APPLICATION TRAFFIC USING TENSORFLOW MACHINE LEARNING

[1]Netravati Patil, [2] Pavitra Gadhar
[1]Student, [2]Assistant Professor
Computer Science and Engineering,
Rural Engineering College, Hulkoti, Gadag, Karnataka, India

Abstract: TensorFlow is an interface for expressing machine learning algorithms and also implementation for executing such algorithms. In the day-by-day these applications are becoming more complicated and diverse as the network environment grows. It is important to classify application traffic accurately. There are many ways to classify applications traffic; machine learning based approaches are becoming more efficient in modern days. This is because machine learning methods are more appropriate than existing methods for accurate and efficient applications traffic classification. Payload signature methods have limitations to deal with various patterns and increasing application traffic complexity. In this paper, we propose a method for extracting flow features and a system for classifying applications traffic based on Machine Learning. This paper describes the TensorFlow interface and an implementation of that interface that we have built at Google. The TensorFlow API and a reference implementation were released as an open-source package.

***Index Terms*— Traffic Classification, Machine Learning, Learning Feature, Flow Feature Extraction, Normalization**

I.INTRODUCTION

With the speed Internet volume growth, network environment is daily becoming complex and diverse. So that application traffic patterns are becoming even more diverse and complex. In this situation, network traffic monitoring and analysis is essential for effective network operation and stable service provision. In order to analyze the network traffic, application traffic classification method must be preceded. And, it is important to find the appropriate methods for precise classification of application traffic. The most broadly known method on these days is the payload signature based classification method. We extract the existing application traffic payload signatures and match that payload signature with other traffic patterns.

The application traffic classification based on payload signature has more advantages with high accuracy and performance. There are also many problems of this method. It is computationally expensive for real time handling of large amounts of traffic on high speed network. And the classification method has a limitation that it is difficult to cope with new patterns other than the existing ones. The application traffic classification method based on the machine learning can solve these problems. If a new pattern of application traffic is found, it will be classified itself based on patterns learned before. Also many companies provide machine learning OpenAI tools so that people can use it freely. Among the machine learning tools, we use TensorFlow for this system. Because it is not only well-known and also easy to use.

There are several cautions to classify application traffic based on machine learning. First, it is important to apply appropriate learning features to be used in Machine Learning. Inappropriate learning features result in poor accuracy compared to existing traffic classification methods. Second, the experiment should be conducted many times under various learning features. It is important to find the optimal learning features. Because the results are greatly depend on the learning features. In order to find optimal learning features, many experiments should be conducted with various learning features.

II.ABOUT THE TECHNOLOGY

2.1 What is TensorFlow?

TensorFlow is a Python-friendly open source library for numerical computation that makes machine learning faster and easier. Machine learning is a complex discipline. But implementing machine learning models is far less daunting and difficult than it is used, machine learning frameworks- such as Google's Tensor-Flow that ease the process of acquiring data, training models, serving predictions, and refining future results.

Created by the Google Brain team, TensorFlow is an open source library for numerical computation and large scale machine learning. TensorFlow bundles together a slew of machine learning and deep learning models and algorithms and makes them useful by way of a common metaphor. It uses Python to provide a convenient front-end API for building applications with the framework, while executing those applications in high-performance C++.

TensorFlow is used to train and run deep neural networks for handwritten digit classification, image recognition, word embeddings, recurrent neural networks, sequence-to-sequence models for machine translation, natural language processing and PDE (Partial differential equation) based simulations. Best of all, TensorFlow supports production prediction at scale, with the same models used for training.

2.2 How TensorFlow works?

TensorFlow allows developers to create dataflow graphs-structures that describe how data moves through a graph, or a series of processing nodes. Each node in the graph represents a mathematical operation, and each connection or edge between nodes is a multidimensional data array, or tensor.
TensorFlow provides all of this for the programmer by way of the Python language. Python is easy to learn and work with, and provides convenient ways to express how high-level abstractions can be coupled together. Nodes and tensors in TensorFlow are Python objects, and TensorFlow applications are themselves Python applications.
TensorFlow applications can be run on most any target that's convenient: a local machine, a cluster in the cloud, iOS and android devices, CPUs or GPUs. If you use Google's own cloud, you can run TensorFlow on Google's custom TensorFlow Processing Unit (TPU) silicon for further acceleration. The resulting models created by TensorFlow, though, can be deployed on most any device where they will be used to serve predictions.
TensorFlow can be used for all machine learning algorithms, but deep learning is kind of specific to neural networks. Deep learning is like using neural networks to build machine intelligent systems .And Tensor-Flow is a proper library for programming but deep learning is just approach.
For example, linear regression is a supervised machine learning approach, using Tensor-Flow we can build linear regression. But this linear regression is not deep learning.
TensorFlow can be used to build deep learning algorithms. We can build deep learning neural networks from TensorFlow with more than 1 hidden layer. We can use this for image recognition, object recognition and various classification problems.
To conclude, we can say TensorFlow is the bigger library to perform various operations both for normal and deep neural networks .Deep learning is a approach for forming Neural networks with more than 1 hidden layer
.

