A Appendix A

A.1 General Prompt and Reasoning Trace of IDS-Agent

The general prompt of the IDS-Agent is illustrated in Figure 3. The process begins with instructing the IDS-Agent to load network traffic data and perform feature preprocessing. Afterward, we utilize a range of classifiers to analyze the data. To enhance decision-making, the IDS-Agent retrieves prior successful examples from its knowledge base for comparison. In cases where discrepancies arise between the predictions of different models, we prompt the IDS-Agent to consult internal or external knowledge bases for additional insights to resolve the conflict. Finally, the IDS-Agent consolidates the findings and presents the result in a structured JSON format. Figure 4 provides an example of the reasoning trace produced by the IDS-Agent during this process.

General Prompt:
You are a helpful assistant that can implement multi-step tasks, such as intrusion detection. I will give you the traffic features, you are asked to classify it using tools. The final output must be in a JSON format according to the classifier results. You should plan first such as:

1. Load the traffic features from the CSV file. You can use the load_data_line tool to obtain the complete traffic.
2. Preprocessing the feature. This can be done using the data_preprocessing tool. Input the traffic in the original format.
3. load classifiers for classification. This can be done using the classifier tool. You can use multiple classifiers. The tool params include a classifier name, which must be one from {model_names} and the preprocessed features.
4. Retrieve previous successful reasonings to help you predict. This can be done using the memory_retrieve tool with the classifier's names and their classification results as input.
5. When there are discrepancies/disagreements for different models, you can search from vector database/google/wiki to get more information about the difference of attacks to help you make decisions.
6. At the end, you should summarize the results from these classifiers and provide a final result. Summarize the classification with Balance sensitivity, which means balancing the false alarm rate and the missing alarm rate. The predicted label should be the original format of classifier prediction. The final output format **must** be:

Final Answer:
```json
{ 'line_number': \line_number,
'analysis': str, \here is the Analysis,
'predicted_label_top_1': str,
'predicted_label_top_2': str,
'predicted_label_top_3': str,
}```
**User Input:**
Now, classify the traffic from file name {file_name} with index {line_number}

Figure 3: General prompt and user input

## A.2 Additional Case Studies

The cases in Figure 5 and Figure 6 highlight the enhanced reasoning ability of IDS-Agent with the knowledge retriever. It is shown that IDS-Agent not only considers the top-1 predictions but also the second and third predictions and their confidences. Moreover, in these examples, when the models have discrepancies in the predictions, the IDS-Agent automatically accesses external databases to extract additional knowledge, aiding in feature analysis and supporting its final decision. By leveraging these external knowledge sources, the IDS-Agent gains a deeper understanding of complex attack patterns and anomalies, enhancing both accuracy and decision-making. This dynamic capability allows the IDS-Agent to better adapt to new or evolving threats in the IoT environment.

## A.3 The Performance of Different ML Classifiers

Table 6 shows the F-score of different ML classifiers on the ACI-IoT'23 dataset as well as our method. Among the six classifiers, MLP achieves the highest F-score of 0.96. The IDS-Agent outperforms all six classifiers and the majority vote method. Moreover, we achieve a high F-score on the UDP Flood attack, while the majority method only has an F-score of 0.55.

Table 7 shows the F-score of different ML classifiers on the CIC-IoT'23 dataset as well as our method. Among the six classifiers, Random Forest achieves the highest F-score of 0.75. For the IDS-Agent, we use the GPT-4o as the core LLM. It is shown that our attack achieved a higher F-score compared with the majority vote classifier. Moreover, we achieved a higher F-score on the
```

