow j}$
11: end for
12: return $S(X_{ij}^t, m_i^{t\leftarrow j})$
```

The ML-Net algorithm shows the key steps for combining online machine learning with message passing to predict link failures. For each neighbor j, node i first checks if a message was received from j at time t. If so, i updates its belief about the $i \to j$ link failing using both its own prediction and j's belief from the message. This allows propagating information through the network. If no message is received, i relies only on its own link prediction. Finally, the algorithm returns the combined prediction score S for each link $i \to j$ based on the updated beliefs. This demonstrates how ML-Net leverages both local online learning models and information exchange with neighbors to achieve accurate real-time RLF prediction in dynamic FANET environments.

A. *Notations*

The notations used in this paper are summarized in Table I. Also, Table I contains further notation defined below in the paper.

Symbol	Meaning
S	Set of drones
L_{ij}^t	Path loss between receiver j and transmitter i
n	Size of set S, $n = \lvert S \rvert$
$\mathcal{K}$	The set of sources and destination in the network
$G^t = (S, E^t)$	Directed graph that represents the network in timestamp t
X_{ij}^t	multivariate time series associated with measures from link ij
P_i	The transmission power
v_{ij}^t	Binary variable represented the link failure between node i to j in time t
e_{ij}^t	Edge between node i to j in time t
$m_{i \leftarrow j}^t$	Message sent by j to i (contains v_{ji}^t)

TABLE I: Summary of notations used in this paper

V. Experimental Setup

Simulation is the most reliable and cost-effective approach to examining the performance of real-world problems. For performance evaluation of the proposed technique, OMNET++ [15] is used as the primary tool for FANET environment generation and routing protocol implementation. This section describes the data generation process. Let $\mathcal{K}$ represent the set of the source and destination pairs within the network that should communicate.

We simulated the network for $|S| = 20$ and for $|\mathcal{K}| = 3$. In Figure 3 we can see an example of a network with 20 UAV's and $|\mathcal{K}| = 3$.

The routing protocol we chose to use is AODV (Ad-Hoc On-Demand Distance Vector) [14]. This protocol is designed for wireless and mobile ad hoc networks and has been broadly adopted for FANETs. Next, we present the schemes for comparison and the evaluation metrics.

A. Schemes for Comparison

In the following, we briefly describe the state-of-the-art online machine learning algorithms and their hyperparameters used in this paper to compare with *ML-Net*. These algorithms were selected as the comparison baseline solutions due to their advanced nature.

1) The Extremely Fast Decision Tree (EFDT) [9]: EFDT constructs a tree incrementally. The EFDT seeks to select and deploy a split as soon as it is confident the split is useful and then revisits that decision, replacing the split if it becomes evident that a better split is available. The EFDT learns rapidly from a stationary distribution, and eventually, it learns the asymptotic batch tree if the distribution from which the data are drawn is stationary. We chose a EFDT with the following hyperparameters: grace period of 200 and split confidence of $1e^{-7}$.

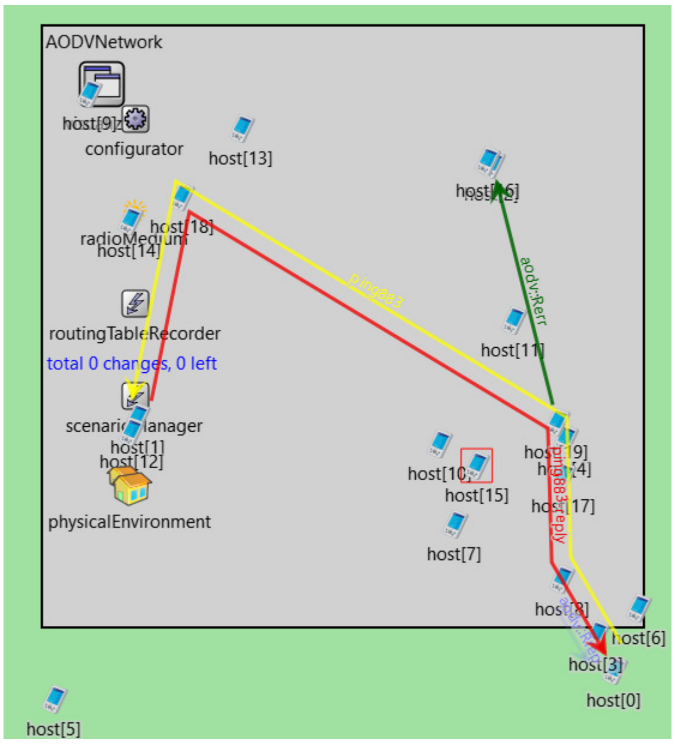

Fig. 3: Example of network with 20 nodes and 3 pairs of source and destination.