2.3 What do developers need to do to use Tensorflow?

TensorFlow was created with processing power limitations in mind, making it easier for mobile and web developers to make use of the library and create AI-powered features for consumer products. Developers with a basic background in neural networks can use the framework for data sets, estimators, training, training and inference.
Developers with no background in neural networks may want to start with a higher neural network API, such as Keras.io. Written in Python, Keras is capable of running top of TensorFlow, CNTK and Theano and is good for easy and fast prototyping.
To start with TensorFlow, a developer needs to know:

- Python or C++
- A little bit about arrays (specifically the numpy.array, which is provided by Numpy, numerical computation library for Python, that TensorFlow uses to deal with an array/matrix)

III.RELATED WORK

There have been various studies on application traffic classification. The most common classification method among the various methods is a payload signature based classification method. This method extracts a unique signature for each payload of each application traffic and classifies it based on the extracted signature. Such classification based on payload signature is difficult to apply if the data is encrypted. A system automatically generates a signature that has been developed, which allows signatures to be generated even if the data is encrypted.
Classification of application traffic with generated signatures gives high accuracy. It gradually becomes difficult to flexibly cope with the complicated and diverse application traffic pattern. Machine learning method can be solution of this problem. The greatest advantage of machine learning can cope with diverse situation even if a new situation occurs abruptly. Therefore, if machine learning is applied to application traffic classification, it can be flexibly copied with a complicated and diverse application traffic pattern. Both methods result high accuracy and performance. But, payload signature method is not only expensive but also hard to cope with diverse traffic pattern. Machine learning method can solve these problems.

3.1 TensorFlow execution model

TensorFlow uses a single dataflow graph to represent all computation and state in a machine learning algorithm, including the individual mathematical operations, the parameters and their update rules, and the input preprocessing. The dataflow graph expresses the communication between sub computations explicitly, thus making it easy to execute independent computations in parallel and to partition computations across multiple devices.

TensorFlow differs from batch dataflow systems in two respects:

• The model supports multiple concurrent executions on overlapping sub graphs of the overall graph.
• Individual vertices may have mutable state that can be shared between different executions of the graph.

The key observation in the parameter server architecture is that mutable state is crucial when training very large models, because it becomes possible to make in-place updates to very large parameters, and propagate those updates to parallel training steps as quickly as possible. Dataflow with mutable state enables TensorFlow to mimic the functionality of a parameter server, but with

additional flexibility, because it becomes possible to execute arbitrary dataflow sub graphs on the machines that host the shared model parameters. As a result, our users have been able to experiment with different optimization algorithms, consistency schemes, and parallelization strategies.

IV.PROPOSED CLASSIFICATION SYSTEM

In this paper, application traffic classification system is proposed based on machine learning. Figure 1 shows the overall structure of the proposed system. Proposed structures are divided into 4 types.

