Machine learning in NTC

Network Traffic Classification

Robert Szmurło

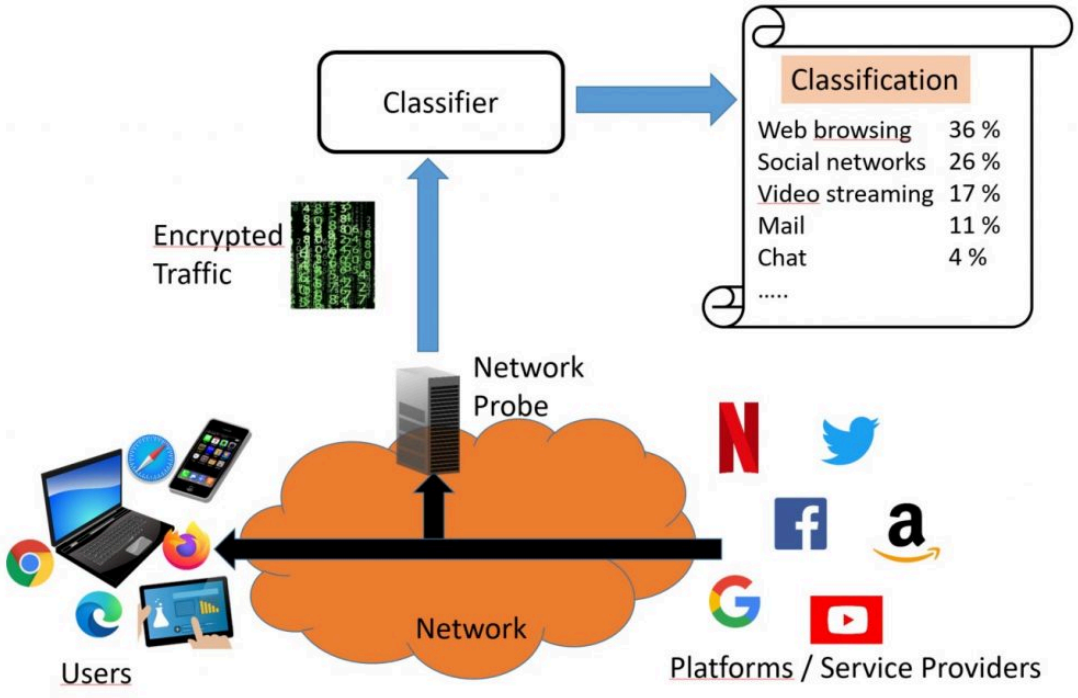
e-mail: robert.szmurlo@pw.edu.pl

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

Tutorial motivation

After completing the tutorial you will have a general information and understand the basic concept of using Machine Learning in Network Traffic Classication.

You will learn the basics related to NTC and how AI can help.



Tutorial plan

- General introduction to machine learning
- Survey of on Network Traffic Classication methods and approaches
- Hands on a simple example project in Python on publicly available dataset

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).

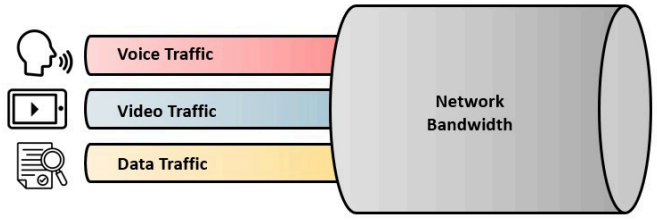
1. **Network Security**: Identifying and mitigating malicious traffic and intrusions.
2. **Quality of Service (QoS) Management**: Prioritizing critical applications for better performance.
3. **Bandwidth Management**: Controlling and optimizing bandwidth usage to prevent congestion.
4. **Content Filtering 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.

More examples

https://networksimulationtools.com/machine-learning-in-networking-projects/

- Node Localization – K-NN, Reinforcement Learning (RL) – Estimate distance accurately, calculate the range
- Node Coverage and Connectivity – Decision trees, ANN, Evolutionary Computation– Node classification in a sensor network (connected or failed nodes, identify nodes with poor or good connectivity)
- Routing Layer Issues – Decision Tree, Random Forest – predict optimal routing paths depending on the lead data traffic.
- MAC Layer Issues – SVM, Decision Tree, ANN– Channel assignment that is efficient
- Sensor data aggregation – K-means, SVM, Reinforcement – determine optimal cluster heads in WSN nodes, set up the dynamic configuration for WSNs.
- Event Monitoring and Target Detection – PCA, Deep Learning, Evolutionary Computing, Bayesian Learning – track multiple targets with efficient event monitoring
- Energy Harvesting – SVM, Deep Learning, Evolutionary computing – Predicting how much battery power is required for optimum network lifetime, to predict the availability of energy harvest in the future.
- Node Query Processing – kNN – Beacons on nodes, data transfer by handshake.

Lets pick a topic: Network Traffic Classification



To operate networks, Internet access providers classify traffic by category of application and by service providers. This classification is impacted by service name encryption. How could this practice evolve?

Traffic classification has been studied for two decades and applied to a wide range of applications from QoS provisioning and billing in ISPs to security-related applications in firewalls and intrusion detection systems

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.

Classifying traffic, what for?

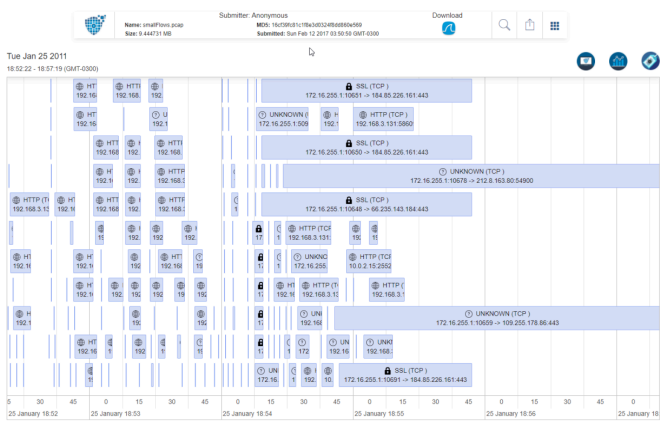
Classification allows to know the market shares of applications or categories of applications. Usage analysis by customer segment and by offer in turn allows to bring more relevant offers to the market

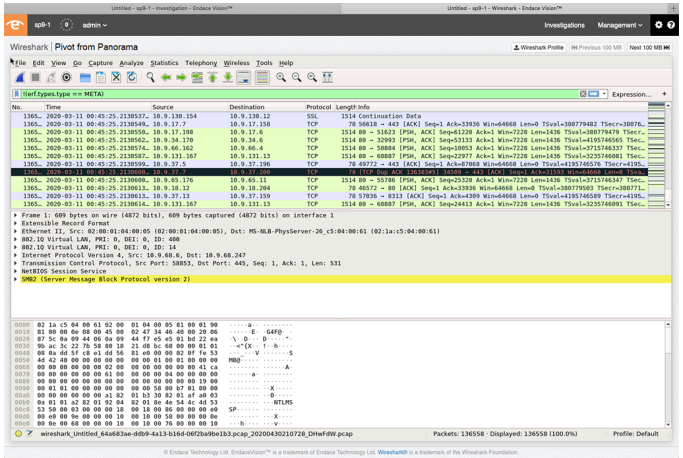
Traffic classification also allows to measure the quality of service by application or category of applications. This allows to predict traffic evolutions in order to adapt network capacities and offer the best possible quality to users.

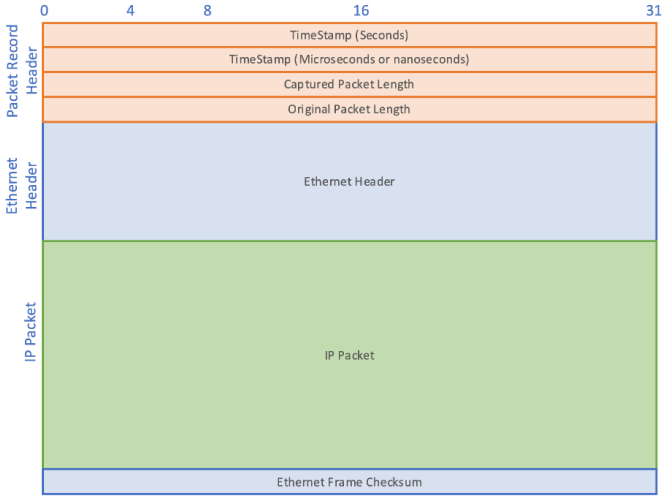
For example, traffic classification helped anticipate Netflix's move to 4K. It also helps preparing the impact of a football match on data rates, depending on the broadcaster.

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

How the traffic look like?

- 



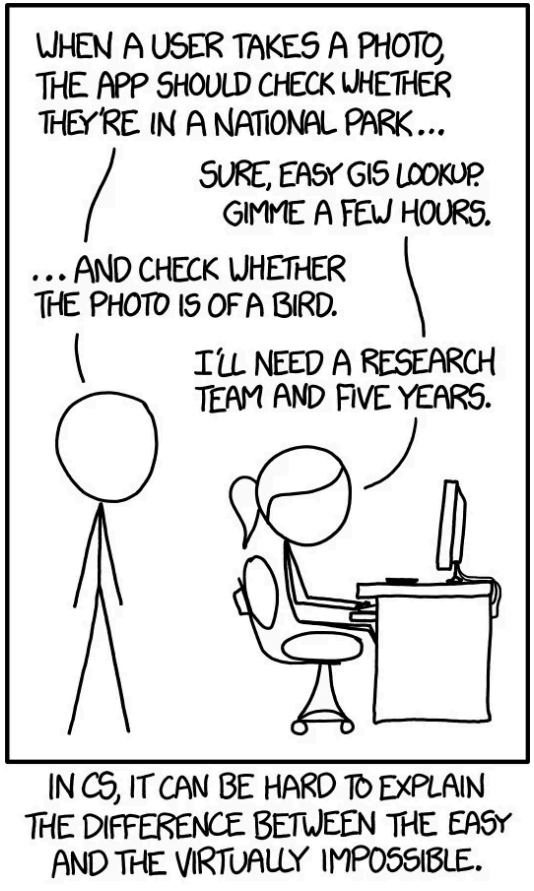


Classical approaches and difficulties

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),

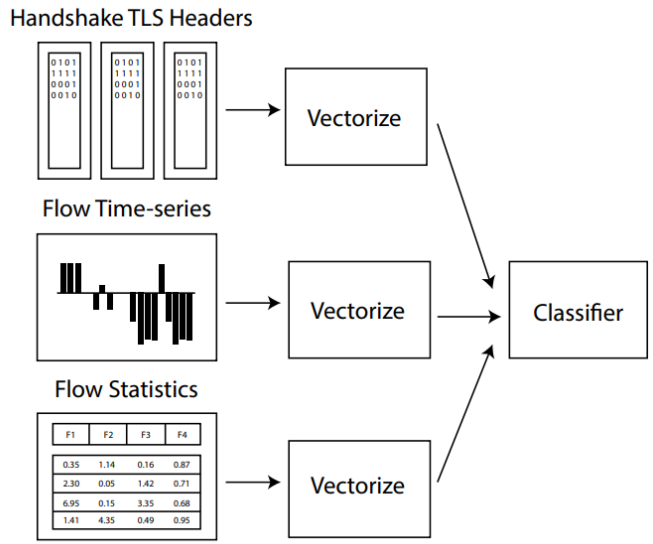


Port-based classification solution's advantages are reflected in its simplicity to implement the low computing resources requirements and the high-speed classification process. On the other hand, it has two main disadvantages. First, various applications nowadays deploy masquerading, that is, using standard 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.

