TALLINN UNIVERSITY OF TECHNOLOGY
School of Information Technologies

Keisuke Konno IVSB201825

Multi-class Classification of Botnet Detection by Active Learning

Bachelor's thesis

Supervisor: Hayretdin Bahsi
PhD

Tallinn 2023

TALLINNA TEHNIKAÜLIKOOL
Infotehnoloogia teaduskond

Keisuke Konno IVSB201825

Botneti tuvastamise mitmeklassiline klassifikatsioon aktiivse õppimise abil

Bakalaureusetöö

Juhendaja: Hayretdin Bahsi
PhD

Tallinn 2023

Author’s declaration of originality

I hereby certify that I am the sole author of this thesis. All the used materials, references to the literature and the work of others have been referred to. This thesis has not been presented for examination anywhere else.

Author: Keisuke Konno

15.05.2023

Abstract

This paper investigates the use of active learning with multiclass classification techniques to enhance the accuracy and efficiency of IoT botnet detection systems. The resource-intensive and time-consuming task of labelling data for ML solutions may be mitigated through active learning, which reduces the data size required for training the ML models. The study focuses on the utilization of both labelled and unlabelled datasets to train the model, with an emphasis on pool-based sampling. It explores various query strategies for selecting informative instances from the unlabelled dataset, evaluates their effectiveness, and compares their performance with traditional classification techniques. Multi-class classification techniques are adopted to provide detailed information on botnet traffic, facilitating incident analysis. Ultimately, the research aims to improve the effectiveness of IoT botnet detection systems through the adoption of active learning with multiclass classification techniques.

This thesis is written in English and is 69 pages long, including 7 chapters, 23 figures and 47 tables.

List of abbreviations and terms

ML	Machine Learning
C&C	Command & Control

Table of contents

List of figures

List of tables

1 Introduction

Botnets are a type of network malware that are used to carry out a variety of malicious activities such as identity theft, distribution of malware or spam and primarily DDoS attacks. They are created by infecting a large number of devices, which allows the attacker to control them remotely and use them to carry out the malicious activities. Botnets have become huge threat in recent years, with attacks becoming more sophisticated and difficult to detect.

According to the IBM article [1], DDoS attacks carried out by IoT devices have increased every year since the massive attack in 2016, which was used a botnet called Kaiten, also known as Mirai. Various types of botnets targeting IoT devices have been developed and discovered since then.

As various forms of botnets continue to develop and advance using novel techniques, safeguarding against them using conventional and outdated strategies becomes progressively challenging. Traditional approaches, such as signature-based detection system and manual monitoring, are incapable of accurately identifying a mere 19% of the alerts generated by these solutions [2]. Furthermore, signature-based detection systems rely on pre-defined signatures to identify malicious activity, which can be easily bypassed by botnets that use new and unknown methods. Manual monitoring system is not scalable and requires a significant human resource.

In recent years, Machine Learning (ML) has emerged as a promising alternative for botnet detection. ML algorithms have the ability to learn and adapt to new patterns, making them particularly effective in identifying unusual behaviors in network traffic. This is especially true for supervised models [3] [4].

One study, conducted by M. Stevanovic and J. M. Pedersen, proposes a novel botnet detection system that utilizes flow-based traffic analysis in conjunction with supervised ML. The system demonstrates high accuracy in classifying traffic, even with a limited amount of data per flow [5]. Another study, by Dau Xuan Hoang and Quynh Chi

Nguyen, presents a botnet detection model that employs ML techniques on DNS query data [6].

The use of ML extends to malware detection as well. A study by CSIT suggests that employing a machine learning approach can facilitate earlier detection of Android malware. By utilizing multiple classifiers, the approach improves detection accuracy and expedites analysis [7].

However, one disadvantage of supervised models is its reliance on a large amount of labelled data to train ML models, which can consume considerable time, human resources, and monetary expenses. To overcome this problem, unsupervised algorithms are recommended by the research community as a solution to this issue. Although, certain amount of labelled data is still required for some algorithms containing only benign.

In the context of ML-based botnet detection, active learning has proven to be a workable approach to the challenge of acquiring labelled data [8] [9]. By iteratively selecting the most informative samples for labelling, active learning algorithms can reduce the human effort and resources required to label large datasets. Integrating active learning into the ML pipeline can lead to improved efficiency and effectiveness in botnet detection, while reducing the labelling effort typically associated with ML models.

This study aims to benchmark the effectiveness of active learning for botnet detection from binary to multi-class classification, with a particular focus on identifying different types of bots. Accurate classification of malware is essential for identifying appropriate mitigation strategies and minimising the impact of the threat. In addition, the identification of the characteristics of malware can provide insight into future prevention efforts.

2 Background

2.1 Active Learning

Active learning, a form of semi-supervised learning, has been proposed as a solution to this shortage of labelled dataset problem. Unlike traditional supervised learning, active learning utilizes both labelled and unlabelled datasets to train a ML system. By selectively choosing the most informative unlabelled samples, active learning algorithms can achieve better performance with fewer training steps or instances. In essence, the key idea behind active learning is to optimize the selection of the most informative samples, leading to a more efficient training process.

The selection of samples from the unlabelled dataset at each iteration and the updating of the model capabilities are based on informativeness. There are different scenarios in which active learning can be applied, depending on the problem setting. Some examples include pool-based sampling, where a fixed pool of unlabelled data is available for selection; stream-based sampling, where the data arrives continuously in a stream; and membership query synthesis, where the model can actively ask for labels from an oracle [10]. This paper uses pool-based sampling, which is flexible and easy to implement in real-life situations. This makes it an ideal choice for many types of problems.

In the pool-based scenario, ML model trained with a small amount of data with sampling algorithm (query strategy) selects the most informative instance (query instances) from an unlabelled dataset, which can be highly effective in the iterative training of ML models. Subsequently, the selected sample is presented to an oracle, who is malware analyst in this instance, for labelling. After labelling, the labelled sample is added to the pool of labelled instances, which are then used for training the ML model. This iterative process can ultimately reduce the size of the required labelled dataset.

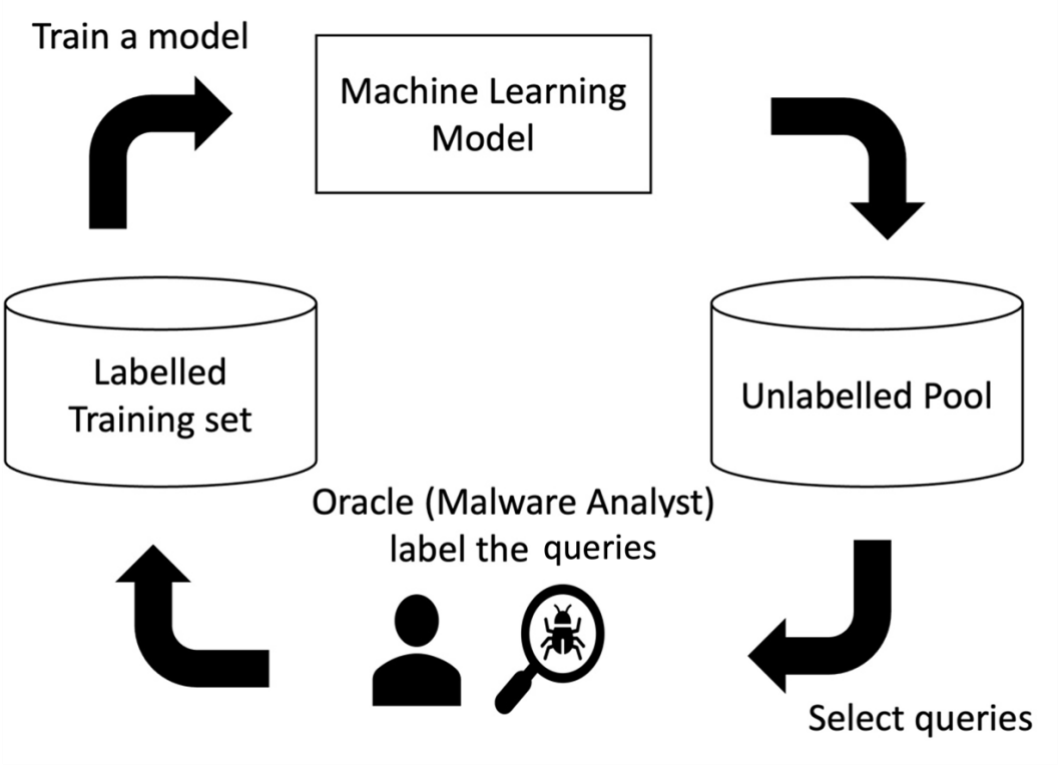

Figure 1. Pool-based active learning cycle

The success of active learning in selecting informative instances from an unlabelled dataset depends heavily on the chosen query strategy. Therefore, this study evaluates and compares various query selection strategies. In the following paragraphs, the different query strategies are described in detail.

2.2 Uncertainty Sampling

In uncertainty sampling, the most informative instance is selected for labelling by evaluating the uncertainty or degree of doubt of the ML model about each instance's label and selecting the one with the highest uncertainty. Examples of informativeness scores are given below:

2.2.1 Classification Uncertainty (U)

It measures how uncertain the ML model is about the predicted class label for each instance. It is calculated as:

$$\mathrm{U(x)} = 1 - \mathrm{Max}\big(\mathrm{P(y|x)}\big)$$

where x is the instance, y is class labels and $\mathrm{P(y|x)}$ is the probability of each class given the instance x. Max function returns the highest probability value among all class labels. The instance with the smallest score will be selected.

2.2.2 Classification Margin (M)

It selects an instance that has the smallest difference between the class with the highest probability and the class with the second highest probability. It is calculated as:

$$M(x) = P(y_{max}|x) - P(y_{max-1}|x,)$$

where y_{max} is the class with the highest probability and y_{max-1} is the class with the second highest probability for a specific instance x.

2.2.3 Classification Entropy (E)

This is a measure of how much uncertainty there is in the probability distribution over the possible class labels for an instance. It is calculated as:

$$E(x) = -\sum_{k} P(y|x) \log P(y|x)$$

where x is the instance and $P(y|x)$ is the probability of each class y given the instance x. The summation calculates the sum of the negative logarithm of the probabilities of the possible labels. The instance with largest score will be selected.

2.3 Query by Committee

Query by Committee selects informative instances for labelling by training a committee of ML models. It takes a majority vote from multiple ML models and selects the instance with the most split votes. It tends to be like uncertainty sampling in the sense that instances are selected from uncertainty. Examples of informativeness scores are given below:

2.3.1 Vote Entropy (VE)

calculates the entropy of the voting result information and select the instance with the largest score. It is calculated as:

$$VE = argmax_x - \sum_{y} \frac{V(y)}{C} \log \frac{V(y)}{C}$$

where, I $V(y)$ represents the number of votes for a particular class and C represents the size of the committee. $\frac{V(y)}{C}$ represents the consensus probability of label.

2.3.2 Consensus Entropy ($\boldsymbol{CE}$)

calculates the consensus entropy of the voting result information and select the instance with largest score. It is calculated as:

$$CE = -\sum_{y} P_{Cs} \log P_{Cs}$$

where $P_{Cs} = \frac{1}{C}\sum_{c=1}^{C} P(y_i)$ is the consensus probability. This represents the average of the class probabilities of each ML model.

2.3.3 Maximum Disagreement ($\boldsymbol{MD}$)

calculates the consensus probability by calculating average of the class probabilities. Kullback-Leibler divergence is calculated instead of entropy. Kullback-Leibler divergence calculates the difference between the model's predicted distribution of labels and the true distribution of labels. Instance with the highest divergence score is selected. It is calculated as:

$$MD = argmax_x \frac{1}{C}\sum_{i}^{c} D(P_{\theta^c}||P_{CS})$$

where $D(P_{\theta^c}||P_{CS}) = \sum_i P(y_i|x;\theta^c) log \frac{P(y_i|x;\theta^c)}{P_{CS}}$ calculates the Kullback-Leibler divergence. θ^c represents the ML model of the committee, which makes $P(y_i|x;\theta^c)$ means the predicted probability of label y by the model, and P_{CS} is the consensus probability.

2.4 Ranked batch-mode Sampling

While the standard pool-based sampling can only return one instance per query, ranked batch-mode sampling addresses this limitation by enabling the learner to query multiple instances simultaneously, thereby gathering more information and reducing the overall cost of labelling. Formula proposed by Cardoso et al is following:

$$Score = \alpha\big(1 - \phi(x, X_{labeled})\big) + (1-\alpha)U(x)$$

where α is ratio of the training set and total available instances $\alpha = \frac{X_{labelled}}{X_{unlabelled}+X_{labelled}}$, $X_{labelled}$ is the labelled dataset and $X_{unlabelled}$ is the unlabelled dataset.

ϕ is a similarity function which represents the resemblance between the instances. This function gives higher scores to instances that are different from the already labelled documents, so that they can explore new areas of the instance space. $U(x)$ is the least confident score. Overall, this sampling method selects the highest-scoring sample, remove it, recalculate scores for the remaining instances, and repeat until the desired batch size is reached. Query and label the batch in a single training step to update the learning model.

3 Related Work

The field of cybersecurity has witnessed continuous advancement in machine learning techniques over the years, with various applications including malware detection, spam email classification, and threat intelligence. Notably, machine learning has been employed in network intrusion detection, with active learning techniques being a particular focus to enhance labelling efficacy.

In a previous study, Alejandro and Hayretdin conducted a benchmark experiment on binary classification, evaluating different active learning scenarios such as uncertainty sampling, query by committee, and ranked batch-mode sampling, and examining the impact of incorrect labelling on performance [5]. The authors concluded that active learning cycles produced significantly better results than passive baselines, requiring at least ten times less data. However, the study did not consider how a trained active learner might perform against datasets from other botnet domains. This aspect raises the question of the generalizability of the active learning approach across different domains and whether a model trained on one dataset can effectively detect intrusions in other botnet domains. To address this issue, this study explores the transferability of active learning models and investigate their performance on diverse datasets. Furthermore, it should be noted that while binary classification is important, multi-class classification is also a critical task in network intrusion detection. This studie explores the effectiveness of active learning approaches in the context of multi-class classification, as this has the potential to improve the accuracy of intrusion detection systems in identifying various types of network attacks. Additionally, investigating the transferability of active learning models across multiple domains is necessary to ensure their applicability in real-world scenarios.

4 Methodology

4.1 Experiment Environment

List of experiment tools and libraries are shown below:

Table 1. Experiment Environment

Programming Language	Python 3.8.10
Python Library and Usage	modAL (0.4.1) Active Learning Scikit-learn (1.2.1) Pre-processing, ML algorithms, evaluation Matplotlib (3.5.1) Generate graphs Pandas (1.3.4) Pre-processing Numpy (1.21.4) Pre-Processing

4.2 Dataset

ML datasets play a crucial role in developing effective intrusion detection systems for IoT networks. Two such datasets, MedBIoT [11] and N-BaIoT [12], have been created to provide comprehensive and labelled data for botnet research. Both datasets contain benign and malicious network traffic, enabling the training of ML models to capture pattern of the traffic patterns and testing the trained ML model to evaluate performance.

4.2.1 MedBIoT Dataset

The dataset comprises network traffic data recorded from 83 authentic and emulated IoT devices within a medium-sized network. It encompasses 3 types of malwares (Mirai, Torii, BashLite) deployed during the initial stages of botnet deployment, including the Command & Control (C&C) phase between the botmaster and the proliferation of malware. This facilitates data labeling and concentrates on the early detection of threats and prevention of attacks. The dataset consists of 100 features derived from statistical methods applied sequentially to raw network packet data (e.g., packet size, packet count, packet jitter in 100ms, 500ms, 1.5 sec). The features are presumed to be inherently interpretable by a human analyst. This labelled dataset is suitable for both supervised and unsupervised learning and can be used for IoT botnet research and intrusion detection systems. The research used Sonoff Tasmota smart switch, TPLink

smart switch, and TPLink light bulb as real devices, and a lock, switch, fan, and light as emulated devices.

4.2.2 N-BaIoT Dataset

This other dataset on the other hand, encompasses 2 types of malwares (Mirai, BashLite) and benign captured network traffic from 9 devices includes doorbell, webcam, thermostat, baby monitor, security camera in small-sized network. The data was captured under the actual environment focuses on the attack phase of the botnet. The dataset contains 115 features. But still, structure of the dataset is the same as MedBIoT with additional 15 features. The dataset focusses on attack phase of the botnet unlike MedBIoT.

