Misbehavior Detection in Vehicular Networks: An Ensemble Learning Approach

Roshan Sedar[1], Charalampos Kalalas[1], Paolo Dini[1], Jesus Alonso-Zarate[2] and Francisco Vazquez-Gallego[2]

[1] Centre Tecnològic de Telecomunicacions de Catalunya (CTTC), Barcelona, Spain

[2] i2CAT Foundation, Barcelona, Spain

IEEE Global Communications Conference

4–8 December 2022
Rio de Janeiro, Brazil
In-Person and Virtual Conference

IEEE GLOBECOM®

cttc

Outline

1. Background and motivation
2. Ensemble learning for misbehavior detection
3. Experiments and results
4. Conclusion

Background and motivation

Security vulnerabilities in vehicular networks

- Communication among vehicles, road-side units, and road users becomes highly vulnerable to malicious actors.

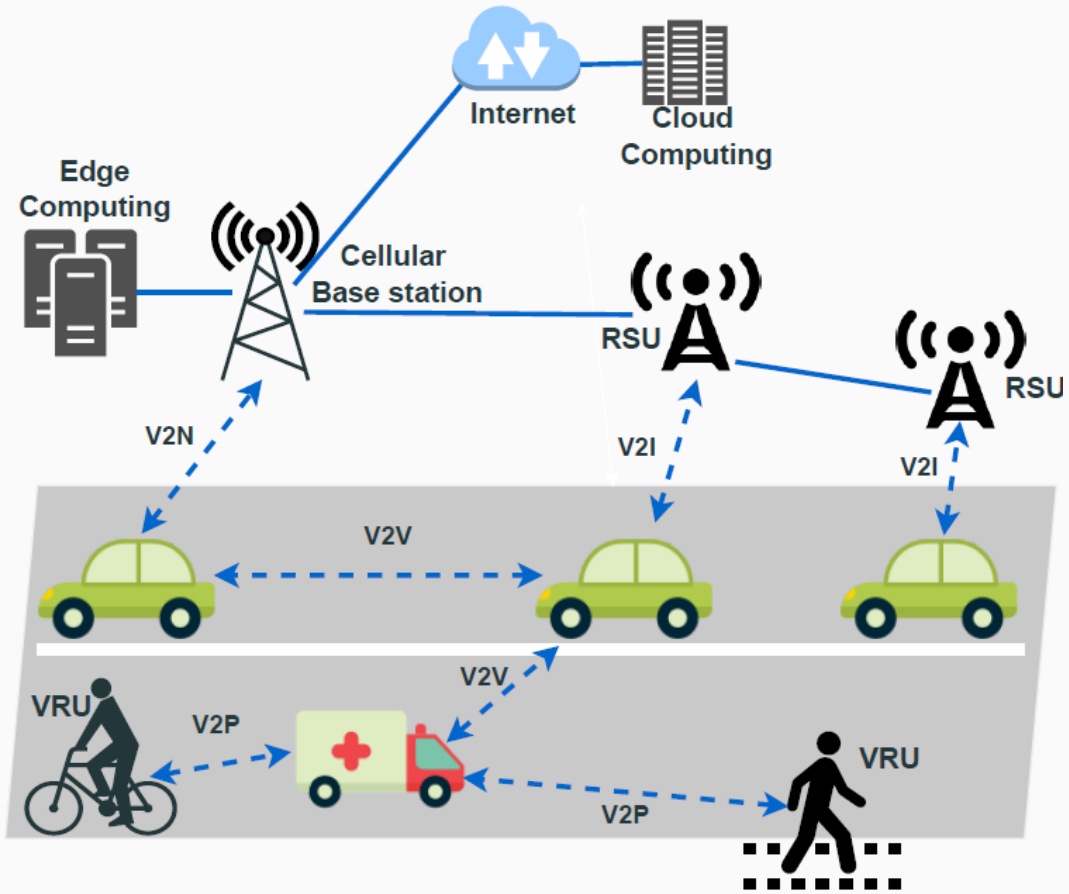


Figure 1: Vehicle-to-everything (V2X) connectivity

Security vulnerabilities in vehicular networks

- Communication among vehicles, road-side units, and road users becomes highly vulnerable to malicious actors.
- Novel security mechanisms are essential to address vulnerabilities and reduce the extent of their detrimental effects on safety-critical vehicular use cases.

