Advances in Technology Innovation, vol. 11, no. 4, 2026, pp. 345-362

Automated Mining and Analysis of Packet Exchange Characteristics through Communication Packet Behavior

Shao-Chien Lin, Tse-Chuan Hsu*

Department of Computer Science and Information Management, Soochow University, Taipei, Taiwan, ROC

Received 13 August 2025; revised 23 July 2026; accepted 28 July 2026

DOI: https://doi.org/10.46604/aiti.2026.15564

Abstract

Internet of things (IoT**)** devices often communicate with external services without clear visibility into their geographic destinations. This study aims to develop a gateway-based framework for predicting the region associated with remote endpoints from periodically collected connection records. Public endpoints are identified, enriched with MaxMind GeoIP2 attributes, and deduplicated into 4,320 unique instances across 18 region labels. Four classifiers are evaluated across four feature configurations to assess how prediction performance changes as GeoIP dependencies are reduced. Using only the transformed IP address, the Random Forest model achieved 92.94% accuracy and 85.24% macro F1 on the held-out test set without requiring GeoIP databases during inference. Temporal validation on endpoints first observed after training yielded 90.91% accuracy and 75.42% macro F1. These results demonstrate that part of the GeoIP-derived region mapping can be compressed into a compact deployable model, supporting communication transparency in IoT environments.

Keywords: internet of things, cybersecurity, traffic analysis, machine learning, endpoint region prediction

1. Introduction

Internet of things (IoT) devices have experienced rapid adoption in both residential and enterprise environments, driven by advancements in wireless communication protocols, embedded computing, and cloud-based services. Their adoption spans diverse applications, including smart homes, industrial automation, healthcare monitoring, and environmental sensing. Dinh and Lim (2016)[1] demonstrated that IEEE 802.15.4-based IoT solutions can support scalable and cost-effective smart home deployments through communication technologies characterized by low power consumption and short transmission ranges. Such studies illustrate the broader trend of IoT expansion, in which heterogeneous devices and communication standards are increasingly integrated into everyday infrastructure, underscoring the importance of continued research into their deployment, performance, and management.

IoT devices are increasingly connected within residential and enterprise networks. Many devices automatically establish persistent connections with remote servers during regular operation, often without the user's knowledge. This opaque nature of communication flows makes tracking the direction, frequency, and nature of transmitted data difficult, thereby reducing network visibility. This lack of visibility presents potential network security risks, as malicious or compromised devices could exploit these channels for unauthorized activity. Improving communication transparency and promptly detecting anomalous or high-risk behavior within IoT networks has become a key research challenge.

In network traffic monitoring, Wireshark is a widely used packet analysis tool for cybersecurity and network troubleshooting. Wireshark primarily relies on a batch processing approach. Users must capture the network traffic and manually load PCAP files for filtering and analysis. While this method is adequate for small-scale packet analysis, it becomes

* Corresponding author. E-mail address: tchsu@scu.edu.tw

inefficient as the traffic volume grows with the increasing number of IoT devices. To address these limitations, researchers in recent years have turned their attention to machine learning-based approaches, which aim to automate anomaly detection by learning from historical traffic behavior, enhancing both the accuracy and real-time capability of network monitoring.

This study proposes a traffic-monitoring method that integrates gateway-level connection analysis, GeoIP-based attribute enrichment, and machine learning to improve the transparency of device communication flows. Connection records are collected at a network gateway, and the externally facing remote endpoint of each connection is identified as the main object of analysis. The geographic and organizational attributes of these remote endpoints are obtained from the MaxMind GeoIP2 databases, and the classifiers are trained to predict the geographic region associated with each remote endpoint.

Through this approach, the regions with which devices communicate, including destinations that may otherwise be hidden or unknown to users, can be inferred from observed traffic. The study further analyzes how different combinations of input features affect prediction performance and how many external GeoIP databases are required during inference under different deployment settings. This analysis evaluates whether the model reduces dependence on fine-grained GeoIP lookups while maintaining predictive capability, which is particularly for lightweight deployment on resource-constrained edge devices.

The primary objective of this study is to develop an accurate model for predicting the geographic region of remote endpoints, thereby enhancing the transparency of device communications within a network. By associating the predicted regions of remote endpoints with specific device communication profiles, network administrators can gain greater visibility into abnormal access patterns and cross-border traffic. Because the trained model is exported in a compact format, it is suitable for integration with network infrastructure components such as routers or gateways. Region-level predictions may also support future security-oriented applications, such as flagging or restricting traffic directed toward regions considered high-risk. In this way, the proposed approach aims to contribute to more informed and proactive network security management.

2. Related work

As the risk of network devices transmitting data autonomously is increasing, according to the study by Wu et al. [2], 24 of the 32 IoT devices evaluated in their experimental setting were observed to transmit data to multiple destinations. Remarkably, one device communicated with as many as 24 distinct endpoints, whereas the remaining devices each transmitted data to a single destination. The accurate identification of packet transmission destinations was primarily enabled by the IoT Network Analyzer, a Python-based PCAP analysis framework developed by the authors. This tool supports efficient batch processing and in-depth inspection of IoT traffic traces and incorporates the IP2Location-Lite API to resolve the geographical locations associated with packet IP addresses.

In addition to the data transmitted by devices, a study conducted by Andrews et al. [3] demonstrated the setup of an IoT laboratory where all devices were connected to a network switch. Network traffic was captured on a computer mirrored to the switch's SPAN port. PCAP data enables detailed device behavior analysis, revealing that network traffic patterns change significantly depending on whether the device is idle, actively interacting, or undergoing a firmware update. By combining these changes with high-level device information compiled by the authors, the study provides more accurate insights for device identification and version tracking.

Wireshark is one of the most widely used tools for packet analysis. In the study by Soepeno [4], Wireshark was demonstrated to effectively capture and analyze network packets, supporting not only troubleshooting tasks but also in-depth security analysis. Furthermore, numerous studies have confirmed the practical effectiveness of Wireshark in real-world applications. For instance, the research conducted by Alyami et al. [5], Kim et al. [6], and Rizal et al. [7] all employed Wireshark as a packet analysis tool. Its robust traffic monitoring and data inspection capabilities have provided critical support for network behavior analysis and cybersecurity research.

IP geolocation techniques are frequently used in academic research and practical applications to determine the geographic locations associated with IP addresses. This method helps analyze packet sources and identify potential threat zones and is widely used in content distribution restrictions, fraud detection, and cybersecurity incident forensics. Those based on precompiled geolocation databases are particularly popular among the various IP geolocation methods. These database-based techniques use data aggregated from multiple sources (including WHOIS records, network infrastructure topology, ISP registration information, and user-provided information) to associate IP address ranges with geographic locations. These methods are particularly effective when direct cooperation from end-user devices (such as providing GPS coordinates or Wi-Fi triangulation data) is unavailable or impractical. This non-intrusive monitoring method allows for scalable and non-invasive location estimation based on observed IP addresses.

