Network Traffic Classification: Machine Learning vs. Deep Learning Approach

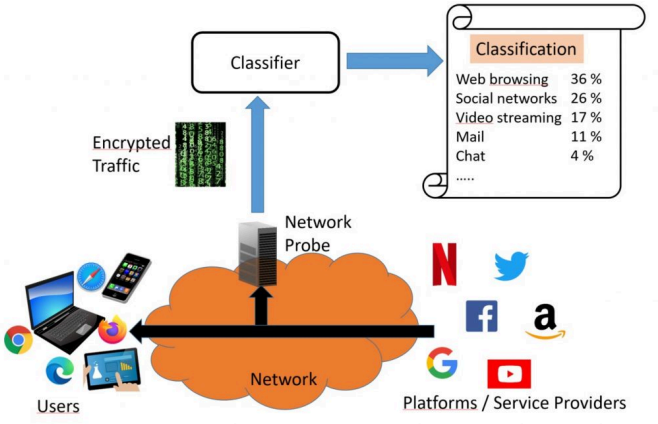
Robert Szmurło

e-mail: robert.szmurlo@pw.edu.pl

https://github.com/szmurlor/cpee-ntc-2024/

Motivation and background

- Understand difference between Machine Learning and Deep Learning on Network Traffic Classication use case.
- Port-based, data packet inspection, and classical machine learning methods have been used extensively in the past, but their accuracy has declined due to the dramatic changes in Internet traffic, particularly the increase in encrypted traffic.
- Thus, the major issue is that classification is impacted by service name encryption. How could this practice evolve?
- See how transformer networks can help in a generative approach.

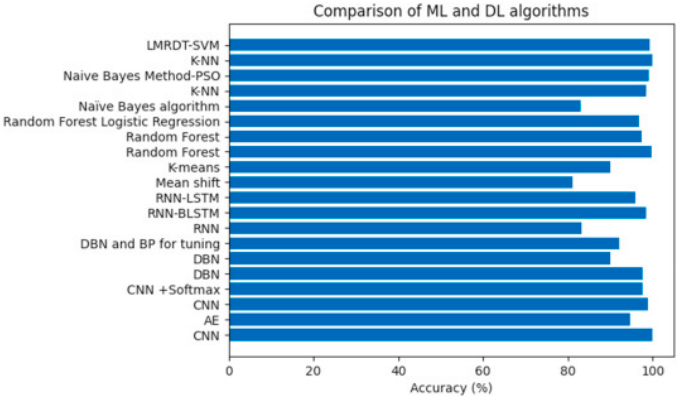


https://www.sciencedirect.com/science/article/pii/S2665917423001630

Background: AI and ML for computer networking

What is is used for? How can it be usefull?

Network classification is the first step of network traffic analysis, and it is the core element of network intrusion detection systems (IDS).



1. **Network Security**: Identifying and mitigating malicious traffic and intrusions.
2. **Quality of Service (QoS) Management**: Prioritizing critical applications for better performance.
3. **Bandwidth Management**: Controlling and optimizing bandwidth usage to prevent congestion.
4. **Content Filtering and Parental Controls**: Filtering web content based on categories for security and compliance.
5. **Application Performance Monitoring (APM)**: Analyzing traffic patterns to troubleshoot performance issues.
6. **Network Planning and Optimization**: Using traffic data for capacity planning and infrastructure upgrades.

NTC (Network Traffic Classification) - purpose

To operate networks, Internet access providers classify traffic by category of application and by service providers.

NTC (Network Traffic Classification) - purpose

To operate networks, Internet access providers classify traffic by category of application and by service providers.

Classification allows to know the market shares of applications or categories of applications. Usage analysis by customer segment and by offer in turn allows for bringing more relevant offers to the market

NTC (Network Traffic Classification) - purpose

To operate networks, Internet access providers classify traffic by category of application and by service providers.

Classification allows to know the market shares of applications or categories of applications. Usage analysis by customer segment and by offer in turn allows for bringing more relevant offers to the market

Traffic classification also allows to measure the quality of service by application or category of applications. This allows for predicting traffic evolutions in order to adapt network capacities and offer the best possible quality to users.

NTC (Network Traffic Classification) - purpose

To operate networks, Internet access providers classify traffic by category of application and by service providers.

Classification allows to know the market shares of applications or categories of applications. Usage analysis by customer segment and by offer in turn allows for bringing more relevant offers to the market

Traffic classification also allows to measure the quality of service by application or category of applications. This allows for predicting traffic evolutions in order to adapt network capacities and offer the best possible quality to users.

For example, traffic classification helped anticipate Netflix's move to 4K. It also helps to prepare the impact of a football match on data rates, depending on the broadcaster.

NTC (Network Traffic Classification) - purpose

To operate networks, Internet access providers classify traffic by category of application and by service providers.

Classification allows to know the market shares of applications or categories of applications. Usage analysis by customer segment and by offer in turn allows for bringing more relevant offers to the market

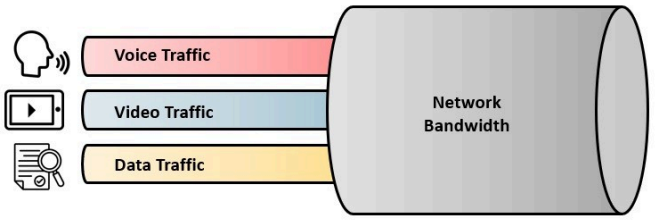
Traffic classification also allows to measure the quality of service by application or category of applications. This allows for predicting traffic evolutions in order to adapt network capacities and offer the best possible quality to users.

For example, traffic classification helped anticipate Netflix's move to 4K. It also helps to prepare the impact of a football match on data rates, depending on the broadcaster.

In fraud detection, traffic classification helps to identify misuse, for example when unbilled traffic is used for applications other than those intended

Presentation plan

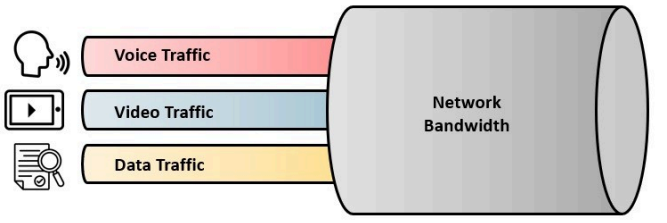
- Survey of Network Traffic Classication methods and approaches
- Overview of Machine and Deep Learning approaches



- Hands on a simple example project in Python on publicly available dataset
  - using Machine Learning approach,
  - using Deep Learning approach.

Presentation plan

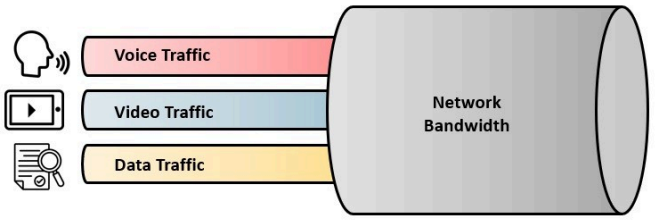
- Survey of Network Traffic Classication methods and approaches
- Overview of Machine and Deep Learning approaches



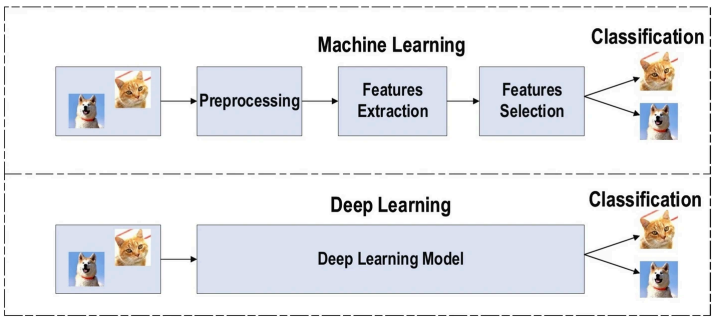
- Hands on a simple example project in Python on publicly available dataset
  - using Machine Learning approach,
  - using Deep Learning approach.
- Warning! Spoiler!

Presentation plan

- Survey of Network Traffic Classication methods and approaches
- Overview of Machine and Deep Learning approaches

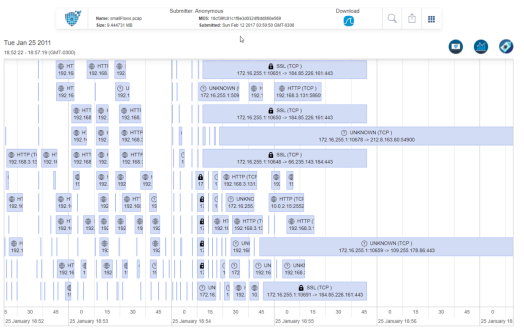


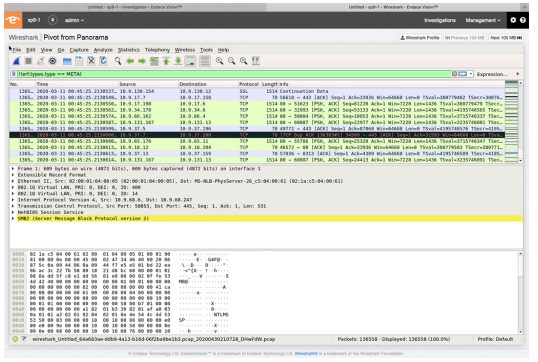
- Hands on a simple example project in Python on publicly available dataset
  - using Machine Learning approach,
  - using Deep Learning approach.
- Warning! Spoiler!

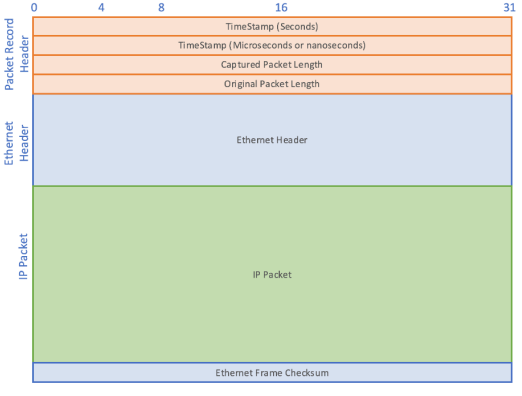


How the traffic look like?