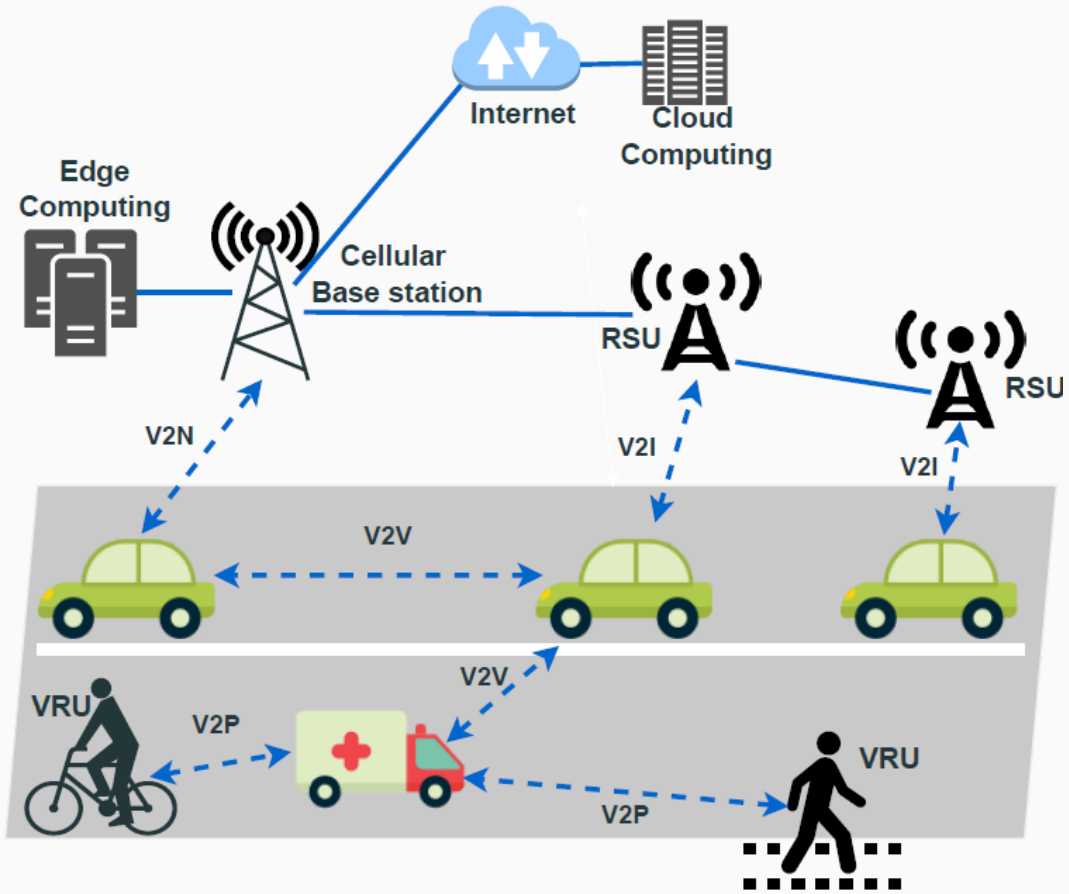
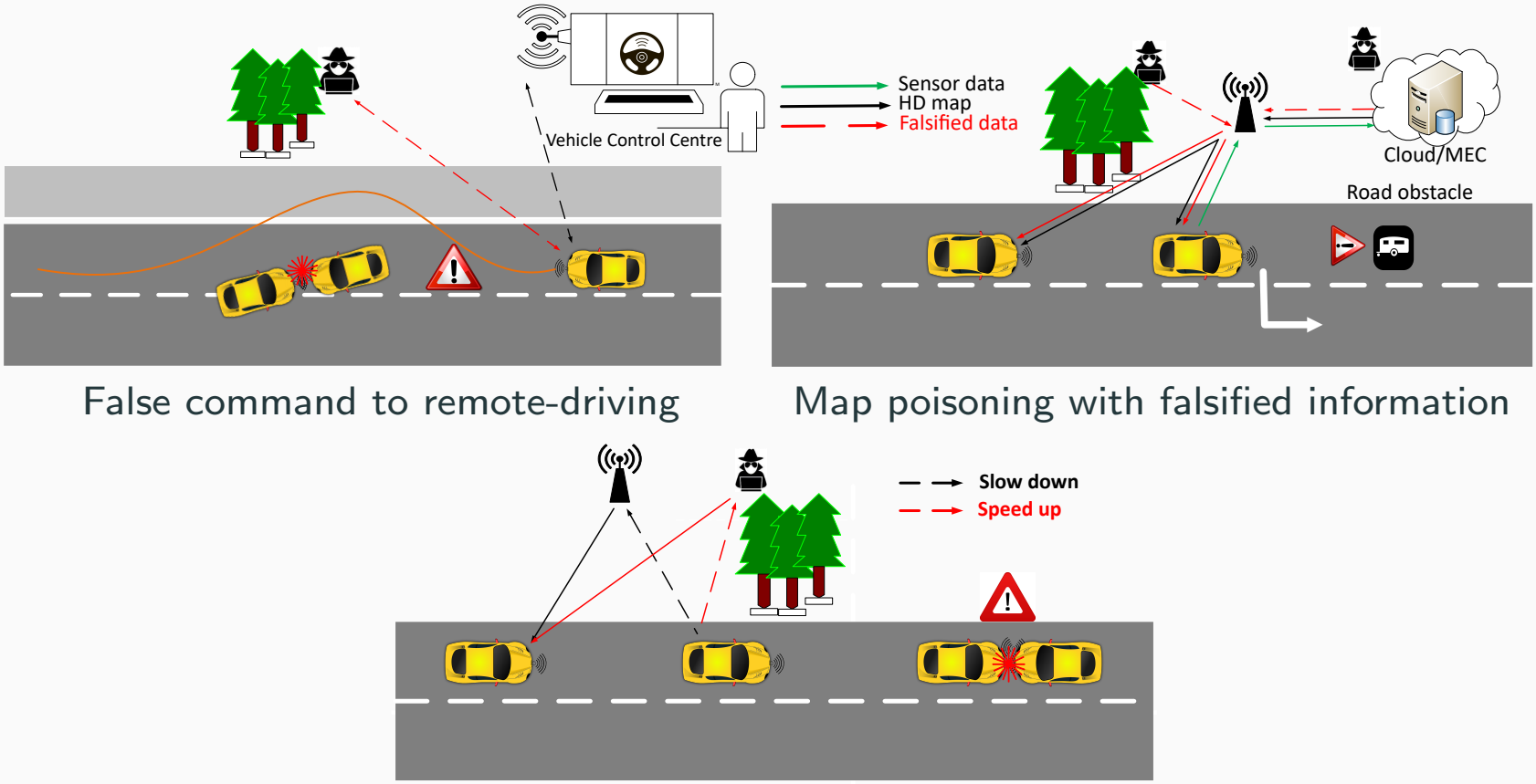


Figure 1: Vehicle-to-everything (V2X) connectivity

Misbehavior in vehicular networks



False command to remote-driving

Map poisoning with falsified information

Hazard data modification

Figure 2: Examples of misbehavior in the form of false data injection attacks

Challenges for misbehavior detection

Detection approaches:

- Entity-centric: Ephemeral V2X connections & high mobility
- Data-centric: Lack of global state information & assumption of honest majority

Challenges for misbehavior detection

Detection approaches:

- Entity-centric: Ephemeral V2X connections & high mobility
- Data-centric: Lack of global state information & assumption of honest majority

Machine learning tools, although promising, face challenges:

- Limited access to labeled training examples
- Dependence on security threshold values
- Unprecedented malicious activity & unforeseen changes in V2X traffic

Our approach

Ensemble learning framework:

- Unsupervised learning layer for discovering hidden patterns from unlabeled V2X traffic traces

Our approach

Ensemble learning framework:

- Unsupervised learning layer for discovering hidden patterns from unlabeled V2X traffic traces
- Reinforcement learning (RL) layer for consistently improving detection experience over unknown V2X environments without relying on security thresholds

Ensemble learning for misbehavior detection

Network scenario and attack model

- Sybil attack
- Data replay attack
- Denial-of-service (DoS) attack
- Disruptive attack

