Rohith H P, et. al. International Journal of Engineering Research and Applications
www.ijera.com
ISSN: 2248-9622, Vol. 12, Issue 7, (Series-I) July 2022, pp. 79-91

RESEARCH ARTICLE OPEN ACCESS

Towards A Robustness Assessment of Deep Learning Based Network Traffic Classification

Rohith H P
Rohith.hp84@gmail.com
Lecturer, CS & E Dept.
Vivekananda Polytechnic, Puttur. D.K.

Prof. Balapradeep K N
deepkatoor@yahoo.com
Associate Professor, CS & E Dept.
K.V.G. college of Engineering, Sullia.D.K

ABSTRACT

To recognise content, services, and applications, network traffic control systems are actively investigating network traffic monitoring. While modern firewalls can decode packets, privacy advocates do not find this to be a desirable feature. As a result, decoding any data from encrypted transmission is a challenging task. Machine learning algorithms have been discovered in previous research that might be utilised to identify apps and services. For traffic detection, high-level characteristics are retrieved from network packet data and a sophisticated machine learning classifier is constructed.We propose a categorization strategy based on an ensemble of deep learning architectures for packet, payload, and inter-arrival time sequences. To the best of our knowledge, this is the first time deep learning architectures have been employed to tackle the Server Name Indication (SNI) classification problem. The most advanced machine learning approaches are outperformed by our ensemble model.

In our case, itanalysesthebehaviourofdata over the network,andthenthisdataisconsidered asattackornormalbasedonthebuiltmodelbehaviour. MostoftheexistingdetectionsystemsrelyheavilyonhumananalyststomeasureLogs losstodifferentiatebetweenmalware and other category of service.Withtheincreasenetworktraffic,manualworkbyhumansinthedetectionsystemisanon-trivialproblem. Thus,machinelearningtechniquesarefastemerging,wherewecantrainthesystemand evendetectanomalyattacks. We are using CICIDS 2020 datasetthataretrainedtoclassify normal and attack data using Random Forest Classification, Decision Tree Classification, Support Vector Classifier, Voting Classifier and Convolution Neural Network. These modelsareevaluatedbasedonmetricssuchasAccuracy, FalsePositiveRate, True Positive Rate and we are able to achieve 98% accuracy using Convolutional Neural Network.

I. INTRODUCTION

TLS is one of the most significant cryptographic protocols for maintaining Internet communication security. The protocol allows client/server applications to communicate securely [1, preventing eavesdropping, message forgery, and message manipulation]. TLS has become an essential part of the internet, and websites are encouraged to use it to protect user privacy and security. TLS is widely used in HTTP, SMTP, FTP, and VoIP where privacy and security are required, and the number of websites using HTTP in TLS tunnels (HTTPS) has increased dramatically over the last decade [2].Figure 1 shows how the client and server connect via HTTPS services and a TLS handshake. This discussion will determine protocol versions, cryptographic techniques, SSL certificates for authentication, and shared secrets based on public-key cryptography. After a successful handshake, the client and server can start exchanging data across an encrypted channel [1].

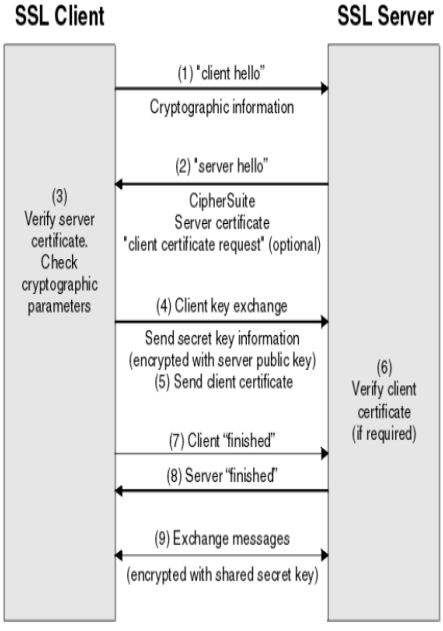


FIG NO. 1 - TLS HANDSHAKING PROTOCOL

As demonstrated in Figure 1, SNI is a TLS handshake extension that provides the destination hostname and may be retrieved from the Client-Hello message. SNI is a critical component of HTTPS traffic inspection for many businesses and institutions. To safeguard users security, firewalls check SNI to see if a server name is allowed. Intermediaries that censor their internet services also utilise SNI as a filter [4]. Users' privacy is not completely safeguarded since SNI is not encrypted, and a man-in-the-middle can listen in and discover the requested websites [5].SNI may also be used to get around firewalls and eavesdroppers [6]. Since mid-2020, an improvement known as Encrypted SNI (ESNI) has been proposed to address the issue of domain eavesdropping [4, 7]. If effectively implemented, this change will assist privacy advocates while also providing new challenges for network administrators and eavesdroppers.

These solutions depended on human effort to discover patterns in unencrypted payloads or match port numbers on a continual basis. New approaches based on conventional machine learning algorithms, such as random forest (RF), Decision Tree, Support Vector Machine, Voting Algorithm, and Convolutional Neural Network, have evolved as a result of inefficiency and lack of accuracy.Classic machine learning techniques had been achieving state-of-the-art accuracy in the traffic categorization problem for some years [7]. However, these relatively simple algorithms were unable to catch more complicated patterns found in today's Internet traffic, and their accuracy has suffered as a result. Deep learning models recently achieved best-in-class performance in traffic classification [15]. They are useful for traffic categorization because of their capacity to understand complicated patterns and conduct automated feature extraction.

Deep learning algorithms can reach excellent accuracy, but they require a lot of labelled training data. Labeling is a time-consuming and inconvenient process in the network traffic categorization activity [14]. Researchers frequently collect flows of each class in isolation and in a controlled setting with little background traffic in order to appropriately categories each flow. This is a time-consuming and labor-intensive technique. Furthermore, traffic patterns seen in a controlled setting may differ dramatically from real-world traffic, making the conclusion invalid.

II. LITERATURE SURVEY

In various published papers, a proposedmethodologythatreliesonevaluating variousMachine Learningclassifiersbycomparingtheaccuracyrateand valueofRandom Forest, Decision Tree, Support Vector Machine, Voting Algorithm and Convolutional Neural Network. Theimplemented experiments demonstrated that the Random Forest Classifier achieved the lowest value offalsenegativeandalsotheConvolutional Neural Networkhasachievedthehighestaverage accuracyrate.

