DEIM Forum 2017 H2-4

モバイルアドホックネットワークにおける機械学習を用いた攻撃端末特定手法

Boqi GAO[†], Takuya MAEKAWA[†], Daichi AMAGATA[†], and Takahiro HARA[†]

[†] Osaka University, Graduate School of Information Science and Technology
1–5, Yamadaoka, Suita, Osaka 565–0871 Japan
E-mail: †{gao.boqi,maekawa,amagata.daichi,hara}@ist.osaka-u.ac.jp

Abstract モバイルアドホックネットワーク（MANETs）は，既存のインフラを利用せず，無線モバイル端末により自律的に構築されることから，災害などの緊急事態における利用形態として注目されている．しかし，端末が自律的に構築する利点は，攻撃端末の参入を許してしまうという欠点もあり，セキュリティ面の課題に直面する．MANETs では，これまでに多くの攻撃モデルが考えられてきたが，既存研究は一つのモデルにのみ着目し，複数の攻撃モデルが同時に存在する環境については考えられていない．本稿では，複数の攻撃モデルが同時に存在する環境においても攻撃端末を特定することを目的とし，機械学習に基づく手法を提案する．提案手法では，ルーティングプロトコルとして AODV を使用することを想定し，パケットの送受信率などから通常端末と攻撃端末を識別する分類機を作成する．シミュレーション実験の結果から，提案手法はどのような攻撃モデルが存在していても高精度で攻撃端末を特定できることを確認した．

Key words MANETs, security, machine learning, classifier

1. Introduction

Mobile ad-hoc networks (MANETs) have been receiving much research interests [7], due to their self-organizing and infrastructure-less characteristics. In MANETs, specifically, nodes move freely and use multi-hop messages to communicate with each other. Since this kind of infrastructure-less networks do not require base stations to transmit messages, they are applicable in many fields, e.g., emergency correspondence and rescuing work. Due to that characteristic, however, MANETs are vulnerable to malicious nodes. If malicious nodes join a MANET, or if some normal nodes are hacked by adversaries, the MANET suffers from their attacks. That is, normal nodes encounter cases that communications result in failures.

So far, many attack models, such as black hole attack [5], rushing attack [10], and sybil attack [1], have been considered. It is important to identify malicious nodes which execute the above attacks, for secure communications. Existing literatures assume that there is only a single attack model in a MANET. This is obviously unrealistic, because some malicious nodes do an attack, and some other malicious nodes do different attacks. We thus address a challenging problem of identifying malicious nodes in MANETs where multiple attack models exist.

A promising solution is to build a classifier and apply it on each node. Nodes can utilize the classifier to classify their neighbor nodes to normal and malicious nodes. Note that network parameters, such as number of nodes, mobile velocity, and network size, are unknown, because of self-organizing characteristic. We therefore have to design a robust classifier to network parameters, which is also challenging.

In this paper, we propose a method which builds a robust classifier to environments of multiple attacks and networks with different parameters. To summarize, the contributions of this paper are as follows.

（1） We classify famous attacks in MANETs into four categories. We select and test useful features based on the classification.

（2） We propose a method that builds a robust classifier to classify neighbor nodes based on their behaviors.

（3） We propose a weighting mechanism of ensemble learning for identifying malicious nodes.

（4） Our simulation result shows that our method can identify malicious nodes with high accuracy.

The organization of this paper is as follows. In Section 2, we introduce the preliminary of our work including assumption, attack models, and related work. In Section 3, we present our proposed method that builds a robust classifier. We present our result of simulation experiments in Section 4. Finally, in Section 5, we conclude this paper and introduce future work.

2. Preliminary

2.1 Assumptions

The environment is assumed to be a mobile ad-hoc network (MANET) constructed by wireless mobile nodes. These mobile nodes move freely and their communication range is the same. As a routing protocol, AODV [16] is employed. AODV is an on-demand routing protocol. AODV determines a route to a destination only when a node wants to send a packet to the destination, using four message types: route request (RReq), route reply (RRep), route error (Rerr), and hello message. In AODV, if a source node wants to send a data packet to a destination node and there is no existing route in its routing table, the source node broadcasts an RReq. Its neighbour node which receives the RReq checks whether it is the destination node. If so, it sends an RRep to the originator of the RReq. If not, it broadcasts the RReq. Other nodes which receive the RReq do the same procedure until the destination node receives the RReq. Nodes monitor the link status of next hops in active routes. When a node detects a link failure in an active route, it sends an Rerr to notify other nodes of the link failure. Moreover, each node checks whether it has sent a message or not within an interval called hello interval. If it has not, it broadcasts a hello message to notify other nodes of its existence.

