Internet of Things 11 (2020) 100203

Contents lists available at ScienceDirect

Internet of Things

journal homepage: www.elsevier.com/locate/iot

An improved PRoPHET - Random forest based optimized multi-copy routing for opportunistic IoT networks

Srinidhi NN, Sagar CS, Deepak Chethan S, Shreyas J, Dilip Kumar SM

Dept. of Computer Science and Engineering, University Visvesvaraya College of Engineering, Bangalore, India

A R T I C L E   I N F O

Article history:
Received 30 September 2019
Revised 5 March 2020
Accepted 17 March 2020
Available online 20 March 2020

Keywords:
Internet of things
Machine learning
Mobility
Multi-copy
Random forest

A B S T R A C T

Opportunistic networks are one of the important categories of ad hoc networks in Internet of Things (IoT), which considers human social activities like daily routines, activities and many more to provide efficient communication. In opportunistic networks, mobile nodes are used to establish communication between nodes despite of non-availability of a dedicated route between them. Furthermore, nodes don't acquire any knowledge in advance about the characteristics of the network such as the network topology and the location of the other nodes. Hence, designing a routing algorithm becomes a challenging task since traditional routing protocols used in the Internet are not feasible for the characteristics inherent type of network. The proposed work propounds a multi-copy routing algorithm based on machine learning named *iPRoPHET* or improved PRoPHET (Probability routing protocol using history of encounters and transitivity). *iPRoPHET*, uses dynamically changing contextual information of nodes and the delivery probability of PRoPHET to carry out message transfer. The *iPRoPHET* uses machine learning classifier known as random forest to classify the node as a reliable forwarder or a non-reliable forwarder based on the supplied contextual information during each routing decision. The classifier trained with large amount of data extracted using simulation leads to precise classification of the nodes as reliable or unreliable nodes for carrying out the routing task. Obtained results from the simulation proves that the proposed algorithm outperforms with respect to delivery probability, hop count, overhead ratio, latency but over costs with respect to average buffer time in par with similar multi-copy routing algorithms. The uniqueness of this paper lies in data extraction, categorization and training the model to obtain reliable and unreliable nodes to facilitate efficient multi-copy routing in IoT communication.

1. Introduction

With the development in technogies, more and more devices are getting connected to the Internet, which leads to the new criterion in networking and communication called the Internet of Things (IoT) [1]. As a result, more devices are getting prevalently monitored and cannot be deployed statically. These IoT devices while being transported by humans or different carriers will interact either by statically or dynamically. For instance, in vehicular based IoT network statically deployed devices may entail to recognize the mobile agents and employ them in assembling and delivering their information to the intended target [2]. In recent times, opportunistic mobile networks rose as a novel mechanism for transmissions in wireless systems. Unlike mobile ad hoc networks (MANETs) [3] that needs end-to-end connected routes for information transfer, the

E-mail address: srinidhinagesh@gmail.com (S. NN).

https://doi.org/10.1016/j.iot.2020.100203
2542-6605/© 2020 Elsevier B.V. All rights reserved.

communication in opportunistic mobile networks takes place on the co-operation of opportunistic connections among mobile nodes without the availability of end-to-end communication paths. These mobile devices establishes connection when people comes in contact and these network type are considered as human social networks. Hence, the opportunistic networks utilize the individual behaviors and social relations to provide more effective and reliable message distribution systems. In OppIoT [4], data acquisition, propagation, and distribution of data which typically necessitate opportunistic connections utilizing mobility, short-range transmission, and technologies. During such operations, routing protocol configuration poses challenge due to its inherent complexity in securing connectivity among devices and recognizing a relevant forwarder to forward data packet to its desired destination. Opportunistic networks [5], which are the derived class of OppIoT and considered as evolved form of MANETs. In OppNets, nodes are allowed to establish communication among several different nodes even though they don't have routing connection. Nodes are connected momentarily and the routes keep dynamically changing due to nodes mobility or frequent nodes activation and deactivation. When the packets are en route between the source node and the target nodes, any viable node in between can be opportunistically employed as the next hop whenever it is plausible to deliver the packets to the intended destination with minimum routing distance. In Opportunistic routing message will be disseminated in two methods as single-copy and multi-copy. In case of single-copy routing delivery probability is less because only one copy of message will exist in the network but intended destination fails to receive message whenever message is dropped, example for single copy routing are First Contact, Direct Transmission/Delivery etc. While in multi-copy multiple copies of same message exist in the network in order to maximize deliver probability. Multi-copy is further categorized into two groups as n-copy and unlimited copy routing protocols [6]. In case of n-copy routing only limited and configurable number of message copies will be maintained in the network, whereas in unlimited copy routing messages will be propagated without any restrictions on the message generation and hence it results in increased packet overhead in the network. PRoPHET, Spray and Wait are the examples for n-copy routing and epidemic, Probabilistic routing are examples for unlimited copy routing protocols [7]. The proposed method uses n-copy routing and three copies of data has been maintained in the network. Several protocols related to OppNets that have been proposed such as epidemic, HBPR, HibOp etc., which uses parameters like message propagation pattern, context erudition of nodes (i.e. current position, the possibility of delivery, target node distance, prior and existing history of the node), last synergy time and various other parameters in order to predict which node is opted as next hop to deliver the message to the intended destination. In this work, a machine learning based algorithm which can be employed to OppNets is proposed. The proposed algorithm that differ radically from the existing algorithm as this algorithm uses the random forest to decide the neighboring node to route the data packet based on the present context and location information. Random forest or otherwise referred to as random decision forest are one of the most popular ensemble learning techniques that are used in both classification and regression. Several decision trees are randomly created with several subsets of data during the training period. The class is assigned based on the mode. Random forest algorithm uses the technique of bootstrap aggregation, which helps in improving the stability and accuracy of the algorithm along with reducing the variance and overfitting of the proposed *iPRoPHET*. Nodes in OppIoT are vulnerable to attack from malicious node which poses as potential IoT node [8]. Authentication architecture [9], which uses smart card to verify nodes genuinity, dummy location privacy-preserving algorithm [10] and preserving the location privacy of mobile device users in location-based services [11], can be considered in the proposed work to help nodes in OppIoT network to hide nodes location and provides dummy data to malicious node in order to prevent it from joining the network.

Rationale of the proposed work are (1) The dynamic nature of nodes in opportunistic network over time poses challenge in order route the information and also to identify the intermediate nodes which are used for forwarding messages to destination. (2) To enhance the efficiency of network parameters of the multi-copy routing algorithm. (3) Usage of machine learning classifier to classify nodes and also to identify potential forwarder nodes. (4) Making use of available past data along with current context information of PRoPHET algorithm to improve classification level.

Contribution of the proposed work are

- Optimizing IoT network parameters like delivery probability, hop count, overhead ratio and latency of the existing *iPRoPHET* algorithm.
- Classifying intermediate forwarder nodes into reliable and unreliable nodes based on supplied contextual information during each routing decision.
- Existing PRoPHET algorithm does not take advantage of the available past data, it only uses the current context information in its static model. Hence *iPRoPHET* algorithm uses both past and current context information to classify forwarder nodes.
- Random forest is used in the proposed algorithm which uses the ensemble learning technique and by constructing several trees and taking an aggregate of them would make the model robust.
- Use of random forest in proposed *iPRoPHET*, provides better classification since it uses boosting and bagging techniques and helps in avoiding overfitting compared to their counterparts and random forest classifier trained with large amount of data extracted using simulation leads to precise classification of the nodes.

The paper has been arranged as follows, Section 2 presents a summary of various existing works with respect to multi-copy and oppurtunisitic networks. Problem statement is presented in Section 3. Proposed algorithm has been presented in Section 4. The experimental evaluation and results has been explained in Section 5. Conclusion along with future work has been furnished in Section 6.