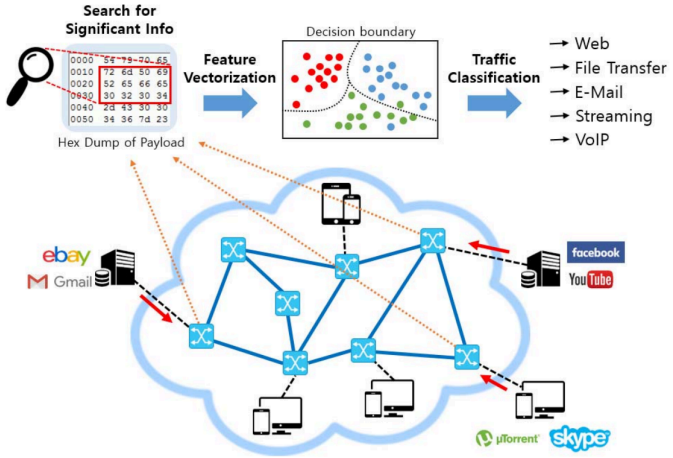
Encryption

With the upcoming encryption of service names, deep 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

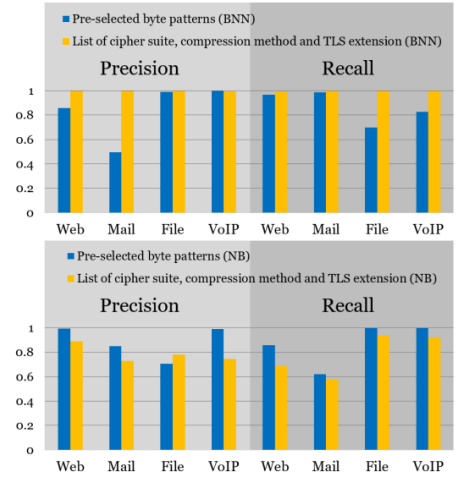
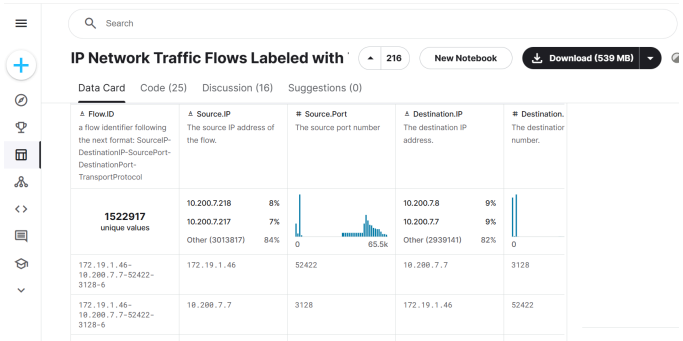


Fig. 2. Precision and recall results of the classification methods.

Encrypted traffic dataset example

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

- Context



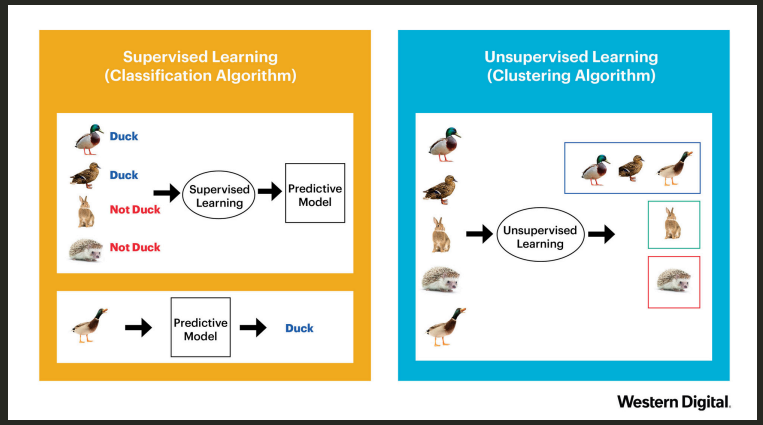
The data was collected in a network section from Universidad Del Cauca, Popayán, Colombia by performing packet captures at different hours, during morning and afternoon, over six days (April 26, 27, 28 and May 9, 11 and 15) of 2017. A total of 3.577.296 instances were collected and are currently stored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

Machine Learning

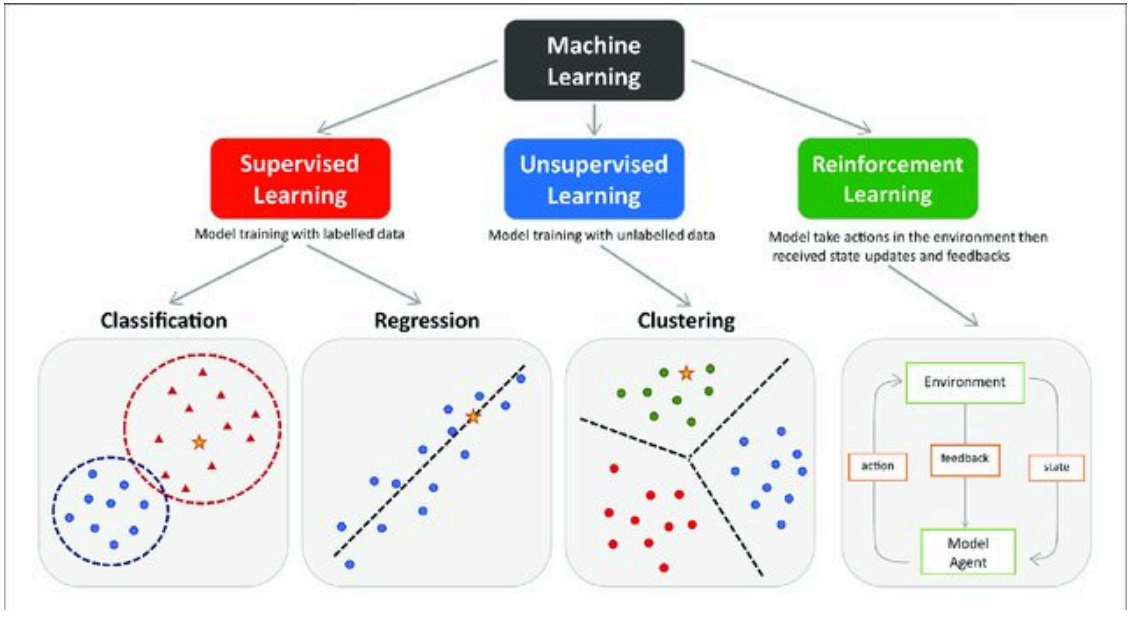
Machine learning-based traffic statistical classification,



Machine learning usage scenarios

In general AI used for the following problems:

- **classification** (recognition, identification, detection),
- **regression** (prediction, generation of content - GPT),
- **clusterization** (grouping, identifying trends, indentifying categories, differences),
- **reinforcement learning** (motion planning, decision making, systems control, anomaly detection)



What is a Machine Learning?

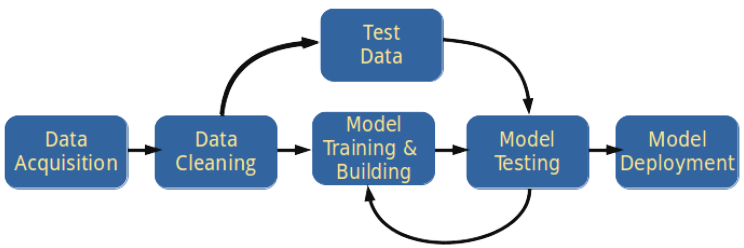
https://en.wikipedia.org/wiki/Machine_learning

Machine learning (ML) gives computer systems the ability to "learn" (e.g., progressively improve performance on a specific task) from data by using statistical techniques, without being explicitly programmed to make arbitrary decision based on some prior experience of a programmer or an expert.