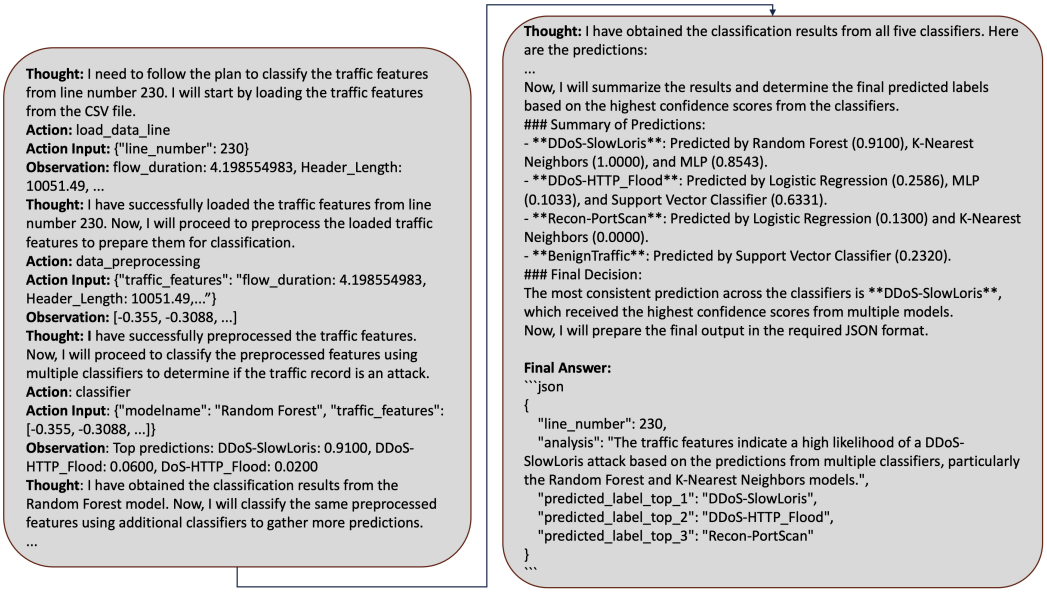


Figure 4: An example of the reasoning trace and final answer.

Thought: I have obtained predictions from all classifiers. Now, I will summarize the results to determine the final classification.
The predictions from the classifiers are as follows:
- Random Forest: DDoS-SYN_Flood (0.6458)
- K-Nearest Neighbors: DoS-SYN_Flood (1.0000)
- Logistic Regression: DDoS-SYN_Flood (0.6473)
- MLP: DDoS-SYN_Flood (0.7261)
- Support Vector Classifier: DDoS-SYN_Flood (0.6329)
- Decision Tree: DDoS-SYN_Flood (0.6736)

The majority of classifiers predict "DDoS-SYN_Flood" as the top label. However, K-Nearest Neighbors predicts "DoS-SYN_Flood" with a certainty of 1.0000. To resolve this discrepancy, I will search for more information regarding the features of DDoS and DoS attacks to understand the differences better.

Action: knowledge_retriever
Action Input: {"query": "What are the features of DoS and DDoS attacks and their differences?"
Observation: DoS (Denial-of-Service) and DDoS (Distributed Denial-of-Service) attacks have several features and key differences:
Features of DoS Attacks:...
Features of DDoS Attacks:...
Key Differences:
- **Source of Attack:** DoS attacks come from one source, while DDoS attacks come from multiple sources.
- **Volume of Traffic:** DDoS attacks typically generate a larger volume of traffic, making them more damaging.
- **Detection and Mitigation:** DDoS attacks are harder to detect and eliminate quickly due to their distributed nature, whereas DoS attacks can be easier to manage since they originate from a single source.

