IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 20 September 2023, accepted 9 October 2023, date of publication 11 October 2023, date of current version 18 October 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3323927

RESEARCH ARTICLE

Tabular-to-Image Transformations for the Classification of Anonymous Network Traffic Using Deep Residual Networks

NATHAN BRINER[1], DRAKE CULLEN[1], JAMES HALLADAY[1], DARRIN MILLER[1], RILEY PRIMEAU[1], ABRAHAM AVILA[1], RAM BASNET[1], AND TENZIN DOLECK[2]

[1]Department of Computer Science and Engineering, Colorado Mesa University (CMU), Grand Junction, CO 81501, USA
[2]Faculty of Education, Simon Fraser University, Burnaby, BC V5A 1S6, Canada

Corresponding author: Ram Basnet (rbasnet@coloradomesa.edu)

This work was supported by the State of Colorado through funds appropriated for Cybersecurity Law Dubbed "Cyber Coding Cryptology for State Records."

ABSTRACT With the meteoric rise in anonymous network traffic data, there is a considerable need for effective automation in traffic identification tasks. Though many shallow and deep machine learning network traffic classification solutions have been proposed, they often rely on tabular data, making them unable to detect complex spatial relationships. However, recent advancements in computer processing power have increased the viability of transforming tabular data into images for training deep convolutional neural networks, transforming structured data problems into spatial ones. To identify the most effective methods for representing tabular anonymous network traffic data as images, we compared five deep learning classifiers trained on data from different tabular-to-image algorithms–Image Generator for Tabular Data (IGTD), DeepInsight, vector-of-feature wrapping (normalized and non-normalized), and our newly introduced Binary Image Encoding (BIE) technique in the classification of eight network application types. Furthermore, we examine whether deep residual models trained on tabular-to-image data can outperform the top-performing shallow learner, XGBoost, at classifying anonymous network traffic. We found that ResNet-50, a pre-trained instance of deep residual network, trained on image datasets using IGTD and the novel Binary Image Encoding outperformed XGBoost trained on tabular data. Our ResNet-50 models trained using IGTD and BIE achieved F1-scores of 96.0% and 98.49% respectively, improving on the baseline of 95.1% achieved by XGBoost.

INDEX TERMS Tabular-to-image techniques, binary image encoding, convolutional neural networks, network traffic, anonymous traffic, deep learning, XGBoost, image generator for tabular data, DeepInsight, ResNet-50.

I. INTRODUCTION

Network traffic classification is crucial for improving network management and security [1]. For instance, real-time applications like video streaming may require lower latency for a better user experience than web browsing [2]. Classifying this traffic can enable better optimization by internet service providers (ISPs) to prioritize real-time applications with more suitable network nodes [1]. Moreover, classifying network traffic may aid in malicious network traffic interception, which local governments typically mandate [3], [4], [5]. Automation of this task has garnered greater interest as the scale of network traffic increases and new threats to network security reveal themselves.

Using the information in each packet that was transmitted or through a collection of packets and their metadata, called a flow, ISPs can classify traffic based on the application that produced it and optimize their infrastructure to scale to the evolving needs of their customers [6]. Due to the emergence of a suite of encryption and anonymization technologies such

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

as Secure Shell Protocol (SSH), Hypertext Transfer Protocol (HTTPS), The Onion Router (TOR), and Virtual Private Networks (VPNs), it can be difficult to rely on conventional techniques to discover the origin of the anonymous and encrypted traffic [7]. To solve this problem, machine learning algorithms have been successfully employed to classify the applications producing network traffic [8], [9], [10].

A promising new machine learning approach to classifying network traffic is transforming structured tabular network traffic data into unstructured images. Audio, visual, and raw packet capture data are examples of unstructured data, whereas structured data is typically numerical or categorical data organized in a tabular form [11]. The basic premise behind tabular-to-image (T2I) transformations is to convert structured data into a form more suitable for deep learning algorithms. Deep learners tend to outperform shallow learners when it comes to the classification of unstructured data, so by converting structured data into unstructured data these deep classifiers can be utilized to potentially improve predictive performance [11]. Convolutional neural networks (CNNs) are one such model that can exploit properties such as locality and order between components of the data [12]. Prior research [9], [13], [14] has proven the effectiveness of CNNs in classifying network traffic when trained on T2I data. Other CNN variants, such as deep residual networks (ResNet), have also yielded high performance in a broad range of classification tasks.

While previous research has established the potential of CNNs trained on T2I data, only a limited number of these techniques have been tested in the network traffic domain [9], [13]. Moreover, there is minimal research comprehensively comparing the effectiveness of these T2I methods. With numerous T2I methods presented throughout several problem domains, determining the best method to use in the network traffic domain can better optimize network traffic classification tasks. These gaps in knowledge served as motivation for the experiments presented in this paper.

In our work, we measure the efficacy of five T2I algorithms in classifying anonymous network traffic by comparing five ResNet-50 classifiers trained on data generated with IGTD, DeepInsight, feature wrapping (normalized and non-normalized), and Binary Image Encoding (BIE), a new T2I method introduced in this work. We also compare these models to the shallow learner, XGBoost, trained on the original tabular data as a baseline comparator. Using this methodology, we can find whether T2I techniques allow the ResNet classifier to outperform XGBoost and identify the most optimal T2I technique for the network traffic problem domain.

The following are the major contributions of our work:

- Introduce BIE, a new T2I algorithm that utilizes data encoded as binary representations of double-precision floating point numbers. We believe BIE can be applied to many problem domains, though it may have specific advantages for network traffic classification.
- Provide a direct comparison of the efficacy among various T2I techniques. An experimental comparison of many T2I algorithms has not been explored in previous research for network traffic classification.
- Apply DeepInsight and IGTD T2I methods to the network traffic domain. These methods were initially introduced and tested on genomic data and their effectiveness when applied to classifying network traffic has not been previously explored.
- Create an open-source image dataset for further evaluation of the T2I techniques.
- Establish that T2I methods employed with ResNet-50 can outperform shallow classifiers on the classification of anonymous network traffic.

The rest of the article is structured as follows: Section II discusses related T2I and network traffic research and existing knowledge gaps, Section III introduces the datasets used in this work, Section IV introduces the classifiers used in this work, Section V explains the T2I methods that are the focus of this work, Section VI outlines our experiment methodology, Section VII presents the findings of our experiments, Section VIII concerns the limitations and future areas of study, and the paper concludes in Section IX.

II. RELATED WORKS

Our literature review found that the related works could be loosely categorized into three groups. Section II-A groups the related works that primarily focus on anonymous traffic classification regardless of the machine learning algorithms used. Section II-B aggregates works that primarily use CNNs in anonymous network traffic classification and section II-C groups works that use tabular-to-image techniques from different problem domains.

A. ANONYMOUS NETWORK TRAFFIC CLASSIFICATION

Anonymous network traffic classifiers categorize network traffic using machine learning models. The following works establish pre-existing approaches used for anonymous traffic classification [13], [14], [15], [16].

Allhusen et al. [15] trained multiple shallow learners to classify darknet and benign internet traffic from the CIC-Darknet2020 (CIC-DN) dataset (see section III-A). They analyzed the effects of the following feature groups on model performance: all original features after data cleaning, all features excluding source and destination port, 11 manually selected features, and the manually selected features without source and destination port. The Ridge-300 classifier outperformed all other models with an accuracy of 99% on their selected feature set.

Gupta et al. [16] trained an XGBoost model to classify Tor, VPN, and normal traffic from the CIC-DN dataset. The authors selected 36 of the original 83 features to reduce redundancy and increase classification efficiency. They found that XGBoost obtained the highest classification accuracy (98%) compared to seven other classifiers and concluded that the classifier still performed well despite the class imbalance in the dataset.

Lan et al. [13] proposed a deep-learning solution for darknet traffic identification and application classification called DarknetSec. Their method consists of custom attention-based 1D CNN and is compared to other state-of-the-art classifiers such as VGG19 with RF on the CIC-DN dataset. They found that their proposed method outperformed the other tested classifiers yielding >92% accuracy on 8 different application classes.

He and Li [14] developed a one-dimensional CNN model to classify anonymous proxy traffic with smaller image sizes. First, they converted two-way and one-way Spatio-temporal features to one-dimensional images. Next, they compared their method to other CNN-based models on the ISCXVPN2016 dataset as well as a self-generated dataset using Shadowsocks and Wireshark. Their method attained the same performance (>99%) as other methods while reducing memory storage by over 90% and minimizing computational overhead.

B. NETWORK TRAFFIC CLASSIFICATION WITH CNNs

CNN-based deep learning techniques are becoming a more popular approach to network traffic classification problems as the rapid growth of computation power enables quicker model training [9], [17].