ML uses algorithms that can learn from and make predictions on data – such algorithms overcome following strictly static program instructions by making data-driven predictions or decisions

ML process:

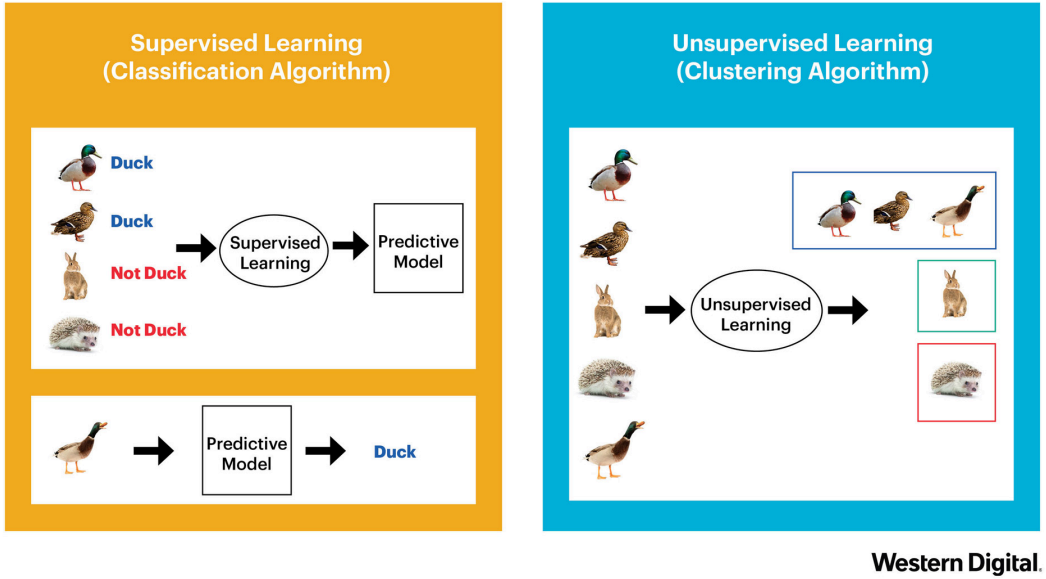
- Train and Test iteratively changing parameters to tune the model to test data set



3 types of machine learning

https://blog.westerndigital.com/machine-learning-pipeline-object-storage/supervised-learning-diagram/

- Labeled data - **supervised learning**
- Unlabelled dataset - **unsupervised learning**
- Semi-labeled dataset - **semi-supervised learning**



Supervised learning

https://machinelearningmastery.com/supervised-and-unsupervised-machine-learning-algorithms/

Supervised learning is where you have input variables ($X = [x_0, x_1, \ldots, x_n]$) and an output variable (Y) and you use an algorithm to learn the mapping function from the input to the output - building a model.

$$Y = f(X)$$

The goal is to **approximate** the mapping function so well that when you have a new input data (X) that you can predict the output variables (Y) for that data - generalization.

Recap:

- labeled data - predict a label based on a set of features,
- trained on labeled examples (expected output is known),
- its goal is to infer the natural structure present within a set of data points.

Supervised learning - types

https://towardsdatascience.com/supervised-vs-unsupervised-learning-14f68e32ea8d

Supervised learning problems can be further grouped into regression and classification problems.

- **Classification**: A classification problem is when the output variable is a category, such as “red” or “blue” or “disease” and “no disease”.
- **Regression**: A regression problem is when the output variable is a real value, such as “dollars” or “weight”.

Some common types of problems built on top of classification and regression include recommendation and time series prediction respectively.

Some popular examples of supervised machine learning algorithms are:

- Linear regression for regression problems.
- Random forest for classification and regression problems.
- Support vector machines for classification problems.

Unsupervised Machine Learning

Unsupervised learning is where you only have input data $(X=[x_1,\ldots,x_n])$ and no corresponding output variables.

The goal for unsupervised learning is to model the underlying structure or distribution in the data in order to learn more about the data.

These are called unsupervised learning because unlike supervised learning above there is no correct answers and there is no teacher. Algorithms are left to their own devises to discover and present the interesting structure in the data.

Unsupervised learning problems can be further grouped into clustering and association problems.

- **Clustering**: A clustering problem is where you want to discover the inherent groupings in the data, such as grouping customers by purchasing behavior.
- **Association**: An association rule learning problem is where you want to discover rules that describe large portions of your data, such as people that buy X also tend to buy Y.

Some popular examples of unsupervised learning algorithms are:

- k-means for clustering problems.
- Apriori algorithm for association rule learning problems.

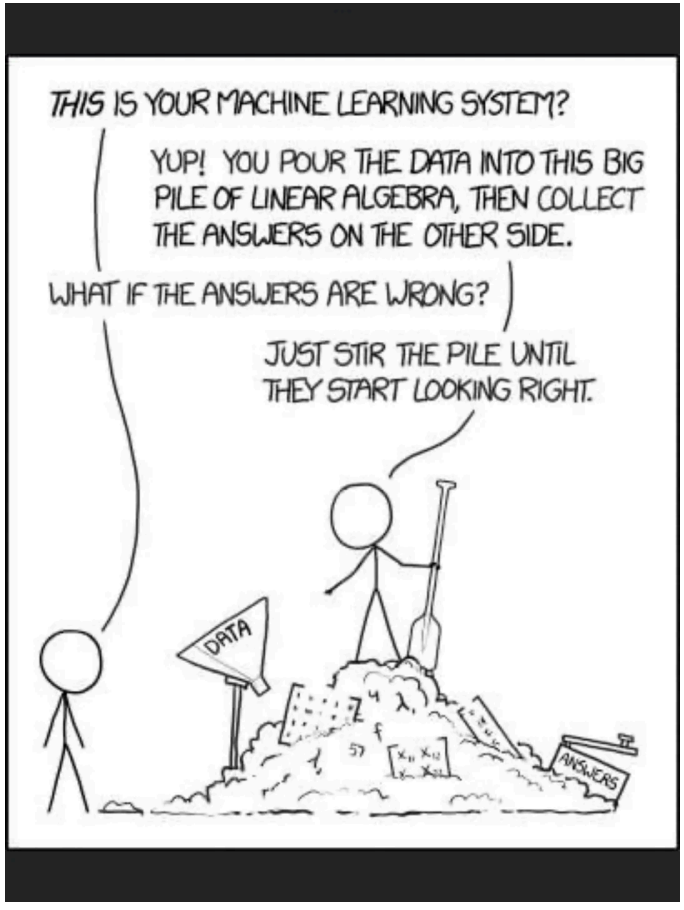
Recap:

- the system is trained on unlabeled data - it has to determine
- robotics, gaming and navigation

Three main issues of ML

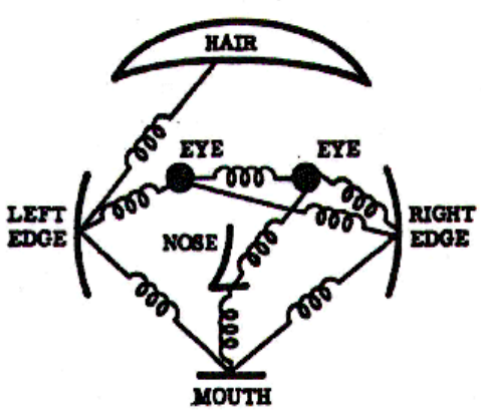
http://people.csail.mit.edu/torralba/shortCourseRLOC/slides/introduction.ppt

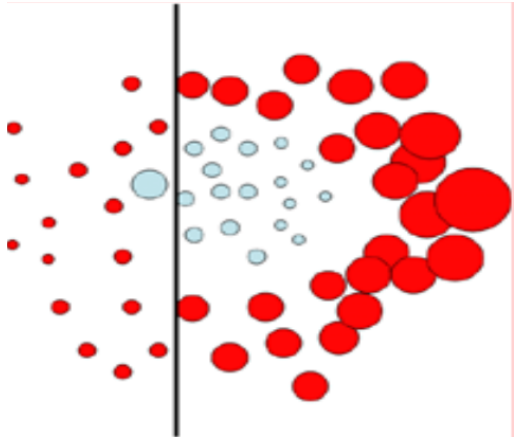
- Representation
  - How to represent an object category
- Learning
  - How to form the classifier, given training data
- Generalization
  - How the classifier is to be used on novel data



Representation 1

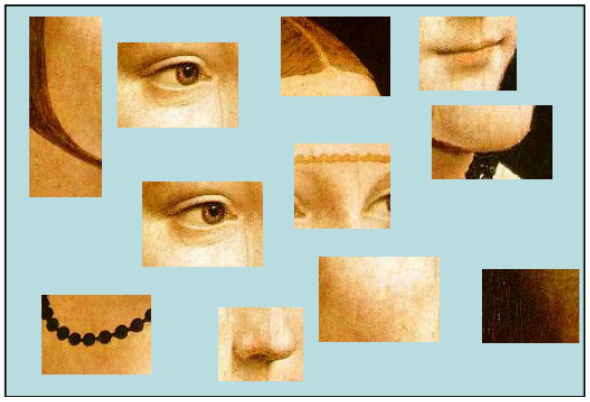
- Generative (creative) / discriminative / hybrid

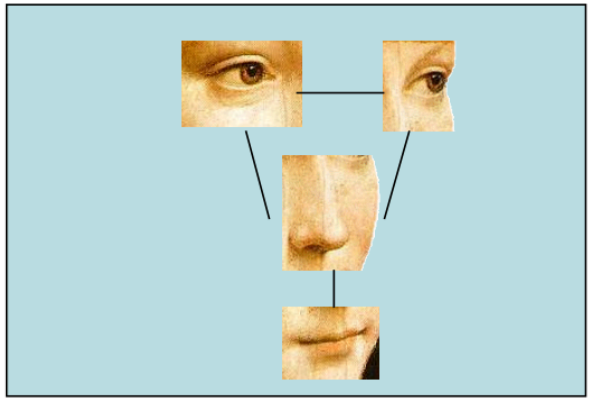




Representation 2

- Generative (creative) / discriminative / hybrid
- Account only form and lookout or also position and layout in some graph of dependencies





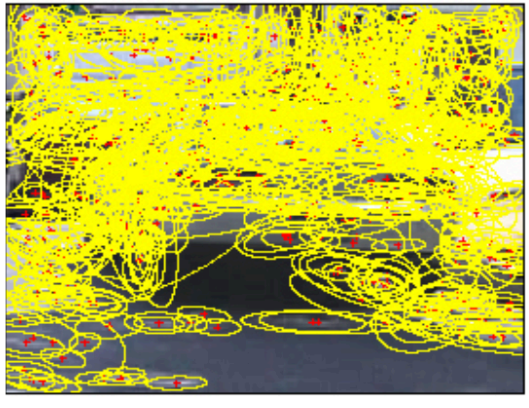
GPT-2, GPT-3, GPT-4 Models introduced attention mechanism, which allows to train the relations between word in a sentence, earlier the text was mostly treated as an **ordered** sequence. The attention is bidirectional.