Shabir et al. 2016, were among the first to address the challenge of service identification for HTTPS-specific traffic (for example, maps.google.com versus drive.google.com) [8]. Their task includes gathering HTTPS traces from user sessions and labelling each connection with the SNI extension. The usual packet and inter-arrival time statistics, as well as extra statistical aspects relating to the encrypted payload, are included in their suggested statistical framework. They use Decision Tree and Random Forest classifiers to get the best results.

Okada et al. 2018, build on this work by concentrating on the establishment of statistical characteristics for application classification (FTP, DNS, HTTP, etc.) based on packet size and packet transfer timings [12]. When used with Support Vector Machines classifiers, these characteristics attain great accuracy. However, application level identification is insufficient to answer our study issue because our goal is to discover the underlying service name rather than the traffic type.

L. Bernaille and R. Teixeira, 2019,Gaussian mixture model However, owing to their simplicity, manual feature extraction (which is becoming increasingly difficult with today's heavily encrypted data), and lack of high learning capacity to catch more complicated patterns, their accuracy has lately dropped [12].

H. Zhou, Y. Wang, X. Lei, and Y. Liu, 2019,For traffic type categorization, a LeNet-5 convolutional Neural Network (CNN) model was utilised, which was created in 1998 for handwritten number recognition. As input to the model, a 2-dimensional picture is reorganised from several statistical characteristics. They claim good accuracy, but the model can't be utilised for live applications [15] since the statistical aspects need the observation of the full flow.

V. TONG, H. A. TRAN, S. SOUIHI, and A. MELLOUK 2020,For QUIC protocol traffic categorization, the authors employ both statistical characteristics and payload data. To discriminate between chat and voice calls with other classes, they initially employ statistical characteristics and a random forest method. If further classes are found, they categorise video streaming, file transfer, and Google music using payload data and a CNN model. Their first stage

necessitates the observation of the whole flow, making it only suited for offline applications. Although encrypted, payload information has been utilised in other articles as well. On the ISCX dataset, a CNN and stacked Auto-Encoder (SAE) are combined to categorise traffic kinds and applications in [17]. Deep neural networks are used as a black box in these procedures, with no human-understandable properties identified.

Chen, K. He, J. Li, and Y. Geng, 2020,Reproducing Kernel Hilbert Space is used to turn time-series properties of each flow into 2-dimensional pictures (RKHS). The resulting pictures are fed into a CNN model. They compare their CNN model against SVM, decision trees, and naive Bayes, which are all traditional machine learning methodologies. The CNN model beats traditional machine learning algorithms with an accuracy of over 99 percent. A convolutional neural network, an LSTM model, and various combinations are used in [19] to classify a variety of services, including YouTube and Office365. When time-series features and header features are combined with the CNN/LSTM architecture, they obtain an accuracy of roughly 96 percent.

S. Rezaei and X. Liu 2021, is the only study that addresses the requirement for a large labelled dataset. This method uses a semi-supervised learning approach in which a CNN model is pre-trained to predict numerous statistical properties from sampled packets. They make advantage of sampled packets' time-series properties. The final few layers are then replaced with new ones, and the model is retrained using a tiny labelled dataset. Because statistical characteristics can be computed simply when whole flows are accessible, their technique does not require human labor for labelling the pre-trained dataset [21].

PROBLEM DEFINATION

Recently published a paper on traffic categorization from the standpoint of machine learning. Data gathering, feature extraction, feature reduction and selection, algorithm selection, and model deployment were the five processes they used to introduce traffic categorization. Each stage's technology is summarized and evaluated. The Project also went through the many data categories that were employed in the categorization algorithms, including statistical traits, payloads, and host behaviors. In comparison to this survey, the following are the differences and advantages of our Project.

While the scope of our review is broader, we will focus on machine learning-based categorization algorithms. Second, unlike [13], our study is based on a comprehensive set of evaluation criteria. As a result, our assessment is conducted in a consistent manner and provides significant insight. Comparing outcomes allows us to naturally come up with fascinating insights. Third, we look at the datasets and features on different data levels, such as flow and packet level, to see how they affect classification performance. This analysis, however, is lacking in [13]. Finally, in contrast to [13] and other previous surveys, we uncover new outstanding concerns and suggest innovative study avenues.

Despite the fact that there are a number of surveys on network traffic classification. Our Project starts with a variety of concerns. We discovered that previous evaluations don't go into enough detail into behavior- and correlation-based categorization approaches. They don't take into account user privacy, feature redundancy limits, or other factors. This encourages us to finish the Project.

PROPOSED SYSTEM

While ESNI benefits privacy supporters over eavesdropping, previous research has shown that apps and services may be predicted with high accuracy without using SNI data [8, 9]. Despite using HTTPS encryption, Chen et al. find that side data may be obtained from a variety of web applications [9]. According to their approach, the eavesdropper can only view the number of packets and their timing/size. The data included health information, household income, and search queries. Large traffic and communication fluctuations in web apps are the major source of these side-channel breaches, and defending against them is complex and application-specific [9].

Because high SNI classification accuracy shows that such protocols are unable to fully protect user privacy from side-channel attacks, traffic and communication differences between web apps might pose a significant threat to ESNI and other ways to circumvent SNI identification. The typical diagram as shown below :

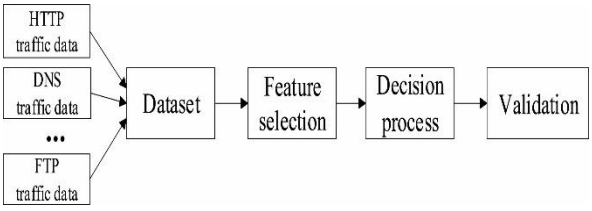


FIG No. 2: Process of Traffic Classification

The primary goal of this study is to evaluate how well deep learning can classify

HTTPS SNI. The SNI will act as our ground truth labels, and only encrypted TLS packet contents without the SNI extension will be utilized. On the assumption that SNI isn't faked or created, we'll test if deep learning can improve service identification accuracy. To our knowledge, this is the first study to use deep learning on HTTPS data to categories SNI.