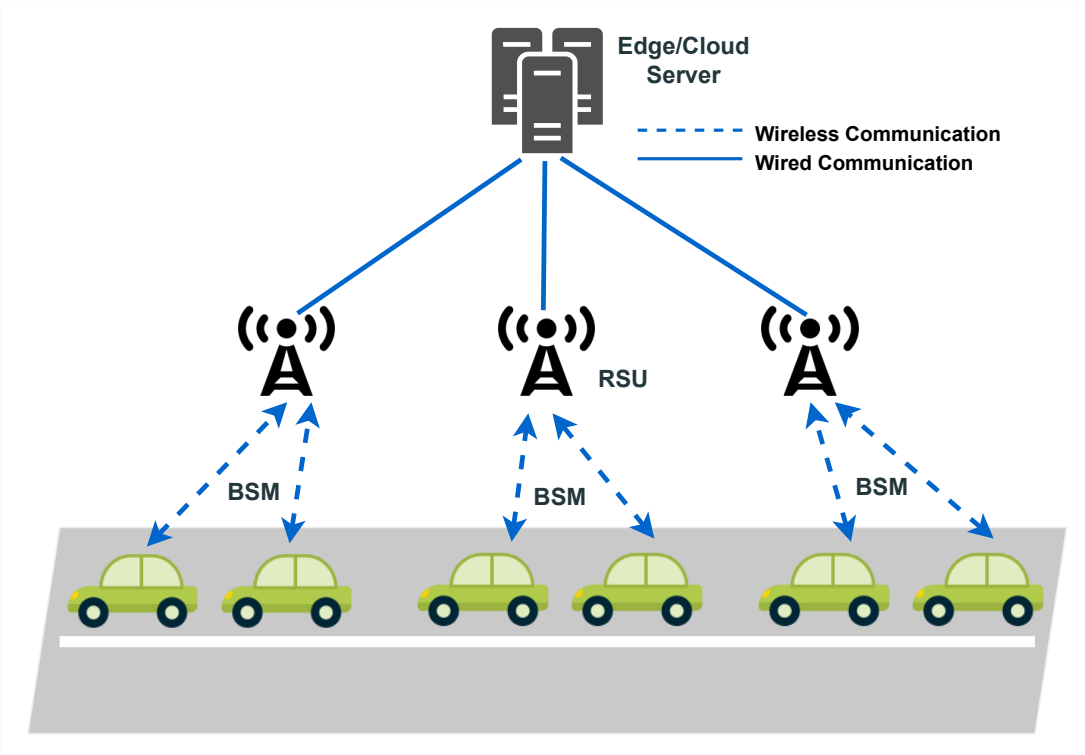
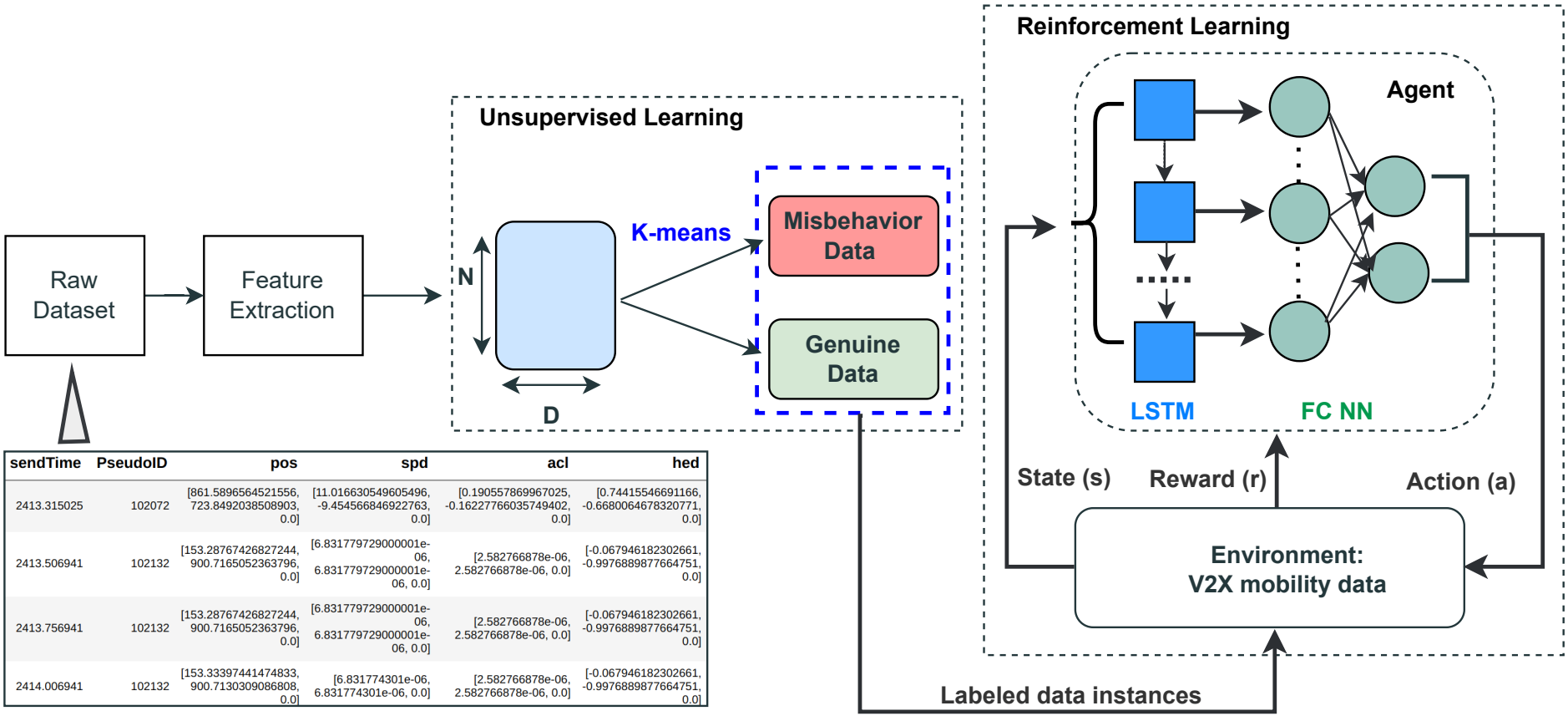


Figure 3: Considered scenario with vehicles transmitting basic safety messages (BSMs)

Methodology



sendTime	PseudoID	pos	spd	acl	hed
2413.315025	102072	[861.5896564521556, 723.8492038508903, 0.0]	[11.016630549605496, -9.454566846922763, 0.0]	[0.190557869967025, -0.16227766035749402, 0.0]	[0.74415546691166, -0.6680064678320771, 0.0]
2413.506941	102132	[153.28767426827244, 900.7165052363796, 0.0]	[6.831779729000001e-06, 6.831779729000001e-06, 0.0]	[2.582766878e-06, 2.582766878e-06, 0.0]	[-0.067946182302661, -0.9976889877664751, 0.0]
2413.756941	102132	[153.28767426827244, 900.7165052363796, 0.0]	[6.831779729000001e-06, 6.831779729000001e-06, 0.0]	[2.582766878e-06, 2.582766878e-06, 0.0]	[-0.067946182302661, -0.9976889877664751, 0.0]
2414.006941	102132	[153.33397441474833, 900.7130309086808, 0.0]	[6.831774301e-06, 6.831774301e-06, 0.0]	[2.582766878e-06, 2.582766878e-06, 0.0]	[-0.067946182302661, -0.9976889877664751, 0.0]

Figure 4: Proposed ensemble learning framework for unsupervised data preprocessing and RL-based misbehavior detection.