Representation 3

- Generative (creative) / discriminative / hybrid
- Account only form and lookout or also position and layout in some graph of dependencies
- Invariance (representation should be solid and compare objects from: different perspectives, light, scale, deformation, noise)

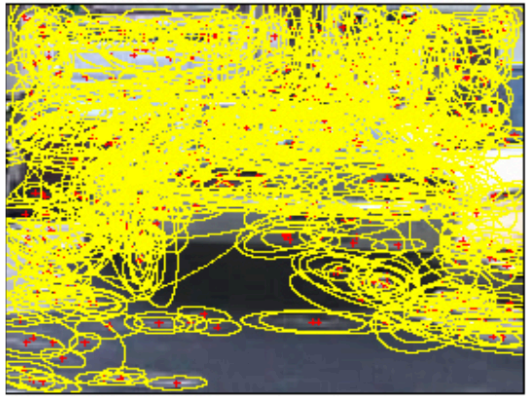
Representation 4

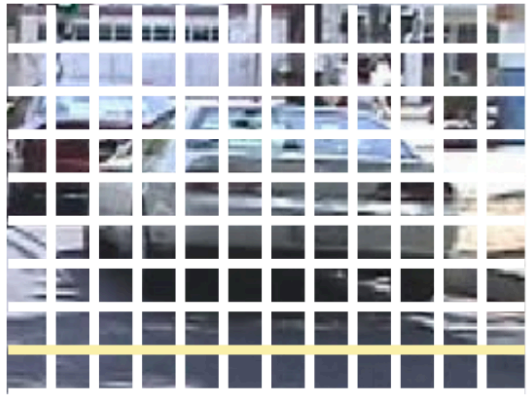
- Generative (creative) / discriminative / hybrid
- Account only form and lookout or also position and layout in some graph of dependencies
- Invariance (representation should be solid and recognize objects from: different perspectives, light, scale, deformation, noise)
- Set of extracted features / or raw pixel image?



Representation 4

- Generative (creative) / discriminative / hybrid
- Account only form and lookout or also position and layout in some graph of dependencies
- Invariance (representation should be solid and recognize objects from: different perspectives, light, scale, deformation, noise)
- Set of extracted features / or raw pixel image?



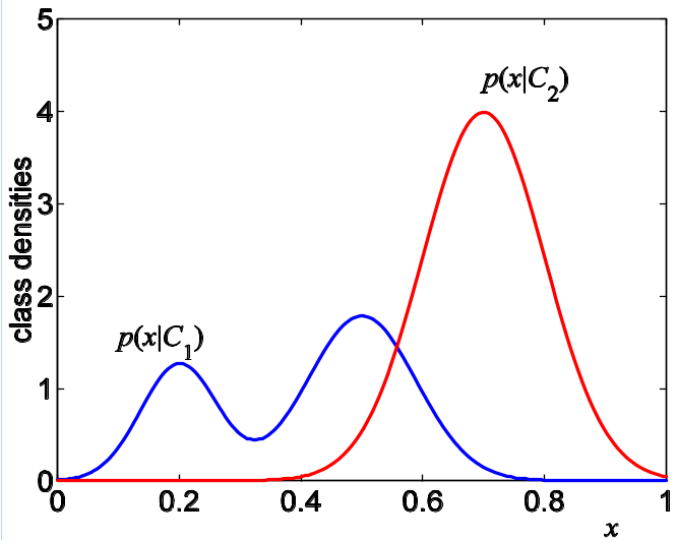


- data analysis and preprocessing for extracting features (this can be hand engineered) or full machine learning like convolutional neural networks

Learning (training)

Main methods of learning based on generative or discriminative approaches

Generative

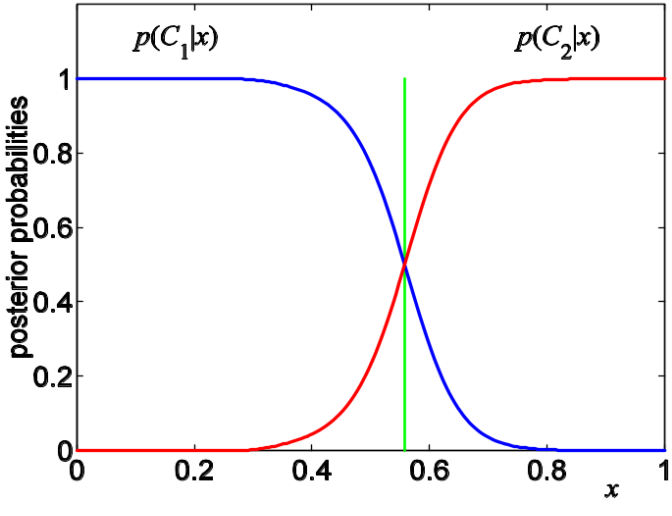


$$f(x) = \theta_1 p(x|C_1) + \theta_2 p(x|C_2)$$

The key is to find

$$\theta_1 = ?, \theta_2 = ?$$

Discriminative

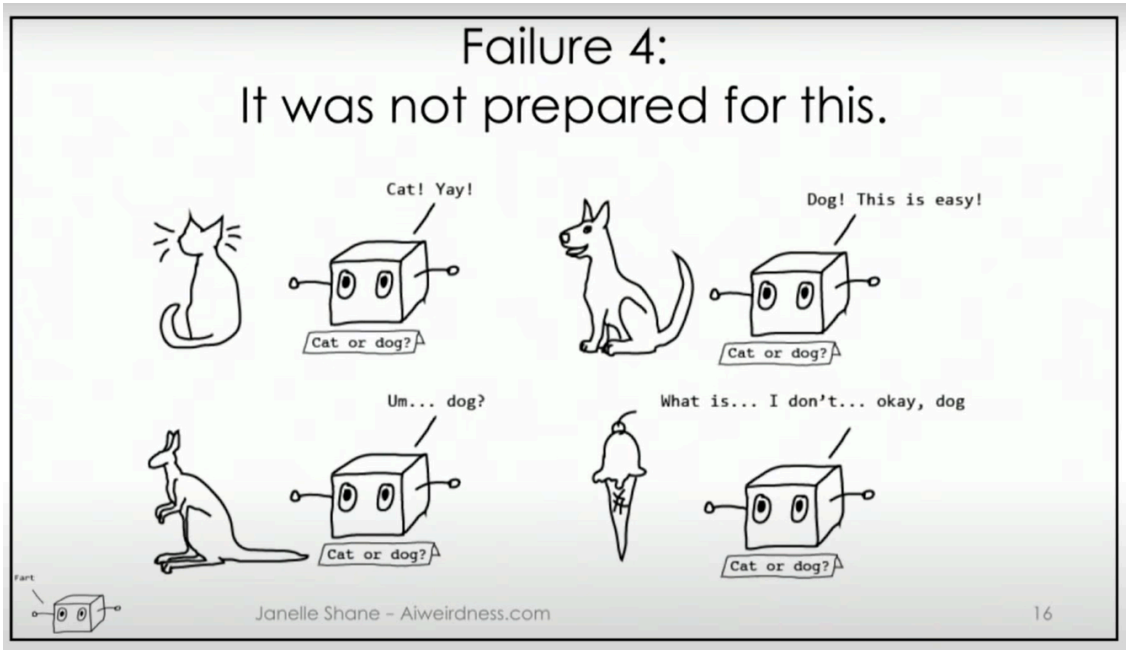


$$p(x|C_1) > ?? > p(x|C_2)$$

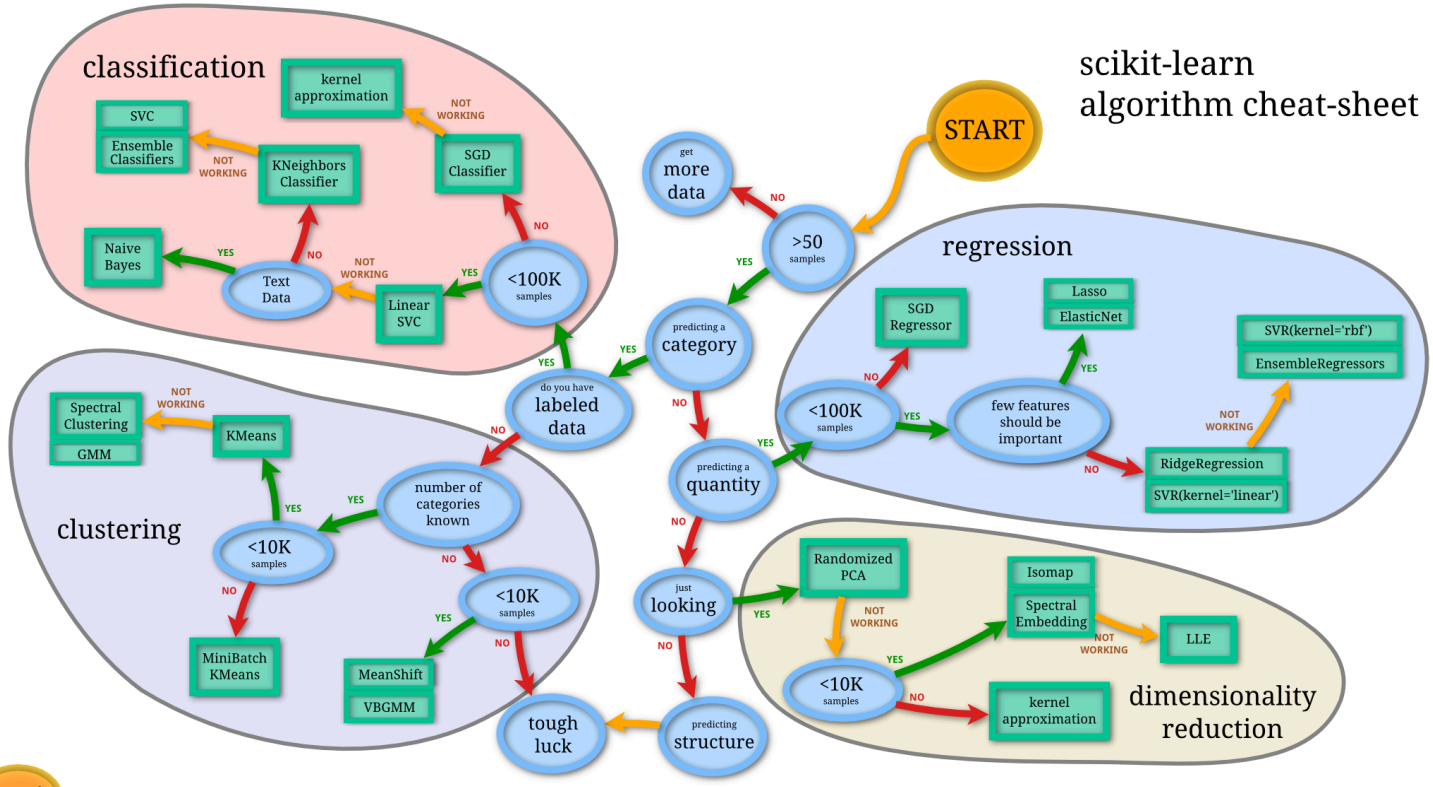
The key is simply to divide into parts.