Besides, there could be some malicious nodes in the MANET. That is, some adversaries may join the MANET. Their objective is to interrupt communications in the MANET by executing attacks introduced in Section 2.2. We assume that malicious nodes can change their attack models.

2.2 Attack models

We introduce attacks executed by malicious nodes. The following attacks are famous attacks in existing researches about MANETs security.

Black hole attack [5]: In black hole attacks, a malicious node acts like a black hole in universe. By sending fake RReps to all RReqs, the malicious node pretends to have a route to a given destination. This makes its neighbor nodes send packets to it. After receiving data packets, the malicious node drops them.

Grey hole attack [17]: The grey hole attack is just like black hole attack. The difference between black hole attack and grey hole attack is that in grey hole attack, the malicious node drops data packets with a certain probability.

Sybil attack [1]: In sybil attack, a malicious node performs like some other nodes. Malicious nodes send RReps while pretending specific normal nodes. In such a situation, nodes around the malicious nodes consider the malicious nodes as normal nodes. Packets are consequently dragged to the malicious nodes, which disturbs packet transmissions.

Routing packet dropping attack: In routing packet dropping attack, a malicious node randomly drops routing messages. For example, in AODV, a malicious node randomly drops RReqs, RReps, and Rerrs.

Rushing attack [10]: Rushing attack is mainly against on-demand routing protocols. When a malicious node receives an RReq from its neighbor node, it broadcasts the RReq quickly over the network before other nodes do. They thus select it as a target for RRep.

Wormhole attack [9]: In wormhole attack, a malicious node stores all received messages, and send them to another *faraway* malicious node in their own channel, for example, a wired link or an out of band hidden channel. The other malicious node which receives these messages broadcasts them locally. This attack creates an illusion that the neighbor nodes of these two malicious nodes are very close to each other.

Jelly fish delay attack [15]: In jelly fish delay attack, malicious nodes hold received messages for a while before transmission.

Flooding attack [12]: Flooding attack is an attack for energy consumption. This attack has many patterns. One famous pattern is called RReq flooding attack. In RReq flooding attack, a malicious node broadcasts an RReq without destination. Because all nodes transmit this message, redundant traffic incurs.

2.3 Related work

Classification is an effective way to detect malicious nodes [19]. It has been extensively used for intrusion detection in wired networks. However, few works investigated the capability of classification approach for malicious node detection in MANETs.

Literature [13] conducted some experiments that compare the performances of five well known classifiers: Naive Bayes, MLP, Linea, GMM and SVMs. However, [13] considers only four attacks, and does not investigate the robustness of classifiers to network parameters, such as area size and number of nodes. In [14], Nadeem and Howarth presented an intrusion detection and adaptive response mechanism for MANETs. This detects a range of single attacks and provides an effective response to attacks. However, it cannot detect malicious nodes which execute multiple attacks at the same time.

Literature [2] proposed a machine learning based reputation system for MANETs, and considered digital signature based mechanisms that do not require trusted third parties. A new technique, called dynamic thresholds was also proposed, to improve classification accuracies. This work, however, did not test the system on the environment of multiple attacks and did not investigate the scalability to network parameters.

3. Proposed method

In this section, we present our method for building a robust classifier. We build weak classifiers and calculate a weight for each weak classifier. Finally, classification result is obtained from an ensemble classifier which is created by weights and all the weak classifiers.

3.1 Building weak classifiers

To build weak classifiers, we consider useful features, and the features are obtained by simulations. Features are key elements in classification approaches, and selecting features is important to build a robust classifier. To consider and select such useful features, we first classify the attack models to some categories.

3.1.1 Getting features

Classification of attacks. As discussed before, a large number of attacks have been considered in MANETs, and lots of studies classify attacks based on layers or routing protocols [3]. However, these classifications do not help building a classifier because they do not classify the attacks based on their characteristics. We therefore classify the attacks based on their objectives, namely (i) causing link failure (CLF), (ii) dropping data packets (DDP), (iii) dragging routes to attackers (DRA), and (iv) causing redundant traffic (DRT).