For the analysis of observed IP addresses and non-invasive location estimation, a widely used commercial geolocation database, GeoIP2, is required. Developed by MaxMind, this database is employed in various Internet services to provide network location analysis. However, its accuracy has been questioned. Therefore, Komosny et al. [8] empirically evaluated eight major IP geolocation databases, including GeoIP2, using a high-confidence ground truth dataset established by precise GPS coordinates. Their results show that although most databases have an accuracy rate of more than 90% at the national level, their performance drops significantly at the regional and city levels - for example, GeoIP2 only correctly identifies about 30% of locations at the city level. The research findings show that the location accuracy of the database is related to the population and area of the city. The main reason is that the data volume of blocks with high usage density is larger and more likely to be recorded, so the data density and network activity level significantly affect the performance of the geolocation database. Commercial IP geolocation repositories are therefore useful for broad-level source identification. Still, caution should be exercised when applying them to fine-grained geolocation tasks, as their accuracy and reliability may be limited.

Machine learning has been extensively applied across various domains to address various predictive tasks, making rigorous evaluation of model performance increasingly important. Chaudhari et al. [9] emphasized the importance of using consistent and comprehensive evaluation metrics, including precision, recall, F1-score, and accuracy, which are calculated based on confusion matrices, to assess the effectiveness of classification models. The study evaluated multiple machine learning models under identical dataset conditions, ensuring fairness and reliability in performance comparison. This standardized evaluation methodology provides a valuable reference for validating classification models in other machine learning applications where reproducibility and performance transparency are essential.

Table 1 provides a structured comparison between the present study and several representative studies that address related aspects of network traffic visibility, IoT communication analysis, device identification, and IP geolocation. Previous research has mainly concentrated on examining captured packets, identifying IoT devices from traffic patterns, analyzing the destinations contacted by connected devices, or evaluating the geographic accuracy of existing IP geolocation databases. Although these studies offer important foundations for understanding device communication behavior and network destination information, their objectives differ from those of the present work.

This study formulates remote-endpoint region prediction as a supervised machine-learning classification task and investigates whether the geographic mapping represented by GeoIP databases can be partially retained in a compact predictive model. In addition to comparing multiple classifiers and feature configurations, the evaluation also considers endpoint-level duplication leakage, class imbalance, temporal generalization to newly observed endpoints, dependence on external GeoIP databases, and the deployment costs associated with model size and inference latency. Since the compared studies employed different datasets, collection environments, feature representations, and evaluation objectives, their reported numerical results are not directly comparable. The purpose of Table 1 is therefore to clarify the methodological differences among these studies and to show how present work extends existing research by combining gateway-level connection collection, endpoint-level preprocessing, comparison, GeoIP-dependency analysis, and temporal evaluation within region-prediction framework.

Table 1 Comparison with representative related studies

Study	Data acquisition or source	Primary objective	Main information used	Evaluation focus	Distinction from the present study
Wu et al. [2]	PCAP traffic analyzed using IoT Network Analyzer	Analyze IoT communication destinations	Packet records, endpoint IP addresses, and IP2Location-Lite	Visibility of external communication destinations	Focuses on destination analysis rather than machine-learning-based region prediction
Andrews et al. [3]	Passive PCAP capture through a mirrored SPAN port	IoT device identification and version tracking	Traffic behavior under idle, active, and firmware-update states	Device behavior and identification	Uses packet-behavior patterns but does not predict remote-endpoint regions
Komosny et al. [8]	High-confidence GPS ground truth and eight IP-geolocation databases	Evaluate IP-geolocation database accuracy	IP-geolocation database outputs and physical ground truth	Country-, region-, and city-level geolocation accuracy	Evaluates existing databases rather than compressing their region mapping into a deployable classifier
Proposed study	Periodic gateway conntrack records and live PyShark capture	Predict the region associated with remote endpoints	IP address and progressively reduced GeoIP-derived attributes	Four classifiers, four feature sets, held-out testing, temporal validation, and deployment cost	Examines whether endpoint-region mapping can be retained while reducing runtime GeoIP dependency

3. Methodology

The proposed methodology consists of a five-stage pipeline for gateway-based packet sniffing and connection analysis: packet sniffing, packet attribute extraction, data preprocessing, model training, and remote-endpoint region prediction, as illustrated in Fig. 1. In this pipeline, connection records are collected at the network gateway to provide a centralized view of the communication activities generated by devices within the monitored network. The externally facing remote endpoints are then identified and enriched with geographic and organizational information obtained from GeoIP2 databases. After invalid, private, or otherwise unsuitable records are removed, the remaining attributes are cleaned, encoded, and transformed into model-compatible numerical representations. These processed data are subsequently used to train and evaluate a machine-learning classifier, which is finally applied to newly observed connection records to predict the geographic region associated with each remote endpoint and provide a more interpretable view of external network communications.



Fig. 1 Flowchart of the proposed automated packet exchange characteristics mining and analysis framework

A key consideration in this methodology is the periodic collection of connection-tracking records. Since the same long-lived connection may appear in multiple snapshots, the preprocessing and evaluation procedures are designed to reduce the risk of duplicated-record leakage and inflated performance estimates. In addition, the experiment evaluates multiple feature combinations to examine how model performance changes when GeoIP-derived input features are progressively removed.

3.1 Packet sniff

Packet capture is a fundamental component of network monitoring and traffic analysis. By examining the structure and metadata of network packets, administrators and researchers can gain insight into packet-exchange behavior, communication destinations, and potential operational or security issues. As noted by Asrodia et al. [10], packet sniffing provides an effective basis for analyzing network traffic and understanding communication activities within a network environment. In contemporary Internet of Things and wireless network environments, such observation is particularly important because connected devices often communicate with external services automatically and continuously.

In the proposed method, packet acquisition serves two purposes. The first is to collect gateway-level connection records for constructing the training corpus. The second is to capture live traffic at the monitoring point during inference, so that newly observed communication records can be transformed into the same feature representation used during model training.

For model training, network traffic is collected at the gateway rather than at an individual host. In the experimental environment, a NETGEAR R7000 router running OpenWrt firmware is configured as the gateway device. The router periodically exports its Linux netfilter connection-tracking table by means of a scheduled cron task. At each hourly interval, the active connection entries are dumped and serialized into JSON format. Each exported record represents one observed connection-tracking entry and contains the timestamp, network-layer protocol, transport-layer protocol, source and destination IP addresses, source and destination MAC addresses, and source and destination ports. Collecting data at the gateway provides a centralized vantage point that aggregates the outbound and inbound communication records of downstream devices. This setting represents a typical home or small-office deployment in which a single egress point mediates external communication.

The data used in this study were collected at regular intervals over an extended period. The corpus was exported as two cumulative snapshots. The first snapshot covered the period from March 7 to July 16, 2025, while the second extended the observation period to August 18, 2025. This cumulative structure subsequently supported the temporal hold-out validation described in Section 3.4, in which the model was evaluated using endpoints first observed after the training period.

Because the conntrack table is sampled periodically rather than captured as a complete packet-by-packet trace, the resulting records are treated as connection-level observations rather than full packet data. Accordingly, payload content and per-packet length information are excluded from the analysis. Instead, the study relies on connection-level metadata, specifically the IP address of the remote endpoint for geographic enrichment and the MAC address for vendor identification, along with protocol and transport-type fields to restrict the dataset to IPv4 TCP connections. Port numbers are also excluded from the model inputs.

A characteristic of conntrack-based sampling is that a long-lived or frequently re-established connection may appear in multiple consecutive snapshots. If such repeated observations are directly used in model training and testing, persistent connections may be over-represented, and near-identical records may inflate the estimated classification performance. This issue is explicitly addressed during data preprocessing by reducing repeated observations to endpoint-level instances, as described in Section 3.3.