Generalization

The term ‘generalization’ refers to the model’s capability to adapt and react properly to previously unseen, new data, which has been drawn from the same distribution as the one used to build the model. In other words, generalization examines how well a model can digest new data and make correct predictions after getting trained on a training set.



'Classical' Machine Learning Guide map



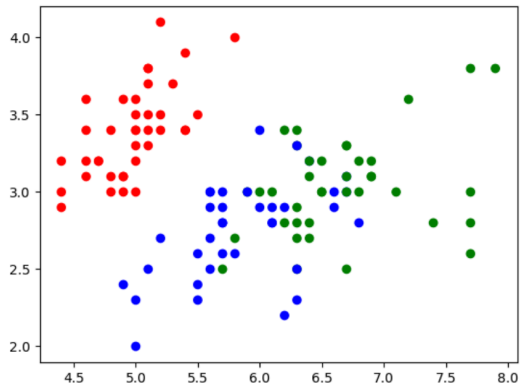
Back

Hands on Python short example

https://colab.research.google.com/drive/1T34I98_x55wUXe5schQ6vnIBm5NhvQiz?usp=sharing

```
# common imports

import numpy as np
import pandas as pd
import sklearn
from sklearn import datasets
import matplotlib.pyplot as plt
import matplotlib
from sklearn import tree
from sklearn.metrics import confusion_matrix
from sklearn.model_selection import train_test_split
```

```
# read the dataset
iris = datasets.load_iris()

X = iris['data']    # features
y = iris['target']  # supervised expected answers

# split data to train and test
X_train, X_test, y_train, y_test = train_test_split(X,y)

# analize data
cm = matplotlib.colors.ListedColormap(["red","blue","green"])
plt.scatter(X_train[:,0], X_train[:,1], color=cm(y_train))

# choose and train model
clf = tree.DecisionTreeClassifier()
clf = clf.fit(X_train, y_train)

# evaluate trained model
print(confusion_matrix(y_train, clf.predict(X_train)))
print(clf.score(X_train,y_train))

print(confusion_matrix(y_test, clf.predict(X_test)))
print(clf.score(X_test,y_test))
```



Hands on tutorial and show case for NTC

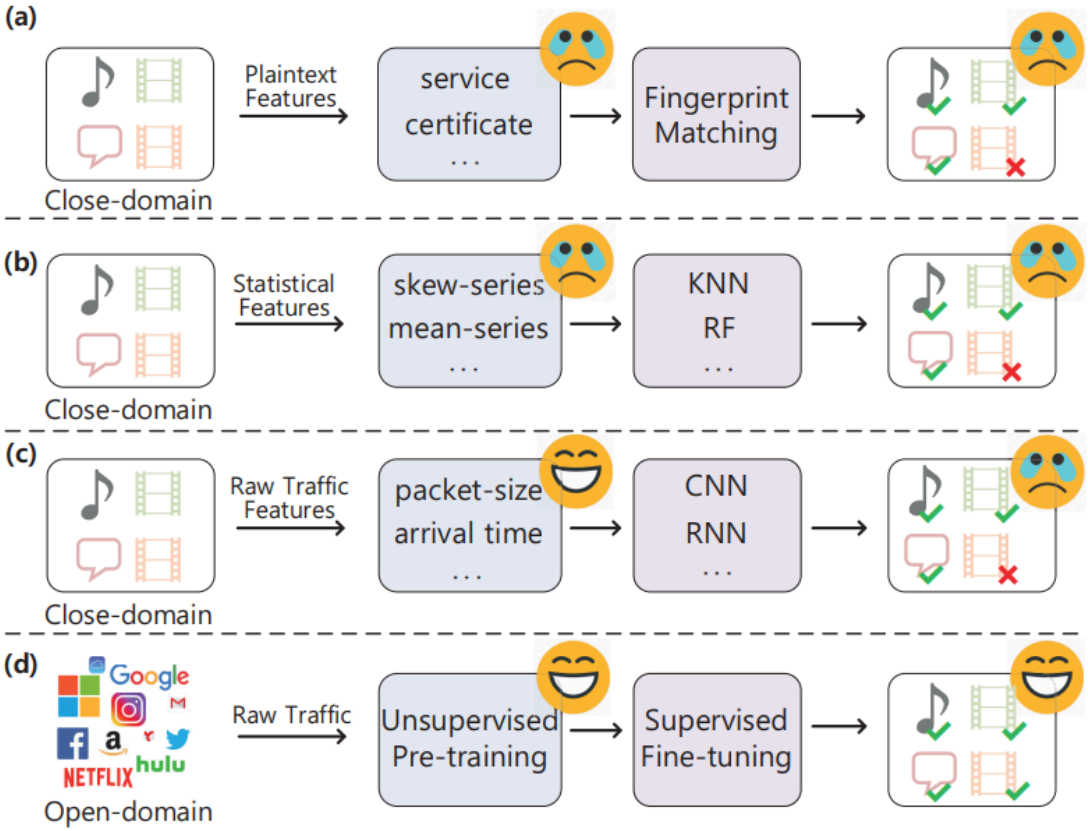
Classical NTC - header and statistical data.

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

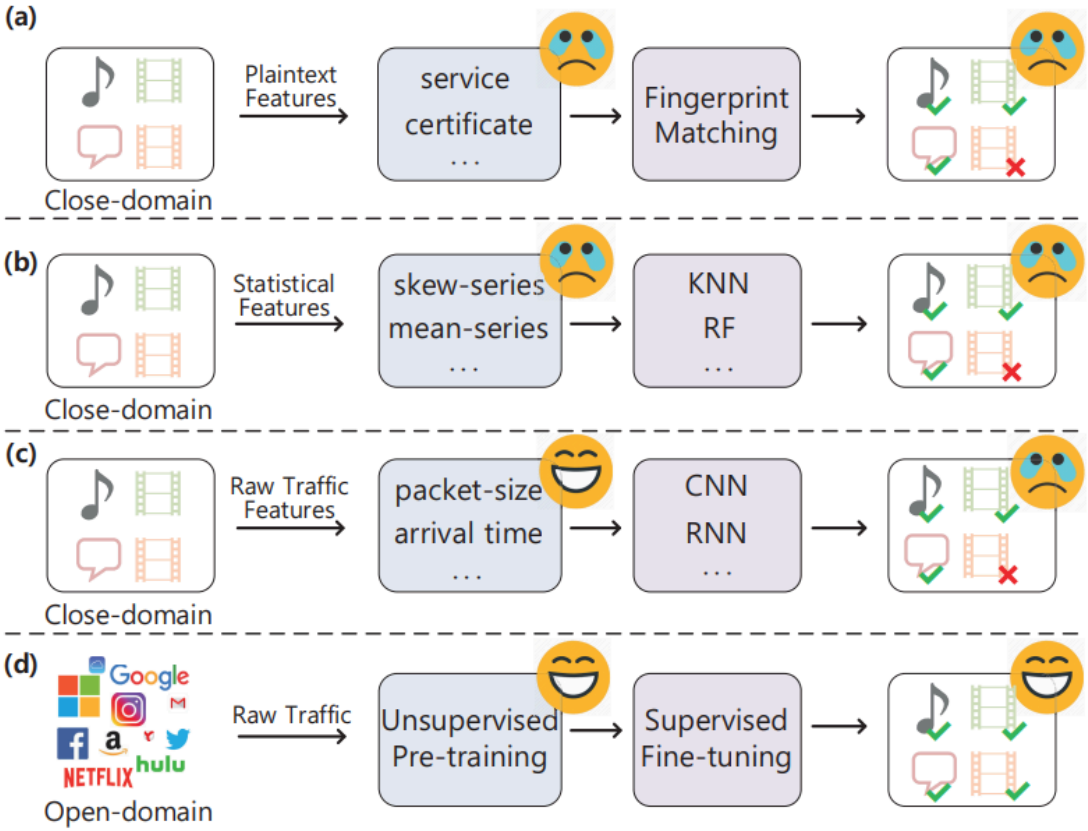
Flow.ID	a flow identifier following the next format: SourceIP-DestinationIP-SourcePort-DestinationPort-TransportProtocol
Source.IP	The source IP address of the flow.
Source.Port	The source port number
Destination.IP	The destination IP address.
Destination.Port	The destination port number.
Protocol	The 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 classication approaches



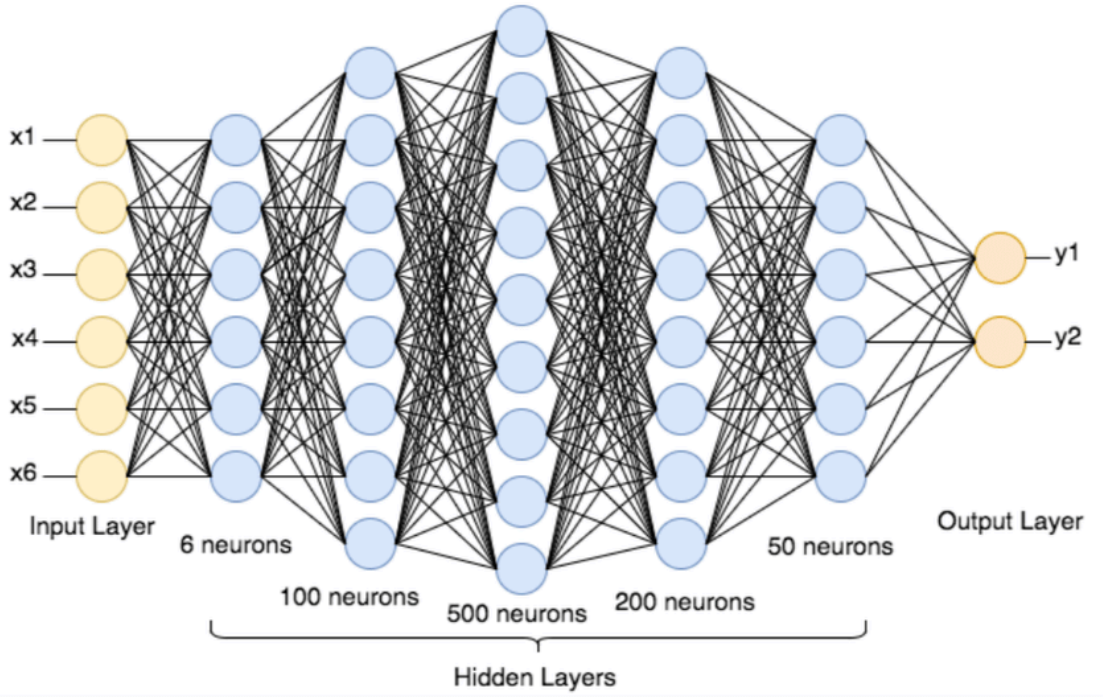
Encrypted traffic classication approaches



So we come to the third case. Can we use deep networks approach on the features directly?