The result of our classification of attacks is summarized in Table 1. Note that some attacks, such as black hole and sybil attacks, are categorized into multiple classes. For example, in black hole attack, malicious nodes send RReps and drop data packets. This attack has two objectives: packet dropping and route dragging.

Table 1 Classification of attacks

Class	Attacks
CLF	Routing packet dropping, sybil attack
DDP	Black hole attack, grey hole attack
DRA	Sybil attack, rushing attack, wormhole attack, black hole attack
DRT	RReq flooding attack, hello flooding attack

In our work, a classifier is set on each node. That is, each node works as an observing node and classifies its neighbors. Before we introduce the features we use, we introduce the information that each node observes in Table 2. This information is used to create the features. Recall that in AODV, messages are transmitted by broadcast. Observing nodes thus can overhear the messages.

In our method, we employ two kinds of features: behavior features for classifying neighbors and environment features for estimating network environments.

Behavior features. In Table 3, we define 14 features as behavior features. These features are designed from the classification of attacks. It can be seen that we use *ratios* calculated from the information that observing nodes hold. The reason why we select ratios as features is that ratios are not influenced by network operation time. For example, a network operated for 100 seconds would have different total packet transmission numbers to a network operated for 300 seconds. That is, using number of packets is not suitable for different network situations. The relationship between behavior features and the classifications are illustrated below.

RRep sent ratio (RepSenRatio), RRep ignored ratio (RepIgnRatio), Rerr sent ratio (RerSenRatio), and Rerr received ratio (RerRecRatio) are designed for CLF. A link failure is caused by network topology change and ignoring routing messages such as RReps. If routing messages are ignored by malicious nodes, normal nodes cannot update their rout-

Table 2 Definition of information that observing node obtains

Information	Definition
NReqRec	# RReq overheard from one neighbor
NRepRec	# RRep overheard from one neighbor
NRepSen	# RRep sent to one neighbor
NRerRec	# Rerr overheard from one neighbor
NRerSen	# Rerr sent to one neighbor
NHelRec	# hello message overheard from one neighbor
NDatRec	# data packets overheard from one neighbor
NDatSen	# data packets sent to one neighbor
TReqRec	total # RReq observing node overhears
TReqSen	total # RReq sent by observing node
TRepRec	total # RRep observing node overhears
TRepSen	total # RRep sent by observing node
TRerRec	total # Rerr observing node overhears
TRerSen	total # Re-rr sent by observing node
THelRec	total # hello message observing node overhears
THelSen	total # hello message sent by observing node
TDatRec	total # data packets observing node overhears
TDatSen	total # data packets sent by observing node

Table 3 Behavior features

Behavior features	Definition
RepSenRatio	NRepRec / TRepRec
RepRecRatio	NRepRec / TReqRec
RepIgnRatio	NRepRec / NRepSen
ReqRecRatio	NReqRec / TReqRec
DatSenRatio	NDatSen / TDatSen
DatRecRatio	NDatRec / TDatRec
DatIgnRatio	NDatRec / TDatSen
RerRecRatio	NRerRec / TRerRec
HelRecRatio	NHelRec / THelRec
AllPckRatio	(NDatSen+NRepSen) / (NDatRec+NRepRec)
RepUslRatio	NDatRec / NRepSen
RepReqRatio	NRepRec / TReqSen
RerSenRatio	NRerSen / TRerSen
HelCheckRatio	NHelRec / THelSen

ing tables. Normal nodes thereby cannot transmit messages at the latest network topology. To deal with this problem, ratios about the received and sent routing messages are designed to check whether neighbour nodes drop routing messages or not.

Data received ratio (DatRecRatio), Data ignored ratio (DatIgnRatio), and RRep useless ratio (RepUslRatio) are designed for DDP. We can know situations that neighbour nodes transmit data packets properly, from the ratios calculated from the number of overheard data packets. For example, if the DatIgnRatio of one neighbor node is much lower than those of other neighbour nodes, it is reasonable to consider that this neighbor node ignores some data packets.

