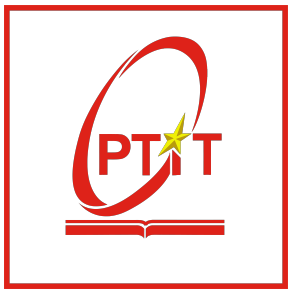

POSTS AND TELECOMMUNICATIONS INSTITUTE OF TECHNOLOGY

HO CHI MINH CITY

INFORMATION TECHNOLOGY DEPARTMENT

Assignment Report

Topic: Encrypted Traffic Classification (ISCX VPN-nonVPN)

Course: Machine Learning

Students:

Hoang-Thien Nguyen

N21DCCN080

Huu-Thang Ha

N21DCCN080

Supervisor:

Dr. Hong-Son Nguyen

May 12, 2025

Contents

List of Tables

List of Figures

1 Introduction

Traffic classification technologies have received increased attention over the last decade due to the implementation of mechanisms for network quality of service, security, accounting, design and engineering. The networking industry as well as the research community have dedicated many efforts to the research of these technologies and came up with several classification techniques. However, the continuous expansion of Internet and mobile technologies are creating a dynamic environment where new applications and services emerge every day, and the existing ones are constantly evolving. Moreover, encryption is becoming pervasive in today's Internet, serving as a base for secure communications. This constant creation, evolution, and securization of applications makes traffic classification a great challenge for the Internet research community.

Traffic classification can be categorized based on its final purpose: associating traffic with encryption (e.g., encrypted traffic), protocol encapsulation (e.g., tunneled through VPN or HTTPS); according to specific applications, (e.g., Skype), or according to the application type (e.g., Streaming, Chat), also called traffic characterization. Some applications (e.g., Skype, Facebook) support multiple services like chat, voice call, file transfer, etc. These applications require identifying both the application itself and the specific task associated with it. Very few traffic classification techniques in the literature address this challenging trends. In early 90's, the initial traffic classification techniques associated transport layer ports with specific applications, a simple and fast technique. But, its low accuracy and unreliability rendered the development of Deep Packet Inspection (DPI) approaches. The DPI approach analyzes packets and classifies them according to some stored signature or pattern. However, DPI techniques that require payload examination are not computationally efficient, specially over high-bandwidth network. Moreover, they are often circumvented by encapsulated, encrypted, or obfuscated traffic that precludes payload analysis. Selecting effective and reliable features for traffic analysis is still a serious challenge.

Generally speaking the classification of network traffic falls mainly into two categories: flow-based classification, using properties such as flow bytes per second, duration per flow, etc. and packet-based classification, using properties such as size, inter-packet duration of the first (or n) packets, etc. In this paper, we focus on analyzing regular encrypted traffic and encrypted traffic tunneled through a Virtual Private Network (VPN). The characteriza tion of VPN traffic is a challenging task that remains to be solved. VPN tunnels are used to maintain the privacy of data shared over

the physical network connection holding packet-level encryption, therefore making very difficult to identify the applications running through these VPN services. In this report, we aim to classify network traffic into VPN and non-VPN categories using three popular machine learning models: Random Forest, Support Vector Machine (SVM), and Gradient Boosting. These models have proven to be effective in handling structured data and can offer high accuracy in classification tasks.

2 Dataset Overview

In this project, we use the ISCX VPN-nonVPN Traffic Dataset, which is provided by the Canadian Institute for Cybersecurity (CIC) at the University of New Brunswick. This dataset is specifically designed for research on encrypted traffic classification and includes both VPN and non-VPN traffic collected from various applications and services.

2.1 Dataset Description

The ISCX VPN-nonVPN dataset consists of network traffic that was captured and labeled across a variety of real-world applications, including:

- Browsing: Under this label we have HTTPS traffic generated by users while browsing or performing any task that includes the use of a browser. For instance, when we captured voice-calls using hangouts, even though browsing is not the main activity, we captured several browsing flows.
- Email: The traffic samples generated using a Thunderbird client, and Alice and Bob Gmail accounts. The clients were configured to deliver mail through SMTP/S, and receive it using POP3/SSL in one client and IMAP/SSL in the other.
- Chat: The chat label identifies instant-messaging applications. Under this label we have Facebook and Hangouts via web browser, Skype, and IAM and ICQ using an application called pidgin.
- Streaming: The streaming label identifies multimedia applications that require a continuous and steady stream of data. We captured traffic from Youtube (HTML5 and flash versions) and Vimeo services using Chrome and Firefox.