What is a deep neural network?

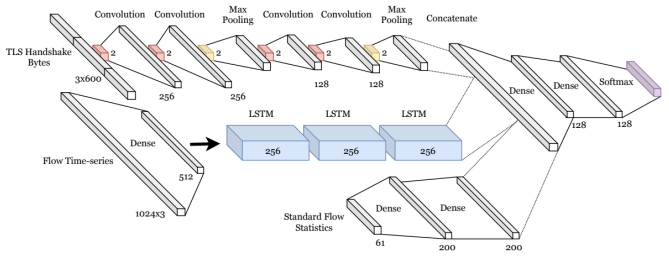
https://playground.tensorflow.org/



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

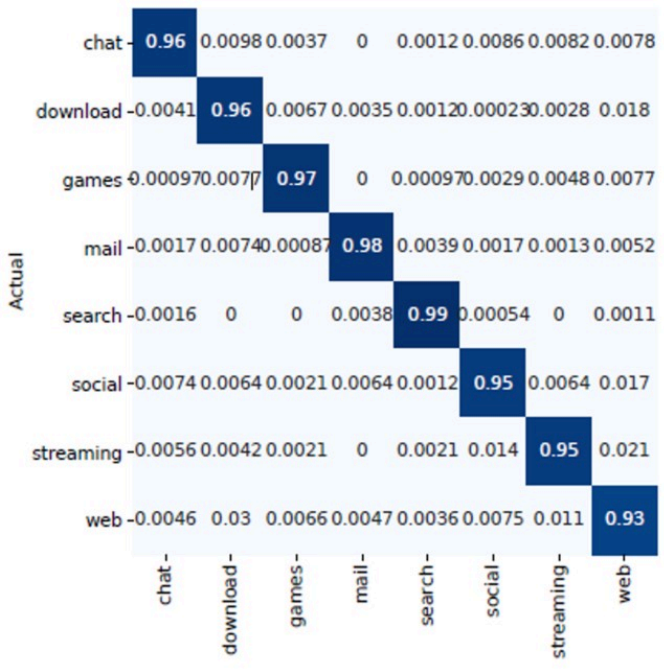
"A Look Behind the Curtain: Traffic Classification in an Increasingly Encrypted Web", *IMAN AKBARI, MOHAMMAD A. SALAHUDDIN, LENI VEN, NOURA LIMAM, and RAOUF BOUTABA, University of Waterloo, Canada BERTRAND MATHIEU, STEPHANIE MOTEAU, and STEPHANE TUFFIN*, Orange Labs, France



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.

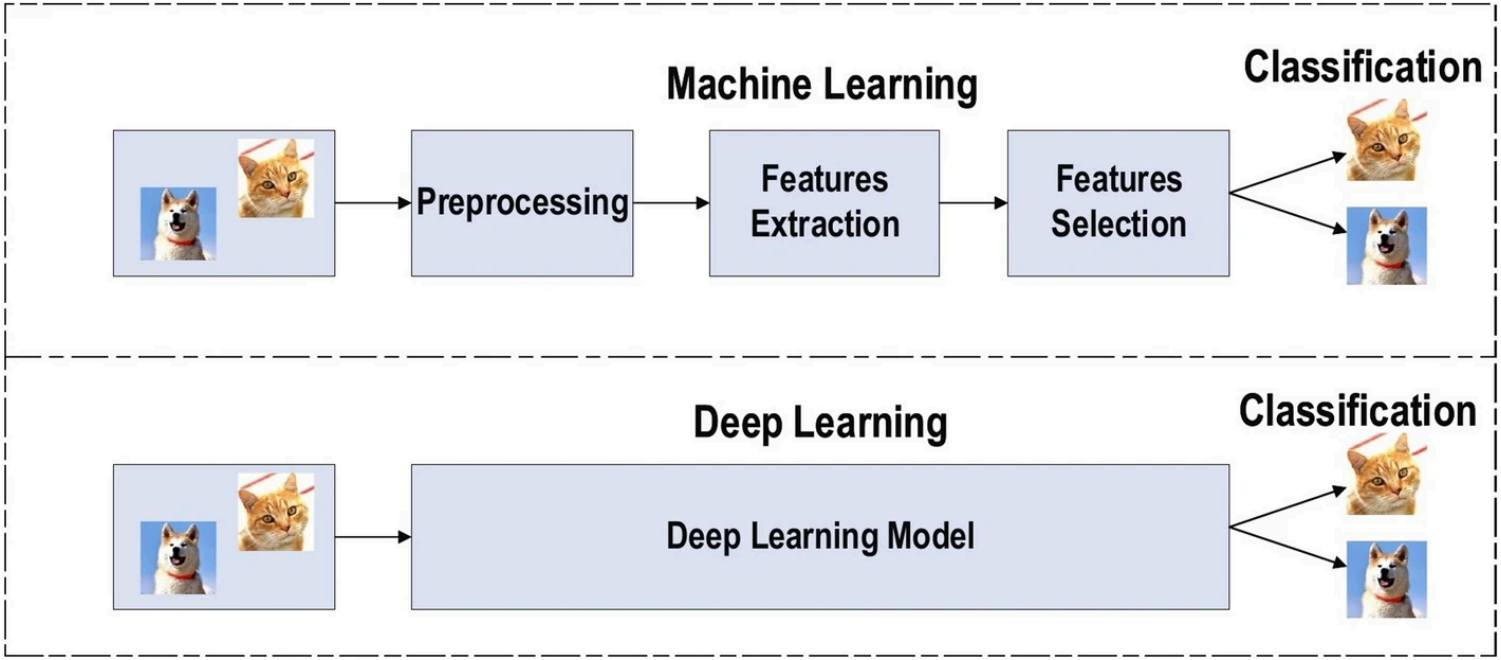
The Orange-Waterloo solution is Hybrid model

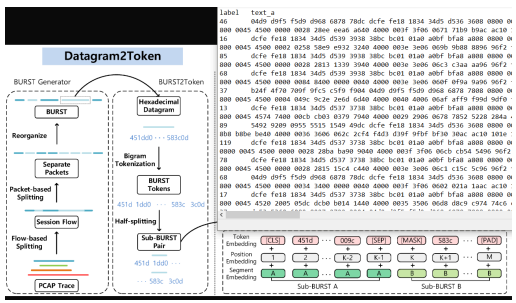
It offers superior performance over other state-of-the-art solutions, a decision tree-based solution widely used for classification, which only ensures a success rate of 81% or other CNN UCDavis solution, which has a detection rate of 91%. CNN UCDavis presents a detection quite close to Orange-Waterloo, 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).

Deep learning approach to encrypted traffic classification





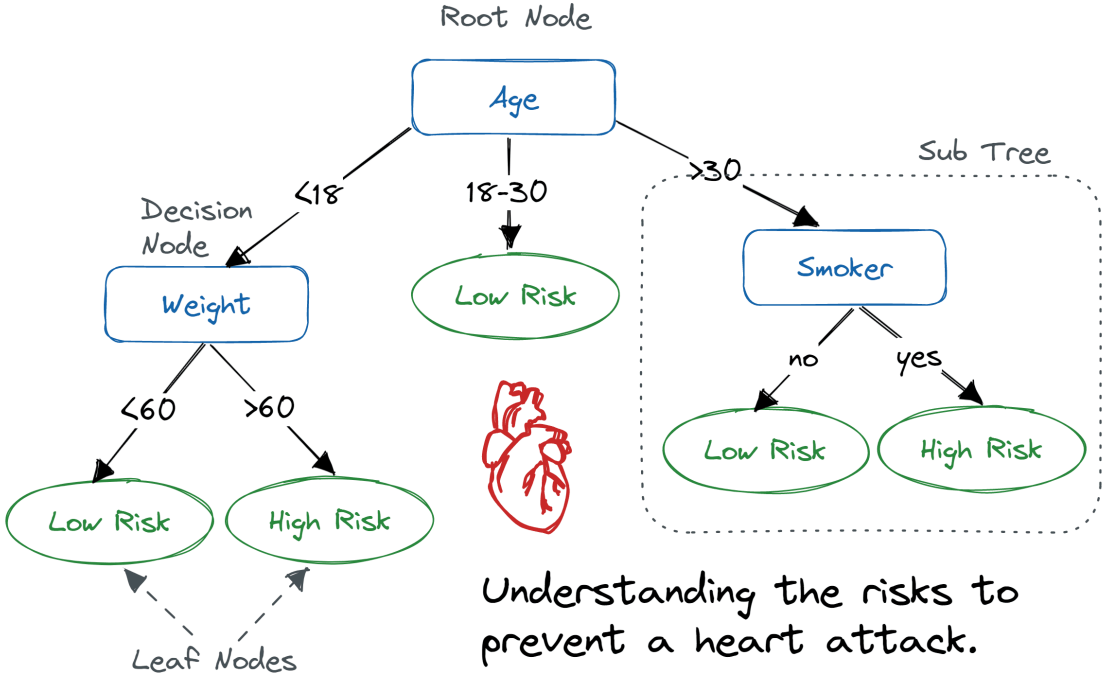
The input in Deep Learning is not a vector of engineered features, but streams interpreted as "images" in form of hexadecimal tokens.

Can we use decision trees for images?

https://colab.research.google.com/drive/1nDwTxYk20oDrqV9GG3sybSo5dSehvAsX?usp=sharing

So we concluded that we want to do NTC but, we want to use 'raw' packet contents instead of engineered features. Can we use decision trees for images?