Data packet sent ratio (DatSenRatio), RRep checked by RReq ratio (RepReqRatio), RRep received ratio (RepRecRatio), and all routing packets ratio (AllPckRatio) are designed for DRA. The reason is that if one malicious node wants to drag routes to itself, it will send more RReps and receive many data packets.

RReq received ratio (ReqRecRatio), Hello check ratio (HelCheckRatio), and Hello message received ratio (HelRecRatio) are designed for DRT. Because malicious nodes, which execute flooding attacks, send extra hello messages and/or RReqs to incur redundant traffic, the features are useful to identify such malicious nodes.

Environment features. We next define environment features in Table 4. We define a fixed interval time as a window time. In a nutshell, each observing node records the numbers of sent and received packets in a window time. Let x_i be a number that each observing node counts in a window time. Assume that we have ω window times, then each node obtains an environment feature, i.e., $\frac{\sum x_i}{\omega}$. All environment features are created by this approach, and the environment features are utilized to estimate the network environments.

Table 4 Summary of environment features

Env. features	Definition
AvgReqRec	Avg. # RReq observing node overhears
Evergreen	Avg. # RReq sent by observing node
AvgRepRec	Avg. # RRep observing node overhears
AvgRepSen	Avg. # RRep sent by observing node
AvgDatRec	Avg. # data packets observing node overhears
AvgDatSen	Avg. # data packets sent by observing node
AvgRerRec	Avg. # Rerr observing node overhears
AvgRerSen	Avg. # Rerr sent by observing node
AvgHelRec	Avg. # hello messages observing node overhears
AvgHelSen	Avg. # hello messages sent by observing node
AvgNeiMet	Avg. # neighbor nodes

3.1.2 Under-sampling

The number of malicious nodes is generally less than the number of normal nodes. Therefore, normal nodes typically observe the behaviors of other normal nodes. This means that an observing node obtains less information on malicious nodes compared with that on normal nodes. Such imbalanced information (data) influences the performances of classification algorithms [8]. We hence need to alleviate the influence of imbalanced data.

There are two ways to deal with imbalanced data: oversampling and under-sampling. Under-sampling is used in our method because sufficient data can be obtained from simulations. Information on normal nodes would be similar, meaning that important information will not lose even if some normal nodes' information is cut off. We use spread subsample [4] as the under-sampling method. Spread subsample is a method to generate a random subsample of a dataset. The ratio of instances (vectors consisting of behavior and environment features) obtained from normal nodes and malicious nodes, which are used for training, is controlled to 1:1.

3.1.3 Classifier building

In our approach, to build a weak classifier, we employ random forest [11]. Random forest is an improvement method over bagged decision trees. Random forest can avoid over fitting even without pruning because it employs the column and instance random sampling method. Random forest can deal with a large number of instances, hence is useful to create a highly accurate classifier. Recall that we use behavior and environment features. Behavior features are ratios while environment features are numbers. Decision trees are useful to deal with both values.

Using random forest has three advantages, (i) fast training time, (ii) capability of dealing with a huge number of instances, (iii) and accelerating efficiency, due to a large number of decision trees.

3.2 Calculation of weight

It is intuitively known that similar network settings result in similar mean of environment features, and different network settings result in different mean of environment features. From contraposition, similar mean of environment features result in similar network settings. Also, similar network settings result in similar classifier models. That is, if two given networks have similar mean of environment features, the classifiers of these two networks are similar.

To validate this, we conduct a preliminary experiment. This experiment investigates the influence of different network environments on environment features and rank of behavior features. We employ four network parameters: max speed of nodes v_{max}, area size, ratio of malicious nodes, and number of nodes. As an example, we show the influence of v_{max}, on environment features. Area size, ratio of malicious

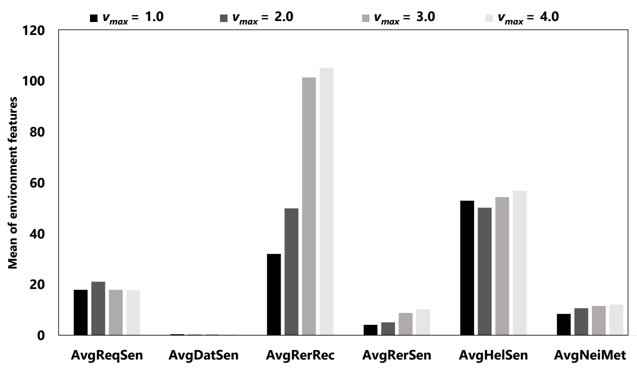

Figure 1 Different v_{max} to environment features (1)