Overall, DDoS attacks are generally considered more severe and challenging to defend against compared to DoS attacks.
Final Answer:
{
"line_number": 83,
"analysis": "The majority of classifiers predict 'DDoS-SYN_Flood', indicating a distributed denial-of-service attack. The K-Nearest Neighbors classifier predicts 'DoS-SYN_Flood' with high certainty, but given the nature of the features and the consensus among other classifiers, 'DDoS-SYN_Flood' is deemed the more appropriate classification.",
"predicted_label_top_1": "DDoS-SYN_Flood",
"predicted_label_top_2": "DoS-SYN_Flood",
"predicted_label_top_3": "DDoS-SynonymousIP_Flood"
}

Figure 5: Additional case study on the DDoS attacks of the `IDS-Agent`. From the final analysis, it is shown that the `IDS-Agent` not only considers the predicted labels but also considers the confidence of different classifiers.

benign traffic compared with six classifiers and the majority vote method, which means our method has a lower false alarm rate, which is an important metric for intrusion detection. Figure 7 shows the confusion matrix of the majority voting classifier and `IDS-Agent`.

Table 6: The F-score of different ML classifiers on the ACI-IoT'23 dataset. For the `IDS-Agent`, we use the GPT-4o as the core LLM.

Model	RF	LR	KNN	MLP	DT	SVC	Majority Vote	IDS-Agent
Benign	0.90	0.59	0.91	0.91	0.91	0.80	0.91	0.91
DNS Flood	0.95	0.10	0.80	0.95	0.91	0.91	1.00	0.95
Dictionary Attack	1.00	0.71	0.98	0.95	1.00	0.92	1.00	1.00
ICMP Flood	1.00	0.98	0.98	1.00	0.95	0.98	0.98	0.98
OS Scan	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
Ping Sweep	0.98	0.98	0.97	0.98	0.97	0.98	1.00	1.00
Port Scan	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
SYN Flood	1.00	1.00	1.00	1.00	0.98	1.00	1.00	1.00
Slowloris	1.00	0.43	1.00	1.00	1.00	0.97	1.00	1.00
UDP Flood	0.60	0.00	0.45	0.74	0.50	0.00	0.55	0.80
Vulnerability Scan	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
Macro Avg	**0.95**	**0.71**	**0.92**	**0.96**	**0.93**	**0.87**	**0.96**	**0.97**

Table 7: The F-score of different ML classifiers on the CIC-IoT'23 dataset. For the `IDS-Agent`, we use the GPT-4o as the core LLM.

Model	DT	KNN	LR	MLP	RF	SVC	Majority Vote	IDS-Agent
BenignTraffic	0.79	0.77	0.79	0.75	0.75	0.73	0.74	0.84
DDoS-ACK_Fragmentation	0.98	0.95	0.95	0.93	0.95	0.98	0.95	1.00
DDoS-HTTP_Flood	0.58	0.53	0.24	0.79	0.68	0.38	0.69	0.70
DDoS-ICMP_Flood	0.98	0.95	0.98	0.95	1.00	1.00	1.00	1.00
DDoS-ICMP_Fragmentation	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
DDoS-PSHACK_Flood	1.00	1.00	0.98	1.00	1.00	1.00	1.00	0.95
DDoS-RSTFINFlood	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
DDoS-SYN_Flood	0.72	0.08	0.63	0.73	0.76	0.64	0.75	0.75
DDoS-SlowLoris	0.76	0.74	0.00	0.89	0.79	0.44	0.79	0.82
DDoS-SynonymousIP_Flood	0.70	0.74	0.70	0.72	0.74	0.65	0.74	0.78
DDoS-TCP_Flood	0.69	0.08	0.66	0.67	0.69	0.67	0.67	0.71
DDoS-UDP_Flood	0.71	0.23	0.77	0.75	0.71	0.74	0.71	0.74
DDoS-UDP_Fragmentation	0.97	0.95	0.95	0.97	0.97	0.97	0.97	0.95
DoS-HTTP_Flood	0.65	0.79	0.73	0.85	0.76	0.89	0.83	0.84
DoS-SYN_Flood	0.62	0.57	0.31	0.72	0.76	0.37	0.72	0.80
DoS-TCP_Flood	0.32	0.60	0.00	0.00	0.32	0.00	0.17	0.33
DoS-UDP_Flood	0.53	0.65	0.55	0.52	0.48	0.46	0.48	0.46
MITM-ArpSpoofing	0.54	0.58	0.09	0.60	0.58	0.58	0.62	0.67
Mirai-greeth_flood	0.95	0.98	0.90	0.97	0.97	1.00	0.97	0.95
Mirai-greip_flood	0.98	0.97	0.89	0.98	0.98	1.00	0.98	0.95
Mirai-udpplain	0.95	0.98	1.00	0.98	0.98	1.00	0.98	1.00
Recon-HostDiscovery	0.54	0.45	0.40	0.48	0.55	0.39	0.47	0.53
Recon-OSScan	0.30	0.30	0.08	0.00	0.15	0.10	0.17	0.15
Recon-PortScan	0.41	0.36	0.24	0.31	0.44	0.31	0.37	0.31
Macro Avg	**0.74**	**0.68**	**0.62**	**0.73**	**0.75**	**0.68**	**0.70**	**0.75**

