IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 19 January 2023, accepted 9 March 2023, date of publication 24 March 2023, date of current version 5 April 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3261562

RESEARCH ARTICLE

Explainable YouTube Video Identification Using Sufficient Input Subsets

WALEED AFANDI[1], SYED MUHAMMAD AMMAR HASSAN BUKHARI[1], MUHAMMAD U. S. KHAN[1], (Member, IEEE), TAHIR MAQSOOD[1], MUHAMMAD A. B. FAYYAZ[2], ALI R. ANSARI[3], AND RAHEEL NAWAZ[4], (Member, IEEE)

[1]Department of Computer Science, COMSATS University Islamabad, Abbottabad 22060, Pakistan
[2]OTEHM, Manchester Metropolitan University, M15 6BH Manchester, U.K.
[3]Department of Mathematics and Natural Sciences, Gulf University for Science and Technology, Mubarak Al-Abdullah 32093, Kuwait
[4]Pro Vice Chancellor (Digital Transformation), Staffordshire University, ST4 2DE Stoke-on-Trent, England, U.K.

Corresponding author: Muhammad A. B. Fayyaz (m.fayyaz@mmu.ac.uk)

This work was supported by the National Centre of Cyber Security (NCCS), Pakistan, and Higher Education Commission (HEC), under Grant RF-NCCS-023 (This project has been partially supported by Gulf University for Science and Technology and the Centre for Applied Mathematics and Bioinformatics (CAMB) under project code: ISG–278736).

ABSTRACT Neural network models are black boxes in nature. The mechanics behind these black boxes are practically unexplainable. Having the insight into patterns identified by these algorithms can help unravel important properties of the subject in query. These artificial intelligence based algorithms are used in every domain for prediction. This research focuses on patterns formed in network traffic that can be leveraged to identify videos streaming over the network. The proposed work uses a sufficient input subset (SIS) model on two separate video identification techniques to understand and explain the patterns detected by the techniques. The first technique creates the fingerprints of videos on a period-based algorithm to handle variable bitrate inconsistencies. These fingerprints are passed to a convolutional Neural Network (CNN) for pattern recognition. The second technique is based on traffic pattern plot identification that creates a graph of packet size with respect to time for each stream before passing that to a CNN as an image. For model explainability, a sufficient input subset (SIS) model is used to identify features that are sufficient to reach the same prediction under a certain threshold of confidence by the model. The generated SIS of each input sample is clustered using DBSCAN, K-Means, and cosine-based Hierarchical clustering. The clustered SIS highlight the common patterns for each class. The SIS patterns learnt by each model of three individual videos are discussed. Furthermore, these patterns are used to investigate misclassification and provide a rationale behind it to justify the working of the classifier model.

INDEX TERMS Video identification, fingerprinting, deep learning, classification, variable bitrate.

I. INTRODUCTION

In the modern era, artificial intelligence (AI) is implemented in a vast range of real-world problems ranging from cellular biology to data security. The most prominent tools of AI are machine learning (ML) and deep learning (DL) models. ML and DL models are composed of highly sophisticated algorithms which make them black-boxes in nature. One cannot conceive why a model made a certain decision. The rationale for a black-box decision is extremely important in many scenarios, such as medical image classifier for cancer detection [1], [2] and video identification in network traffic [3].

Currently, the internet traffic is encrypted using HTTPS for security purposes, which makes video identification difficult in network traffic. However, several researchers show that videos and user activity can be detected even in the encrypted network traffic using machine learning models [3], [4], [5]. The application of such internet activity identification works is useful in monitoring traffic in highly confidential government workplaces as well as national security centers.

The associate editor coordinating the review of this manuscript and approving it for publication was Gangyi Jiang.

Most streaming services, such as YouTube, Facebook, and Twitch use dynamic adaptive streaming over HTTP (DASH) protocol. The DASH protocol divides a video into small chunks and these chunks are sent to the client streaming the video. For improving the quality of experience (QoE), the chunks are encoded using variable bitrate (VBR) encoding. However, numerous studies [3], [6] show that the combination of DASH and VBR is prone to leak information about video content. Patterns about particular videos that help in identifying those videos are learned by deep learning algorithms demonstrated in [6], [7], [8], and [9]. However, explaining the judgement of these models —models' rationale —for human understanding has remained a challenge for researchers.

For model explainability, Carter et al. [10], [11] propose a methodology—sufficient input subsets (SIS)—for understanding the rationale of image processing models. The authors describe a model's rationale as an explanation for a model decision stating that a sparse subset of input features is responsible for a particular black-box decision. This paper investigate the effectiveness of SIS methodology for video identification models in network traffic. Moreover, this also provides insight into traffic patterns caused by sparse feature sets that account for video identification. Knowledge of patterns can help to develop even more efficient but simple models.

This work investigates the rationale behind the decisions of two models, the absolute difference fingerprint (ADF) [12] and the traffic pattern plot (TPP) that detect videos in network traffic. The former model creates a fingerprint for each video stream on bytes per period and trains them on a convolutional neural network (CNN) while the latter plots all the received packets' sizes to their packet arrival time of each stream and trains these plots on a CNN as well.

This research uses the aforementioned models to train streams of YouTube videos. Then the proposed algorithm by [3] and [4] is used to generate sufficient input subsets (SIS). The resultant SIS are clustered based on their similarity and leveraged for the explainability of the model. The similar patterns formed by each video's SIS are discussed in the Results and Discussion section. Model explainability is also used to investigate a misclassification to demonstrate the working of the model

II. RELATED WORK

Since the dawn of the Internet, internet networks have faced many security threats regarding privacy and eavesdropping. However, with time, such issues have been mitigated. Data flowing through encrypted networks has become harder to exploit using existing means. Reference [13] proposes a cryptography technique that is impenetrable even by brute force with a quantum computer. However, with the advent of machine learning and deep learning algorithms, researchers have started taking a different approach for deciphering network data. Work in [3] extracts bytes per seconds (BPS) and utilizes a convolutional neural network to identify the streaming video. The work [4] uses the K-means algorithm to predict the bitrate and resolution of an encrypted video stream. Similarly [5] also leverages K-means clustering for video identification on the account of network patterns. Although these patterns are evident that they exist in networks, but they are not humanely conceivable.

Although many explainable procedures, including [14] and [15] have demonstrated a model that is interpretable by humans. Reference [16] breaks down the neural network, however, such approaches are model dependent and are understandable only by experts. Often a generic model is profoundly customized to achieve great accuracy. Some models are not accessible and act as a black-box [8], [9], therefore an explainable mechanism is required that does not depend on the nature and works as standalone framework. Some model-independent approaches [17], [18], [19], [20], [21], [22] produce local explanations of the prediction f on a particular input x. These models only weigh the impact of features on the output of f at x. References [23] and [24] are based on input signal-based explanation techniques that utilize gradients of f to highlight patterns that lead to high output values of f.