Unsupervised learning: K-means algorithm

- Generates ground truth information necessary for RL-based detection

Algorithm 1 Clustering and Labeling with K-means

1: **Input**: $X : x^{(1)}, ..., x^{(m)}$ **Output**: $Y : y^{(i)}, ..., y^{(m)}$
2: Initialize cluster centroids $\mu_1, \mu_2, ...\mu_k \in \mathbb{R}^n$ randomly
3: **for** k = 1 to 11 **do**
4: Repeat until convergence : {
5: For every i, $c^{(i)} := \arg\min_j \|x^{(i)} - \mu_j\|^2 \ \forall j, j \leq k$
6: For each j, $\mu_j := \frac{\sum_{i=1}^{m} 1\{c^{(i)} = j\} x^{(i)}}{\sum_{i=1}^{m} 1\{c^{(i)} = j\}}$
7: }
8: $\mathrm{J}^{(k)}(c, \mu) = \arg\min_{\mu, c} \sum_{i=1}^{m} \sum_{k=1}^{K} 1\{c_i = k\} \|x^{(i)} - \mu_k\|^2$
9: **end for**
10: Optimum $K \leftarrow$ **Elbow** method $\leftarrow$ plot $\mathrm{J}^{(k)}$ vs k
11: **for** i = 1 to m **do**
12: Run K-means on X with optimum K (lines 4 - 7)
13: **end for**
14: **for** i = 1 to m **do**
15: **if** $x^{(i)} \in$ lowest cluster samples
16: $y^{(i)} = 1$
17: **else if**
18: $y^{(i)} = 0$
19: **end if**
20: **end for**

Unsupervised learning: K-means algorithm

- Generates ground truth information necessary for RL-based detection
- In each iteration, it measures the similarity of data instances by computing their Euclidean distance from the centroid on the dimension of the feature vector

Algorithm 1 Clustering and Labeling with K-means

1: **Input**: $X : x^{(1)}, ..., x^{(m)}$ **Output**: $Y : y^{(i)}, ..., y^{(m)}$
2: Initialize cluster centroids $\mu_1, \mu_2, ...\mu_k \in \mathbb{R}^n$ randomly
3: **for** k = 1 to 11 **do**
4: Repeat until convergence : {
5: For every i, $c^{(i)} := \arg\min_j \|x^{(i)} - \mu_j\|^2 \ \forall j, j \leq k$
6: For each j, $\mu_j := \frac{\sum_{i=1}^{m} 1\{c^{(i)} = j\} x^{(i)}}{\sum_{i=1}^{m} 1\{c^{(i)} = j\}}$
7: }
8: $\mathrm{J}^{(k)}(c, \mu) = \arg\min_{\mu, c} \sum_{i=1}^{m} \sum_{k=1}^{K} 1\{c_i = k\} \|x^{(i)} - \mu_k\|^2$
9: **end for**
10: Optimum $K \leftarrow$ **Elbow** method $\leftarrow$ plot $\mathrm{J}^{(k)}$ vs k
11: **for** i = 1 to m **do**
12: Run K-means on X with optimum K (lines 4 - 7)
13: **end for**
14: **for** i = 1 to m **do**
15: **if** $x^{(i)} \in$ lowest cluster samples
16: $y^{(i)} = 1$
17: **else if**
18: $y^{(i)} = 0$
19: **end if**
20: **end for**

Unsupervised learning: K-means algorithm

- Generates ground truth information necessary for RL-based detection
- In each iteration, it measures the similarity of data instances by computing their Euclidean distance from the centroid on the dimension of the feature vector
- Instances belonging to the cluster with the lowest number of samples are labeled as misbehaving

Algorithm 1 Clustering and Labeling with K-means

1: **Input**: $X : x^{(1)}, ..., x^{(m)}$ **Output**: $Y : y^{(i)}, ..., y^{(m)}$
2: Initialize cluster centroids $\mu_1, \mu_2, ...\mu_k \in \mathbb{R}^n$ randomly
3: **for** k = 1 to 11 **do**
4: Repeat until convergence : {
5: For every i, $c^{(i)} := \arg\min_j \|x^{(i)} - \mu_j\|^2 \ \forall j, j \leq k$
6: For each j, $\mu_j := \frac{\sum_{i=1}^{m} 1\{c^{(i)} = j\} x^{(i)}}{\sum_{i=1}^{m} 1\{c^{(i)} = j\}}$
7: }
8: $\mathrm{J}^{(k)}(c, \mu) = \arg\min_{\mu, c} \sum_{i=1}^{m} \sum_{k=1}^{K} 1\{c_i = k\} \|x^{(i)} - \mu_k\|^2$
9: **end for**
10: Optimum $K \leftarrow$ **Elbow** method $\leftarrow$ plot $\mathrm{J}^{(k)}$ vs k
11: **for** i = 1 to m **do**
12: Run K-means on X with optimum K (lines 4 - 7)
13: **end for**
14: **for** i = 1 to m **do**
15: **if** $x^{(i)} \in$ lowest cluster samples
16: $y^{(i)} = 1$
17: **else if**
18: $y^{(i)} = 0$
19: **end if**
20: **end for**