Thought: I have obtained the predictions from multiple classifiers. Here are the results:
1. **Random Forest**: - **MITM-ArpSpoofing**: 0.4200 - BenignTraffic: 0.2500 - Recon-PortScan: 0.1300
2. **K-Nearest Neighbors**: - **Recon-PortScan**: 0.4000 - MITM-ArpSpoofing: 0.4000 - BenignTraffic: 0.2000
3. **Logistic Regression**: - **BenignTraffic**: 0.3689 - MITM-ArpSpoofing: 0.3105 - DoS-UDP_Flood: 0.1391
4. **MLP**: - **MITM-ArpSpoofing**: 0.4019 - BenignTraffic: 0.3296 - Recon-HostDiscovery: 0.1095
5. **Support Vector Classifier**: - **BenignTraffic**: 0.6327 - MITM-ArpSpoofing: 0.2328 - Recon-HostDiscovery: 0.0855
6. **Decision Tree**: - **Recon-OSScan**: 1.0000 - Recon-PortScan: 0.0000 - DDoS-UDP_Flood: 0.0000
Now, I will analyze the results to determine the final classification. The predictions show a mix of benign and attack classifications, with some classifiers leaning towards "MITM-ArpSpoofing" and others towards "BenignTraffic." The Decision Tree classifier is particularly confident in classifying it as "Recon-OSScan."To resolve the discrepancies, I will search for more information regarding the features of the attacks and benign traffic to make a more informed decision.
Action: knowledge_retriever
Action Input: {"query": "what is the features of MITM-ArpSpoofing and how to detect it"
Observation: The features of MITM-ARP-Spoofing include: 1. **Interception of Communication**: The attacker positions themselves between two network devices, allowing them to intercept and potentially modify the data being transmitted.
2. **ARP Cache Poisoning**: The attacker sends falsified ARP messages to both devices, associating their MAC address with the IP address of the other device, thus redirecting traffic through their machine....
Final Answer:
{
"line_number": 83,
"analysis": "The traffic features indicate a potential MITM-ArpSpoofing attack based on the predictions from multiple classifiers, particularly Random Forest and MLP. The Decision Tree classifier also suggests a reconnaissance attack, but the overall consensus leans towards MITM.",
"predicted_label_top_1": "MITM-ArpSpoofing",
"predicted_label_top_2": "BenignTraffic",
"predicted_label_top_3": "Recon-PortScan",
}

Figure 6: Additional case study on the MITM attacks of the IDS-Agent. From the final analysis, it is shown that the IDS-Agent not only considers the top 1 prediction but also considers the second and third predictions.

A.4 Details for Sensitivity Customization

We adjust the detection sensitivity by prompting the core LLM with "Summarize the classification with {sensitivity} sensitivity, {sensitivity details}". Here, 'sensitivity details' will be 'discover the attack as the priority', 'balance the false alarm rate and the missing alarm rate', and 'do not alert unless you are very sure', for 'sensitivity' being 'aggressive', 'balanced' and 'conservative'. The detection performances of IDS-Agent for different detection sensitivities are shown in Table 8. It is shown that the 'Aggressive' command achieves a higher recall on the attacks while the 'Conservative' command achieves a higher recall on the benign examples. The classification results, detailed in Table 8 of the appendix, show that the IDS-Agent effectively follows these sensitivity instructions without requiring expert intervention or additional tuning.