Krupski et al. [17] surveyed 136 papers concerning CNN techniques in the network traffic domain, giving a specific focus on data transformation schemes. They created a taxonomy to categorize different data transformation and CNN techniques, differentiating them by their network data type, data transformation, CNN model structure, and input dimensionality. They found that 2D encodings utilizing feature wrapping and one-hot encoding were some of the most common techniques used in the existing literature.

Lashkari et al. [9] introduced DeepImage, a tabular-to-image pipeline for detecting and classifying Darknet traffic using CIC-DN dataset. DeepImage synthesizes gray-scaled images composed of the most important features from the dataset. A custom CNN was trained on the images to detect and characterize Darknet traffic with an accuracy of 86% when classifying among eight application types.

C. TABULAR TO IMAGE ALGORITHMS

In order to leverage the strengths of CNNs and improve classifier accuracy on tabular data, Tabular-to-Image (T2I) algorithms were introduced in previous works [11], [17], [18].

Sun et al. [11] proposed SuperTML, a technique for transforming tabular data into image data that can be paired with pre-trained CNNs for advanced classification tasks. SuperTML works by arranging feature values onto a 2D image. Features of greater importance are projected with larger font sizes. Moreover, SuperTML reduces the need for data preprocessing as missing values are projected as '?', and non-numeric values are placed on the image without the need for encoding. They tested SuperTML data with a pre-trained CNN and compared it with XGBoost on three separate datasets. SuperTML performed equally well or outperformed XGBoost in each test.

Buturović and Miljković [18] developed a method for classifying tabular data with CNNs through an image-generating algorithm called Tabular Convolution (TAC). This method treats input vectors as kernels and converts the data into an image using convolutions of a fixed base image. Features are converted to kernels by creating a square matrix with an odd number of rows and columns. If the number of features is not the square root of an odd number, the square is either padded or trimmed towards the nearest odd square. TAC was applied to gene expression data and trained using ResNet, and results were compared to shallow classifiers (XGBoost, LightGBM, and Support Vector Machines). TAC outperformed all shallow learning methods obtaining an accuracy of 91.1% compared to the highest performing shallow learner's accuracy of 89.6%. This result was obtained using several thousand epochs of training; whereas a similar performance to non-CNN methods was seen when using 50 epochs. They conclude that the additional computation time required for TAC is negligible on modern computer architecture.

Table 1 compares the research of prior literature that explored the CIC-DN dataset, T2I techniques, or similar classifiers.

D. MOTIVATION AND PURPOSE

From the literature review, we find that insufficient research on T2I methods and the application of these methods for anonymous network traffic classification have left the following gaps in knowledge:

- Research conducted on T2I methods is often only tested in the domain in which the technique was created, such as genomic datasets in the case of IGTD [12] and OmicsMapNet [19].
- Previous image generation techniques applied to network traffic are primarily formed from raw packet capture (PCAP) data as opposed to tabular data [17], [20], [21], [22], [23], [24]. The focus has been on applying CNN architectures to a specific dataset, instead of the importance of T2I techniques.
- There is minimal research providing direct performance comparisons of several T2I techniques

We address the gaps in existing research by empirically evaluating T2I methods in the network traffic domain while providing a detailed analysis of five T2I schemes. Among all the methods we explored, two have yet to be applied in the network traffic domain and BIE is a novel technique, to our knowledge.

III. DATASET

In this work, we use the CIC-DN dataset balanced with synthetic SMOTE data from CMU-SynTraffic-2022 (CMU) dataset. This data was then used in conjunction with the five T2I algorithms to create our CMU-SynTraffic2023-ImageDataset (CMU-I). We explain these datasets further in the next subsections.

A. CIC-Darknet2020

Lashkari et al. [9] amalgamated the CIC-Darknet2020 (CIC-DN) dataset by combining their ISCX-Tor2016 [25] and ISCX-VPN2016 [26] datasets. The CIC-DN dataset is provided in both raw Packet-Capture (PCAP) files as well as tabular data files that were preprocessed over a fixed time interval using CIC-FlowMeter v4.0 [27]. The tabular data samples consist of time-based features such as flow duration as well as statistical features which makes them highly representative of traffic flows. The dataset consists of eight anonymous traffic application types and contains 117,620 samples encompassing both Tor and VPN traffic. This dataset was chosen for our experiments due to its comprehensive selection of application types, having an adequate number of samples for training, and being well researched in prior works [9], [13], [15], [16], [28].

The eight application types comprising CIC-DN are audio streaming (Vimeo and YouTube), browsing (Firefox and Chrome), chat (ICQ, AIM, Skype, Facebook, and Hangouts), email (SMTPS, POP3S, and IMAPS), file transfer (Skype, FTP over SSH (SFTP) and FTP over SSL (FTPS) using Filezilla and an external service), p2p (uTorrent and Transmission), video streaming (Vimeo and YouTube), and VoIP (Facebook, Skype, and Hangouts voice calls). Class imbalance was an apparent problem with the original dataset as 47% of the samples are p2p traffic while VoIP and email samples consist of less than 1% of the total data. We address this limitation with the synthetic data generation scheme discussed in the following section.

B. DATA BALANCING AND CLEANING

Since models trained on imbalanced datasets often perform poorly in real-world deployment, it was necessary to balance the CIC-DN data to improve model generalizability [30]. In our previous work, we tested the viability of several data generation techniques to synthesize and balance the network traffic data in the CIC-DN dataset [29]. We found Synthetic Minority Oversampling Technique (SMOTE) to be the top-performing upsampling technique as it improved the F1-score over baseline by 7.5% [29]. The upsampled network data we created was amalgamated and published as CMU-SynTraffic-2022 (CMU) [28]. In this work, we utilize the real network traffic data from the CIC-DN dataset upsampled with synthetic SMOTE data from the CMU dataset as the baseline tabular dataset for our experiments.

From our tabular dataset, 14 zero-valued features and six additional features–Flow-id, Source/Destination IP, Timestamp, and Source/Destination port–were removed as they either overfit the model or contained duplicate information. This process left 64 features in the resulting dataset. After removing samples containing NaN and Inf values and up-sampling minority classes, our final training data contained 240,000 samples with 30,000 samples in each class. We used an 80/20 train-test split for the training of our models. This means that 80% of our data was used for training while the remaining 20% was used for testing the models and calculating our performance metrics.

C. TABULAR-TO-IMAGE DATASETS

We employ each of the five T2I techniques to transform all 240,000 samples into corresponding images. The images are grouped into folders based on their application type. The dataset dubbed CMU-SynTraffic2023-ImageDataset (CMU-I) is published online [31] for further scrutiny. Section VI-C provides detailed insight into the generated images while providing visualizations of selected samples.

IV. CLASSIFIERS

This section briefly discusses the XGBoost and ResNet-50 classifiers evaluated in our experiments as well as our reasoning for selecting these specific classifiers.

A. XGBoost

XGBoost is an optimized distributed gradient boosting classifier that has become the tool of choice in many machine-learning applications due to its high performance [32], [33]. XGBoost is built upon gradient boosting, which is an ensemble technique that combines the output of multiple weaker machine learning models to produce a more accurate prediction [34]. XGBoost provides L1 and L2 regularization to tune and further reduce overfitting and reduce loss; mainly it enables users to tune various hyperparameters to constrain trees, makes adjustments in the learning rate during the learning process, and provides random sampling techniques [34]. XGBoost was chosen as the baseline classifier as it outperformed all other classifiers in previous similar experiments and is a top-performing classifier over a wide range of experiments [11], [29], [32], [33].

We used gridsearch to optimize our XGBoost model. Gridsearch, is an algorithm which automatically tries different hyperparameter values during training to find the most performant combination. Through the use of gridsearch we found that our XGBoost performed optimally with a learning rate of 1, a max depth of 9, and 180 estimators. Our implementation of XGBoost used the default values for L1 and L2 regularization which were $\alpha = 0$ and $\lambda = 1$ respectively.

B. ResNet

CNNs have produced high accuracy in image classification and increasing the depth of these networks results in improved classification accuracy. However, deeper neural networks are more difficult to train because simply stacking more layers onto a network introduces the problem of vanishing/exploding gradients [35]. As layers are stacked, the partial derivative of the loss function will either approach zero and vanish, or approach infinity, causing it to explode. Neural networks utilize this value during backpropagation to adjust the weights of nodes. With a vanished or exploded partial derivative, the network is unable to learn as it can no longer adequately update the weights in the network.