-







Classical approaches and difficulties

https://www.sciencedirect.com/science/article/pii/S2352864822001845

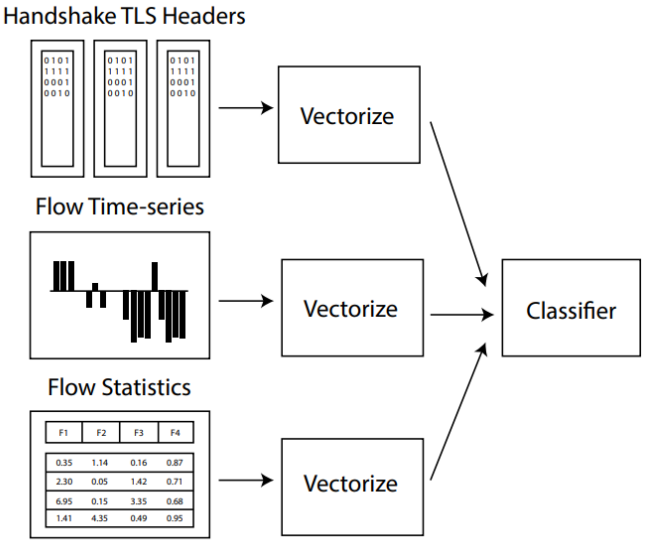
Classic approach

- Port - based identification,
- DPI - Deep Packet Inspection, avoids masquerading and randomization, string matching (regular expressions),

Port-based classification solution's advantages are reflected in its simplicity to implement the low computing resources requirements and the high-speed classification process. On the other hand, it has two main disadvantages. First, various applications nowadays deploy masquerading, that is, using standard ports to deliver other protocol traffic, such as malware traffic over HTTP. Second, many applications are deployed randomly, which refers to the utilization of non-standard/dynamic ports to deliver their network traffic, such as VoIP applications

DPI approach, sometimes referred to as signature-based identification, overcomes the shortcomings of the port-based classification approach since it does not require accessing port numbers. Therefore, it avoids masquerading and randomization [24]. DPI classifies and analyzes the payload of the network traffic to identify the causal applications. A signature is extracted from the packets' content, including characters, strings, bit pattern, and symbols that classify the applications. A signature library comprises of signature records, each of which is related to an application. This technique allows the classifier to examine the content of a single packet or aggregate packets, and checks them according to a signature library; if a match occurs, then an alert is raised, and the traffic is correlated to an application. This technique enhances the accuracy compared to the port-based identification, even if the application uses non-standard ports.

Encryption



With the upcoming encryption of service names, Deep Packet Inspection will no longer directly reveal the application.

The question that arises is therefore:

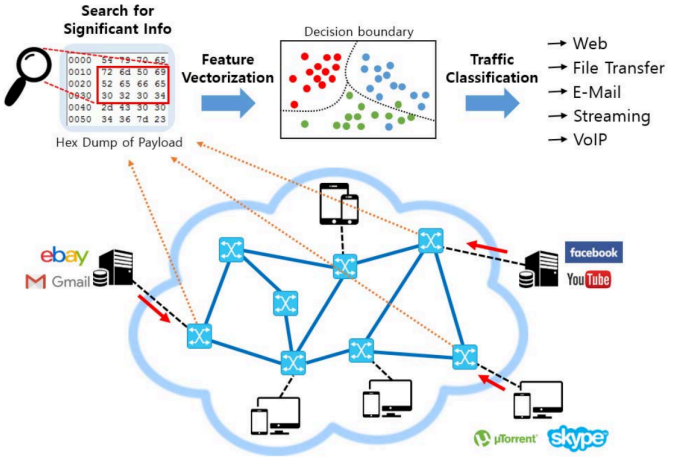
do the intrinsic characteristics of the encrypted flows constituting the network traffic allow to identify the applications or the category of applications?

The intrinsic characteristics of flows are the encrypted traffic properties that remain observable. These are primarily three time series:

- the packet sizes,
- inter-packet delays,
- traffic direction (user to server or server to user)
- and possibly some unencrypted information elements in the exchanged packets but also statistics summarizing the flow as its duration, throughputs and data volume per direction.

Finally, the information exchanged during the establishment of the encryption session is rich and can also help identifying the application.

Use unencrypted part of TLS protocol



https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=8805803

TLS protocol is layered on top of the reliable transport protocols, and it can support various application protocols such as HyperText Transfer Protocol (HTTP), Simple Mail Transfer Protocol (SMTP), and File Transfer Protocol (FTP). **Before the TLS protocol transmits its first byteof encrypted data, it performs a series of TLS handshake processes, in which the unencrypted contents are exposed.**

In this stage, the server and client authenticate each other; then, they negotiate an encryption algorithm and cryptographic keys. This information is included in initial packets of the handshaking process, which are well known as a client/server hello message. Thus, the **hello messages** have a great potential to be used for identifying which application generated the traffic flow, even the following packets comprise the encrypted data.

Bayesian Neural Network based Encrypted Traffic Classification using Initial Handshake Packets, Jiwon Yang, Jargalsaikhan Narantuya, and Hyuk Lim, 2019

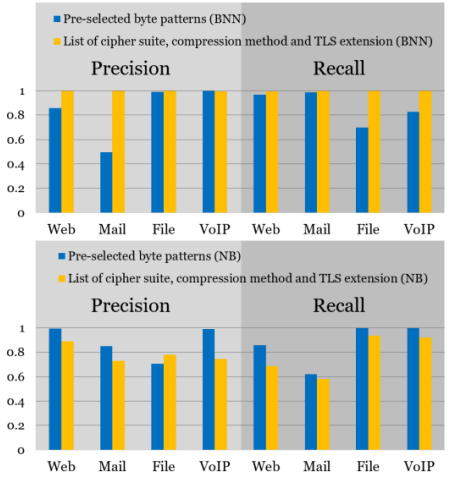
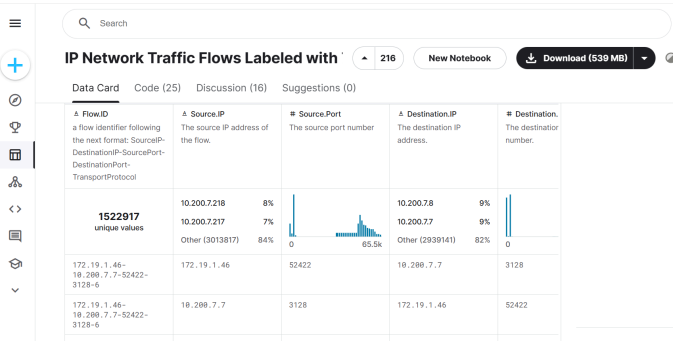


Fig. 2. Precision and recall results of the classification methods.

Encrypted traffic dataset example

https://www.kaggle.com/datasets/jsrojas/ip-network-traffic-flows-labeled-with-87-apps

- Context

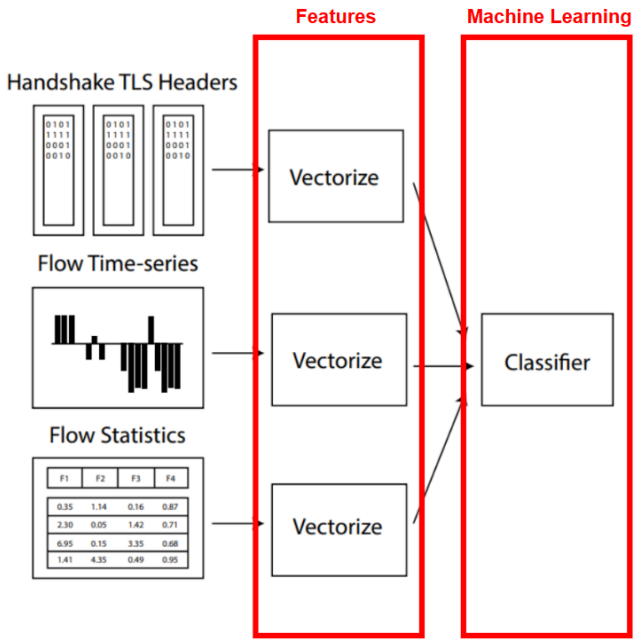


The data was collected in a network section from Universidad Del Cauca, Popayán, Colombia by performing packet captures at different hours, during morning and afternoon, over six days (April 26, 27, 28 and May 9, 11 and 15) of 2017. A total of 3.577.296 instances were collected and are currently stored in a CSV (Comma Separated Values) file.