Table 8: The classification results of different detection sensitivities.

Sensitivity	Aggressive			Balance			Conservative		
Metrics	Precision	Recall	F1-score	Precision	Recall	F1-score	Precision	Recall	F1-score
Benign	0.96	0.90	0.92	0.87	0.96	0.91	0.60	0.98	0.75
DNS Flood	0.91	1.00	0.95	0.91	1.00	0.95	0.94	0.80	0.86
Dictionary Attack	0.91	1.00	0.95	1.00	1.00	1.00	1.00	0.65	0.79
ICMP Flood	0.95	1.00	0.89	0.95	1.00	0.98	0.95	1.00	0.98
OS Scan	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
Ping Sweep	0.95	1.00	0.98	1.00	1.00	1.00	1.00	1.00	1.00
Port Scan	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
SYN Flood	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
Slowloris	0.95	1.00	0.98	1.00	1.00	1.00	1.00	0.40	0.57
UDP Flood	1.00	0.80	0.89	1.00	0.53	0.69	1.00	0.47	0.64
Vulnerability Scan	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
Macro Avg	**0.97**	**0.97**	**0.97**	**0.98**	**0.95**	**0.96**	**0.95**	**0.85**	**0.87**

A.5 The Zero-day Attack Detection Details

We prompt GPT-4o to classify an example as an unknown attack if multiple classifiers output low confidence for their top predictions or if there are conflicting predictions among different classifiers.

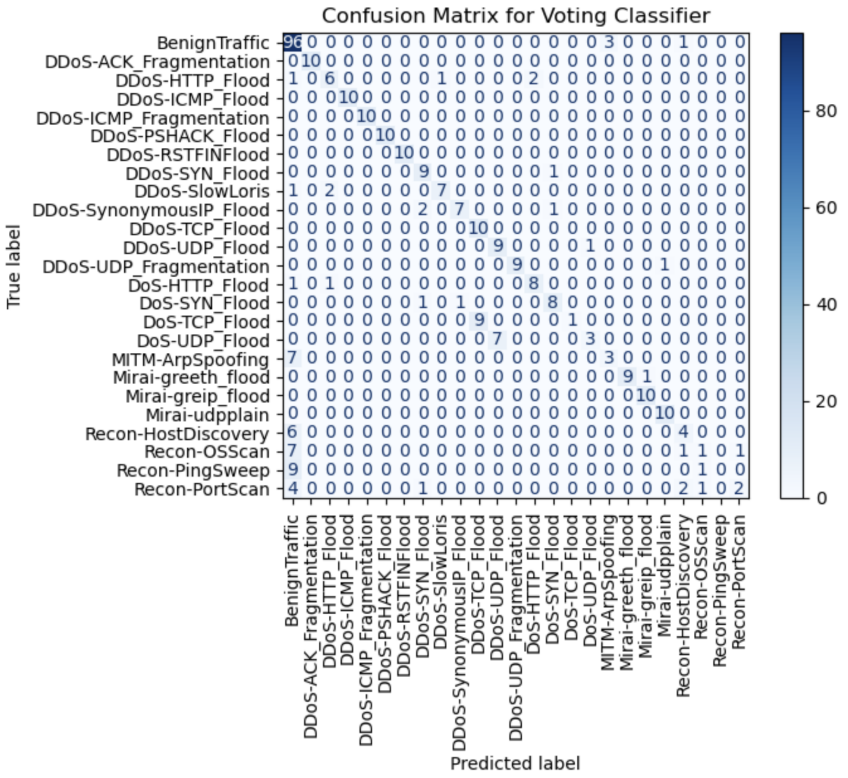
This is based on our observation that, for unknown attacks, machine learning models typically exhibit relatively low confidence levels, as shown in Figure 8. Specifically, we instruct the LLM to consider an example as a potential unknown attack if more than two models have low confidence (e.g., below a threshold of 0.7). Moreover, if more than two models have low confidence or if different models produce significantly divergent predictions, we direct `IDS-Agent` to search the knowledge base for characteristics of the most probable predicted attacks. If the traffic features do not match these attack characteristics, we confirm the example as an unknown attack and provide this as the final output.