2) Adaptive Random Forest (ARF) [4]: ARF algorithm for the classification of evolving data streams enables the Random Forests algorithm for evolving data stream learning. There are effective resampling methods and flexible operators in ARF that can cope with different types of concept drifts without requiring complex optimizations for different data sets. We chose a ARF with the following hyperparameters: grace period of 50, lambda is 6, max features are 3, split confidence of 0.01, and tie threshold of 0.05.

3) Hoeffding Tree Classifier (HTC) [6]: This method involves determining an upper bound for the learner's loss based on the number of examples used in each step of the algorithm. In this algorithm, the number of examples required for each step is minimized while ensuring the model performance is not significantly different from the one obtained from an infinite dataset. We chose a HTC with the following hyperparameters: grace period of 200, min samples reevaluate of 20, split confidence of $1e^{-7}$, and tie threshold of 0.05.

4) Streaming Random Patches Classifier (SRPC) [5]: The Streaming Random Patches algorithm is a machine learning technique that involves randomly selecting subsets or "patches" of data from a continuous stream of input. These patches are used to train and update a model iteratively over time, allowing the model to adapt and learn from new incoming data.

B. Evaluation Metric

We use the following performance metrics to evaluate the performance of different machine learning algorithms:

1) Precision: The ratio of the number of true positives over the total number classified as positive. In the context of our problem, it is the ratio of correctly predicted link failures versus the total number of link failures predicted. The precision value is computed as follows:

$$P = \frac{T_P}{T_P + F_P}, \tag{5}$$

where P is the precision value, T_P is the number of "true positives," and F_P is the number of "false positives."

2) Recall: The ratio of the number of data points associated with link faults correctly classified over the total the number of data points associated with link faults that have occurred. The recall value is given by

$$R = \frac{T_P}{T_P + F_N} \tag{6}$$

3) F_1-Score: The F_1-Score is the harmonic average of the precision and recall values. It takes a value in $[0, 1]$. Higher the value of F_1-Score, the better the performance of the machine learning technique. It is computed as follows:

$$F_1 = \frac{2P \cdot R}{P + R} \tag{7}$$

4) Cohen's kappa- κ : Cohen's kappa measures the agreement between two raters who each classify N items into C mutually exclusive categories the Cohen's kappa formula can be written as:

$$\kappa = \frac{2(T_P \cdot T_N - F_N \cdot F_P)}{(T_P + F_P) \cdot (F_P + T_N) + (T_P + F_N) \cdot (F_N + T_N)} \tag{8}$$

C. Features

This section provides an exploratory data analysis of key features in the communication system. Visualizations are utilized to understand feature distributions and relationships in the data.

1) **Time**: This feature represents the elapsed time since the start of the simulation or experiment. It captures the dynamics of the communication system over time.
2) **IdTransmitter**: This categorical feature identifies the transmitter for each signal received. It allows the model to capture any specific characteristics related to individual transmitters.
3) **64-QAM SNR**: These features represent the Signal-to-Noise Ratio (SNR) for the 64-level Quadrature Amplitude Modulation (64-QAM) scheme. SNR is a measure of signal quality, with higher values generally indicating better quality.
4) **64-QAM BER**: These features represent the Bit Error Rate (BER) for 64-QAM. BER is another measure of signal quality, with lower values indicating fewer errors and therefore better quality.

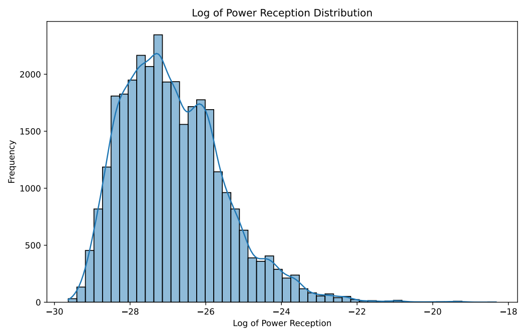

Fig. 4: Distribution of Log Received Power

5) **powR**: This feature represents the power of the received signal. It is a critical factor in communication systems, as a stronger received signal usually implies a better communication link.