RL-based detection

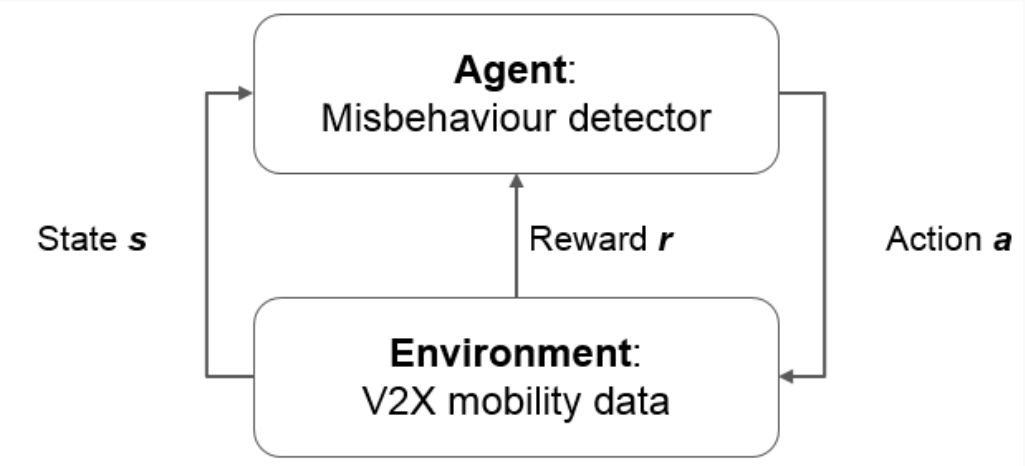


Figure 5: Detection based on RL model

RL-based detection

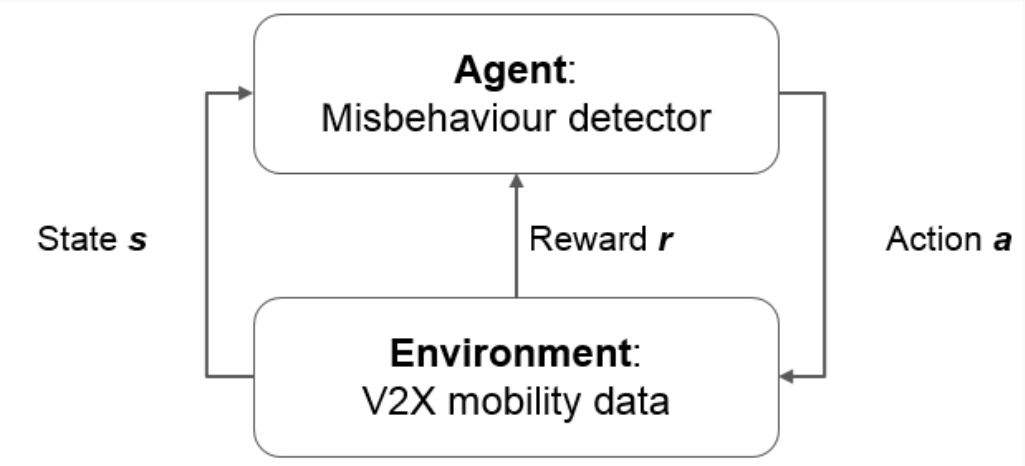


Figure 5: Detection based on RL model

- $Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha(r_t + \gamma \max_{a_{t+1}} Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t))$.

RL-based detection

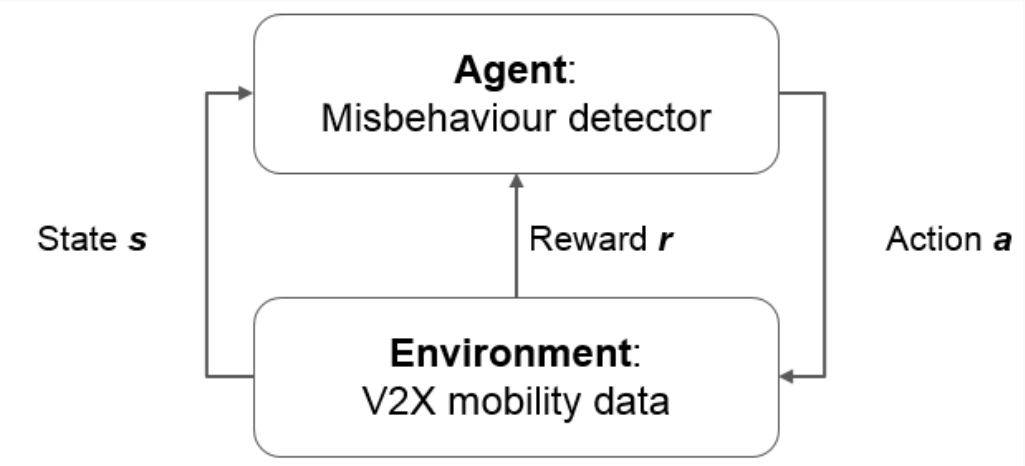


Figure 5: Detection based on RL model

- $Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha(r_t + \gamma \max_{a_{t+1}} Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t))$.
- State contains the sequence of previous actions and the current vehicular data.

RL-based detection

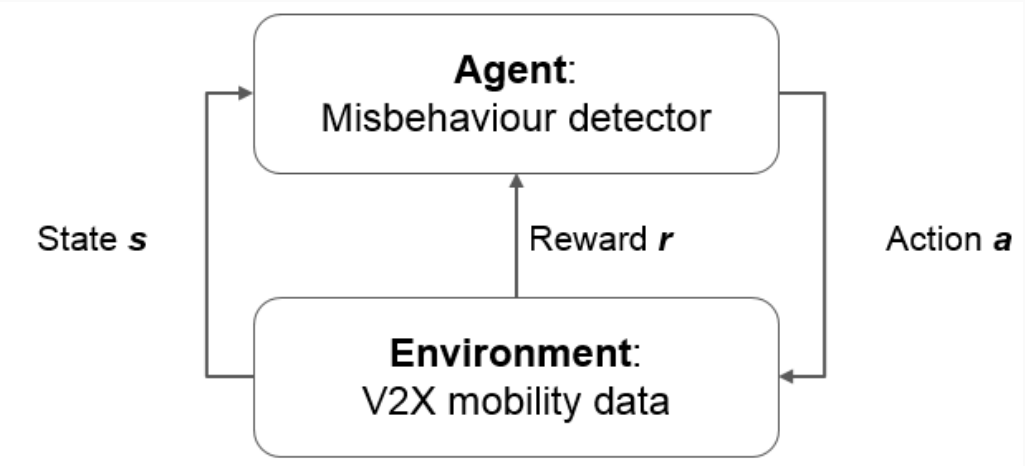


Figure 5: Detection based on RL model

- $Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha(r_t + \gamma \max_{a_{t+1}} Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t))$.
- State contains the sequence of previous actions and the current vehicular data.
- Agent selects the action a as: $\pi^*(s) = \arg\max_{a \in A} Q^*(s, a)$.