This research extracts data from network traffic based on the above-mentioned video identification techniques. We extract network data and perform two dissimilar data preprocessing on them and pass these data to two unique models for training. Compared to the aforementioned techniques, this research employs a technique similar to [10] and [11] to produce subsets that allow the same decision to be reached with minimum number of features while keeping the remaining features masked. These subsets are synthesized for model explainability, visualize pattern formation in video streaming network, and highlight model over interpretation.

III. METHODOLOGY

This section discusses data acquisition and manipulation (pre-processing) for each video identification technique, and fingerprinting details.

A. DATA COLLECTION

The main source for generating video traffic used in this research is **YouTube**. 43 random non-identical monetized videos are selected for dataset creation. 55 streams of each video were captured for 120 seconds using **Wireshark** and converted into packet capture (PCAP) files. The process of playing the videos in a loop 55 times was automated using **Selenium** and **Chrome** browsers. For model explainability, this research discusses only three separate videos; Video A, B, and C.

B. ADF PRE-PROCESSING

The PCAP files contain packets sent and received by all domains. However, for ADF, the data is filtered by extracting **only the downlinks** of the specified domain (i.e., YouTube). The capture time is limited to the first 120 seconds of the video. Furthermore, the headers are removed from the packets

and the IP is masked. This filtered dataset is exported to a comma-separated values (.CSV) file format. Therefore, the resulting CSV file contains 120 columns, each column containing the number of bytes per second (BPS) in that respective second. For example, the first column contains the size of bytes received in 1^{st} second of capturing and the second column contains the size of bytes received in 2^{nd} second and so on.

1) BYTES PER PERIOD (BPP)

As discussed previously in introduction section, due to network condition inconsistency, VBR causes the bytes received at a particular second in one captured stream may shift to the next second in another captured stream or vice versa. For example, in a given stream S_1, the size of bytes received are 5, 10 and 15 and 0 bytes at 1^{st}, 2^{nd}, and 3^{rd} and 4^{th} second respectively. In a second stream S_2, the first four bytes received are 0, 5, 10, and 15 bytes in their respective seconds. The total size of bytes in the first four seconds are same in both streams, this leads to inconsistency in bytes recieved. This inconsistency in order of bytes received may cause issues in model training. To mitigate this issue, the seconds are aggregated into periods. Each period contains 6 seconds of the stream consecutively. For example, the first period contains the first 6 seconds of BPS, second period contains the next 6 seconds of the stream, and so on. Conclusively, the BPS are converted into BPP, effectively reducing the number of features from 120 to 20.

2) FINGERPRINTING

Each stream in the form of BPP is fingerprinted. This is done to increase the model's accuracy in identifying the video stream. For a video stream containing n periods, we get a sequence of $b = (b_1, b_2, \ldots, b_i, \ldots)$. For two adjacent period b_{i-1} and b_i, fingerprints can be generated as $x = (x_1, x_2, \ldots, x_i, \ldots)$. The fingerprint of a stream is formed by applying the following equation.

$$x_i = |x_i - x_{i-1}| \tag{1}$$

As the periods are subtracted adjacently, the resulting periods are now 19 and these act as the fingerprints. The fingerprints are exported to a CSV containing 19 columns.

C. TPP PRE-PROCESSING

The pre-processing of TPP is relatively simple compared to ADF. The PCAP file is used to extract all the **downlinks and uplinks** of the specified domain in terms of packets and their sizes during the first 120 seconds of capture. All the packets are plotted onto a graph. The x-axis of the graph shows the time and the y-axis shows the size of the packet. Sample TPP of three different videos A, B and C are shown in Fig. 1 Before these graphs are passed onto the CNN, they are cropped to the extent that only the plots on the graph are visible and not the x and y labels of the graph.

IV. MODEL DETAILS AND EXPERIMENTAL SETUP

A. CONVOLUTIONAL NEURAL NETWORK

The video classification model used primarily in this research is the convolutional neural network (CNN). It is one of the most popular models of deep neural networks. They are well known for their ability to extract features from the training data without requiring manual feature extraction. These models are preferred for predictions requiring pattern recognition, including image object detection [25], hate speech detection [26], [27], document classification [28], bot detection [29], human activity detection [30], malware detection [31], audio identification [32]. Typically, a CNN model is composed of the following layers:

- Convolutional layers: A convolutional layer consists of kernels (or filters) that are optimized throughout the training. The filter size is usually small to assist the model in learning more feature maps for improved accuracy. These feature maps go through an activation function (*ReLU* used in this research) and are directed towards the pooling layer.
- Pooling layers: The function of a pooling layer is to summarize the results of the previous convolutional layer in a way that no key features are lost. This action is performed to highlight recurring patterns along with reducing the memory consumption by the model. These summarized features are passed to the next in-line layer.
- Dropout layers: The dropout layer arbitrarily drops some of the outputs from the previous layer so that the model can generalize more features rather than relying on a selected set of features that can induce overfitting in the model. The number of features to be disabled is defined by p which is the probability value ranging between 0 to 1.
- Flatten layers: The flatten layer is used for dimension conversion (from multi-dimension to single-dimension) on the feature maps for them to be accepted by the dense layer.
- Fully connected layers: A dense layer is a fully connected layer having all of its neurons connected with the previous and next dense layer. The features are passed through the neurons where an activation function is applied to them and the resultant features are delivered to the next adjacent layer. The final dense layer is called an Output layer that has the same number of neurons as that of the total classes in the dataset [33].

The details of the individual CNN model constructed for the two applied techniques are discussed below:

1) ADF CONVOLUTIONAL NEURAL NETWORK

The ADF CNN model is trained on the fingerprints generated on the basis of BPP. The model has 13 input nodes in its input layer; four 1-dimensional (1D) convolution layers with *ReLU* as its activation function. Every 1D convolution layer has a max pooling layer followed by it. The fourth max pooling layer has a dropout layer next to it which is followed by a

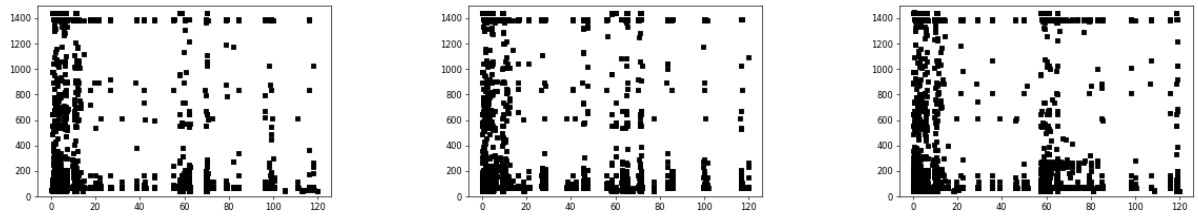

FIGURE 1. TPP of (a) video A, (b) video B, and (c) video C.