ResNet is a deep residual neural network introduced to mitigate this drawback through residual learning. The residual aspect of ResNet allows for its enhancement over other CNNs because it can create a network with more depth. In a simple deep network, the output from each convolutional layer is passed directly as the input to the next layer which causes vanishing/exploding gradients. In comparison, ResNet introduces residual connections that enable the network to skip one or more layers. These connections allow information to directly propagate to all layers of the networks. As a result, ResNet models have fewer filters and lower complexity than other neural networks, such as VGG nets [36]. ResNet models also show significantly higher accuracy than previous models in the field. In our experiments, we utilize a specific instance of ResNet with 50 layers called ResNet-50 due to its smaller training times and low error while being a top CNN classifier for computer vision [35].

We apply transfer learning to our ResNet models in order to improve performance. Transfer learning involves taking a pre-trained model, removing its output layer, and adding additional layers to be trained for a more specific task (in this case anonymous traffic classification). The advantage of transfer learning is that it can make use of the information learned from the previous, more general training to enhance performance on the new task [37]. Our ResNet model was implemented using the TensorFlow Keras library and pre-trained on ImageNet.

Our goal is to compare the top performing shallow model trained on tabular data (XGBoost) with ResNet trained on T2I data. We also provide empirical evidence on whether T2I techniques are a viable approach to multi-class classification of various application types in anonymous traffic detection and categorization problems.

V. TABULAR-TO-IMAGE ARCHITECTURES

CNNs are effective at analyzing data with spatial differences between features. This makes them ideal for application on image and audio datasets where the important information about the data is based on the order of the features [12]. In these cases, the data is homogeneous which allows for CNNs to distinguish spatial differences. However, when it comes to heterogeneous data, such as tabular data, CNNs can not be directly applied. This limitation inspires the process of transforming tabular data into images to apply CNNs. Applying CNNs to the transformed data can result in superior prediction performance compared to other shallow models trained directly on tabular data. The potential for performance improvement motivates research on the most effective method of converting tabular data to images by evaluating new and pre-existing T2I algorithms. The development of DeepInsight pioneered this transformation of data to images, followed by more effective algorithms like SuperTML, TAC, and IGTD that improve the process.

Sections V-A through V-D explain the T2I algorithms we use in our experiments.

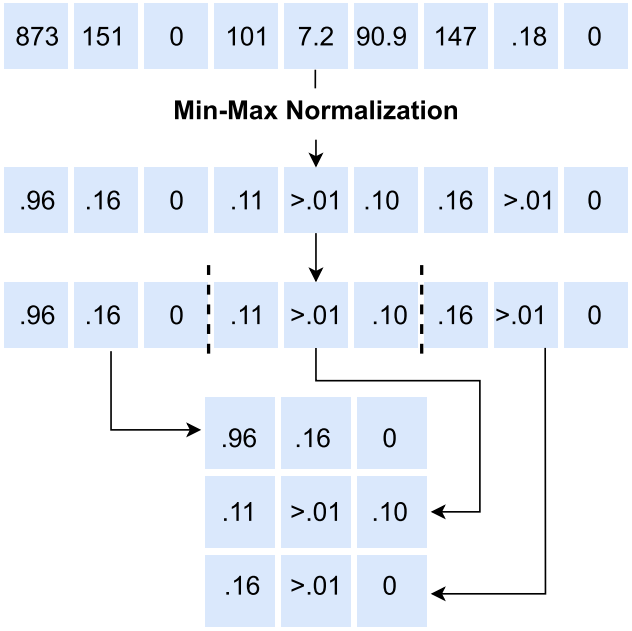

FIGURE 1. Feature wrapping visualization.

A. FEATURE WRAPPING

Ko et al. [38] introduced a method of converting raw network traffic data into images for classification by wrapping the binary data of a traffic flow into square images. A similar technique was then adapted to transform tabular data into images as a vector of feature wrapping [17], [24]. The vector of feature wrapping technique takes a one-dimensional tabular data sample and normalizes its values before creating a square 2D image. We employ min-max normalization (1) for the feature wrapping technique used in our experiments. Since categorical features were discarded in our dataset, we did not employ the one-hot encoding technique. After normalization, each sample is then split into equal-length sub-vectors which are stacked on top of each other to form a square image. If a sample doesn't have enough values to form a square image, it can be padded with additional zeros [17], [24]. This feature-wrapping method is illustrated in Figure 1.

$$x_{normalized} = \frac{x - x_{min}}{x_{max} - x_{min}} \tag{1}$$

B. DeepInsight

To better identify variations in genomic and biological data, Sharma et al. [39] introduced a T2I scheme called DeepInsight. The algorithm clusters similar or related features into a two-dimensional feature space using different dimensionality reduction techniques such as t-SNE or PCA. Then the convex hull algorithm is used to find the smallest bounding rectangle of the feature data points. Once the rectangle has been calculated, it is rotated to be horizontal or vertical and then converted to a pixelated image. They tested DeepInsight by generating image data from four different datasets (text, gene data, and 2 artificial datasets) and trained CNNs on these

datasets. The performance of the CNNs was then compared to traditional machine learning methods. The DeepInsight system achieved the highest accuracy metrics across each dataset, with an accuracy of 95% on average.

C. IGTD

Zhu et al. [12] introduced Image Generator for Tabular Data (IGTD) to improve existing image generation techniques. The optimization algorithm converts tabular data to images by assigning each feature to a pixel. The assignment is determined by ranking the pairwise distances between features and the pairwise distances between the assigned pixels. The algorithm then minimizes the difference between these two measurements. Pairwise distances are calculated through a distance measure such as the Euclidean distance or the Pearson Correlation Coefficient. This assigns similar features to pixels close to one another and dissimilar features to ones farther apart. The efficiency of this method stems from a greedy iterative process of swapping the pixel assignments of features to best reduce the distance between them.

Unlike DeepInsight, IGTD produces dense image representations where each pixel represents a unique feature. This results in smaller images that take less time when training CNNs. IGTD also does not require domain knowledge and has excellent feature preservation as closer features are more similar. The size and shape of the image generated can be adjusted, which makes it more applicable to a variety of domains. They compared IGTD to CNNs trained with DeepInsight and REFINED images on datasets for gene expression profiles of cancer cell lines and molecular descriptors of drugs. CNNs trained on IGTD provided similar or better prediction performance when compared to the other T2I methods and models trained on the original tabular data. Despite its origination in a different domain, we wanted to examine IGTD's applicability in the network traffic domain.

D. BINARY IMAGE ENCODING

Inspired by the one-hot-encoding technique, we introduce Binary Image Encoding (BIE), a novel T2I scheme. The one-hot encoding method was originally introduced by Wang et al. [22] and involved converting binary network flow data into a 2D image by applying one-hot encoding on each byte of the sample. The reasoning for this process is that raw network data often does not have an ordering and its values are better represented as categorical features. This is because the information in raw network data are features like protocol types or flags rather than meaningful numerical values [17].

Instead of treating features as unordered categorical values, BIE makes use of the structure of binary representations of floating point values. Fig. 2 outlines a binary encoded double as well as the conversion to a decimal representation. Double precision binary numbers consist of a sign bit, the exponent which dictates the magnitude of the number, and the mantissa which represents the significant digits of the value.

The technique converts each numerical sample value into a double precision binary string as discussed above.

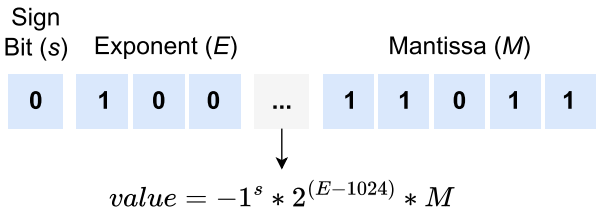

FIGURE 2. Binary encoded double precision floating point value.

The binary values are then stacked on top of each other to create a two-dimensional matrix to be interpreted as an image where zero values become black pixels and one values become white pixels. This process is illustrated in Fig. 3, which depicts numerical feature values being converted into 64-bit binary strings and then being situated on top of each other to form the full BIE image. The pseudocode for converting an input sample into an image is provided in Fig. 4.

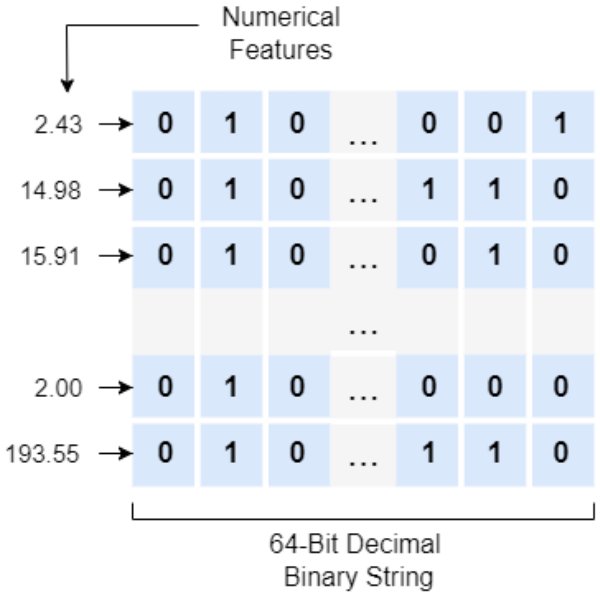

FIGURE 3. Binary image encoding process.