For inference, live traffic is acquired using the PyShark library in a Python environment. PyShark is a wrapper for the Wireshark network-analysis tool and provides a structured interface for real-time packet capture and the analysis of pre-captured packet files. The monitoring host is placed at an observation point where target traffic can be captured, such as the gateway interface or a mirrored network interface. From each observed packet, the system extracts the endpoint identifiers required for subsequent enrichment and prediction, primarily the IP address and MAC address fields. These identifiers are then matched against the GeoIP and MAC OUI reference databases, as described in Section 3.2, and transformed into the feature representation required by the trained model.

Although raw packets contain additional information, such as payload data, packet length, and detailed protocol-specific fields, this study retains only the attributes required for endpoint identification, geographic enrichment, vendor identification, and region prediction. This design reduces storage and processing overhead, simplifies integration with external reference databases, and maintains consistency between the gateway-level training records and the live inference records.

3.2 Packet attribute extraction

After packet capture, each record is processed to identify the associated externally facing endpoint. Since geographic enrichment is meaningful only for routable Internet addresses, both source and destination IP addresses are classified as public or non-public according to standard IP address semantics. A record is retained only when exactly one endpoint is public. The public address is defined as the remote endpoint, whereas the non-public address and its associated MAC address are treated as the local device. Records containing no public endpoint or two public endpoints are excluded to avoid ambiguous endpoint attribution.

The remote endpoint is then enriched using the MaxMind GeoIP2 databases. Previous studies have shown that IP geolocation databases such as GeoIP and IP2Location-Lite provide comparable accuracy for IP-based geographic inference, although their reliability may vary across geographic granularity and network environments [11]. Considering its offline availability and applicability in academic experiments, this study adopts MaxMind GeoIP2 for geographic and organizational enrichment. Three GeoIP2 datasets are queried, including the Country database, which provides the country name; the City database, which provides the city name and the subdivision field used as the region label; and the ASN database, which provides the autonomous-system organization associated with the remote IP address. These enriched attributes provide the semantic basis for transforming raw endpoint identifiers into model-compatible features and labels. To reduce repeated database access, GeoIP lookups are performed once for each distinct remote address and cached for subsequent records. If an address cannot be resolved, it is classified according to IP address semantics or marked as Unknown or Invalid.

In parallel, the local device MAC address is mapped to its hardware vendor using the organizationally unique identifier (OUI), which corresponds to the first three bytes of a MAC address. Although private OUIs may not appear in public registries, prior work has reported no evidence that major manufacturers widely rely on them [12]. Each captured MAC address is normalized to a consistent format, after which its OUI portion is extracted and matched against a MAC-vendor reference database. Matched records are assigned the corresponding vendor name, whereas unmatched records are marked as Unknown.

This device-level attribute provides supplementary context for identifying the local device associated with each observed connection and supports subsequent device-oriented traffic analysis. Although the geographic prediction task is centered on the remote endpoint, retaining local device information helps preserve the relationship between device identity and external communication behavior. The resulting geographic, organizational, and device-level attributes are serialized into JSON format for subsequent preprocessing and model training.

3.3 Data preprocessing

The enriched records are loaded using the pandas library and preprocessed before model training. Records associated with non-routable or internally scoped addresses, including Local, Multicast, Loopback, Reserved, and Unspecified addresses, are excluded to constrain the analysis to externally reachable destinations. The dataset is further restricted to IPv4 TCP connections to maintain a consistent protocol scope for the supervised classification task.

The unit of analysis is defined as a unique remote endpoint rather than an individual snapshot record. Because the connection-tracking table is exported periodically, the same remote endpoint may appear repeatedly across consecutive snapshots, especially for long-lived or frequently re-established connections. Retaining all snapshot rows would over-represent persistent endpoints and could inflate model performance if near-identical observations of the same endpoint were divided between the training and testing sets. To mitigate this duplicated-record leakage, repeated records are deduplicated by unique remote endpoint before model training and evaluation.

After deduplication, each retained remote endpoint is represented as one instance in the learning task. Region labels represented by fewer than 20 endpoint-level instances are subsequently excluded, since such sparsely populated classes may lead to unstable per-class evaluation and unreliable cross-validation results. This filtering step reduces the influence of extremely rare classes and allows the classifier to be evaluated on region labels with sufficient sample support.

Finally, the categorical and address fields are transformed into numeric representations. Each IPv4 address is converted to its corresponding integer value. Categorical fields, including Country name, City name, AS organization, and the target Region name, are encoded into integer labels using a label encoder. To preserve the interpretability of the prediction output, the label encoder fitted to the Region field is serialized and stored, allowing numeric predictions to be mapped back to their original region names during inference.

3.4 Training model

After data preprocessing, XGBoost is adopted as the classification model. XGBoost is a gradient-boosted decision-tree algorithm that has been widely applied in IoT device identification, network-traffic analysis, and cybersecurity-related classification tasks, for example, Ness et al. [13], Chen et al. [14], and Cruz et al. [15]. Its tree-based structure suits heterogeneous tabular features, including numeric addresses and encoded categorical attributes, and it provides strong predictive performance while remaining more deployable than many computationally intensive deep learning architectures.

The prediction task is formulated as a multi-class classification problem. Each endpoint-level instance is assigned to one regional label, and the classifier is trained to predict the Region associated with the remote endpoint. The input variables are derived from the transformed IPv4 address and the encoded geographic or organizational attributes obtained during packet attribute extraction. The target variable is the Region label. Following the preprocessing procedure described in Section 3.3, each remote endpoint is represented only once; therefore, the training and testing partitions do not contain duplicated observations of the same endpoint. This design reduces the risk of inflated evaluation results caused by repeated connection-tracking snapshots of the same remote endpoint.

The dataset is divided into training and testing subsets using an 80:20 ratio. To preserve the distribution of regional labels, stratified sampling is applied during the split. Model construction and hyperparameter tuning are performed using the GridSearchCV module in the scikit-learn library. Grid search evaluates multiple candidate hyperparameter combinations through stratified k-fold cross-validation on the training subset only and selects the configuration that achieves the highest mean validation accuracy. Classification accuracy is used as the primary optimization metric because the task is formulated as a multi-class region prediction problem. The complete model-selection and evaluation procedure is illustrated in Fig. 2.



Fig. 2 Model selection and evaluation procedure using GridSearchCV

After the optimal hyperparameter configuration is identified, the final XGBoost classifier is retrained on the entire training subset and then evaluated on the held-out testing subset. The testing subset is not involved in either model fitting or hyperparameter selection. This procedure separates model selection from final performance evaluation and reduces the risk of overestimating model performance based on a single fitted configuration. Although accuracy is used for hyperparameter selection, per-class precision, recall, and F1-score are also reported to account for differences among regional labels.

To further examine the dependency of the model on GeoIP-derived information, a feature-combination analysis is conducted. The purpose of this analysis is not only to compare predictive performance, but also to identify how many external GeoIP databases would still be required during inference if the trained model were deployed on an edge device. Four nested feature sets are evaluated, ranging from a feature-rich setting that uses multiple GeoIP-derived attributes to a lightweight setting that relies only on the transformed IP address. The evaluated feature configurations are summarized in Table 2.