A.6 The Influence of Hyperparameters

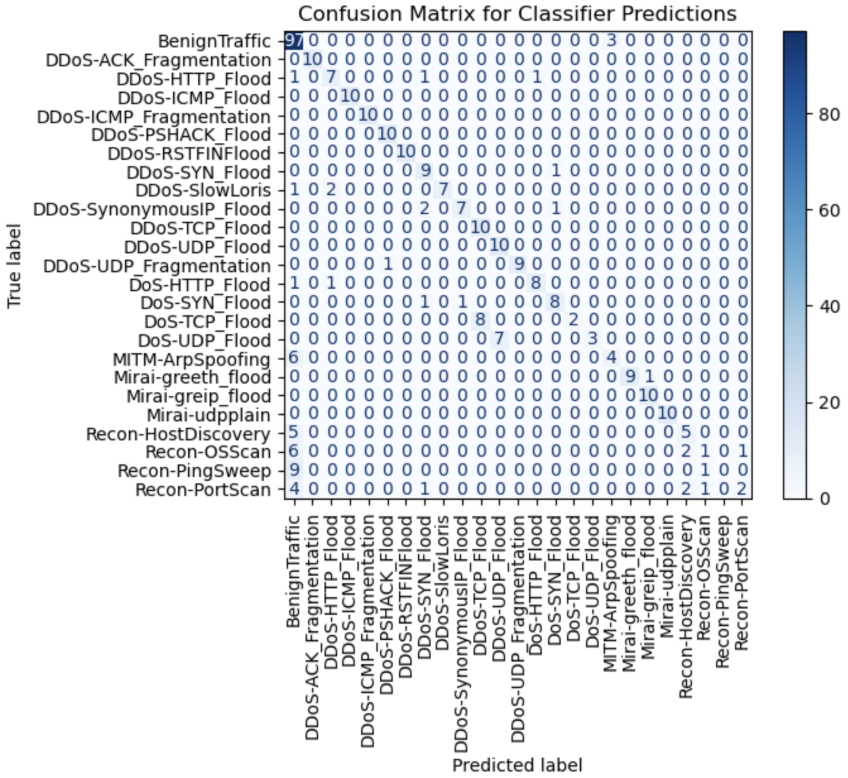
To assess the influence of different values of λ_1 and λ_2 in Eq. 1, we conducted experiments by varying these parameters and measuring the impact on retrieval effectiveness and overall classification performance. Table 9 summarizes the results of our experiments. The experimental results indicate that both recency and content similarity are crucial for effective LTM retrieval. A balanced approach, where λ_1 and λ_2 are equal, provides the best performance, suggesting that the agent benefits from considering both embedding similarity and recency.

Table 9: Performance metrics for different values of λ_1 and λ_2.

λ_1	λ_2	Accuracy (%)	Precision (%)	Recall (%)
0.1	0.9	97.2	97.2	96.5
0.5	0.5	**98.0**	**98.2**	**97.2**
0.9	0.1	97.3	97.1	96.1



(a) Confusion matrix of majority voting classifer



(b) Confusion matrix of `IDS-Agent`

Figure 7: The confusion matrix of majority voting classifier and `IDS-Agent` on the CIC-IoT'23 dataset.