RL-based detection

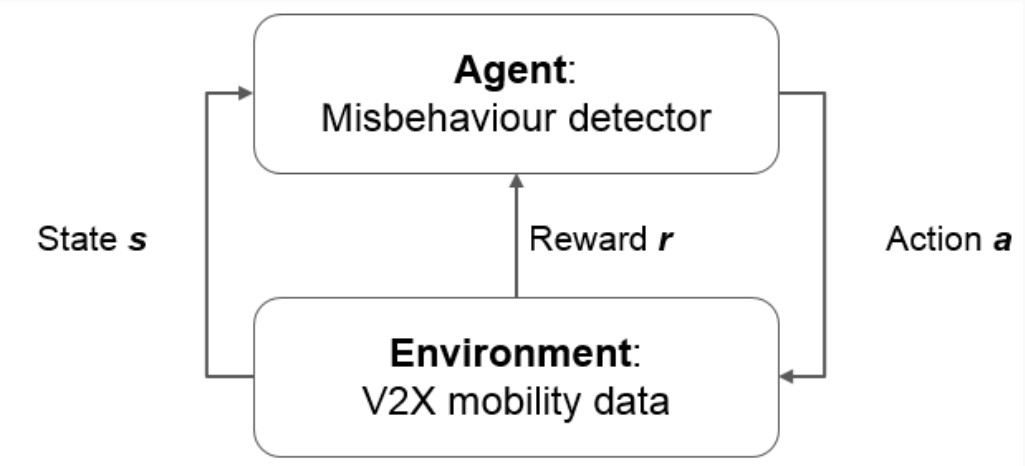


Figure 5: Detection based on RL model

- $Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha(r_t + \gamma \max_{a_{t+1}} Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t))$.
- State contains the sequence of previous actions and the current vehicular data.
- Agent selects the action a as: $\pi^*(s) = \arg\max_{a \in A} Q^*(s, a)$.
- Reward function R as: $R_t = \sum_{k=t}^{T} \gamma^{k-t} r_k$:
  - Positive reward: True Positive (TP) or True Negative (TN).
  - Negative reward: False Positive (FP) or False Negative (FN).

Experiments and results

Dataset

- Vehicle traces in line with real-world field tests[1]; open source synthetic traffic scenario validated with real data provided by the VehicularLab of the University of Luxembourg.

	type	sendTime	sender	senderPseudo	messageID	pos	spd	acl	hed
17	4	25210.186332	33	10332	21562	[1393.9276845310885, 1203.692849621629, 0.0]	[0.049400297340067005, -0.68607427873154 2, 0.0]	[0.166603725521922, -2.313798731172836, 0.0]	[0.063269582720791, -0.997996472890729 1, 0.0]
19	4	25210.436332	33	10332	21595	[1393.9276845310885, 1203.692849621629, 0.0]	[0.049400297340067005, -0.68607427873154 2, 0.0]	[0.166603725521922, -2.313798731172836, 0.0]	[0.063269582720791, -0.997996472890729 1, 0.0]
21	4	25210.686332	33	10332	21622	[1393.9276845310885, 1203.692849621629, 0.0]	[0.049400297340067005, -0.68607427873154 2, 0.0]	[0.166603725521922, -2.313798731172836, 0.0]	[0.063269582720791, -0.997996472890729 1, 0.0]
25	4	25210.936332	33	10332	21696	[1393.9276845310885, 1203.692849621629, 0.0]	[0.049400297340067005, -0.68607427873154 2, 0.0]	[0.166603725521922, -2.313798731172836, 0.0]	[0.063269582720791, -0.997996472890729 1, 0.0]
27	4	25211.186332	33	10332	30134	[1394.1720072035407, 1201.94700381985, 0.0]	[0.183983214273645, -2.555169745474803, 0.0]	[0.158360369223177, -2.199314281648395, 0.0]	[0.063269582720943, -0.997996472890729 2, 0.0]
...	...	...	...	...	...	...	...	...	...
7165	4	25368.936332	33	10332	715829	[127.9440058255349, 885.9631063084606, 0.0]	[-8.275102433876064, -0.48628168363595903, 0.0]	[4.492259181435928, 0.263999685982995, 0.0]	[-0.9703792835172421, 0.24158651891312902, 0.0]
7179	4	25369.186332	33	10332	716850	[122.01936718688863, 885.6254381946134, 0.0]	[-3.793979120628686, -0.21788847407861903, 0.0]	[4.49260375347267, 0.25802779102927603, 0.0]	[-0.9731995855753991, 0.22996209825940903, 0.0]
7192	4	25369.436332	33	10332	717806	[122.01936718688863, 885.6254381946134, 0.0]	[-3.793979120628686, -0.21788847407861903, 0.0]	[4.49260375347267, 0.25802779102927603, 0.0]	[-0.9731995855753991, 0.22996209825940903, 0.0]
7205	4	25369.686332	33	10332	718725	[122.01936718688863, 885.6254381946134, 0.0]	[-3.793979120628686, -0.21788847407861903, 0.0]	[4.49260375347267, 0.25802779102927603, 0.0]	[-0.9731995855753991, 0.22996209825940903, 0.0]
7228	4	25369.936332	33	10332	720286	[122.01936718688863, 885.6254381946134, 0.0]	[-3.793979120628686, -0.21788847407861903, 0.0]	[4.49260375347267, 0.25802779102927603, 0.0]	[-0.9731995855753991, 0.22996209825940903, 0.0]

640 rows × 9 columns

Figure 6: VeReMi dataset

[1] VeReMi Dataset [Online]: https://github.com/josephkamel/VeReMi-Dataset

Settings and performance metrics

- A high-density (37.03 vehicles/km^2) traffic scenario is used for training, while a low-density one (16.36 vehicles/km^2) is used for testing.

Settings and performance metrics

- A high-density (37.03 vehicles/km^2) traffic scenario is used for training, while a low-density one (16.36 vehicles/km^2) is used for testing.
- Varying proportion of misbehaving and legitimate vehicles.

Settings and performance metrics

- A high-density (37.03 vehicles/km^2) traffic scenario is used for training, while a low-density one (16.36 vehicles/km^2) is used for testing.
- Varying proportion of misbehaving and legitimate vehicles.
- Feature engineering in exchanged messages:
  - Timestamp, pseudo-identity, position, speed, acceleration, heading angle.
  - Euclidean norm of position, speed, acceleration and heading angle vectors.

Settings and performance metrics

- A high-density (37.03 vehicles/km^2) traffic scenario is used for training, while a low-density one (16.36 vehicles/km^2) is used for testing.
- Varying proportion of misbehaving and legitimate vehicles.
- Feature engineering in exchanged messages:
  - Timestamp, pseudo-identity, position, speed, acceleration, heading angle.
  - Euclidean norm of position, speed, acceleration and heading angle vectors.
- Detection performance was evaluated based on commonly used metrics:
  - Accuracy $= \dfrac{TP + TN}{TP + TN + FP + FN}$,
  - Precision $= \dfrac{TP}{TP + FP}$,
  - Recall $= \dfrac{TP}{TP + FN}$,
  - F1 $= \dfrac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$.