TABLE 1. Summary of hyperparameters of ADF model.

Hyperparameter	Value
No. of Conv1D layers	(1,2,3,4)
No. of filters in Conv layers	(300, 512, 512, 300)
Kernel size	(5,3,1,1)
Activation function (all layers)	ReLU
Max pooling – pool size	(1, 2, 1, 1)
Padding (all layers)	same
Dropout	0.8
Dense – neurons (output)	43
Loss function	Categorical crossentropy
Optimizer	Adam
Batch size	50
Epochs	300

TABLE 2. Summary of hyperparameters of TPP model.

Hyperparameter	Value
No. of Conv2D layers	(1,2,3)
No. of filters in Conv layers	(100, 48, 32)
Kernel size	(6,5,4)
Activation function (all layers)	ReLU
Max pooling – pool size	(2,2,2)
Padding (all layers)	Same
Dropout	0.8
Dense – neurons	(128,64,43)
Loss function	Categorical crossentropy
Optimizer	Adam
Epochs	50

flatten and a dense layer acting as the output layer with softmax activation function. The output layer has 43 nodes as the total number of labels in the dataset for this research is 43. The model optimizer is Adam optimizer. The hyperparameters of ADF CNN are summarized in Table. 1

2) TPP CONVOLUTIONAL NEURAL NETWORK

The TPP CNN is based on an image classifier model. It contains three 2-dimensional (2D) convolutional layers with a relu activation function, each followed by a 2D max pooling layer. The final pooling layer is followed by a dropout, flatten, and three dense layers where the final dense layer acts as the output layer with softmax as its activation function. The model is optimized using Adam optimizer. TPP CNN is summarized in Table.2

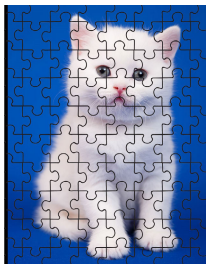

FIGURE 2. A puzzle of a kitten.

B. EXPERIMENTAL SETUP

The time complexity of SIS generation is $O(p^2k)$ where p is the total number of features in an input sample and k is the possible number of subsets that can be generated for the input sample. The model training and SIS generation rely heavily on a Graphics Processing Unit (GPU), Central Processing Unit (CPU), and extensive memory. Hence, these tasks are performed on an Intel Core i7 processor @ 3.4Ghz, 16GB DDR4 Single Channel RAM @ 2400MHz and Nvidia GeForce GTX 1060 6GB GDDR5 with 1280 CUDA cores @ 1480MHz.

V. SUFFICIENT INPUT SUBSETS (SIS)

SIS are the subset features of a given training/testing sample that alone are sufficient for a given model to make the same prediction above τ threshold confidence. This approach is relatively simple compared to existing explanation frameworks as these are easy to understand, even by non-machine learning experts. Furthermore, existing frameworks work only a limited number of models whereas this approach is compatible with most existing models. For this research, the SIS is generated from the training dataset to examine what features are learnt by the model.

Understanding SIS Through a Puzzle Analogy: Consider a puzzle of a kitty as shown in Fig. 2 Each piece of this puzzle can be considered as a feature required to identify the animal in the puzzle. To find the impact of each piece, we remove a single piece from the puzzle and ask a person to predict the animal in the puzzle with certain confidence and label that confidence with the respective puzzle piece. This procedure is applied to all the puzzle pieces recursively as shown in

Fig. 3. This procedure is called backSelect which is discussed in detail later in this paper.

Now after acquiring the information regarding the impact of each piece on the prediction confidence, the pieces are stacked in ascending order based on their labeled confidence i.e., pieces that affect the confidence the most are on the top of the stack and vice versa. Now, consider an empty puzzle sheet and start placing each puzzle piece from the stack on the sheet as depicted in Fig. 4. After placing each piece, the person is asked to predict the animal. This process is repeated until the person can identify the animal (which is a kitty in this case) with a certain confidence level. This process is referred to as findSIS and will be discussed briefly later on. When the prediction confidence criterion is met, a subset of pieces is formed i.e., this set of pieces alone is enough for the person to recognize that the puzzle is of a kitty. These SIS pieces are now discarded from the puzzle. If the person is still able to recognize the kitty in the puzzle even without the SIS pieces removed, then it is implied that there are SIS pieces present in the puzzle that can be discovered. The backselect procedure is again executed as demonstrated in Fig. 5.

A. FEATURE MASKING

To mask a feature, it can simply be removed from the input sample, however, not all models support missing features in input sample. Therefore, to keep the SIS technique model independent the following three masking mechanisms are used in this research:

- Mean mask
- Zero mask
- White mask

1) MEAN MASK

In mean masking, the mean value of each feature is calculated from the entire dataset and the feature to be masked is replaced by its respective mean value.

2) ZERO MASK

In terms of network data, replacing a feature with 0 replicates the scenario in which no byte is received in the specific period.

3) WHITE MASK

White mask is used in TPP only as it deals with images. The white mask mechanism converts a pixel to a white pixel. In TPP, the white pixel means that no packets were received in that duration.

B. SIS PROCEDURE

The SIS generation consists of 3 major parts:

- SIS Collection
- Back Select
- Find SIS

1) SIS COLLECTION

The SIS collection is the primary function that initiates the formation of SIS of a given sample. The Back Select and Find SIS functions are contained inside this function. *backSelect*() and *findSIS*() methods are called recursively until all the possible sufficient input subsets are generated.

2) BACK SELECT

The objective of the BackSelect() algorithm is to calculate the importance of each feature and sort the features accordingly. This is achieved by replacing each feature with a masked value and checking the prediction confidence level using function $f(x)$. The returned value by $f(x)$ is assigned to the feature. This is performed for each feature. The feature that affects the confidence level the most is considered to be an important feature, inversely, if the feature has little to no effect on confidence, it is considered as the least important feature. BackSelect returns a stack R containing the most important feature at the top while the least feature is at the bottom.

3) FIND SIS

After the features are estimated in terms of importance and sorted using BackSelect, the FindSIS function creates a sample X_s in which all the features are initially masked. Then successively, each feature from R is popped and inserted into the sample feature X_s and the sample is passed to f to check whether the confidence level returned by f is equal to or greater than τ ($f(x_s) > \tau$). If the condition holds true, then a subset is formed and returned back to the SIS collection function.

C. SIS GENERATION FOR ADF MODEL

In case of ADF, the input training sample x is one dimensional in which each value is the size of bytes received in the corresponding second. The S is a copy of x containing p, where p is the features of x. The S is passed to the BackSelect function which sorts the features in terms of their importance to the model. The BackSelect function returns R which is S but is sorted. The R is passed onto the FindSIS function in which a completely masked sample x_s is formed and each element (i.e., feature) in R is popped and inserted into the sample x_s and the condition $f(x_s) >= \tau$ is checked. If true, a subset for the fingerprint is formed, else another element is popped from R and the condition is checked again. In the case of ADF, a SIS contains only those periods that are required to predict the same video stream with confidence greater than or equal to τ while the other periods are replaced by the masked values.