Application Traffic Classification System Overview

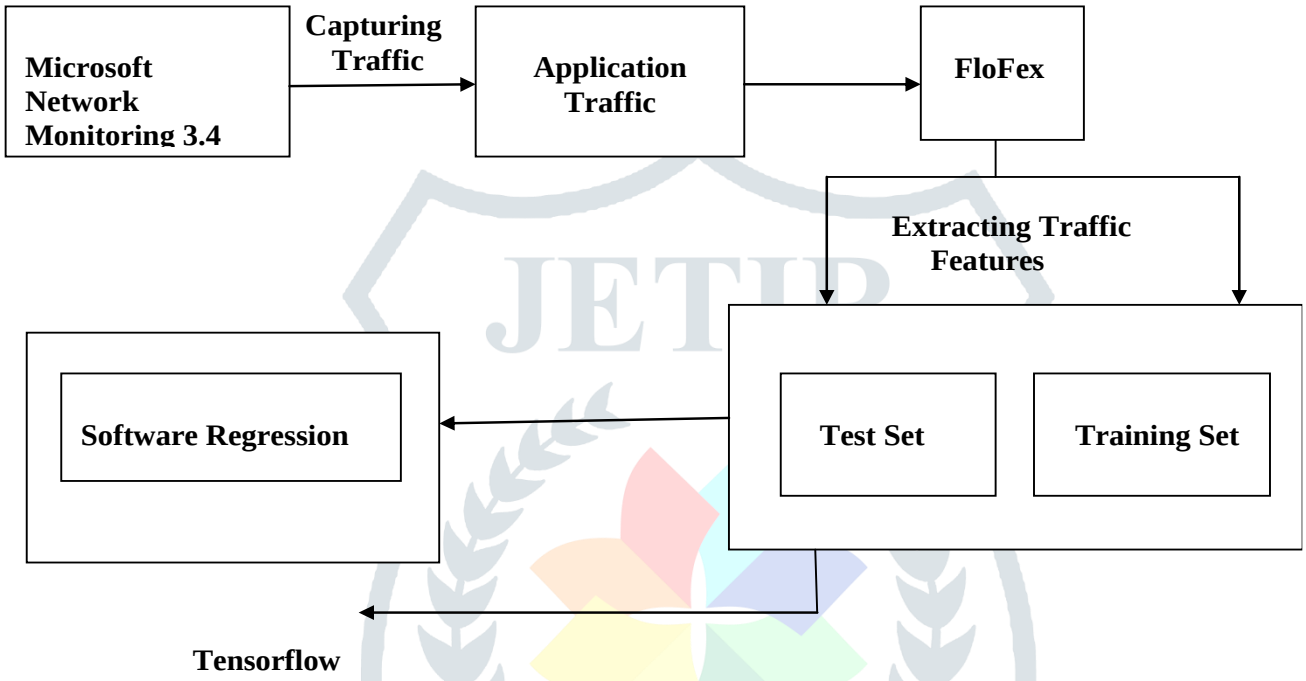

Figure 1: Proposed Traffic Classification System Overview

In the first step, collect application traffic. In this step, we capture from various applications through Microsoft Network Monitor 3.4. Application traffic should be captured in appropriate amount. If the amount of traffic is not adequate, it will not give accurate classification results, but it also takes a long time.

In the second step, extract the learning features of the collected application traffic. In this step, instead of existing learning feature extraction system is proposed learning feature extraction system. Few of learning features are extracted by using existing system. The proposed learning feature extraction system can solve this problem. It will be mentioned more detail in III-1. Extracted learning features of each application are divided into Train Set and Test Set. Train Set is a set of extracted learning feature to be learned through machine learning. Test Set is a set of extracted learning feature to be tested whether the classified traffic is correctly classified at the next stage.

In the third step, classified Train Set and Test Set are normalized. There are number of ways to normalize. However, the result greatly depends on which normalization method is used. Therefore, it is also important to use sophisticated and appropriate normalization method in this step.

Finally, conduct the several experiments with normalized Train Set and Test Set. We will use Softmax regression among the various classification methods of TensorFlow. We will refer each of steps in more detail in III-2.

III-1 FloFex - Learning Feature Extraction System

As mentioned before, it is very important to extract appropriate learning features. Originally extracting methods are not only complicate but few learning features are extracted. Therefore, we propose a system that can extract learning features of traffic more efficient and various than other extracting methods. A system consists of 2 parts.