We believe that representing feature values as vectors of binary encoded floating point numbers could have many benefits for network traffic classification. First, this method does not rely on normalization as many of the previously discussed T2I techniques do. This is advantageous because normalization reduces the range of potential feature values and can also be heavily affected by outliers. Secondly, a binary decimal string isolates the magnitude of a value (exponent) from the precise value of each digit (mantissa). When differentiating network traffic flows, one important factor to consider is the magnitude of the packets exchanged during the flow. For example, video streaming applications will have thousands of packets exchanged in a short time, whereas an email may typically have a lot fewer. To this end, isolating the magnitude

Algorithm 1 Convert Samples To Binary Image

```
function CONVERT_SAMPLE_TO_BINARY(sample, feature_types)
    Initialize an empty list sample_out
    for each feature in sample do
        Convert the feature to the 64-bit binary floating point representation
        Set the value of each bit to be equal to 255 if it is 1 and
        zero if it is a 0
        Append the binary representation of the feature to sample_out
    end for
    Create a new image with dimensions equal to the size of sample_out
    Set the pixel values of the image using the values in sample_out
    Save the image to the specified directory with the specified label
```

FIGURE 4. BIE pseudocode.

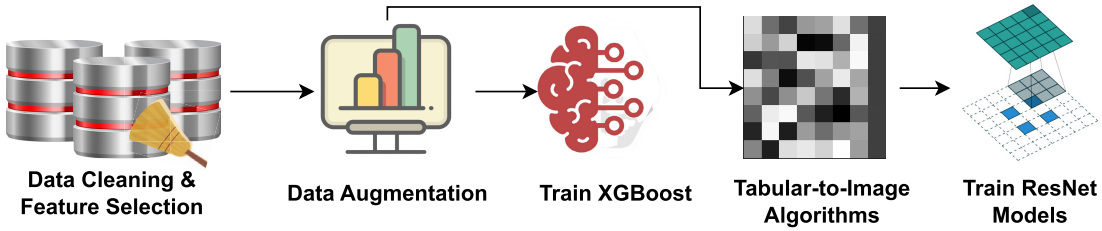

FIGURE 5. Binary image encoding process.

of the value in the image representation may make a classification based on packet quantity in a flow easier. Finally, the method expands the information of each value by partitioning it into meaningful parts as opposed to IGTD, where each pixel corresponds to a given feature, limiting image's information by the number of available features.

VI. RESEARCH METHODOLOGY

Fig. 5 outlines our research methodology. Subsequent sections present the experimental outline, processes for collecting metrics, tabular-to-image algorithmic conversion processes, and model training.

A. EXPERIMENTAL OUTLINE

First, we establish baseline results by training the shallow learning XGBoost classifier on CMU dataset. The XGBoost classifier was trained to distinguish among eight application types and its performance metrics were recorded.

Next, we generate five (5) new image datasets using each of the T2I algorithms which are then used to train five ResNet-50 models. After collecting performance metrics on these models, we compared their performance to XGBoost with an eye toward providing empirical evidence of the importance of T2I techniques.

B. PERFORMANCE METRICS

We report accuracy, F1-score, Area Under the Curve (AUC), mean squared error (MSE), mean absolute error (MAE), and tabular-to-image encoding time to evaluate model performance. True positives (TP), true negatives (TN), false positives (FP), and false negatives (FN) are used to calculate the aforementioned metrics [40].

F1-score is the harmonic mean of precision and recall where precision is the proportion of correctly classified positive classifications and recall is the percent of TPs that a model predicted accurately. ROC curves are a visual representation of the TP rate in relation to the FP rate. The area under the curve (AUC) is calculated by finding the total area under a ROC curve. MAE is simply the average of the absolute differences between the model's predicted values and actual values, whereas MSE is the squared differences between the predicted and actual values. MAE reflects the overall error giving equal consideration to all data samples. In contrast, MSE is more affected by outliers so a larger MSE can indicate that there are large outliers potentially from class confusion. Our loss function is categorical cross entropy, which is a standard loss function for measuring the general fit for multi-class models [41]. F1-score and AUC are less susceptible to imbalanced data and can help determine whether a model is overfitting to training data. We primarily

use F1-score to compare model performance. Tabular-to-image encoding time (in seconds) is also reported as it can be an important metric to consider when performing real-time anonymous traffic detection and continuous model learning.

$$Accuracy = \frac{(TP + TN)}{(TP + FP + TN + FN)} \tag{2}$$

$$Precision = \frac{TP}{TP + FP} * 100 \tag{3}$$

$$Recall = \frac{TP}{TP + FN} * 100 \tag{4}$$

$$F1 - score = \frac{TP}{TP + \frac{1}{2}(FP + FN)} * 100 \tag{5}$$

C. TABULAR TO IMAGE ENCODINGS

Our preliminary experimentation found that image size had minimal effect on model performance; however, conversion time reduces considerably when generating smaller images. All T2I algorithms were given the same input dataset and generated 240,000 corresponding images. The image samples shown in Figures 6, 7, 8, 9, and 10 were generated from the same samples for each of the T2I techniques.

Our novel BIE scheme generates images (Fig. 6) containing 64 rows and 64 columns. Each row represents a feature and each column is the corresponding 64-bit binary representation for that feature where ones (1) are represented as light pixels and zeros (0) as dark pixels.

Images generated by IGTD (Fig. 7) consist of 8 rows and 8 columns for a total of 64 squares. Each square represents a feature where a darker value indicates a higher feature value and vice-versa.

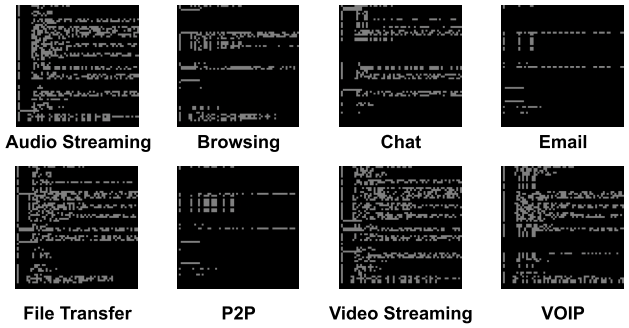

FIGURE 6. Sample images generated using binary encoding.

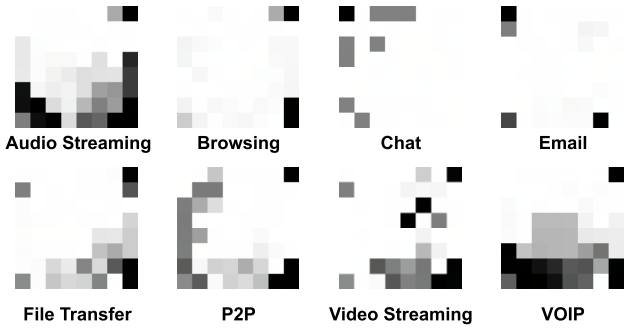

FIGURE 7. Sample images generated using IGTD.

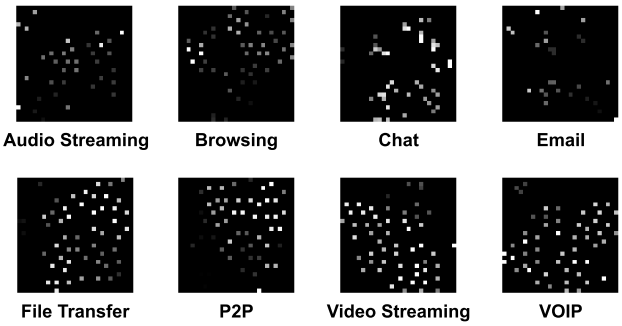

FIGURE 8. Sample images generated using DeepInsight.