6) **angle_between**: This feature represents the angle between the transmitter and receiver. The relative orientation between these devices can affect the signal strength and thus the link quality.
7) **Link**: This is the target variable that we want to predict. It is a binary variable representing the quality of the communication link, with '1' indicating a good link and '0' indicating a bad link. This allows us to frame the problem as a binary classification task.

These features provide a comprehensive description of each signal reception event, capturing information about the transmitter, signal quality, received signal power, and relative device orientation. This enables construction complex models to predict the communication link quality.

Figure 4 shows the histogram that presents the distribution of the logarithm of the received Power Reception. The x-axis shows the log power values, and the y-axis shows the frequency. The kernel density estimate smooth line approximates the probability density function.

The log transformation handles skewed data by making the distribution more symmetric. The log-transformed power distribution appears Gaussian, concentrated around -26.

The KDE peak indicates the most frequent log power value. The distribution spread provides insights into power variability. This transformation can help modeling techniques assume normality.

Figure 5 is a histogram that shows the distribution of angles between transmitters and receivers. The x-axis represents the angle from 0 to 180 degrees. The y-axis represents the frequency. This reveals common orientations in the simulation.

The overlaid Kernel Density Estimate (KDE) approximates the distribution. It assists in identifying shape characteristics like peaks. These insights are key to understanding angle variability and its potential impact on system performance.

The heatmap presented in Figure 6 illustrates feature correlations. Each cell shows the correlation coefficient between feature pairs. Cool colors indicate negative correlations. Warm

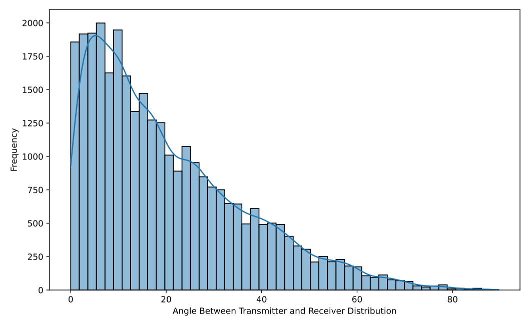

Fig. 5: Distribution of Transmitter-Receiver Angle

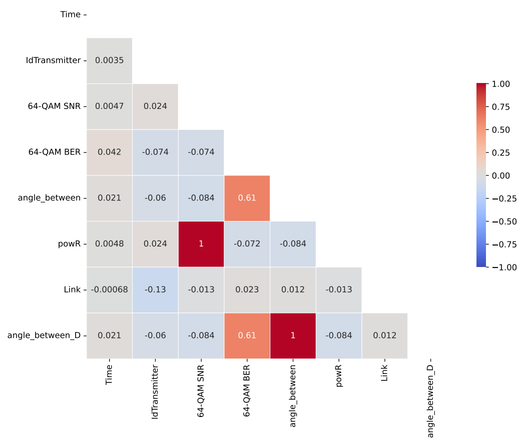

Fig. 6: Correlation Matrix

colors indicate positive correlations. Neutral colors indicate minimal correlation.

This visualization identifies relationships between features. It assists in feature selection by highlighting potential redundancies.

D. Evaluation and Numerical Results

We evaluate the proposed method using data collected from simulations implemented in OMNET++ [15]. As part of our simulation, we randomly deployed 20 drones and randomly selected three pairs of source destinations from these drones as shown in Figure 3 for an example. The drones run the AODV routing protocol.

Each node participating in communication extracts the following parameters (features) when receiving a message from its neighbor: reception power, transmission power, SINR, modulation, location, and orientation. Features collection does not require particular messages. Simulation runs for 3000 seconds, during which the link failure is collected.

Table II summarizes the parameters used by us in the simulation.

We first used the schemes from Section V-A to predict RLF so that we could compare our method with their performance. The compression mechanism is implemented using [11]. We calculate $\hat{Pr}(v_{ij}^{t+1} = -1|X_{ij}^t)$ using all the schemes for comparison, estimate the RLF by Eq. 3, and provide the performance for each of the cases.