Clustering performance

Attack scenario	K-means	Spectral
Constant position	0.719	0.206
Random speed	0.719	0.152
Random speed offset	0.718	0.059

Table 1: Average silhouette score

- Spectral clustering algorithm treats data clustering as a graph partitioning problem and offers equivalent simplicity as K-means
- Average silhouette coefficient is computed for each sample using the mean intra-cluster and inter-cluster distance

RL-based detection performance

- **Effectively** detected attacks

Type	Attack	Accuracy	Precision	Recall	F1
1	Constant Position	0.9892	0.9648	1.0	0.9820
2	Constant Position Offset	0.9853	0.9512	1.0	0.9750
3	Random Position	0.9915	0.9724	1.0	0.9860
4	Random Position Offset	0.9831	0.9454	1.0	0.9719
5	Constant Speed	0.9918	0.9733	1.0	0.9864
6	Constant Speed Offset	0.9895	0.9661	1.0	0.9874
7	Random Speed	0.9924	0.9751	1.0	0.9874
8	Random Speed Offset	0.9913	0.9716	1.0	0.9856
9	Sudden Stop	0.8038	0.5839	0.7080	0.6400
10	Disruptive	0.9610	0.9868	0.9205	0.9525
11	Data Replay	0.9698	0.9826	0.9461	0.9640
12	Delayed Messages	0.9438	0.8445	1.0	0.9157
13	DoS	0.9539	0.9928	0.8922	0.9398
14	DoS Random	0.6411	0.6338	1.0	0.7759
15	DoS Disruptive	0.6353	0.6306	1.0	0.7735
16	Traffic Congestion Sybil	0.9895	0.9661	1.0	0.9827
17	Data Replay Sybil	0.7527	0.6166	0.9612	0.7512
18	DoS Random Sybil	0.7973	0.9507	0.4845	0.6419
19	DoS Disruptive Sybil	0.6501	0.8608	0.0714	0.1318

Table 2: Detection performance per attack

RL-based detection performance

- **Effectively** detected attacks
- **Moderately** detected attacks

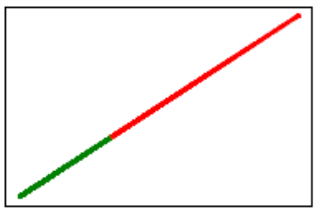
Type	Attack	Accuracy	Precision	Recall	F1
1	Constant Position	0.9892	0.9648	1.0	0.9820
2	Constant Position Offset	0.9853	0.9512	1.0	0.9750
3	Random Position	0.9915	0.9724	1.0	0.9860
4	Random Position Offset	0.9831	0.9454	1.0	0.9719
5	Constant Speed	0.9918	0.9733	1.0	0.9864
6	Constant Speed Offset	0.9895	0.9661	1.0	0.9874
7	Random Speed	0.9924	0.9751	1.0	0.9874
8	Random Speed Offset	0.9913	0.9716	1.0	0.9856
9	Sudden Stop	0.8038	0.5839	0.7080	0.6400
10	Disruptive	0.9610	0.9868	0.9205	0.9525
11	Data Replay	0.9698	0.9826	0.9461	0.9640
12	Delayed Messages	0.9438	0.8445	1.0	0.9157
13	DoS	0.9539	0.9928	0.8922	0.9398
14	DoS Random	0.6411	0.6338	1.0	0.7759
15	DoS Disruptive	0.6353	0.6306	1.0	0.7735
16	Traffic Congestion Sybil	0.9895	0.9661	1.0	0.9827
17	Data Replay Sybil	0.7527	0.6166	0.9612	0.7512
18	DoS Random Sybil	0.7973	0.9507	0.4845	0.6419
19	DoS Disruptive Sybil	0.6501	0.8608	0.0714	0.1318

Table 2: Detection performance per attack

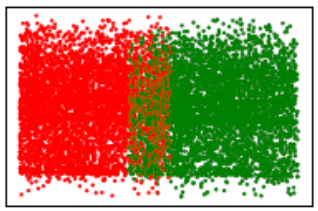
RL-based detection performance

- **Effectively** detected attacks
- **Moderately** detected attacks

Clustering output (red:misbehavior):

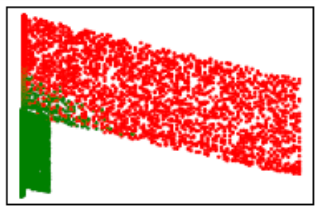


Constant position



DoS random Sybil

Sudden stop



DoS disruptive Sybil

Type	Attack	Accuracy	Precision	Recall	F1
1	Constant Position	0.9892	0.9648	1.0	0.9820
2	Constant Position Offset	0.9853	0.9512	1.0	0.9750
3	Random Position	0.9915	0.9724	1.0	0.9860
4	Random Position Offset	0.9831	0.9454	1.0	0.9719
5	Constant Speed	0.9918	0.9733	1.0	0.9864
6	Constant Speed Offset	0.9895	0.9661	1.0	0.9874
7	Random Speed	0.9924	0.9751	1.0	0.9874
8	Random Speed Offset	0.9913	0.9716	1.0	0.9856
9	Sudden Stop	0.8038	0.5839	0.7080	0.6400
10	Disruptive	0.9610	0.9868	0.9205	0.9525
11	Data Replay	0.9698	0.9826	0.9461	0.9640
12	Delayed Messages	0.9438	0.8445	1.0	0.9157
13	DoS	0.9539	0.9928	0.8922	0.9398
14	DoS Random	0.6411	0.6338	1.0	0.7759
15	DoS Disruptive	0.6353	0.6306	1.0	0.7735
16	Traffic Congestion Sybil	0.9895	0.9661	1.0	0.9827
17	Data Replay Sybil	0.7527	0.6166	0.9612	0.7512
18	DoS Random Sybil	0.7973	0.9507	0.4845	0.6419
19	DoS Disruptive Sybil	0.6501	0.8608	0.0714	0.1318

Table 2: Detection performance per attack

Benchmark comparison

- Benchmark misbehavior detectors[a]:
  - Support vector machine (SVM)
  - Multilayer perceptron (MLP)

[a] J. Kamel et al., "Simulation framework for misbehavior detection in vehicular networks," IEEE Trans. Veh. Technol., vol. 69, no. 6, pp. 6631– 6643, 2020

Attack type	Approach	Accuracy	Precision	Recall	F1
1	**K-means + MLP**	0.9902	1.0	0.9669	**0.9831**
	K-means + SVM	0.9418	1.0	0.8031	0.8908
	K-means + RL	0.9892	0.9648	1.0	0.9820
9	**K-means + MLP**	0.5412	0.2057	0.3007	0.2443
	K-means + SVM	0.5348	0.2066	0.3122	0.2486
	K-means + RL	0.8038	0.5839	0.7080	**0.6400**
10	**K-means + MLP**	0.4604	0.4407	1.0	0.6118
	K-means + SVM	0.9385	0.8868	0.9805	0.9313
	K-means + RL	0.9610	0.9868	0.9205	**0.9525**
16	**K-means + MLP**	0.6141	0.6084	0.9781	0.7502
	K-means + SVM	0.6711	0.6582	0.9257	0.7693
	K-means + RL	0.9895	0.9661	1.0	**0.9827**

Table 3: Detection performance comparison

Benchmark comparison

- Benchmark misbehavior detectors[a]:
  - Support vector machine (SVM)
  - Multilayer perceptron (MLP)
- Potentially inaccurate or mislabeled training data limit the performance of SVM and MLP.