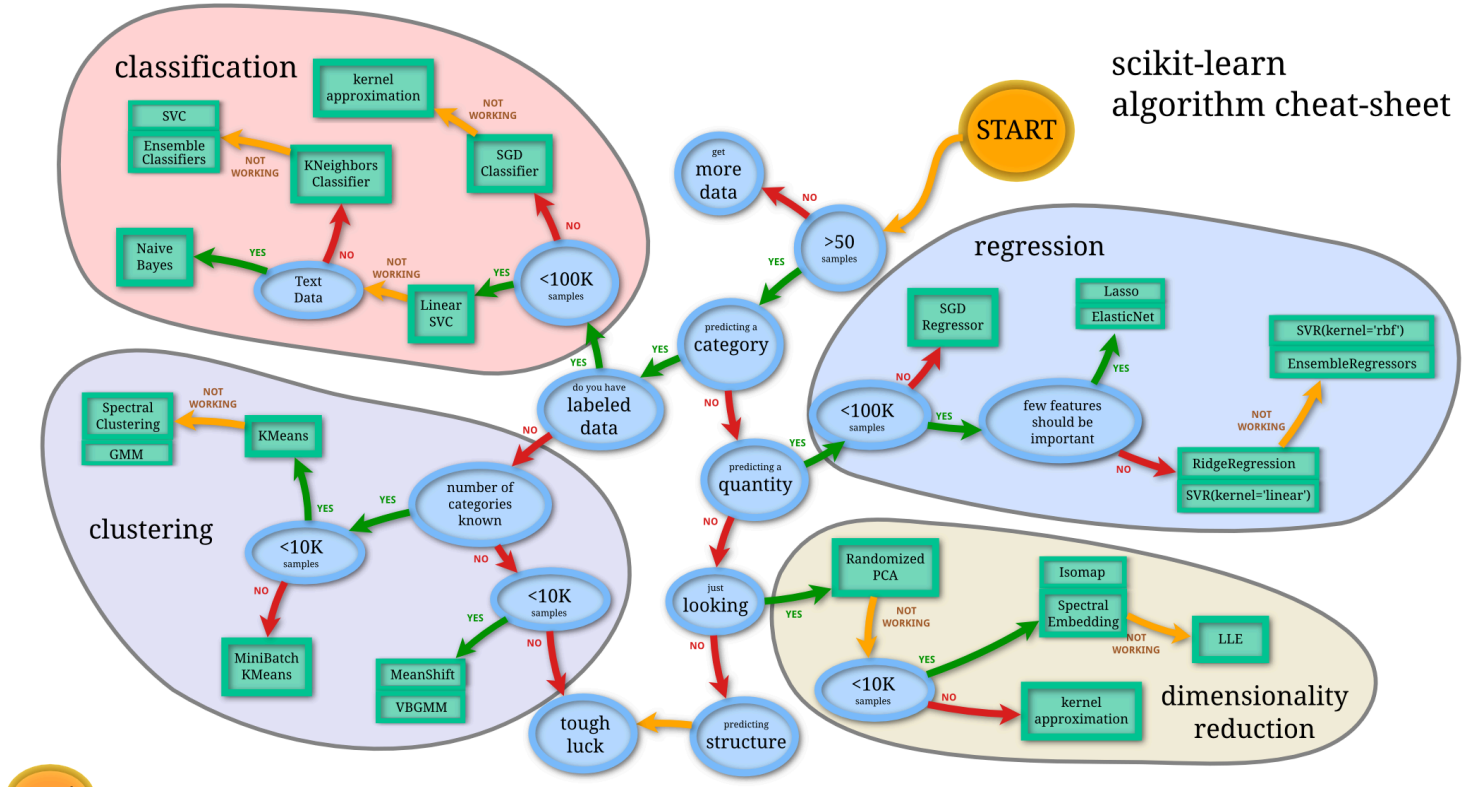
- Content

This dataset contains 87 features. Each instance holds the information of an IP flow generated by a network device i.e., source and destination IP addresses, ports, interarrival times, layer 7 protocol (application) used on that flow as the class, among others. Most of the attributes are numeric type but there are also nominal types and a date type due to the Timestamp.

Having the features we can approach the Machine Learning



'Classical' Machine Learning Guide map



Back

Hands on tutorial and show case for NTC

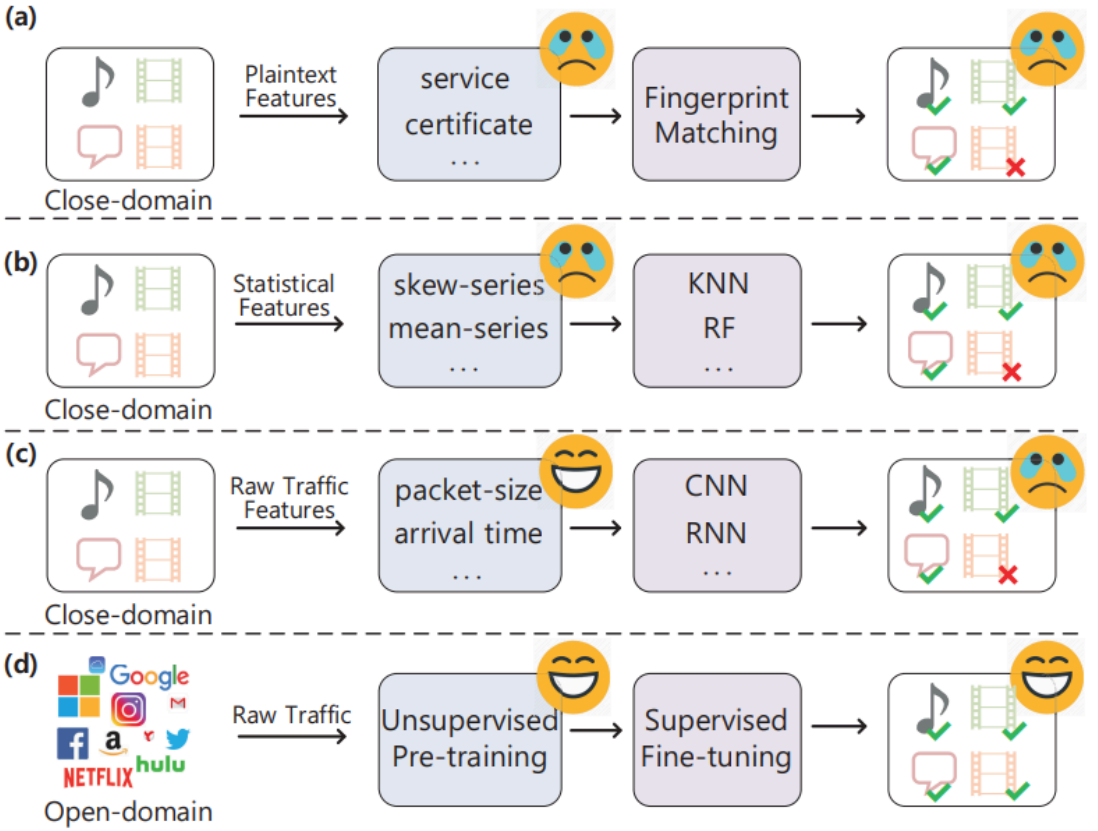
Classical NTC - header and statistical data.

Task: https://colab.research.google.com/drive/1g0fRcq-sdzYgxu6cEYxbG2vVEmAI2iai?usp=sharing

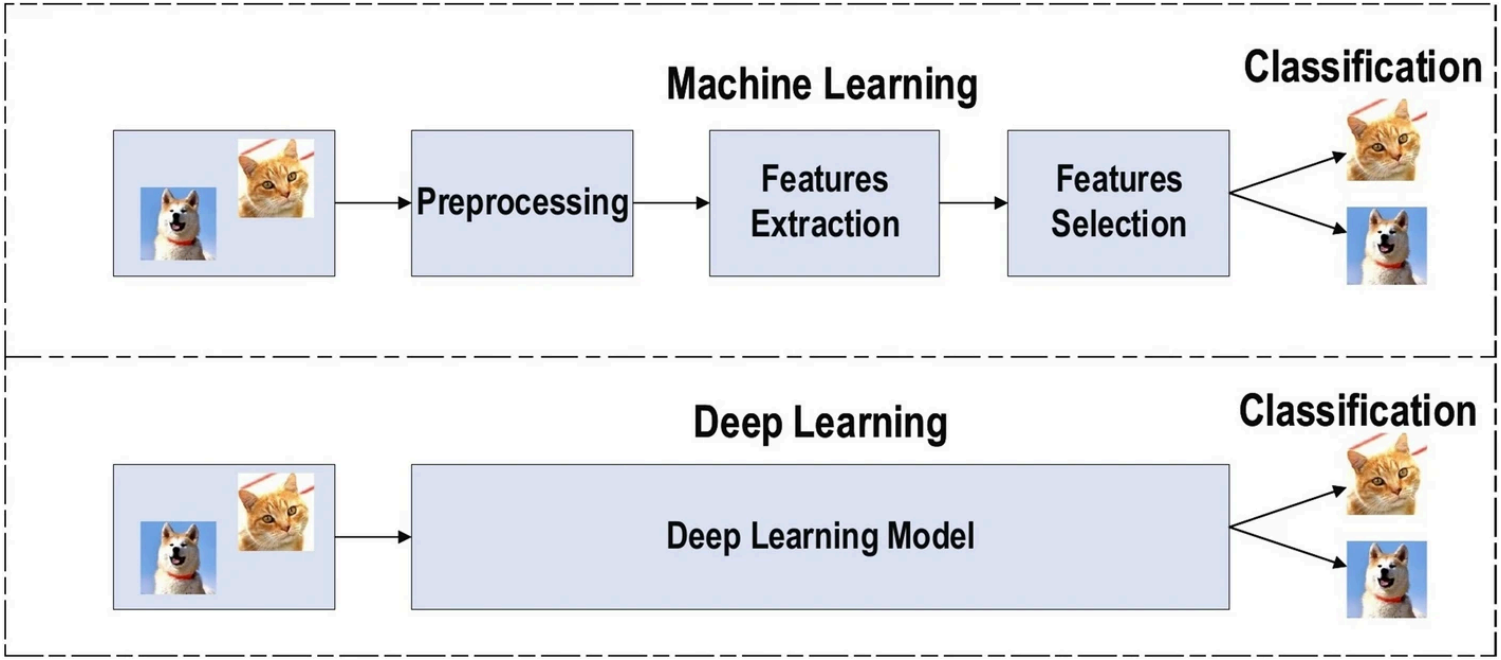
Flow.ID	a flow identifier following the next format: SourceIP-DestinationIP-SourcePort-DestinationPort-TransportProtocol
Source.IP	The source IP address of the flow.
Source.Port	The source port number
Destination.IP	The destination IP address.
Destination.Port	The destination port number.
Protocol	The transport layer protocol number identification (i.e.,TCP = 6, UDP = 17).
Timestamp	The instant the packet was captured stored in the next date format: Dd/mm/yyyy HH:MM:SS
Flow.Duration	The total duration of the flow
Total.Fwd.Packets	The total number of packets in the forward direction.
Bwd.Avg.Bytes.Bulk	Average number of bytes bulk rate in the backward direction.
Bwd.Avg.Packets.Bulk	Average number of packets bulk rate in the backward direction.
Bwd.Avg.Bulk.Rate	Average number of bulk rate in the backward direction.
Subflow.Fwd.Packets	The average number of packets in a subflow in the forward direction. The core idea of multipath TCP is to define a way to build a connection between two
Subflow.Fwd.Bytes	The average number of bytes in a subflow in the forward direction.
Subflow.Bwd.Packets	The average number of packets in a subflow in the backward direction.
Subflow.Bwd.Bytes	The average number of bytes in a subflow in the backward direction.
Init_Win_bytes_forward	The total number of bytes sent in the initial window in the forward direction. TCP uses a sliding window flow control protocol. In each TCP segment, the re
Init_Win_bytes_backward	The total number of bytes sent in the initial window in the backward direction.
act_data_pkt_fwd	Count of packets with at least one byte of TCP data payload in the forward direction.
min_seg_size_forward	Minimum segment size observed in the forward direction.
Active.Mean	The mean time a flow was active before becoming idle.
Active.Std	Standard deviation time a flow was active before becoming idle.
Active.Max	Maximum time a flow was active before becoming idle.
Active.Min	Minimum time a flow was active before becoming idle.
Idle.Mean	Mean time a flow was idle before idle before becoming active.
Idle.Std	Standard deviation time a flow was idle before becoming active.
Idle.Max	The maximum time a flow was idle before becoming active.
Idle.Min	The minimum time a flow was idle before becoming active.
Label	The state of the flow (benign or malign).
L7Protocol	This attribute represents the code number of the layer 7 protocol as obtained from nDPI in Ntopng application. It is a number that varies from 0 to 226 (e.g
ProtocolName	This attribute is the objective class of the dataset. It holds the application name following the code