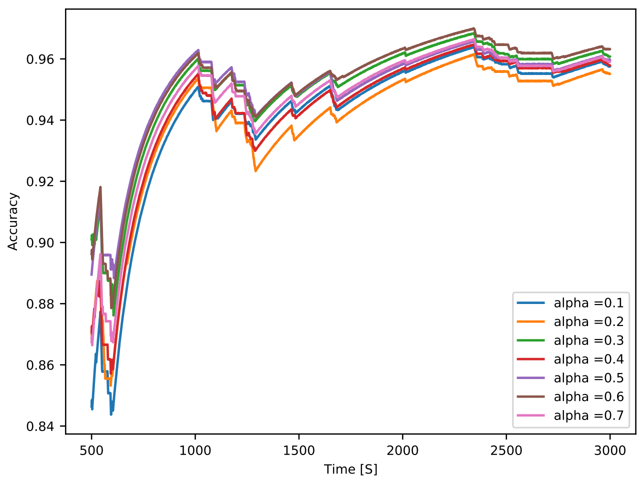

Fig. 7: α selection for *ML-Net*

Figure 7 shows the performance of *ML-Net* as a function of α. The x-axis represents the time of the simulation, and the y-axis represents the accuracy of the RLF prediction. As we can learn from the figure, the accuracy of the RLF prediction generally increases as we increase the value of α. This is because a larger value of α gives more weight to the node's own information and less to its neighbor's information. We can also see that when α is very small, the algorithm is overly influenced by the neighbors' information, and the predictions may not be accurate enough. On the other hand, when α is very large, the algorithm may not be able to adapt quickly to changes in the network topology due to a lack of influence from the neighbors' information. Based on Figure 7, we chose the alpha value to be 0.6, as it strikes a balance between the node's own information and its neighbor's information and achieves the best accuracy value.

Parameter	**Value**
Number of drones	20
Simulation Playground Size	800 $\times$ 800 m^2
Bandwidth	2MHz
σ	10^{-3}
Power	1.4mW
λ	10dB
SINR threshold	4dB
α	0.6
Speed	$\mathcal{N}(200, 0.001)mps$

TABLE II: Simulation Configuration

We provide a detailed explanation of the performance analysis of *ML-Net* with different online learning algorithms in calculating $\hat{Pr}(v^{t+1}ji = -1|X^tji)$. To evaluate the performance of our proposed approach, we compare it with other

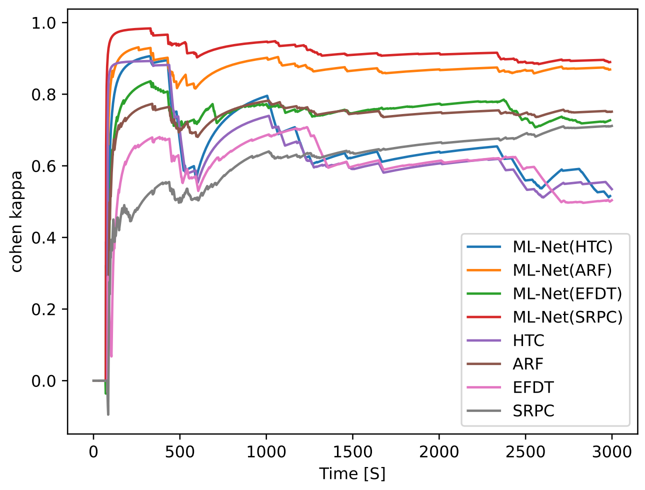

Fig. 8: κ achieved by schemes for comparison versus *ML-Net*

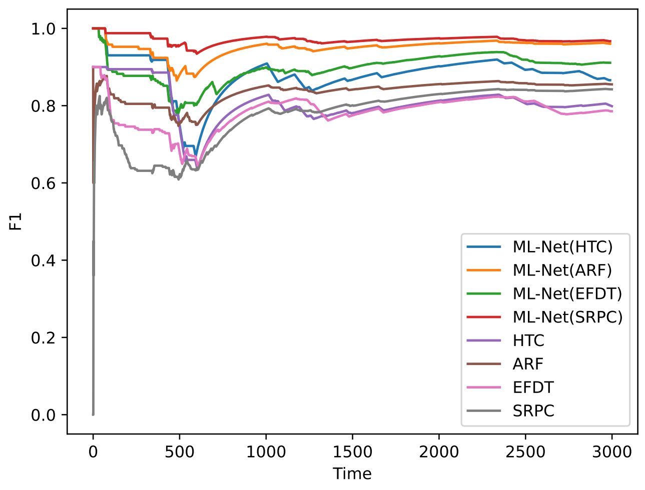

Fig. 9: F_1

existing schemes (see subsection V-A). The evaluation is based on several performance metrics, including precision, recall, accuracy, Cohen's kappa, and F_1.