DATA PREPROCESSIONG

The training and testing datasets are first pre-processed, then using a suitable classifier like Random Forest Classifier, Decision Tree Classifier, Voting Algorithm, Support Vector Machine and Convolution Neural Network is done. The testing set is then used to test the learning and outputs the results. The decision results are in the form of accuracy, confusionmatrix, precision, recall and f-measure. The data preprocessing stage are broadly divided into four stages, and they are as follows.

- Importing libraries.
- Importing datasets.
- Separating dependent and independent variables.
- Splitting the data.
- Training the Model.
- Deploying the Model

Many academics have focused on anomaly detection because of its potential for identifying novel assaults. However, owing to system complexity, which necessitates extensive testing, assessment, and tweaking before to deployment, its applicability to real-world applications has been impeded. The most idealistic way for testing and assessment is to run these systems on actual labelled network traces with a full and large set of intrusions and anomalous behavior.

This is a significant challenge in and of itself, because datasets are extremely rare, owing to the fact that many are internal and cannot be shared due to privacy concerns, while others are heavily anonymized and do not reflect current trends, or they lack certain statistical characteristics, so a perfect dataset has yet to exist. As a result, researchers are forced to use inadequate datasets. As network behaviours and patterns change and intrusions evolve, it's become increasingly important to move away from static, one-time datasets and toward more dynamically generated datasets that not only reflect the traffic compositions and intrusions of the time, but are also modifiable, extensible, and reproducible.

A systematic strategy has been created to produce datasets for analysing, testing, and evaluating intrusion detection systems, with an emphasis on network-based anomaly detectors, to solve these inadequacies. The major goal of this project is to provide a systematic strategy for generating a varied and complete benchmark dataset for intrusion detection using user profiles, which are abstract representations of network events and behaviours. The profiles will be integrated to create a diversified variety of datasets, each with its own set of characteristics, each covering a different aspect of the assessment domain.

Brute-force, Heartbleed, Botnet, DoS, DDoS, Web assaults, and network penetration from within are among the seven attack scenarios included in the final dataset. The attacker infrastructure consists of 50 computers, whereas the victim company consists of 5 departments with 420 workstations and 30 servers. Each machine's network traffic and system logs are included in the dataset, as well as 80 characteristics derived from the collected traffic using CICFlowMeter-V3.

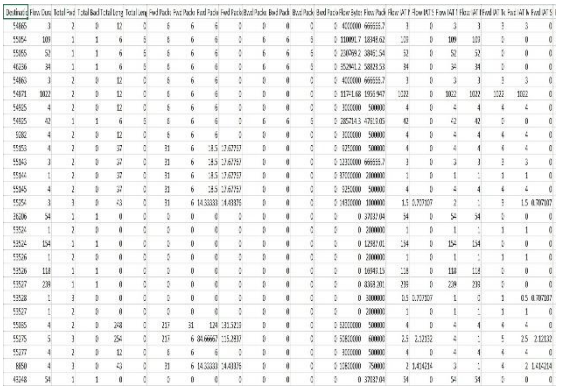


FIG No. 6.1 : DATA FILE FORMAT

6.1 : DATA EVALUATION.

For all of our machine learner models, log loss is the evaluation metric. The following is calculated:

$$LogLess = 1/N * = \sum_{i=1}^{n} \quad \sum_{i=1}^{n} Yij * logPij$$

Where

N **=** The number of files in the test set.

M = The Number of labels

Log = Natural Logarithm

Yij = 1 if observation i is in class j and 0 otherwise.

Pij = is the predicted probability that observation i belongs to class j.

FIG NO. 6.2 : .BYTE CODE FORMAT

A logloss model is equivalent to a logloss model with a low rating. We also present our place in the leading board of Kaggle Every model we had designed. We don't use any assembly dictionary

SYSTEM DESIGN

The suggested approach's flowchart is depicted in the following diagram. The training phase and the application phase are the two primary portions of this flowchart.

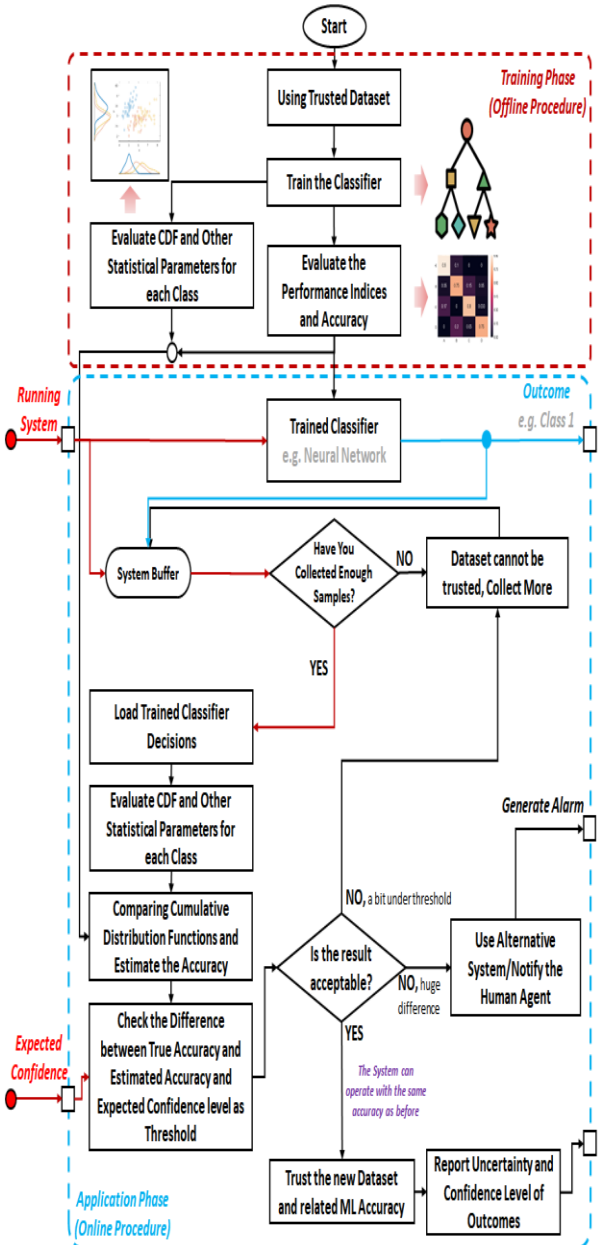


FIG No. 7.1: SYSTEM DESIGN

Training Phase, The training phase is an offline technique in which a reliable dataset is utilised to train the intelligence algorithm, which can be either machine learning or deep learning. The categorization abilities of machine learning will be the emphasis of this research. As a result, a trustworthy dataset will be used to train the classifier, and its performance will be monitored using current KPIs. In the meanwhile, each class's probability density function and statistical parameters will be computed and kept for comparison.