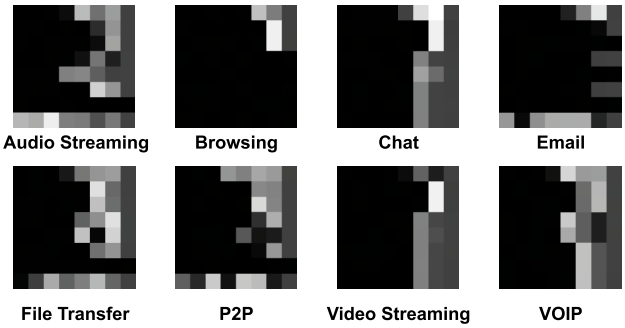

FIGURE 9. Sample images generated using feature wrapping.

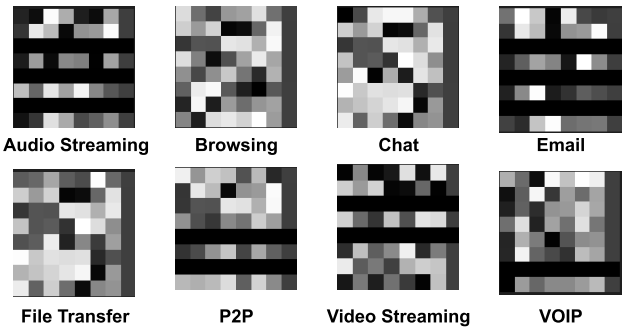

FIGURE 10. Sample images generated using feature wrapping normalized.

Unlike the other algorithms, DeepInsight does not create a grid-based image. The image (Fig. 8) is constructed as a bounding box that encompasses all the features using the convex hull algorithm. Dark pixels indicate no value and the lighter the color, the higher the feature value.

Similar to IGTD, each box in the Feature Wrapping images (Fig. 9) contains the value of a single feature. The Feature Wrapping Normalized follows the same process, but the features are normalized before encoding (Fig. 10).

D. ResNet MODEL TRAINING

The system used to train the models ran on Ubuntu 20.04.3 LTS with an Intel i7-7700k CPU, GTX 1080 GPU,

TABLE 1. Performance metric comparison.

Method	T2I Time	Loss	Acc	F1	Prec	Recall	AUC	MAE	MSE
BIE	211.4	0.0666	0.975	0.9749	0.9797	0.9703	0.9996	0.0095	0.0045
IGTD	33.7	0.1297	0.9606	0.9608	0.9616	0.9599	0.9976	0.0164	0.0085
DeepInsight	131.3	.2119	.9216	.9225	.9465	.9000	.9968	.0273	.0134
Feature Wrapping Normalized	22.5	0.2717	0.9011	0.9039	0.9411	0.8700	0.9945	0.0348	0.01696
Feature Wrapping	24.9	0.4475	0.853	0.8573	0.8924	0.8254	0.9837	0.0462	0.0259
XGBoost	-	-	.9508	.9509	.9519	.9508	.9985	0.0151	0.0057

and 16 Gigabytes of RAM. All ResNet models were trained on the GPU while T2I algorithms were conducted on the CPU. Each model took on average 7,950 seconds (2 hours, 12 minutes and 30 seconds) to train for 100 epochs with 192,000 samples in the training dataset. Since the datasets all had the same number of samples and identical image dimensions, the models took approximately the same time train only varying by a few seconds.

We employed the ResNet-50 model pre-trained on the ImageNet database as our deep learning model. The output of the model was flattened before being passed to a custom fully connected network. The fully connected network consisted of two dense layers with respective output sizes of 512 and 8 units. The first dense layer incorporated a ReLU activation function while the second layer utilized the softmax function.

Through experimentation, we found the stochastic gradient descent optimizer to outperform other optimizers when implemented with a learning rate of 0.01 and momentum of 0.7. The learning rate is monitored with the ReduceLROnPlateau callback which will reduce the learning rate by a factor of 0.05 with a minimum learning rate of 0.000002 if the loss fails to improve. The model was set to train for 100 epochs with an early stopping callback to terminate training if the loss did not improve in five subsequent epochs. After each epoch, the model weights were saved and a real-time graph was updated using Tensorboard. 5-fold cross-validation was used to train the model to ensure that the models are generalizable.

VII. RESULTS AND DISCUSSION

In this section, we present the results of the five ResNet classifiers trained on the CMU-I image datasets compared to the XGBoost classifier trained on tabular CMU dataset. Then, we discuss the tradeoffs of the structured data and T2I approaches providing insights into the viability of each method in a potential real-world deployment.

A. CLASSIFIER RESULTS

Table 1 compares the performance metrics among the five ResNet-50 classifiers trained on each T2I method and XGBoost in the classification of eight application types. Values highlighted in green are the top metric across all classifiers whereas blue-highlighted values are metrics that exceed those of XGBoost. It can be seen that the proposed Binary Image Encoding is the top-performing method across all measured metrics (excluding image generation time), improving over XGBoost's F1-score by 2.4 percentage points. IGTD was the only other method that saw higher metrics over the baseline, improving upon F1-Score by approximately 1 percentage point. Figure 11 provides better visual comparisons among the evaluated methods.

Figure 12 depicts the differences in image generation time among the T2I methods. Notably, Binary Encoding took significantly longer (210 seconds) to convert the 240,000 tabular samples to their corresponding image representation which amounts to 0.9 ms per sample on average. This could be attributed to the fact that this technique is novel to this work and there may be room for further optimization. DeepInsight also took considerably longer, potentially due to its reliance on the computationally expensive Convex Hull and t-SNE algorithms. The other T2I methods had relatively shorter generation times, taking only 20-30 seconds to produce all 240,000 samples (0.1 ms per sample).

B. OCCLUSION SENSITIVITY ANALYSIS

To better understand our BIE ResNet model's predictions, we employed occlusion sensitivity analysis. Occlusion sensitivity analysis is a popular way to visualize CNNs by blocking out a portion of a predicted image and seeing how the model's confidence is affected [42], [43]. More important areas for the classification of that image should yield lower predicted confidence when covered. This process allows us to create occlusion sensitivity maps which visualize the parts of

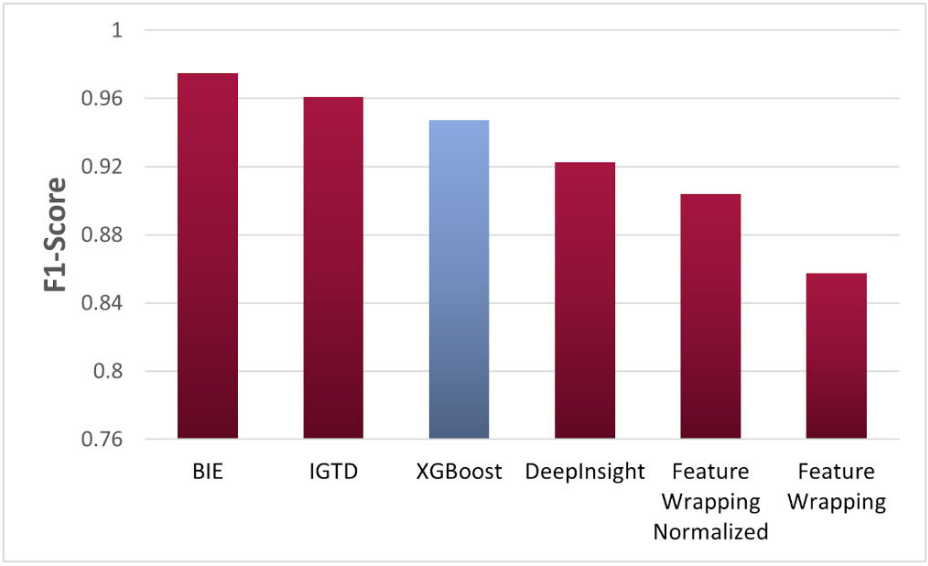

FIGURE 11. F1-score comparison for ResNet-50 models trained on T2I methods vs. XGBoost.

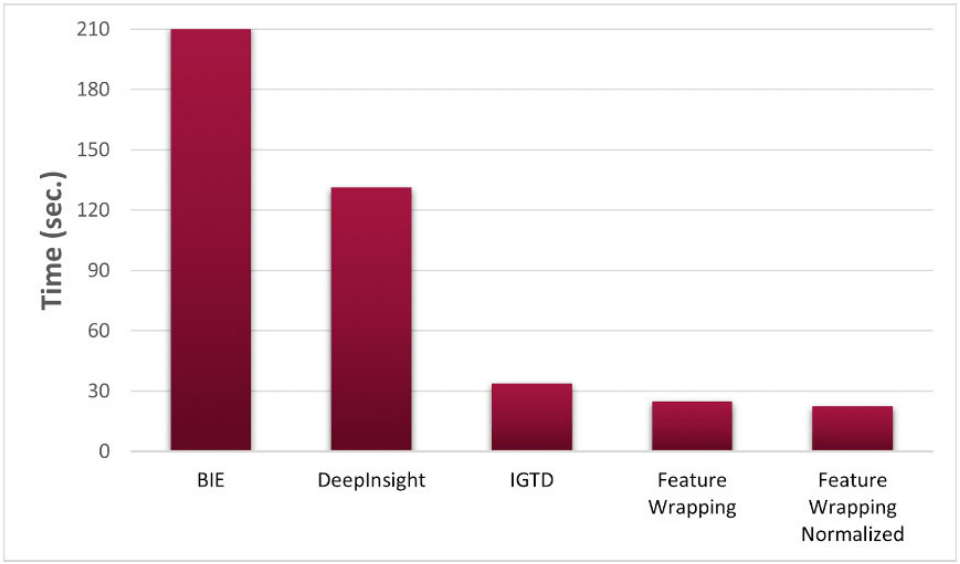

FIGURE 12. Image generation times for T2I methods (240,000 samples).