4.2.3 Dataset Pre-processing

The study uses the MedBIoT dataset to measure benchmark performance and the N-BaIoT dataset to observe whether it exhibits the same characteristic as an active learner trained with the MedBIoT dataset. The study assumes that an Oracle can interpret the dataset's features and labels, which are used to train and test ML models. Initially, 120,000 instances are randomly extracted from the entire MedBIoT dataset, with 30,000 instances for each of the Benign, Mirai, Bashlite, and Torii labels. The data is split into two parts for training and testing while maintaining a balanced class composition. The training dataset has 80,000 instances and is used for training ML models. The testing dataset has 40,000 instances and is used to evaluate generalizability. In terms of N-BaIoT dataset, the dataset is set up to mimic MedBIoT dataset as closely as possible and consists of 90,000 instances with 30,000 instances each for Benign, Mirai and Bashlite labels.

In addition to the dataset size, feature selection and feature scaling were performed. Pearson's correlation coefficient was calculated from the features, and those with correlation coefficients above 0.8 were excluded. After removing the features, the data were standardized using the Standard Scaler formula, which is given below:

$$Z = \frac{(x - u)}{s}$$

where x is the sample score, u is the mean value of the set, and s is the standard deviation. In the experiment, the mean and standard deviation values were calculated

from the training dataset. This standard scaler was applied to both the training and testing data.

Since the active learning cycle requires the availability of unlabelled data (pool), the pool of unlabelled data was always extracted from the training dataset. The testing dataset was used to evaluate the accuracy of the active learning models at every iteration.

4.3 Evaluation Scores

When evaluating the performance of an ML model, statistical scoring measures are used to quantify its effectiveness. These scores help objectively assess how well the model can make predictions on new, unseen data. To ensure that all classes are treated equally, weighted method is used for calculating each score. The weighted accuracy method assigns a weight to each class in a classification problem based on its representation in the dataset. It then calculates the overall accuracy of the model by taking the sum of the product of the number of true positives and the weight for each class, divided by the total number of samples in the dataset. In this study, F1 score is the main evaluation metric because it considers both false positives and false negatives, making it a reliable measure of a model's ability to correctly classify samples from all classes. The statistical measures used as evaluation scores are listed below:

4.3.1 Accuracy

Accuracy refers to the proportion of correctly classified samples out of all the samples in the dataset. Equation is explained below:

$$Accuracy = \frac{True\ Positives + True\ Negatives}{True\ Positives + True\ Negatives + False\ Positives + False\ Negatives}$$

$$Weighted\ Accuracy = \sum_{i=1}^{n} w_i * Accuracy_i$$

where $w_i = \frac{Number\ of\ instances\ in\ class\ i}{Total\ number\ of\ instances}$ and $n = number\ of\ class$

4.3.2 Precision

Precision measures how many of the positive predictions made by the model are correct. As a method, micro is used for the calculation same as accuracy. Equation is explained below:

$$Precesion = \frac{True\ Positives}{True\ Posives + False\ Posives}$$

$$\text{Weighted Precesion} = \sum_{i=1}^{n} w_i \ * \ \text{Precesion}_\text{i}$$

where $w_i = \frac{Number\ of\ instances\ in\ class\ i}{Total\ number\ of\ instances}$ and $n = number\ of\ class$

4.3.3 Recall

Recall a measure of a model's ability to correctly identify all instances of a given class. Equation is explained below:

$$Recall = \frac{True\ Positives}{True\ Posives + False\ Negatives}$$

$$\text{Weighted Recall} = \sum_{i=1}^{n} w_i \ * \ \text{Recall}_\text{i}$$

where $w_i = \frac{Number\ of\ instances\ in\ class\ i}{Total\ number\ of\ instances}$ and $n = number\ of\ class$

4.3.4 F1

The F1 score is a metric that balances the precision and recall of a classification model into a single value. It provides a measure of the model's overall performance, ranging from 0 to 1, with higher scores indicating better performance. Equation is explained below:

$$F1 = \frac{2 * Precesion * Recall}{Precesion + Recall}$$

4.4 Baseline Model Performance

To evaluate the impact of active learning, it's important to compare its effects with the baseline performance of machine learning models. Typically, static, basic models are

used as the baseline for intrusion detection in production, with few updates. In our study, we compare 5 commonly used supervised classification algorithms, including Random Forest, K-Nearest Neighbours, Decision Tree, Logistic Regression, and Support Vector Machine, to establish a baseline performance. Based on the results, we select the algorithm that outperforms the others as the main algorithm for active learning.

The algorithms are trained on the training set and then tested on the test set, with their hyperparameters set to default values without optimisation. The evaluation scores in the above section are measured when the model is tested on the test data set. To find out the best number of features to use, the top 3, 5, 10, 15, 20 features are selected from the dataset based on Fisher's score. To account for variations caused by the algorithm's random number seed, each test was performed 5 times.

4.5 Active Learning Experiments

After comparing and selecting baseline models, the algorithm with the best performance was chosen for the active learning experiment. To evaluate the active learning performance, each query strategy was implemented and tested 5 times, with average scores calculated for each test. Training dataset is used as initial seed and unlabelled pool to train active learner, and testing dataset is used for performance evaluation in each query iteration.

Four types of query strategies were used in total for this experiment as follows:

4.5.1 Uncertainty Sampling

The active learning cycle uses a single ML model to generate initial detection model trained with initial seed. This cycle is repeated 1,000 times and one instance is chosen from unlabelled pool set using a query strategy in each iteration to update the model, hence 1,000 instances are used for the criteria. In each iteration, evaluation scores from section 4.2 are measured with testing dataset. The experiment employs 3 query strategies: classification uncertainty, classification margin, and classification entropy. In every iteration, the evaluation scores from the testing dataset are measured. Additionally, to evaluate the impact of different patterns of initial seed and unlabelled pool size, 4 different sizes of initial seeds (4, 12, 40, and 200 instances) and 6 different sizes of unlabelled pools (1000, 4000, 8000, 12000, 20000, and 50000 instances) are

tested. Initial seed and unlabelled pool are extracted from the training dataset randomly every time. For initial seed, each instance number is balanced and contains the same number of instances for each class, and every combination of initial seed and pool size is tested.

4.5.2 Uncertainty Sampling in Binary Classification

To compare binary class classification and multi-class classification, labels in the dataset used in the multi-class experiment are changed. The classes of Mirai, Bashlite, and Torii are labeled as one botnet. However, the ratio of the dataset by class is kept the same (i.e., 30,000 instances for Benign, Mirai, Bashlite, Torii equally). The data is split into two parts for training and testing, with a balanced class composition. The training dataset contains 80,000 instances, and the testing dataset contains a total of 40,000 instances. Since the purpose of this is score comparison, only classification uncertainty is used as a query strategy. The same initial seed and unlabelled pool settings from the uncertainty sampling experiment are used.

4.5.3 Ranked batch-mode Sampling

Similar to uncertainty sampling, a single ML model is used to generate an initial detection model, and the cycle is repeated 1,000 times. However, in this case, the query strategy selects batches of instances from the unlabelled pool with different sizes (4, 8, 20, 40 batch instances) in each iteration to update the ML model. Number of instances for each class for the batch instances are balanced. The total number of iterations depends on the batch size; for example, if 4 batch instances are selected every iteration, the iteration is repeated 250 times. The same evaluation scores are measured using a testing dataset, and the same initial seed and unlabelled pool patterns from uncertainty sampling experiment are used.

4.5.4 Query By Committee

Multiple ML models are used to generate a detection model in this cycle. A committee of models is formed, and 3 query strategies (vote entropy, consensus entropy, maximum disagreement) are used to query instances from the unlabelled pool in every iteration. The cycle is repeated 1,000 times, and one instance is chosen from the unlabelled pool in each iteration. The same evaluation scores are measured using a testing dataset, and the same initial seed patterns are used. To minimize the variable, only the best pool size

of the uncertainty sampling from uncertainty sampling experiment is used for the query by committee experiment. Different size of committee is used (2, 3, 5, 7, 10).

4.5.5 Random Sampling

In this experiment, a query strategy that selects instances randomly is used, in comparison with the active learning cycle. The cycle is repeated 1,000 times, with one instance being chosen from the unlabelled pool in each iteration. The same settings as uncertainty sampling are applied for initial seeds and unlabelled pool sizes. The same evaluation scores are measured using a testing dataset, along with the same initialization seed and unlabelled pool patterns.

4.5.6 Testing with N-BaIoT Dataset

To investigate the generality of the active learner model in depth, we conduct the same experiment using an N-BaIoT dataset. The dataset is split into 60,000 and 30,000 instances for training and testing purposes, respectively. The training dataset is used as the initial seed and unlabelled pool set, just like in the previous active learning experiment. The testing dataset is used to measure performance at every iteration. We evaluate the predicted and original classes using a statistical score function. To maintain consistency in our research, we restrict the query strategy to random sampling, ranked batch-mode sampling and the top-performing methods from the previous section: uncertainty sampling and query by committee. Since the dataset has only 3 classes (Benign, Mirai and Bashlite), different initial seeds are used. As initial seed patterns, 3, 9, 30 and 150 are used and the numbers are balanced and contain the same number of instances for each class. The other settings are the same.

5 Results

In summary, the Random Forest algorithm outperformed all other algorithms and was selected as the primary algorithm for the active learning experiments. In these experiments, uncertainty sampling yielded the most consistent and effective results compared to query by committee and random sampling. The results of Ranked batch-mode sampling were almost the same or lower than the result of random sampling. The following sections present the results of all experiments.

5.1 Dataset Pre-processing

Pre-processing aims to improve the quality of the data, making it more accurate and reliable for ML training. For this purpose, Pearson's linear correlation coefficient was calculated on MedBIoT dataset, and features correlated each other more than 0.8 are removed. As a result, 20 features out of 100 features are remained in the dataset. Same features are extracted from N-BaIoT dataset.

After the feature selection, feature scaling was implemented with standard scaler. The mean value and standard deviation value are calculated based on the training data of MedBIoT, and scaled data are calculated for training data and testing data of MedBIoT data and whole N-BaIoT dataset as second testing dataset.

5.2 Baseline Model Performance

5 algorithms, namely Random Forest, K-Nearest Neighbours, Decision Tree, Logistic Regression, and Support Vector Machine, were used as baselines. They were first trained with the training set and then evaluated with the testing set. To determine the optimal number of features, we tested the algorithms using the top 3, 5, 10, 15, and 20 features based on Fisher's score and this process was repeated 5 times.

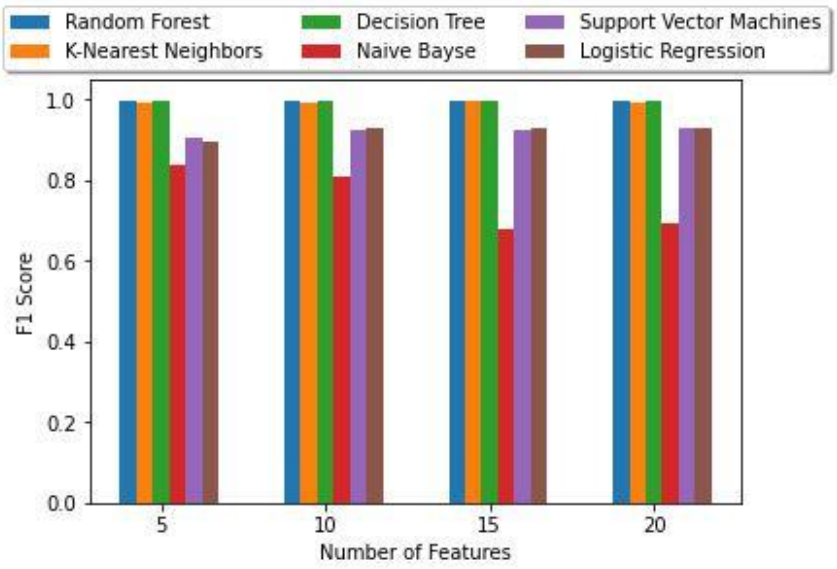

Figure 2. Comparison of Baseline Performance

The average score was calculated, and the results are shown in Figure 2. The graph indicates that the Random Forest algorithm outperformed the other algorithms in all cases, although only slightly better than the Decision Tree and K-Nearest Neighbours algorithms. Therefore, the Random Forest algorithm was chosen for the active learning experiments.

5.3 Active Leaning Experiments

The following paragraphs shows the performance result for the active learning experiments described in section 4.5.

5.3.1 Random Sampling

The results are shown in Figure 3. Detailed score comparisons are available in Table 2-Table 4. The graphs demonstrate that a larger initial seed results in faster convergence to a higher score. This is because starting point before querying phase is already well scored. However, the effect of pool size is not apparent since the lines in each graph overlap significantly. An F1 score of 90% is achieved within 100 queries with an initial seed of 4 or 12, and with an initial seed of 40 or 200, F1 score of over 0.9 are achieved even before querying from the unlabelled pool. The highest score achieved in this experiment is 0.98.

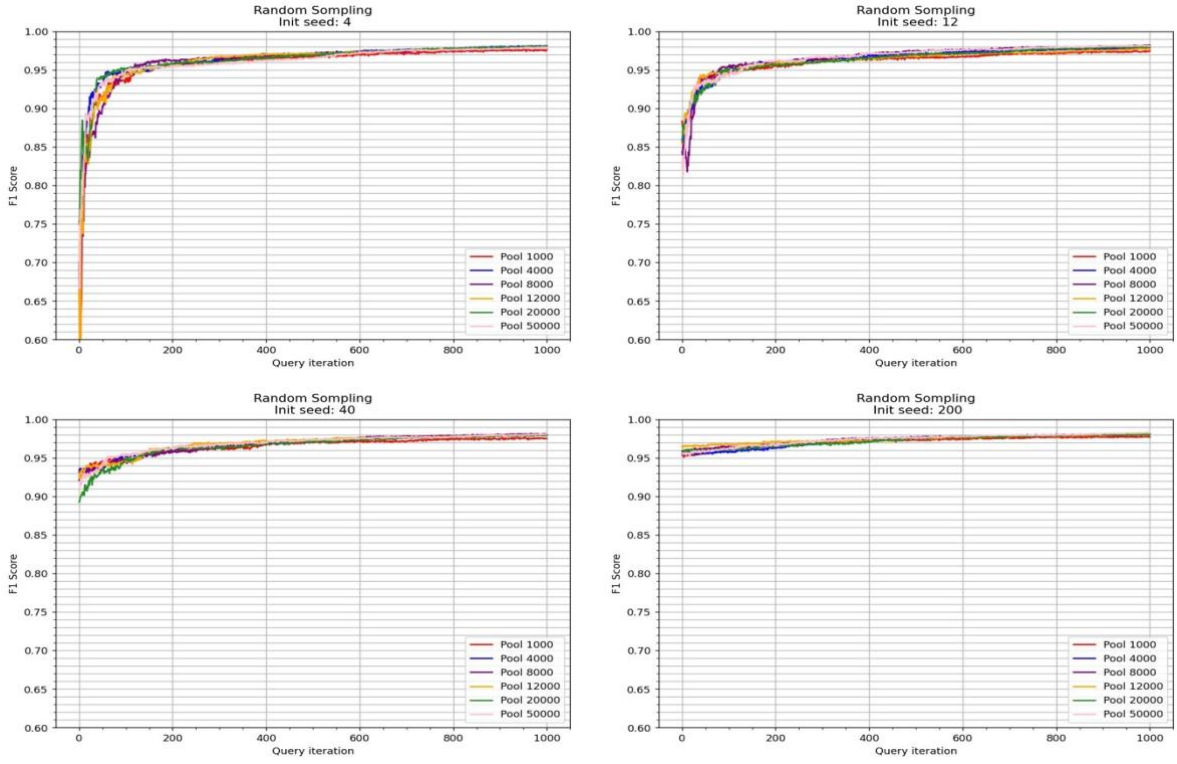

Figure 3. Random Sampling Result

Table 2. Random Sampling: Highest F1 score in each initial seed

Init Size	Highest - F1	Pool Size	Query
4	0.9813294870941099	4000	984
12	0.9823417204348603	8000	999
40	0.98180018794845	8000	966
200	0.9831825282961116	50000	991

Table 3. Random Sampling: Number of queries when the F1 score exceeds 0.9 for the first time

Init Size	F1 - 0.90	Pool Size	Query
4	0.909300349124136	1000	29
12	0.9023976452328604	1000	17
40	0.9344501746582596	1000	0
200	0.9533674586553728	1000	0

Table 4. Random Sampling: Highest F1 score in each unlabeled pool size in the initial seed 12 graph

Pool Size	Highest F1	Query
1000	0.9748526424459261	969
4000	0.9788838103232719	987
8000	0.9823417204348603	999
12000	0.9779469298607155	979
20000	0.9801597337014755	974
50000	0.9816666994192763	995

5.3.2 Uncertainty Sampling

The result of the classification uncertainty is shown in Figure 4 with detailed score comparisons available in Table 5-Table 7 For comparison, one of random sampling result is shown in the graphs.