2. Related work

Many protocols that are used in OppNets are related to the nodes context-information and message distribution. Representative ones are summed as follows. Epidemic routing [12], is a flood-based routing algorithm whose foremost objective is to deliver the packet with high probability to the desired destination with a minimum premise about the topology and the network. The work is based on epidemic algorithm which manages two buffers one for the message originating from the node itself and the other buffer is used to store messages generated from different nodes. Each node contains a summary vector that carries a short description of messages currently cached in the buffer. When nodes come in contact of one another, they revise their summary vectors and each node has an extra buffer to prevent verbose connection with other nodes. Nodes exchange their summary vector with each other after a predefined time and after exchanging their vectors the nodes rival their vectors to ascertain which message is missing. Then, the nodes request for the copies of the messages which they do not contain. But the proposed epidemic protocol uses higher bandwidth and buffer space which will cause network congestion and it do not examine nodes parameters to predict the possibility of message delivery. The HBPR (History Based Prediction for Routing), was proposed by Dhurandher et al. [13], uses election of the intermediate node as the message carrier is decided based on specific parameters. These parameters are the duration for which a pair of nodes meet, node's history, time at which a couple of nodes meet, and the path of the mobility of the node concerning the root and the target node. The schema consists of three stages, i.e. identifying the home location, wherein the nodes movement is utilized to recognize the subsequent mode location; the message production stage, during which the intermediate node is determined, and also the home location is renewed and lastly, the subsequent phase for selecting hop, wherein metric is used to judge on the assortment of the best succeeding message forwarder. Protocol has achieved improved results when compared with Epidemic concerning the message delivery number and overhead ratio, however only after accounting the individual mobility paradigm. PRoPHET routing protocol [14], uses non-arbitrary movement and association patterns in practical application situation to replicate the message to different nodes in order to enhance the routeing performance. It is established on the basis that if a node has communicated with a different node or visited a region regularly, then there is a higher likelihood of revisiting the region or the node. In order to accomplish this, a delivery probability is defined for every node which associates itself with other nodes. Let assume that a node T wants to remit a packet to node U if this node T comes in association with another node R which has a higher delivery probability to node U compared to T only then a copy of the packet is copied to the node R. The deliverability is established based on the buffer size aggregate, position, popularity and predictability of delivery. If a source node comes in association concurrently with various nodes the message is copied to the node which has the highest value of deliverability. In distance-based PRoPHET, routing algorithm distance is used as an added factor. Here each node containing the packet checks the distance from all the bordering nodes and picks the nodes with is closed as the packet forwarder. Thereby improving the probability of delivery and lessening the delay probability. Prowait routing algorithm which was proposed by Dhurandher et al. [15], can be considered as an amalgamation of the PRoPHET and Spray and wait routing protocol. Prowait model utilities an uncomplicated forwarding schema, in which when two nodes engage, the transmission of packet happens from the node with lower probability of delivery to the node have higher probability concerning the desired destination. They mainly weigh a couple of parameters during packet transmission 1) election of an adjacent node 2) choosing the node for the next hop. For the election of the node, the PRoPHET protocol is used to estimate delivery predictability and transferring of the data is done using the spray and wait routing protocol. Choosing the node for the next hop is done using the PRoPHET routing algorithm. Prowait routing algorithm keeps the replicate data packets so that the packet is delivered to the intended destination with minimum delay. It reduces the usage of resource wastage by depreciating the amount of flooding compared to the epidemic protocol. However, it doesn't predict forwarding of the message packet in order to improve delivery delay in the network. HiBO protocol (history-based routing protocol for OppNets) was proposed by Boldrini et al. [16], which utilizes the present context statistics about the nodes to ascertain the suitable node to convey the message to the desired destination. The context statistics are collected from the simulation environment. When the node desire to transmit the message, the node utilizes the tables to decide about a union exists among the context statistic of its adjacent nodes and the related message data. In this case, data packets are transmitted to the adjacent node where the union co-exists. This routing algorithm will have higher delay in deciding which is the neighboring node and requires additional effort in maintaining the tables. EDR (encounter and distance based routing algorithm) [17], protocol uses context statistics to convey the message from the source to the destination. Here the routing protocol determines dynamically contact value for every couple of nodes (i.e. the frequency at which a pair of nodes meet). Moreover, the Euclidean distance is also estimated dynamically for every couple of nodes. A parameter called the best-forwarder is evaluated based on the values considered earlier and the packet is transmitted to those intermediate nodes only if their forwarder value is more than the threshold. As it is context related protocol, it does not function well concerning packet delivery probability and overhead ratio. A Markov chain prediction model proposed by Jeon and Lee [18], in which given nodes mobility behavior is outlined on the given nodes former mobility information state to anticipate the nodes expected movement. Using this data, mobile nodes packet delivery ratio to destination node is calculated. The proposed protocol enhances the probability of delivery and reduces the delay in par with conventional routing protocols for OppNets. But, this protocol performs inadequately compared to other mobility models like a random waypoint. MLProph [19], which is an infrastructure-less OppNet routing protocol utilizes a machine learning procedure to decide preceding network information and nodes parameters like the size of the buffer, hop-count, momentum and delivery. Based on these parameters, node likelihood as a forwarder is determined. However, it does not take into account of node's message delivery probability which is prime parameter to be considered during routing.

However, in the proposed model random forest algorithm is considered to decide the neighboring node to route the data packet based on the present context and location information. Several decision trees are randomly created with several subsets of data during the training period, which helps in improving the stability and accuracy of the algorithm along with reducing the variance and overfitting.

3. Problem statement and challenges of the proposed work

In this section challenges and the problem statement of the proposed work is explained.

3.1. Challenges

There are various challenges that opportunistic network in IoT faces. This is due to the dynamic nature of nodes over time depending on the application, which makes it difficult for routing the information and to identify the intermediate nodes which are used for forwarding messages to destination.

- In OppIoT, data acquisition, propagation, and distribution of data which typically necessitate opportunistic connections utilizing mobility, short-range transmission, and technologies. During such operations, routing protocol configuration poses challenge due to its inherent complexity in securing connectivity among devices and recognizing a relevant forwarder to forward data packet to its desired destination.
- Data extraction: For training the machine learning based random forest model, a huge data set has to be obtained. To obtain data that is close to the real world was a challenge. ONE simulator [20], provides the closest solution and to extract the data from the ONE simulator, the source code has to be modified in order to train the model.
- Model Creation: After extracting the data from ONE simulator, a robust random forest classifier had to be constructed. In order to do so, features and algorithm related parameters has to be chosen wisely.
- Model Integration: After the model is constructed, it has to be properly integrated with the ONE simulator in order to classify the nodes.

3.2. Problem statement

The problem statement of the proposed work is as follows.

1. To classify the node as a reliable forwarder or a unreliable forwarder based on the supplied contextual information during each routing decision in order to optimize delivery probability, hop count, overhead ratio and latency of the IoT communication.
2. To train the random forest classifier with large amount of data extracted from the simulator to precise the classification of the nodes as reliable or unreliable node for carrying out the routing information.

4. Proposed algorithm

This section provides in detail explanation of the proposed *iPRoPHET* algorithm.

4.1. Terminology

- S: Source node
- D: Destination node
- X: Selected forwarder node
- G: Set of all neighboring nodes
- N_i: Neighboring node i in the set S
- M_i: Message
- n: Current node, tasked with making a routing decision
- M: Set of all messages that are generated during data generation phase
- M_s: Set of messages that are delivered
- M_d: Set of messages that are dropped or aborted
- E: Set of all the events recorded during data extraction phase of PRoPHET
- $P(N_i)$: Delivery probability as calculated by PRoPHET for the neighbor node Ni using current node n
- $feature\_name(x)$: Value of the attribute *feature_name* in the record x in any set.