D. SIS GENERATION FOR TPP

The SIS generation of TPP is the same as that of ADF for the most part. As a TPP is an image of resolution 174×257. If each pixel is considered as an individual feature, then the total number of features becomes approximately 44,000

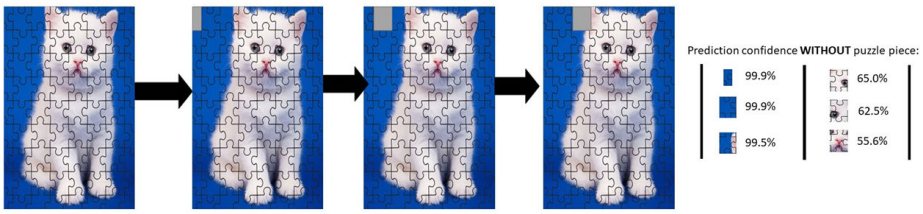

FIGURE 3. Puzzle analogy based on backSelect algorithm.

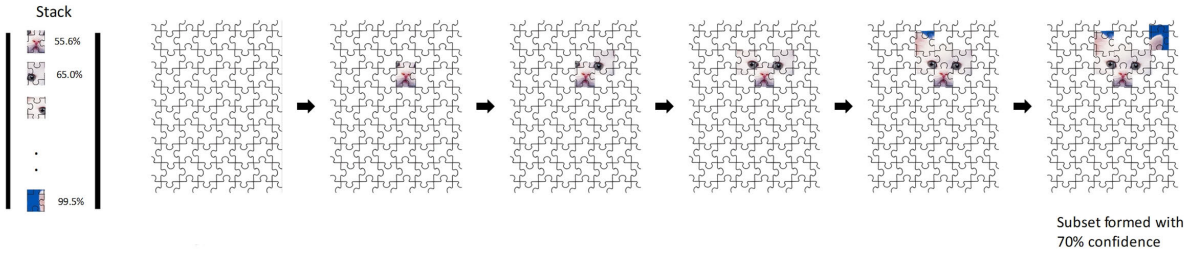

FIGURE 4. Puzzle analogy based on findSIS algorithm.

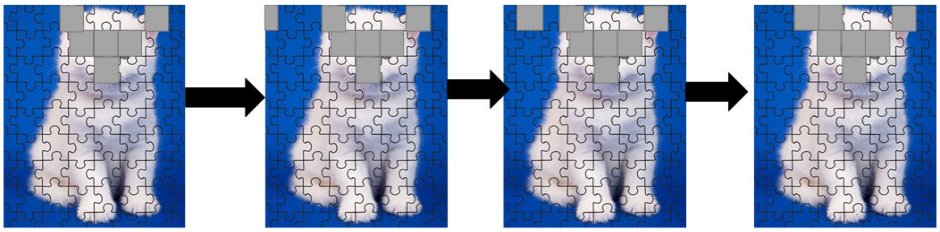

FIGURE 5. Repeating backSelect algorithm after finding first subset.

which is not feasible for SIS generation due to the high time complexity of the SIS algorithm as discussed in an experimental setup. The two-dimensional nature of image data requires the algorithm to be slightly modified to account for the excessive number of features. The image is decomposed into smaller pieces as shown in Fig.6. The Fig.6 demonstrates that an input sample image of resolution 4×4 is broken down into four 2×2 blocks. Each block acts as a separate feature. Thus, the number of features is reduced from supposedly 16 to just 4. Likewise, for TPP, the block size is set to 25×25. If the block size is too large, the generated SIS will include too much noise (unwanted pixels), and if the size is too small, the algorithm becomes too resource intensive. Thus, for the resources available in this research, a 25×25 block size is optimal for SIS generation. The masking mechanism used in SIS generation is white masking. The rest of the process for SIS generation is the same as the ADF SIS generation. The BackSelect function rearranges features (blocks for TPP image) based on their importance, whereas FindSIS inserts each feature in a completely masked image to form a SIS.

E. CLUSTERING OF SIS

All the generated SIS for each class are clustered. Clustering is an unsupervised machine learning function that classifies objects in terms of their similarity without having an insight into true or false labels [34]. Clustering is applied to the generated SIS to agglomerate all the similar patterns found among the subsets, making pattern recognition more prominent and humanly conceivable. Clustering is done through an ensemble model composed of the following clustering algorithms:

- Density-Based Spatial Clustering of Applications with Noise (DBSCAN): DBSCAN is a well-known clustering algorithm that groups the data points together with their distance, which is usually the Euclidean distance, along with a minimum number of points [35]. DBSCAN is also able to detect noise in the dataset (points that are far off from the cluster). This algorithm takes two parameters; epsilon and minPoints. The epsilon value is the distance between two points required to be considered as a part of the cluster. The minPoints is the number of points required to be considered a cluster. The parameters are optimized using graph based parameter estimation used by [36]. These parameters are estimated in a way so that the total number of clusters formed are balanced i.e., not too many or not too low. Thorough testing of these parameters shows that 3 clusters were formed for the SIS of each class along with some SIS considered as noise.

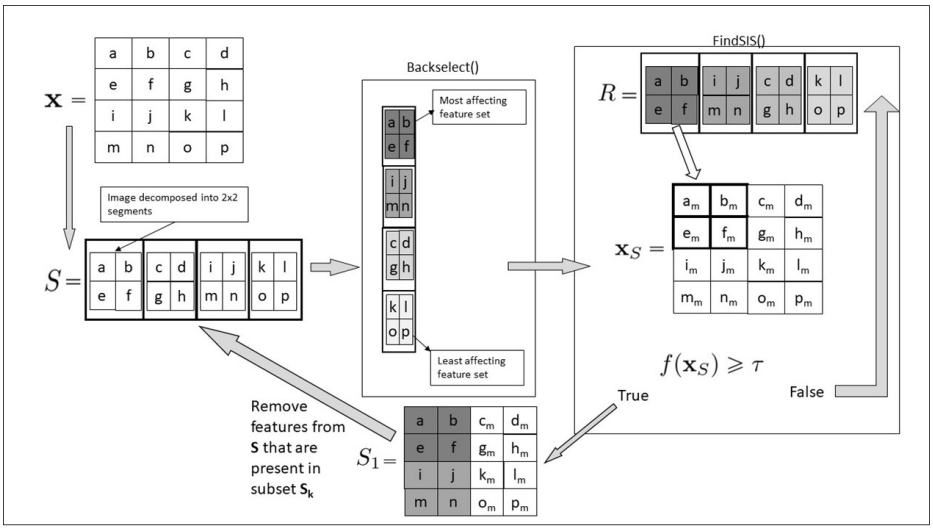

FIGURE 6. Architectural design of TPP SIS generation.