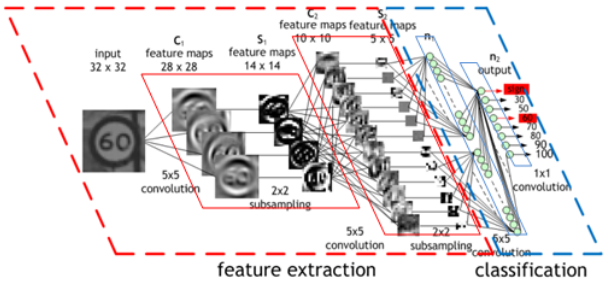
Yes..., The general idea is to interpret each pixel in the raster image is a separate feature. Thus, having a scan of a hand written digit in dimensions 8x8 pixels, we have in total 64 **distinctive** features.

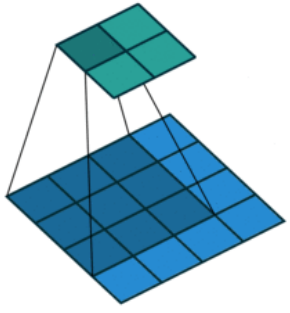


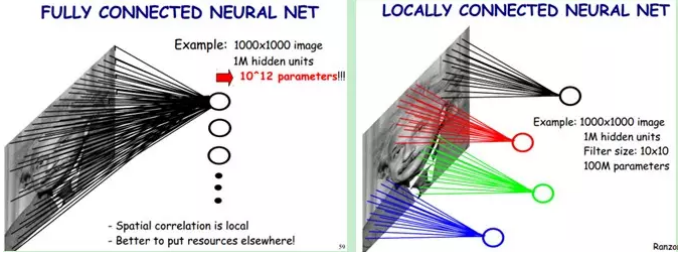
DecisionTrees for images? Come on...

Convolutional neural networks

A convolution converts all the pixels in its receptive field into a single value. For example, if you would apply a convolution to an image, you will be decreasing the image size as well as bringing all the information in the field together into a single pixel. The final output of the convolutional layer is a vector.



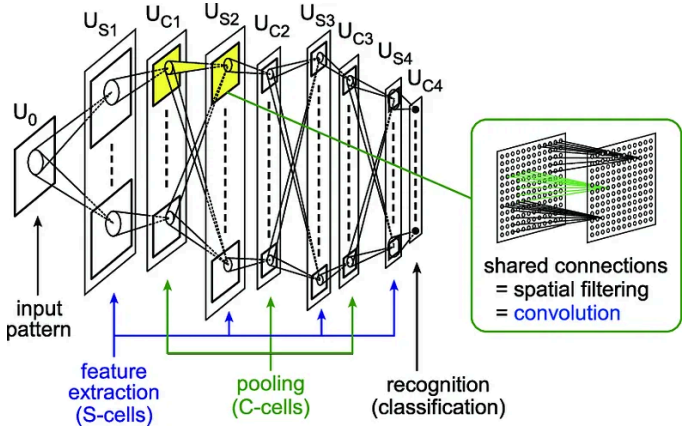




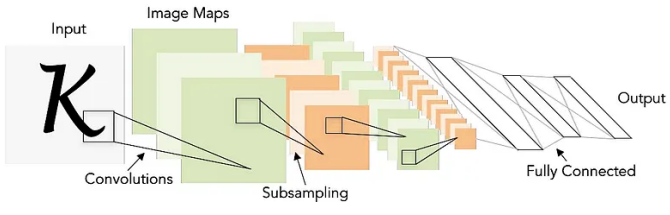
https://medium.com/appyhigh-technology-blog/convolutional-neural-networks-a-brief-history-of-their-evolution-ee3405568597

Deep networks brief history

In 1980 a concept of Neocognitro was introduced, probably the earliest precursor of CNNs. The concepts of feature extraction, pooling layers, and using convolution in a neural network were introduced and finally recognition or classification.



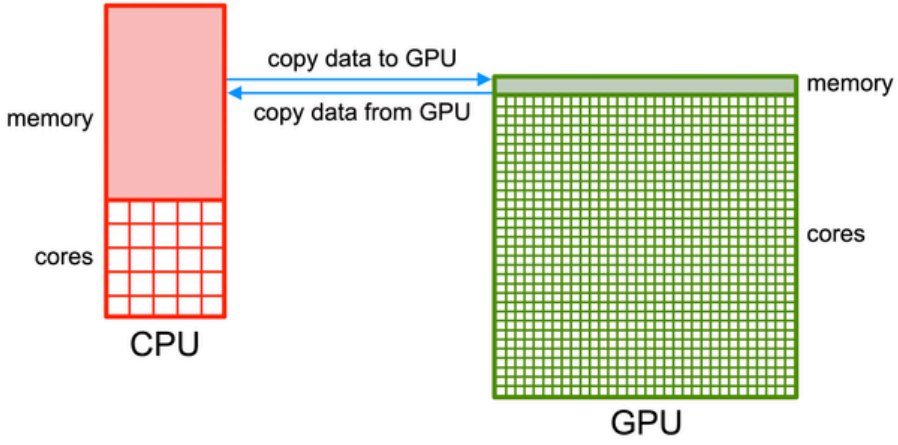
In 1989-199 the LeNet-5 (1989–1998) was introduced. The overall architecture was [CONV-POOL-CONV-POOL-FC-FC]. It used 5x5 convolution filters with a strike of 1. The pooling (subsampling) layers were 2x2 with a stride of 2. It has about 60 K parameters.



It all started with development of GPU

https://rodriguesthiago.me/posts/importance_gpu_in_training_neural_networks/

> GPUs were the go-to engine for gaming enthusiasts, centered around accelerating graphic rendering. However, around 2007/2008, thanks to the emergence of NVIDIA's CUDA architecture, GPUs underwent a makeover. They began to be used in high-performance computing and data science, dramatically expanding their applications.



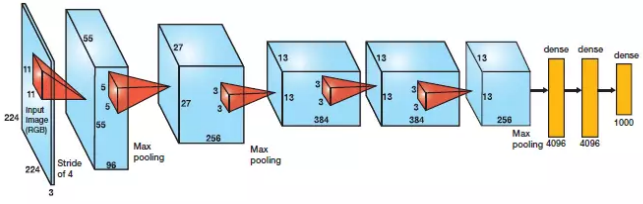

```
data = open("input.dat");          # read the data on the CPU
copyToGPU(data);                   # copy the data to the GPU
matrix_inverse(data.gpu);          # perform a matrix operation on the GPU
copyFromGPU(data);                 # copy the resulting output to the CPU
write(data, "output.dat");         # write the output to file on the CPU
```

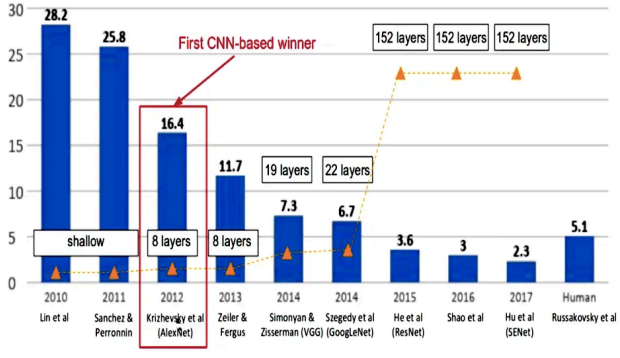
Alexnet (2012)

Finally in 2012, Alex Krizhevsky and Geoffrey Hinton came up with a CNN architecture popular to this day as AlexNet, which reduced the error from 25.8% to 16.4% which was a significant improvement at that time.

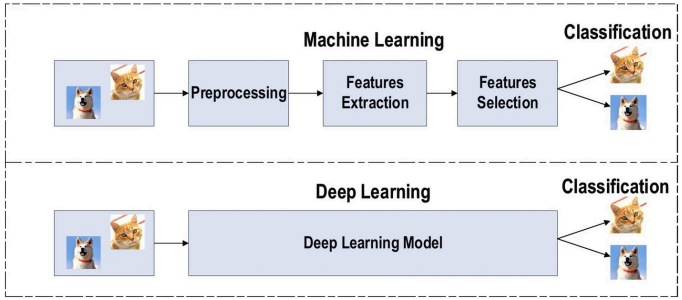
AlexNet (paper) was the first winner of the ImageNet challenge and was based on a CNN, and since 2012, every year's challenge has been won by a CNN; significantly outperforming other deep and shallow (traditional) machine learning methods.

It has about 60 M parameters.