Table 2 Feature-combination settings for evaluating dependency on GeoIP-derived information

Set	Input features	GeoIP databases required at inference
A	Country, City, AS organization, IP	Country, City, ASN
B	Country, AS organization, IP	Country, ASN
C	AS organization, IP	ASN
D	IP only	none

Set A includes Country, City, AS organization, and IP address. Since the Region label is derived from the subdivision field of the GeoIP City database, this setting is treated as an upper-bound reference rather than a leakage-free deployment configuration. It indicates the performance that can be achieved when fine-grained GeoIP-derived geographic information is available at inference time. Set B removes the City feature while retaining Country and AS organization, thereby evaluating whether city-level information is necessary for region prediction. Set C further removes the Country feature and retains only AS organization and IP address, reducing dependency on direct geographic attributes. Set D uses only the transformed IP address and requires no GeoIP lookup during inference. Comparing these settings allows the study to evaluate how predictive performance changes as geographic input features are progressively removed.

This feature-combination design directly addresses the concern that the model may learn mappings among GeoIP-derived fields rather than generalizable traffic characteristics. In particular, the comparison between Set A and Set B examines the influence of the City feature, while the comparison among Sets B, C, and D evaluates whether the model can maintain predictive ability when explicit geographic attributes are reduced or removed. Therefore, the analysis provides evidence for the extent to which the trained model can reduce dependency on fine-grained GeoIP lookup during edge-side inference.

After training, model performance was evaluated on the held-out test set using overall accuracy and a per-class classification report, including precision, recall, and F1-score, as well as the macro-averaged F1-score. These metrics provide both aggregate and class-level assessments of predictive performance, which is particularly important for multi-class classification tasks with potentially imbalanced class distributions. Unlike accuracy and weighted averages, which may be dominated by frequent classes, the macro-averaged F1-score assigns equal weight to each region and is therefore less susceptible to inflation by the dominant class. A majority-class baseline was also included for reference using the most_frequent strategy of scikit-learn's DummyClassifier. This baseline performs no learning and always predicts the most frequent region label observed in the training set. Consequently, its accuracy corresponds to the majority-class proportion, while its macro-averaged F1-score is expected to remain near zero. The baseline serves as a minimum no-skill reference point; the trained classifier can be considered to have learned meaningful patterns beyond the dominant class distribution only if it substantially outperforms this baseline, particularly in terms of macro-averaged F1-score.

To further assess the robustness of the evaluation, two complementary analyses are conducted. The first examines whether the observed performance is specific to the choice of classifier, whereas the second evaluates whether the trained model can generalize to remote endpoints observed during a later collection period.

To determine whether the observed performance depends on the choice of classifier, four classifiers are evaluated under each of the four feature configurations defined in Table 2. In addition to XGBoost, the comparison includes a Random Forest classifier, a support vector machine (SVM) with a radial basis function kernel, and multinomial logistic regression. For Random Forest, the number of trees, maximum tree depth, and minimum number of samples per leaf are included in the hyperparameter search. For the SVM, the regularization strength and kernel coefficient gamma are tuned. The regularization strength is also tuned for multinomial logistic regression. Each classifier and feature-set combination is tuned independently using GridSearchCV with stratified five-fold cross-validation conducted exclusively on the training subset. The selected model is then evaluated on the same held-out testing subset.

Because support vector machines and logistic regression are sensitive to differences in feature scale, their inputs are standardized within a scikit-learn Pipeline. This procedure ensures that the scaler is fitted only to the training portion of each cross-validation fold and prevents information from the corresponding validation portion from influencing preprocessing. Standardization is not applied to XGBoost or Random Forest because tree-based classifiers do not require feature scaling. This complete comparison allows the effects of classifier choice and GeoIP-derived feature availability to be examined under a consistent evaluation protocol.

In addition to the stratified random split, a temporal hold-out evaluation is conducted to examine generalization to remote endpoints observed during a later collection period. The first cumulative snapshot, covering March 7 through July 16, 2025, defines the training window, whereas the second snapshot extends the collection period to August 18, 2025. Rather than applying a direct date-based split to individual records, the temporal partition is constructed at the endpoint level. Specifically, the validation candidates are obtained from the set difference between the unique remote endpoints contained in the second snapshot and those already present in the first snapshot. Consequently, the validation set contains only endpoints that were not observed during the training window, and the two partitions are disjoint at the endpoint level.

The training window contains 4,011 unique remote endpoints, of which 3,770 remain after applying the preprocessing procedure described in Section 3.3. The subsequent collection period contains 597 endpoints that were not observed in the first snapshot. Among these endpoints, 550, corresponding to 92.1%, fall within the 18-label space established from the training data and are retained for the closed-set temporal evaluation. These endpoints collectively cover 17 of the 18 region labels. The remaining 47 endpoints carry region labels outside that space: 39 belong to labels that were observed during the training window but were excluded by the minimum-support threshold defined in Section 3.3, whereas 8 belong to labels that were not observed during the training window. Both groups therefore fall outside the closed-set label space and are excluded from the reported performance metrics.

All feature encoders used in the temporal evaluation are fitted exclusively on data from the training window. Previously unseen categorical input values in fields such as Country, City, and AS organization are assigned the reserved out-of-vocabulary code in accordance with the inference procedure described in Section 3.5. This mechanism applies only to unseen input categories and is distinct from the treatment of target Region labels outside the 18-label classification space. Hyperparameters are selected through GridSearchCV with shuffled stratified five-fold cross-validation conducted within the training window only. The selected model is then retrained using the complete preprocessed training-window dataset and evaluated once on the retained temporal validation set. The results of the classifier comparison and temporal hold-out evaluation are reported in Section 4.

Finally, each tuned classifier is serialized using the format appropriate to its implementation. XGBoost models are exported in the library's native JSON format, while Random Forest, SVM, and logistic regression models are serialized using joblib, which also preserves the fitted preprocessing objects contained in the Pipeline. Because the two tree ensemble models use different persistence formats, their artifact sizes are measured directly rather than compared solely based on serialization

format. Initialization time is defined as the time required to load a serialized model or open a memory-mapped GeoLite2 database. Latency for each record is measured using the same held-out endpoints employed in the accuracy evaluation. A warm-up pass is performed before timing to exclude initialization during the first invocation and the initial cost of loading database pages, after which individual predictions and database lookups are timed separately. Median latency is reported because memory-mapped database lookups exhibit a strongly right-skewed distribution influenced by occasional page faults. The reported latency includes model inference or database lookup only and excludes feature encoding, which is common to all configurations. Inference is performed using a single thread and one endpoint at a time to reflect the intended online deployment setting rather than batch scoring. The resulting measurements are reported in Section 4.

3.5 Packet region prediction

During inference, records obtained from the live traffic capture described in Section 3.1 are processed using the same endpoint-identification and feature-construction procedures applied during model training. Records associated with non-routable or internally scoped addresses are excluded, and the externally facing public IP address is retained as the remote endpoint. The IPv4 address of each retained endpoint is then converted into its integer representation. Depending on the selected feature configuration defined in Table 2, the required geographic and organizational attributes are retrieved to construct the model input. The selected feature configuration determines whether Country, City, and AS organization attributes are retrieved; Set D requires no GeoIP database lookup.

All preprocessing objects are fitted exclusively on the training data and stored for subsequent inference. Categorical input fields required by the selected feature configuration, including Country, City, and AS organization, are encoded using the same encoders employed during model training. These encoders are not refitted on newly observed records, because doing so would alter the categorical mappings associated with the trained classifier. When an input category was not observed during training, it is assigned the reserved out-of-vocabulary code. This mechanism applies only to unseen input values and is distinct from the treatment of target Region labels outside the predefined closed-set label space.