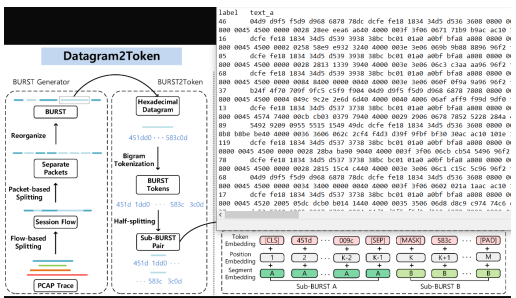
With solutions: https://colab.research.google.com/drive/1K5ON-f6e_W_QBh5swyMX1cDbgX2aqlWA?usp=sharing

Encrypted traffic approaches overview - deep networks



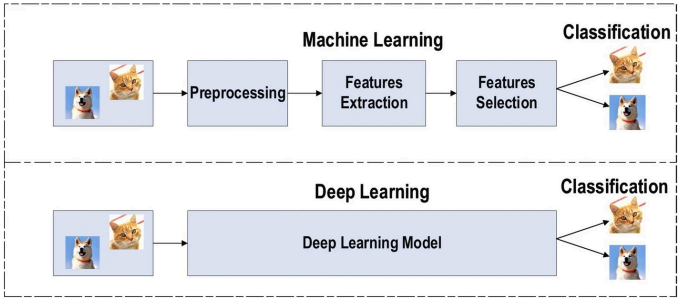
Deep learning approach to encrypted traffic classification



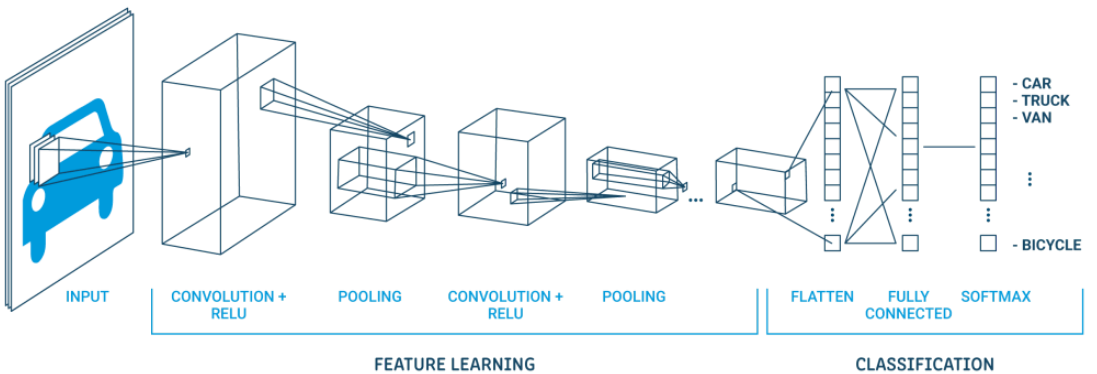


The input in Deep Learning is not a vector of engineered features, but streams interpreted as "images" in form of hexadecimal tokens.

Deep networks Machine learning technologies



- Types of deep networks:
  - deep neural networks
  - convolutional deep networks
  - r-cnn networks
  - lstm networks
  - transformer networks (attention)
  - ViT - Vision Transformers
  - GAN (generative adversarial networks - generative networks)

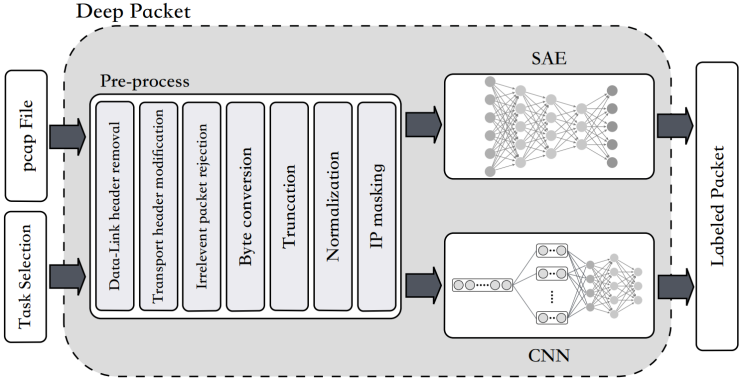


https://arxiv.org/abs/1709.02656

Deep Packet: A Novel Approach For Encrypted Traffic Classification Using Deep Learning, 2017

Mohammad Lotfollahi et al.

"Deep Packet," can handle both traffic characterization in which the network traffic is categorized into major classes (eg., FTP and P2P) and application identification in which end-user applications (eg., BitTorrent and Skype) identification is desired. After an initial pre-processing phase on data, packets are fed into Deep Packet framework that embeds stacked autoencoder and convolution neural network in order to classify network traffic. Deep packet with CNN as its classification model achieved recall of 0.98 in application identification task and 0.94 in traffic categorization task. Tested on UNB ISCX VPN-nonVPN dataset.



https://arxiv.org/abs/1709.02656

Deep packet: architectures

The general structure of an autoencoder.

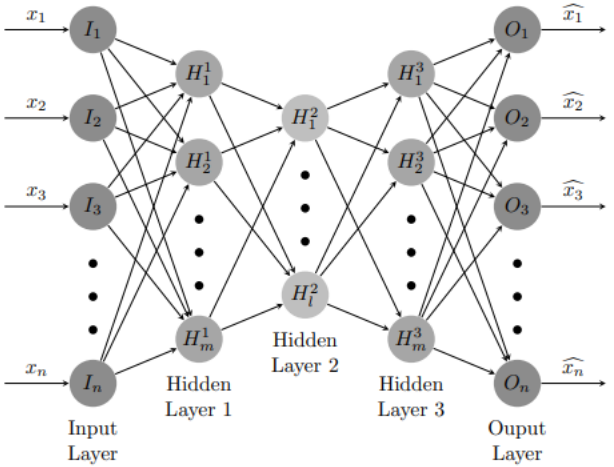
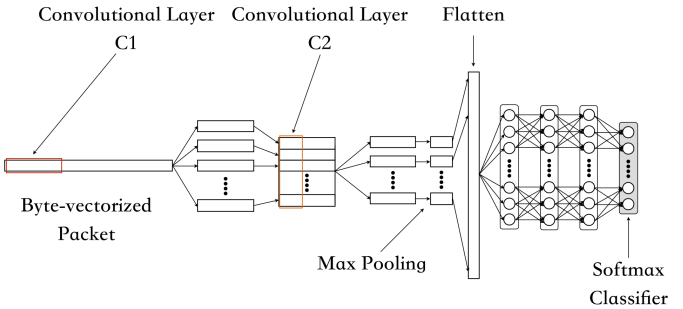


Illustration of the onedimensional CNN architecture



https://arxiv.org/abs/1709.02656
https://www.unb.ca/cic/datasets/vpn.html

Deep packet: testcase data characteristic

UNB ISCX VPN-nonVPN dataset

Application	Size
AIM chat	5K
Email	28K
Facebook	2502K
FTPS	7872K
Gmail	12K
Hangouts	3766K
ICQ	7K
Netflix	299K
SCP	448K
SFTP	418K
Skype	2872K
Spotify	40K
Torrent	70K
Tor	202K
Voipbuster	842K
Vimeo	146K
YouTube	251K

(a)

Class Name	Size
Chat	82K
Email	28K
File Transfer	210K
Streaming	1139K
Torrent	70K
VoIP	5120K
VPN: Chat	50K
VPN: File Transfer	251K
VPN: Email	13K
VPN: Streaming	479K
VPN: Torrent	269K
VPN: VoIP	753K

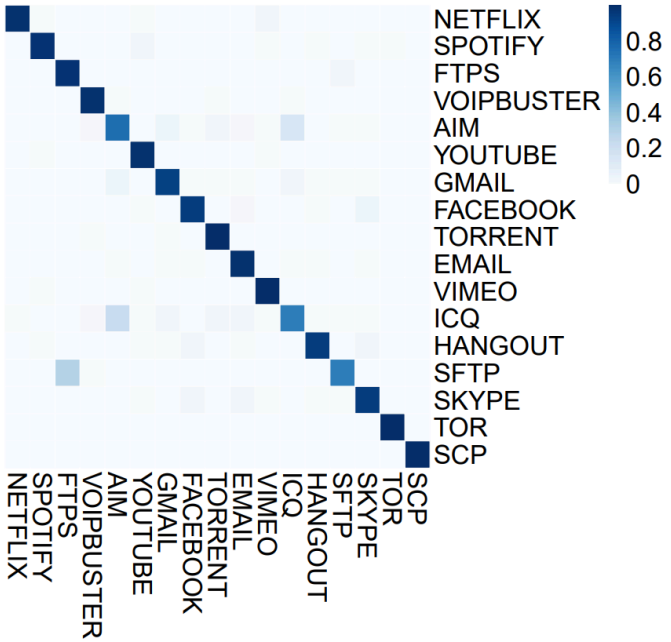
(b)

Table 1: Number of samples in each class for (a) Application identification, and (b) Traffic characterization.