- File Transfer: This label identifies traffic applications whose main purpose is to send or receive files and documents. For our dataset we captured Skype file transfers, FTP over SSH (SFTP) and FTP over SSL (FTPS) traffic sessions.
- VoIP: The Voice over IP label groups all traffic generated by voice applications. Within this label we captured voice-calls using Facebook, Hangouts and Skype.
- P2P: This label is used to identify file-sharing protocols like Bittorrent. To generate this traffic we downloaded different .torrent files from a public a repository (archive.org) and captured traffic sessions using the uTorrent and Transmission applications.

2.2 Feature set

The dataset provides pre-extracted features in CSV format, where each row corresponds to a network flow/session. Key features include:

- Duration: The duration of the flow.
- Fiat: Forward Inter Arrival Time, the time between two packets sent forward direction (mean, min, max, std).
- Biat: Backward Inter Arrival Time, the time between two packets sent backwards (mean, min, max, std).
- Flowiat: Flow Inter Arrival Time, the time between two packets sent in either direction (mean, min, max, std).
- Active: The amount of time time a flow was active before going idle (mean, min, max, std).
- Idle: The amount of time time a flow was idle before becoming active (mean, min, max, std).
- Fb psec: Flow Bytes per second.
- Fp psec: Flow packets per second.

3 Feature Engineering

Feature engineering plays a crucial role in shaping the performance of machine learning models by transforming raw data into informative and meaningful input representations. In this project, feature engineering focused on essential preprocessing steps to ensure the data was clean, consistent, and suitable for model training. The process began by removing duplicate records to eliminate redundancy and potential bias in the learning process. Numerical features were standardized using a standard scaler to bring all variables to a common scale, which is particularly important for models sensitive to feature magnitude. Additionally, categorical variables were transformed into numerical format using label encoding, allowing them to be effectively utilized by machine learning algorithms. While relatively simple, these steps laid the necessary foundation for robust model development in the subsequent stages.

3.1 Duplicate Removal

The presence of duplicate samples in a dataset can negatively affect the training process by introducing redundancy and bias. When a model repeatedly sees the same data points during training, the loss function may overemphasize these frequent but identical examples, leading the model to become biased toward them. This can reduce the model's ability to generalize and may skew predictions toward the overrepresented patterns. For example, if several identical records of a particular class are present, the model might learn to prioritize that class disproportionately. To mitigate this issue, a duplicate removal step was applied to the dataset, ensuring that each training instance contributes equally to the learning process and that the model does not inadvertently favor repeated patterns.

3.2 Standard Scaler Normalization

In datasets containing numerical features, it is common for different features to exist on varying scales. For instance, one feature may represent a count ranging from 0 to 1000, while another may be a proportion between 0 and 1. Such disparities in magnitude can cause models that rely on gradient-based optimization (e.g., logistic regression, neural networks) or distance calculations (e.g., k-nearest neighbors, support vector machines) to behave unpredictably or inefficiently. Features with larger

ranges tend to dominate the learning process, potentially leading to suboptimal performance or slower convergence.

To address this, standard scaling was applied to all continuous numerical features. This transformation adjusts each feature to have a mean of zero and a standard deviation of one, effectively normalizing their distributions and aligning them onto a common scale. The standard scaler calculates the z-score for each value using the formula:

$$z = \frac{x - \mu}{\sigma} \tag{1}$$

where x is the original value, μ is the mean, and σ is the standard deviation of the feature. By applying this transformation, the influence of individual features is balanced, allowing the model to treat them equally during training. Moreover, standardization contributes to numerical stability and often improves the speed of convergence in iterative optimization algorithms. Overall, this step ensured that model training was not biased by the arbitrary units or magnitudes of the input features.

3.3 Label Encoding

Machine learning algorithms typically require all input data to be in numerical form. However, real-world datasets often contain categorical variables, which represent qualitative characteristics such as labels, types, or categories. These cannot be directly processed by most models unless they are first converted into a numerical representation. One common technique for handling such data is label encoding, which assigns each unique category in a feature to a distinct integer value.

In this project, label encoding was used to transform all categorical features into numerical values. For example, a feature with categories like {'Low', 'Medium', 'High'} might be encoded as {0, 1, 2}. This process enabled the inclusion of categorical variables in the training pipeline without the need for complex transformations. Label encoding is particularly effective when the number of categories is small and the model used can handle categorical relationships without being misled by the artificial ordering introduced by integers.