Figures 9, 10, 11 and 12 demonstrate the performance of *ML-Net* with different online learning algorithms in calculating $\hat{Pr}(v^{t+1}ji = -1|X^tji)$. We observe that the proposed method outperforms the comparison schemes, regardless of the online learning algorithm used. This indicates that our approach is not dependent on the specific online learning algorithm utilized to calculate $\hat{Pr}(v^{t+1}ji = -1|X^tji)$.

Specifically, the evaluation was carried out using several performance metrics. Precision refers to the fraction of correctly identified negative instances among all the predicted negative instances. The recall is the fraction of correctly identified negative instances among all the actual negative instances. Accuracy refers to the fraction of correctly predicted instances among all the instances. Cohen's kappa is a statistical measure of inter-rater agreement between two raters for categorical items. Variable F_1 is the harmonic mean of precision and recall.

The results of the performance analysis are depicted in the figures mentioned above. The results show that the proposed approach outperforms the comparison schemes in terms of all the performance metrics evaluated. Therefore, we can conclude that *ML-Net* is an effective method for calculating $\hat{Pr}(v^{t+1}ji = -1|X^tji)$ in an online learning setting.

To summarize, we see that *ML-Net* outperforms, for each of the evaluation metrics, the previously known scheme that achieves the best performance for this metric.

VI. Conclusions

This paper addresses the critical issue of Radio Link Failures in FANETs, which stands as a significant challenge requiring innovative solution. We propose a novel approach that combines online machine learning algorithms with message-passing, a technique that has not been explored in this context before. Our proposed solution, known as *ML-Net*, significantly

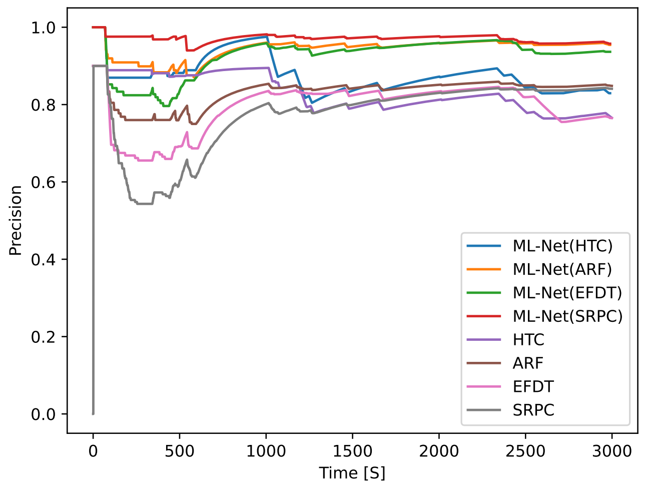

Fig. 10: Precision achieved by schemes for comparison vs. *ML-Net*

outperforms the best existing competitors across all evaluation metrics, indicating the immense potential of this approach in addressing real-time prediction challenges in communication networks.

Our obtained results are highly promising and suggest that this novel combination of online learning algorithms and message-passing algorithms has the potential to revolutionize the field of FANETs and pave the way for more effective and efficient communication networks in the future. Also, it would be interesting to analyze analytically the influence of parameter α.

Acknowledgments

This research was (partially) funded by the Israeli Science Foundation (Grant No. 465/22), Israeli Ministry of Science (Grant No. 0005355), and by the Army Research Office under Grant Number W911NF-22-1-0225. The views and conclu-

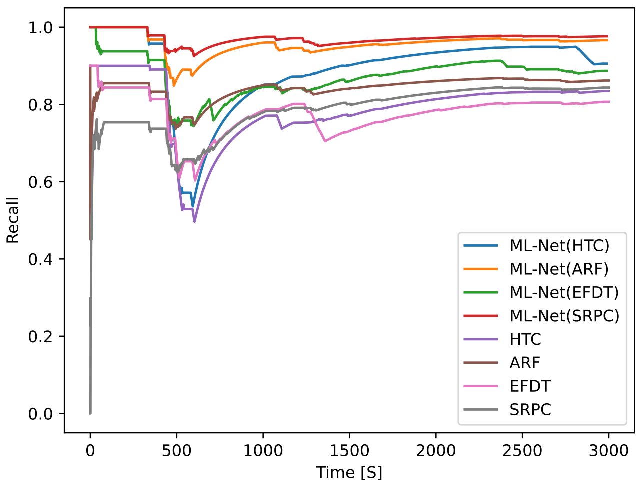

Fig. 11: Recall achieved by schemes for comparison vs. *ML-Net*

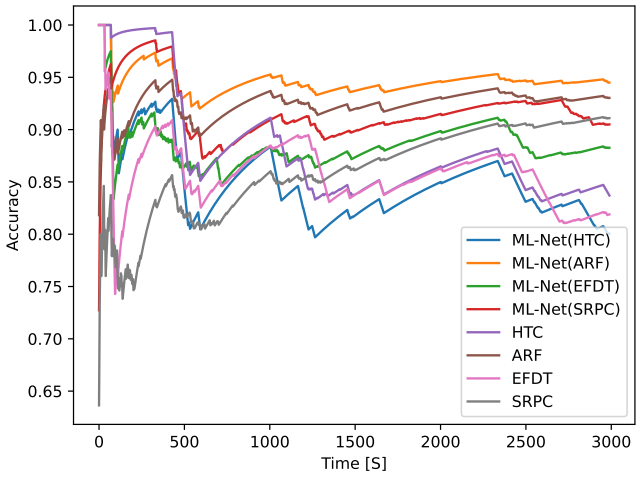

Fig. 12: Accuracy achieved by schemes for comparison vs. *ML-Net*

sions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Army Research Office or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation herein.