BIE images that are important for classification for different classes.

We generated our occlusion sensitivity maps by replacing part of the target image with a gray patch, classifying the image, and then mapping the model's confidence value to that region. We repeated this process for 1000 images for each class and averaged the values to find the most salient regions. Figure 13 shows the average occlusion sensitivity maps for all classes.

Observing these figures we can see that critical regions are distinct among the classes. For instance, we observe that audio streaming is affected most by the group of features in the top half of the image, but only the part of the feature comprising the exponent and most significant digits of the mantissa. Chat also appears to be most affected by occluding information on the left side of the image, but does not see a significant drop in confidence when the right side is omitted. This could support the idea that for some traffic classes, the magnitude of certain features is the most relevant piece of information for classification. However, the critical regions of other classes like browsing and Email are more broad and scattered across the entire image.

Figure 14 shows individual traffic samples overlaid with their occlusion map. These images can provide a more

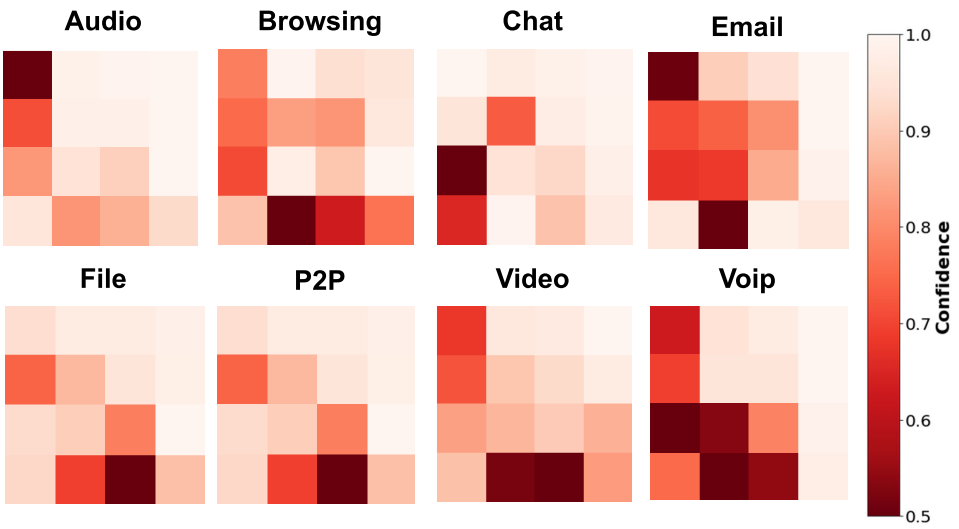

FIGURE 13. Averaged occlusion maps for all classes (1,000 samples).

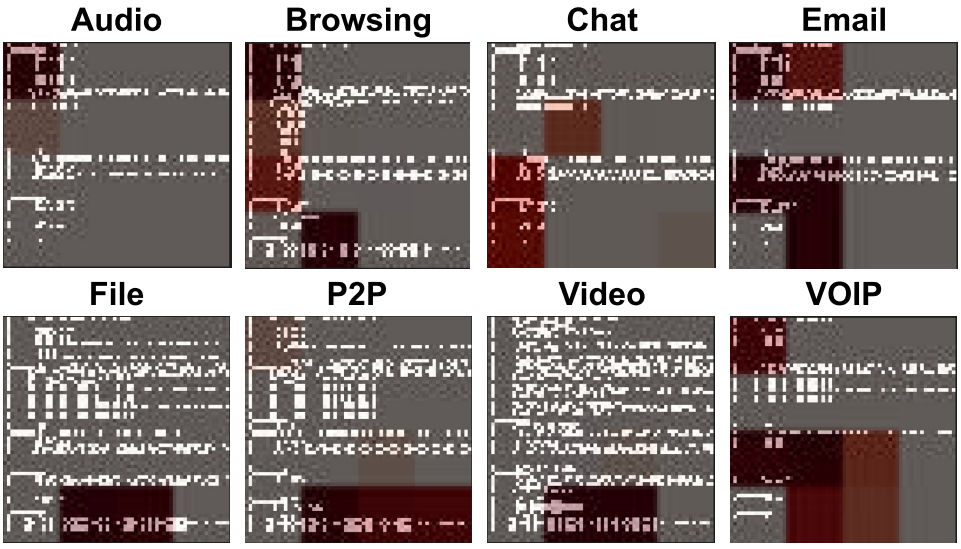

FIGURE 14. Traffic samples overlaid with their corresponding occlusion map.

detailed look at the critical regions. Once again we see that audio, browsing, and chat have critical regions on the leftmost side of the image. The lower middle part of the image seems to be the most salient region for the File, P2P, and Video samples.

It should be noted that occlusion sensitivity analysis is highly dependent on patch size, so salient regions may be different when analysis is conducted with different patch sizes. Additionally, our averaged maps were conducted on a relatively small number of images, so they may not represent the entire distribution of the data. Drawing concrete inferences about how BIE images are classified is currently not possible, but these visualizations give a better idea of how the ResNet classifier differentiates different traffic types.

C. DISCUSSION

In this section we analyze the results of our experiments in context with prior works while discussing the viability of the T2I techniques.

Both IGTD and DeepInsight achieved competitive metrics proving their ability to generalize in disparate problem domains. These methods also outperformed feature wrapping, which is a more established T2I technique for network classification problems [17]. While it is hard to draw direct comparisons of metrics to other research (as their data, features, classification objective, or number of classes may differ), our models trained on IGTD and BIE data obtained >96% accuracy in classification of eight network application types which was greater than any of the reviewed literature using the same dataset and classification task [9], [13].

When considering the applicability of the evaluated T2I methods, there is a definite tradeoff between computation time and accuracy. A real-time detection system would require the additional overhead of transforming collected network traffic into images before being evaluated by the model. Depending on available computational resources and the amount of network traffic, the added latency of detection could make the system unsuitable. However, we have demonstrated that T2I methods can noticeably increase classification accuracy over shallow classifiers. Furthermore, IGTD offers an increase in accuracy while also keeping the image generation time comparatively lower.

Online learning, the process of continually updating a model based on new data, can be negatively impacted by the slow training time of deep learning models. Training each of our ResNet models took 2 hours (30 times longer than the same by XGBoost) on our hardware. If the proposed RestNet-50 models are deployed for anonymous traffic classification, online learning may not be viable depending on available computational resources.

Our experiments also showed that the choice of T2I method is important to the overall performance of the classification system as most of the T2I methods failed to improve upon or match the performance of XGBoost. IGTD and BIE also outperform previous similar works [9] and [13] which achieved accuracies of 86% and 92% on the same eight application types. BIE may have performed well for the reasons stated in section V-D. IGTD has similarities to feature wrapping in the sense that each feature corresponds to a pixel value; however, IGTD is unique in that it correlates features by importance which may have contributed to its superior performance.

Finally, with $>2\%$ improvement on base-line classifier (XGBoost) and $>1\%$ improvement on the state-of-art T2I technique (IGTD) especially in a multi-class classification problem of determining various application types in anonymous network traffic data, we argue that our novel BIE scheme is a viable T2I technique in this domain.

VIII. LIMITATIONS AND FUTURE WORK

Due to limitations in computational resources, minimal hyper-parameter tuning was performed. Future works may benefit from experimenting with additional hyper-parameter tuning on the ResNet models and T2I parameter tuning (such as the dimensionality reduction technique used in DeepInsight and the image generation size). Moreover, other pre-trained CNN classifiers (such as ResNet-N with variation in depth, N [35]) should be evaluated and compared to non-pre-trained CNN-based models in addition to other computer vision techniques such as transformers. Finally, more visualization methods and sensitivity analysis should be applied to BIE trained models to better understand feature-class relationships.