4.2. System model

Consider N nodes forming an opportunistic network with a means to communicate and having sufficient energy. Every node has sufficient buffer to store the state information like the nodes movement information, current location, delivery probability, direction, distance between source and destination, past history and power status. All these attributes can be used to determine the route for the message transfer. PRoPHET is a n-copy routing algorithm of multi-copy routing category and uses previous contact history between the nodes to calculate the delivery probability between them. Even though this algorithm provides a sufficient performance, the method to select the next node could be optimized by applying machine learning models. PRoPHET does not completely utilize all the context information that are present in the nodes. With the data driven approach, it is possible to select best predictors/features and use them to drive the routing decisions. In the proposed Random Forest Based Optimized Multi-Copy Routing for Opportunistic IoT Networks, random forest algorithm is used to choose the neighboring node to route the data packet based on the present context and location information. Random forest are one of the most popular ensemble learning techniques that are used for both classification and regression. Several decision trees are randomly constructed with varied subsets of data during the training period. The class is assigned based on the mode. Random forest algorithm uses the technique of bootstrap aggregation, which helps in improving the stability and accuracy of the algorithm along with reducing the variance and overfitting. Formally, consider a training set $X = x_1, \ldots, x_n$ along with their corresponding outputs $Y = y_1, \ldots, y_n$ giving dataset $D_n = (X_1, Y_1), \ldots, (X_n, Y_n)$. Random forest is a collection of random base regression trees $\{rn(x, \theta_m, D_n), m \geq 1\}$, where $\theta_1, \theta_2, \ldots$ are outputs of a randomizing variable θ. The outputs of these random trees are aggregated to get an estimate. Thus, random forest model is given by,

$$\bar{rn}(X, D_n) = E_\theta[rn(X, \theta, D_n)] \tag{1}$$

Where E_θ is the expectation with respect to θ. The dataset is divided into various subspaces based on this randomizing variable theta. It is also used to find out on what basis the successive cuts are done when constructing the individual trees. During the learning period, the algorithm will choose a random subset of features, called as feature bagging. If a given feature is a strong predictor, it is used repeatedly in several different trees. To find the optimal number of trees, cross-validation techniques is applied. The main advantage of Random forest over normal decision trees is that the predictions from single tree are susceptible to noise. But the aggregation of forest will overcome from this drawback. Random forest algorithm can also be viewed as weighted neighborhood schemes. Here random Forests algorithm is be used to classify the neighboring nodes as reliable and non-reliable forwarders based on the contextual information. Our proposed algorithm has mainly two stages: (1) Collecting routing data from ONE simulator (2) Building and training the model.

4.2.1. Stage 1: collecting routing data from one simulator

One simulator [20], is one of the best simulation environment for simulating opportunistic mobile networks. For this stage, modification of the one simulator source code is done to extract the necessary information to train the algorithm. A total of 3 datasets are generated, they are.

1. Event Dataset: This dataset is used to describe every event that took place inside the simulation environment. An event is a discrete piece of entity in a discrete-event based simulation system. In our context an event represented as activity inside the environment such as relaying of the message. The features of this dataset contains the following information in columns and rows.
   (a) *key*: Used to identify the inter related records in various datasets.
   (b) *current_node*: This is the node which contains message related to routing decision.
   (c) *destination_node*: The node to which the message has to be delivered.
   (d) *neighbor_nodes*: The list of all the node ids which are in range to the current node.
   (e) *message_id*: id of the message which is to be forwarded.
   (f) *selected_forwarder*: The node selected as the forwarder for the message in PRoPHET algorithm.
   (g) *predictability_of_the_current_node*: Delivery prediction for the current node as calculated by the original PRoPHET algorithm.
   (h) *predictability_of_the_selected_forwarder*: Delivery prediction for the selected forwarder based on which it was selected as a best forwarder.
2. Message Dataset: This dataset contains rows that describes the details of the message like its source, intended destination, message size and traveled path. An important feature is that it contains *delivery status*, which is used to know whether the message was finally delivered to the intended destination or whether it was aborted in midway. Based on this feature, the dataset was further categorized into three parts:
   (a) Delivered: This dataset contained messages that are ultimately delivered to the final destination.
   (b) Aborted or dropped: Messages that are dropped during routing.
   (c) Relayed: Messages that are transferred to intermediate nodes.
   The features of this dataset contains the following information as columns of the row:
   - *key*: As described earlier.
   - *message_id*: A unique identifier to identify the message.
   - *message_size*: Size of the message in bytes.
   - *source*: Node from which the message is generated.
   - *destination*: Node to which intended message should be delivered.
   - *number_of_hops*: Number of nodes traversed in its path to destination.
   - *message_creation_time*: Time during which message was created in simulated seconds.
   - *message_deletion_time*: Time during which the message was deleted in simulated seconds.

Algorithm 1: Labelling the dataset.

```
Result: Labelled dataset that can be fed into the model
1  Begin ;
2  foreach event e in set E do
3      foreach message M in set M_s do
           /* if key of the records are same and the current_node and selected_forwarder_node are present
              successively in the path of themessage and message_id in event e and message_id in message M are
              same                                                                                       */
4          if message_id[M] == message_id[e] and Key(e) == Key(M) and current_node(e) + selected_forwarder(e) in path[M]
           then
5              delivery_status[e] = “deliver";
6              processed[e] = True;
7      foreach event e in set E do
8          if processed[e] != True then
9              delivery_status[e] = “not_delivered";
10             processed[e] = True
11     foreach event e in set E do
12         remove processed[e];
```

3. Event meta-data dataset: This dataset describes every entity like the source node, the current node, intermediate node, selected forwarder node that were involved in every event in the event dataset.
   (a) Host info: Host info of the node is the set of information about the node at that particular point of event time. This includes nodes x coordinate, y coordinate, buffer size, next time to move, free buffer space, nodes speed which is recorded in 9 columns. The features of this dataset contains the following information in columns of generated dataset for every event recorded.
      - *key*: As described earlier.
      - h_c: Host information of current node.
      - h_d: Host information of destination node.
      - h_n: Host information for all the neighbors.
      - f_{id}: Id of the selected forwarder.
      - *message_id* : As described earlier.

   At the end of this stage, number of datasets containing sufficient number of rows were generated to pre-process the data and to train the algorithm.

4.2.2. Stage 2: building and training the model

After the datasets are extracted to suitable form, pre-processing of data is performed on the event dataset to obtain an extra feature called *delivery status*. *Delivery status* provides an extra feature in the form of label to the proposed algorithm to predict nodes while making routing decisions. The label is a categorical variable with two values as *delivered* and *not delivered*. A label of *delivered* for a row in the event dataset indicates that the message that was relayed as part of the event has delivered successfully and *not delivered* indicates that this event didn’t lead to a successful delivery of the message. The following section, provides a brief about how the labeling is done.

1. Messages in the *delivered* category of message dataset were considered. The path feature contains the list of every node the message traveled in an ordered fashion till it reached the destination. Let’s say that two of the successive nodes in the path for the message M were node n_1 and n_2 with key k. This indicates that message was forwarded from node n_1 to node n_2 in some event that was recorded in the event dataset.
2. Now, using the above information, it is required to find a row in the event dataset that matches the above data. Find a row such that it should satisfy the below conditions.
   - current node $= n_1$
   - selected forwarder $= n_2$
   - message_id $= M$
   - key $= k$

   A row which matches the above conditions are labelled as *delivered* while the rest of the rows are labelled as *not delivered*.

Algorithm 1, is used to create the final datasets from the intermediate datasets that were obtained from the simulation. The final dataset obtained is used to train the random forest classifier to obtain reliable and unreliable nodes.