While label encoding imposes an implicit ordinal relationship between categories, which might not exist in the original data, it was chosen in this case due to its simplicity and compatibility with the selected models. In situations where no natural order exists or when the categorical variable

contains many distinct levels, alternative techniques such as one-hot encoding or target encoding may be preferred. Nonetheless, for the current dataset, label encoding provided an efficient and straightforward solution for incorporating categorical information into the modeling process.

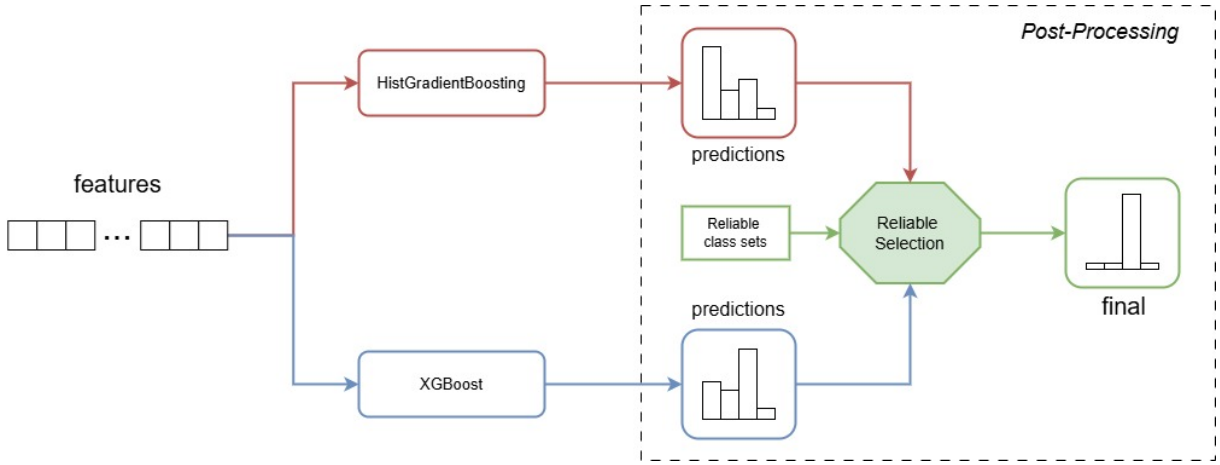

Figure 1: Demonstration of proposed pipeline for traffic classification on ISCX 2016 Dataset

4 Methodology

To build a robust and accurate predictive system, we adopt a hybrid strategy that leverages the complementary strengths of multiple machine learning models. Specifically, two powerful tree-based classifiers are employed: Histogram-Based Gradient Boosting (HistGradientBoosting) and XGBoost. Both models are well-suited for structured tabular data and offer different advantages in terms of bias-variance trade-off, interpretability, and learning dynamics.

Rather than aggregating predictions through traditional ensemble techniques such as majority or soft voting, we implement a post-processing mechanism called *Reliable Selection*. In this approach, the final prediction for each instance is determined based on a predefined set of class-level reliabilities for each model. If one model is known to be more reliable for a particular class, its prediction is prioritized; otherwise, probabilistic confidence or averaged predictions are used to guide the decision. This targeted strategy enables more informed selection and helps avoid errors from less reliable model outputs. By selectively trusting each model where it performs best, the Reliable Selection method aims to enhance overall accuracy, reduce overfitting, and improve generalization across diverse input distributions. The following subsections describe the structure and training of the base models, the construction of the reliable class sets, and the logic behind the selection

process. The overall method can be visualized in Figure 1.

4.1 Histogram-Based Gradient Boosting Classifier

The Histogram-Based Gradient Boosting (HistGradientBoosting) classifier is a highly efficient implementation of gradient boosting, introduced in scikit-learn as an alternative to traditional gradient boosting methods. Unlike standard tree-based methods, HistGradientBoosting first bins continuous features into discrete histograms, reducing memory usage and accelerating training—particularly on large datasets.

This model was selected for its balance between training speed and predictive accuracy, as well as its ability to natively handle missing values and categorical features (if configured appropriately). Like other boosting methods, it builds trees sequentially, where each tree attempts to correct the residual errors of its predecessors. To mitigate overfitting, regularization techniques such as learning rate reduction and early stopping were applied. Hyperparameters including the number of iterations, maximum depth, and learning rate were carefully tuned using grid search and validation metrics.

