Literature Review: Deep Learning for Wireless Community Networks

Ryan Murphy
University of Cape Town
MRPRYA002@myuct.ac.za

ABSTRACT

Community networks are locally owned and operated communication infrastructures that are established and maintained by communities. They are independent of Internet Service Providers (ISPs) and are decentralized. Since they are independent of ISPs they generally have less resources available due to their limited funding and limited infrastructure. This creates the need for quality of service(QoS) assurances. To achieve QoS we use deep learning for network traffic classification. The literature review will talk about the use of network traffic classification and why deep learning has proven to be the most effective method of traffic classification by comparing it to different methods. The review will compare different deep learning models by their accuracy and effectiveness at classifying different data formats. Finally, the most compelling choice of model will be shown to be either the RNN or CNN and the strengths and weaknesses of these models will be compared.

Keywords

Deep Learning, Community Networks, Traffic Classification, Neural Networks, Stacked Autoencoders

1 INTRODUCTION

Network traffic classification is becoming increasingly prevalent and important as network technology advances. With emerging technologies like Internet of Things(IoT) network resource management could be a problem for current network technology [16] due to the high throughput of internet traffic. Furthermore, traffic classification can improve QoS. Real-time traffic classification can help network operators to respond to flows of different network applications quickly which could improve QoS. Furthermore, it can help the network gain insights to their network by understanding patterns and significant variables that classify data. Another reason for performing traffic classification is to detect anomalies and to detect intrusions in the network. This could prove very valuable to a community network as they may not have the money to recover from intrusion.

Deep learning is being used in a variety of different fields and has become increasingly popular. In this literature review we will discuss the different datasets used in traffic classification, the different ways to preprocess this data used and how deep learning and other methods are able to classify models. Different machine learning and deep learning techniques will be discussed and finally why deep learning is the method of choice for this project. Originally, network administrators would classify network traffic using port numbers but with the advent of port obfuscation[34] this is no longer possible. Manual packet inspection has also become decreasingly popular due to data now becoming encrypted [27]. Humans have simply become ineffective at traffic classification. Machine learning, specifically deep learning, is the solution to these problems and there has been lots of work done by different researchers to find the most effective and efficient way of using these algorithms [32].

The experiments will be done using a variety of deep learning models. These models include Multi-Layered Perceptrons(MLPs), Stacked Auto-Encoders (SAEs), Convolutional Neural Networks(CNNs) and Recurrent Neural Networks(RNNs). Furthermore, hybrid and ensemble models, which is when two or more models are used, will also be discussed. These models will learn spatial and temporal dependencies of the data in order to make accurate classifications. The models will be trained on training data and tested on unseen data. The models will be compared over a range of different CPU, memory and time constraints and a benchmark model and the accuracy-computational complexity trade-offs will be explored.

2 BACKGROUND

2.1 Community Networks

Community networks are community governed and operated network infrastructures for digital communication [13]. The users are the ones who run it and manage it with funding coming from the users themselves. The purpose of these networks is to provide internet connection and accessibility to places where ISPs or governments are yet to reach. They are extremely important in low resource communities as they provide low-cost internet access. Since the people using community networks are the ones who run them, naturally there is not as much expertise, funding or capital compared to an ISP run network. Due to the lack of resources, QoS is something that these networks absolutely require.

2.2 Motive for Traffic Classification

Network traffic is data being moved from one computer to another over a network. Traffic classification is the process of organizing traffic into separate classes based on features of the data. Classifying network has become a very popular technique due to its many uses and benefits. With emerging technology like Internet of Things (IoT) network classification is important because it allows traffic classification of heterogeneous devices and services to find out information such as required latency, traffic volume etc. Furthermore, traffic classification has been used for anomaly and intrusion detection [24].

Network traffic classification for community networks can be used to ensure QoS. real-time classification can allow network operators to respond to flows of different network classifications quickly[33]. Community networks can have a wide range of application services on their networks, but they suffer from limited resources. Classification can allow the network operator to allocate more resources to the flow that has been predicted by the model and allow for effective use of resources, leading to QoS assurances. Furthermore, it also allows insight to what resources are being used the most and where most of their limited resources are being used on.

The end goal of this project is to perform traffic classification on data using pcap files from a community network. We perform network traffic classification to find the most accurate and computationally inexpensive model that can be used for traffic classification.

2.3 Machine Learning and Deep Learning

Machine learning algorithms are algorithms that are able to learn patterns and adapt without explicit instructions. This is done by feeding these algorithms training data for the model to learn from by making changes to its parameters. The model is then tested on unseen data to see how well the model that was trained on the training data performed. This unseen data is called test data. Traditional machine learning algorithms include Decision Trees, Regression models, K-nearest neighbours, and others [18].