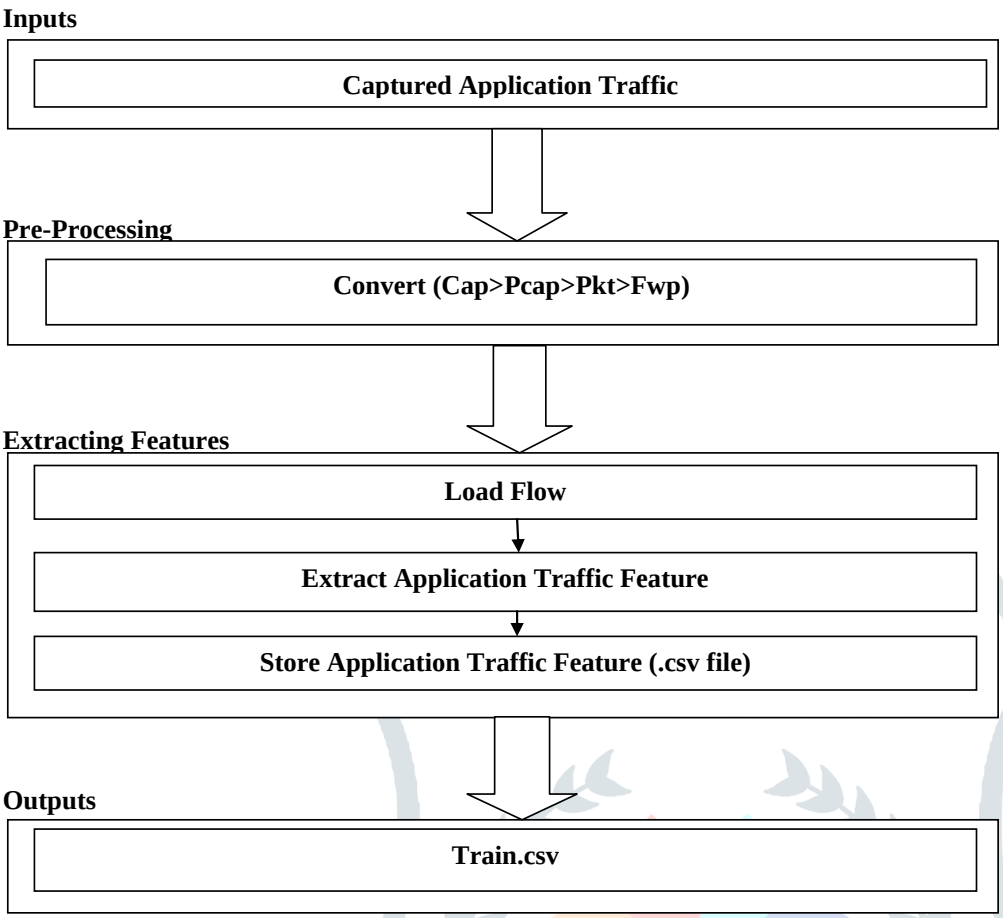

Figure 2: FloFex System Structure

The structure of the FloFex is shown in Figure 2. First, transform the captured application traffic coming into the input into the proper form. Second, the learning feature of the application traffic is extracted based on the modified fwp file. Finally, the learning features of the extracted application traffic are stored in the output.

The structures of the extracted learning features are shown in Figure 3. There are about 300 learning features that can be extracted from FloFex. They can be divided into main feature and sub feature. The reason for dividing into two parts is to explain and classify efficiently.

Main feature occupies most of learning features extracted by FloFex. In Main feature, 288 Features are derived. The main feature can be divided into 6 types according to time area (During 1 ~ 5 seconds, and Total time). Main feature also can be divided into 2 parts as packet size, and inter arrival time. Packet size corresponds a size of packets, and inter arrival time corresponds the time between packets in each flow. Each of them divided into total, forward, and backward, and each has 8 statistical values (max, min, median, sum, mean, variance, 1st_variance, 3rd_variance).

Sub feature can be divided into four major parts as shown in Figure 3 of Sub feature. In sub feature, 15 features are derived. First, packet count is the number of packets in each flow, which can be divided into total, forward, and backward. Packet count with payload is also the number of packets. But packet count with payload is the number of packets excluding the packets which payload length is zero. Second, the flow size, duration, and protocol correspond to the size (number of bytes), duration, and protocol of each flow. Source port, source address, destination port and destination address also correspond to the port number of source and destination and address of source and destination corresponding to each flow.

Finally, PPS (Packet per Second) is defined as the number of packets sent per second. PPS is divided into PPS and FPPS BPPS according to total packet, forward, packet and backward packet as well as packet count. That is, they represent the percentage of total, forward, and backward packets sent for one second each.