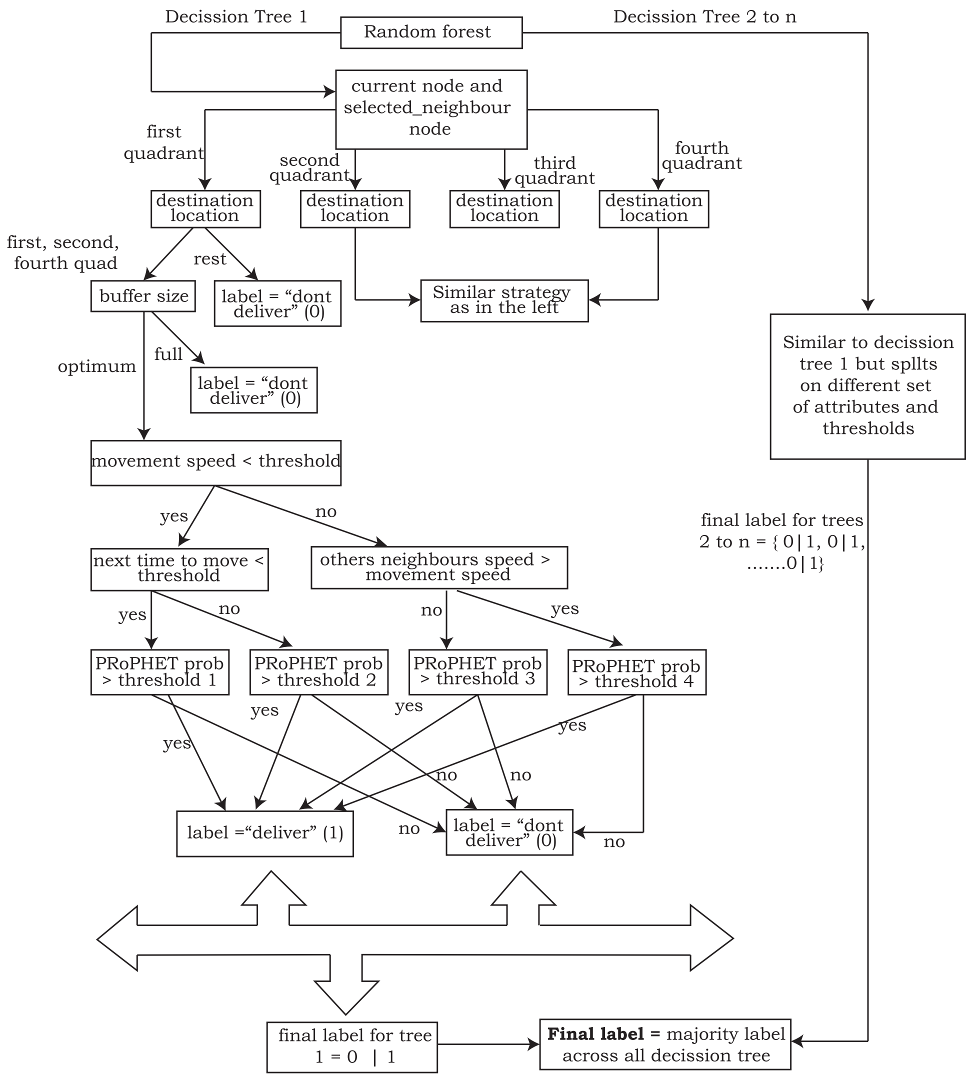
Decission Tree 1
Random forest
Decission Tree 2 to n
current node and selected_neighbour node
first quadrant
second quadrant
third quadrant
fourth quadrant
destination location
destination location
destination location
destination location
first, second, fourth quad
rest
buffer size
label = "dont deliver" (0)
Similar strategy as in the left
Similar to decission tree 1 but splits on different set of attributes and thresholds
optimum
full
label = "dont deliver" (0)
movement speed < threshold
yes
no
next time to move < threshold
others neighbours speed > movement speed
final label for trees 2 to n = {0|1, 0|1, ...0|1}
PRoPHET prob > threshold 1
PRoPHET prob > threshold 2
PRoPHET prob > threshold 3
PRoPHET prob > threshold 4
label ="deliver" (1)
label = "dont deliver" (0)
final label for tree 1 = 0 | 1
Final label = majority label across all decission tree

Fig. 1. Random forest architecture.

4.3. *Usage of random forest in proposed algorithm*

Random Forest is one of the most popular machine learning algorithms for both classification and regression. The existing PRoPHET algorithm does not take advantage of the available past data, it only uses the current context information in its static model. Thus, it could be made more robust by integrating machine learning capabilities to it. Random forest uses the ensemble learning technique. By constructing several trees and taking an aggregate of them would only make the model robust. Apart from this, it also provides better classification since it uses boosting and bagging techniques and helps in avoiding overfitting compared to their counterparts. Random forest is better suited for this particular application because it can work on multidimensional input data and the data used to train the model has a dimension. In an opportunistic network environment, each and every node contains the context and location information of its neighbors. This contextual information is used with the proposed model to make predictions.

Fig. 1 shows the random forest architecture used to segregate reliable and unreliable nodes while making each routing decision. The architecture consists of *n* number of decision trees which are the natural choice of base models/learners in the random forest technique. Bootstrap aggregation, also known as bagging is then used to combine the results of these base models to produce the final results. *Majority vote* is the combining technique used in classification tasks which produces the output/label as the one that has been predicted by most of the base learners. In this figure, the decision tree has been expanded for one among the *n* decision trees used in the random forest architecture. Each base decision tree is trained using random sample of inputs with replacements and different set of features to provide overall effect of combining the results leads to reduced variance in the final output.

Table 1
Simulation parameters.

Parameter	Value
Simulation Area	(4500 × 3400) sq.m
Total number of nodes	198
Total nodes group	6
Pedestrains group	3
Nodes number in each pedestrian group	30
Pedestrian walking speed	0.5-1.5 km/h
Pedestrian Buffer size	15 Mb
Tram groups	3
Nodes number in each tram group	2
Tram speed	6.5 Km/h
Tram Buffer size	59 Mb
Transmission speed of Bluetooth	250 K
Transmission range of Bluetooth	20 m
High speed interface transmission speed	10 Mb
High speed interface transmission range	1500 m
TTL for message in every group	100 min
Message generation intervals	25–35 s
Size of message	500 Kb - 1 Mb
Mobility model	Shortage path map based movement model
Runtime of Simulation	100,000 s

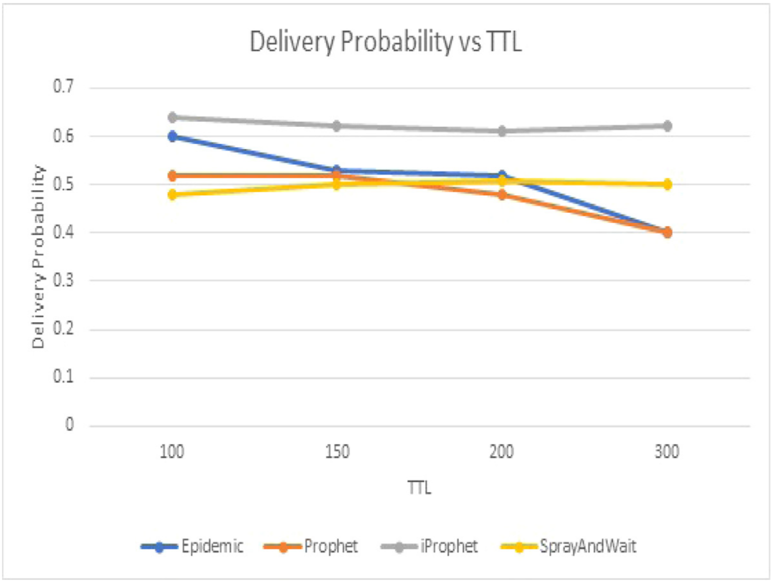

Fig. 2. Comparison of the delivery probability vs TTL.