Application Phase, Real-time and unlabeled data will be fed into the system via an online approach. Consider an autonomous vehicle that has been taught to recognise impediments and should be able to avoid a collision. As a result, the trained classifier should be able to discriminate between the road and other objects during the application phase. The lack of a label on the data is a significant and vital issue in the application process. As a result, it is impossible to guarantee that the classifier will perform as accurately as it did during the training phase. The untrusted labels of the classifier will be utilised in the application phase, as will the probability cumulative distribution function (CDF) and statistical characteristics of each class.The accuracy is estimated using the CDF-based statistical difference of each class in the training and application phases. If the estimated accuracy and expected confidence difference were very small, the classifier results and accuracy could be trusted (in this case, the autonomous car could continue to operate); if the difference was small, the system could request more data and re-evaluation to ensure the distance was accurate. If the difference is significant, the classifier's findings and accuracy are no longer valid, and the system should switch to a different technique or alert a human agent.

7.1 : BASELINE MODEL

To begin the machine learning-based classification, 80 percent of each dataset was utilised for training and testing, and 20% was used for validation, using 10-fold cross-validation. For classification, both linear and nonlinear classifiers were chosen. Linear approaches include Linear Discriminant Analysis (LDA) and Classification And Regression Tree (CART). Furthermore, nonlinear approaches such as Random Forest (RF), K-Nearest Neighbours (KNN), and Support Vector Machine (SVM) are used.

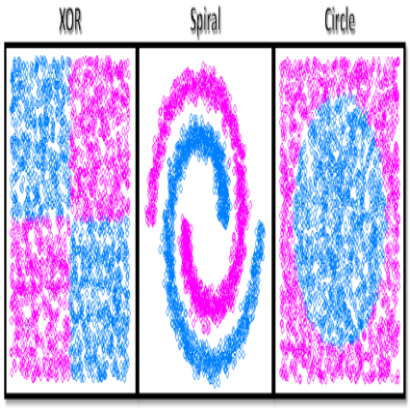


FIG No. 7.1 : Scatter Plot of Base-Line Machine Learning Model

The accuracy and Kappa measure are used as KPIs to assess each classifier's performance. Finally, the Kolmogorov-Smirnov Distance (KSD), Kuiper Distance, Anderson-Darling Distance (ADD), Wasserstein Distance (WD), and a combination of ADD and Wasserstein-Anderson-Darling Distance (WAD) were chosen for evaluation as Empirical Cumulative Distribution Function (ECDF)-based statistical distance measures.

7.2 : EXPLORATORY ANALYSIS

Because of the vast volume of details, we carried out exploratory analyzes to fully understand the results. We have evaluated the following:

- What is the distribution of the class frequency?
- How does each file size class differ?
- Previous literature shows that in software with the same type and levels the opcode ratio is identical. So we wanted to find out if this applies to our dataset.

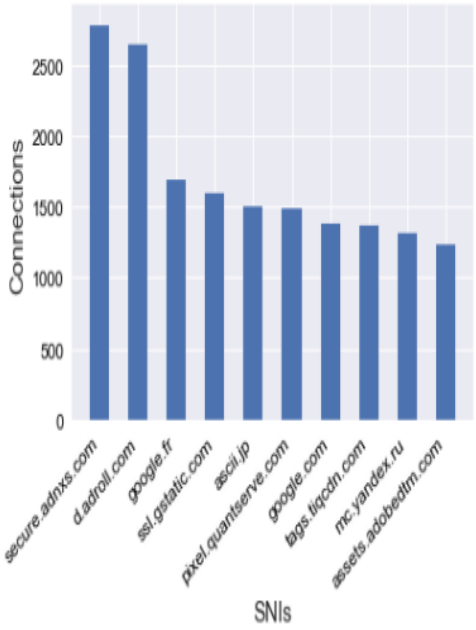


FIG No. 7.2.1: CLASS OF DEEP LEARNING AND NO: OF SAMPLES

Fig No. 7.2.1 displays the Network Packets family class distribution. We may infer that the groups are not uniformly divided. The most popular class is class 10 and the least regular class is class 10.

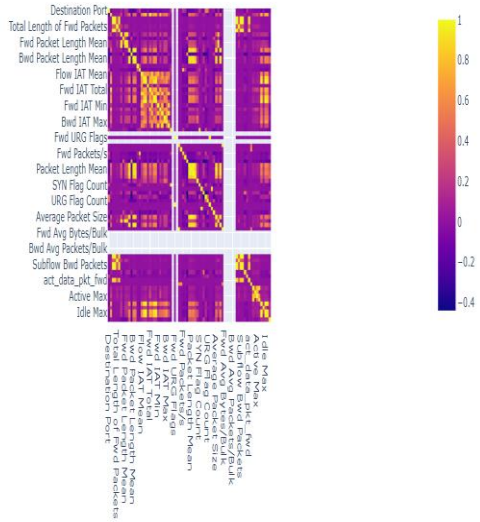


FIG No. 7.2.2: FILE SIZE VS CLASS

Fig No. 7.2.2 displays the Correlation Matrix Graph of file sizes in .csv formatand binarybytes with circular colors reflecting specific network packet traffic groups. We can deduce that some classes can be differentiated by file sizes.For instance, class 1 tends to have lower stream of bytes and network packet file sizes.