References

[1] Miguel L Bote-Lorenzo, Eduardo Gómez-Sánchez, Carlos Mediavilla-Pastor, and Juan I Asensio-Pérez. Online machine learning algorithms to predict link quality in community wireless mesh networks. *Computer Networks*, 132:68–80, 2018.

[2] Gregor Cerar, Halil Yetgin, Mihael Mohorcic, and Carolina Fortuna. Machine learning for wireless link quality estimation: A survey. *IEEE Commun. Surv. Tutorials*, 23(2):696–728, 2021.

[3] Óscar Fontenla-Romero, Bertha Guijarro-Berdiñas, David Martinez-Rego, Beatriz Pérez-Sánchez, and Diego Peteiro-Barral. Online machine learning. In *Efficiency and Scalability Methods for Computational Intellect*, pages 27–54. IGI Global, 2013.

[4] Heitor M Gomes, Albert Bifet, Jesse Read, Jean Paul Barddal, Fabrício Enembreck, Bernhard Pfharinger, Geoff Holmes, and Talel Abdessalem. Adaptive random forests for evolving data stream classification. *Machine Learning*, 106(9):1469–1495, 2017.

[5] Heitor Murilo Gomes, Jesse Read, and Albert Bifet. Streaming random patches for evolving data stream classification. In *2019 IEEE international conference on data mining (ICDM)*, pages 240–249. IEEE, 2019.

[6] Geoff Hulten, Laurie Spencer, and Pedro Domingos. Mining time-changing data streams. In *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 97–106, 2001.

[7] Tao Liu and Alberto E. Cerpa. Temporal adaptive link quality prediction with online learning. *ACM Trans. Sen. Netw.*, 10(3), may 2014.

[8] Christopher J Lowrance and Adrian P Lauf. An active and incremental learning framework for the online prediction of link quality in robot networks. *Engineering Applications of Artificial Intelligence*, 77:197–211, 2019.

[9] Chaitanya Manapragada, Geoffrey I. Webb, and Mahsa Salehi. Extremely fast decision tree. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '18, page 1953–1962, New York, NY, USA, 2018. Association for Computing Machinery.

[10] Dana Marinca and Pascale Minet. On-line learning and prediction of link quality in wireless sensor networks. In *2014 IEEE Global Communications Conference*, pages 1245–1251, 2014.

[11] Jacob Montiel, Jesse Read, Albert Bifet, and Talel Abdessalem. Scikit-multiflow: A multi-output streaming framework. *The Journal of Machine Learning Research*, 19(1):2915–2914, 2018.

[12] Ramya Panthangi M., Mate Boban, Chan Zhou, and Slawomir Stanczak. Online learning framework for v2v link quality prediction. In *2019 IEEE Global Communications Conference (GLOBECOM)*, pages 1–6, 2019.

[13] John David Parsons and Prof J David Parsons. *The mobile radio propagation channel*, volume 2. Wiley New York, 2000.

[14] Charles E Perkins and Elizabeth M Royer. Ad-hoc on-demand distance vector routing. In *Proceedings WMCSA'99. Second IEEE Workshop on Mobile Computing Systems and Applications*, pages 90–100. IEEE, 1999.

[15] Andras Varga. Omnet++. In *Modeling and tools for network simulation*, pages 35–59. Springer, 2010.