In the decision tree 1, the first node considers the location co-ordinates of the current node, the selected neighbor node and makes the decision based on the quadrant of *selected neighbor* with respect to *current node. Selected neighbor node* is the one that the current node has established the connection with and the *current node* needs to make decision about the reliability while transferring it to the *neighbor node*. For each quadrant, the next decision node considers the location co-ordinates of the destination to proceed to the next decision node. If the *Selected neighbor node* belong to the first quadrant, then the destinations lying in the first, second, and fourth quadrant are considered as valid, the *Selected neighbor node* could eventually move and therefore allowed to next level of the decision tree. Similarly, for neighbors in other quadrants, i.e. quadrants of the destination nodes surrounding the neighbor quadrants are considered as *valid*. For e.g, for second quadrant, quadrant 1,2,3 are the surrounding quadrants and quadrant 2,3,4 are for third quadrant. If the destination doesn't lie in the surrounding quadrant, then the message with that destination is not transferred to the *Selected neighbor node*. The next decision node considers the buffer size of the *Selected neighbor node* node. The nodes with optimum level of free buffer size in order to accommodate the incoming message are proceeded down the tree, where as the nodes with lesser space will be discarded (i.e. the message is not forwarded). The next decision is based on the criteria of movement speed of the *Selected neighbor node* node. The movement speed is compared against the threshold as calculated by the decision tree and is allowed to proceed along different paths. If the movement speed is less than the threshold, the nodes are not immediately treated as *unreliable* as shown in the decision nodes. Instead, the decision nodes will make comparison of the feature value across various other neighbors feature values to make appropriate decision regarding reliability. Finally, the *Selected neighbor node* PRoPHET values meeting the threshold set by their respective groups is considered for final message transfer. This procedure is repeated for other base learners with different set of features and random sampling of the data.

For a given node, extracted data about one neighbor at a time will be fed into trained model. Based on the features, the model classifies the neighbors as reliable forwarder or non-reliable forwarder. After going through all the neighbors, data packets are forwarded to all of the reliable forwarders as presented in the Algorithm 2 . Since it is an improvement over

Algorithm 2: Improved PRoPHET algorithm.

```
1  Begin ;
   // Original PRoPHET
2  foreach node $N_i$ as i do
3      dp[i] <- random()
4  for every 5 s do
5      foreach node $N_i$ as i do
6          dp[i] < - GetPRoPHETProbabilty(i);
   // End of original PRoPHET
7  foreach message m in current node n buffer do
8      foreach node $N_i$ as n in neighborhood set G do
9          P(n) < - dp[n];
10         if dp[n] $\geq P(N_i)$ then
11             if $N_i == X$ then
12                 update X;
13                 update G;
14                 Label = $RandomForest1(X, G, P(X), P(n))$;
15             if Label == “deliver" then
16                 Relay the message to node $X$;
17             else
18                 if Label == “don’t deliver" then
19                     Don’t relay the message to the node $X$;
20         if dp[n] $< P(N_i)$ then
21             if $N_i == X$ then
22                 update X;
23                 update G;
24                 Label = $RandomForest2(X, G, P(X), P(n))$;
25             if Label == “deliver" then
26                 Relay the message to node $X$;
27             else
28                 if Label == “don’t deliver" then
29                     Don’t relay the message to the node $X$;
```

the PRoPHET algorithm, the Prophet probability is also considered one of the feature for classification. In order to see how the model performed in real world, the source code of the PRoPHET algorithm in ONE simulator was modified to include the trained model.

5. Experimental evaluation

In this section, environment considered for simulating proposed work along with discussion about results obtained is explained.

5.1. Simulation settings

In this section, the evaluation of the proposed *iPRoPHET* is carried out using the Opportunistic Network Environment (ONE) [20] simulator. The simulation is done on a terrain size of 4500 × 3400 sq. m with 51 to 198 nodes deployed randomly. The transmission range of the node is set to 20 m. The different parameters considered during simulation are shown

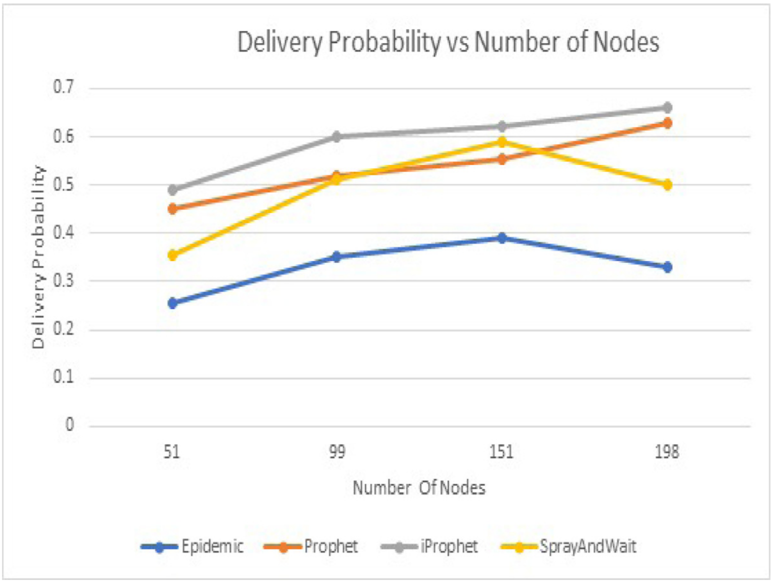

Fig. 3. Comparison of the delivery probability vs no. of nodes.

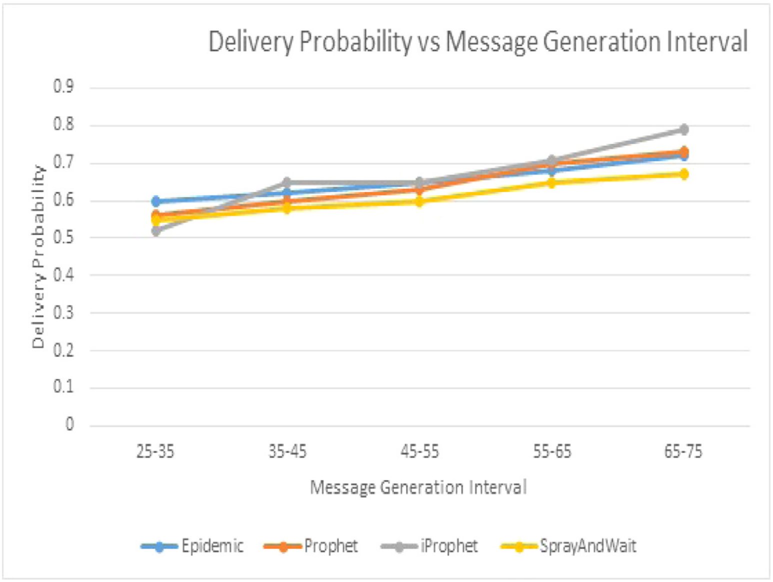

Fig. 4. Comparison of the delivery probability vs message generation interval.

in Table 1. Here the proposed *iPRoPHET* performance has been compared with Epidemic [12], Prophet [14] and Spray and Wait [21] routing protocols, with changing TTL values from 100 to 300 s, number of nodes from 51 to 198, and message generated interval from 25 to 35 s.

5.2. Simulation results and performance analysis

This section, proposed algorithm performance is evaluated with respect to delivery probability, buffer time, hop count, overhead ratio and latency in par with similar existing algorithm.