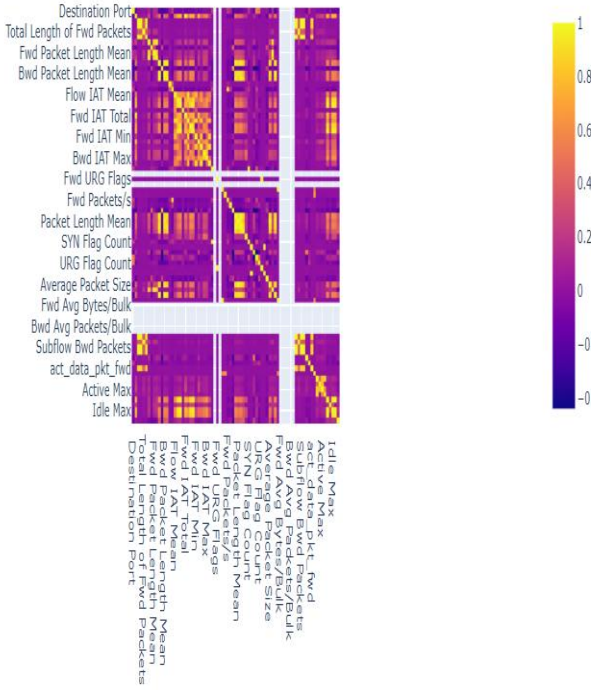
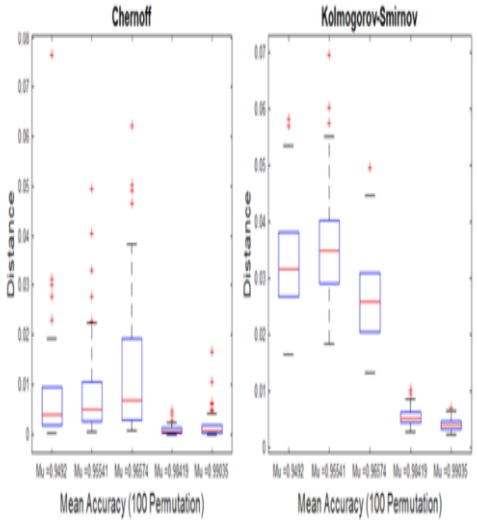


FIG No. 7.2.3: Ratio of opcode of each class

Fig No. 7.2.3 displays the plot of the Top 4 frequent network data packets ratios per class. We can conclude that this ratio is unique to few of the classes (e.g. class 1 vs class 20), which would therefore be useful in the grading. Although we only drew up the top four frequent functions, we thought that the frequencies of all 147 opcodes would be interesting and whether they would help to improve the classification accuracy.

7.3 : CORELATION ANALYSIS

Pearson's correlation between Wednesday's data classes and statistical ECDF-based distances The WD and WAD distances, as can be shown, have a stronger relationship with the classes. The association between the metrics is also seen in this diagram. The KSD and the KD appear to be linked. The WD and WAS appear to be linked as well. Because of the similarities in their phrasing, these connections may be explained. The P-values for the aforementioned correlations were all 0, indicating that the correlation hypotheses were correct.



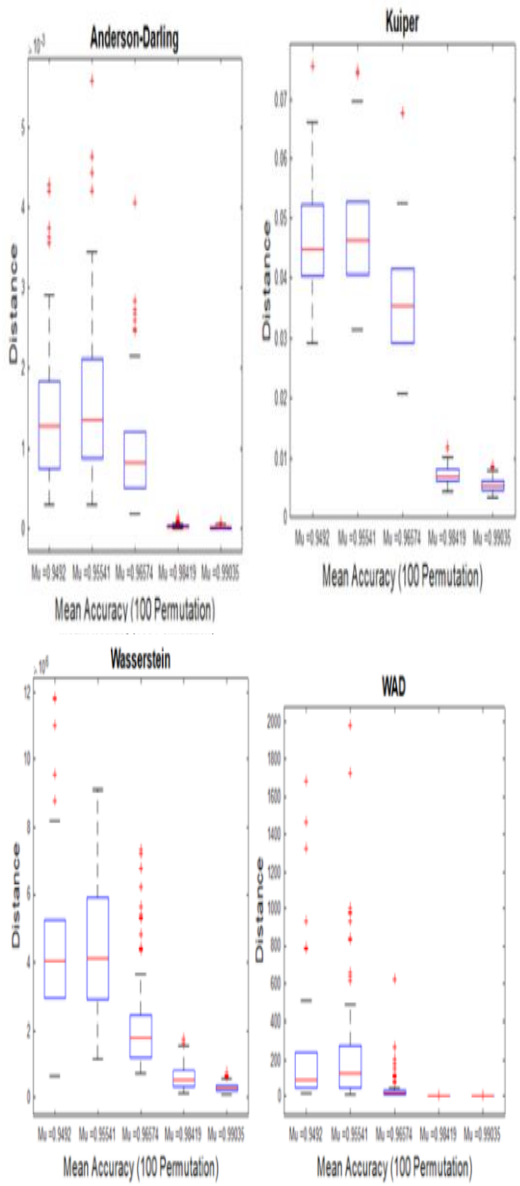


Fig No. 7.3.1 : OVER 100 REPETITIONS, A BOX PLOT OF STATISTICAL DISTANCE MEASUREMENTS VS. ACCURACY WAS CREATED.

The figure shows the class diagram of ids.There are classes like user, signup page,login page,prediction page,ELM,RF and result.The prediction page has subclasses like ELM and RF.Each class has its own attributes and methods. The class user has attributes like user name and mobile number. The signup page has attributes like name,mobile number,email id,user name,password.The data type of the attributes is also mentioned along with the attributes.It has a method called submit (). Each class has multiple attributes and methods.

7.4: STATE MODEL

The state model contains different states and describes how the control flows from one state to another. The state model has several phases namely testing and training.

FIG No. 7.4.1 : THURSDAY SECURITY INTRUSION DETECTION CONFUSION MATRIX IN CICIDS2020 DATASET

In testing phase,data pre-processing,pattern extraction is done. It has CICIDS2020 as the dataset.The testing set undergoes data pre-processing and the pattern extraction is done. The training phase has CICIDS2020 dataset and feature selection is done using hybrid method.We make use of anomaly model database.The classifiers used are elm and RF.The feature selection will select the best feature.The testing set and training set is compared and classifier is used to determine if the dataset is prone to attack or not.In the front end,the result will be displayed.It shows the accuracy of the classifier.It even gives the attack type and the final score.

7.5: SEQUENCE DIAGRAM

Sequence diagram shows how the sequence of message and acknowledgements are passed between the objects of a system.