Similar to random sampling, a larger initial seed leads to faster convergence to a higher score. Additionally, a larger pool size results in a higher maximum F1 score, demonstrating that a sufficient number of instances are required to improve the classifier's performance, and a relatively large unlabelled pool size is necessary to obtain good instances. However, the top three largest unlabelled pool sizes are slow to converge to their best score, indicating that the query strategy struggles to find good instances from the unlabelled pool. Based on these two factors, it is better to have a decent amount of unlabelled pool and many instances does not necessarily give a better score faster. The score of random sampling (8,000 unlabelled pool) is added to the graphs.

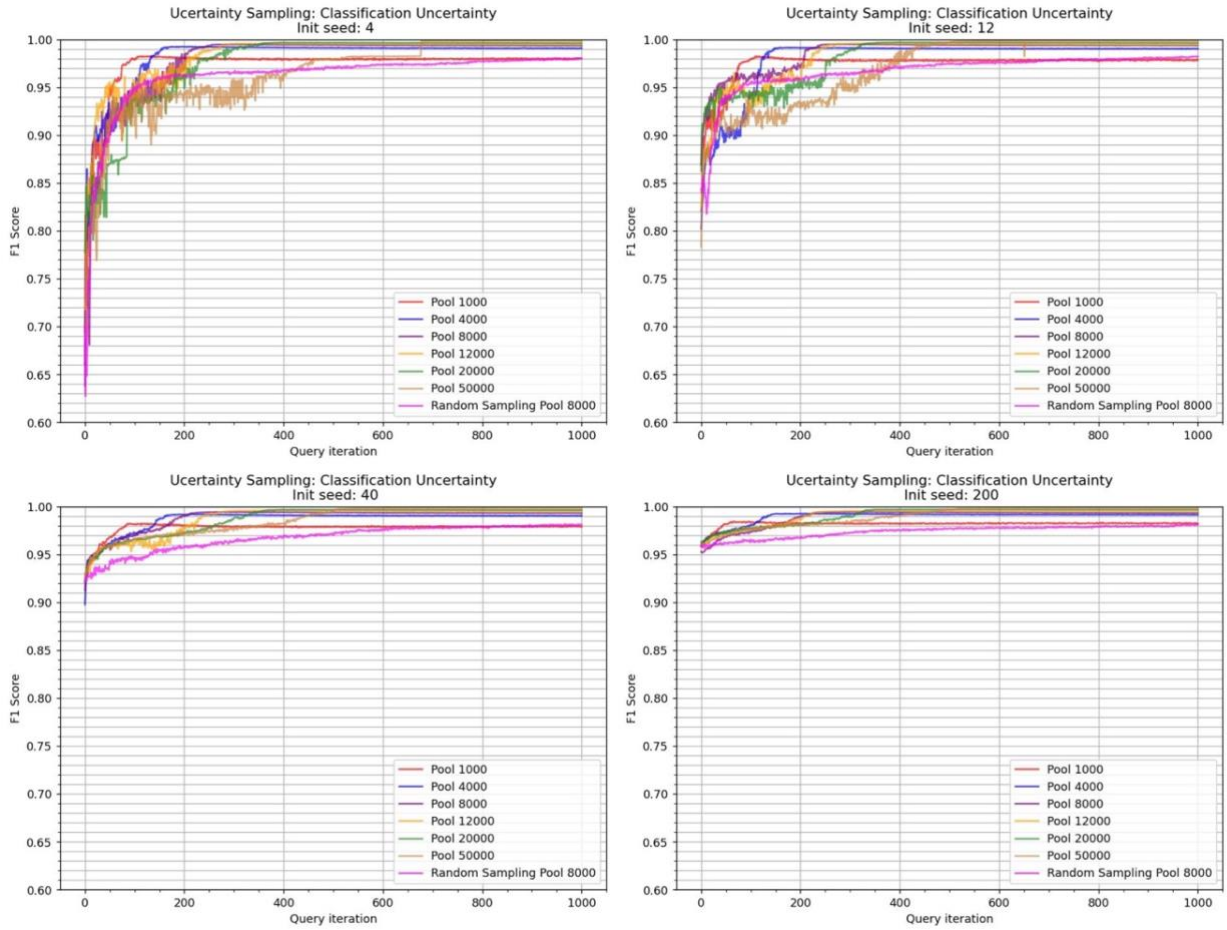

Figure 4. Uncertainty Sampling: Classification Uncertainty Result

Table 5. Uncertainty Sampling (Classification Uncertainty): Highest F1 score in each initial seed

Init Size	Highest F1	Pool Size	Query
4	0.9983503530477529	50000	680
12	0.9982203855890621	50000	795
40	0.998230338287456	50000	817
200	0.9981354874327671	50000	833

Table 6. Uncertainty Sampling (Classification Uncertainty): Number of queries when the F1 score exceeds 0.9 for the first time

Init Size	F1 - 0.90	Pool Size	Query
4	0.9010494971171464	1000	42
12	0.9035725587760008	1000	9
40	0.9035725587760008	1000	0
200	0.9585481934647515	1000	0

Table 7. Uncertainty Sampling (Classification Uncertainty): Highest F1 score in each unlabelled pool size in the initial seed 12 graph

Pool Size	Highest F1	Query
1000	0.9824130274734404	112
4000	0.9916174442184691	175
8000	0.9950015985313481	261
12000	0.9958123969302564	322
20000	0.9969513627368339	393
50000	0.9982203855890621	795

The result of classification margin is shown Figure 5 with detailed score comparisons are available in Table 8-Table 10. For comparison, one of random sampling result is shown in the graphs.

The classification margin yields slightly better results than classification uncertainty. Upon examining the graph for initial seed 4, the first four unlabelled pool sizes exceeded an F1 score of 0.95 within 100 queries, while the two largest unlabelled pool sizes took approximately 250 queries to reach the same score. In the graph for initial seed 12, all unlabelled pool sizes reached an F1 score of 0.95 within 100 queries and converged to 0.98 after 300 queries. However, Pool 1000 only reached a maximum of 0.98 for every initial seed. The graphs for initial seeds 40 and 200 show almost the same results as classification uncertainty.

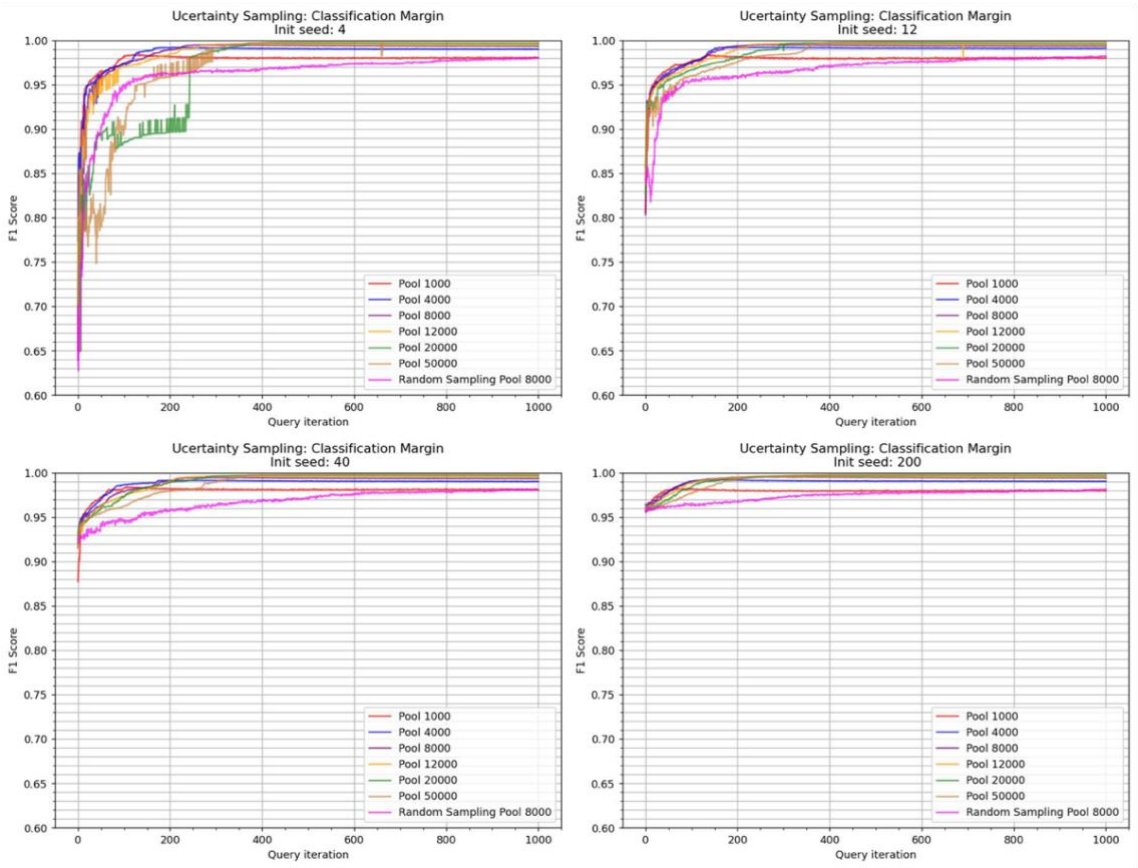

Figure 5 Uncertainty Sampling: Classification Margin Result

Table 8. Uncertainty Sampling (Classification Margin): Highest F1 score in each initial seed

Init Size	Highest F1	Pool Size	Query
4	0.9984701267019821	50000	703
12	0.9985001618738553	50000	638
40	0.998260162994467	50000	587
200	0.9984052317373571	50000	825

Table 9. Uncertainty Sampling (Classification Margin): Number of queries when the F1 score exceeds 0.9 for the first time

Init Size	F1 - 0.90	Pool Size	Query
4	0.9069369629837702	1000	7
12	0.9087198106648391	1000	6
40	0.9034405775006003	1000	2
200	0.9570015402132714	1000	0

Table 10. Uncertainty Sampling (Classification Margin): Highest F1 score in each unlabelled pool size in the initial seed 12 graph

Pool Size	Highest F1	Query
1000	0.9833623937767078	141
4000	0.9921374140806629	240
8000	0.9950421501676487	281
12000	0.9965105274983552	345
20000	0.997095587461186	457
50000	0.9985001618738553	638

The results of the classification entropy are shown in Figure 6 with detailed score comparisons are available in Table 11-Table 13. For comparison, one of random sampling result is shown in the graphs.

The graphs demonstrate that the patterns in each pool converge slower than other uncertainty sampling query strategies. In the graph with an initial seed of 4, the F1 score of pool sizes larger than 8000 fluctuates around an accuracy of 0.9 until 200 queries, after which the pool sizes converge from the smallest ones. Similarly, to other uncertainty sampling query methods, the pool size of 1000 reaches an F1 score of 0.98 at the 100th query, with the pool sizes of 4000, 8000, 12000, 20000, and 50000 converging to a score of 0.99 at 300, 350, 400, 500, and 800 queries, respectively. In the graph with an initial seed of 12, convergence occurs at the same query point; however, in the first 100 queries, the scores exceed 0.9. In the graphs with initial seeds of 40 and 200, all pool sizes converge at the same query count as the initial seeds of 4 and 12, except for pool size 50000, which converges faster than the initial seed of 40.

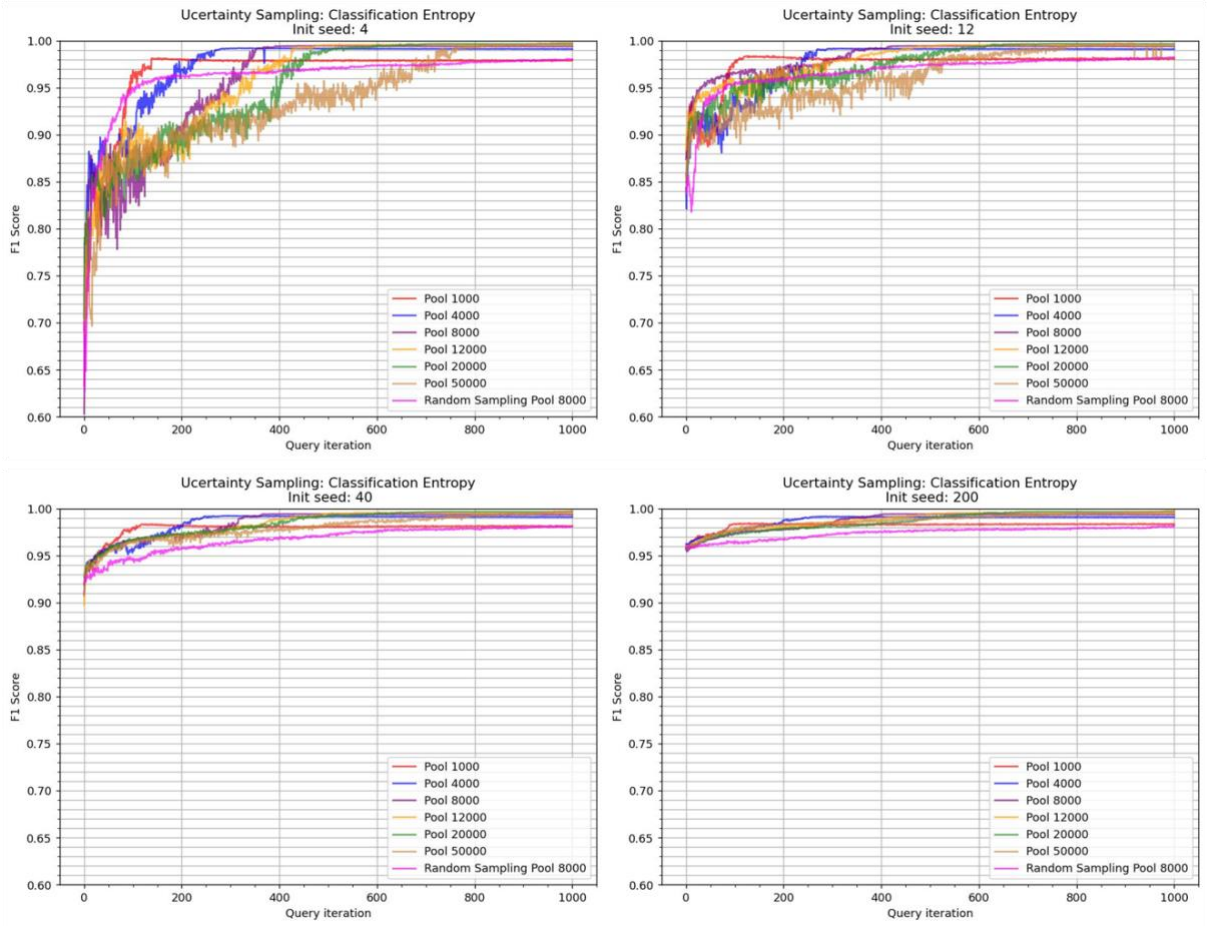

Figure 6 Uncertainty Sampling: Classification Entropy Result

Table 11. Uncertainty Sampling (Classification Entropy): Highest F1 score in each initial seed

Init Size	Highest F1	Pool Size	Query
4	0.9976158747699089	50000	990
12	0.9976260310674334	50000	973
40	0.9973859927688146	50000	996
200	0.9977458445218123	50000	998

Table 12. Uncertainty Sampling (Classification Entropy): Number of queries when the F1 score exceeds 0.9 for the first time

Init Size	F1 - 0.90	Pool Size	Query
4	0.9063903641250107	1000	73
12	0.9010464033908141	1000	4
40	0.9196395641216741	1000	0
200	0.9579836773312191	1000	0

Table 13. Uncertainty Sampling (Classification Entropy): Highest F1 score in each unlabelled pool size in the initial seed 12 graph

Pool Size	Highest F1	Query
1000	0.9837432959603376	179
4000	0.9916634552981123	304
8000	0.9944786993514766	523
12000	0.9955788457091777	544
20000	0.9967170781310735	865
50000	0.9976260310674334	973

5.3.3 Uncertainty Sampling in Binary Classification

The Figure 7 displays the outcome of uncertainty sampling in binary classification when considering classification uncertainty and detailed score comparisons are available in Table 14, Table 15, Table 16.