Deep learning on the other hand is a subset of machine learning. Deep learning uses artificial neural networks to learn from data. These neural networks are very powerful at learning complex patterns and extract hierarchical features in data making them very versatile and effective.[12]. Graphics processing units (GPUs) which are commonly used in computer graphics are now being used in machine learning due to their high degree of parallelism[19]. Deep learning models perform many calculations on each piece of data fed into it and these datasets can be very large. Therefore, they take lots of time to train. But with GPUs they are able to speed things up by exponentially due to their many cores. This is one of the main reasons deep learning has become so popular in recent time. Furthermore, with libraries like TensorFlow and PyTorch, deep learning has become a lot easier. TensorFlow and Pytorch are both deep learning libraries in Python that enable users to program and train deep learning models and deploy them to production[20, 21]. They eliminate the requirement for users to build these models from scratch.

2.4 Challenges and Constraints

One of the purposes of this project is to allow for online, real-time classification. This will allow the network administrator to allocate resources according to which network traffic is requiring the most resources. Therefore, one of the main challenges is to classify traffic accurately, but also in a way that is not too computationally expensive and that takes too much time. Dicks[6] showed that LSTM models were able to achieve accuracy levels greater than 90% and performed better than multilayered perceptrons. But this did not come without thousands of parameters which made it more computationally expensive.

3 RELATED WORK

Many approaches have been used for network traffic classification such as statistical techniques, machine learning techniques and deep learning techniques[1, 2, 5]. Furthermore, there have been other methods used to classify network traffic by using different features of network. Besides model selection, there are different ways to classify network traffic. In this review we will cover port-based, packet inspection and flow-based methods[15, 17, 29]. Although these different approaches lead to an issue when comparing different deep learning approaches. Most work either uses flow-based or packet inspection to classify data or a mix of both. This makes it difficult to compare models because some models are better at classifying using flows rather than packets or vice versa. Therefore, it is important to note the type of classification used when comparing models. Although it makes model comparison more difficult, these different methods of classification allow for models that are not suited to some types of data to still perform well. For example, a Long Short Term Memory network performs well on temporal data[28] and will likely perform well on flow-based data. A Multilayer Perceptron however, most likely will not perform as well on this data but could outperform other models on packet level data. This has allowed for heterogeneity in this line of research.

Furthermore, there is not one standard dataset that is used to conduct network traffic classification. This is why some previous methods have become outdated due to changing datasets, especially with the addition of data encryption[26]. A common dataset is the ISCXVPN dataset[7] which is a labelled dataset and has network traffic from HTTP, Chrome, Skype etc. Other datasets like USTC-TFC2016[31] is also a commonly used dataset. Like ISCXVPN it is labelled and contains traffic types like Facetime, Gmail, and Skype. These different datasets could be an issue due to the fact that some may have different amounts of data, some data may be more unbalanced than others and some may have different fields that are better predictors than other datasets. These imbalances amongst datasets also makes it difficult to compare models. Moreover, there are only a few datasets that used data from community networks and more specifically the dataset we will be using. An approach that follows standard and good machine learning practices should be followed for results to be replicated with the same or different data.

Another difficulty is that many papers on network classification were used to classify intrusions rather than actual network traffic classification[22, 31]. Although these papers are still of use because they follow the same principles that are followed for traffic classification and they follow the basic guidelines for machine learning. The problem is that the datasets these models were trained on were different and likely imbalanced which could lead to unreliable accuracy results. Pektas et al.[22] used the CICIDS2017 dataset and there were roughly 88000 normal packets and 3000 intrusive packets. Although datasets are different, they share the same predictors and can be used for flow or packed inspection, making them eligible to compare models with.

3.1 Port-Based Classification

In the early days of network traffic classification, port-based traffic classification was very effective. This was done by using the packet's port numbers to classify network traffic to their correspondence protocols[3]. However, these methods are now considered to be less accurate for applications with dynamic ports. Although, by using a technique called discretization alongside the use of port numbers and machine learning models. Lim et al.[15] achieved accuracy levels greater than 93%. Although, they did prove that they can still be used in traffic classification, this approach deviated from traditional port-based methods and used statistical methods as well. In general researchers are not often using port-based methods alone but are rather using a mix of port-based and other techniques. Overall, the port-based method alone is very quick but too simple prone to port obfuscation and deception attacks [34] . Although port numbers should not be disregarded completely as they are still very important when using flow-based data as serve as one of the identifiers for flows.

3.2 Deep Packet Inspection

Traditional Deep Packet Inspection (DPI) is the method of checking the content of packets as they traverse a designated checkpoint, promptly taking action based on the packet's contents and predefined rules established by an enterprise, internet service provider, or network administrator. This is done by humans or computers that do not perform machine learning algorithms on them. It is to be expected that DPI performs poorly on encrypted network traffic. Yet through the use of machine learning packet inspection can still be useful. Lotfollahi et al.[17] proposed a method called Deep Packet, that used the first 1480 bytes of the IP payload as well as the IP header as input to the model and achieved results of 98% accuracy on the ISCXVPN. An advantage of packet inspection is that it can be useful in real-time classification due to the fact that packets are independent of other packets and can be quickly classified because they do not first need to be aggregated into flows. Although deep packet was less commonly observed in the literature due to flows generally performing better and being more versatile.