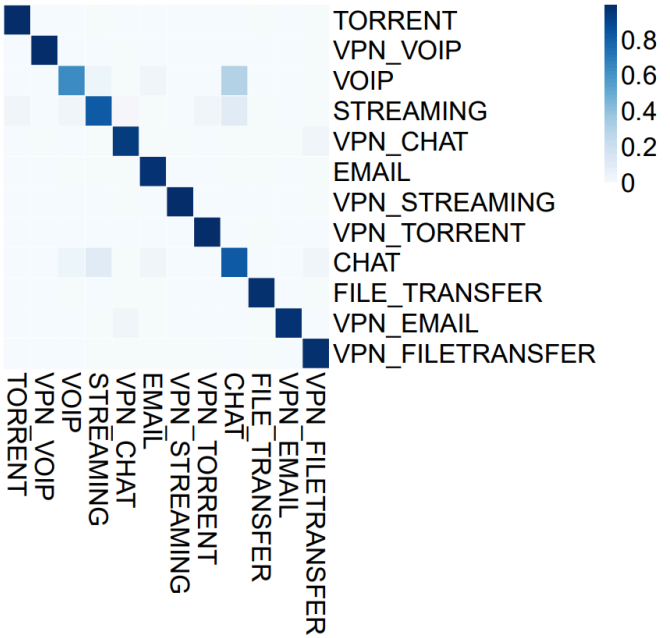
https://arxiv.org/abs/1709.02656

Deep packet: results

Application identification results



Traffic characterization results



https://arxiv.org/abs/1709.02656

What about encription?

As the authors of the Deep Packet paper say:

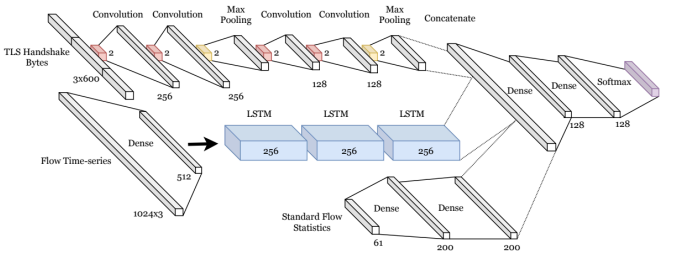
The majority of "ISCX VPN-nonVPN" dataset traffics are also encrypted. One might wonder how it is possible for Deep Packet to classify such encrypted traffics. Unlike DPI methods, Deep Packet does not inspect the packets for keywords. In contrast, it attempts to learn features in traffic generated by each application. Consequently, it does not need to decrypt the packets to classify them.

An ideal encryption scheme causes the output message to bear the ***maximum possible entropy****. In other words, it produces patternless data that theoretically cannot be distinguished from one another. However, due to the fact that all practical encryption schemes use* ***pseudo-random generators****, this hypothesis is not valid in practice. Moreover, each application employs* ***different (non-ideal) ciphering scheme*** *for data encryption. These schemes utilize different pseudo-random generator algorithms which leads to* ***distinguishable patterns****. Such variations in the pattern can be used to separate applications from one another. Deep Packet attempts to extract those discriminative patterns and learns them. Hence, it can classify encrypted traffic accurately.*

https://hellofuture.orange.com/en/encrypting-or-classifying-internet-traffic-do-we-have-to-choose/
https://dl.acm.org/doi/pdf/10.1145/3447382

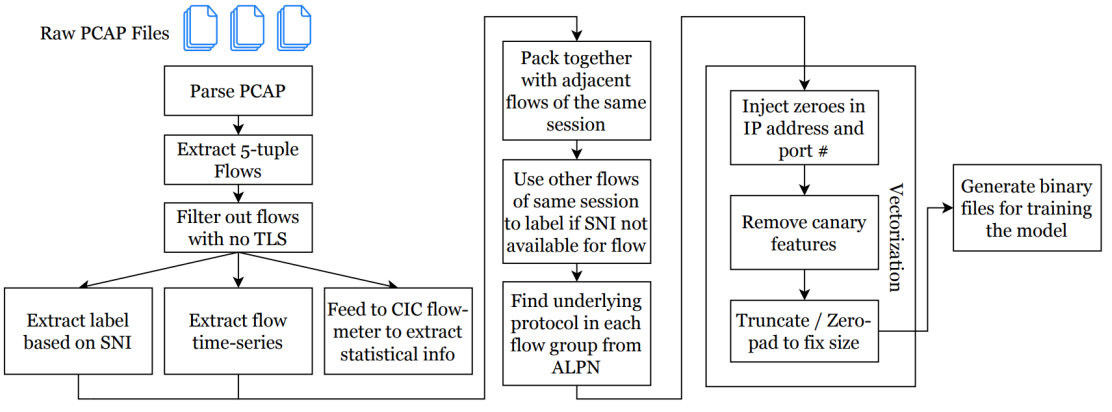
Example use case employing a Deep Learning

"A Look Behind the Curtain: Traffic Classification in an Increasingly Encrypted Web", *IMAN AKBARI, MOHAMMAD A. SALAHUDDIN, LENI VEN, NOURA LIMAM, and RAOUF BOUTABA, University of Waterloo, Canada BERTRAND MATHIEU, STEPHANIE MOTEAU, and STEPHANE TUFFIN*, Orange Labs, France, 2021



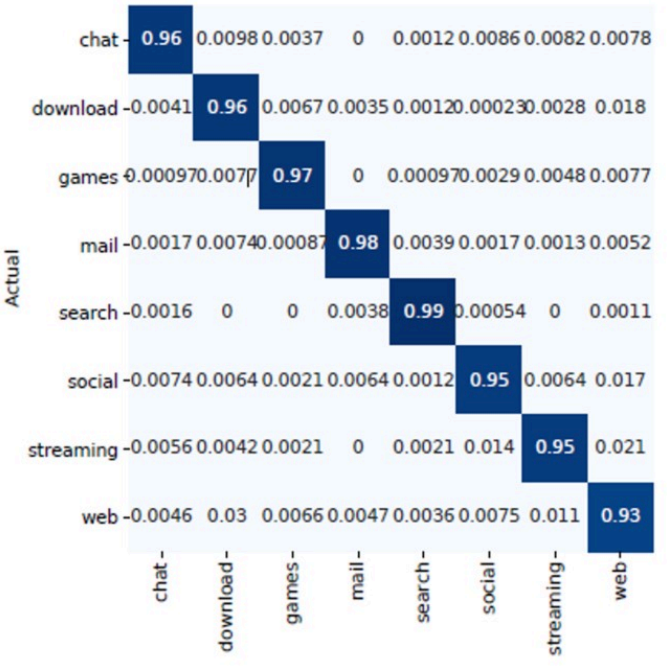
Orange, in collaboration with the University of Waterloo, in Canada, has developed an AI (Artificial Intelligence) model, composed of convolutional neural networks and long and short-term memory recurrent networks, permitting to classify the categories of applications (web browsing, video streaming, chat, etc.) with a success rate of 96% and applications (Facebook, Gmail, Netflix, etc.) with a success rate of 97%. The tests were carried out to classify into 8 categories of applications (see Figure 4) and 19 applications.

The Orange-Waterloo solution - overview of the pre-processing steps



The Orange-Waterloo solution is Hybrid model

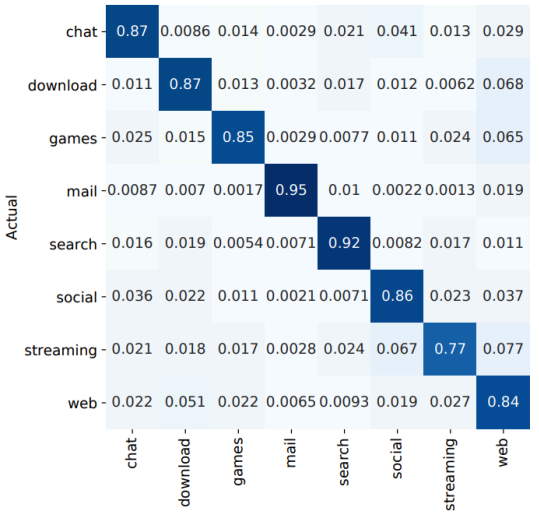
It offers superior performance over other state-of-the-art solutions, a decision tree-based solution widely used for classification, which only ensures a success rate of 81% or other CNN UCDavis solution, which has a detection rate of 91%. CNN UCDavis presents a detection quite close to Orange-Waterloo, however, it exhibits many false positives (misclassification) as Orange-Waterloo reduces this false-positive rate by 50% compared to CNN UCDavis.