We trained models to classify eight application types from the Tor and VPN protocols, but there are many anonymous protocols unexplored in this work. For instance, SSL/TLS, SSH, and HTTPS may not be detected or falsely classified by the current models as they were not provided in the training data. Additionally, deep learning models benefit from larger datasets, so model performance may have been impacted by the relatively small dataset. Though synthetic data was generated using SMOTE to alleviate this concern, future work could look into gathering more anonymous network traffic to address this limitation. Nonetheless, two of the CNN models still outperformed the shallow counterpart, possibly mitigating the concern.

The CIC-DN dataset did not provide the flow interval used to generate the tabular dataset from the raw pcap file. A variety of flow intervals should be tested to find the interval that optimizes model performance.

The CMU-I dataset may be used to train/optimize additional classifiers or be used as a baseline for other T2I techniques not considered in this work. Furthermore, the experimental workflow utilized to generate the CMU-I dataset should be applied to other domains to determine the general applicability of the approach. Since BIE is a novel technique, future work should test this method on other datasets and classifiers.

IX. CONCLUSION

This work explored the viability of five T2I methods (IGTD, DeepInsight, vector-of-feature wrapping (normalized and non-normalized), and the novel BIE) and their efficacy in classification of eight anonymous network application types. These techniques were used to generate five image traffic datasets (CMU-I) for training ResNet-50. To establish baseline results for comparison, the XGBoost classifier was trained on the balanced tabular dataset (CMU). From these experiments, we found that IGTD and BIE introduced in this paper improved classification metrics when compared to XGBoost, with a tradeoff of greater computation time while encoding structured samples into images. As a novel method, the results from BIE are promising; however, further evaluation is warranted for general applicability of the technique across other problem domains.

While many network traffic classification schemes based on CNNs have been proposed, only a smaller subset prioritizes data transformation, and even fewer apply image generation techniques to this field. This paper sought to demonstrate the potential of these techniques while also evaluating their real-world practicality and generalizability in the domain of anonymous network data classification. Furthermore, we have published our datasets for further scrutiny in the field.

ACKNOWLEDGMENT

Any opinions, findings and conclusions, or recommendations expressed in this paper are those of the authors and do not necessarily reflect the views of the funding sources. The authors would like to thank Jackson Warren and Karson Fye for their early help with getting this research project started.

REFERENCES

[1] A. Azab, M. Khasawneh, S. Alrabaee, K.-K.-R. Choo, and M. Sarsour, "Network traffic classification: Techniques, datasets, and challenges," *Digit. Commun. Netw.*, Sep. 2022, doi: 10.1016/j.dcan.2022.09.009.

[2] H. Mohajeri Moghaddam, B. Li, M. Derakhshani, and I. Goldberg, "SkypeMorph: Protocol obfuscation for tor bridges," in *Proc. ACM Conf. Comput. Commun. Secur.*, Oct. 2012, pp. 97–108, doi: 10.1145/2382196.2382210.

[3] A. Azab, M. Khasawneh, S. Alrabaee, K.-K. R. Choo, and M. Sarsour, "Network traffic classification: Techniques, datasets, and challenges," in *Digital Communications and Networks*. Elsevier, Sep. 2022, doi: 10.1016/j.dcan.2022.09.009.

[4] S. AlDaajeh, H. Saleous, S. Alrabaee, E. Barka, F. Breitinger, and K.-K. R. Choo, "The role of national cybersecurity strategies on the improvement of cybersecurity education," *Comput. Secur.*, vol. 119, Aug. 2022, Art. no. 102754, doi: 10.1016/j.cose.2022.102754.

[5] S. Alrabaee, M. Al-Kfairy, and E. Barka, "Efforts and suggestions for improving cybersecurity education," in *Proc. IEEE Global Eng. Educ. Conf. (EDUCON)*, Mar. 2022, pp. 1161–1168, doi: 10.1109/EDUCON52537.2022.9766653.

[6] M. Zhang, W. Sun, J. Tian, X. Zheng, and S. Guan, "An internet traffic classification method based on echo state network and improved salp swarm algorithm," *PeerJ Comput. Sci.*, vol. 8, p. e860, Feb. 2022, doi: 10.7717/peerj-cs.860.

[7] R. M. AlZoman and M. J. F. Alenazi, "A comparative study of traffic classification techniques for smart city networks," *Sensors*, vol. 21, no. 14, p. 4677, Jul. 2021, doi: 10.3390/s21144677.

[8] I. Sharafaldin, A. H. Lashkari, S. Hakak, and A. A. Ghorbani, "Developing realistic distributed denial of service (DDoS) attack dataset and taxonomy," in *Proc. Int. Carnahan Conf. Secur. Technol. (ICCST)*, Oct. 2019, pp. 1–8, doi: 10.1109/CCST.2019.8888419.

[9] A. H. Lashkari, G. Kaur, and A. Rahali, "DIDarknet: A contemporary approach to detect and characterize the darknet traffic using deep image learning," in *Proc. 10th Int. Conf. Commun. Netw. Secur.*, Nov. 2020, pp. 1–13, doi: 10.1145/3442520.3442521.

[10] J. Halladay, D. Cullen, N. Briner, J. Warren, K. Fye, R. Basnet, J. Bergen, and T. Doleck, "Detection and characterization of DDoS attacks using time-based features," *IEEE Access*, vol. 10, pp. 49794–49807, 2022, doi: 10.1109/ACCESS.2022.3173319.

[11] B. Sun, L. Yang, W. Zhang, M. Lin, P. Dong, C. Young, and J. Dong, "SuperTML: Two-dimensional word embedding for the precognition on structured tabular data," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jun. 2019, pp. 2973–2981, doi: 10.1109/CVPRW.2019.00360.

[12] Y. Zhu, T. Brettin, F. Xia, A. Partin, M. Shukla, H. Yoo, Y. A. Evrard, J. H. Doroshow, and R. L. Stevens, "Converting tabular data into images for deep learning with convolutional neural networks," *Sci. Rep.*, vol. 11, no. 1, p. 11325, May 2021, doi: 10.1038/s41598-021-90923-y.

[13] J. Lan, X. Liu, B. Li, Y. Li, and T. Geng, "DarknetSec: A novel self-attentive deep learning method for darknet traffic classification and application identification," *Comput. Secur.*, vol. 116, May 2022, Art. no. 102663, doi: 10.1016/j.cose.2022.102663.

[14] Y. He and W. Li, "A novel lightweight anonymous proxy traffic detection method based on spatio-temporal features," *Sensors*, vol. 22, no. 11, p. 4216, Jun. 2022, doi: 10.3390/s22114216.

[15] A. Al-Omari, A. Allhusen, A. Wahbeh, M. Al-Ramahi, and I. Alsmadi, "Dark web analytics: A comparative study of feature selection and prediction algorithms," in *Proc. Int. Conf. Intell. Data Sci. Technol. Appl.(IDSTA)*, San Antonio, TX, USA, 2022, pp. 170–175, doi: 10.1109/IDSTA55301.2022.9923042.

[16] N. Gupta, V. Jindal, and P. Bedi, "Encrypted traffic classification using extreme gradient boosting algorithm," in *Proc. Int. Conf. Innov. Comput. Commun.* (Advances in Intelligent Systems and Computing). Singapore: Springer, Aug. 2021, pp. 225–232, doi: 10.1007/978-981-16-3071-2_20.

[17] J. Krupski, W. Graniszewski, and M. Iwanowski, "Data transformation schemes for CNN-based network traffic analysis: A survey," *Electronics*, vol. 10, no. 16, p. 2042, Aug. 2021, doi: 10.3390/electronics10162042.

[18] L. Buturović and D. Miljkovic, "A novel method for classification of tabular data using convolutional neural networks," Cold Spring Harbor Lab., Cold Spring Harbor, NY, USA, Tech. Rep., May 2020, doi: 10.1101/2020.05.02.074203.

[19] S. Ma and Z. Zhang, "OmicsMapNet: Transforming omics data to take advantage of deep convolutional neural network for discovery," 2018, *arXiv:1804.05283*.

[20] L. Mohammadpour, T. C. Ling, C. S. Liew, and A. Aryanfar, "A mean convolutional layer for intrusion detection system," *Secur. Commun. Netw.*, vol. 2020, pp. 1–16, Oct. 2020, doi: 10.1155/2020/8891185.