3.3 Flow-Based Classification

Unlike DPI, when performing classification based on flows the payload is not used at all. Instead, data is organized into flows and classified based on these flows. Flows are aggregations of packets that share the same five tuples[3]. These are packets with the same source and destination IP addresses, the same source and destination port numbers and the same protocol. Source and destination port numbers and IP addresses can be reversed to represent bidirectional flows. These are flows that travel from a source to a destination and vice versa. On the other hand, unidirectional flows are flows that only go in one direction, source to destination. Wang et al.[30] concluded that classifying on bidirectional flows is more effective than unidirectional flows as they contain more interaction information and allows the model to learn more about the features. Flows benefit from the fact that temporal data can be accessed by calculating inter-arrival times. These can be of great use as there are many machine learning models that thrive on time series data. Pektas et al.[22] used flow-based methods using a two different models an accuracy of up to 99% using the CICIDS2017 dataset. They achieved this accuracy using a long short term memory network which is very well optimized for time series data. Although one of the disadvantages of flow-based classification is that a high amount of overhead is required to store these packets before aggregated into flows.

3.4 Dataset and Preprocessing

The dataset and preprocessing pipeline we will use is what Tooke, Weisz and Dicks used [6, 29, 32].The data we will be using is collected from the OceanView community network in Cape Town. These are PCAP files that were collected at the gateway of the network from February 2019 [10]. The data will be extracted into flows using a library called *pkt2flow*. Since the data from the dataset does not contain labels and we will be using supervised learning techniques, we need to label these flows. An open-source library called *nDPI* will be used to label these flows. The IP payload for each packet will be extracted using another library called *scapy*. The number of bytes in each payload will vary but will not be more than 1480 bytes. Payloads with less will be padded with zeros. The data will then be normalized, and these IP payloads will be fed to the model.

3.5 Statistical Techniques

One of the less commonly used methods was processing data into statistics. These includes statistics like means and standard deviations of different predictors and modelling probability distribution functions (PDFs). Crotti et al.[5] used statistical analysis also on data flows. They used information such as inter-arrival time packet size distribution, source and destination IPs and worked out and aggregating the PDFs and passing them through a Gaussian filter to obtain one vector of aggregated scores for each protocol. On unseen data an anomaly score was calculated and normalized and compared with the PDFs of each flow to make a classification. They achieved accuracy recall scores of 79%-99% across different protocols. Although these methods rely heavily on human engineered data and is very difficult to recreate not making it very generalizable.

3.6 Traditional Machine Learning Techniques

Traditional machine learning methods are methods that do not include neural networks. AlZoman et al.[2] used k-nearest neighbours and decision trees for traffic classification for smart city networks and achieved results greater than 96% accuracy for each model. However, their performance heavily depends on the human-engineered features, which limit their generalizability[27].

4 DEEP LEARNING APPROACHES

Deep learning can be thought of as a subset of machine learning that only uses neural networks [12].This section will mention Multilayer Perceptrons(MLPs),Recurrent Neural Networks(RNNs), Convolutional Neural Networks(CNNs), and Stacked Auto Encoders(SAEs)

There are many reasons for the rise in popularity of deep learning models over traditional machine learning models. Deep learning algorithms are able to extract important features automatically while traditional machine learning models require manual feature engineering[12]. Another reason is that deep learning algorithms are generally more accurate than machine learning models. Furthermore, frameworks such as TensorFlow and PyTorch make deep learning easier than ever before[20, 21].

Machine learning models are trained on training data and evaluated on test data which is unseen. This lets the developer know how well the model did by using metric such as F1 score, accuracy etc.

4.1 Multilayer Perceptron

Multilayer Perecptrons are the most well-known and frequently used type of neural network[23]. They are popular due to their simplicity and their effectiveness over a wide range prediction and classification tasks. MLPs are a class of feed-forward neural networks having multiple layers. These include the input layer, activation layers and the output layer. Input(s) are fed through the input layer and are fed through a node. In this node they are transformed by being multiplied by some weight term and a bias term is added. The resulting value is then put through an activation function which then transforms it once again. Activation functions change the value from the previous neurons by plugging them into a mathematical function and using the output of this function to pass through the next layer. Many activation functions exist such as Sigmoid, ReLU and many others. These are chosen by the programmer and are suitable for different tasks.[25] This process is repeated by feeding the result forward into another node or multiple nodes in the next layer. This is why they are called feed-forward neural networks. The final result is then put through a final activation function which then produces the final result which could be a prediction or a classification. For classification the activation function will produce a value between 0 and 1 to represent probability and based on that probability a classification is made. The result is then compared with the value observed in real life with those particular inputs. It uses a loss function to see how well the model fit the data. Loss functions include, cross entropy, RMSE and more. The loss is then used to update the weight and biases using an technique called backpropogation. Backpropagation, a crucial training technique for neural networks, uses gradient descent to iteratively update the network's weights and biases. It calculates the loss functions gradient with respect to the parameters of the network at the output layer and propagates it backward through the network, which allows for learning and performance improvement[14]. MLPs are the most basic form of deep learning techniques and serve as the foundation for other neural networks. Although very popular they generally perform worse than other models in network traffic classification. This is due to the fact that they are more general and come up short when placed against more specialised model that fits better to the data [32].