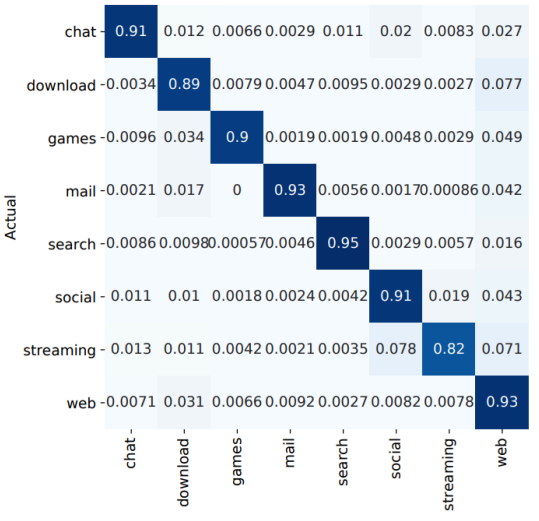
With the Orange-Waterloo solution, the precision (proportion of correct predictions among all predictions made) and recall (proportion of positive predictions for samples that are really positive) values are between 90% and 100% depending on the categories. The f1-score rate, a combination of precision and recall values, has an overall average of around 94%, demonstrating the good quality of the model (see Figure 5).

https://dl.acm.org/doi/pdf/10.1145/3447382

The Orange-Waterloo hybrid solution compared to UCDavis CNN



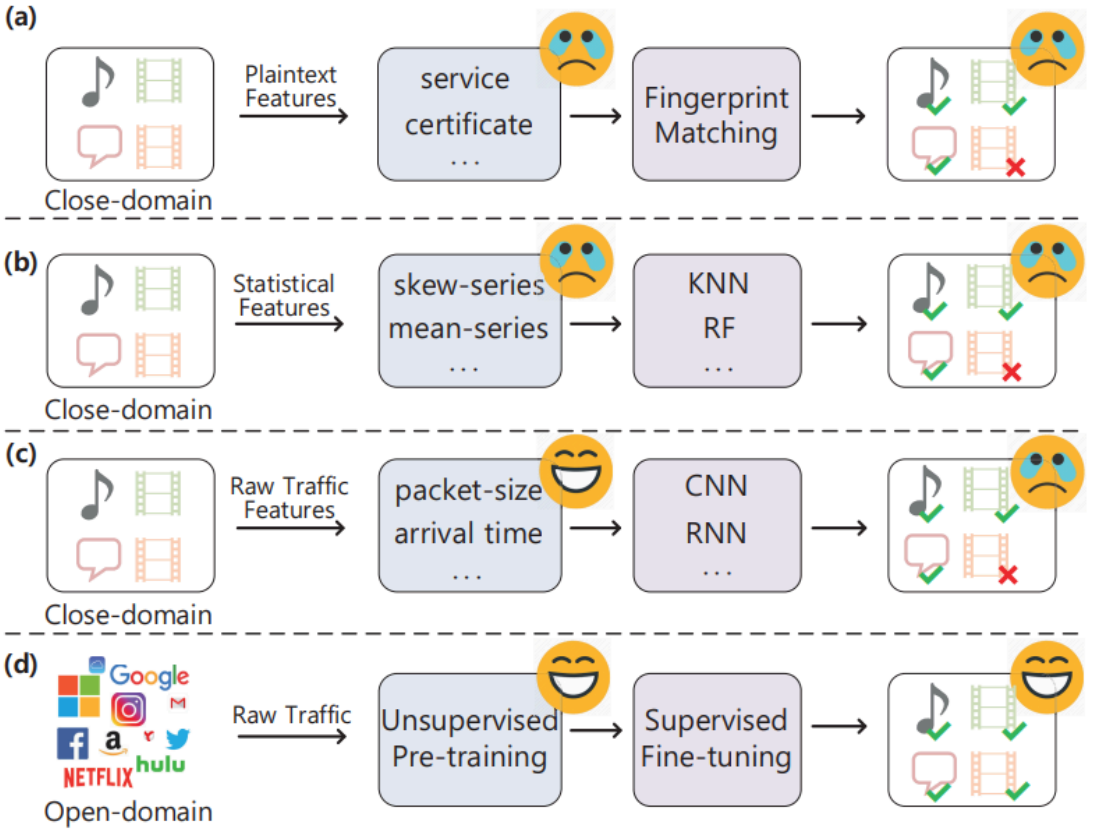
(a) Flow-only Stacked LSTM



(b) UCDavis CNN

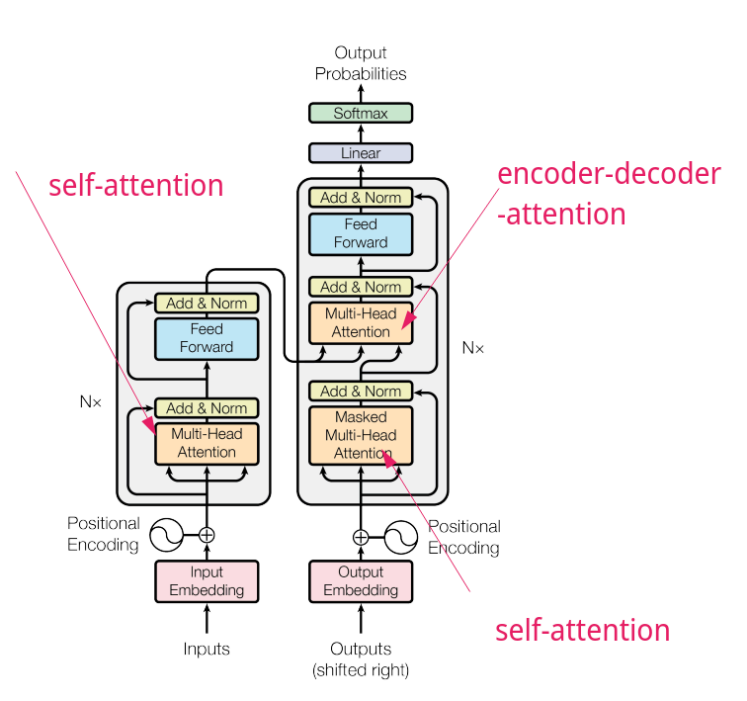
Although being inferior in performance to models that also include raw traffic as input, it is important to note that the flow time-series features will always be available regardless of how the encrypted protocols evolve.

Encrypted traffic approaches overview - generative networks



Attention is all you need

Transformer generative networks



Transformers - generative AI

https://bbycroft.net/llm

Transformer models are a type of deep learning model that is used for natural language processing (NLP) tasks. They can learn long-range dependencies between words in a sentence, which makes them very powerful for tasks such as machine translation, text summarization, and question answering.

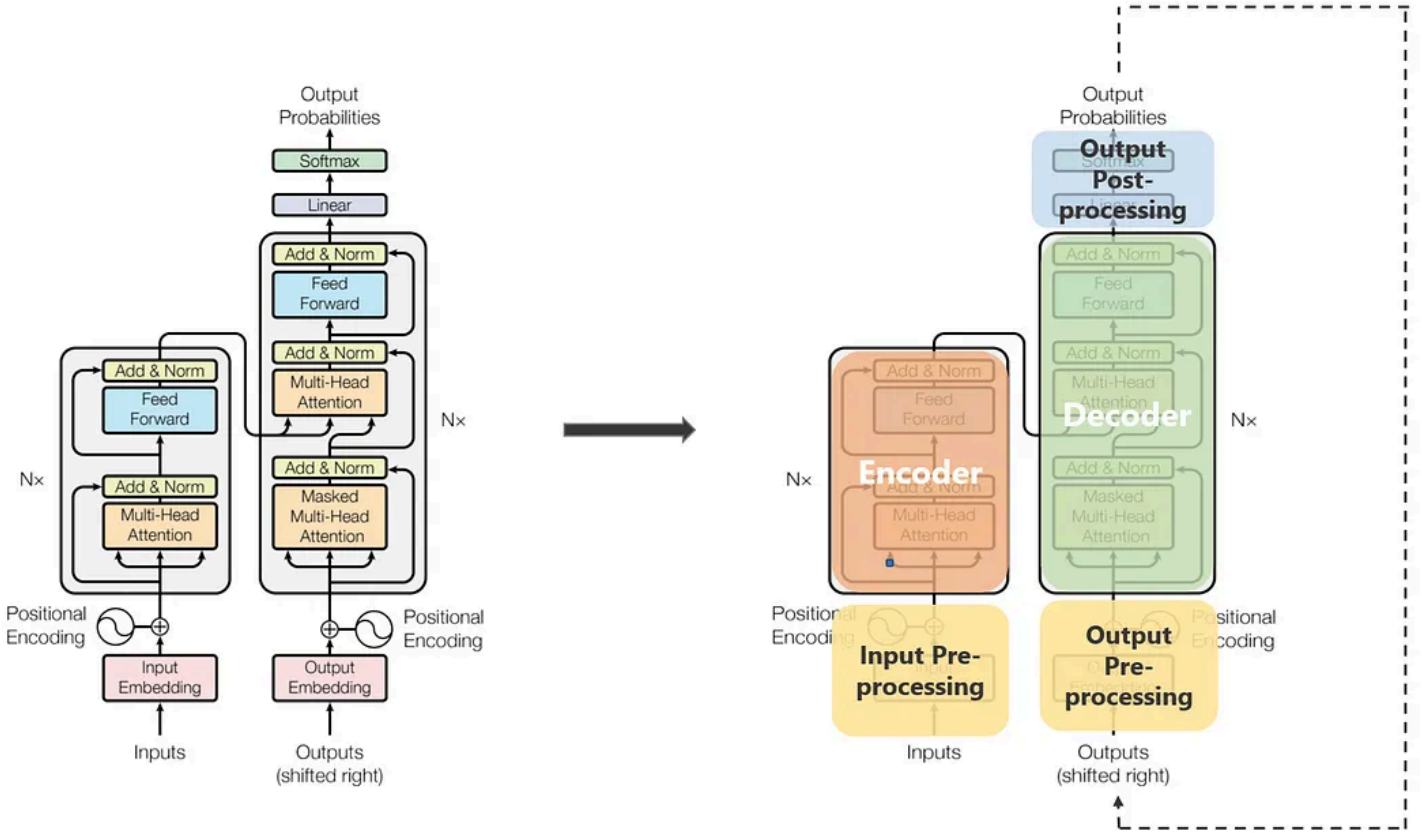
Generative AI, what it means?

It means that the main purpose of a Large Language Model (LLM, Transformer), when it is presented a text (a prompt) to predict with highest possible probability the next word it should be.

"The cats does not like to play with ?...?"