4.2 XGBoost Classifier

XGBoost (Extreme Gradient Boosting) is a widely adopted gradient boosting framework known for its performance and scalability. It incorporates several enhancements over traditional boosting techniques, such as regularization (L1 and L2), shrinkage (learning rate control), and support for parallel computation, making it both accurate and efficient for a wide range of tasks.

XGBoost was included in this ensemble due to its proven track record in competitive machine learning settings and its strong performance on structured datasets. It is particularly adept at capturing complex patterns through deep trees while still controlling overfitting through effective regularization. In this project, key hyperparameters such as n_estimators, max_depth, learning_rate, and subsample were optimized through randomized search with cross-validation to achieve a good balance between model complexity and generalization.

4.3 Reliability-Based Prediction Selection Strategy

Instead of combining model outputs through traditional ensemble methods such as soft voting, this project adopts a custom post-processing strategy to select predictions from multiple classifiers based

on class-level reliability. The core idea is to exploit the complementary strengths of the Histogram-Based Gradient Boosting classifier (GBT) and the XGBoost classifier (XGB) by giving preference to each model's predictions only when they are known to be reliable for specific target classes. This targeted approach improves decision confidence and model robustness, especially in scenarios with class imbalance or overlapping distributions.

The selection procedure begins by defining sets of reliable classes for both GBT and XGB. These sets represent label categories for which a given model is empirically known—based on prior validation or domain knowledge—to yield consistent and trustworthy predictions. For instance, GBT may perform more reliably on certain VPN-related classes, while XGB may be better suited for non-VPN traffic classes.

During inference, both classifiers produce predicted labels and associated class probabilities for each test sample. A final prediction is selected using a set of rules:

1. If the prediction from GBT belongs to its reliable class set and XGB's does not, the GBT prediction is used.

2. Conversely, if XGB's prediction is considered reliable and GBT's is not, the XGB prediction is taken.

3. If both predictions fall within their respective reliable sets, the model with higher confidence (i.e., maximum probability score) is favored.

4. If neither prediction is considered reliable, the system falls back to averaging the predicted probabilities from both models and selects the class with the highest combined score.

This logic is implemented in a dedicated reliable selection function, which processes predictions and probabilities from both models along with the encoded reliable class sets and outputs a final decision vector. By decoupling prediction selection from fixed weighting schemes, this method adapts dynamically based on context and observed performance patterns. It combines interpretability with strategic model trust calibration, offering a tailored ensemble-like behavior without the need for full model integration.

5 Evaluation & Results

5.1 Experimental Settings

In this experiment, we used two classifiers: HistGradientBoostingClassifier and XGBoostClassifier, both with specific hyperparameters. The HistGradientBoostingClassifier was configured with 150 iterations, a regularization parameter of 10, a learning rate of 0.2, and no limit on tree depth, with a fixed random seed (42) for reproducibility. The XGBoostClassifier was set with 200 estimators, a learning rate of 0.2, a maximum depth of 10, and a subsample rate of 1.0, also with a fixed random state. To enhance performance, we defined reliable class names based on prior knowledge as the higher accuracy, referenced by Table 4: for HistGradientBoosting, the reliable classes were {'FT', 'STREAMING', 'VPN-CHAT', 'VPN-MAIL', 'VPN-P2P', 'VPN-VOIP'}, and for XGBoost, they were {'BROWSING', 'CHAT', 'MAIL', 'P2P', 'VOIP', 'VPN-BROWSING', 'VPN-FT', 'VPN-STREAMING'}. These class names guided the reliable selection process, aiming to improve the overall performance of the model pipeline. The data is splitted into about 80% for training and 20% for testing

5.2 Metrics

To evaluate model performance, we use five standard classification metrics: Accuracy, Precision, Recall, F1 Score, and ROC-AUC. These metrics offer insights into both overall correctness and class-wise prediction quality.

- **Accuracy** measures the proportion of correct predictions:

$$Acc = \frac{TP + TN}{TP + TN + FP + FN}$$

- **Precision** measures how many predicted positives are actually correct:

$$Precision = \frac{TP + TN}{TP + TN + FP + FN}$$

- **Recall** measures how many actual positives are correctly identified:

$$Recall = \frac{TP}{TP + FN}$$

- **F1-Score** represents the harmonic mean of precision and recall:

$$\text{F1} = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall}$$

- **ROC-AUC** represents the area under the ROC curve, reflecting the model's ability to distinguish between classes across thresholds.