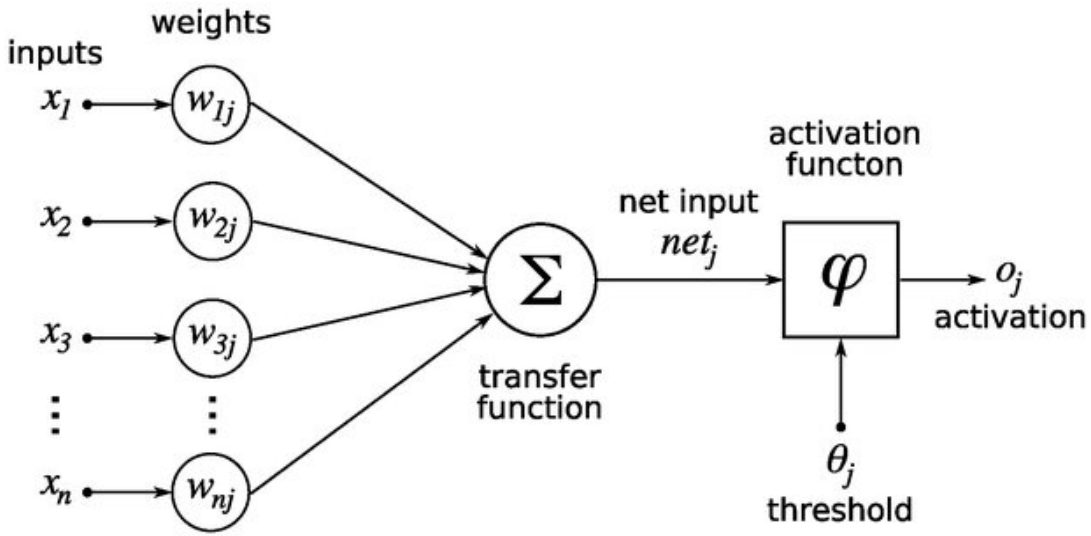




Deep networks Machine learning technologies

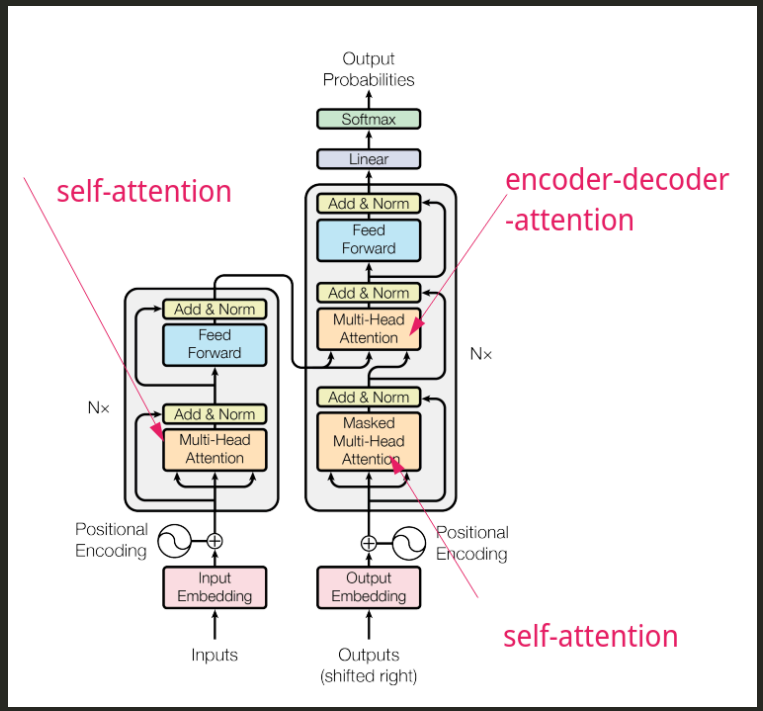


- Types of deep networks:
  - deep neural networks
  - convolutional deep networks
  - r-cnn networks
  - lstm networks
  - transformer networks (attention)
  - ViT - Vision Transformers
  - GAN (generative adversarial networks - generative networks)



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.

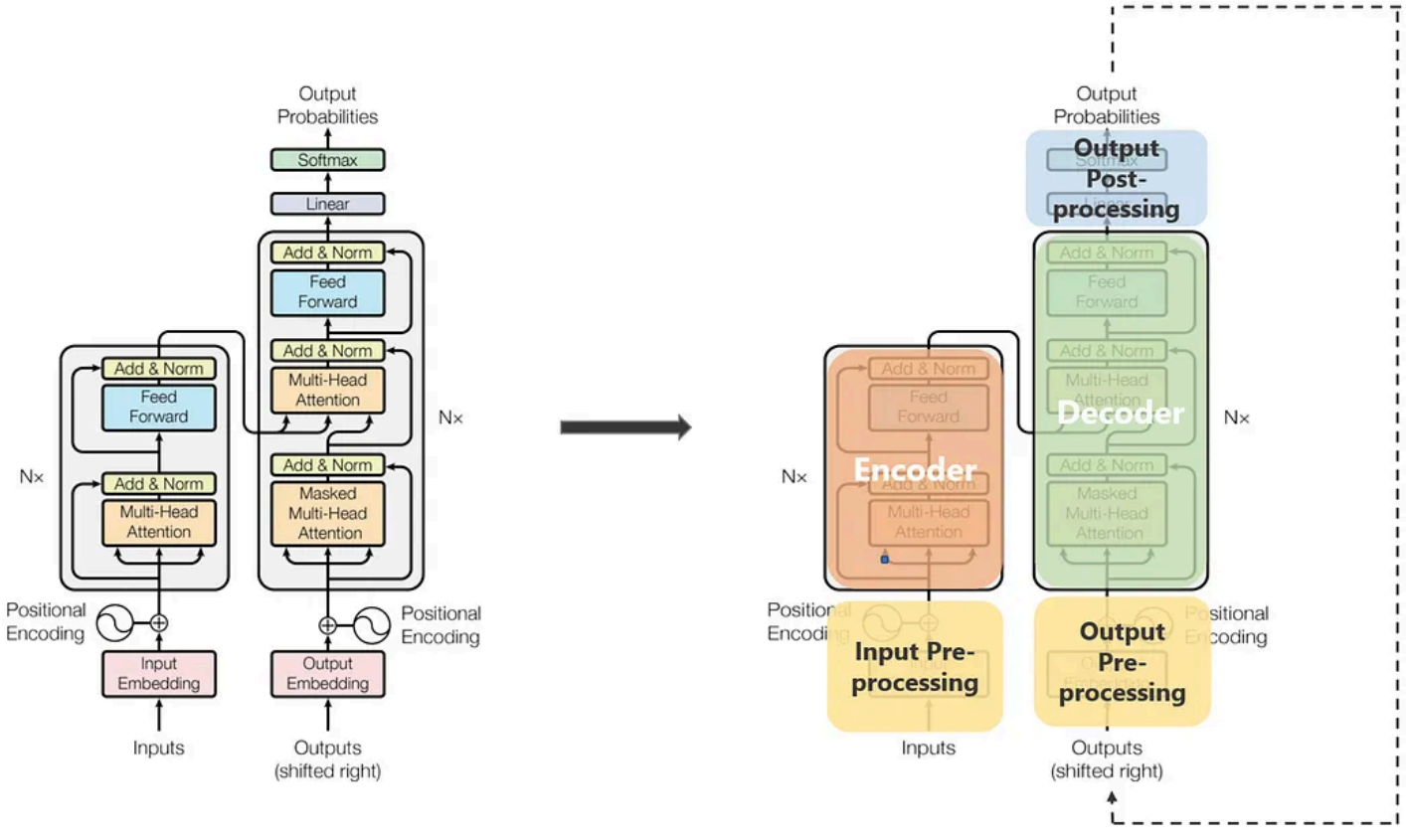
Generative AI, what it means?

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.

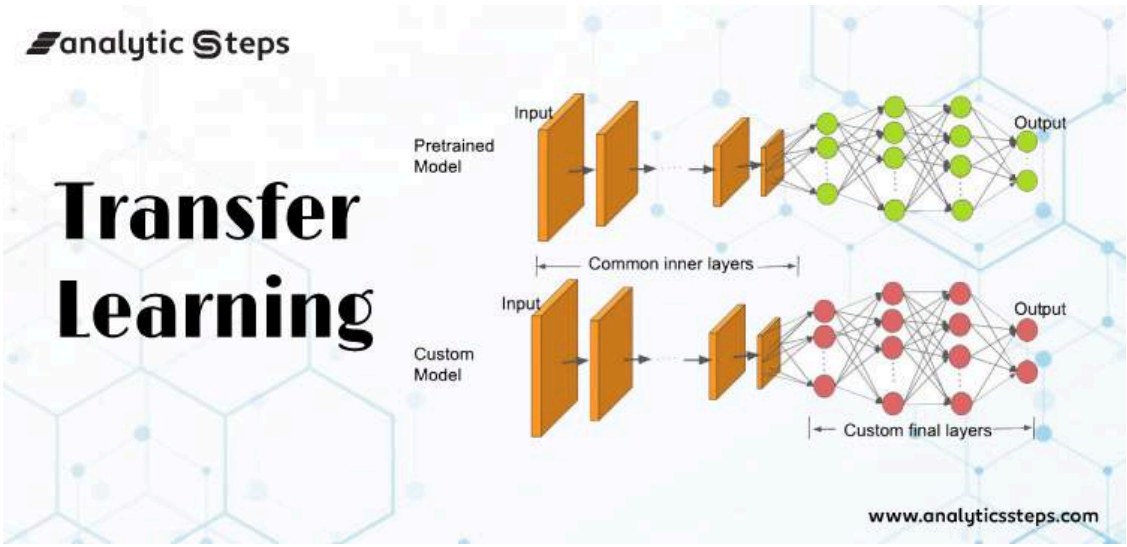
1. Using outputed text as input for the decoder. The next word dependeds on what I have said so far. This is why Chat GPT can create long and meaningfull texts.

The main problem - training

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

But...

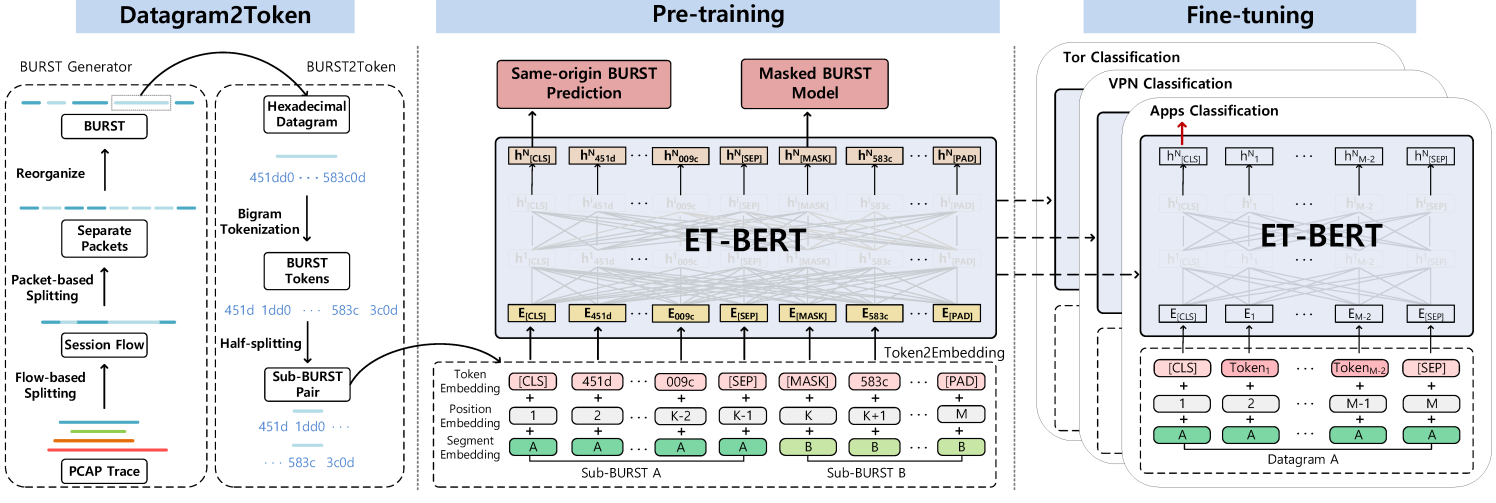
There is a technique called transfer learning (or pretrained model).



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 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/bethune-ntc-2024/