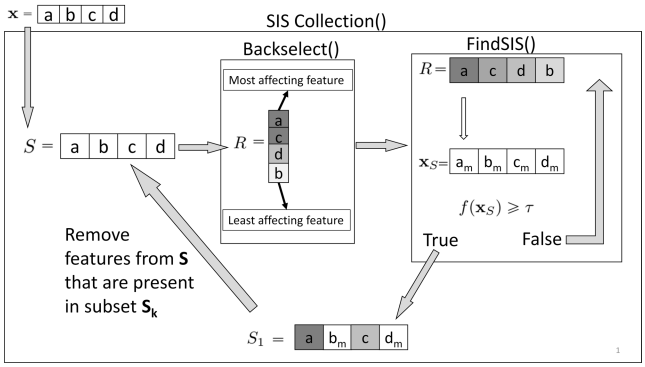

FIGURE 7. Architectural design of ADF SIS generation.

- K-means clustering: K-means algorithm is one of the basic yet most commonly used unsupervised clustering algorithms. This algorithm works by randomly deploying k centroids on the dataset. Then the positions of each centroid are optimized iteratively. The algorithm halts either when the centroids have been stabilized or the defined number of iterations are reached [37]. The point closest to the centroid is considered as a part of the cluster. For uniformity of results, the value of k is assigned 3 - the same as the number of clusters formed by DBSCAN.
- Cosine similarity-based Hierarchical clustering: Much like the above mentioned clustering algorithms, hierarchical clustering forms clusters based on similarity. At first, each point is considered as an individual cluster. Then two clusters that are close to one another in terms of similarity are merged and form a single cluster [38]. There are multiple approaches to cosine hierarchical clustering, the one implemented in this research requires a threshold value for the similarity between two points. The threshold value is optimized to form only three clusters as in DBSCAN and K-means algorithms.

F. CLUSTERING PROCEDURE

The clustering results of the three clustering algorithms are ensembled together. For instance, if a group of subsets belong to cluster '1' in DBSCAN, the same group belongs to cluster '2' in K-means and '3' in Cosine based clustering, then the cluster '1', '2', and '3' are considered as an ensembled cluster. For the simplicity of explainability, this research considers only the ensembled cluster that has the highest number of members.

VI. RESULTS AND DISCUSSION

A. PATTERNS IN ADF TECHNIQUE

As mentioned in the methodology section, two methodologies are used for creating masked values for ADF i.e., zero masked value and mean masked value. We discuss the patterns generated in both methodologies. All the feature subsets are plotted onto the graph with their normalized differences and in another graph, these plots of subsets are connected through a line, highlighting the pattern being formed in the subsets. For simplicity, we discuss the ensembled cluster with the highest number of elements. The generated SIS by fingerprint model for each YouTube video is discussed below:

1) VIDEO A

In zero masked SIS cluster, it can be observed that feature subsets are formed after the 10^{th} period in Fig. 8. A pattern is notable in these plots, as subsets at period 11 and 12 are in pairs. Likewise, this pair pattern can be observed between period 15-16, and 19-20. With the mean masked SIS, the cluster with the most members is identical to zero masked SIS and the pattern of plotted subsets is almost similar to that of zero masked SIS. However, there is a slight shift in subsets

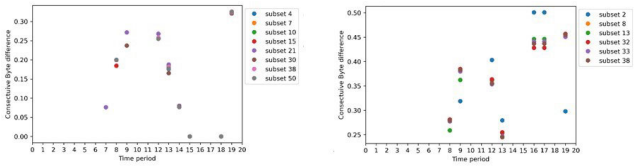

FIGURE 8. Clustered SIS of video A with zero mask (left) and mean mask (right).

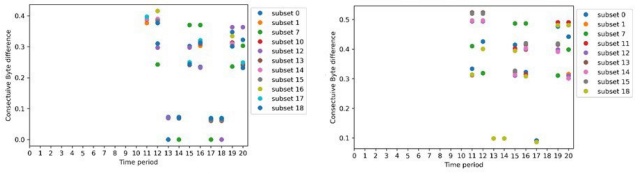

FIGURE 9. Clustered SIS of Video B with zero mask (left) and mean mask (right).

formed at period 13-14 and 17-18. Furthermore, the subsets plotted at period 11 are closer in terms of their magnitude of difference.

2) VIDEO B

Zero masked SIS, when clustered produces a set where the number of subset features are quite limited as shown in Fig. 9. The most common features in this cluster are at periods 8,9,12,13,16,17 and 19 while their respective magnitudes lie in a similar range. Mean masked SIS have a similar pattern as zero masked, however the features at periods 16 and 17 are absent. Meanwhile, the remaining subset features are quite in common with zero masked SIS in terms of magnitude.

3) VIDEO C

Both, zero masked and average masked SIS produce feature subsets where the majority have zero consecutive byte difference in Fig. 13. The periods from 10 to 20, and 12 to 20 in zero masked and mean masked SIS respectively have features containing zero. This is a clear sign of model overfitting as the majority features learned by the model are last periods where the magnitude is zero. In terms of network traffic, it implies that no bytes were received during the aforementioned periods. The overfitting in mean masked SIS is slightly less severe than zero masked SIS subsets.

B. PATTERNS IN TRAFFIC PATTERN PLOT

The pattern plots are standard black and white images. The black plots on the graph represent the packet's arrival time at x-axis and the size of the packet at y-axis. In the case of image SIS, the mask value is 255 which is essentially the value of a white pixel. The highlighted red square is equivalent to 25 × 25 size to represent the SIS block features. The traffic pattern plot SIS generated for each video is discussed below:

1) VIDEO A

Some recurring patterns can be observed from this clustered subset Fig.10. The two columns of a block from time period 0 to 20 seconds. The first column has a packet size ranging from 1000 to 1400 bytes whereas the second column ranges from row 200 to 1400 bytes with some omissions in between. The rows of the block between Packet size 1000 to 1400 bytes. The first row (between 1200 to 1400 bytes) lies between a time frame of 0 to 30 seconds and 50 to 90 seconds. The second row (between 1000 to 1200 bytes) ranges from 0 to 20 seconds and 50 to 120 seconds with some omissions in between.

2) VIDEO B

Unlike video A, the clustered SIS of video B patterns are relatively simpler in Fig. 11. The blocks can be classified into three rows. The first row (between 600 to 800 bytes) ranges from time period 40 to 70 seconds is consistent, however, in some subsets the range includes more than 70 seconds. The second row (between 400 to 600 bytes) ranges from 0 to 110 seconds with slight inconsistencies in between. Likewise, the third row (between 200 to 400) lies between a period of 0 to 110 seconds with some omissions.