Using the deep packet inspection Khedkar et al.[14]managed to achieve 99% accuracy classifying malicious traffic and beat other machine learning models like support vector machines, AdaBoost and XGBoost by 10-20%. One of the most basic deep learning methods significantly outperformed other machine learning models.

Although, MLPs do have their shortcomings. MPLs do not consider temporal dependencies in the data and do not work as well as other neural networks like RNNs to classify time series data. In the context of flow traffic classification, inter-arrival time of packets is a feature to consider. This is supported by Dicks[6] paper that showed the maximum accuracy a MLP could achieve was 70.5%. Comparatively, Dicks used an long short term memory model(LSTM) which is a type of RNN achieved 91% on flow data from the OceanView dataset. Although it is worth noting that the MLP was a lot quicker and used a lot less computing power.

Figure 1 provides a high-level diagram of a simple MLP.

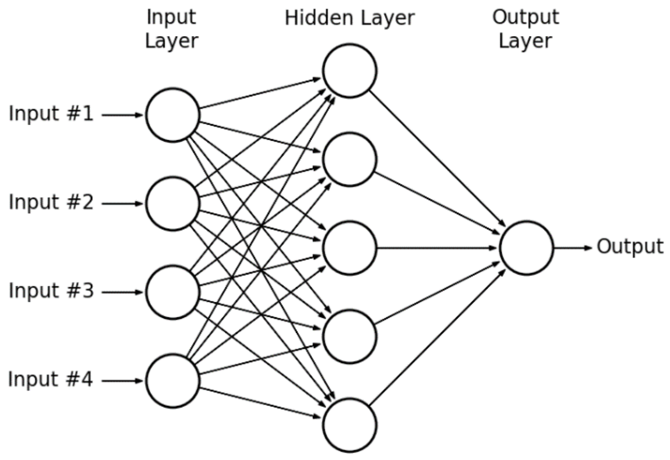

Figure 1: Multilayer Perceptron

4.2 Convolutional Neural Networks

Convolutional neural networks have evolved from the MLP. They use similar techniques for their feed-forward phase and use gradient descent algorithms to update their weight and biases. They were originally built for computer vision and image classification and are still popular in the field of computer vision.

CNNs consist of three layers: the convolutional layer, the pooling layer and the fully connected layer. The convolutional layer is where majority of the mathematics occurs[11]. It uses a kernel that acts as a sliding window to perform mathematics on an amount of bits that matches the kernel size. The pooling layer selects the value that is transformed by the kernel using either max or average pooling to select the value. The fully connected layer performs classification based on the previous layers results, typically using the ReLU function. CNNs widely used for their ability to detect spatial features of data. [11]

4.2.1 One Dimensional CNN. For one dimensional CNNs, the kernel slides across the data in one dimension only. They are known to work well with time series data. El Huq Qazi et al.[24] used a 1D CNN to classify network traffic for intrusion detection and achieved an accuracy of 99% by aggregating data from the CICIDS2017 dataset into flows. The author justifies the use due to the one-dimensionality of the data and that it is time series based. Tooke[29] achieved 84% accuracy using flows and the OceanView dataset with a 1D CNN on the dataset that we will be using using flows. This out-performed the MLP on every dataset size. The author justified the use of the 1D CNN due to their ability to recognize dependencies between successive packets. Furthermore, the 1D CNN was able to compete well with the MLP in terms of packets classified per second and outperformed all the MLPs in terms of accuracy.

4.2.2 Two Dimensional CNN. For two dimensional CNNs, the kernel has two dimensions, a width and a height and it slides across the data, capturing information in two dimensions. One of their main advantages is to pick up on spatial dependencies. Although most commonly used in computer vision, they can also be used in network traffic classification. Wang et al.[31] used a 2D CNN to detect malware over a network and used a series of publicly available datasets. Data was preprocessed into IDX format in order to represent data as images before being fed through the network. CNNs are commonly used in computer vision tasks which is why the authors used an image to represent the data. They achieved accuracy results greater than 90% for each dataset. Although the datasets were all publicly available and different to ours and the classification type was classifying malware not traffic types. Furthermore, aggregating data into a picture is very specific to this task and the preprocessing is very unique to this type of task and relies very much on human engineered data. This will be very impractical to do when performing real-time classification. Yet it does still prove the power and use of 2D CNNs with accuracy above 90%. Weisz[32] managed to attain 90.1% accuracy using the OceanView dataset with a 2D CNN using flows and formatting the data into an image format. He justified the use of these models due to their ability to pick up on spatial patterns. Although, these accuracy results were very good, 2D CNNs tended to be more computationally expensive than MLPs due to their high amount of parameters.