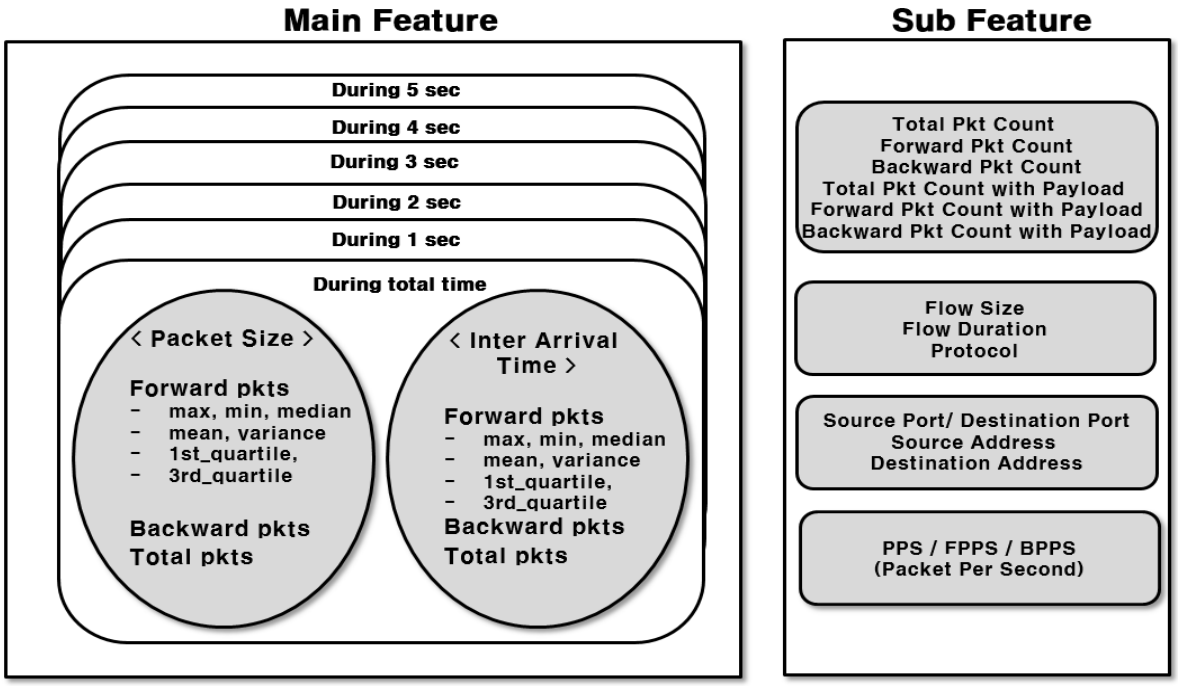

Figure 3: Learning Feature List Structure Extracted From FloFex

To summarize, there are 24 learning features can be extracted in each time zone (During 1~5 seconds, Total time). So, packet size and inter arrival time each has 144 learning features, resulting in total of 288 learning features. In the sub feature, there are 15 learning features. Therefore, total learning feature counts come out to 303.

III-2 Traffic Classification based on Machine Learning

In Figure 1, there are 4 parts of application traffic classification system based on machine learning which is proposed in this paper. We collected application traffic and extract learning features by using FloFex. Then extracted learning feature is divided into Train Set and Test Set. Because it has 2 Sets has a different of role. Train Set is a Set for learning. It occupies for 70 ~ 80% of the extracted learning features. Test Set is a Set to verify learning. Test Set occupies the rest of the extracted learning features. Figure 4 is a simplified representation of the first and second step described in Figure 1.
In the third step, normalize learning features which are extracted from FloFex. Because the difference between each values of extracted learning features is too large to classify. If we do not normalize these values, it classified inaccurately. Therefore, it is important to use an appropriate normalization method. Although there are many methods of normalization, we use Min-Max Normalization. In the final step, the experiment is performed with the normalized Train Set and Test Set.