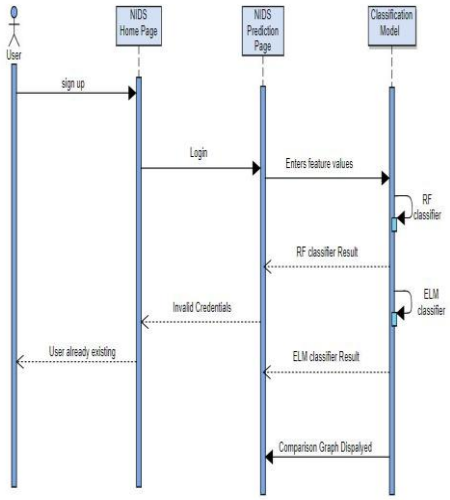


FIG No. 7.5.1 : SEQUENCE MODEL

User can give values to each feature to predict the type of data using three classifiers such as ELM Classifier, Random Forest. Based on the values entered by the user, each model predicts its output that is, whether the data is normal or a particular type of attack. Here dashed arrows indicate dependency and solid arrows indicate association.The figure shows the sequence diagram of ids.It has user, home page,prediction page and classifier model as entities.The user will sign up to the home page.If the user logins with the wrong user id, it gives invalid.Once the user is logged in, he can click on predict and enter the feature values for testing set.There are 2 models used like elm and RF classifiers. The elm gives its result and Random Forest and Convolution Neural Network gives its result.The result will have the accuracy of the model and final score.Then it displays the graph of comparison of 2 models.

IMPLEMENTATION

We built a number of other models for machine learning, trainable using data set features, once we had the Baseline Model. The most common features of our models are based on the "size." We used features such as opcode frequency with multiple data packet instructions, system call frequency, file sized and class files. We used a collection of such features as our final model to determine whether this resulted in a better prediction.

8.1 :TOWARDS EXPLAINABLE ARTIFICIAL INTELLIGENCE

We address an issue related to our suggested technique in order to demonstrate how it may be used for this purpose as well. Explainable AI (XAI) is a technique or framework for improving the interpretability of machine learning algorithms and their outputs [24]. Using the statistical ECDF-based distance metrics previously discussed, our suggested technique may also be utilized to improve the interpretability of ML classifiers.We will explore a tiny example here, with the intention of expanding on this topic in future publications. Figure 10 shows the class labels vs. the sample time for the Wednesday data from the previously discussed security dataset. There are six different classifications in this dataset, each with a different number of occurrences. A sliding window with a d = 1500 size is utilised in this illustration. Initially, 1500 samples from class one are used as a reference, and then the remainder of the data for each window are compared using statistical ECDF-based distance measurements.

It's worth noting that the output smoothness is proportional to the size of the sliding window. The change in average distance vs. class, as seen in the graph, demonstrates the substantial connection that exists. Furthermore, it appears that class number five is relatively resistant to statistical change, but class number six has a small sample size, making statistical differences insignificant. Reduce the size of the sliding window to address the difficulty of identifying class six. This graph may be made for a variety of classifiers to demonstrate how their judgments are connected to ECDF-based distance metrics. We plan to examine ECDF-based distance inside other algorithms in the future to better understand their activities.

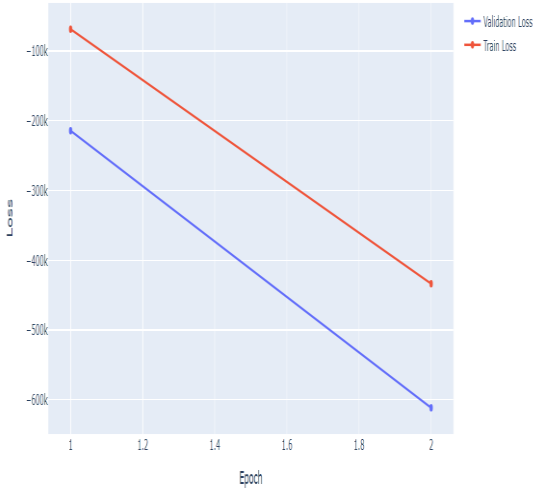


FIG No8.1.1. LOSE VS EPOCH FOR MLP

8.2 : OPCODE FREQUENCY CLASSIFIER

From our initial research and review we find that the opcode frequency of some ASM

instructions plays an important role in the detection of the malware family. So, in both training and test data, we extracted counts of each ASM command. We had about 147 ASM instructions totally different. We trained a Logistic Regression Model with these 147 frequency features. The value of the regularization parameter 'C,' resultant in optimal log loss, was determined by a 10-fold cross-validation system. This model resulted in a log loss of 1.09. This strengthens the basic line as well as the file size versions.

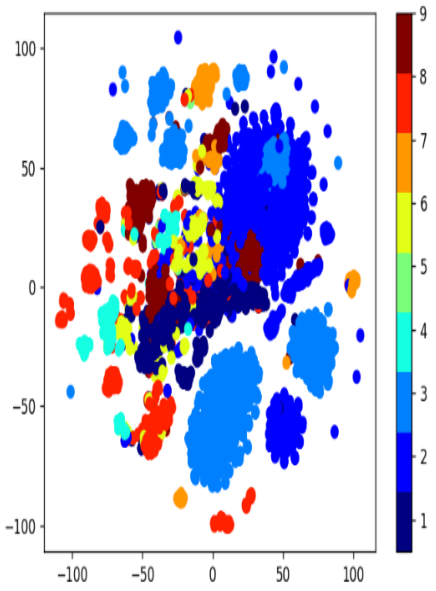


FIG No. 8.2.1 : OPCODE FREQUENCY CLASSIFIER

8.3 : LOGISTIC REGRESSION CLASSIFIER

Logistic regression, though, is a linear classifier, which wouldn't be useful if the dataset isn't dimensional. We were not sure if our dataset was linearly separable using the 147-dimensional data package. This led to the idea to use an ensemble model combining several linear separators, eventually leading to a non-linear classification. So we learned a variant of Random Forest using the Decision Trees Ensemble.

In order to determine the value for the NumEstimators parameter which defines the number of decision trees to be fit, we again employed 10-fold cross-validation. This model resulted in a load loss of 0.058. This is by far the greatest log loss we have ever accomplished. On the Kaggle Leaderboard we stood 168th (out of 377) on this platform.