Wang et al. [30]used a flow-based approach to classifying encrypted traffic. They used a 1D CNN and a 2D CNN on the ISCXVPN dataset. For the 2D they extracted the information of the flows and converted them to image format to be read by the CNN. The 1D CNN used a one-dimensional vector. Over a wide range of experiments, the 1D CNN performed better than the 2D CNN. Since the data was split into flows time series aspects could be extracted and unsurprisingly the 1D CNN performed better than the 2D CNN. Figure 2 shows a high-level overview of a 2D CNN.

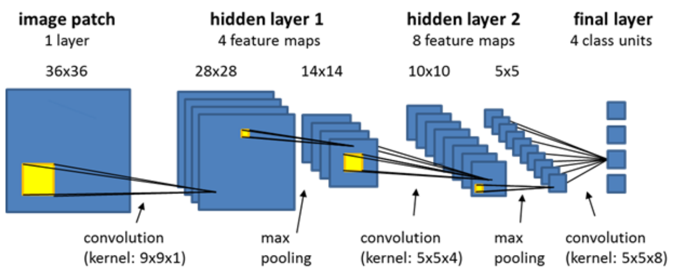

Figure 2: 2D CNN

4.3 Recurrent Neural Networks

Recurrent Neural Networks use MLPs as their foundation. Much like other neural networks they consist of input layers activation layers and output layers. In addition, they have a recurrent layer that helps information to persist over time. RNN connections do this by essentially storing computed by the previous pass through of the neural network to allow data to persist the use of time series. These models are widely used in time series analysis. Regular RNNs are not so often used due to the vanishing gradient problem [8] and not much work has been done using them to classify network traffic. Therefore, we will be looking at Long Short Term Memory networks(LSTM).

4.3.1 Long Short Term Memory Network. Long Short Term Memory Networks use gating mechanisms to control the flow of information and gradients.

Aceto et al.[1] used an LSTM along with other models to classify encrypted traffic from mobile networks flows using a private dataset. Accuracies of 81-97% were observed among different mobile operating systems. They also outperformed 2D CNNs due to its ability to perform well on time series data. 1D CNNs also achieved similar accuracy results also due to their ability to capture temporal dependencies.

Dicks[6] used an LSTM on the OceanView dataset and achieved accuracies greater than 90% using an LSTM. Although the LSTM performed well, LSTMs are very highly parameterized and compared to other models. Dicks observed great test accuracy compared to the MLP due to its ability to pick up on temporal dependencies, although the speed of classification was not suitable for online classification. He proposed using batched data to speed up the process which sped up the classification process without loss of accuracy.

Figure 3 shows a high-level overview of an RNN

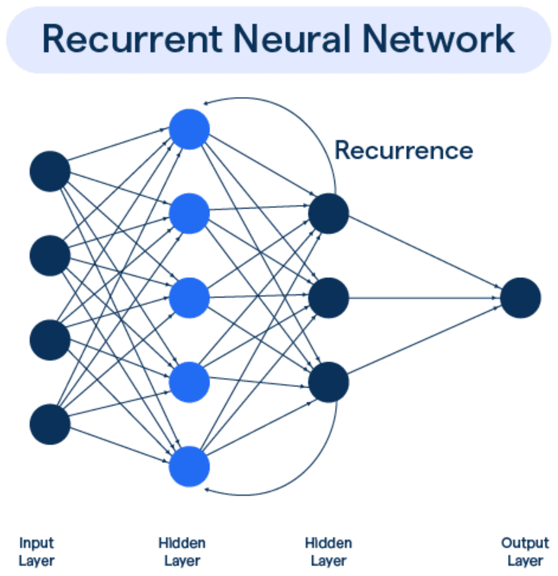

Figure 3: Recurrent Neural Network

4.4 Stacked Auto Encoders

Stacked Auto Encoders (SAE) are an unsupervised learning model that learns hierarchical representations of the data. They are composed of multiple layers of auto encoders stacked on top of each other. Input is fed through the input layer and compressed at the encoding levels. Then at the decoding layers the model tries to reconstruct the input, minimising the error between the reconstructed input and the original input.

Hochst et al.[9] used a neural auto encoder which is a single autoencoder on a dataset they collected themselves. They preprocessed it by aggregating data into flows and then taking the statistics of these flows. However, this method was not very effective, only achieving them an average F1 score per class of 76%. This is most likely due to the fact that the autoencoder was on its own and not a stacked autoencoder.

Lotfollahi et al. [17] used a stacked autoencoder and achieved an average accuracy of 92%. This was using the same deep packet method mentioned earlier. The first 1480 bytes of the IP payload as well as the IP header as input to the model. This proves the power of stacked autoencoders opposed to using single autoencoders and could also signal that SAEs are more suited to packet-based data. These results make a compelling point for the use of SAEs yet they still performed slightly worse than the CNN used. Figure 4 shows a high-level overview of an SAE.