3) VIDEO C

The recurring patterns of video C are quite minimal as opposed to that of videos A and B in Fig. 12. The first column (between 0 to 10 seconds) has two recurring blocks ranging between 600 to 800 and 1200 to 1400 bytes respectively. Similarly, the column at the 60th second has two blocks that are in between 600 to 800 and 1200 to 1400 bytes respectively, and are consistent throughout the subsets.

VII. INVESTIGATING MISCLASSIFICATION

Model explainability should allow investigation of misclassifications made by a classifier. Doing so allows for better understanding of the workings of the model. For such purpose, this study takes into consideration a false negative case of video A in ADF. For simplicity, only the misclassified instance with a single zero masked subset of video A is discussed in this research. The difference is visualized in Fig.14 Comparing the quantitative byte difference between 11^{th} to 12^{th} period (highlighted red), it is evident that the difference slightly increases from 0.3 to 0.4 in the valid subset whereas the difference of the same period of the false negative sample is massively reduced from 0.3 to 0.05. Conversely, in the next consecutive periods – 12^{th} to 13^{th} (highlighted green) the byte difference drops from 0.4 to 0.1 in the valid subset while the difference is unchanged (0.05 to 0.05) in the false negative sample. Contrary to the previous periods, period 13^{th} to 14^{th} (highlighted blue) shows that the byte difference remains the same (0.1 to 0.1) in the subset whereas the difference drops from 0.05 to 0. Both samples show an increasing byte difference trend in the period 14^{th} to 15^{th} with varying magnitudes. Pressing forward, the original

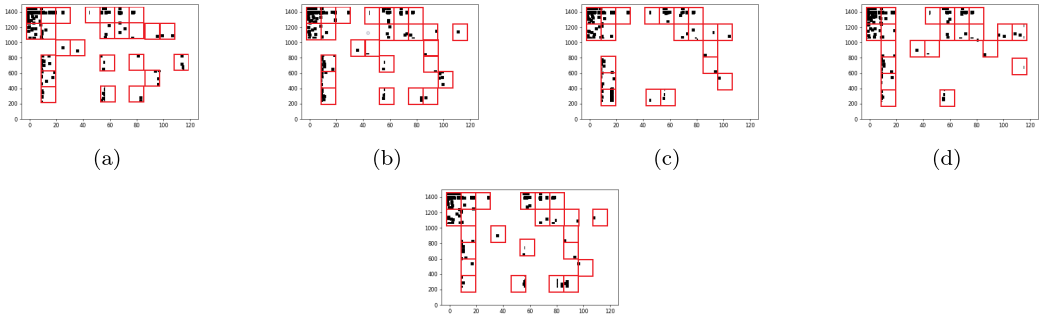

FIGURE 10. Clustered subsets of video A.

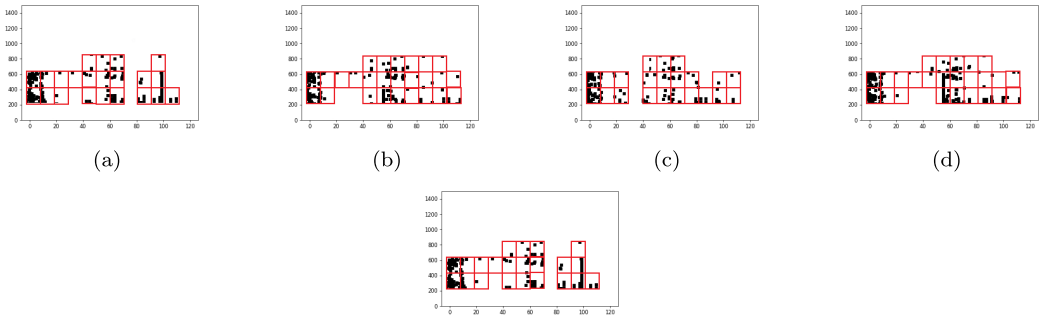

FIGURE 11. Clustered subsets of video B.

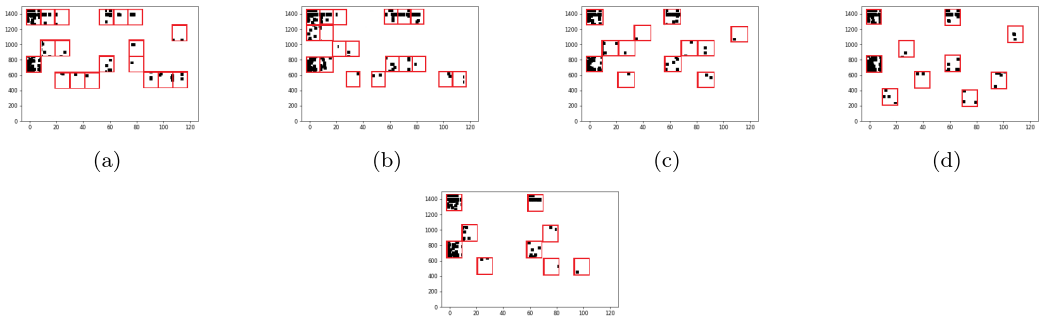

FIGURE 12. Clustered subsets of video C.

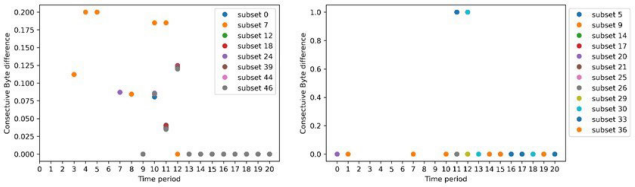

FIGURE 13. Clustered SIS of Video C with mean mask (left) and zero mask (right).

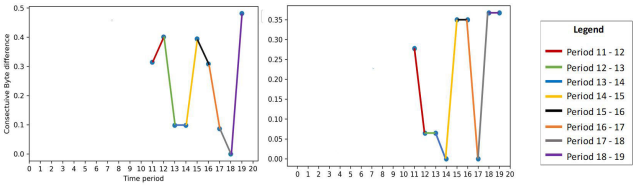

FIGURE 14. Video A subset (left) and false negative classified sample of video A (right).

sample shows a drop in difference whereas the difference is constant in the misclassified sample between period 15^{th} to 16^{th} (highlighted black). Period 16^{th} to 17^{th} (highlighted orange) shows a similar pattern of downfall in difference in

both samples but the magnitude of the difference is varying. A major contradicting trend can be observed between periods 17^{th} to 18^{th} (highlighted gray); the subset sample shows a further decrease in byte difference, but the trend is increasing massively in the misclassified sample. The period 18^{th} to 19^{th} (highlighted violet) shows an increasing trend of byte difference in the subset while the sample shows a consistent difference trend Inspecting the false negative sample concludes that this sample is a massive outlier which leads to incorrect prediction of the model. Furthermore, this investigation also provides insight into the working of the model, and what patterns is it searching for, and what deviations in patterns cause it to misclassify.