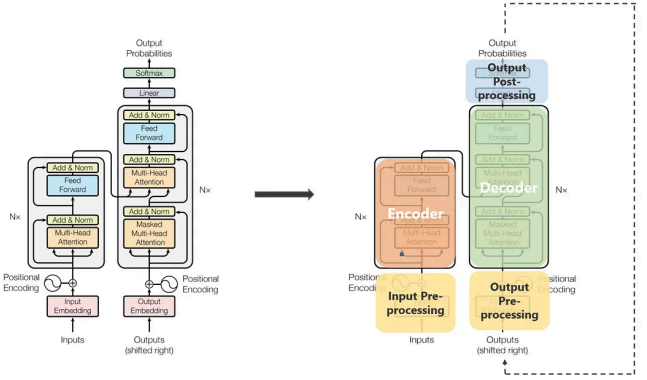
"The cats like to play with ?...?"

Transformer architecture



Transformer key concepts:

1. Self attention, sometimes called intra-attention is an attention mechanism relating different positions of a single sequence in order to compute a representation of the sequence.




```
“I poured water from the bottle into the cup until it was full.”
it => cup
“I poured water from the bottle into the cup until it was empty.”
it=> bottle
```

By changing one word “full” — > “empty” the reference object for “it” changed. If we are translating such a sentence, we will want to know the word “it” refers to.

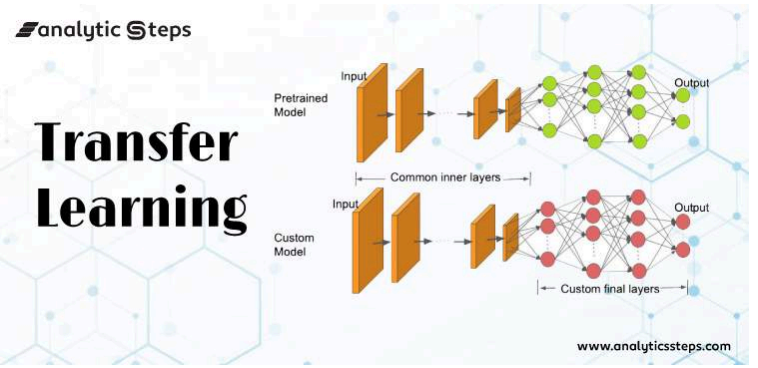
1. Using outputed text as input for the decoder. The next word dependeds on what I have said so far. This is why Chat GPT can create long and meaningfull texts.

The main problem - insanely expensive training

Training a transformer requires huge amount of data, time and energy. Only big companies, countries, national institutions can afford tranining a transfomer model.

Mostly the technique called fine-tuning (or pretrained model, or transfer learning).

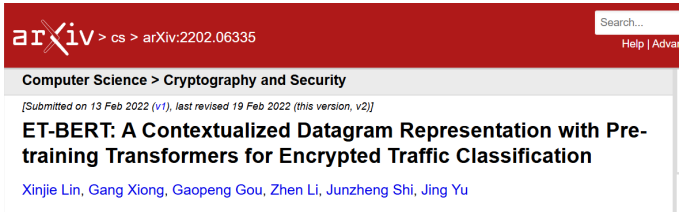
Classical transfer learning (only classification layers are trained)



https://arxiv.org/abs/2202.06335

Transformers network usecase ET-BERT

ET-BERT: A Contextualized Datagram Representation with Pre-training Transformers for Encrypted Traffic Classification, Xinjie Lin, et al., 2022



*In this paper, we propose a new traffic representation model called **Encrypted Traffic Bidirectional Encoder Representations from Transformer (ET-BERT)**, which pre-trains deep contextualized datagram-level representation from large-scale unlabeled data. The pre-trained model can be fine-tuned on a small number of task-specific labeled data and achieves state-of-the-art performance across five encrypted traffic classification tasks, remarkably pushing the F1 of ISCX-Tor to 99.2% (4.4% absolute improvement), ISCX-VPN-Service to 98.9% (5.2% absolute improvement), Cross-Platform (Android) to 92.5% (5.4% absolute improvement), CSTNET-TLS 1.3 to 97.4% (10.0% absolute improvement).*

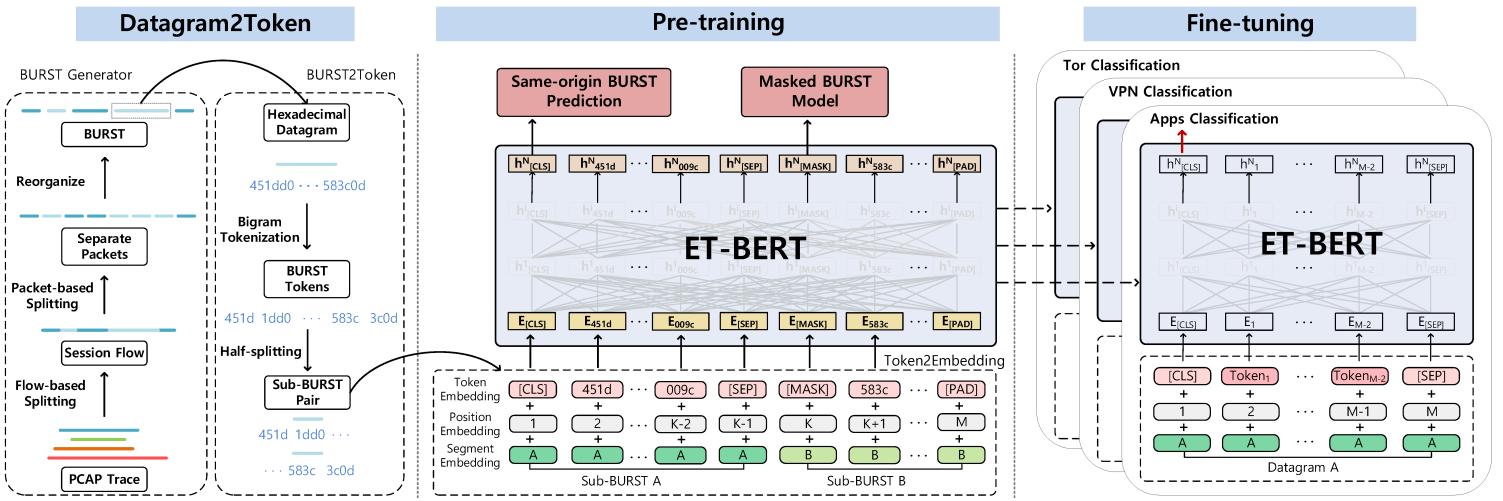
The goals:

- Construct a general model for encrypted traffic classification (no need for feature engineering, no need for large labeled traffic data)
- Generalizability and transferability (consider encrepted data commonality, effective transition to different scenarios)

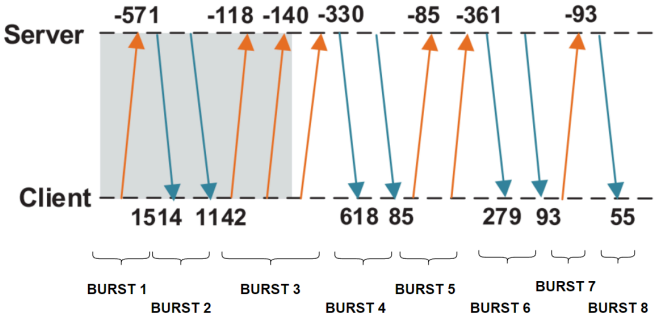
Transformers network ET-BERT

https://github.com/linwhitehat/ET-BERT
https://arxiv.org/abs/2202.06335

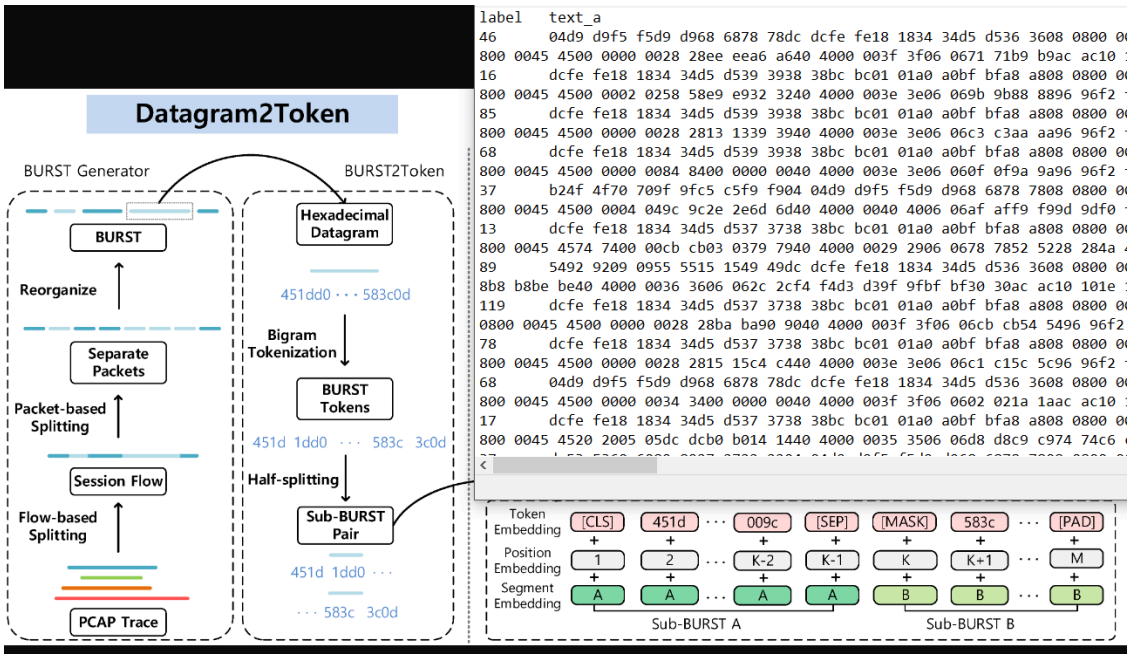
ET-BERT is a method for learning datagram contextual relationships from encrypted traffic, which could be directly applied to different encrypted traffic scenarios and accurately identify classes of traffic. First, ET-BERT employs multi-layer attention in large scale unlabelled traffic to learn both inter-datagram contextual and inter-traffic transport relationships. Second, ET-BERT could be applied to a specific scenario to identify traffic types by fine-tuning the labeled encrypted traffic on a small scale.