Regardless of the initial seed and pool size, the maximum F1 score is about 0.99. Comparing the graphs of binary and multi-class result with the same strategy, we observe that binary classification outperforms multi-class classification, as it only takes about 100 instances to exceed the F1 score of 0.98 regardless of unlabeled pool size.

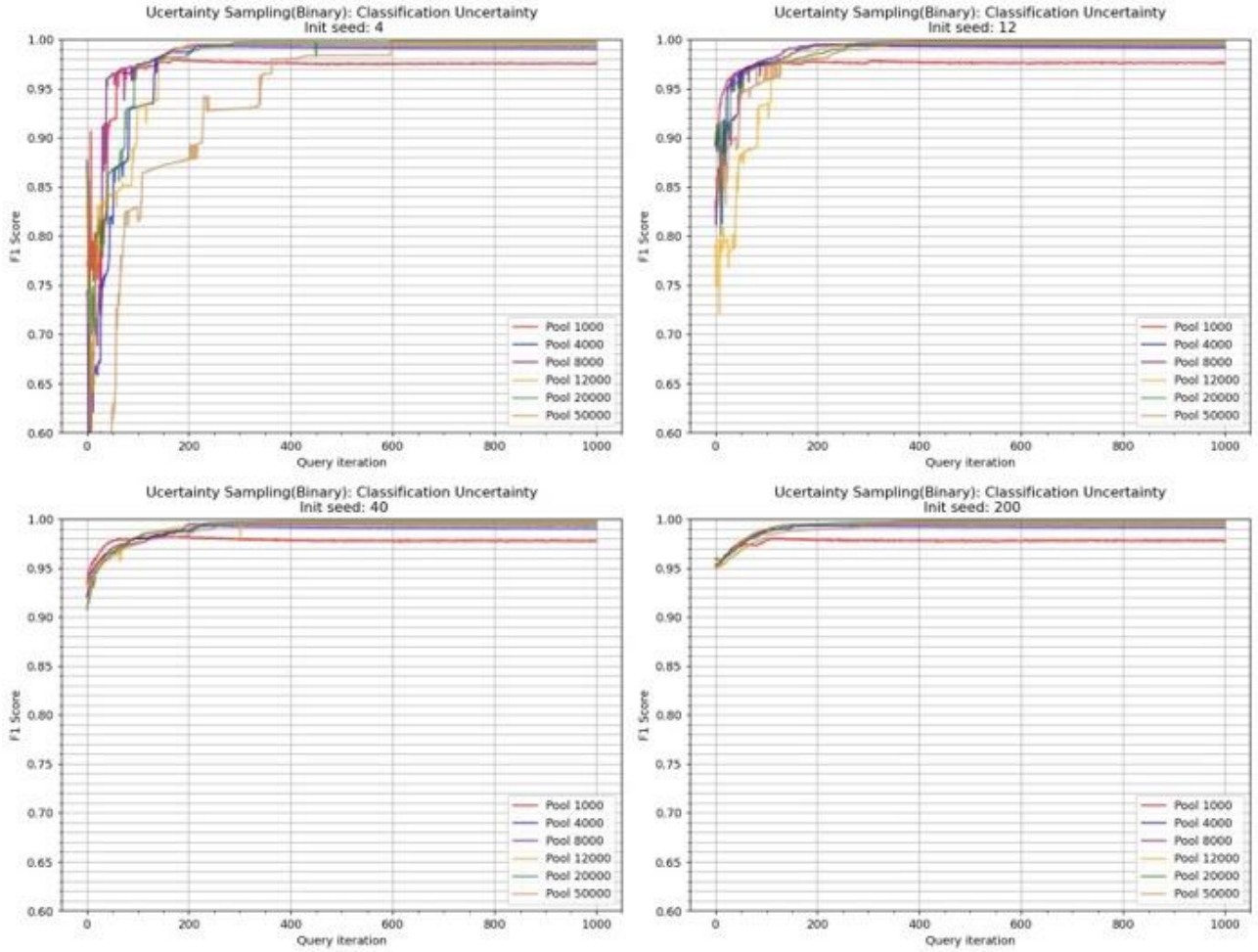

Figure 7. Uncertainty Sampling (Binary) Result

Table 14. Binary Uncertainty Sampling (Classification Uncertainty): Highest F1 score in each initial seed

Init Size	Highest F1	Pool Size	Query
4	0.998231819406671	50000	631
12	0.9984212978833014	50000	551
40	0.9982966599433704	50000	479
200	0.9984561954062894	50000	585

Table 15. Binary Uncertainty Sampling (Classification Uncertainty): Number of queries when the F1 score exceeds 0.9 for the first time

Init Size	F1 - 0.90	Pool Size	Query
4	0.9061612174750495	1000	7
12	0.905154323766163	1000	1
40	0.9328065825179648	1000	0
200	0.9508481780305941	1000	0

Table 16. Binary Uncertainty Sampling (Classification Uncertainty): Highest F1 score in each unlabelled pool size in the initial seed 12 graph

Pool Size	Highest F1	Query
1000	0.9785984981920983	311
4000	0.9940898618542867	214
8000	0.9950334361671882	272
12000	0.9962828343934642	270
20000	0.9973189676979078	348
50000	0.9984212978833014	551

5.3.4 Ranked Batch-mode Sampling

For the experiment, 4, 8, 20, and 40 batch instances were used with the same initial seed and pool patterns as the other sampling method. The results for 4 batch instances and an initial seed of 12 instances for 8, 20, and 40 batch instances are presented in this paragraph. In addition, the result of random sampling is added to the graph for comparison. The query iteration was adjusted according to the ranked batch-mode sampling's iteration. (i.e., as the 4 batch instances query iteration being the average of every 4 iterations in random sampling)

The result of 4 batch instances is shown Figure 8 with detailed score comparisons are available in Table 17-Table 20

The first assumption was that a larger unlabelled pool size would result in better convergence of the F1 score. However, in this sampling method, larger pool sizes are struggling to achieve better results at maximum overall.

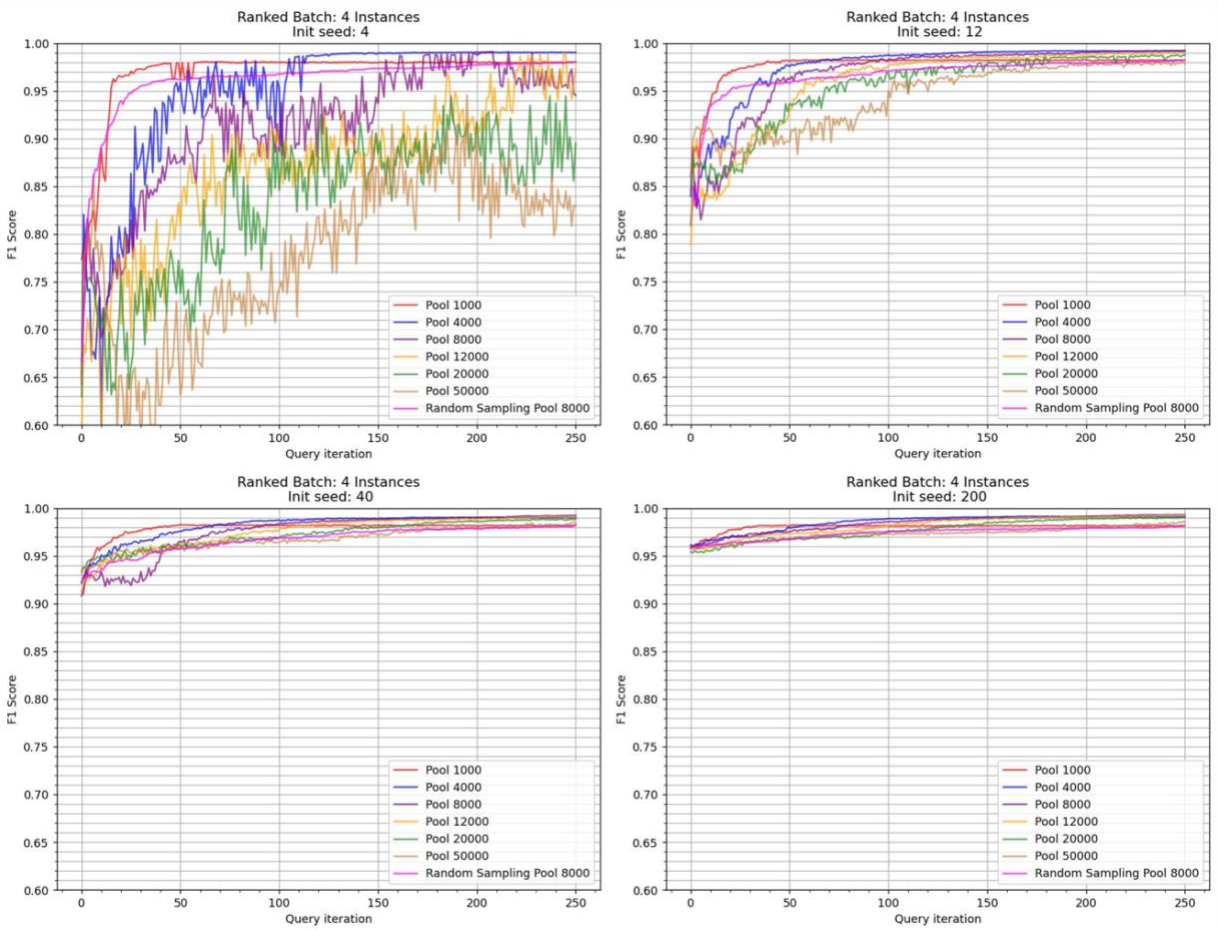

Figure 8. Ranked Batch-mode Sampling: 4 Batch Instances Result

Table 17. Ranked Batch-mode Sampling (4 batch instances): Highest F1 score in each initial seed

Init Size	Highest F1	Pool Size	Query
4	0.9910911131668874	8000	832
12	0.992162100783205	8000	996
40	0.9923571561616351	8000	996
200	0.9931023423066968	8000	992

Table 18. Ranked Batch-mode Sampling (4 batch instances): Number of queries when the F1 score exceeds 0.9 for the first time

Init Size	F1 - 0.90	Pool Size	Query
4	0.905225183697479	1000	52
12	0.9052755351931431	1000	20
40	0.9116688996579242	1000	0
200	0.9580390100552684	1000	0

Table 19. Ranked Batch-mode Sampling (4 batch instances): Highest F1 score in each unlabelled pool size in the initial seed 12 graph

Pool Size	Highest F1	Query
1000	0.9826777221322175	796
4000	0.9919784843142804	736
8000	0.992162100783205	996
12000	0.9903267751770748	1000
20000	0.9877460414451813	1000
50000	0.9798412308506468	996

Table 20. Ranked Batch-mode Sampling (4 batch instances): Number of queries when the F1 score exceeds 0.98 for the first time in the initial seed 12 graph

Pool Size	Highest F1	Query
1000	0.9826777221322175	796
4000	0.9919784843142804	736
8000	0.992162100783205	996
12000	0.9903267751770748	1000
20000	0.9877460414451813	1000
50000	0.9798412308506468	996

For comparison, the result of 8, 20 and 40 batch instances are shown in Figure 9. Detailed score comparisons are shown in Table 21-Table 26.

As the initial seed of instances increases, the difference in scores becomes smaller. However, larger unlabelled pool sizes yield slightly lower scores compared to smaller pool sizes, and this is shown in other batch instance settings. For instance, in the case of initial seed of 12 instances and a pool size of 1000, a F1 score of 0.97-0.98 is achieved within 50 query iterations in a 4 batch instances setting, and at around 25, 10, and 5 iterations for 8, 20, and 40 batch instance settings, respectively. This means that about 200 instances are required to achieve the same score. When comparing bigger pool sizes, a pool size of 50000 requires around 800 instances to achieve the maximum score (0.98). This proves that the size of batch instances affects little the required number of instances to achieve a certain score.

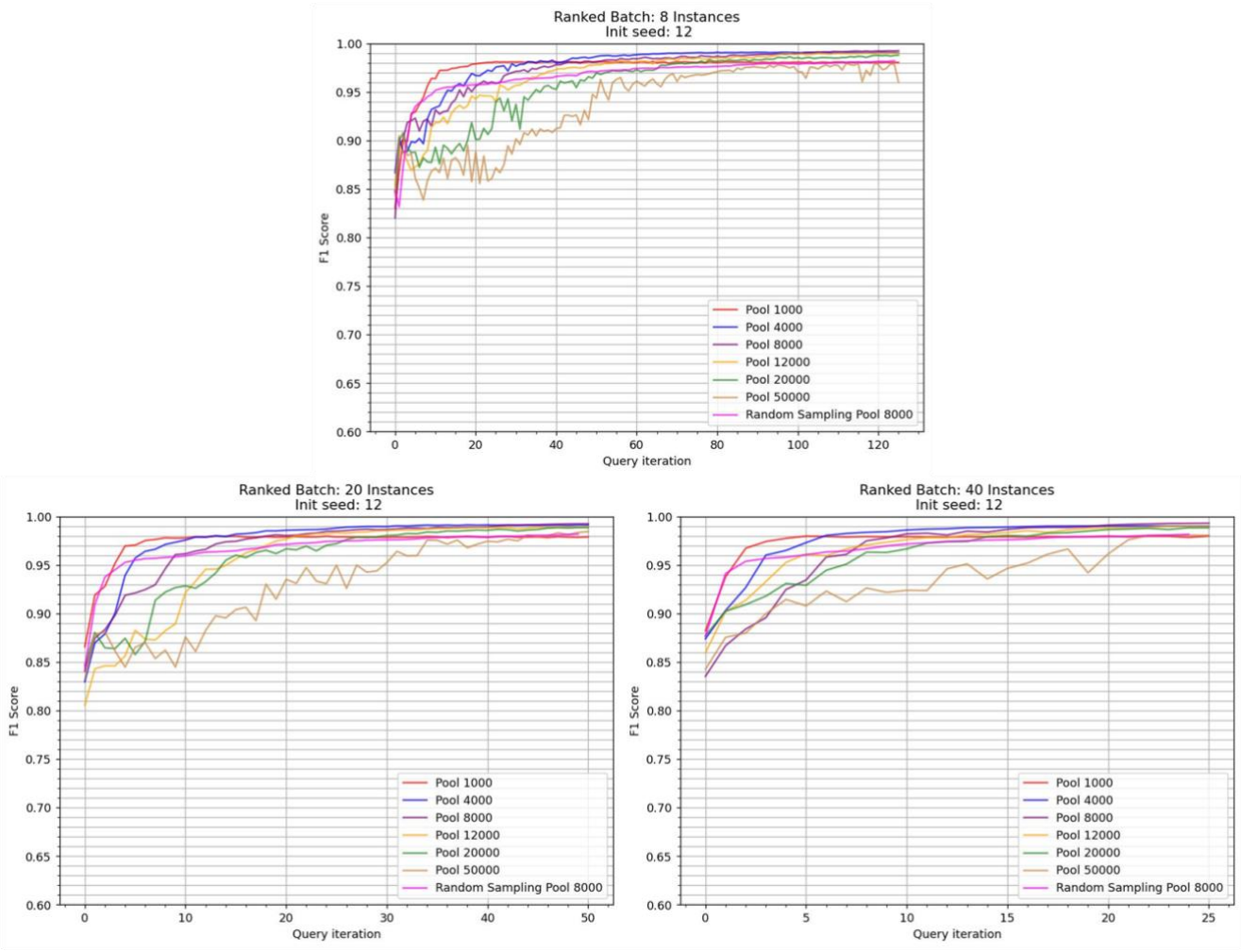

Figure 9. Ranked Batch-mod Sampling: 8, 20, 40 Batch Instances

Table 21. Ranked Batch-mode Sampling (8 batch instances): Highest F1 score in each unlabelled pool size in the initial seed 12 graph

Pool Size	Highest F1	Query
1000	0.9813593164398956	512
4000	0.9910758000395342	840
8000	0.9923730757998293	1000
12000	0.9902893132658521	1000
20000	0.9877726700859174	1000
50000	0.9793489498678432	896

Table 22. Ranked Batch-mode Sampling (20 batch instances): Highest F1 score in each unlabelled pool size in the initial seed 12 graph

Pool Size	Highest F1	Query
1000	0.9797463777346567	260
4000	0.9912853350063205	760
8000	0.9924411462660337	1000
12000	0.9894953400067024	1000
20000	0.9885315401864843	980
50000	0.9843409895761951	1000

Table 23. Ranked Batch-mode Sampling (40 batch instances): Highest F1 score in each unlabelled pool size in the initial seed 12 graph

Pool Size	Highest F1	Query
1000	0.9798222453258779	1000
4000	0.9902691810068175	800
8000	0.9930345554336146	1000
12000	0.9900777985430272	920
20000	0.9882548057195603	1000
50000	0.9807379341098184	960

Table 24. Ranked Batch-mode Sampling (8 batch instances): Number of queries when the F1 score exceeds 0.98 for the first time in the initial seed 12 graph

Pool Size	F1 – 0.98	Query
1000	0.9802991896265922	176
4000	0.9811448534555302	264
8000	0.9805949323907892	344
12000	0.9805823476731955	440
20000	0.9818069249817116	608
50000		

Table 25. Ranked Batch-mode Sampling (20 batch instances): Number of queries when the F1 score exceeds 0.98 for the first time in the initial seed 12 graph

Pool Size	F1 – 0.98	Query
1000		
4000	0.980022017608072	260
8000	0.9810463337800133	380
12000	0.9801621665172174	420
20000	0.9803308543281097	600
50000	0.9806706983402392	880

Table 26. Ranked Batch-mode Sampling (40 batch instances): Number of queries when the F1 score exceeds 0.98 for the first time in the initial seed 12 graph

Pool Size	F1 – 0.98	Query
1000		
4000	0.9804330267185175	240
8000	0.9820505547055909	400
12000	0.9812469271820013	520
20000	0.9803271269432401	600
50000	0.9802889969008526	880

5.3.5 Query by Committee

Based on convergence speed and the maximum F1 score from the result of uncertainty sampling, a pool size of 8000 unlabelled samples yielded better results and was therefore chosen as the fixed unlabelled pool size setting. For comparison, one of random sampling result is shown in the graphs.