VIII. CONCLUSION AND FUTURE WORK

The SIS subsets of network patterns give a detailed insight into patterns that are formed during video streaming. The SIS subsets also emphasize that most recurring patterns are formed after 60 seconds of video streaming, therefore, current video identification techniques require at least more than 60 seconds of streaming data for accurate prediction. Furthermore, SIS subsets demonstrate that video identification methods can be subjected to model overinterpretation if the dataset contains long periods of empty bytes. For future work, we try to utilize these subsets by treating them as data samples and re-train them to see whether these subsets help in improving the model accuracy. Moreover, as we are currently limited by the technology of our time, with the availability of more efficient computing resources, we can synthesize per pixel based SIS of TPP to diminish the noise completely and get a much better insight into network patterns.

REFERENCES

[1] J. Kleinberg, H. Lakkaraju, J. Leskovec, J. Ludwig, and S. Mullainathan, "Human decisions and machine predictions," *Quart. J. Econ.*, vol. 133, no. 1, pp. 237–293, 2017.

[2] N. V. Patel, "Why doctors aren't afraid of better, more efficient AI diagnosing cancer," The Daily Beast, Tech. Rep., 2017. [Online]. Available: https://www.thedailybeast.com/why-doctors-arent-afraid-of-better-more-efficient-ai-diagnosing-cancer?ref=scroll

[3] M. U. S. Khan, S. M. A. H. Bukhari, S. A. Khan, and T. Maqsood, "ISP can identify YouTube videos that you just watched," in *Proc. Int. Conf. Frontiers Inf. Technol. (FIT)*, Dec. 2021, pp. 1–6.

[4] S. Wassermann, M. Seufert, P. Casas, L. Gang, and K. Li, "Let me decrypt your beauty: Real-time prediction of video resolution and bitrate for encrypted video streaming," in *Proc. Netw. Traffic Meas. Anal. Conf. (TMA)*, 2019, pp. 199–200.

[5] Y. Liu, S. Li, C. Zhang, C. Zheng, Y. Sun, and Q. Liu, "ITP-KNN: Encrypted video flow identification based on the intermittent traffic pattern of video and k-nearest neighbors classification," in *Proc. 20th Int. Conf. Comput. Sci. (ICCS)*. Amsterdam, The Netherlands: Springer, Jun. 2020, pp. 279–293.

[6] R. Schuster, V. Shmatikov, and E. Tromer, "Beauty and the burst: Remote identification of encrypted video streams," in *Proc. 26th USENIX Secur. Symp. (USENIX Security)*, 2017, pp. 1357–1374.

[7] A. Dvir, A. K. Marnerides, R. Dubin, and N. Golan, "Clustering the unknown—The YouTube case," in *Proc. Int. Conf. Comput., Netw. Commun. (ICNC)*, Feb. 2019, pp. 402–407.

[8] A. Reed and M. Kranch, "Identifying HTTPS-protected Netflix videos in real-time," in *Proc. 7th ACM Conf. Data Appl. Secur. Privacy*, Mar. 2017, pp. 361–368.

[9] J. Gu, J. Wang, Z. Yu, and K. Shen, "Walls have ears: Traffic-based side-channel attack in video streaming," in *Proc. IEEE Conf. Comput. Commun. (IEEE INFOCOM)*, Apr. 2018, pp. 1538–1546.

[10] B. Carter, J. Mueller, S. Jain, and D. Gifford, "What made you do this? Understanding black-box decisions with sufficient input subsets," in *Proc. 22nd Int. Conf. Artif. Intell. Statist.*, 2019, pp. 567–576.

[11] B. Carter, S. Jain, J. W. Mueller, and D. Gifford, "Overinterpretation reveals image classification model pathologies," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 34, 2021, pp. 15395–15407.

[12] W. Afandi, S. M. A. H. Bukhari, M. U. S. Khan, T. Maqsood, and S. U. Khan, "Fingerprinting technique for YouTube videos identification in network traffic," *IEEE Access*, vol. 10, pp. 76731–76741, 2022.

[13] R. Bavdekar, E. J. Chopde, A. Bhatia, K. Tiwari, S. J. Daniel, and Atul, "Post quantum cryptography: Techniques, challenges, standardization, and directions for future research," 2022, *arXiv:2202.02826*.

[14] Y. Sha and M. D. Wang, "Interpretable predictions of clinical outcomes with an attention-based recurrent neural network," in *Proc. 8th ACM Int. Conf. Bioinf., Comput. Biol., Health Informat.*, Aug. 2017, pp. 233–240.

[15] T. Lei, R. Barzilay, and T. Jaakkola, "Rationalizing neural predictions," 2016, *arXiv:1606.04155*.

[16] W. J. Murdoch, P. J. Liu, and B. Yu, "Beyond word importance: Contextual decomposition to extract interactions from LSTMs," 2018, *arXiv:1801.05453*.

[17] M. T. Ribeiro, S. Singh, and C. Guestrin, "'Why should I trust you?': Explaining the predictions of any classifier," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2016, pp. 1135–1144.

[18] D. Baehrens, T. Schroeter, S. Harmeling, M. Kawanabe, K. Hansen, and K.-R. Müller, "How to explain individual classification decisions," *J. Mach. Learn. Res.*, vol. 11, no. 6, pp. 1803–1831, 2010.

[19] K. Simonyan, A. Vedaldi, and A. Zisserman, "Deep inside convolutional networks: Visualising image classification models and saliency maps," 2013, *arXiv:1312.6034*.

[20] S. Bach, A. Binder, G. Montavon, F. Klauschen, K.-R. Müller, and W. Samek, "On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation," *PLoS ONE*, vol. 10, no. 7, Jul. 2015, Art. no. e0130140.

[21] A. Shrikumar, P. Greenside, and A. Kundaje, "Learning important features through propagating activation differences," in *Proc. Int. Conf. Mach. Learn.*, 2017, pp. 3145–3153.

[22] M. Sundararajan, A. Taly, and Q. Yan, "Axiomatic attribution for deep networks," in *Proc. Int. Conf. Mach. Learn.*, 2017, pp. 3319–3328.

[23] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *Proc. 13th Eur. Conf. Comput. Vis. (ECCV)*. Zürich, Switzerland: Springer, Sep. 2014, pp. 818–833.

[24] P.-J. Kindermans, K. T. Schütt, M. Alber, K.-R. Müller, D. Erhan, B. Kim, and S. Dähne, "Learning how to explain neural networks: PatternNet and PatternAttribution," 2017, *arXiv:1705.05598*.

[25] R. L. Galvez, A. A. Bandala, E. P. Dadios, R. R. P. Vicerra, and J. M. Z. Maningo, "Object detection using convolutional neural networks," in *Proc. IEEE Region 10 Conf. (TENCON)*, Oct. 2018, pp. 2023–2027.