Figs. 2–4 compares the delivery probability of various algorithms when the TTL, Number of Nodes and Message Generation Interval is varied accordingly. It can be observed that delivery probability of algorithms other than Epidemic and *iPRoPHET* decreases with the increase in TTL because as increase in TTL will directly increase message lifetime and hence the messages remain in the node buffer for longer time than usual which results in more messages getting dropped. However, in *iPRoPHET* algorithm, the delivery probability increases because of the selection of the next forwarder using random forest

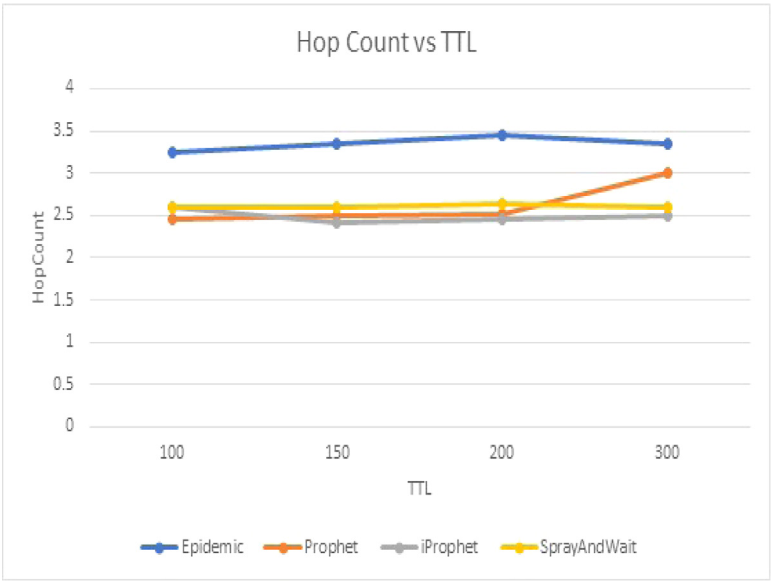

Fig. 5. Comparison of the hop count vs TTL.

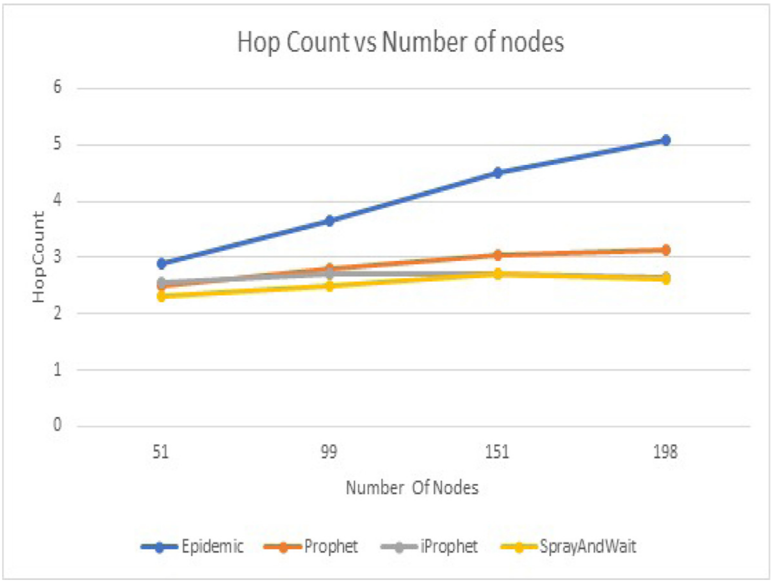

Fig. 6. Comparison of the hop count vs no. of nodes.

classifier which considers PRoPHET probability and other contextual information of the forwarders. The next forwarders are selected not only when their PRoPHET probability is greater but also when the probability is lower which helps to clear the messages from the buffer as quickly as possible. When the number of nodes is increased, the algorithm gets to work with more forwarders enhancing the delivery accuracy of the messages. It can be observed that the *iPRoPHET* algorithm outperforms all other algorithms when it comes to delivery probability as the message generation interval is increased. When message interval is increased, the overhead of messages in the network reduces and the router queues are free to accommodate the messages which reduces message dropping from the nodes buffer queue. This decrease in overhead ratio is depicted in Fig. 11 (Overhead ratio vs Message Generation Interval). This, accompanied with the fit forwarders helps messages to get delivered to their respective destinations with improved probability.

Figs. 5 –7 compares the Hop Count of various algorithms when the TTL, Number of Nodes and Message Generation Interval is varied accordingly. It can be seen that across all variations, *iPRoPHET* has a lesser hop count than PRoPHET algorithm. This is attributed to the fact that the *iPRoPHET* algorithm tries to maximize the delivery probability by even considering forwarders with lesser PRoPHET probability which could possibly have other better contextual attributes. When the number

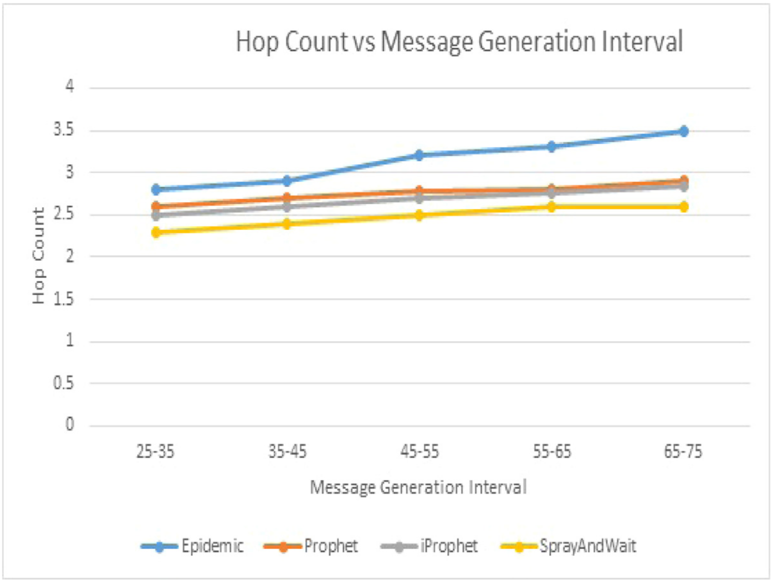

Fig. 7. Comparison of the hop count vs message generation interval.

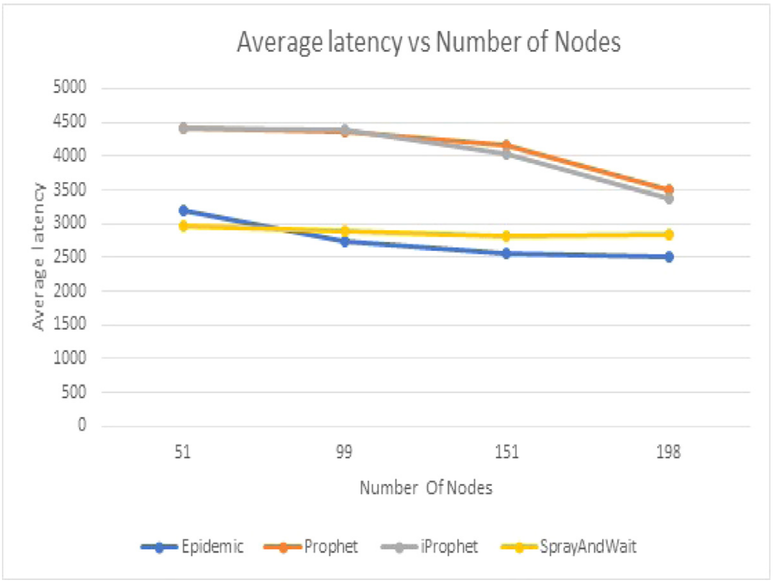

Fig. 8. Comparison of the average latency vs no. of nodes.

of nodes is increased, it leads to corresponding increase in number of forwarders available for routing the messages. Among these forwarders, the random forest classifier picks the fit nodes accurately leading to lesser overall hop count. Similar reason holds good when varied against TTL. When TTL increases, hop count of other algorithms except SprayAndWait increases because of their nature of selection of forwarder nodes which are less likely to forward it to destination. Also, the messages stay in the buffer for longer period of time in PRoPHET since nodes won't forward the messages if the PRoPHET probability is lower for the forwarders. When the message generation interval is varied, the number of messages in the network decreases, but the precision of the classifier remains same. Therefore, the overall decrease in hop count is observed across all the graphs.