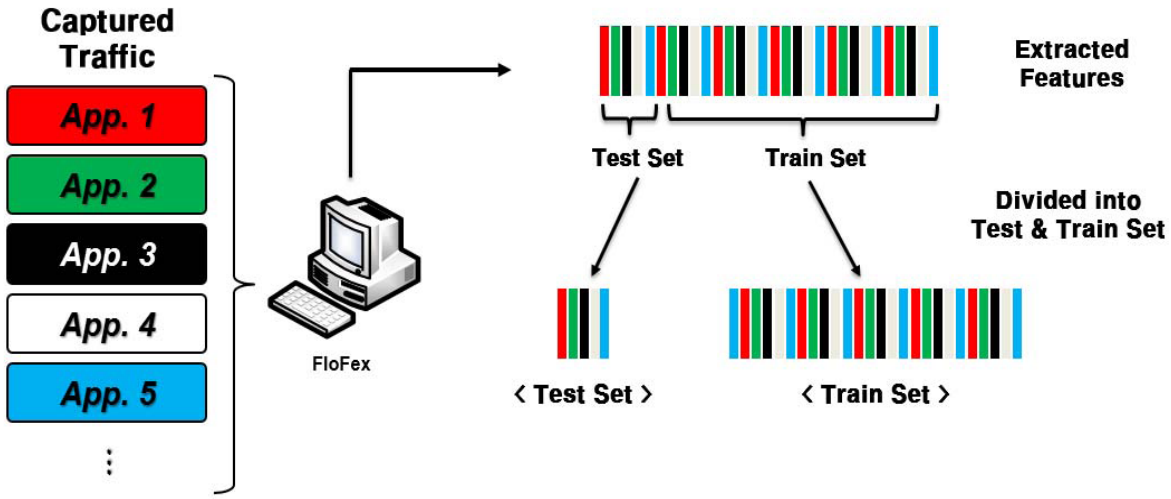

Figure 4: 1, 2 Steps of Traffic Classification based on Machine Learning

First, train with the Train Set and test with the Test Set. There are many ways to learning and classifying by using Tensorflow. Among them, we will use softmax regression. Softmax regression is a classification method that is used mainly for three or more classification objects (Multinomial Classification). When there are 2 classification objects, we should use logistic regression (Binary Classification). It is more simple method than softmax regression. However, the number of application traffic which we should classify is more than 2. Therefore, it is suitable to use softmax regression.
Second, test with Train Set and check its accuracy. In this case, we should conduct many experiments to obtain the optimal learning rate and the number of loops with the highest accuracy. And apply them to the most optimal extracted learning features with the obtained learning rate and the number of loops. Then, we make an application traffic classification model based on machine learning.

V.EVALUATION

In order to prove the validity of the proposed system, we conduct many experiments. We collected the web browser application traffic. The web browsers used in this experiment are Chrome, Firefox, Internet Explorer, Swing, and Whale. All of these 5 web browsers application traffic are collected in same conditions.

The number of loops is the number of learning. Usually if the number of loops is bigger, the learning result is better. But it takes a lot of time to run many loops. Learning rate indicates the rate of learning. If the learning rate is too small, then it will spend a lot of time. On the other hand, if the learning rate is too big, then spending time will be shortening, but the results of accuracy will be poor. Therefore, it is important to find the optimal learning rate through experiments. Cost shows the difference between the actual value classified and the predicted value based on learning. Cost is the better when it converges to 0. Accuracy, on the other hand, represents the percentage of correctly classified traffic among the analyzed traffic. Accuracy is the better when it converges to 100%.

In this section, the proposed method is evaluated as:

a) **Aggregation Method:** Two aggregation methods were examined. The cumulative method (cum) is around 15% worse than using the non-cumulative version (noncom), where the flow is separated into multiple segments and the statistical values are computed individually.

b) **Number of Clusters:** The number of clusters is defined by the number of hidden nodes of our neural auto encoder. In our parameter scan, we evaluated 10, 15, 20, 30, 40, 60, 80 and 100 clusters. The experiment shows that a sweet spot can be identified when 60 clusters are used, since they averaged results are not significantly better when more clusters are used.

c) **Scalar:** The scalar has a major impact on the classification results. While average precision and recall are only at roughly 50% when no scalar is used, the standard scalar improves average precision and recall up to around 60%.60%.

d) **Classification Quality:** The neural auto encoder in the configuration with 100 clusters, learning in 30 epochs with a standard scalar based on the full dataset produced the best result. These 100 clusters are mapped by our classifier to our 7 chosen classes. An average precision of 80% and an average recall of 75% are achieved, which results in an F1 score of 76%.

e) **Execution Time:** Our runtime experiments were performed on a 2 x 2.26 GHz Intel Xeon quad-core machine. While the generation of flow objects from the pcap file took around 2.20 ms per flow and the computation of the feature vectors took about 1.67 ms per flow, the actual classification only took 0.006 ms per flow. With a rate of over 200,000 flows per second, our method can be used at the network edges, where access points or mobile devices themselves can classify the traffic and dynamically change connection properties using SDN and SDWN technology to ensure optimal resource usage.