[26] M. U. S. Khan, A. Abbas, A. Rehman, and R. Nawaz, "HateClassify: A service framework for hate speech identification on social media," *IEEE Internet Comput.*, vol. 25, no. 1, pp. 40–49, Jan. 2021.

[27] Z. Mahmood, I. Safder, R. M. A. Nawab, F. Bukhari, R. Nawaz, A. S. Alfakeeh, N. R. Aljohani, and S.-U. Hassan, "Deep sentiments in Roman Urdu text using recurrent convolutional neural network model," *Inf. Process. Manage.*, vol. 57, no. 4, Jul. 2020, Art. no. 102233.

[28] I. Safder, S.-U. Hassan, A. Visvizi, T. Noraset, R. Nawaz, and S. Tuarob, "Deep learning-based extraction of algorithmic metadata in full-text scholarly documents," *Inf. Process. Manage.*, vol. 57, no. 6, 2020, Art. no. 102269.

[29] S. Mohammad, M. U. S. Khan, M. Ali, L. Liu, M. Shardlow, and R. Nawaz, "Bot detection using a single post on social media," in *Proc. 3rd World Conf. Smart Trends Syst. Secur. Sustainablity (WorldS4)*, Jul. 2019, pp. 215–220.

[30] M. U. S. Khan, A. Abbas, M. Ali, M. Jawad, and S. U. Khan, "Convolutional neural networks as means to identify apposite sensor combination for human activity recognition," in *Proc. IEEE/ACM Int. Conf. Connected Health, Appl., Syst. Eng. Technol.*, Sep. 2018, pp. 45–50.

[31] M. Khan, D. Baig, U. S. Khan, and A. Karim, "Malware classification framework using convolutional neural network," in *Proc. Int. Conf. Cyber Warfare Secur. (ICCWS)*, Oct. 2020, pp. 1–7.

[32] K. J. Piczak, "Environmental sound classification with convolutional neural networks," in *Proc. IEEE 25th Int. Workshop Mach. Learn. Signal Process. (MLSP)*, Sep. 2015, pp. 1–6.

[33] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, Feb. 2015.
[34] N. Grira, M. Crucianu, and N. Boujemaa, "Unsupervised and semi-supervised clustering: A brief survey," *Rev. Mach. Learn. Techn. Process. Multimedia Content*, vol. 1, pp. 9–16, Jul. 2004.
[35] K. M. Kumar and A. R. M. Reddy, "A fast DBSCAN clustering algorithm by accelerating neighbor searching using groups method," *Pattern Recognit.*, vol. 58, pp. 39–48, Oct. 2016.
[36] N. Rahmah and I. S. Sitanggang, "Determination of optimal epsilon (EPS) value on DBSCAN algorithm to clustering data on peatland hotspots in Sumatra," *IOP Conf. Ser., Earth Environ. Sci.*, vol. 31, Jan. 2016, Art. no. 012012.
[37] D. T. Pham, S. S. Dimov, and C. D. Nguyen, "Selection of K in K-means clustering," *Proc. Inst. Mech. Eng. C, J. Mech. Eng. Sci.*, vol. 219, no. 1, pp. 103–119, 2005.
[38] J. Murthy, "Clustering based on cosine similarity measure," Tech. Rep., 2012.

WALEED AFANDI is a Research Assistant with the Department of Computer Science, COMSATS University Islamabad, Abbottabad. His research interest includes computer security.

SYED MUHAMMAD AMMAR HASSAN BUKHARI is a Senior Research Assistant with the Department of Computer Science, COMSATS University Islamabad, Abbottabad. His research interest includes computer security.

MUHAMMAD U. S. KHAN (Member, IEEE) received the Ph.D. degree in electrical and computer engineering from North Dakota State University, USA, in 2015. He is an Assistant Professor with COMSATS University Islamabad, Abbottabad. His research interests include data science, artificial intelligence, and computer security.

TAHIR MAQSOOD is currently an Assistant Professor with COMSATS University Islamabad, Abbottabad, Pakistan. His research interests include resource allocation, multi/manycore systems, reliable systems, the Internet of Things, and mobile edge computing.

MUHAMMAD A. B. FAYYAZ received the bachelor's degree in mechanical engineering from Manchester Metropolitan University (MMU), in 2017, and the Ph.D. degree from MMU. He received a full scholarship for his Ph.D. studies from MMU.

He is currently a Lecturer of business analytics and statistics and visualization with the OTEHM Department, MMU. His Ph.D. research topic was the integration of AI into the safety aspect of the railway system in Britain. He has strong academic and technical experience in applied AI, in particular, computer vision and data analytics. He is involved in different national and international research projects in applied AI and data analytics.

ALI R. ANSARI has over 30 years of experience in teaching and research in the area of applied mathematics. He was the Dean of the College of Arts and Sciences, Gulf University for Science and Technology, Kuwait, for ten years, where he is currently a Professor of applied mathematics. He has more than 100 publications in the area of applied mathematics.

RAHEEL NAWAZ (PFHEA) (Member, IEEE) is currently the Pro Vice-Chancellor (Digital Transformation) with Staffordshire University, where he provides strategic leadership and management of the university's continuing digital transformation agenda and leads transformative action across the university to deliver enhanced academic experience. Before that, he was the Director of the Business Transformations Research Centre, Manchester Metropolitan University (MMU), where he led nearly 200 researchers specializing in digital transformation, analytics, Industry 4.0, transforming markets, and transforming places. He was also the Director of Digital Technology Solutions, MMU, where he led Met's Award-Winning Cross-Faculty Digital Degree Apprenticeships portfolio delivering UG/PG programs to nearly 900 apprentices from more than 70 leading technology employers. He was also the founding Head of the Apprenticeships Research Unit, Text and Data Mining Laboratory, MMU. He is a renowned expert in industry-academia co-creation, especially for high-impact work-integrated programs, such as Degree Apprenticeships. He has advised on the establishment and launch of such programs in Spain, New Zealand, Canada, and Pakistan. He has addressed the Westminster Education Policy Forum twice and has recently been commissioned by the QAA to systematically analyze the degree apprenticeship pedagogies across English universities and advise the sector on best academic practices. He is also a leading researcher in artificial intelligence and digital education. He holds several adjunct professorships and scientific directorships across Asia and North America. He sits on the boards of research and charitable organizations, such as the National Centre for Artificial Intelligence (Pakistan), TechSkills (U.K.), and NTF (U.K.). He has advised national policy organizations, including the Prime Minister's Task Force on Science and Technology (Pakistan). He has authored over 150 peer-reviewed research articles and his career grant capture stands at over 14 million. He has graduated with 19 Ph.D. students so far. According to Google Scholar, he is among the top 10 most cited scholars in the world in the fields of digital transformations, applied artificial intelligence, and educational data science.