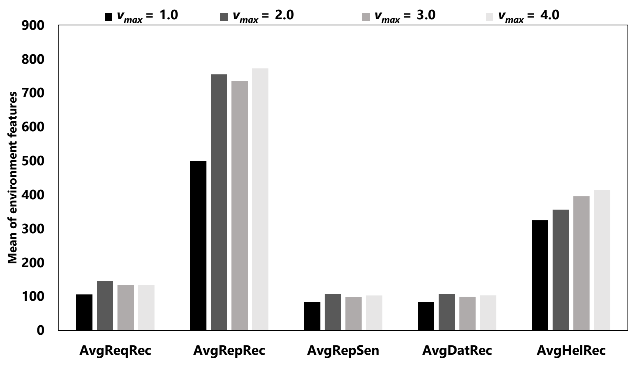

Figure 2 Different v_{max} to environment features (2)

nodes, and number of nodes are respectively set as 700 [m] × 700 [m], 0.2, and 70. We vary v_{max} from 1.0 [m/s] to 4.0 [m/s].

Figures 1 and 2 show the difference of environment features with different v_{max}. From these figures, we can see that as v_{max} increases, AvgRerRec, AvgRerSen, AvgNeiMet, and AvgHelRec increase. As the nodes in the network move faster, network topology changes more frequently and link failures happen more frequently. Consequently, nodes need to send more Rerrs, and meet more neighbors in a window time. They hence need to broadcast more hello messages.

Table 5 illustrates the influence of v_{max} on rank of behavior features. This table shows that AllPckRatio becomes more important as v_{max} becomes larger. AllPckRatio is calculated by (NDatSen + NRepSen) / (NDatRec + NRepRec). As v_{max} becomes larger, network topology changes more frequently. Malicious nodes thus influence larger part of the MANET. AllPckRatio hence increases.

From this investigation, we can see that environment features and ranks of behavior features are influenced by network environments. In the creation of classifiers, ranks of features play an important role. It is hence essential to build

Table 5 Rank of behavior features with different v_{max}

R.	$v_{max} = 1.0$	$v_{max} = 2.0$	$v_{max} = 3.0$	$v_{max} = 4.0$
1	RepRecRatio	RepRecRatio	RepReqRatio	RepSenRatio
2	RepSenRatio	RepReqRatio	RepSenRatio	AllPckRatio
3	RepReqRatio	RepSenRatio	AllPckRatio	RepUslRatio
4	RepUslRatio	HelCheckRatio	RepIgnRatio	RepReqRatio
5	DatRecRatio	RepIgnRatio	RepRecRatio	DatRecRatio

classifiers to fit network parameters. Calculation of weight helps finding similar network settings of the weak classifiers. The principal weight calculation is that the more similar the environment features of a given instance and mean of environment features of a training data are, the larger the weight is.

Given the environment features of a test instance t, we describe these environment features as a vector V_t. Also, environment features of a training data set is described as a vector V_p. For each V_p, we calculate the Euclidean distance, $dist$, between V_t and V_p. During this procedure, the maximum and the minimum among all the Euclidean distances are recorded as $dist_{max}$ and $dist_{min}$. Finally, $w[i]$, which is a weight for weak classifier i, is obtained by:

$$w[i] = 1 - \frac{dist - dist_{min}}{dist_{max} - dist_{min}} \quad (1)$$

Note that small Euclidean distance has a large weight.

3. 3 Weighting based classification

After we calculate weights, we use weights and weak classifiers to obtain final classification results. The result of each weak classifier is always probability. In our case, for example, class of classification is either normal, or malicious. In the case of 0.4 as normal and 0.6 as malicious, the classification result is malicious. We obtain such probabilities from the weighting based classification method.