Here, TP, FP, TN, and FN refer to true positives, false positives, true negatives, and false negatives, respectively.

Model	Accuracy	Precision	Recall	F1	ROC-AUC
Random Forest	87.23	84.49	82.96	83.68	98.57
SVM	50.26	42.65	31.77	32.76	90.17
XGBoost	89.40	86.54	85.40	85.94	99.33
Gradient Boosting	82.56	79.51	77.20	78.21	97.99
Histogram Gradient Boosting	88.95	86.54	85.40	85.94	99.33
Voting Ensemble	89.53	**87.12**	85.69	**86.35**	99.26
Ours (Reliable Selection)	**89.58**	86.99	**85.74**	86.32	**99.35**

Table 1: Performance comparison of different classifiers.

5.3 Results

Table 1 summarizes the performance of various models on the test set using the evaluation metrics introduced earlier. Among all individual models, XGBoost and Histogram Gradient Boosting achieved strong and nearly identical performance, both with an accuracy above 88% and F1 scores above 85%. Random Forest also performed well, though slightly below the gradient-boosting-based methods. The Support Vector Machine (SVM) model demonstrated significantly lower performance across all metrics, with an accuracy of just 50.26% and an F1 score of 32.76%, highlighting its limitations for this specific task or feature space. A traditional Voting Ensemble, which combines multiple classifiers, slightly improved performance compared to individual models, achieving an F1 score of 86.35 and a ROC-AUC of 99.26.

Our proposed approach, the Reliable Selection strategy, achieved the highest accuracy (89.58%), recall (85.74%), and ROC-AUC (99.35%), while maintaining competitive precision and F1 scores. Notably, although the Voting Ensemble attained slightly higher precision (87.12%), our method offered a better balance between all metrics, particularly in terms of generalization and robustness across classes. These results confirm that the selective trust in model predictions based on class-level reliability can outperform both individual classifiers and standard ensemble techniques.

6 Discussion

Model	Technique	Accuracy	F1	ROC-AUC
Random Forest	Base	87.23	83.68	98.57
	SMOTE	86.45	82.15	98.21
	RandomOverSampler	87.12	83.56	97.53
	RandomUnderSampler	79.17	74.60	97.14
	EditedNearestNeighbours	76.26	69.00	93.90
SVM	Base	50.26	32.76	90.17
	SMOTE	42.96	38.46	88.72
	RandomOverSampler	42.36	37.69	88.65
	RandomUnderSampler	36.31	29.54	83.95
	EditedNearestNeighbours	44.29	25.76	81.78

Table 2: Performance comparison of sampling techniques with Random Forest and SVM classifiers

6.1 Handling Imbalance Data

To address class imbalance, various data sampling techniques such as SMOTE, Random Over/Under Sampling, and Edited Nearest Neighbours were applied to baseline models like Random Forest and SVM. However, as shown in Table 2, these methods did not consistently improve model performance and in several cases, even led to degradation.

For Random Forest, the base model achieved an accuracy of 87.23%, F1 score of 83.68, and ROC-AUC of 98.57, which outperformed all versions using sampling. While some techniques like RandomOverSampler produced comparable F1 scores (83.56), others like Edited Nearest Neighbours significantly reduced all metrics (e.g., F1 = 69.00, ROC-AUC = 93.90). Similarly, SVM performance remained poor across all settings, with sampling failing to improve either its F1 or ROC-AUC.

These findings suggest that simple resampling strategies may distort the underlying data distribution or introduce noise, especially when synthetic or redundant samples are created. Moreover, our final method—Reliable Selection—was specifically designed to account for reliability at the class level by leveraging the strengths of two robust classifiers. This approach inherently mitigates the impact of minority-class errors without needing to artificially rebalance the dataset.

Therefore, we opted not to use sampling techniques in our final pipeline, as our reliability-based post-processing not only preserved the integrity of the original data but also achieved better overall performance across all metrics.

Norm	Method	RF	SVM	GB	Average
MinMax	Accuracy	87.35	48.18	82.61	72.71
	F1 Score	83.72	31.21	78.52	64.48
	ROC-AUC	98.53	88.82	98.00	95.12
Standard	Accuracy	87.23	50.26	82.56	**73.35**
	F1 Score	83.68	32.76	78.21	**64.88**
	ROC-AUC	98.57	90.17	97.99	**95.58**
Standard + PCA(10)	Accuracy	75.24	47.58	65.08	62.63
	F1 Score	69.67	29.35	58.24	52.42
	ROC-AUC	96.23	88.79	93.95	92.32

Table 3: Performance comparison of RF, SVM, and GB under different preprocessing techniques.