[a] J. Kamel et al., "Simulation framework for misbehavior detection in vehicular networks," IEEE Trans. Veh. Technol., vol. 69, no. 6, pp. 6631– 6643, 2020

Attack type	Approach	Accuracy	Precision	Recall	F1
1	K-means + MLP	0.9902	1.0	0.9669	**0.9831**
	K-means + SVM	0.9418	1.0	0.8031	0.8908
	K-means + RL	0.9892	0.9648	1.0	0.9820
9	K-means + MLP	0.5412	0.2057	0.3007	0.2443
	K-means + SVM	0.5348	0.2066	0.3122	0.2486
	K-means + RL	0.8038	0.5839	0.7080	**0.6400**
10	K-means + MLP	0.4604	0.4407	1.0	0.6118
	K-means + SVM	0.9385	0.8868	0.9805	0.9313
	K-means + RL	0.9610	0.9868	0.9205	**0.9525**
16	K-means + MLP	0.6141	0.6084	0.9781	0.7502
	K-means + SVM	0.6711	0.6582	0.9257	0.7693
	K-means + RL	0.9895	0.9661	1.0	**0.9827**

Table 3: Detection performance comparison

Benchmark comparison

- Benchmark misbehavior detectors[a]:
  - Support vector machine (SVM)
  - Multilayer perceptron (MLP)
- Potentially inaccurate or mislabeled training data limit the performance of SVM and MLP.
- RL-based detection is shown to be less sensitive to inaccurate labels.

Attack type	Approach	Accuracy	Precision	Recall	F1
1	K-means + MLP	0.9902	1.0	0.9669	**0.9831**
	K-means + SVM	0.9418	1.0	0.8031	0.8908
	K-means + RL	0.9892	0.9648	1.0	0.9820
9	K-means + MLP	0.5412	0.2057	0.3007	0.2443
	K-means + SVM	0.5348	0.2066	0.3122	0.2486
	K-means + RL	0.8038	0.5839	0.7080	**0.6400**
10	K-means + MLP	0.4604	0.4407	1.0	0.6118
	K-means + SVM	0.9385	0.8868	0.9805	0.9313
	K-means + RL	0.9610	0.9868	0.9205	**0.9525**
16	K-means + MLP	0.6141	0.6084	0.9781	0.7502
	K-means + SVM	0.6711	0.6582	0.9257	0.7693
	K-means + RL	0.9895	0.9661	1.0	**0.9827**

Table 3: Detection performance comparison

[a] J. Kamel et al., "Simulation framework for misbehavior detection in vehicular networks," IEEE Trans. Veh. Technol., vol. 69, no. 6, pp. 6631– 6643, 2020

Real-time detection capabilities

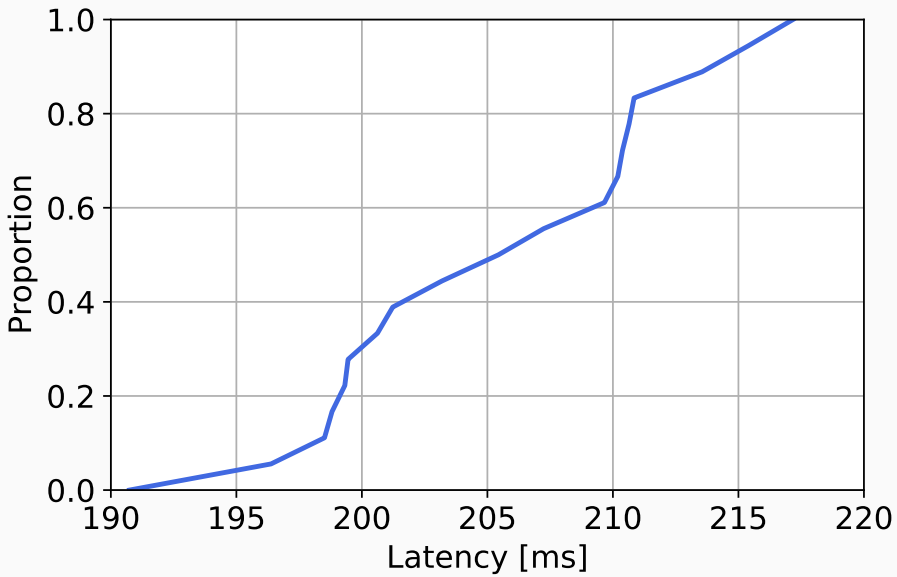


Figure 7: CDF of overall latency for testing datasets.

- Overall latency consists of the cumulative time elapsed for (*i*) environment setup (*ii*) loading a trained model (*iii*) detection.
- Average latency measured for steps (*i*)-(*iii*) is: 19.93 ms, 182.12 ms, and 3.15 ms, respectively.

Conclusion

Concluding remarks and path forward

- An ensemble learning methodology is introduced to accurately detect misbehaving vehicles in vehicular networks.
- While the majority of attack variants can be effectively detected, detection was curtailed for certain misbehavior types.
- RL-based misbehavior detection is shown to be more robust to noisy training data compared to its classifier counterparts.

Concluding remarks and path forward

- An ensemble learning methodology is introduced to accurately detect misbehaving vehicles in vehicular networks.
- While the majority of attack variants can be effectively detected, detection was curtailed for certain misbehavior types.
- RL-based misbehavior detection is shown to be more robust to noisy training data compared to its classifier counterparts.

Future work:

- Incorporate trust of road side units into collaborative misbehavior detection, by leveraging the real-time capabilities of our framework.

Thank you for your attention!

Questions?

Contact: ckalalas [at] cttc [at] es

This work has been funded by the "Ministerio de Asuntos Economicos y Transformacion Digital" and the European Union-NextGenerationEU in the frameworks of the "Plan de Recuperacion, Transformacion y Resiliencia" and of the "Mecanismo de Recuperacion y Resiliencia" under references TSI-063000-2021-39/40/41, by the H2020-INSPIRE-5Gplus project (Grant agreement No. 871808), and by ONOFRE-3 PID2020-112675RB-C43 funded by MCIN/ AEI /10.13039/501100011033 project.