8.4 : RANDOM FOREST CLASSIFIER

We observed that training a Random Forest Model for 10373 train data files with 147 features takes an enormous amount of time. Therefore, by selecting just the most appropriate functions, we intended to minimize the fixed function. The "sklearn.ensemble" library. In Python, Random Forest "has an embedded rating system that offers a collection of apps with their ranks.

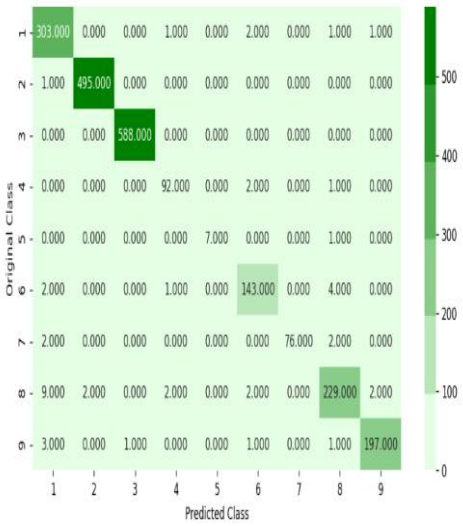


FIG No. 8.4.1. RANDOM FOREST CONFUSION MATRIX

This works according to the criteria of "information gain." The top ranking features will contribute to the best knowledge advantage if separated. So we have extracted the top ten scoring features between the opcode frequencies with this measure. We now conditioned a model for Random Forest with these 10 characteristics. This model resulted in a log loss of 0.07. While it is much higher than the previous 0.058 log missed, we think it's a decent value for the time saved in computation. We stood on the leaderboard for this platform in 190th position (out of 377).

8.5 : KNN CLASSIFIER

There are different functions in each class of malware. Ramnit (Class 1) attempts to deactivate the firewall of a computer , for example, Vundo (class 5) is an adware to generate popups and ads and Kelihos is the bot that sends spam messages.

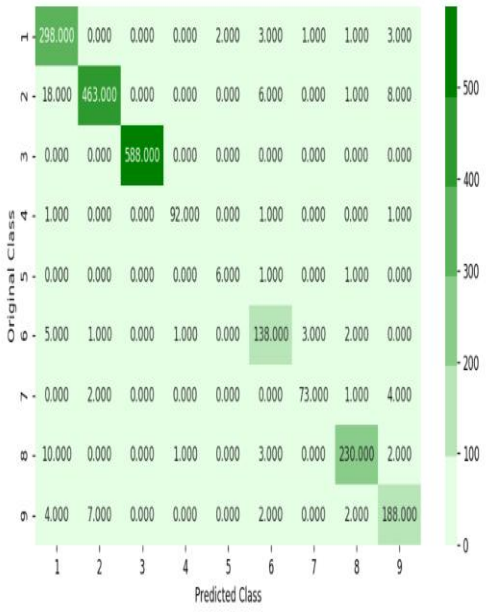


FIG No. 8.5.1 : KNN CONFUSION MATRIX

We therefore found it a distinctive attribute to be the amount of syscalls. In Opcode frequency model we incorporated syscall counts and educated a KNN classification. However, the log loss of our model was actually increased to 0.27. This new function may have introduced noise to the results, so it seemed redundant. However, it is worth exploring a model that uses the actual syscalls (instead of the frequency) as a feature.

8.6 : XGBOOST CLASSIFIER

The library XGBoost implements the decision tree algorithm for boosting gradients. There was a misunderstanding. It is called gradient optimization because it uses an algorithm of gradient lowering to mitigate failure as new models are implemented. This method encourages statistical analysis issues in both regression and classification.

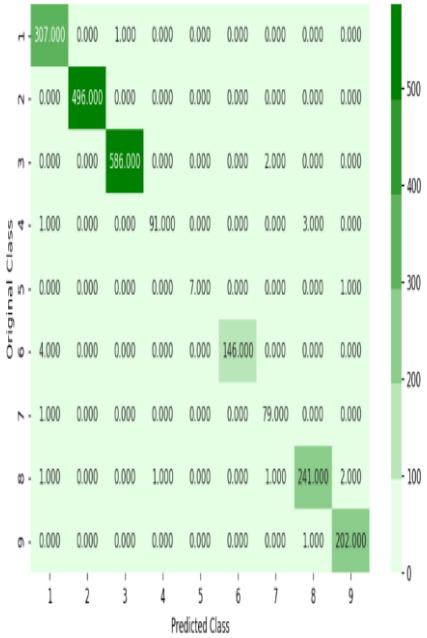


FIG No. 8.6.1 : XGBOOST CONFUSION MATRIX

Right from the top we note that for each class the correlation matrices appear very similar. We may conclude that the various bands are strongly associated. Intuitive, one would believe that if one band takes an object, the other group should capture something. It doesn't seem surprising.When we had checked our experiment independently using different sets of characteristics, the final concept was to use a random forest model and to train it to decide whether it is more reliable. We then had the characteristics of the file size and the opcode frequency (top 10) together and a Random Forest Model training tutorial.

TABLE NO. 8.6.1 : XGBOOST CLASS MATRIX

Class 1	**Class 2**	**Class 3**	**Class 4**	**Class 5**	**Class 6**	**Class 7**	**Class 8**	**Class 9**
96.40%	99.72%	99.90%	99.51%	92.01%	96.99%	99.51%	98.56%	99.62%

This model resulted in a log loss of 0.049. This is the strongest log loss of all the models we've accomplished. On Kaggle's leading board we stood 151st (out of 377).While log-loss was our standard measure of evaluation, it would be interesting, if we predicted "hard labels" for the classification

work, to evaluate the accuracy of our model. The hard labels are calculated by choosing the class with the highest probability. The accuracy per class of cross validation in the Random Forest Training Set is as mentioned above Table.

FINAL RESULT

The following table provides an overview of the results of all our classification models. The Kaggle Leaderboard for each of our models and the cross-validation performance for the training data are also discussed.