Let w be a weight vector whose element is the weight of a weak classifier. Also, let p_i^n and p_i^m be the probabilities (classification result) obtained by a weak classifier i. p_i^n is probability of normal and p_i^m is probability of malicious. We define the total points of malicious and normal as mal_{sum} and nor_{sum}, which are calculated by:

$$mal_{sum} = \sum w[i] \cdot p_i^m \quad (2)$$

$$nor_{sum} = \sum w[i] \cdot p_i^n \quad (3)$$

If $mal_{sum} \geqq nor_{sum}$, the result is malicious. Otherwise, it is normal.

4. Experiment

In this section, we conduct experiments of our proposed method.

4. 1 Setting

We used network simulator Qualnet [18] to obtain the features described in Section 3. 1. 1. In our simulation, the number of nodes is n. Each mobile node transmits messages and data packets using IEEE 802.11b device. The radio propagation range of each node is adjusted to about 100 meters, and the network bandwidth is 11Mbps. Nodes move according to the random way point model [6], where the maximum move speed is v_{max} and the pause time is 2 seconds. We randomly choose a pair of source node and destination node every 2 seconds. If the source node has an active route to the destination node, the source node sends a data packet to the destination node directly. Otherwise the source node broadcasts an RReq to find a route to a destination node.

Note that attack models are categorized into two patterns: passive and proactive. Passive attacks include black hole attack, grey hole attack, sybil attack, routing packet dropping attack, rushing attack, wormhole attack, and jelly fish attack. Proactive attacks include RReq flooding attack and hello flooding attack. Malicious nodes execute a random passive attack when they receive messages. Malicious nodes execute RReq or hello flooding attack for 30 seconds and stop the attack for 30 seconds after the attack. (So, after the stop, they re-start one of the two attacks again.) During the attack, they broadcast an RReq or a hello packet every 0.5 second.

Training and testing data are obtained from the simulation executed by varying the parameters illustrated in Table 6. M ratio is the ratio of malicious nodes over n nodes. The time of each simulation is 300 seconds.

Table 6 Parameter configuration

Parameters	Values
v_{max} [m/s]	1.0, 2.0, 3.0, 4.0
Area size [m] × [m]	500 × 500, 600 × 600, 700 × 700, 800 × 800, 900 × 900, 1000 × 1000
M ratio	0.1, 0.2, 0.3, 0.4
n	50, 60, 70, 80, 90, 100

As we employed random forest as the classifier, so we tuned the parameters of them properly based on our simulations. The parameters of random forest were decided, i.e., batchSizePercent, batchSize, NumIterations, and numDecimalPlaces are 100, 100, 100, and 2, respectively.

4. 2 Result

We evaluate the performance of our proposed methods on several criteria.
(i) Accuracy: this measures the rate of how many instances are correctly classified among all the instances.
(ii) Detecting rate: this measures how many instances observed from malicious nodes are correctly classified.
(iii) False alarm rate: this measures how many instances observed from normal nodes are wrongly classified.
The objective of our method is to detect malicious nodes, so we consider detecting rate as the most important among these 3 criteria.

We compare the results in different situations.

（1） Situation 1 (S1): v_{max} of training set and testing set are different and the other parameters are the same.

（2） Situation 2 (S2): The area sizes of training set and testing set are different and the other parameters are the same.

（3） Situation 3 (S3): The M ratios of training set and testing set are different and the other parameters are the same.

（4） Situation 4 (S4): The node numbers of training set and testing set are different and the other parameters are the same.

（5） Situation 5 (S5): All the parameters in training set and testing set are totally different.

（6） Situation 6 (S6): From Table 6, we get data from 576 network parameter setting. We randomly choose 115 setting for testing sets, and the other 461 setting to obtain training sets.
We choose these situations because we want to find influence of each network parameter on our classifier, and we also want to test the robustness of our ensemble classifier based on different or unknown network environments. We compared our weighting mechanism with a naive method, in which the weight of all weak classifiers are set as 1.

Figure 3 shows the result of accuracy of the naive method and the weighting mechanism. From this figure, we can see that the naive method cannot perform well, and the accuracy is lower than 80% in each situation. Accuracy of the weighting mechanism, on the contrary, is very stable, and always higher than 80%, outperforming the naive method.

Figure 4 shows the result of false alarm rate. From this figure, false alarm rate of the weighting mechanism is lower than the naive method, and it is also very stable, always lower than 20%. In particular, even in S5 and S6, when the network parameters are random or totally different, the weighting mechanism can keep low false alarm rate, while the false alarm rate of the naive method is high.