In the cellular automata traffic flow model, the traffic state can be represented by the discrete speed value of vehicles, thus the traffic flow can be deemed as a discrete dynamical system. In the evolution process of traffic flow, complex networks are constructed by representing the traffic state as node and the evolution relationship in timescale as link. The emerging times of link is defined as its weight, then the node strength is equal to the emerging times of the corresponding traffic state.
As a result, a weighted network is obtained. The dynamics of stop-and-go traffic are studied by investigating the statistical properties of network. Simulation results show that scale-free behavior commonly exists in the evolution process of stop-and-go traffic. The degree distributions have the power law form. The node with high degree also has large strength. The structure of the network is not influenced by the randomization probability and density as long as the stop-and-go traffic is reproduced.

VI.ADVANTAGES AND DISADVANTAGES

Advantages:

1. You can do anything in Tensor-flow.

2. Tensorflow is the industry standard meaning that a job in machine learning will most likely require knowledge and experience in tensor-flow. According to tensor-flow's website, companies like airbnb, AMD, NVidia, UBER, Drop Box, Ebay, and Google (obviously), snap chat, Intel, and twitter all use tensorflow.
3. You can use keras as an interface for tensor-flow. This means you can write some parts of your code with keras then do other parts that require the low level functionality with tensor-flow.
4. Tensorflow can also be used as a general hardware acceleration library (not recommended for end user cases due to complicated installation).
5. If you are pursuing deep learning as a career, you will need to learn it at some point.
6. There are tons of great tutorials out there for tensorflow and because of the high levels of adoption; it is easy to find resources.
7. Tensorflow is compatible with goggle's TPUs which can run tensor operations much faster than just a GPU allowing your deployed models to run in the cloud much faster and cheaper than normal cloud instances (TPUs can only execute models, they can't train them).
8. It is open source.

Disadvantages:

1. Tensorflow is very low level with a steep learning curve.
2. You will probably not need any of the super low level stuff.
3. Due to tensor-flow's odd structure, it can be hard to figure out what some errors are even regarding. This makes the general debugging process difficult.

In summary, tensorflow is so widely used because it's just awesome. The worst part about tensor-flow is that it's just plain hard but if you use keras to build most of the things in tensor-flow, you can still use tensor-flow to do the nitty gritty low level stuff that makes your project unique.

VII.CONCLUSION AND FUTURE WORK

In this paper, we proposed an application traffic classification system based on machine learning. We also designed a Flow Feature Extraction system (FloFex) for the proposed classification system. FloFex is able to extract more diverse and efficient than existing learning feature extraction system. Through various experiments, we confirm high accuracy and performance similar to existing traffic classification based on payload signature. And we also solve several problems of traffic classification based on payload signature. In the future, we will try to find more optimal learning features than used in these experiments. We will also conduct various experiments with other application traffic to make precise traffic classification models.

There are several areas for future work, such as,

a) Using deep and especially stacked auto encoders to improve the mapping of classes to clusters.
b) Replacing the SoftMax classification method by other methods to improve the classification.
c) Training the network with sub flows of varying lengths to use our approach for nearly real time classification after observing only a few seconds of the packet stream.

REFERENCES

[1] S.-H. Lee, K.-S. Shim, Y.-H. Goo and M.-S. Kim, "Application Traffic Classification using Tensorflow Machine Learning Tool", Nov. 18. 2016, pp224-225

[2] J.-S. Park, S.-H. Yoon, H.-M. Ann and M.-S. Kim "Classification of Internet Traffic Based on Payload Signature". May. 15-16, 2014, pp10-14

[3] Y.-H. Goo, K.-S. Shim, S.-K. Lee, M.-S. Kim "Payload Signature Structure for Accurate Application Traffic Classification" Proc of the Asia-Pacific Network Operations and Management Symposium (APNOMS) 2016, Kanazawa, Japan, Oct. 5-7, 2016

[4] S.-H. Lee, J.-S. Park, S.-H. Yoon and M.-S. Kim, "High performance payload signature based internet traffic classification system", Proc. Of the Asia-Pacific Network Operations and Management Symposium (APNOMS) 2015, Busan, Korea, Aug, 19-21, 2015.pp491-494

[5] J.-S. Park, S.-H. Yoon H,-M. Ann, M.-S. Kim," Internet Traffic Classification Based on Payload Signature". 2014 Communication Network Management Conference (KNOM2014), Chungnam National University, Daejun, May. 15-16, 2014, pp10-14

[6] Y.-H. Goo, S.-H. Lee, K.-S. Shim, M.-S. Kim "Traffic Classification Method Using Association Model of Network Flow", Journal of KICS, Vol.44 No.04, Apr. 2017, pp. 433-438 1157-1182