The result of vote entropy is shown Figure 10 with detailed score comparisons are available in Table 27-Table 29. What is common across all four graphs is that they converge to an F1 score of 0.99 at around 200 queries. The graph for the initial seed of 4, before 200 queries, illustrates that the number of committees does not guarantee quick convergence. VE2, 5, and 7 reach 0.95 at around 100 queries, while VE3 and 10 struggle to achieve 0.85 to 0.9. However, as the initial seed increases, the lines on the graph gradually overlap. For instance, in the graph for the initial seed of 12, the difference between the lines is smaller than the graph for the initial seed of 4 before 200 queries. In the graphs for the initial seed of 40 and 200, the lines almost completely overlap.

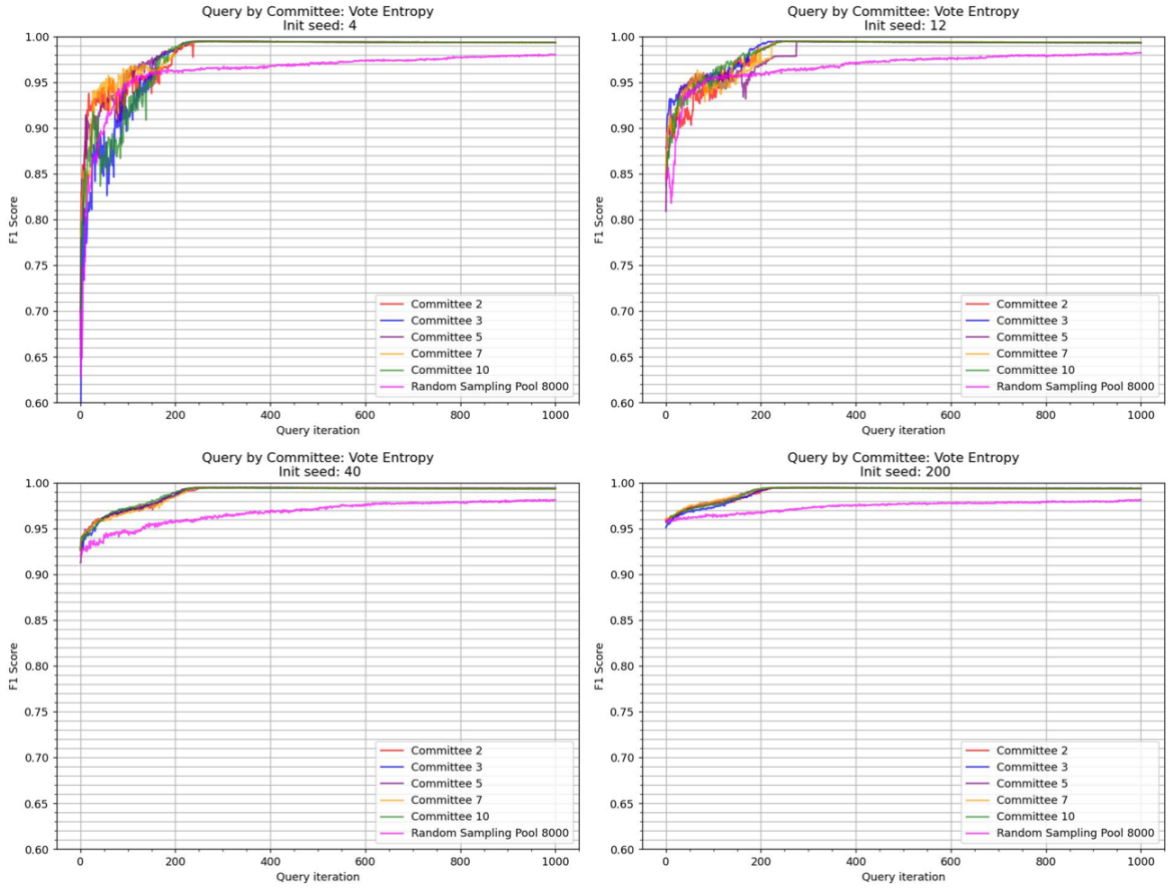

Figure 10. Query by Committee: Vote Entropy Result

Table 27. Query by Committee (Vote Entropy): Highest F1 score in each initial seed

Init Size	Highest F1	Committee Size	Query
4	0.9948018012713729	2	288
12	0.994872516729363	7	288
40	0.9951078454749875	3	285
200	0.994662889185015	10	275

Table 28. Query by Committee (Vote Entropy): Number of queries when the F1 score exceeds 0.9 for the first time

Init Size	F1 - 0.90	Committee Size	Query
4	0.9111441171973421	2	13
12	0.9021925918929927	2	5
40	0.9266248594862517	2	0
200	0.9596037195310242	2	0

Table 29. Query by Committee (Vote Entropy): Highest F1 score in each committee size in the initial seed 12 graph

Committee Size	Highest F1	Query
2	0.9947018408442376	251
3	0.9948584732687047	233
5	0.9945132815371533	298
7	0.994872516729363	288
10	0.9948471589479071	271

The result of consensus entropy is shown Figure 11 with detailed score comparisons are available in Table 30-Table 32.

Similar to the vote entropy, the lines become similar as the initial seeds increase in size. The F1 scores converge to 0.99 after approximately 400 queries, which is 200 more queries than the vote entropy result. There is no regular pattern observed in the number of committees, as the order of F1 scores changes in every graph.

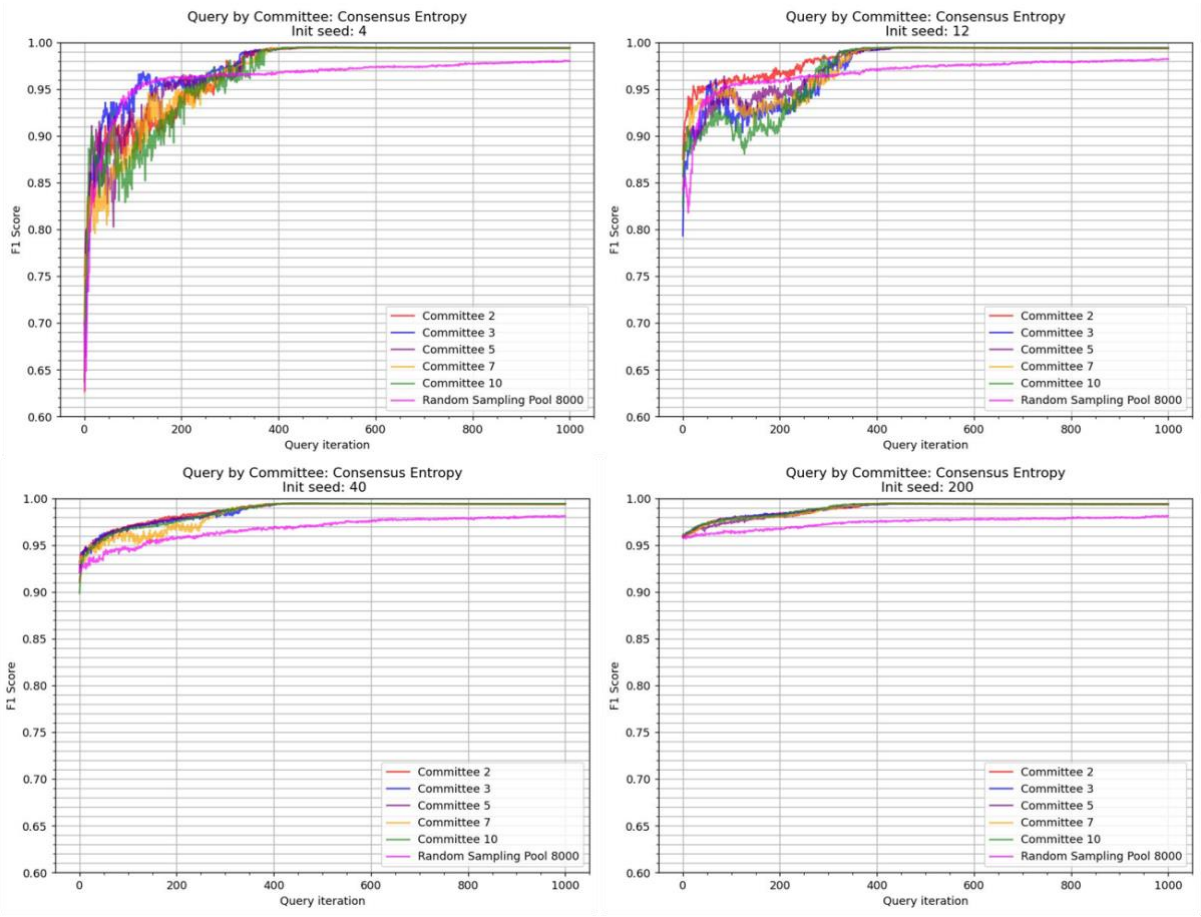

Figure 11. Query by Committee: Consensus Entropy Result

Table 30. Query by Committee (Consensus Entropy): Highest F1 score in each initial seed

Init Size	Highest F1	Committee Size	Query
4	0.9946487613733987	3	520
12	0.9946203939612873	7	442
40	0.9947083729348851	10	476
200	0.994733740559116	7	492

Table 31. Query by Committee (Consensus Entropy): Number of queries when the F1 score exceeds 0.9 for the first time

Init Size	F1 - 0.90	Committee Size	Query
4	0.9033914867143871	2	29
12	0.9045590466658009	2	2
40	0.910854501435472	2	0
200	0.9579881842439683	2	0

Table 32. Query by Committee (Consensus Entropy): Highest F1 score in each committee size in the initial seed 12 graph

Committee Size	Highest F1	Query
2	0.9940850074287189	458
3	0.9945303725806613	478
5	0.9943653021091476	481
7	0.9944242585595904	486
10	0.9946203939612873	442

The result of max disagreement is shown Figure 12 with detailed score comparisons are available in Table 33-Table 35.

For the graph with an initial seed of 4 and a committee of 7 and 10, the score reaches 0.95 after 200 queries. For the committee of 3 and 5, the score reaches 0.95 after 250 and 500 queries, respectively. The committee of 2 starts decreasing the score within 50 queries, regardless of the initial seed. However, other than the committee of 2, all the other committees converge to a score of 0.99 after 1000 queries, which is the highest number of required queries among all the committee strategies.

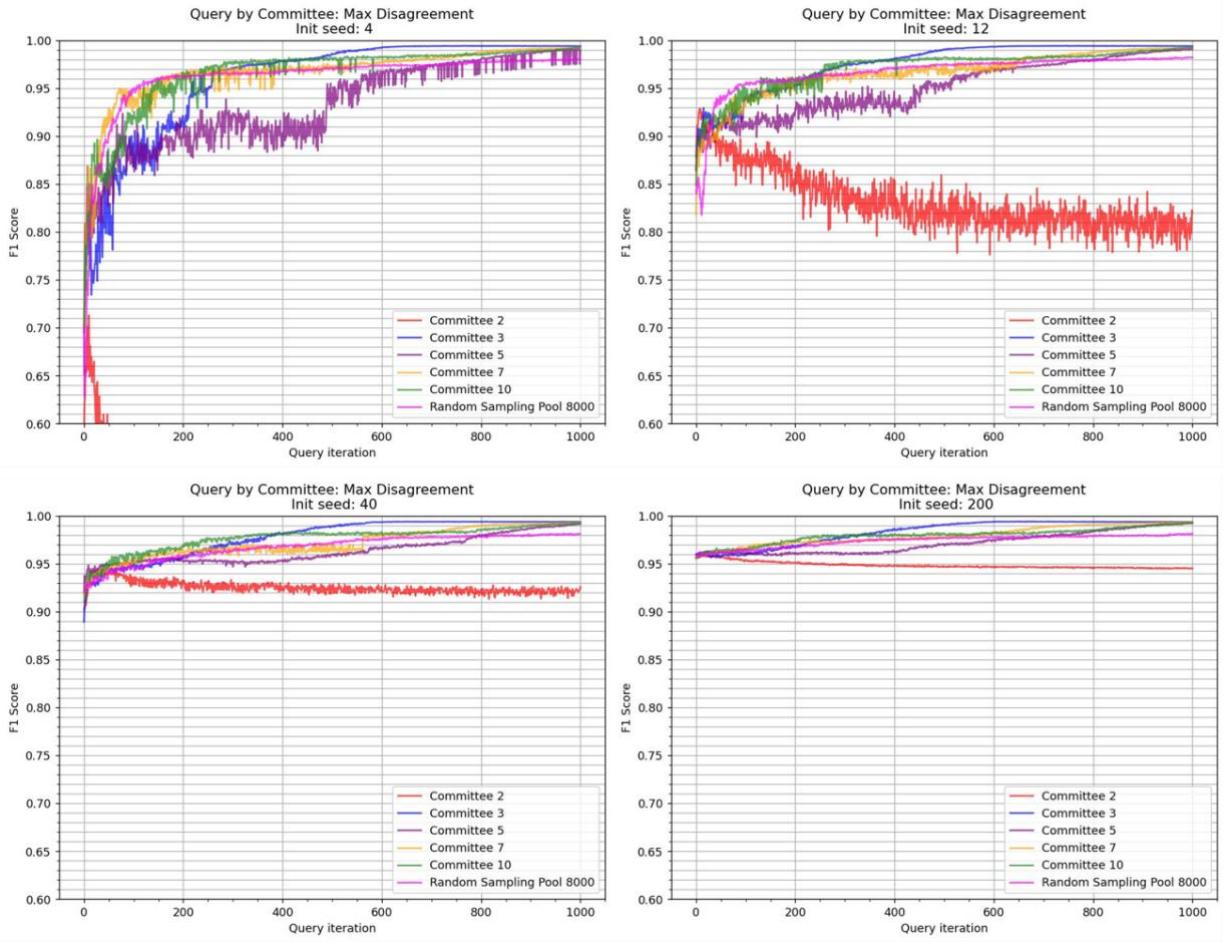

Figure 12. Query by Committee: Max Disagreement Res

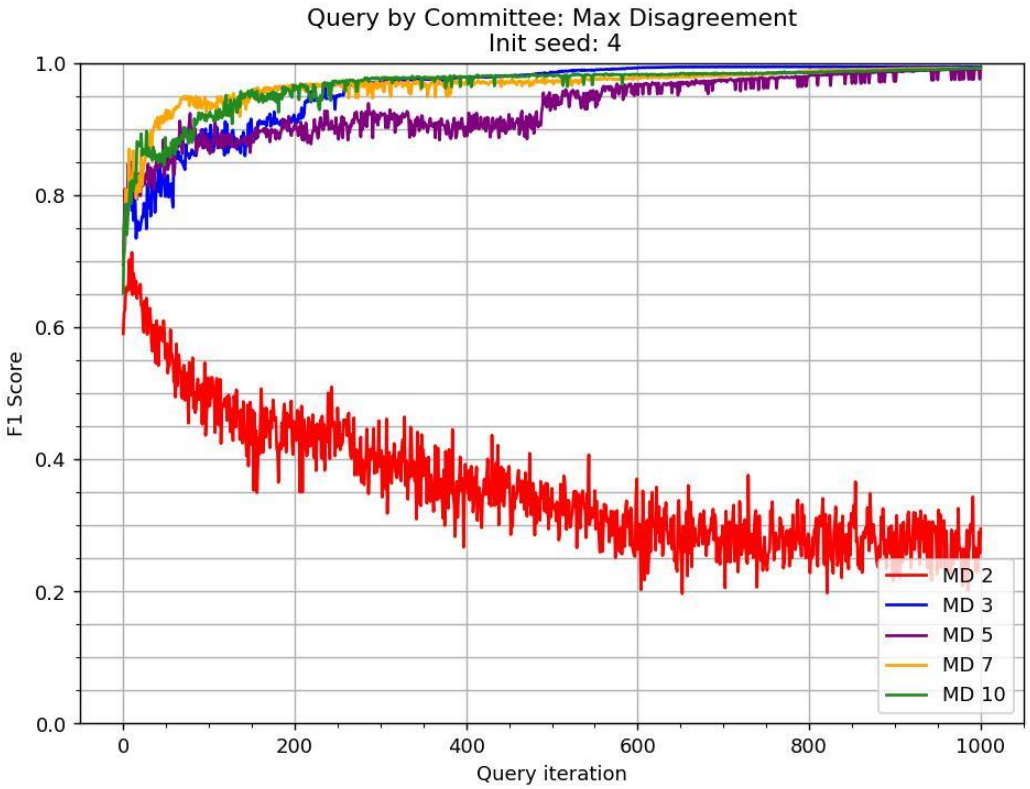

Figure 13. Query by Committee: Max Disagreement - Init Seed 4 Zoom Out

Table 33. Query by Committee (Max Disagreement): Highest F1 score in each initial seed