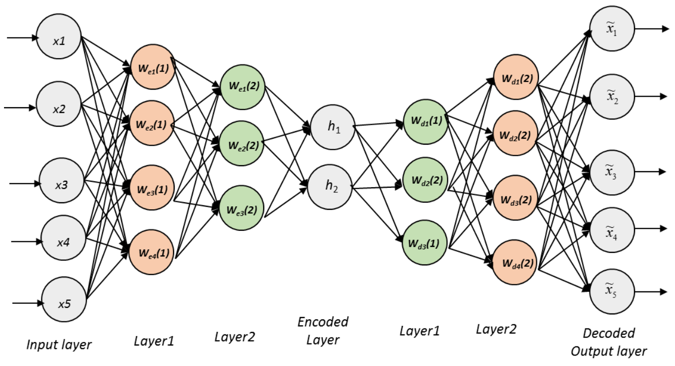

Figure 4: Stacked Auto Encoder

4.5 Ensemble Models

Ensemble models are models that combine two or more models together to leverage the strengths of specific models. Usually, data will be fed through a model and the output of the first model will be the input of the next model. They are there to enhance the performance of models.

Bayat et al.[4] used a range of different models including CNNs, RNNs, random forests and ensemble models. They used a CNN-RNN ensemble method to classify network traffic from a Google Chrome dataset. These models were trained and evaluated on packet-based and bidirectional flow-based data and inter-arrival times.Furthermore, they added a feature that specified whether a packet was client to server or server to client. The ensemble model consisted of two convolutional layers followed by a Gated Recurrent Unit(GRU) model which is a type of RNN.They managed to achieve an accuracy level above 95%. The use of a CNN with an RNN allowed them to use capture spatial features, which the CNN captures effectively and temporal features which the RNN captures effectively, proving the effectiveness of these models. Although the paper did not mention anything about how computationally expensive this model was. Based on the fact that RNNs and CNNs are generally computationally expensive, one could expect this model to be relatively computationally expensive.

However, Ramraj et al.[26] used a Multilayer perceptron to classify traffic from ISCXVPN dataset. The paper does not specifically mention whether they are performing flow-based or packet-based analysis but due to the use of port mirroring which is typically associated with packet level analysis. The regular MLP performed well on classifying traffic with an accuracy of 80%. Another experiment was performed with using an ensemble model which used a MLP and a support vector machine (SVM). An SVM is a supervised machine learning algorithm that works by finding the optimal hyperplane that best separates different classes in the input data. This model performed even better, achieving 90%. SVMs are effective at identifying patterns in feature space whereas MLPs are good at learning complex features in the dataset. This could be why the performance increased. Although, the paper again did not mention the computational costs these models incurred but one would expect them to be less than the CNN-RNN method due to CNNs and RNNs generally having many parameters.

Research Article	Dataset	Flow/Packet	Method
Aceto et al.[1]	Private Dataset	Flow	LSTM
CNN			
Bayat et al.[4]	Google Chrome Dataset	Flow	CNN-RNN
Dicks [6]	OceanView	Flow	LSTM
Hochst et al. [9]	Private Dataset	Flow	SAE
Khedkar et al. [14]	Not explicit	Packet	MLP
Lotfohalli et al. [17]	ISCXVPN	Packet	SAE&CNN
Pektas et al. [22]	CICIDS2017	Flow	CNN&LSTM
El Huq Qazi et al. [24]	CICIDS2017	Flow	CNN
Ramraj et al.[25]	ISCXVPN	Packet	CNN-SVM
Tooke [29]	OceanView	Flow	CNN
Wang et al.[30]	ISCXVPN	Flow	CNN
Wang et al.[31]	USTCTFC2016	Flow	CNN
Weisz[32]	OceanView	Flow	CNN

Table 1: Deep Learning Research Articles and Methods

5 DISCUSSION

Clearly, there is a lot of literature and work already been done on this line of research. The use of deep learning for community networks has not only been performed by others but has also been performed on the same dataset by Tooke, Dicks and Weisz [6, 29, 32]. These projects serve as the framework for our rendition of it. The goal is to improve on theirs with different, more efficient, and effective models.

The CNNs, RNNs, SAEs and ensemble models all generally performed better than the MLP [32]. RNN's like LSTMs were only used on traffic flows due to their ability to pick up on temporal dependencies. Although, one of the main disadvantages with flows is their increased overhead due to having to store packets before they are being fed to the model. This along with the high parameter count of the LSTMs could result in slower classification which could lead to real-time classification not being possible. Therefore, one would need to be careful not to over-complicate the model.

As models become more complex and parameterized, they take longer to train and make classifications. MLPs generally performed worse in terms of accuracy than any of the other models but were able to classify more packets per second. This creates a dilemma of how much speed are these network are we willing to trade off for accuracy and vice versa. Ensemble models could be a solution to this dilemma. These were by far the least most researched models in this line of research and could offer unique solutions. Ramraj et al. [26] used an SVM and an MLP which are two relatively computationally inexpensive models and achieved 90%. Unfortunately, packets classified per second was not recorded but one would expect this to be relatively low. Ensemble models take advantage of the fact that they can use the specialties of two or more different models.