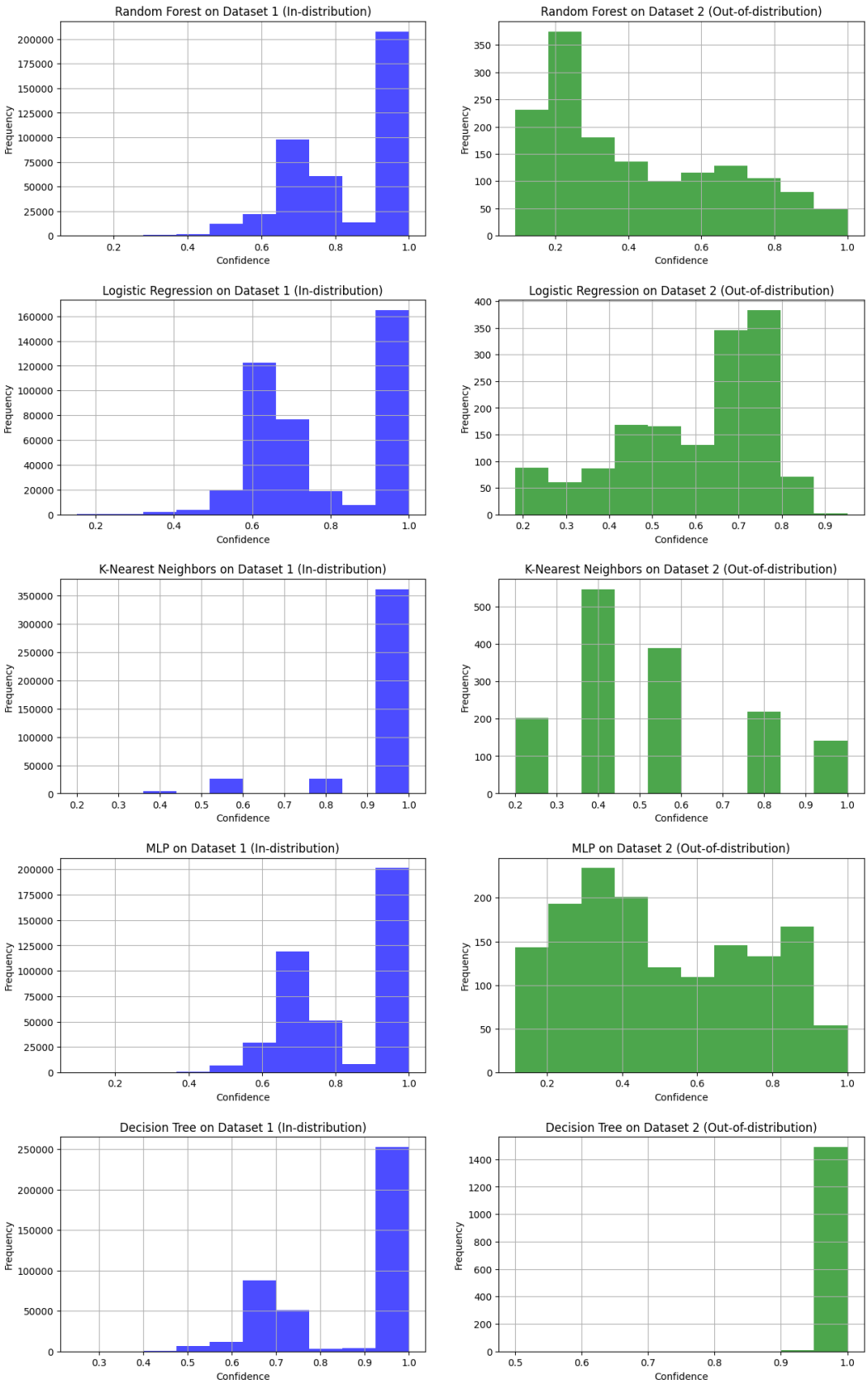


Figure 8: The confidence distributions of difference classifiers on the in-distribution dataset and out-of-distribution dataset.