Init Size	Highest F1	Committee Size	Query
4	0.9941895399314195	3	766
12	0.9942431595378005	3	838
40	0.9939685928428403	3	667
200	0.9941053013205924	3	710

Table 34. Query by Committee (Max Disagreement): Number of queries when the F1 score exceeds 0.9 for the first time

Init Size	F1 - 0.90	Committee Size	Query
4	0.905722936372144	3	102
12	0.9105171132658125	2	2
40	0.9017355476087138	2	0
200	0.955866533202794	2	0

Table 35. Query by Committee (Max Disagreement): Highest F1 score in each committee size in the initial seed 12 graph

Committee Size	Highest F1	Query
2	0.9285285007431217	7
3	0.9942431595378005	838
5	0.9905648100313555	999
7	0.9921420613266317	991
10	0.9921999105118665	990

5.4 N-BaIoT Dataset Testing

As a query strategy, the classification margin was utilized in the experiment due to its superior performance in terms of convergence speed and F1 scores across all aspects.

5.4.1 Random Sampling

The result of random sampling is shown in Figure 14 with detailed score comparisons are available in Table 36, Table 37 and Table 38. The results show that the highest F1 score is 0.996, indicating that even random sampling with the dataset can achieve a relatively high score.

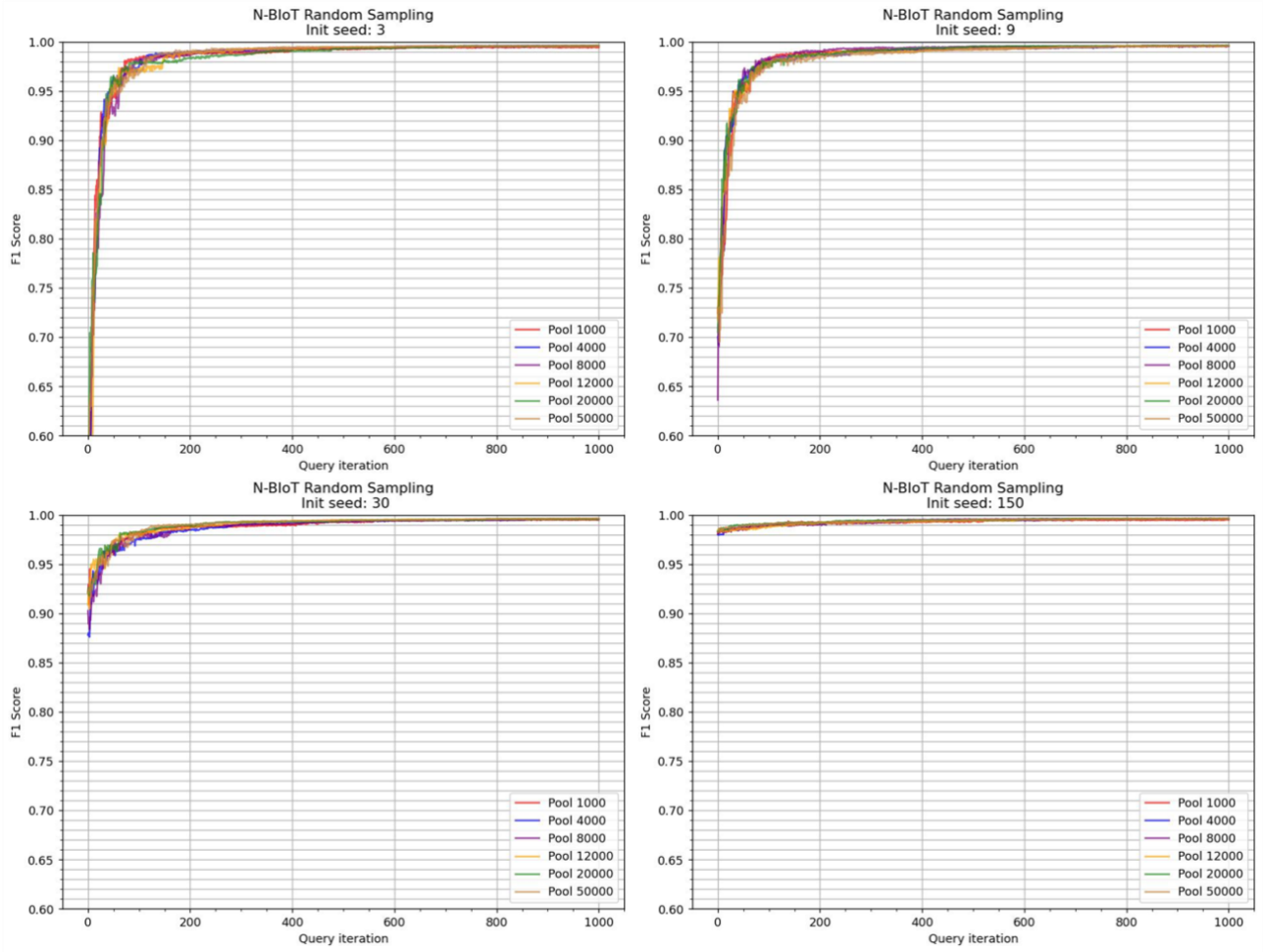

Figure 14. Random Samling: Tested with N-BaIoT Dataset Result

Table 36. Random Sampling(N-BaIoT): Highest F1 score in each initial seed

Init Size	Highest F1	Pool Size	Query
3	0.9961879549487392	8000	992
9	0.9964345892048223	20000	978
30	0.9966277517194528	50000	956
150	0.9966409716357998	8000	867

Table 37. Random Sampling(N-BaIoT): Number of queries when the F1 score exceeds 0.9 for the first time

Init Size	F1 - 0.90	Pool Size	Query
3	0.9037804393334439	1000	23
9	0.905120209649412	1000	27
30	0.9219930851667533	1000	0
150	0.981003072793575	1000	0

Table 38. Random Sampling(N-BaIoT): Highest F1 score in each unlabelled pool in initial seed 9 graph.

Pool Size	Highest F1	Query
1000	0.9956143723434732	873
4000	0.996327078444428	967
8000	0.9962871617858541	951
12000	0.9964345892048223	984
20000	0.9962474910391782	978
50000	0.996327078444428	9939

5.4.2 Uncertainty Sampling: Classification Margin

The results are presented in Figure 15, with detailed score comparisons available in Tables 38 and 39. For comparison, one of random sampling result is shown in the graphs.

The graphs show that as the initial seed increases, the learning curve becomes steeper and as the pool size increases, convergence becomes slower. However, Table 39 indicates that larger pool sizes achieve better scores than smaller pool sizes.

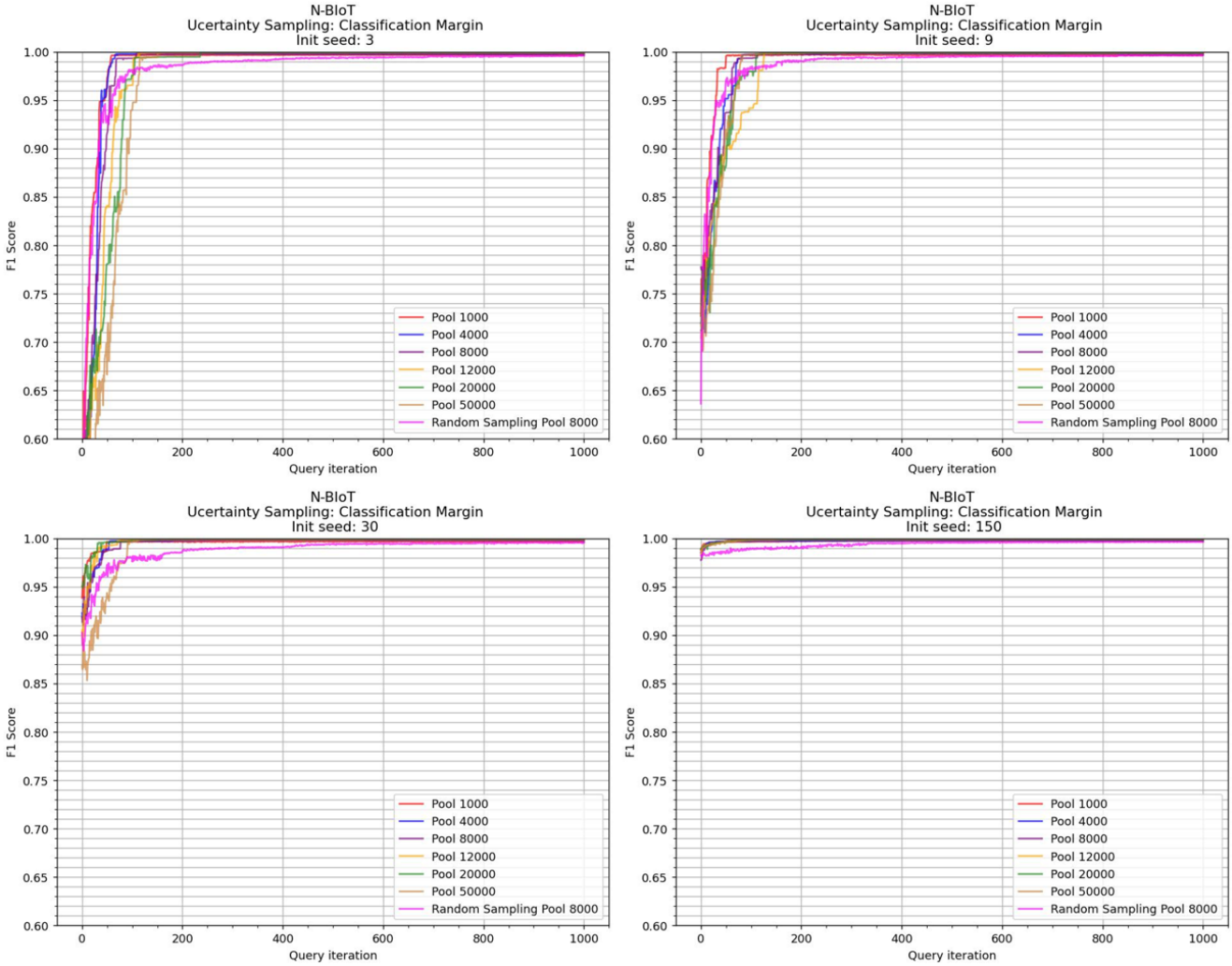

Figure 15. Uncertainty Sampling (Classification Margin): Tested with N-BaIoT Dataset Result

Table 39. Uncertainty Sampling (Classification Margin): Tested with N-BaIoT Dataset
Highest F1 score in each initial seed.

Init Size	Highest F1	Pool Size	Query
3	0.9996333559548909	50000	317
9	0.9996200233004562	50000	256
30	0.9996600199763881	50000	508
150	0.9995667046045353	50000	571

Table 40. Uncertainty Sampling (Classification Margin): Number of queries when the F1 score exceeds 0.9 for the first time

Init Size	F1 - 0.90	Pool Size	Query
3	0.9381713791941528	1000	34
9	0.9031374172914581	1000	20
30	0.9384857330735608	1000	0
150	0.9879379649478708	1000	0

Table 41. Uncertainty Sampling (Classification Margin): Tested with N-BaIoT Dataset
Highest F1 score in each unlabelled pool in initial seed 9 graph.

Pool Size	Highest F1	Query
1000	0.9833623937767078	141
4000	0.9921374140806629	240
8000	0.9950421501676487	281
12000	0.9965105274983552	345
20000	0.997095587461186	457
50000	0.9985001618738553	638

5.4.3 Query by Committee: Vote Entropy

The results are presented in Figure 16, with detailed score comparisons available in Table 42 and Table 43. For comparison, one of random sampling result is shown in the graphs.

The results show that the F1 score is below 0.75 when the initial seeds are 3 and 9, and it improves as the number of initial seeds increases. However, even when the initial seed is 150, it never outperforms the random sampling score.

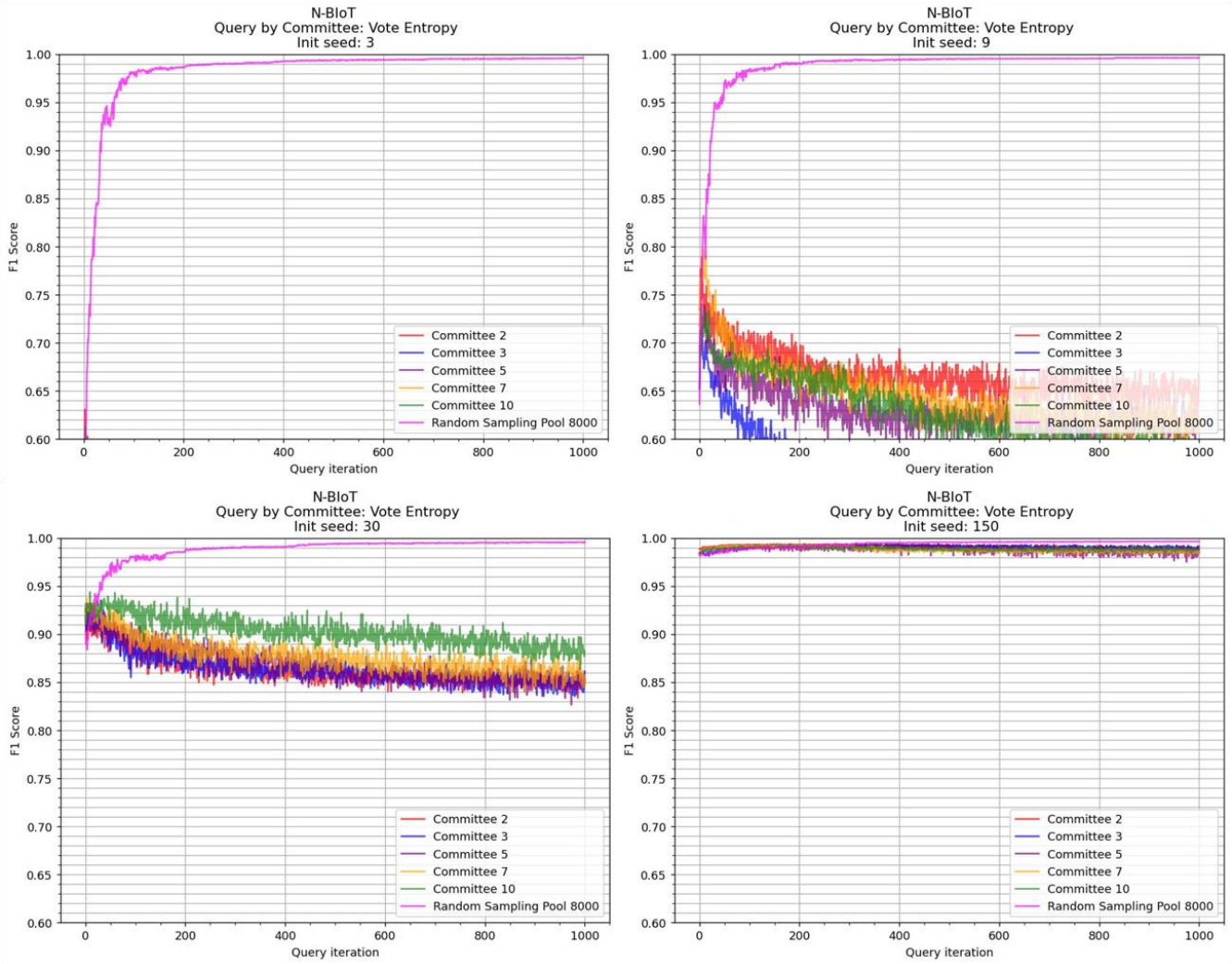

Figure 16. Query by Committee (Vote Entropy): Tested with N-BaIoT Dataset Result

Table 42. Query by Committee (Vote Entropy): Tested with N-BaIoT Dataset
Highest F1 score in each initial seed.