MODEL	CROSS VALIDATION MEAN SCORE (%)	MODEL ACCURACY (%)	DATASET
BASE LINE	89.12	96.35	303/377
RANDOM FOREST CLASSIFIER	96.26	98.97	287/377
DECISION TREE CLASSIFIER	95.92	99.12	169/377
SUPPORT VECTOR CLASSIFIER	95.12	95.33	191/377
VOTING CLASSIFIER MODEL	90.12	96.32	
CONVOULTIONAL NEURAL NETWORK	97.12	99.99	39789/39789

FUTURE WORK

Lack of labeled data, The quality and amount of data used for training determines the performance of machine learning techniques, particularly deep learning algorithms. In order to use DL-based approaches to develop prediction models in NTMA, a large volume of network traffic is required. Although tremendous developments in communication systems and networks, such as IoT and 5G, result in the daily generation of vast volumes of raw data, data labelling remains a time-consuming, computationally intensive, and labor-intensive operation. The majority of data in real-world networking applications is unlabeled or semi-labeled. Integrating DL approaches with other machine learning techniques that can deal with unlabeled or semi-labeled data is a promising option.

Difficulties in using DL for structured data,Structured data, such as network traffic data, is a defined format that organizes data in tables with rows and columns. The earliest and effective uses of DL approaches, however, have been reported on problems with unstructured data, such as video, pictures, text, and audio. Some machine learning professionals are opposed to utilizing deep learning (DL) for structured data because they feel that labelled structured datasets are too small to train DL algorithms. Furthermore, they suggest that traditional machine learning algorithms like KNN and SVM are far more straightforward and intuitive than complicated deep learning algorithms (e.g. GANs). Resource-constrained networks, The majority of deep learning algorithms are built to be learned and used by devices with enough resources. Training a deep learning algorithm with a high number of training samples and parameters necessitates computing, memory, and power-intensive hardware. This runs counter to the increased interest in deploying resource-constrained devices (e.g., IoT devices) with AI and learning-based technologies. As a response to this difficulty, the approaches in have been presented. Given the great gains in resource-constrained IoT devices and the large expansion of data produced at the network's

CONCLUSION

By using deep neural network topologies for SNI detection, this study adds to the literature of TLS-based encrypted traffic categorization. While most previous research has focused on recognizing application types, we are concentrating on detecting HTTPS services. There are two reasons for our interest in SNI classification: (1) There are now active research initiatives aimed at removing the SNI extension to avoid eavesdropping [4, 7], and (2) SNI can be faked by users even in existing systems [6]. However, if SNI can be anticipated with high accuracy from encrypted traffic, such strategies may be rendered worthless.

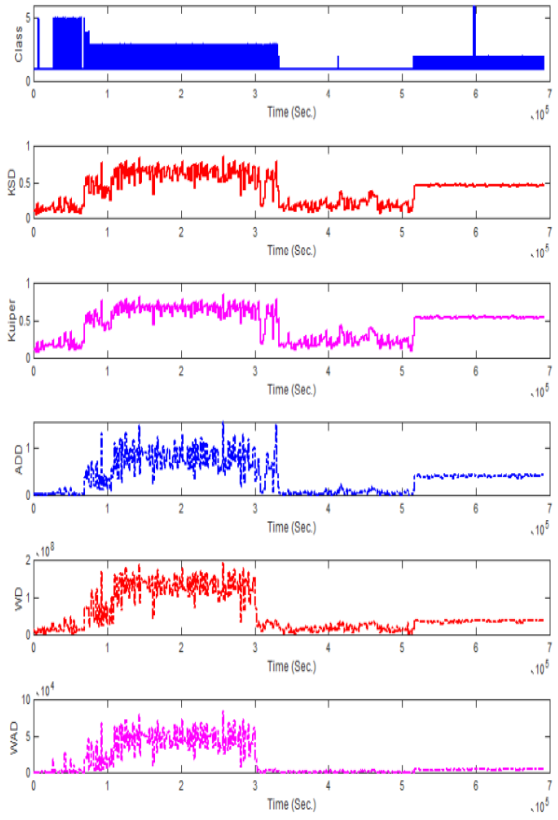


Fig No. : Class label and statistical CICIDS2020-based Distances shown against time (Security dataset: Wednesday)

We can accurately identify SNI using a neural network design that solely considers statistics and sequences of encrypted TCP data, ignoring any header information. Recurrent Neural Networks, Convolutional Neural Networks, and Random Forest (the best machine learning approach based on literature and Auto-Sklearn learn) are all used in our model. We establish a high accuracy for an ensemble model that, to the best of our knowledge, exceeds the state of the art by carefully assessing multiple methodologies and researching the most informative elements of network flow data. This model will be tested on real-time HTTPS traffic in the future to see how well it predicts internet services.

REFRENCE

[1]. the data set is downloaded from the following link http://www.kaggle.com/c/malware-classification

[2]. Dahl, George E., et al. ”Large-scale malware classification using random projections and neural networks.” Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on. IEEE, 2013.

[3]. Annachhatre, Chinmayee, Thomas H. Austin, and Mark Stamp. ”Hidden Markov models for malware classification.” Journal of Computer Virology and Hacking Techniques (2014): 1-15.

[4]. Weber, Michael, et al.”A toolkit for detecting and analyzing malicious software.”Computer Security Applications Conference, 2002. Proceedings. 18th Annual. IEEE, 2002.

[5]. https://www.blackhat.com/presentations/bh-usa-06/BH-US-06-Bilar.pdf

[6]. Austin, T. H., Filiol, E., Josse, and Stamp, S. M. ˆaAIJExploring Hidden Markov Models for Virus ˇ Analysis: A Semantic Approach, Proceedings of the 46th Hawaii International Conference on System Sciences, Wailea, HI, USA, 2013, Jan 7-10, 50395048

[7]. Nataraj, Lakshmanan, et al.”Malware images: visualization and automatic classification.” Proceedings of the 8th international symposium on visualization for cyber security. ACM, 2011.

[8]. https://www.kaggle.com/c/malware-classification/forums/t/13509/brief-description-of-7th-place-solution/72485

[9]. J.O. Kephart and W.C. Arnold, “Automatic Extraction of Computer Virus Signatures,” Proc. Fourth Virus Bull. Int’l. Conf., pp. 178-184, 1994.

[10]. [10]. K. Griffin, S. Schneider, X. Hu, and T. Chiueh, “Automatic Generation of String Signatures for Malware Detection,” Proc. 12th Int’l Symp. Recent Advances in Intrusion Detection, 2009.