From observing the literature, CNNs seem to be the most commonly used and most effective model in this line of research. This could be attributed to their versatility. They are able to pick up on spatial and temporal patterns using 2D and 1D kernels respectively [24, 31]. Furthermore, they outperformed MLPs which are also very versatile, due to them being more specialised and less general than MLPs. 1D CNNs performed better on flow data due to their ability to process temporal data [24]. 2D CNNs however required data to be transformed into IDX or an image-like format[29, 31]. One could make the case that 2D CNNs are more effective than due to the fact that they outperformed 1D CNNs using the OceanView dataset[29, 32]. Yet the 1D CNNs were able to classify more packets per second but were 6% less accurate. The choice of which to use is up to the person using the algorithm. Whether they would be willing to trade accuracy for speed or speed for accuracy is up to them. The 2D CNN may have outperformed the 1D CNN in terms of accuracy but not efficiency.

LSTMs performed well and achieved similar results to the CNNs, but they generally contain the most parameters compared to CNNs, MLPs and SAEs. Yet most of the studies that were observed showed them performing similarly or slightly worse than CNNs and having a lower packet classified per second rate [1, 6, 32].

The SAEs also did not have nearly as much literature as CNNs and LSTMs. Although they did perform well in the study by Lotfollahi et al. [17] they were not tested on the OceanView dataset which aggregates packets into flows.

One cannot say which model is the best for this project, even though a higher accuracy was achieved in one model. This can be attributed to different ways of processing the data and different uses of the model. For example, we have seen that LSTMs perform better on flow data but SAEs performing better on packet-based data. Although overall flow-based data, especially bi-directional flows, seemed to be the most widely used way or preprocessing the data and these models all seemed to adapt and perform well using it. Although flow-based data is not as suited to real-time classification due to the fact that you need to wait to receive enough packets to aggregate them into a flow. Packet-based classification is more suitable for real-time classification [17].

5.1 Conclusion

This paper has spoken about the different approaches for traffic classification and has stated the reasons with evidence why deep learning is most suitable for this project, namely CNNs. Furthermore, an overview of community networks and why they can benefit from this project was also provided Although when we actually execute this project, we may find the results to be different. Community networks can benefit greatly from projects like this especially if the model is fast and accurate. Using the work performed by Tooke, Dicks and Weisz [6, 29, 32] our goal is to improve on their work in this field to find more effective and efficient models. From the papers written, it seems the most effective models are CNNs and then LSTMs and the use of ensemble models have a lot of potential. The next move will be to start analysing models that we think will analyse which models we plan on using based on the work of others mentioned in this review to find a model that can provide QoS to community networks.

REFERENCES

[1] Giuseppe Aceto, Domenico Ciuonzo, Antonio Montieri, and Antonio Pescapé. 2018. Mobile encrypted traffic classification using deep learning. In *2018 Network traffic measurement and analysis conference (TMA)*. IEEE, 1–8.

[2] Razan M AlZoman and Mohammed JF Alenazi. 2021. A comparative study of traffic classification techniques for smart city networks. *Sensors* 21, 14 (2021), 4677.

[3] Ahmad Azab, Mahmoud Khasawneh, Saed Alrabaee, Kim-Kwang Raymond Choo, and Maysa Sarsour. 2022. Network traffic classification: Techniques, datasets, and challenges. *Digital Communications and Networks* (2022).

[4] Niloofar Bayat, Weston Jackson, and Derrick Liu. 2021. Deep learning for network traffic classification. *arXiv preprint arXiv:2106.12693* (2021).

[5] Manuel Crotti, Maurizio Dusi, Francesco Gringoli, and Luca Salgarelli. 2007. Traffic classification through simple statistical fingerprinting. *ACM SIGCOMM Computer Communication Review* 37, 1 (2007), 5–16.

[6] Matthew Dicks, Jonathan Tooke, and Shane Weisz. [n. d.]. A Comparative Evaluation of Deep Learning Approaches to Online Network Traffic Classification for Community Networks. ([n. d.]).

[7] Gerard Draper-Gil, Arash Habibi Lashkari, Mohammad Saiful Islam Mamun, and Ali A Ghorbani. 2016. Characterization of encrypted and vpn traffic using time-related. (2016), 407–414.

[8] Sepp Hochreiter. 1998. The vanishing gradient problem during learning recurrent neural nets and problem solutions. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems* 6, 02 (1998), 107–116.

[9] Jonas Höchst, Lars Baumgärtner, Matthias Hollick, and Bernd Freisleben. 2017. Unsupervised traffic flow classification using a neural autoencoder. In *2017 IEEE 42Nd Conference on local computer networks (LCN)*. IEEE, 523–526.

[10] iNethi Technologies. 2024. iNethi Technologies. https://www.inethi.org.za/. Accessed: March 24, 2024.

[11] Alon Jacovi, Oren Sar Shalom, and Yoav Goldberg. 2018. Understanding convolutional neural networks for text classification. *arXiv preprint arXiv:1809.08037* (2018).

[12] Christian Janiesch, Patrick Zschech, and Kai Heinrich. 2021. Machine learning and deep learning. *Electronic Markets* 31, 3 (2021), 685–695.