Init Size	Highest F1	Committee Size	Query
3	0.6307305373602503	2	2
9	0.7970691298295212	7	8
30	0.9437839982165437	10	9
150	0.9936660871902324	7	236

Table 43. Query by Committee (Vote Entropy): Tested with N-BaIoT Dataset
Highest F1 score in each committee size in initial seed 9

Committee Size	Highest F1	Query
2	0.7891392756775394	5
3	0.7168013773921715	2
5	0.7399560692484599	4
7	0.7970691298295212	8
10	0.7349928738354503	9

5.4.4 Ranked Batch-mode Sampling

The results are presented in Figure 17, with detailed score comparisons available in Tables 40 and 41. For comparison, one of the random sampling results is shown in the graphs. The scores were averaged per batch number to align query lengths. The results show that the larger the initial size, the more stable the learning curve and the better the score. However, when comparing the unlabelled pool sizes, the scores of larger unlabelled pool sizes struggle to perform better than smaller unlabelled pool sizes.

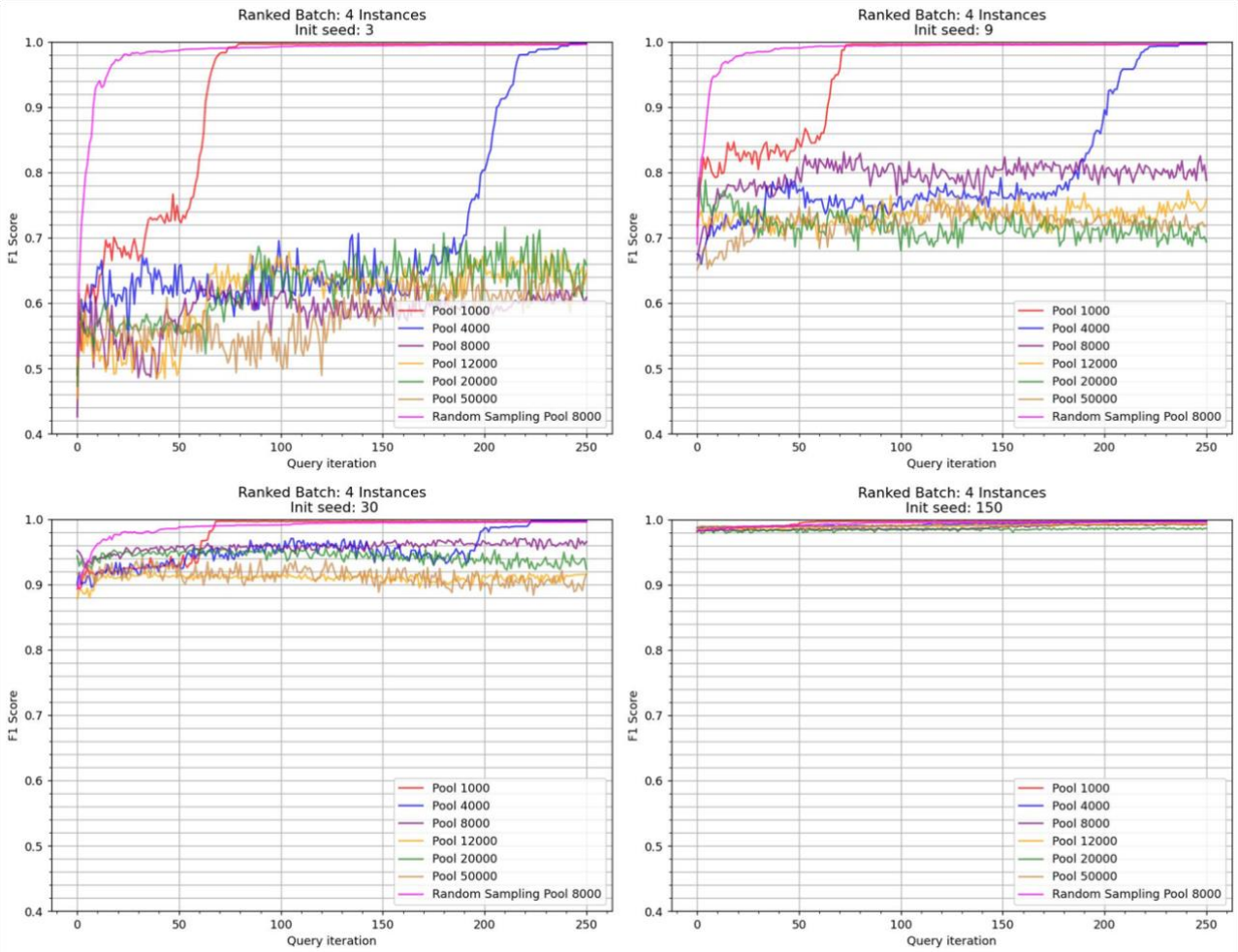

Figure 17. Ranked Batch-mode Sampling (4 Batch Instances): Tested with N-BaIoT Dataset Result

Table 44. Ranked Batch-mode Sampling (4 Batch Instances): Tested with N-BaIoT Dataset Result Highest F1 score in each initial seed.

Init Size	Highest F1	Pool Size	Query
3	0.9979673151737792	4000	980
9	0.9977672419904244	4000	968
12	0.9977142992990666	4000	956
150	0.9976074328568284	4000	964

Table 45. Ranked Batch-mode Sampling (4 Batch Instances): Tested with N-BaIoT Dataset Result Number of queries when the F1 score exceeds 0.9 for the first time

Init Size	F1 - 0.90	Pool Size	Query
3	0.9084481347762257	1000	252
9	0.9008822910442262	1000	256
30	0.9164568491784395	1000	3
150	0.9837004186524221	1000	0

Table 46. Ranked Batch-mode Sampling (4 Batch Instances): Tested with N-BaIoT Dataset Result

Highest F1 score in each unlabelled pool in initial seed 9 graph.

Pool Size	Highest F1	Query
1000	0.9961946165838287	564
4000	0.9977672419904244	968
8000	0.8311392246500559	288
12000	0.7722355456309083	964
20000	0.7919358158848283	4
50000	0.7685277299347156	476

6 Discussion

This study conducted a benchmark test of multi-class classification for botnets using active learning. The test environment was set up with a combination of labelled and unlabelled data, where the unlabelled data could be accurately labelled by a human expert.

6.1 Uncertainty Sampling

Among all query strategies, active learners with uncertainty sampling show the most stable and effective results, as shown in Figure 18. In particular, margin sampling outperforms other strategies and even surpasses other active learners with the largest initial seed setting when using the smallest initial seed setting.

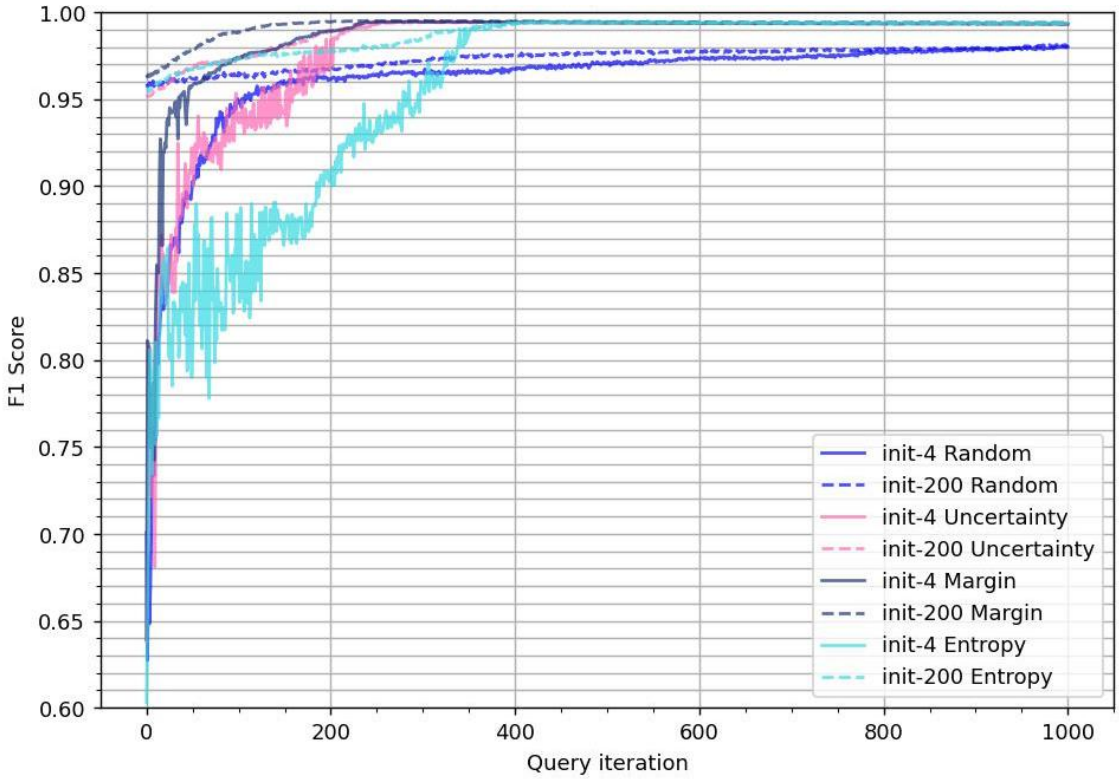

Figure 18. Comparison of Random Sampling and Uncertainty Sampling

6.2 Query by Committee

Regarding query by committee sampling, the results in section 5.3.5 indicate that a larger unlabelled pool size leads to better convergence speed, similar to other query strategies. However, the effect of the number of committees cannot be observed from

the results, as there are cases where a higher number of committees does not necessarily result in a better trend.

Comparison of random sampling and query by committee (5 committee) is shown in Figure 19. In terms of stability and convergence speed, the comparison shows that vote entropy produced the best results among all query by committee samplings, with consensus entropy in second place and max disagreement in last place.

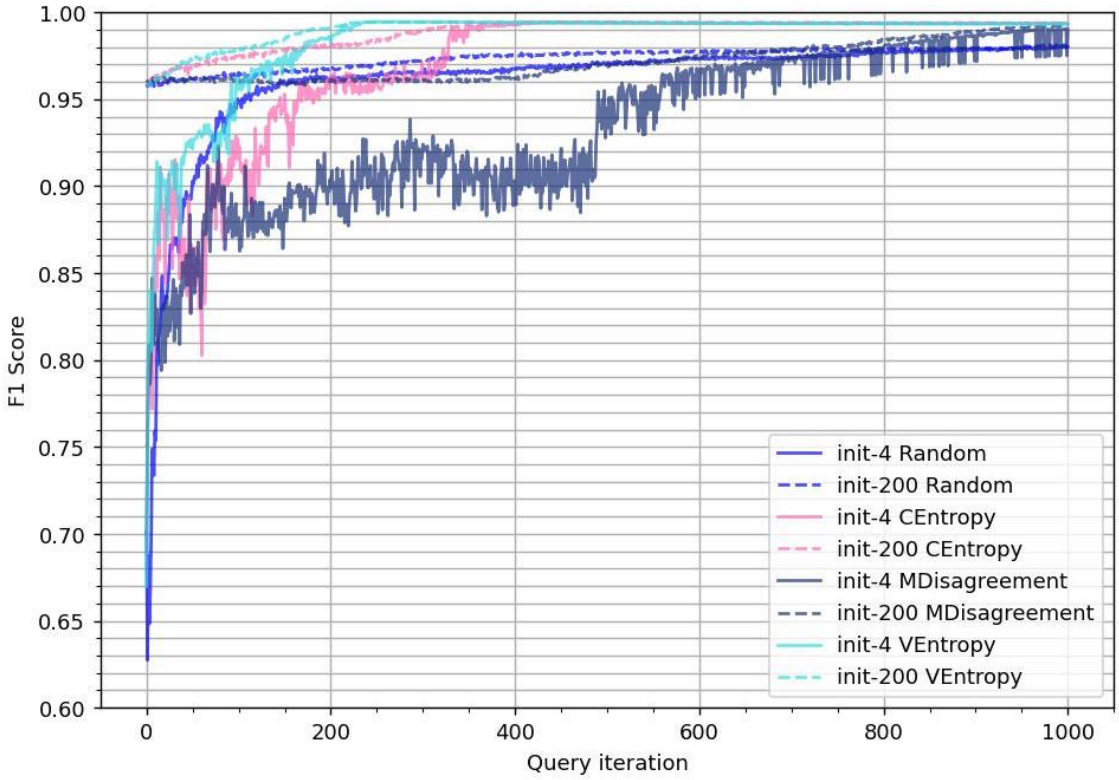

Figure 19. Comparison of Random Sampling and Query by Committee

6.3 Comparison of Uncertainty Sampling and Query by Committee

Both Query by Committee and Uncertainty Sampling produce one query output per active learning iteration. This makes it important to compare the two strategies.

The results of the comparison between uncertainty sampling and query by committee are displayed in Figure 20. Classification margin and vote entropy (10 committees) are used for each strategy and 8,000 unlabelled pool size is used as default setting. Initially, we assumed that query by committee would outperform uncertainty sampling due to its multiple query strategies. However, as depicted in the graph, the classification margin

demonstrates a steeper trend than the vote entropy. This proves that classification margin outperforms vote entropy.

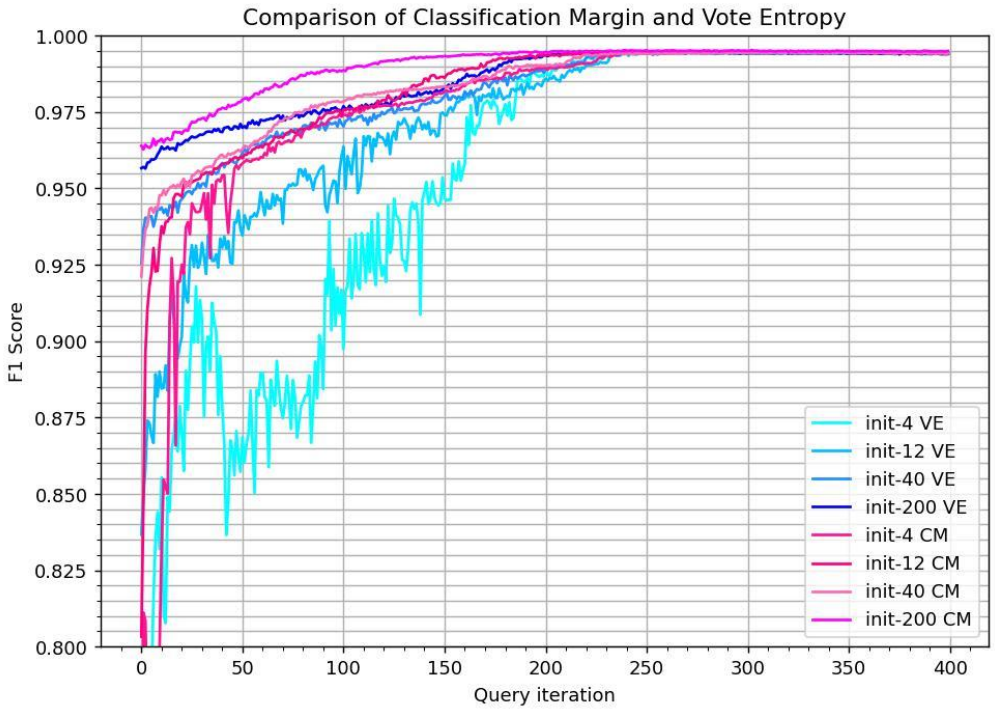

Figure 20. Comparison of Vote Entropy and Classification Margin

6.4 Ranked Batch-mode Sampling

One key characteristic of ranked batch-mode sampling is that the batch size has little effect on the number of queries required to achieve a certain score. Another characteristic is that an active learner with a smaller pool size tends to perform better than one with a larger pool size. shows a comparison of random sampling and ranked batch-mode sampling results with different initial seed sizes (4, 12, 200) and unlabelled pool sizes (1000, 50000). This comparison demonstrates this characteristic. Thus, using an adequate initial seed size and a small unlabelled pool size can lead to better outcomes if the team's objective is to prioritize teamwork, considering that a relatively small unlabelled pool outperforms random sampling and a small initial seed converges slowly.