Figure 5 shows the result of detecting rate. This figure illustrates that our weighting mechanism can keep a stable detecting rate in S1, S2, S3, S4, and S5. Detecting rate of the weighting mechanism is better than that of the naive method in all the situations.

Finally, we test different classifier building methods. We employ SVM, J48 tree and Naive Bayes. We only test them in S5. Figure 6 shows the result. From this figure, the ran-

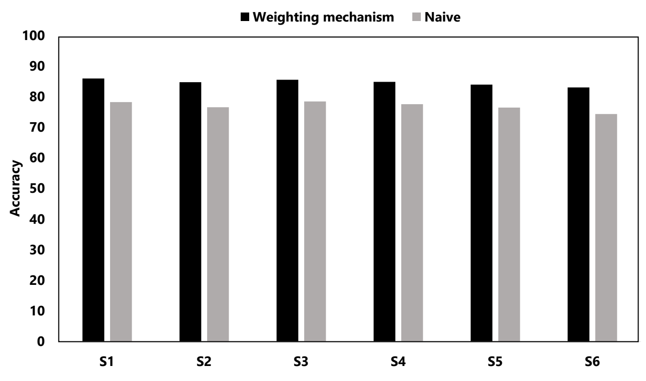

Figure 3 Accuracy

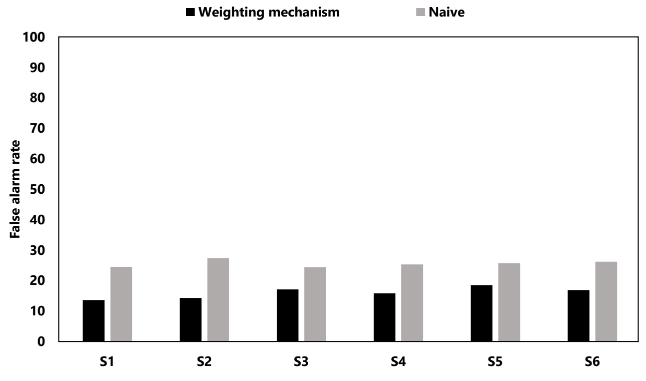

Figure 4 False alarm rate

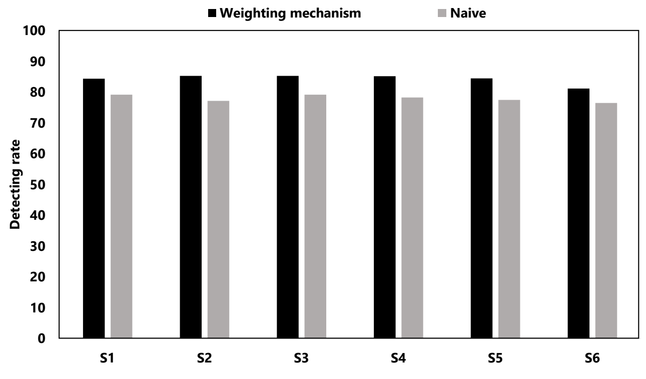

Figure 5 Comparison of detecting rate

dom forest outperforms the other classifiers. Note that J48 tree also performs well, proving that decision trees are effective for our proposed features.

5. Conclusion and future work

This paper addressed the problem of identifying malicious nodes in MANETs with different network parameters and multiple attacks. MANETs are vulnerable to malicious nodes, so it is important to identify malicious nodes in