[21] L. Mohammadpour, T. C. Ling, C. S. Liew, and C. Y. Chong, "A convolutional neural network for network intrusion detection system," in *Proc. APAN*, 2018, pp. 1–6. [Online]. Available: https://www.semanticscholar.org/paper/A-Convolutional-Neural-Network-for-Network-System-Mohammadpour-Ling/1633208c9021851a50723aafac77d4874fd842c8

[22] X. Wang, S. Yin, H. Li, J. Wang, and L. Teng, "A network intrusion detection method based on deep multi-scale convolutional neural network," *Int. J. Wireless Inf. Netw.*, vol. 27, no. 4, pp. 503–517, Oct. 2020, doi: 10.1007/s10776-020-00495-3.

[23] W.-F. Zheng, "Intrusion detection based on convolutional neural network," in *Proc. Int. Conf. Comput. Eng. Appl. (ICCEA)*, Mar. 2020, pp. 273–277, doi: 10.1109/ICCEA50009.2020.00066.

[24] Z. Li, Z. Qin, K. Huang, X. Yang, and S. Ye, "Intrusion detection using convolutional neural networks for representation learning," in *Neural Information Processing*. New York, NY, USA: Springer, 2017, pp. 858–866. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-70139-4_87

[25] A. H. Lashkari, G. D. Gil, M. S. I. Mamun, and A. A. Ghorbani, "Characterization of Tor traffic using time based features," in *Proc. 3rd Int. Conf. Inf. Syst. Secur. Privacy*. Setúbal, Portugal: SciTePress, 2017, doi: 10.5220/0006105602530262.

[26] G. Draper-Gil, A. H. Lashkari, M. S. I. Mamun, and A. A. Ghorbani, "Characterization of encrypted and VPN traffic using time-related features," in *Proc. 2nd Int. Conf. Inf. Syst. Secur. Privacy*. Setúbal, Portugal: SciTePress, 2016, pp. 407–414, doi: 10.5220/0005740704070414.

[27] A. H. Lashkari. (2018). *CICFlowmeter-V4.0 (Formerly Known as ISCXFlowMeter) is a Network Traffic Bi-Flow Generator and Analyser for Anomaly Detection*. [Online]. Available: http://rgdoi.net/10.13140/RG.2.2.13827.20003 and https://github.com/ISCX/CICFlowMeter

[28] D. C. Cullen, J. H. Halladay, N. B. Briner, and R. B. Basnet, "CMU-SynTraffic-2022," *IEEE DataPort*, 2022. [Online]. Available: https://ieee-dataport.org/documents/cmu-syntraffic-2022, doi: 10.21227/WC3Q-JZ97.

[29] D. Cullen, J. Halladay, N. Briner, R. Basnet, J. Bergen, and T. Doleck, "Evaluation of synthetic data generation techniques in the domain of anonymous traffic classification," *IEEE Access*, vol. 10, pp. 129612–129625, 2022, doi: 10.1109/ACCESS.2022.3228507.

[30] S. E. Gómez, L. Hernández-Callejo, B. C. Martínez, and A. J. Sánchez-Esguevillas, "Exploratory study on class imbalance and solutions for network traffic classification," *Neurocomputing*, vol. 343, pp. 100–119, May 2019, doi: 10.1016/j.neucom.2018.07.091.

[31] N. Briner, D. Cullen, R. Basnet, and J. Halladay, "CMU-SynTraffic-2023-ImageDataset," Mendeley, Feb. 2023. [Online]. Available: https://data.mendeley.com/datasets/t5p2v7h3vg/1, doi: 10.17632/T5P2V7H3VG.1.

[32] R. Mitchell, A. Adinets, T. Rao, and E. Frank, "XGBoost: Scalable GPU accelerated learning," 2018, *arXiv:1806.11248*.

[33] (Jun. 2023). *XGBoost Documentation—XGBoost 1.7.6 Documentation*. [Online]. Available: https://xgboost.readthedocs.io/en/stable/index.html

[34] M. Suginoo, "XGBoost: Its genealogy, its architectural features, and its innovation," Medium, Tech. Rep., Oct. 2022. [Online]. Available: https://towardsdatascience.com/xgboost-its-genealogy-its-architectural-features-and-its-innovation-bf32b15b45d2

[35] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778, doi: 10.1109/CVPR.2016.90.

[36] K. He, X. Zhang, S. Ren, and J. Sun, "Identity mappings in deep residual networks," in *Computer Vision—ECCV 2016*. New York, NY, USA: Springer, 2016, pp. 630–645, doi: 10.1007/978-3-319-46493-0_38.

[37] F. Zhuang, Z. Qi, K. Duan, D. Xi, Y. Zhu, H. Zhu, H. Xiong, and Q. He, "A comprehensive survey on transfer learning," *Proc. IEEE*, vol. 109, no. 1, pp. 43–76, Jan. 2021, doi: 10.1109/JPROC.2020.3004555.

[38] T. Ko, S. M. Raza, D. T. Binh, M. Kim, and H. Choo, "Network prediction with traffic gradient classification using convolutional neural networks," in *Proc. 14th Int. Conf. Ubiquitous Inf. Manage. Commun. (IMCOM)*, Jan. 2020, pp. 1–4, doi: 10.1109/IMCOM48794.2020.9001712.

[39] A. Sharma, E. Vans, D. Shigemizu, K. A. Boroevich, and T. Tsunoda, "DeepInsight: A methodology to transform a non-image data to an image for convolution neural network architecture," *Sci. Rep.*, vol. 9, no. 1, Aug. 2019, Art. no. 11399, doi: 10.1038/s41598-019-47765-6.
[40] M. Hossin and M. N. Sulaiman, "A review on evaluation metrics for data classification evaluations," *Int. J. Data Mining Knowl. Manage. Process*, vol. 5, no. 2, pp. 1–11, Mar. 2015, doi: 10.5121/ijdkp.2015.5201.
[41] L. Alzubaidi, J. Zhang, A. J. Humaidi, A. Al-Dujaili, Y. Duan, O. Al-Shamma, J. Santamaría, M. A. Fadhel, M. Al-Amidie, and L. Farhan, "Review of deep learning: Concepts, CNN architectures, challenges, applications, future directions," *J. Big Data*, vol. 8, no. 1, Mar. 2021, Art. no. 53, doi: 10.1186/s40537-021-00444-8.
[42] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *Computer Vision—ECCV 2014*. New York, NY, USA: Springer, 2014, pp. 818–833, doi: 10.1007/978-3-319-10590-1_53.
[43] E. Mohamed, K. Sirlantzis, and G. Howells, "A review of visualisation-as-explanation techniques for convolutional neural networks and their evaluation," *Displays*, vol. 73, Jul. 2022, Art. no. 102239, doi: 10.1016/j.displa.2022.102239.

NATHAN BRINER received the bachelor's degree in computer science with minor in mathematics from Colorado Mesa University. He was the Vice President of CMU's Computer Science Club. He hopes to continue research in machine and deep learning in his professional career.

DRAKE CULLEN is currently pursuing the bachelor's degree in computer science with minors in cybersecurity and mathematics with Colorado Mesa University. He is currently the Former President of the Cybersecurity Club, CMU. He is the President of Upsilon Pi Epsilon (the International Honor Society for Computing and Informatics) and the Treasurer of Kappa Mu Epsilon (the National Mathematics Honor Society), CMU.

JAMES HALLADAY is currently pursuing the bachelor's degree in math and computer science with Colorado Mesa University. His research interests include machine learning and topological data analysis.

DARRIN MILLER is currently pursuing the bachelor's degree in computer science with minor in physics with Colorado Mesa University (CMU). He is a member of the CMU's Cybersecurity Research Team and the CMU's Robotics Research Team. His research interests include machine learning, robotics applications, and physics simulation.

RILEY PRIMEAU is currently pursuing the bachelor's degree in computer science with minor in mathematics with Colorado Mesa University. He is a New Machine Learning Researcher with Colorado Mesa University. He is enthusiastic about contributing to the academic community and gaining more knowledge in the field. He is particularly interested in improving machine learning techniques and applying them to personal data to provide insight into enhancing well-being.

ABRAHAM AVILA is currently pursuing the bachelor's degree in computer science with minor and certificates in mathematics, cybersecurity, and data science with Colorado Mesa University (CMU). He is also the President of the Cybersecurity Club and the Committee Chair of the Computer Science Club, CMU. He is interested in gaining more knowledge in the domain of both machine learning and cybersecurity research.

RAM BASNET received the B.S. degree in computer science from Colorado Mesa University (CMU), in 2004, and the M.S. and Ph.D. degrees in computer science from New Mexico Tech, in 2008 and 2012, respectively. He is currently an Associate Professor of computer science and cybersecurity with CMU. His research interests include information assurance, machine learning, and computer science pedagogy.

TENZIN DOLECK received the Ph.D. degree from McGill University, in 2017. He is currently the Canada Research Chair and an Assistant Professor with Simon Fraser University.