The inference engine then loads the serialized classifier associated with the selected model and feature configuration. Tree-based classifiers, including Random Forest, receive the constructed numerical feature vector directly. For the support vector machine and multinomial logistic regression, feature standardization and classification are retained within the fitted scikit-learn Pipeline, ensuring that the same scaling transformation learned from the training data is applied during inference. For XGBoost, the feature vector is converted into the DMatrix representation required by the XGBoost library before prediction. DMatrix is therefore treated as an implementation-specific component of the XGBoost branch rather than as a general requirement of the inference procedure.

To maintain consistency between model training and deployment, each serialized model is associated with metadata specifying the classifier type, feature configuration, expected feature order, required preprocessing objects, and target-label encoder. Before prediction, the inference system verifies that the constructed feature vector is consistent with the metadata of the loaded model. The selected classifier subsequently outputs a numeric Region label, which is decoded using the Region encoder fitted during training. This model-independent design allows the same packet-capture, endpoint-extraction, and feature-construction pipeline to support different classifiers and levels of GeoIP dependency without modifying the upstream inference procedure.

4. Experimental Results

To evaluate the feasibility of the proposed method in a real network environment, the experimental setup follows the topology shown in Fig. 3. The environment consists of a modem, a NETGEAR R7000 gateway router, a host PC, and several downstream devices. The R7000 router serves as the gateway through which downstream network traffic is routed and

periodically exports its connection-tracking table to construct the training corpus, as described in Section 3.1. In addition, the host PC is configured as a wireless hotspot during the inference and re-verification stage, allowing live packets from connected downstream devices to be captured using PyShark. These newly observed records are then processed and classified by the trained model. This topology provides a centralized observation point for collecting gateway-level connection records and live packet data, thereby supporting the evaluation of remote-endpoint region prediction under a realistic home or small-office network setting.

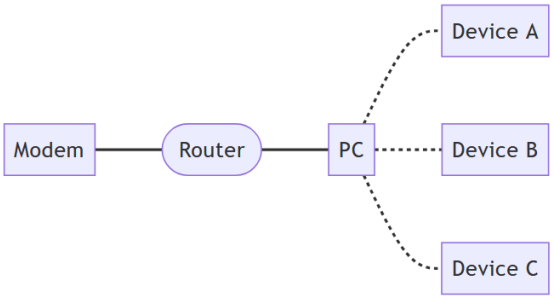


Fig. 3 Network topology of the experimental environment

After applying the preprocessing procedure described in Section 3.3, including endpoint filtering, restriction to IPv4 TCP traffic, deduplication to unique remote endpoints, and removal of region labels represented by fewer than 20 endpoint-level instances, the final dataset contains 4,320 endpoint-level instances across 18 region labels. The resulting label distribution is shown in Table 3. The dataset is markedly imbalanced: the largest class, Unknown, contains 2,141 instances and accounts for 49.56% of the dataset. Therefore, a majority-class predictor that always predicts Unknown is adopted as the baseline for comparison.

Table 3 Region label distribution

Region	Count	Region	Count
Unknown	2,141	Washington	53
Virginia	835	Leinster	41
Taipei City	347	Osaka	37
Tokyo	254	Seoul	35
Oregon	122	Sai Kung District	33
California	106	Hesse	30
Missouri	99	Arizona	27
Iowa	57	North Holland	25
New Taipei City	56	Maharashtra	22

Following the endpoint-level preprocessing and evaluation procedure described in Sections 3.3 and 3.4, the XGBoost classifier was tuned on the training subset and evaluated on the held-out test subset. The resulting performance is compared with that of the original record-level evaluation in Section 4.1 to examine the effect of eliminating duplicated-endpoint leakage. The XGBoost classifier is tuned on the training subset using GridSearchCV and evaluated on the held-out testing subset. Since each remote endpoint is represented only once after preprocessing, the same endpoint-level instance cannot appear in both subsets. Therefore, the reported test performance is not inflated by duplicated connection-tracking snapshots of the same remote endpoint.

4.1 Evaluation protocol comparison

The importance of this design is illustrated by comparing it with the evaluation protocol used in the originally reported experiment, as summarized in Table 4. In the original evaluation, the dataset consisted of 6,397 connection-level records collected at the host, and ten-fold cross-validation over these records yielded 100% accuracy in every fold. However, these 6,397 records corresponded to only 40 distinct remote IP addresses, meaning that each address appeared an average of 159.9 times. Because the records were partitioned without endpoint-level deduplication or grouping, observations associated with

the same remote IP could appear in both the training and validation folds. This train–validation contamination allowed the classifier to learn endpoint-specific mappings and subsequently encounter the same endpoints during validation. Accordingly, the perfect score primarily reflects duplicate-endpoint leakage rather than generalization to previously unseen endpoints.

The present revision addresses this issue at the data level. Deduplication to unique remote endpoints, as described in Section 3.3, makes each endpoint appear exactly once, so no endpoint can be shared between the training and testing subsets. In addition, the minimum support threshold yields an 18-label problem in place of the original 7 labels, which is substantially more challenging. Under this revised protocol, the same model family attains 99.31% test accuracy. This figure is lower than the originally reported result, but it estimates generalization to previously unseen endpoints rather than the reproduction of memorized records.

Table 4 Comparison between the original and revised evaluation protocols

	Original evaluation	Revised evaluation
Collection point	Host PC (wireless hotspot)	R7000 router
Unit of analysis	Connection record	Unique remote endpoint
Distinct remote IP addresses	40	4,320
Region labels	7	18
Classifier	XGBoost	XGBoost
Evaluation protocol	10-fold cross-validation	Stratified 80:20 with GridSearchCV
Accuracy	100.00%	99.31% (Set A)
Macro-F1	100.00%	98.48% (Set A)

4.2 Prediction results across classifiers and feature sets

A qualitative view of the prediction output is shown in Fig. 4. The example is generated from live packet records captured on the host PC using PyShark during the inference and re-verification stage. The prediction is performed using the full feature set, corresponding to Set A in Table 2, where the input features include IP address, Country name, City name, and AS organization. As illustrated in Fig. 4, Region_name represents the GeoIP-derived reference label obtained during attribute extraction, while Predict_Region indicates the region label predicted by the trained XGBoost model. This comparison illustrates how live packet records are transformed into human-readable regional predictions and provides an intuitive view of the model output under the full-feature configuration.