6.2 Normalization

Based on the performance metrics presented in the table, the Standard Scaler was selected as the primary scaler due to its consistent and superior results across multiple classification models. Compared to the MinMax Scaler and the combination of Standard Scaler with PCA, it achieved the highest average scores in Accuracy (73.35%), F1 Score (64.88%), and ROC-AUC (95.58%). These results indicate that Standard Scaler offers a more balanced and reliable preprocessing approach.

The improvement was especially notable for the Support Vector Machine (SVM), which is sensitive to feature scaling. Using the Standard Scaler, SVM's performance improved across all metrics compared to MinMax. In contrast, applying PCA after scaling led to a clear drop in performance, likely due to loss of critical feature information.

Overall, the Standard Scaler proved effective across both scale-sensitive models like SVM and tree-based models like Random Forest and Gradient Boosting, making it the most suitable choice for this task.

	RF	HistGB	XGB	Voting	Ours
B	84.39	86.99	88.27	87.29	88.03
C	80.25	81.45	83.29	83.49	82.73
F	87.24	88.47	86.40	87.07	87.47
M	78.88	79.39	80.89	80.00	79.88
P	87.22	92.25	92.50	92.29	92.12
S	88.41	89.47	88.49	91.67	89.71
V	98.10	98.11	98.49	98.57	98.57
VB	85.40	86.79	87.43	87.74	87.82
VC	77.17	79.80	78.19	79.75	79.55
VF	78.99	82.36	84.35	84.34	84.44
VM	76.07	78.57	77.19	79.46	80.91
VP	70.86	76.87	75.16	76.97	75.82
VS	94.01	92.40	94.08	94.05	94.08
VV	95.92	96.94	96.78	96.96	96.77
Mean	84.90	86.89	86.92	87.18	**87.65**
Std	7.98	6.90	7.24	6.86	**6.84**

Table 4: Accuracy of each class for RF, HistGB, XGB, Voting, and Ours.

6.3 Confusion Metrics

Based on the performance of HistGB and XGB in Table 4 for each class, we observed that each model captures specific and distinct features, allowing them to excel in different classes. This observation led to the assumption that combining these models using heuristic information about their reliable classes could enhance the overall performance of the pipeline. This insight forms the basis of our proposed *Reliable Selection* module.

The effectiveness of our proposed method is clearly demonstrated by its superior average performance and stability across all classes. With a mean accuracy of 87.65%, our approach outperforms all other models, including ensemble methods like Voting (87.18%) and individual classifiers such as XGBoost (86.92%), HistGradientBoosting (86.89%), and Random Forest (84.90%). Additionally, our method achieves the lowest standard deviation (6.84) among all compared models, indicating more consistent and reliable performance across various classes. This combination of the highest average accuracy and lowest variability underscores the robustness and generalization capability of our model.

7 Conclusion & Future Work

This work introduced a hybrid classification framework that combines the strengths of Histogram-Based Gradient Boosting and XGBoost using a post-hoc selection strategy, termed Reliable Selection. Rather than merging predictions through simple majority voting, our method makes informed decisions based on the class-wise reliability of each model and their respective prediction confidences. This allows the final classifier to maintain high accuracy and robustness, even without relying on traditional resampling techniques for class imbalance. Experimental results confirmed the effectiveness of this approach, showing improved performance across various evaluation metrics when compared to both individual models and ensemble baselines.

Future Directions. While our method avoids explicit resampling, class imbalance remains a key challenge, especially in real-world scenarios with skewed distributions. Traditional sampling techniques can introduce noise or even hinder generalization, so future work should explore algorithm-level solutions such as cost-sensitive learning, adaptive loss functions like focal loss, and confidence-aware strategies that better handle rare classes. Another direction is improving how reliability is defined without prior knowledge of class distributions. In dynamic settings, static reliability sets may be inadequate. This highlights the need for data-driven reliability estimation and model self-assessment, potentially enhanced by calibration and explainability techniques for more robust and transparent decision-making.

In summary, while our approach demonstrates a practical and effective way to address class imbalance without altering the training distribution, it also reveals deeper challenges related to learning under data imbalance. Addressing these will require a shift from external data manipulation to internally adaptive and distribution-aware modeling strategies, which remains a rich and evolving area of research.

8 Study Materials

References