[13] Andrea Kavanaugh, John M Carroll, Mary Beth Rosson, Than Than Zin, and Debbie Denise Reese. 2005. Community networks: Where offline communities meet online. *Journal of Computer-Mediated Communication* 10, 4 (2005), JCMC10417.

[14] Shilpa P Khedkar and Aroul Canessane Ramalingam. 2021. Classification and Analysis of Malicious Traffic with Multi-layer Perceptron Model. *Ingénierie des Systèmes d'Information* 26, 3 (2021).

[15] Yeon-sup Lim, Hyun-chul Kim, Jiwoong Jeong, Chong-kwon Kim, Ted" Taekyoung" Kwon, and Yanghee Choi. 2010. Internet traffic classification demystified: on the sources of the discriminative power. In *Proceedings of the 6th International COnference*. 1–12.

[16] Manuel Lopez-Martin, Belen Carro, Antonio Sanchez-Esguevillas, and Jaime Lloret. 2017. Network traffic classifier with convolutional and recurrent neural networks for Internet of Things. *IEEE access* 5 (2017), 18042–18050.

[17] Mohammad Lotfollahi, Mahdi Jafari Siavoshani, Ramin Shirali Hossein Zade, and Mohammdsadegh Saberian. 2020. Deep packet: A novel approach for encrypted traffic classification using deep learning. *Soft Computing* 24, 3 (2020), 1999–2012.

[18] Kevin P Murphy. 2012. *Machine learning: a probabilistic perspective.* MIT press.

[19] John D Owens, Mike Houston, David Luebke, Simon Green, John E Stone, and James C Phillips. 2008. GPU computing. *Proc. IEEE* 96, 5 (2008), 879–899.

[20] Bo Pang, Erik Nijkamp, and Ying Nian Wu. 2020. Deep learning with tensorflow: A review. *Journal of Educational and Behavioral Statistics* 45, 2 (2020), 227–248.

[21] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. 2019. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems* 32 (2019).

[22] Abdurrahman Pektaş and Tankut Acarman. 2019. A deep learning method to detect network intrusion through flow-based features. *International Journal of Network Management* 29, 3 (2019), e2050.

[23] Marius-Constantin Popescu, Valentina E Balas, Liliana Perescu-Popescu, and Nikos Mastorakis. 2009. Multilayer perceptron and neural networks. *WSEAS Transactions on Circuits and Systems* 8, 7 (2009), 579–588.

[24] Emad Ul Haq Qazi, Abdulrazaq Almorjan, and Tanveer Zia. 2022. A one-dimensional convolutional neural network (1D-CNN) based deep learning system for network intrusion detection. *Applied Sciences* 12, 16 (2022), 7986.

[25] Prajit Ramachandran, Barret Zoph, and Quoc V Le. 2017. Searching for activation functions. *arXiv preprint arXiv:1710.05941* (2017).

[26] S Ramraj and G Usha. 2023. Hybrid feature learning framework for the classification of encrypted network traffic. *Connection Science* 35, 1 (2023), 2197172.

[27] Shahbaz Rezaei and Xin Liu. 2019. Deep learning for encrypted traffic classification: An overview. *IEEE communications magazine* 57, 5 (2019), 76–81.

[28] Sima Siami-Namini, Neda Tavakoli, and Akbar Siami Namin. 2019. The performance of LSTM and BiLSTM in forecasting time series. In *2019 IEEE International conference on big data (Big Data)*. IEEE, 3285–3292.

[29] Jonathan Tooke. [n. d.]. Deep Learning Classifiers for QoS and Traffic Engineering in Community Networks. ([n. d.]).

[30] Wei Wang, Ming Zhu, Jinlin Wang, Xuewen Zeng, and Zhongzhen Yang. 2017. End-to-end encrypted traffic classification with one-dimensional convolution neural networks. In *2017 IEEE international conference on intelligence and security informatics (ISI)*. IEEE, 43–48.

[31] Wei Wang, Ming Zhu, Xuewen Zeng, Xiaozhou Ye, and Yiqiang Sheng. 2017. Malware traffic classification using convolutional neural network for representation learning. In *2017 International conference on information networking (ICOIN)*. IEEE, 712–717.

[32] Shane Weisz and Josiah Chavula. 2021. Community Network Traffic Classification Using Two-Dimensional Convolutional Neural Networks. In *International Conference on e-Infrastructure and e-Services for Developing Countries*. Springer, 128–148.

[33] Shengxu Xie, Guyu Hu, Xiulei Wang, Changyou Xing, Yaqun Liu, et al. 2022. A Decision Tree-Based Online Traffic Classification Method for QoS Routing in Data Center Networks. *Security and Communication Networks* 2022 (2022).

[34] Jun Zhang, Xiao Chen, Yang Xiang, Wanlei Zhou, and Jie Wu. 2014. Robust network traffic classification. *IEEE/ACM transactions on networking* 23, 4 (2014), 1257–1270.