	IP	Country	City	Region_name	ASN	Predict_Region
1	31.13.87.36	Taiwan	Taipei	Taipei City	FACEBOOK	Taipei City
3	31.13.87.36	Taiwan	Taipei	Taipei City	FACEBOOK	Taipei City
5	31.13.87.36	Taiwan	Taipei	Taipei City	FACEBOOK	Taipei City
7	31.13.87.36	Taiwan	Taipei	Taipei City	FACEBOOK	Taipei City
9	31.13.87.36	Taiwan	Taipei	Taipei City	FACEBOOK	Taipei City
10	31.13.87.36	Taiwan	Taipei	Taipei City	FACEBOOK	Taipei City
13	31.13.87.36	Taiwan	Taipei	Taipei City	FACEBOOK	Taipei City
15	72.25.64.32	United States	Unknown	Unknown	NVIDIA-NET	Unknown
16	31.13.87.36	Taiwan	Taipei	Taipei City	FACEBOOK	Taipei City
18	72.25.64.32	United States	Unknown	Unknown	NVIDIA-NET	Unknown
20	31.13.87.36	Taiwan	Taipei	Taipei City	FACEBOOK	Taipei City
23	31.13.87.36	Taiwan	Taipei	Taipei City	FACEBOOK	Taipei City
24	31.13.87.36	Taiwan	Taipei	Taipei City	FACEBOOK	Taipei City
26	142.250.204.42	United States	Unknown	Unknown	GOOGLE	Unknown
29	142.250.204.42	United States	Unknown	Unknown	GOOGLE	Unknown

Fig. 4 Sample prediction results

Because the dataset is highly imbalanced, accuracy alone may provide an incomplete view of model performance. The majority class, Unknown, accounts for 49.56% of the full endpoint-level dataset. Since the stratified test split preserves this class proportion, a majority-class predictor achieves 49.54% accuracy on the held-out test set, while obtaining a macro-averaged F1-score of only 3.68%. Therefore, macro-F1 is reported alongside accuracy in Table 5. Unlike accuracy, macro-F1 gives equal weight to each region label and is therefore more informative when evaluating whether the model performs beyond the dominant class.

Table 5 Feature-combination analysis by GeoIP database dependency

Set	Input features	GeoIP DBs	Classifier	CV acc	Test acc	Macro-F1
A	Country, City, AS organization, IP	3	XGBoost	99.28%	99.31%	98.48%
	Country, City, AS organization, IP	3	Random Forest	99.28%	99.65%	99.07%
	Country, City, AS organization, IP	3	SVM (RBF)	97.95%	98.26%	95.61%
	Country, City, AS organization, IP	3	Logistic regression	77.37%	78.47%	52.64%
B	Country, AS organization, IP	2	XGBoost	91.98%	91.09%	83.86%
	Country, AS organization, IP	2	Random Forest	95.14%	95.02%	87.93%
	Country, AS organization, IP	2	SVM (RBF)	79.25%	78.70%	59.57%
	Country, AS organization, IP	2	Logistic regression	41.93%	42.13%	7.63%
C	AS organization, IP	1	XGBoost	82.41%	82.41%	58.58%
	AS organization, IP	1	Random Forest	92.33%	93.87%	85.66%
	AS organization, IP	1	SVM (RBF)	64.99%	65.62%	19.91%
	AS organization, IP	1	Logistic regression	49.54%	49.54%	3.68%
D	IP	0	XGBoost	76.94%	76.50%	49.62%
	IP	0	Random Forest	91.78%	92.94%	85.24%
	IP	0	SVM (RBF)	58.77%	59.14%	7.18%
	IP	0	Logistic regression	49.57%	49.54%	3.68%
-	Majority-class baseline	0	-	-	49.54%	3.68%

Across all sixteen model–feature-set combinations, the GridSearchCV mean accuracy is consistently close to the corresponding held-out test accuracy, indicating that performance is stable under the adopted evaluation protocol. Under the full feature set, the two tree-ensemble methods perform nearly identically. XGBoost achieves 99.31% test accuracy and 98.48% macro-F1, while Random Forest reaches 99.65% and 99.07%, corresponding to a difference of only three predictions among the 864 test instances. The RBF support vector machine follows with 98.26% accuracy and 95.61% macro-F1, whereas logistic regression performs substantially worse at 78.47% and 52.64%. This ranking is consistent with the non-linear and piecewise structure of the mapping between integer-transformed IP addresses and region labels, which is more readily represented by tree-based and kernel-based classifiers than by a linear decision boundary.

The classifiers diverge more clearly as GeoIP-derived features are progressively removed. For XGBoost, removing the City field reduces test accuracy from 99.31% in Set A to 91.09% in Set B and macro-F1 from 98.48% to 83.86%. Performance declines further to 82.41% accuracy and 58.58% macro-F1 in Set C, and to 76.50% and 49.62% in the IP-only Set D. Random Forest exhibits the same overall trend but degrades considerably more gradually. Its test accuracy decreases from 99.65% in Set A to 95.02% in Set B, 93.87% in Set C, and 92.94% in Set D. The complete removal of GeoIP-derived inputs therefore reduces its accuracy by only 6.71 percentage points, compared with 22.81 percentage points for XGBoost. Its macro-F1 likewise remains comparatively high, declining from 99.07% to 85.24%. By contrast, the support vector machine falls from 98.26% to 59.14% accuracy, while logistic regression decreases from 78.47% to 49.54%. The latter is identical to the majority-class baseline and provides no meaningful class-level discrimination once the categorical geographic attributes are removed.

These results yield two principal observations. First, the City attribute is highly informative but not indispensable. Omitting it reduces the number of required GeoIP databases from three to two, at an accuracy cost of 8.22 percentage points for XGBoost and 4.63 percentage points for Random Forest. This provides a lower-dependency configuration when a moderate reduction in performance is acceptable. Second, complete removal of GeoIP lookup remains viable when an appropriate classifier is selected. Under Set D, Random Forest retains 92.94% accuracy and 85.24% macro-F1 using only the integer-

transformed IP address. The model therefore captures a substantial portion of the IP-to-region regularity represented in the geolocation databases. This is plausible because adjacent or related IP addresses frequently belong to the same allocated network block, service provider, cloud platform, or geographic region.

The difference between the two tree ensembles is most pronounced in Set D, where Random Forest exceeds XGBoost by 16.44 percentage points in accuracy and 35.62 percentage points in macro-F1. Diagnostic experiments suggest that this gap is not attributable solely to insufficient tree depth or an inadequate number of estimators. Increasing the XGBoost maximum depth to 30 and the number of estimators to 500 leaves its accuracy at approximately 76%, whereas a single unconstrained decision tree achieves 92.94% accuracy and 85.33% macro-F1, closely matching the Random Forest result. One plausible explanation is that the IP-to-region relationship approximates a piecewise-constant function over the ordered address space. Recursive partitioning can place decision thresholds near network-block boundaries and continue refining the partitions until relatively homogeneous leaves are obtained. Under the adopted shrinkage and regularization settings, XGBoost may reproduce these narrow boundaries less directly than an unconstrained decision tree or Random Forest. The observations suggest that the address-to-region mapping is more compressible than the XGBoost results alone indicate, provided that the inductive bias of the classifier is well aligned with the structure of the input space.

4.3 Deployment cost and model selection

Because Random Forest achieves the best performance across all four feature sets, the deployment costs of the two tree ensemble models were measured directly rather than inferred from their serialization formats. Table 6 reports model size, loading time, and inference latency, together with the corresponding GeoIP lookup costs measured over the same 864 test endpoints. Under Set A, Random Forest produces the smaller model at 525.9 KB, compared with 1,184.7 KB for XGBoost, but requires more inference time at 4.379 ms per record, compared with 1.583 ms. Under Set D, the trade-off becomes more pronounced. Achieving 92.94% accuracy requires a Random Forest containing 300 unrestricted trees and 227,374 nodes, resulting in a model size of 5,838.4 KB and an inference latency of 11.716 ms per record. By comparison, XGBoost occupies 1,437.2 KB and requires 0.867 ms per record.