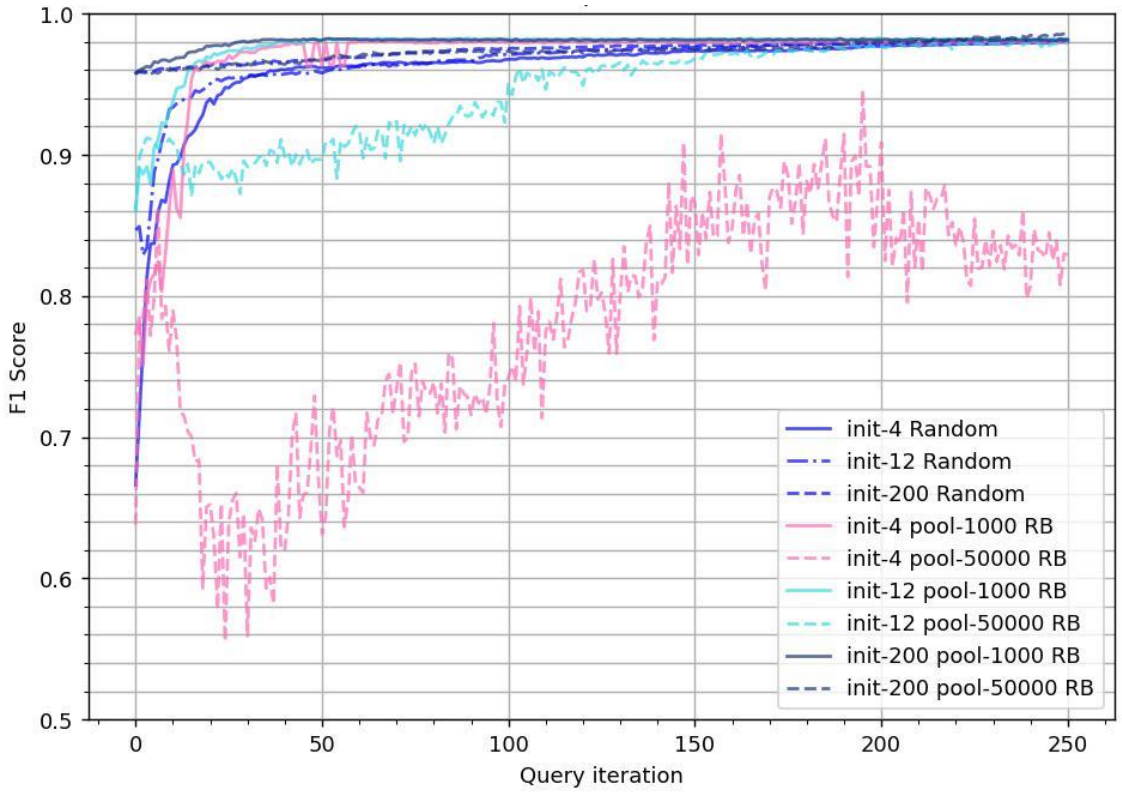

Figure 21. Comparison of Random Sampling and Ranked Batch-mode Sampling

6.5 Key Characteristics Observed in Experimental Outcomes.

ML models in the active learning cycle outperformed the baseline model without active learning and random sampling in most cases. Random sampling achieved a score of 0.982 with 1,011 instances (including initial seed and query instances), while the active learning in the best case achieved a score of 0.98-0.99 with a much smaller number of instances, around 200-300. This demonstrates that ML models with a query strategy in active learning cycles can reduce the number of instances that need to be labelled by 3 to 4 times.

However, there are some cases where active learning and random sampling perform similarly or worse, especially when the initial seed is small, and the unlabelled pool size is large at the same time. For instance, consider the case where the initial seed is 4 and the unlabelled pool size is 50,000. In this case, it costs an additional 50 instances to reach an F1 score of 0.95 from 0.90 for random sampling (Figure 3), while it costs an additional 350, 400, and 200 instances for classification uncertainty (Figure 4), classification entropy (Figure 6), and ranked batch-mode sampling (Figure 8), respectively, with the same setting. Furthermore, all the query strategies continue to

produce unstable figures until they converge in this case. This characteristic proves that when a small initial seed and large unlabelled pool are used in a setting, it causes a delay for the active learner to find suitable instances. This is because when the active learner is trained with few samples, it tends to take time for the query strategy to find good instances. Therefore, active learning cannot be highly effective without a sufficient number of samples that is not too small.

6.6 Analysis of the N-BaIoT dataset Test Result

Testing with the N-BaIoT dataset showed similar characteristics to the MedBIoT dataset, except for the query by committee result. The maximum score for random sampling was 0.996, indicating that this dataset can be scored better than the MedBIoT dataset, which had a maximum random sampling score of 0.98. The classification margin (uncertainty sampling) showed similar characteristics to the MedBIoT dataset and exceeded the random sampling score within 100 queries. This demonstrates the stability and good performance of the classification margin. However, the vote entropy (query by committee) result showed that this sampling method can be unstable and worse than random sampling when using different IoT botnet datasets. This suggests that query by committee may not be a good choice for more than two-class classification. The ranked batch-mode sampling result also showed similar characteristics to the MedBIoT dataset. Overall scores were almost the same or lower than other sampling strategies.

6.7 Comparison of Binary Classification and Multi-class Classification

A comparison of binary classification result and multi-class classification result is shown in Figure 22. Both active leaners used classification uncertainty. The graph shows that the result of binary is slightly better than the result of multi-class. Given the number of classes, we assumed that accuracy could fall as the number of classes increased.

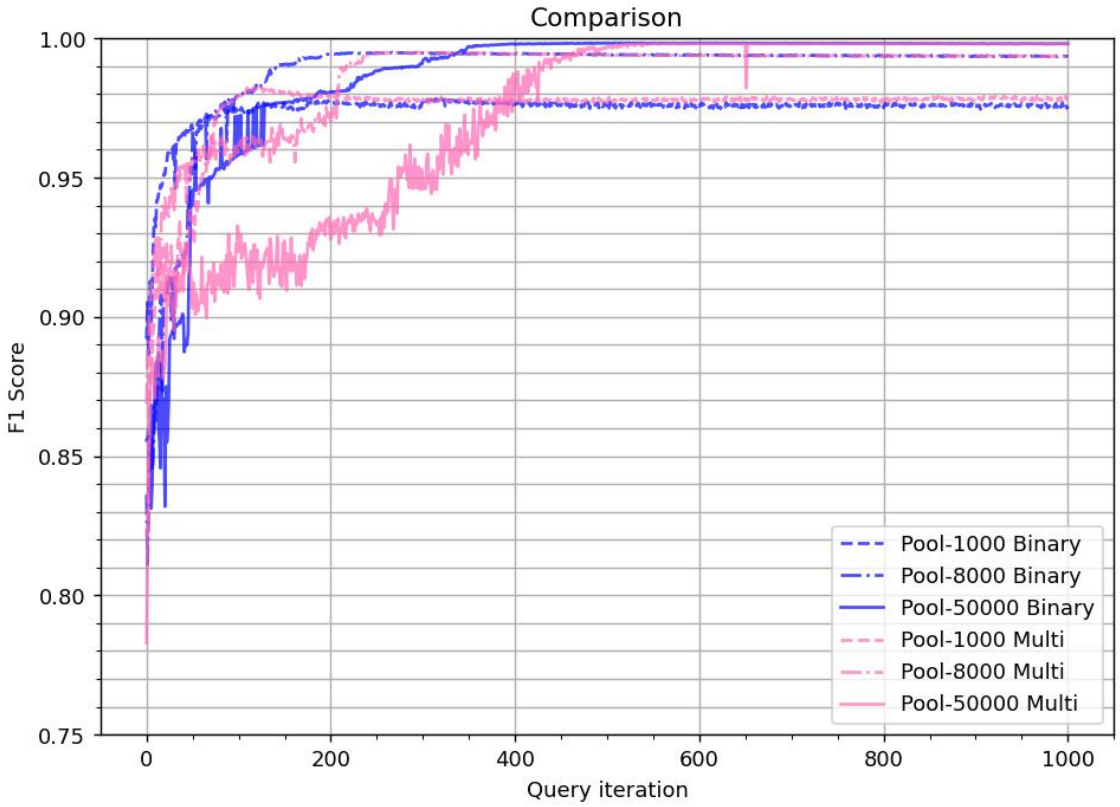

Figure 22. Comparison of Binary and Multi-class Result

6.8 Analysis of Misclassification in Multi-class Classification of Network Traffic

To investigate what caused misclassification, the numbers of selected instances by class and predicted labels were counted in active learning with an initial seed size of 12, an unlabeled pool size of 8000, and 1000 queries. The results show that Mirai and Torii instances tend to get misclassified even though they are selected as query instances more than other botnet instances. This indicates that the network traffic of Mirai is similar to Benign traffic and Torii traffic can also be similar to Benign traffic. These two biases affect the F1 score, which makes the score slightly lower than the multi-class result.

Table 47. Number of Selected Instance (Query) in Each Class

Class	Benign	Bashlite	Torii	Mirai
Count	583	32	163	222

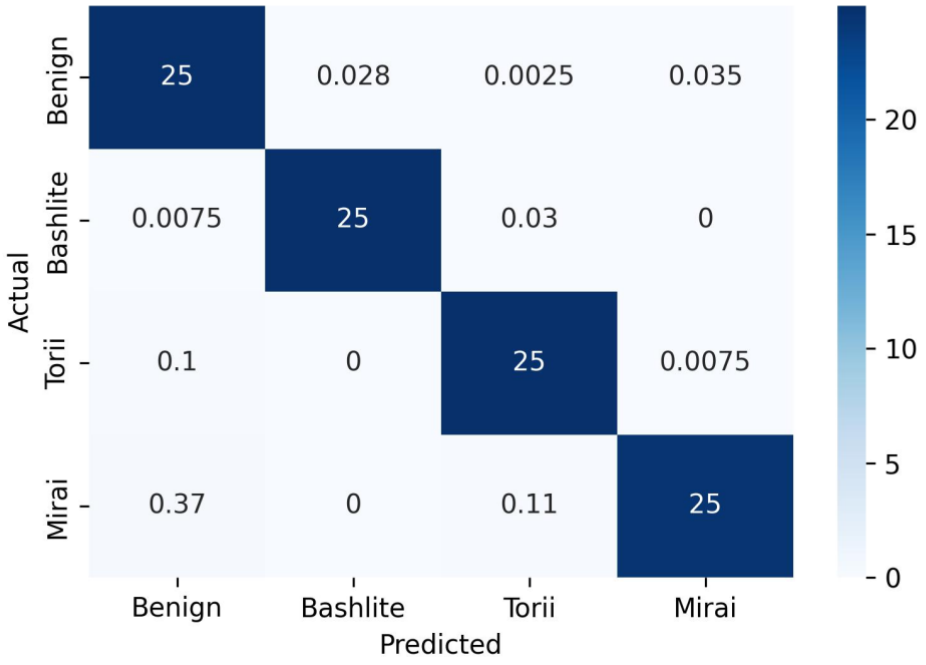

Figure 23. Confusion Matrix of Multi-class Result in Percentage

7 Conclusion

In conclusion, our experiments have demonstrated that active learning is an effective method for identifying IoT botnets and reducing the time and cost involved in labelling data. By using active learning, we were able to reduce the number of instances that required labelling by three to four times. However, it’s important to carefully consider the size of the initial seed and unlabelled pool to ensure that active learning is effective. Our results showed that classification margin was the most effective query strategy, outperforming other strategies even with fewer initial seeds. These findings highlight the potential of active learning for improving the efficiency of IoT botnet identification.

References

[1] IBM Security (2017). The weaponization of IoT devices. https://www.ibm.com/downloads/cas/6MLEALKV (Accessed 2023 April 1st)

[2] Pronemon Institute (2015). The Cost of Malware Containment. https://www.ponemon.org/local/upload/file/Damballa%20Malware%20Containment%20FINAL%203.pdf (Accessed 2023 April 1st)

[3] Ankit Bansal, Sudipta Mahapatra (2017). A comparative analysis of machine learning techniques for botnet detection https://dl.acm.org/doi/pdf/10.1145/3136825.3136874 (Accessed 2023 April 1st)

[4] Hayretdin Bahşi, Sven Nõmm, Fabio Benedetto La Torre (2018) Dimensionality Reduction for Machine Learning Based IoT Botnet Detection https://ieeexplore.ieee.org/abstract/document/8581205 (Accessed 2023 April 1st)

[5] M. Stevanovic, J. M. Pedersen (2015). An analysis of network traffic classification for botnet detection https://ieeexplore.ieee.org/document/7361120 (Accessed 2023 April 1st)

[6] Dau Xuan Hoang, Quynh Chi Nguyen (2018). Botnet Detection Based On Machine Learning Techniques Using DNS Query Data https://sciprofiles.com/publication/view/497407477e26ac59425c462d0009d658 (Accessed 2023 April 1st)

[7] S. Y. Yerima, S. Sezer and I. Muttik (2014). Android Malware Detection Using Parallel Machine Learning Classifiers https://ieeexplore.ieee.org/abstract/document/6982888 (Accessed 2023 April 1st)

[8] Alejandro Guerra-Manzanares, Hayretdin Bahsi. (2022). On the application of active learning for efficient and effective IoT botnet detection https://www.sciencedirect.com/science/article/abs/pii/S0167739X22003399 (Accessed 2023 April 1st)

[9] Jorge L. Guerra Torres, Carlos A. Catania, Eduardo Veas (2019) Active learning approach to label network traffic datasets https://www.sciencedirect.com/science/article/pii/S2214212618304344 (Accessed 2023 April 1st)

[10] Settles, Burr (2009) Active Learning Literature Survey https://minds.wisconsin.edu/handle/1793/60660 (Accessed 2023 April 1st)

[11] Alejandro Guerra-Manzanares, Hayretdin Bahsi, Jorge Medina-Galindo, Sven Nomm. (2020). MedBIoT: Generation of an IoT Botnet Dataset in a Medium-sized IoT Network https://www.researchgate.net/publication/338765489_MedBIoT_Generation_of_an_IoT_Botn et_Dataset_in_a_Medium-sized_IoT_Network (Accessed 2023 April 1st)

[12] Y. Meidan, M. Bohadana, Y. Mathov, Y. Mirsky, D. Breitenbacher, A. Shabtai, and Y. Elovici (2018) N-BaIoT—Network-Based Detection of IoT Botnet Attacks Using Deep Autoencoders https://ieeexplore.ieee.org/abstract/document/8490192 (Accessed 2023 April 1st)

Appendix 1 – Non-exclusive licence for reproduction and publication of a graduation thesis[1]

I Keisuke Konno

1. Grant Tallinn University of Technology free licence (non-exclusive licence) for my thesis “Multi-class Classification of Botnet Detection by Active Learning”, supervised by Hayretdin Bahsi
   1.1. to be reproduced for the purposes of preservation and electronic publication of the graduation thesis, incl. to be entered in the digital collection of the library of Tallinn University of Technology until expiry of the term of copyright.
   1.2. to be published via the web of Tallinn University of Technology, incl. to be entered in the digital collection of the library of Tallinn University of Technology until expiry of the term of copyright.
2. I am aware that the author also retains the rights specified in clause 1 of the non-exclusive licence.
3. I confirm that granting the non-exclusive licence does not infringe other persons' intellectual property rights, the rights arising from the Personal Data Protection Act or rights arising from other legislation.

15.05.2023

1 The non-exclusive licence is not valid during the validity of access restriction indicated in the student's application for restriction on access to the graduation thesis that has been signed by the school's dean, except in case of the university's right to reproduce the thesis for preservation purposes only. If a graduation thesis is based on the joint creative activity of two or more persons and the co-author(s) has/have not granted, by the set deadline, the student defending his/her graduation thesis consent to reproduce and publish the graduation thesis in compliance with clauses 1.1 and 1.2 of the non-exclusive licence, the non-exclusive license shall not be valid for the period.

Appendix 2 – Python Codes for Active Learning Experiments

All the codes used for the experiments are available in
https://github.com/kei5uke/botnet-active-learning