Figs. 8–10 compares the average latency of various algorithms when the Number of Nodes, TTL and Message Generation Interval is varied accordingly. *iPRoPHET* algorithm has only slightly lesser average latency time than PRoPHET because of the selection of the nodes with lesser PRoPHET delivery probability which is required to maximize the overall delivery probability but in doing that, the algorithm may select few outliers that might keep the messages for far longer time than usual and as number of nodes, the outlier count increases which increases the message delivery latency. However, this is

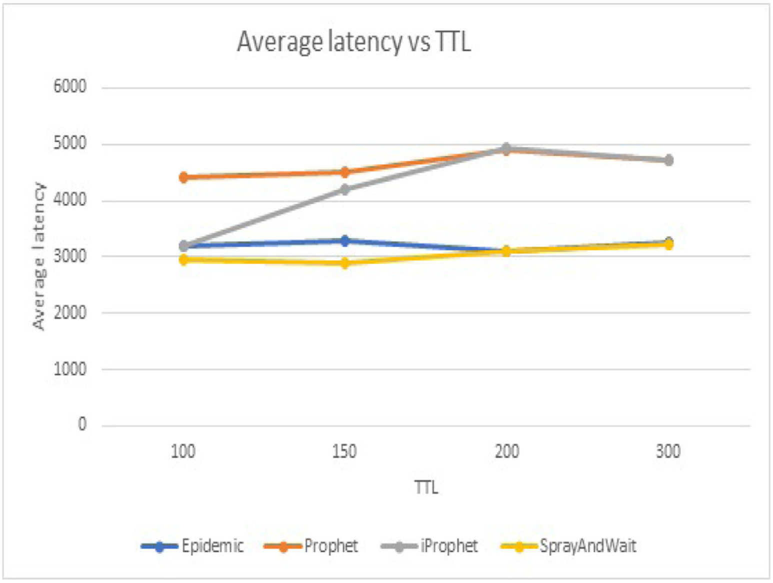

Fig. 9. Comparison of the average latency vs TTL.

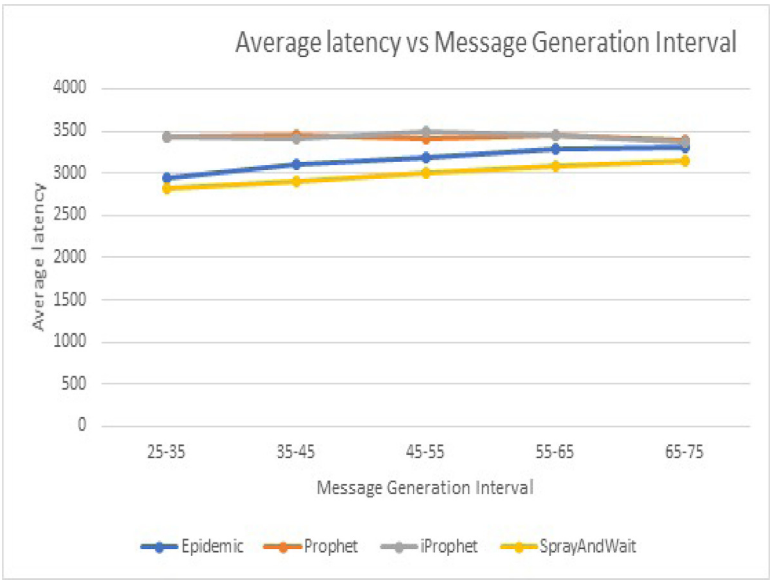

Fig. 10. Comparison of the Average latency vs message generation interval.

compensated by the lesser hop count delivery of the non-outlier nodes that help to keep the latency times in control for majority of the messages. Similar impact of outliers can be seen when the TTL and message generation interval is varied. But the majority of non-outlier nodes overcome such impact with their timely delivery of the messages to their destinations.

Fig. 11 compares the overhead ratio of various algorithms for varying message generation interval from 25 to 35 to 65–75. It is observed that the overhead ratio is better in *iPRoPHET* because of the increased delivery probability resulting from better forwarders chosen as part of random forest classifier.

Figs. 12 and 13 compares the average buffer time of various algorithms when the number of nodes and TTL is varied. *iPRoPHET* algorithm has slightly greater average buffer time because of the selection of the nodes with lesser PRoPHET delivery probability which is required to maximize the overall delivery probability but in doing that, the algorithm may select few outliers that might keep the messages for far longer time than usual and as number of nodes increases, the outlier count increases keeping the average buffer time slightly larger and outlier prone. When the TTL is increased, the similar impact of outliers can be observed. The messages stay in the outlier nodes buffer for longer time than usual without getting forwarded and therefore results in increased average buffer time.

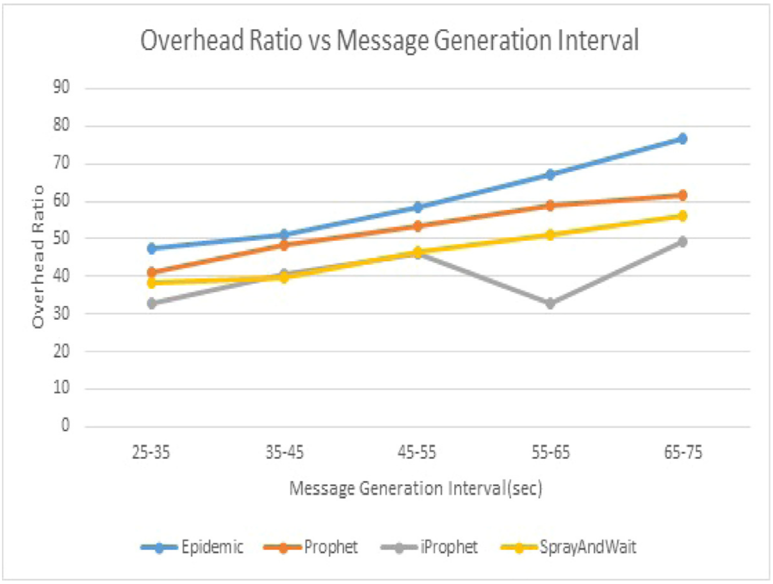

Fig. 11. Comparison of the overhead ratio vs message generation interval.

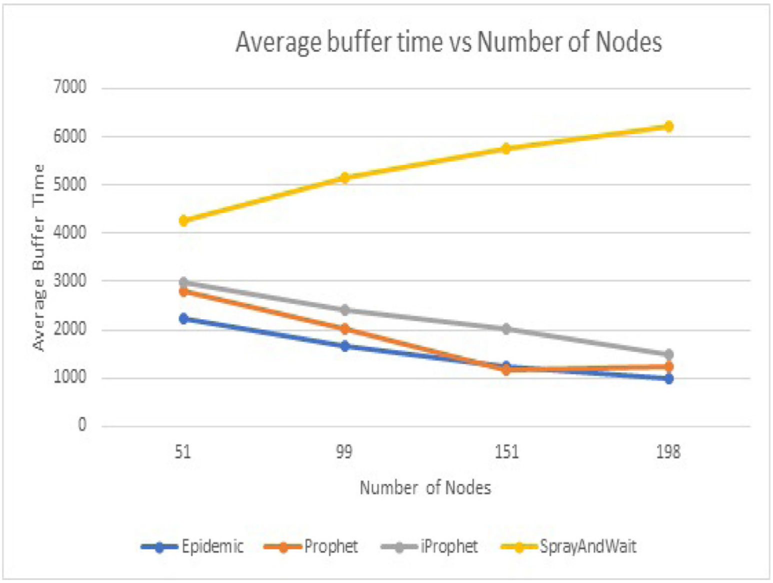

Fig. 12. Comparison of the average buffer time vs no. of nodes.