Table 6 Deployment costs of trained models and GeoLite2 databases

Component	Size	Initialization time	Median latency per record	Trees	Nodes
Set A, Random Forest	525.9 KB	36.61 ms	4.379 ms	100	17,554
Set A, XGBoost	1,184.7 KB	41.69 ms	1.583 ms	1,800	9,654
Set D, Random Forest	5,838.4 KB	184.94 ms	11.716 ms	300	227,374
Set D, XGBoost	1,437.2 KB	46.91 ms	0.867 ms	1,800	13,832
GeoLite2 City database	60.6 MB	15.63 ms	0.0198 ms	–	–
GeoLite2 Country database	9.1 MB	7.69 ms	0.0107 ms	–	–
GeoLite2 ASN database	11.2 MB	12.20 ms	0.0024 ms	–	–
All three databases for one Set A enrichment	80.8 MB	–	0.0309 ms	–	–

GeoLite2 lookups are memory-mapped and therefore exhibit a strongly right-skewed latency distribution. For example, the GeoLite2 City database has a median lookup latency of 0.0198 ms but a 99th-percentile latency of 9.4392 ms. This pattern is likely caused by page faults occurring when previously uncached regions of the file are accessed. Initialization time is also influenced by the operating-system page cache. To maintain a consistent comparison, Table 6 reports median per-record latency for both model inference and database lookup.

These measurements clarify the scope of the savings provided by the database-free configuration. For the region-prediction task considered in this study, the Set D Random Forest model eliminates the need to store and query the three GeoLite2 databases required by the full Set A pipeline during inference. Its 5.7 MB footprint is substantially smaller than the 60.6 MB GeoLite2 City database, which directly supplies the target subdivision field, and the combined 80.8 MB footprint of

the City, Country, and ASN databases used by Set A. This comparison represents task-specific compression of the GeoIP-derived region mapping rather than replacement of the full functionality of the original databases. The databases additionally provide city, country, and autonomous-system information that the model does not predict.

The primary measured benefit therefore lies in storage reduction and the removal of an external runtime dependency rather than in lower computational cost. Performing all three GeoLite2 lookups for one endpoint has a median latency of 0.0309 ms. When this lookup cost is combined with Random Forest inference under Set A, the complete pipeline requires approximately 4.410 ms per record, compared with 11.716 ms for the Set D Random Forest. Despite eliminating database lookups, the Set D configuration therefore has a higher per-record latency and a 6.71-percentage-point lower test accuracy than its Set A counterpart. In exchange, it removes an 80.8 MB external dependency together with the associated requirements for database storage, updates, and version management. This trade-off is particularly relevant to embedded gateways with limited local storage, but is less consequential when the databases can be accommodated and maintained without difficulty.

Within Set D, the choice between the two tree ensemble models represents a direct trade-off between predictive performance and deployment cost. Random Forest requires approximately 4.1 times the artifact size and 13.5 times the per-record inference latency of XGBoost, while improving test accuracy by 16.44 percentage points and macro-F1 by 35.62 percentage points. XGBoost remains preferable when artifact size or inference latency is the primary constraint, whereas Random Forest is preferable when predictive performance is prioritized. Given the objective of reducing GeoIP dependency while retaining region-prediction capability, Random Forest is selected as the deployment model for the database-free configuration. XGBoost is retained as the original reference classifier to preserve continuity with the initial model design.

4.4 Temporal hold-out validation

The temporal hold-out evaluation follows the partition defined in Section 3.4. Of the 597 endpoints first observed after the training window, 550 fall within the predefined 18-label space and are retained for the closed-set temporal evaluation. The remaining 47 endpoints carry labels outside this space and are excluded from the reported performance metrics. The evaluation results for the retained endpoints are presented in Table 7.

Table 7 Temporal holdout evaluation on endpoints first observed after the training window

Feature set	Classifier	CV accuracy in the training window	Validation accuracy	Macro-F1
A	XGBoost	99.26%	98.91%	98.95%
A	Random Forest	99.44%	98.91%	97.86%
D	XGBoost	77.16%	77.82%	28.22%
D	Random Forest	92.25%	90.91%	75.42%

Accuracy remains broadly stable across the temporal boundary, although class-balanced performance declines under the database-free configuration. Under Set A, XGBoost achieves 98.91% validation accuracy and 98.95% macro-F1, compared with 99.31% accuracy and 98.48% macro-F1 under the stratified random split. Random Forest achieves the same validation accuracy of 98.91%, with a macro-F1 of 97.86%. Under Set D, the classifier ordering observed in Table 5 is preserved. Random Forest achieves 90.91% validation accuracy, compared with 77.82% for XGBoost, and both remain above the majority-class baseline of 48.55%. The close agreement between cross-validation accuracy within the training window and accuracy on the temporal validation set provides evidence that the tuned models generalize beyond the endpoints observed during training.

Two aspects of the temporal results warrant further consideration. First, the proportions of categorical values not observed during the training window are low, at 0.7% for Country, 0.0% for City, and 3.5% for AS organization. The out-of-vocabulary encoding mechanism is therefore applied only marginally and is unlikely to be the primary factor affecting performance. Second, macro-F1 under Set D decreases from 49.62% to 28.22% for XGBoost and from 85.24% to 75.42% for Random Forest. This reduction is partly attributable to the class composition of the temporal validation set. Several retained region

labels contain three or fewer endpoints, meaning that a small number of incorrect predictions can substantially affect their per-class F1-scores. Because macro-F1 assigns equal weight to every label regardless of its support, it is particularly sensitive to these sparsely represented classes. The temporal results should therefore be interpreted using accuracy, macro-F1, and weighted-F1 together. Although low class support contributes to the observed decline, the possibility that part of the reduction is associated with temporal distribution shift cannot be excluded.

Taken together, these findings support the feasibility of reducing GeoIP database dependency during edge-side inference. For the region-prediction task considered in this study, a 5.7 MB Random Forest model correctly predicts the region labels of 92.94% of held-out endpoints without runtime access to the 80.8 MB of GeoLite2 databases required by the full Set A pipeline. The same database-free configuration retains 90.91% accuracy on endpoints first observed after the training window. These results should be interpreted as task-specific compression of the GeoIP-derived region mapping rather than replacement of the complete functionality of the original databases.

The primary measured benefit lies in reduced storage and external dependency rather than lower latency. Direct database lookup remains substantially faster than model inference, and the Set D Random Forest configuration incurs a 6.71-percentage-point reduction in test accuracy relative to Set A. The database-free configuration is therefore most relevant when reducing storage requirements and eliminating runtime database dependencies are more important than minimizing inference latency or preserving the highest attainable accuracy.

The classifier comparison shows that database-free region prediction is supported by both tree ensemble methods, although Random Forest provides substantially stronger performance as GeoIP-derived features are removed. This difference indicates that the achievable degree of compression depends on how well the inductive bias of the classifier matches the piecewise structure of the IP address space. The temporal evaluation further shows that accuracy remains comparatively stable for endpoints observed after the training period, although class-balanced performance is less stable for sparsely represented regions. Validation across independent network environments and broader IP ranges remains outside the scope of the present dataset and is left to future work.