ET-BERT input data



- A BURST is defined as a set of time-adjacent network packets originated from either the request or the response in a single session flow.
- The authors use a bi-gram to encode the hexadecimal sequence, where each unit consists of two adjacent bytes.
- In addition, the authors also add the special tokens `[CLS], [SEP], [PAD]` and `[MASK]` for training tasks



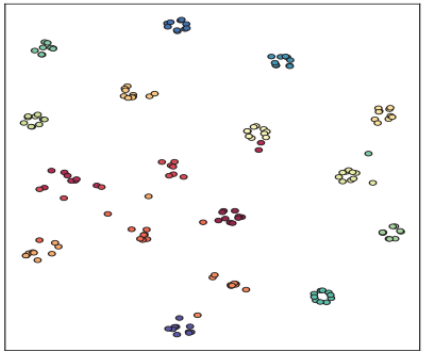
Comparison Results on Cross-Platform, ISCX-VPN-Service and ISCX-VPN-App datasets

Dataset	Cross-Platform(iOS)				Cross-Platform(Android)				ISCX-VPN-Service				ISCX-VPN-App			
Method	AC	PR	RC	F1	AC	PR	RC	F1	AC	PR	RC	F1	AC	PR	RC	F1
AppScanner[36]	0.3205	0.2103	0.2173	0.2030	0.3868	0.2523	0.2594	0.2440	0.7182	0.7339	0.7225	0.7197	0.6266	0.4864	0.5198	0.4935
CUMUL[27]	0.2910	0.1917	0.2081	0.1875	0.3525	0.2221	0.2409	0.2189	0.5610	0.5883	0.5676	0.5668	0.5365	0.4129	0.4535	0.4236
BIND[1]	0.3770	0.2566	0.2715	0.2484	0.4728	0.3126	0.3253	0.3026	0.7534	0.7583	0.7488	0.7420	0.6767	0.5152	0.5153	0.4965
K-fp[12]	0.2155	0.2037	0.2069	0.2003	0.2248	0.2113	0.2104	0.2052	0.6430	0.6492	0.6417	0.6395	0.6070	0.5478	0.5430	0.5303
FlowPrint[37]	0.9254	0.9438	0.9254	0.9260	0.8698	0.9007	0.8698	0.8702	0.7962	0.8042	0.7812	0.7820	0.8767	0.6697	0.6651	0.6531
DF[35]	0.3106	0.2232	0.2179	0.2140	0.3862	0.2595	0.2620	0.2527	0.7154	0.7192	0.7104	0.7102	0.6116	0.5706	0.4752	0.4799
FS-Net[22]	0.3712	0.2845	0.2754	0.2655	0.4846	0.3544	0.3365	0.3343	0.7205	0.7502	0.7238	0.7131	0.6647	0.4819	0.4848	0.4737
GraphDApp[33]	0.3245	0.2450	0.2392	0.2297	0.4031	0.2842	0.2786	0.2703	0.5977	0.6045	0.6220	0.6036	0.6328	0.5900	0.5472	0.5558
TSCRNN[21]	-	-	-	-	-	-	-	-	-	0.9270	0.9260	0.9260	-	-	-	-
Deeppacket[25]	0.9204	0.8963	0.8872	0.9034	0.8805	0.8004	0.7567	0.8138	0.9329	0.9377	0.9306	0.9321	0.9758	0.9785	0.9745	0.9765
PERT[13]	0.9789	0.9621	0.9611	0.9584	0.9772	0.8628	0.8591	0.8550	0.9352	0.9400	0.9349	0.9368	0.8229	0.7092	0.7173	0.6992
ET-BERT(flow)	**0.9844**	0.9701	0.9632	0.9643	**0.9865**	0.9324	**0.9266**	**0.9246**	0.9729	0.9756	0.9731	0.9733	0.8519	0.7508	0.7294	0.7306
ET-BERT(packet)	0.9810	**0.9757**	**0.9772**	**0.9754**	0.9728	**0.9439**	0.9119	0.9206	**0.9890**	**0.9891**	**0.9890**	**0.9890**	**0.9962**	**0.9936**	**0.9938**	**0.9937**

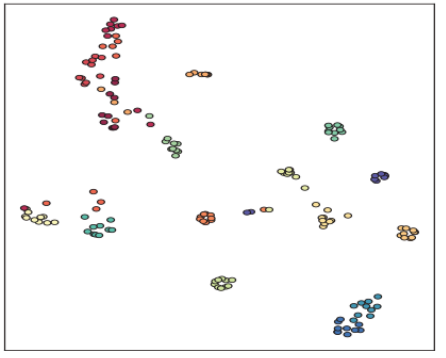
Table 3: Comparison Results on ISCX-Tor, USTC-TFC and CSTNET-TLS 1.3 datasets.

Dataset	ISCX-Tor				USTC-TFC				CSTNET-TLS 1.3			
Method	AC	PR	RC	F1	AC	PR	RC	F1	AC	PR	RC	F1
AppScanner[36]	0.6722	0.3756	0.4422	0.3913	0.8954	0.8984	0.8968	0.8892	0.6662	0.6246	0.6327	0.6201
CUMUL[27]	0.6606	0.3850	0.4416	0.3918	0.5675	0.6171	0.5738	0.5513	0.5391	0.4942	0.5060	0.4904
BIND[1]	0.7185	0.4598	0.4515	0.4511	0.8457	0.8681	0.8382	0.8396	0.7964	0.7605	0.7650	0.7560
K-fp[12]	0.6472	0.5576	0.5849	0.5522	-	-	-	-	0.4036	0.3969	0.4044	0.3902
FlowPrint[37]	0.9092	0.3820	0.3661	0.3654	0.8146	0.6434	0.7002	0.6573	0.1261	0.1354	0.1272	0.1116
DF[35]	0.7533	0.6228	0.6010	0.5850	0.7787	0.7883	0.7819	0.7593	0.7936	0.7721	0.7573	0.7602
FS-Net[22]	0.6071	0.5080	0.5350	0.4590	0.8846	0.8846	0.8920	0.8840	0.8639	0.8404	0.8349	0.8322
GraphDApp[33]	0.6836	0.4864	0.4823	0.4488	0.8789	0.8226	0.8260	0.8234	0.7034	0.6464	0.6510	0.6440
TSCRNN[21]	-	0.9490	0.9480	0.9480	-	0.9870	0.9860	0.9870	-	-	-	-
Deeppacket[25]	0.7449	0.7549	0.7399	0.7473	0.9640	0.9650	0.9631	0.9641	0.8019	0.4315	0.2689	0.4022
PERT[13]	0.7682	0.4424	0.4446	0.4345	0.9909	0.9911	0.9910	0.9911	0.8915	0.8846	0.8719	0.8741
ET-BERT(flow)	0.8311	0.5564	0.6448	0.5886	**0.9929**	**0.9930**	**0.9930**	**0.9930**	0.9510	0.9460	0.9419	0.9426
ET-BERT(packet)	**0.9921**	**0.9923**	**0.9921**	**0.9921**	0.9915	0.9915	0.9916	0.9916	**0.9737**	**0.9742**	**0.9742**	**0.9741**

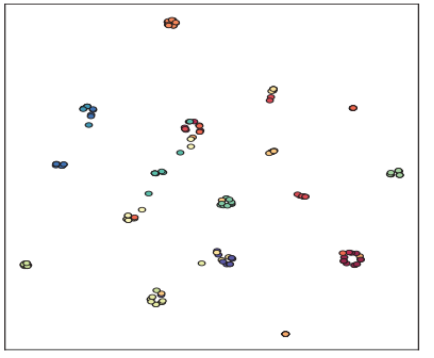
t-SNE Visualization of Classification Boundaries with 6 Methods on ISCX-VPN-App Testset



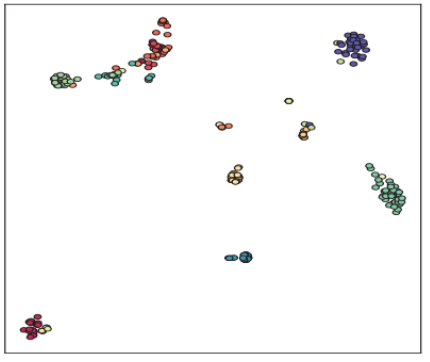
(a) Representation with ET-BERT



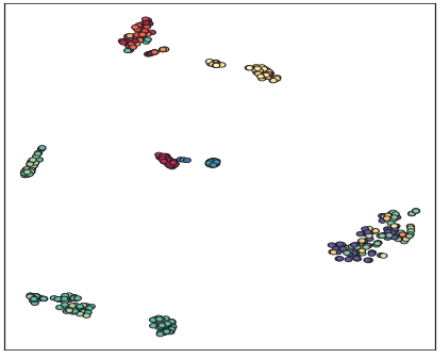
(b) Representation with Transformer at packet



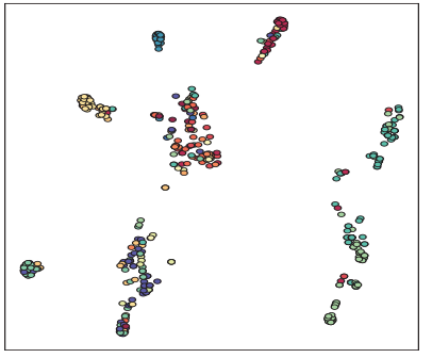
(c) Representation with Deeppacket



(e) Representation with PERT



(f) Representation with Transformer at flow



(g) Representation with DF

ET-BERT usecase showcase

Demo: https://colab.research.google.com/drive/1zhx8twGA6Q9Mpk8vskIEj0kY2PH6GBWI?usp=sharing

Prediction only: https://colab.research.google.com/drive/1G2JahHVZ9sVX-DNfL5RW44icw6KchcPd?usp=sharing

Thank you

https://github.com/szmurlor/cpee-ntc-2024/