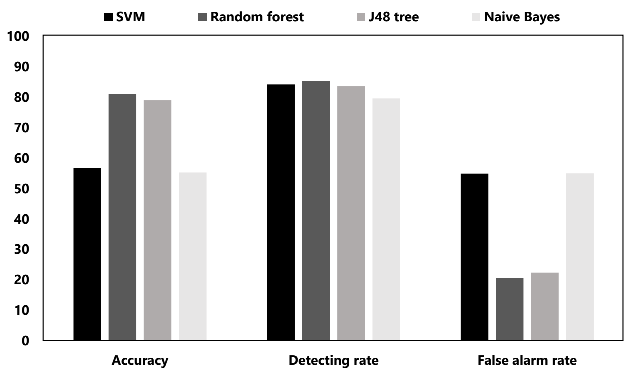

Figure 6 Comparison of SVM, Random forest, J48 tree, and Naive Bayes

MANETs for secure and proper communications. We proposed useful features for weak classifiers which judge neighbor nodes as malicious or normal nodes. For extracting useful features, we classified popular attacks in MANETs into four categories. We designed two sets of features: behavior features for classifying neighbors and environment features for estimating network environments. We used undersampling to deal with imbalanced data in data set obtained from simulations. We then used weights and weak classifiers to achieve final result. A series of experiments were conducted to measure the performances of our proposed method. The simulation results show that our method can identify malicious nodes in MANETs effectively.

As a part of our future work, we plan to create a group decision method for isolating malicious nodes, after they have been identified. Because only identifying malicious nodes is not enough, we have to consider a method to block them, making it no longer execute attacks.

References

[1] S. Abbas, M. Merabti, D. Llewellyn-Jones, and K. Kifayat. Lightweight sybil attack detection in manets. *IEEE systems journal*, 7(2):236–248, 2013.

[2] R. Akbani, T. Korkmaz, and G. Raju. Emltrust: An enhanced machine learning based reputation system for manets. *Ad Hoc Networks*, 10(3):435–457, 2012.

[3] J. Arora, P. Singh, and S. Rani. Attacks in manets–a survey. *EXCEL International Journal of Multidisciplinary Management Studies*, 3(10):151–157, 2013.

[4] P. Arora. A comparative study of instance reduction techniques. *International Journal of Advances in Engineering Sciences*, 3(3):7–13, 2013.

[5] A. Bala, M. Bansal, and J. Singh. Performance analysis of manet under blackhole attack. In *NeTCoM*, pages 141–145, 2009.

[6] C. Bettstetter, H. Hartenstein, and X. Pérez-Costa. Stochastic properties of the random waypoint mobility

model. *Wireless Networks*, 10(5):555–567, 2004.

[7] Z. Gao, Y. Yang, J. Zhao, J. Cui, and X. Li. Service discovery protocols for manets: A survey. In *MSN*, pages 232–243, 2006.

[8] H. He and E. A. Garcia. Learning from imbalanced data. *IEEE Transactions on Knowledge and Data Engineering*, 21(9):1263–1284, 2009.

[9] Y.-C. Hu, A. Perrig, and D. B. Johnson. Packet leashes: A defense against wormhole attacks in wireless networks. In *INFOCOM*, pages 1976–1986, 2003.

[10] Y.-C. Hu, A. Perrig, and D. B. Johnson. Rushing attacks and defense in wireless ad hoc network routing protocols. In *WiSe*, pages 30–40, 2003.

[11] A. Liaw and M. Wiener. Classification and regression by randomforest. *R news*, 2(3):18–22, 2002.

[12] S. Madhavi and K. Duraiswamy. Flooding attack aware secure aodv. *Journal of Computer Science*, 9(1):105–113, 2013.

[13] A. Mitrokotsa and C. Dimitrakakis. Intrusion detection in manet using classification algorithms: The effects of cost and model selection. *Ad Hoc Networks*, 11(1):226–237, 2013.

[14] A. Nadeem and M. P. Howarth. An intrusion detection & adaptive response mechanism for manets. *Ad Hoc Networks*, 13:368–380, 2014.

[15] H. L. Nguyen and U. T. Nguyen. A study of different types of attacks on multicast in mobile ad hoc networks. *Ad Hoc Networks*, 6(1):32–46, 2008.

[16] C. Perkins, E. Belding-Royer, and S. Das. Ad hoc on-demand distance vector (aodv) routing. Technical report, 2003.

[17] J. Sen, M. G. Chandra, S. Harihara, H. Reddy, and P. Balamuralidhar. A mechanism for detection of gray hole attack in mobile ad hoc networks. In *ICICS*, pages 1–5, 2007.

[18] Q. N. Simulator. Scalable network technologies. *Inc.[Online]. Available: www. qualnet. com*, 2011.

[19] Y. Zhang, W. Lee, and Y.-A. Huang. Intrusion detection techniques for mobile wireless networks. *Wireless Networks*, 9(5):545–556, 2003.