5. Conclusions

This study proposed a machine learning-based framework for predicting the geographic region of remote network endpoints by integrating gateway-level connection-record collection, structured attribute extraction, endpoint-level preprocessing, and supervised classification. The framework is motivated by the limited visibility of communication flows in IoT-oriented residential and small-office networks, where devices frequently establish connections with external services without user awareness. By deduplicating periodically sampled connection records into unique remote endpoints, the revised procedure reduces the risk of duplicate-endpoint leakage and provides a more reliable estimate of generalization. The study further evaluates four classifiers across four feature configurations, measures their deployment costs, and examines their performance on endpoints first observed during a later collection period. Collectively, these findings point to several key conclusions:

(1) Effectiveness of the proposed framework

Under the full feature configuration, both Random Forest and XGBoost achieved excellent prediction performance, with test accuracies exceeding 99%, demonstrating the effectiveness of the proposed framework. The close agreement between cross-validation and held-out test performance indicates that the results are stable under the adopted evaluation protocol. However, because Set A includes the City attribute while the target Region label is derived from the subdivision field of the same GeoIP2 City database, this configuration should be interpreted as an upper-bound reference rather than a database-independent deployment setting.

(2) Effects of classifier and feature selection on GeoIP dependency

Performance generally declines as GeoIP-derived attributes are removed, but the extent of this decline depends strongly on the classifier. Random Forest exhibits the most gradual degradation and retains 92.94% accuracy and 85.24% macro-F1 under the IP-only Set D, compared with 76.50% and 49.62% for XGBoost. The support vector machine and logistic regression deteriorate more substantially as geographic attributes are removed. These findings show that the City attribute is informative but not indispensable and that database-free region prediction remains feasible when the classifier is well matched to the piecewise structure of the IP address space.

(3) Deployment trade-offs and temporal generalization

Under Set D, the Random Forest model occupies approximately 5.7 MB and requires a median inference time of 11.716 ms per endpoint, whereas XGBoost occupies approximately 1.4 MB and requires 0.867 ms. Random Forest therefore incurs greater deployment cost but improves test accuracy by 16.44 percentage points. Its principal benefit is the removal of the 80.8 MB runtime dependency on the GeoLite2 databases rather than reduced latency, since direct database lookup remains faster than model inference. The same Random Forest configuration retains high predictive performance on endpoints first observed after the training window, indicating that its predictive capability extends across time, although lower support for several region labels and possible temporal distribution shift reduce class-balanced performance.

(4) Scope, limitations, and future work

The findings represent a task-specific approximation of the GeoIP-derived region mapping rather than a replacement for the complete functionality of the original databases. The input space is limited to the remote IP address and GeoIP-derived attributes, while the target Region labels are themselves obtained from GeoIP. The present task does not directly perform anomaly detection or malicious traffic classification from packet-level behavioral features. Instead, it improves visibility into the geographic destinations of device communications. Although communication with a foreign region is not inherently malicious, unexpected or policy sensitive cross border connections may indicate potential data exposure, security risks, or regulatory compliance concerns. The predicted region may therefore serve as an auxiliary security indicator that supports subsequent investigation rather than a direct determination of malicious behavior.

Several limitations should be considered when interpreting the findings. The dataset remains imbalanced and was collected from a single gateway environment, while the temporal evaluation was restricted to a predefined closed-set label space. In addition, the analysis focused on remote endpoints rather than device-level IoT behavior, and the use of periodically sampled connection-tracking records limited the analysis of fine-grained traffic characteristics. Future work should incorporate packet- or flow-level behavioral features, characterize individual devices, support open-set recognition of previously unseen regions, and validate the framework using independent datasets collected from diverse network environments. Despite these limitations, the framework provides a practical foundation for improving communication transparency in IoT environments by making the geographic destinations of gateway-observed traffic visible without requiring locally maintained GeoIP databases.

References

[1] N. Dinh and S. Lim, “Performance Evaluations for IEEE 802.15.4-based IoT Smart Home Solution,” International Journal of Engineering and Technology Innovation, vol. 6, no. 4, pp. 274–283, 2016.

[2] T. Wu, F. Breitinger, and S. Niemann, “IoT Network Traffic Analysis: Opportunities and Challenges for Forensic Investigators?,” Forensic Science International: Digital Investigation, vol. 38, article no. 301123, 2021.

[3] A. Andrews, G. Oikonomou, S. Armour, P. Thomas, and T. Cattermole, “Reliable Identification of IoT Devices from Passive Network Traffic Analysis: Requirements and Recommendations,” Proceedings of 2023 IEEE 9th World Forum on Internet of Things (WF-IoT), IEEE, pp. 1–6, 2023.

[4] R. Soepeno, "Wireshark: An Effective Tool for Network Analysis," CYBV-Introductory Methods of Network Analysis, 2023.

[5] M. Alyami, I. Alharbi, C. Zou, Y. Solihin, and K. Ackerman, "WiFi-based IoT Devices Profiling Attack Based on Eavesdropping of Encrypted WiFi Traffic," Proceedings of 2022 IEEE 19th Annual Consumer Communications & Networking Conference (CCNC), IEEE, pp. 385–392, 2022.

[6] H. Kim, H. Lee, and H. Lim, "Performance of Packet Analysis between Observer and Wireshark," Proceedings of 2020 22nd International Conference on Advanced Communication Technology (ICACT), IEEE, pp. 268–271, 2020.

[7] R. Rizal, I. Riadi, and Y. Prayudi, "Network Forensics for Detecting Flooding Attack on Internet of Things (IoT) Device," International Journal of Cyber-Security and Digital Forensics, vol. 7, no. 4, pp. 382–390, 2018.

[8] D. Komosny, M. Voznak, and S. U. Rehman, "Location Accuracy of Commercial IP Address Geolocation Databases," Information Technology and Control, vol. 46, no. 3, pp. 333–344, 2017.

[9] J. P. Chaudhari, K. P. Patel, H. K. Mewada, H. S. Jayswal, Y. P. Kosta, K. S. Bhagat, et al., "Recursive Feature Elimination and Optimized Hybrid Ensemble Approach for Early Heart Disease Prediction," Advances in Technology Innovation, vol. 10, no. 1, pp. 58–71, 2025.

[10] P. Asrodia and H. Patel, "Network Traffic Analysis Using Packet Sniffer," International Journal of Engineering Research and Applications, vol. 2, no. 3, pp. 854–856, 2012.

[11] M. Gharaibeh, A. Shah, B. Huffaker, H. Zhang, R. Ensafi, and C. Papadopoulos, "A Look at Router Geolocation in Public and Commercial Databases," Proceedings of the 2017 Internet Measurement Conference, pp. 463–469, 2017.

[12] J. Martin, T. Mayberry, C. Donahue, L. Foppe, L. Brown, C. Riggins, et al., "A Study of MAC Address Randomization in Mobile Devices and When it Fails," arXiv preprint arXiv:1703.02874, 2017.

[13] S. Ness, V. Eswarakrishnan, H. Sridharan, V. Shinde, N. V. P. Janapareddy, and V. Dhanawat, "Anomaly Detection in Network Traffic Using Advanced Machine Learning Techniques," IEEE Access, vol. 13, pp. 16133–16149, 2025.

[14] Z. Chen, Z. Li, J. Huang, S. Liu, and H. Long, "An Effective Method for Anomaly Detection in Industrial Internet of Things Using XGBoost and LSTM," Scientific Reports, vol. 14, no. 1, article no. 23969, 2024.

[15] M. A. A. da Cruz, L. R. Abbade, P. Lorenz, S. B. Mafra, and J. J. P. C. Rodrigues, "Detecting Compromised IoT Devices through XGBoost," IEEE Transactions on Intelligent Transportation Systems, vol. 24, no. 12, pp. 15392–15399, 2022.



Copyright© by the authors. Licensee TAETI, Taiwan. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY-NC) license (https://creativecommons.org/licenses/by-nc/4.0/).