6. Conclusions and future work

Advent of opportunistic mobile networks as a novel mechanism for transmissions in wireless systems, the communication in opportunistic mobile networks takes place on the corporation of opportunistic connections among mobile nodes without the availability of end-to-end communication paths. But, in Opportunistic IoT, data acquisition, propagation, and distribution of data which typically necessitate opportunistic connections utilizing mobility, short-range transmission, and technologies. During such operations, routing protocol configuration poses challenge due to its inherent complexity in securing connectivity among devices and recognizing a relevant forwarder to forward data packet to its desired destination. Hence, in this paper multi-copy routing algorithm for opportunistic networks named *iPRoPHET* or improved PRoPHET is proposed. This method makes use of machine learning classifier called random forest and delivery probability of existing PRoPHET to establish routing of messages. With the simulation results it can be concluded that: (i) While making each routing decision, random forest classifies the nodes as reliable or unreliable which led to better selection of intermediate nodes to carry out efficient routing. (ii) Proposed method outperforms in terms of delivery probability, hop count, overhead ratio and latency in

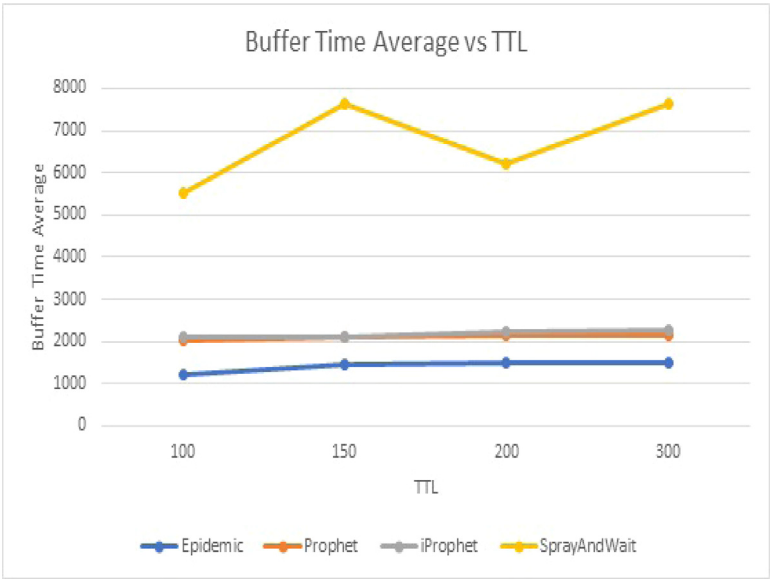

Fig. 13. Comparison of the buffer time average vs ttl.

comparison with similar existing algorithms. (iii) The classifier trained with large amount of data extracted using simulation leads to precise classification of the nodes as reliable or unreliable nodes to provide efficient communication. However the proposed method will have poor performance in terms of average buffer time in par with similar multi-copy routing algorithm. The average buffer time has experienced a performance issue due to the presence of the outlier nodes, which needs to be addressed as part of future enhancement. Furthermore, energy consumption of the nodes needs to be optimized for working with compute-intensive machine learning algorithms in the future work.

Declaration of interests

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgment

This research work has been funded by the Science and Engineering Research Board (SERB-DST) Project File No: EEQ/2017/000681. Authors sincerely thank SERB-DST for intellectual generosity and research support provided.

References

[1] N. Srinidhi, S.D. Kumar, K. Venugopal, Network optimizations in the internet of things: areview, Eng. Sci. Technol.Int. J. 22 (1) (2019) 1–21.
[2] N. Srinidhi, J. Lakshmi, S.D. Kumar, Hybrid energy efficient and QoS aware algorithm to prolong IoT network lifetime, in: International Conference on Ubiquitous Communications and Network Computing, Springer, 2019, pp. 80–95.
[3] S. Dilip Kumar, B. Vijaya Kumar, Energy-aware multicast routing in MANETs: a genetic algorithm approach, International Journal of Computer Science and Information Security (IJCSIS) 2 (1) (2009) 42–47.
[4] B. Guo, D. Zhang, Z. Wang, X.Z. Z. Yu, Opportunistic IoT: exploring the harmonious interaction between human and the internet of things, J. Netw. Comput. Appl. (2013) 1531–1539.
[5] L. Pelusi, A. Passarella, M. Conti, Opportunistic networking: data forwarding in disconnected mobile ad hoc networks, IEEE Commun. Mag. 44 (11) (Nov. 2006) 134–141.
[6] E. Spaho, L. Barolli, V. Kolici, A. Lala, Evaluation of single-copy and multiple-copy routing protocols in a realistic VDTN scenario, in: 2016 10th International Conference on Complex, Intelligent, and Software Intensive Systems (CISIS), IEEE, 2016, pp. 284–289.
[7] A. Keranen, J. Ott, T. Karkkainen, The ONE simulator for DTN protocol evaluation, in: Proceedings of the 2nd International Conference on Simulation Tools and Techniques, ICST (Institute for Computer Sciences, Social-Informatics and Q, 2009, p. 55.
[8] A.S. Sohal, R. Sandhu, S.K. Sood, V. Chang, A cybersecurity framework to identify malicious edge device in fog computing and cloud-of-things environments, Comput. Secur. 74 (2018) 340–354.
[9] R. Amin, N. Kumar, G. Biswas, R. Iqbal, V. Chang, A light weight authentication protocol for IoT-enabled devices in distributed cloud computing environment, Future Gener. Comput. Syst. 78 (2018) 1005–1019.
[10] G. Sun, V. Chang, M. Ramachandran, Z. Sun, G. Li, H. Yu, D. Liao, Efficient location privacy algorithm for internet of things (IoT) services and applications, J. Netw. Comput. Appl. 89 (2017) 3–13.
[11] G. Sun, S. Cai, H. Yu, S. Maharjan, V. Chang, X. Du, M. Guizani, Location privacy preservation for mobile users in location-based services, IEEE Access 7 (2019) 87425–87438.
[12] A.Vahdat, D.Becker, Epidemic Routing for Partially Connected ad hoc networks, TechnicalReportCS-2000-06, 2000.
[13] S.K. Dhurandher, D.K. Sharma, I. Woungang, S. Bhati, HBPR: history based prediction for routing in infrastructure-less opportunistic networks, IEEE AINA 2013, Barcelona, Spain,March 25–28,, 2013.

[14] A. Lindgren, A. Doria, O. Schelen, Probabilistic routing in intermittently connected networks, ACM SIGMOBILE Mob. Comput. Commun.Rev. 7 (3) (July 2003) 19–20.
[15] S.K. Dhurandher, S.J. Borah, M.S. Obaidat, D.K. Sharma, B.B. S. Gupta, Probability-based controlled flooding in opportunistic networks, in: Proc. of Intl. Conference on Wireless Information Networks and System (WINSYS), Colmar, France, July 20–22,, 2015, pp. 3–8.
[16] C. Boldrini, M. Conti, I. Iacopini, A. Passarella, HiBOp: a history based routing protocol for opportunistic networks, in: Proc. of IEEE Intl. Symposium on World of Wireless, Mobile and Multimedia Networks (WOWMOM 2007), 2007, pp. 1–12.
[17] S.K. Dhurandher, S. Borah, I. Woungang, D.K. Sharma, K. Arora, D. Agarwal, EDR: an encounter and distance based routing protocol for opportunistic networks, in: Proc. of IEEE AINA 2016, Crans-Montana, Switzerland,March 23–25,, 2016, pp. 297–302.
[18] I. k Jeon, K. w. Lee, A dynamic Markov chain prediction model for delay-tolerant networks, Int. J. Distrib. Sens. Netw. 12 (Sept. 2016) 2–7.
[19] D.K. Sharma, S.K. Dhurandher, I. Woungang, R.K. Srivastava, A. Mohananey, J.J. Rodrigues, A machine learning based protocol for efficient routing in opportunistic networks, IEEE Systems Journal (2016). doi:10.1109/JSYST.2016.2630923
[20] A. Keranen, Opportunistic network environment simulator, Special Assignment report, Helsinki University of Technology, Department of Communications and Networking, 2008.
[21] T. Spyropoulos, K. Psounis, C.S. Raghavendra, Spray and wait: an efficient routing scheme for intermittently connected mobile networks, in: Proceedings of the 2005 ACM SIGCOMM Workshop on Delay-Tolerant Networking, ACM, 2005, pp. 252–259.