Submitted By Zia Md Galib Ul Alim

Introduction

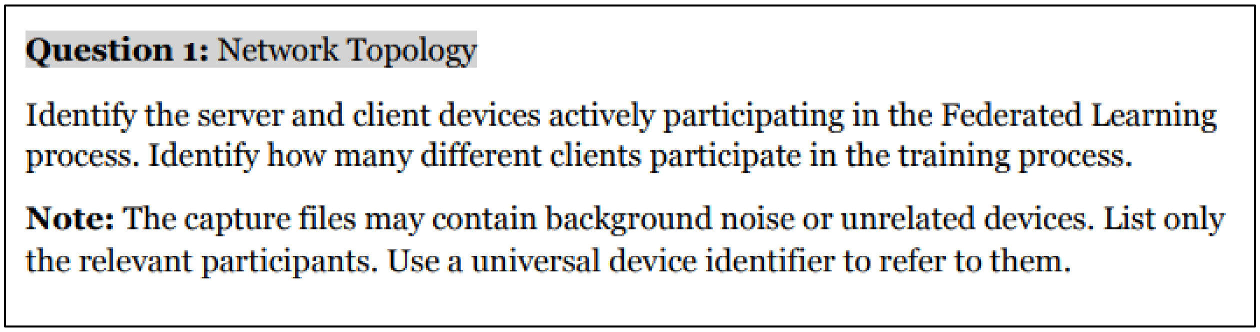
The data presented given me were 6 .pcap files with encrypted information (as in, I only could check the payload length, none of the internals were visible i.e. header and information). I have previously worked with pcap files for two of my papers (recently accepted) and have been more comfortable turning the pcap files into csv before moving forward. Here, I have explored both as csv and the raw pcap files, which seem to be Wireshark extract.

The first part of my code analyzes the pcap files' contents. I have found the following basic contents for each of the pcap files:

- Time Stamps (timestamps)
- Source IP Address (src_ip)
- Destination IP Address (dst_ip)
- Protocol (protocol)
- Source Port Number (src_port)
- Destination Port Number (dst_port)
- Length (length) that I believe is the complete payload length including header and tail.

Converting the data to csv files for a better look, I noticed the requirement of preprocessing, given the fact that there were noises (no source or destination IP addresses associated) as well as several non-FL data such as, the clients communicating between themselves and non-FL behavior.

Answers



Question 1: Network Topology

Identify the server and client devices actively participating in the Federated Learning process. Identify how many different clients participate in the training process.

Note: The capture files may contain background noise or unrelated devices. List only the relevant participants. Use a universal device identifier to refer to them.

I detected the server's address by running another code to determine all the packets sent to each destination, noticing a significantly large number of packets being sent to 192.168.216.85 than other IPs, meaning the 192.168.216.85 has to be the server assuming a normal FL environment. Table 1 shows the results.

Submitted By Zia Md Galib Ul Alim

Table – 1: Identifying Server IP

pcap File	Destination IP Address	Packets Received
1	**192.168.216.85**	**14411**
	192.168.216.223	5826
	192.168.216.22	5080
	192.168.216.1	780
	224.0.0.251	6
2	**192.168.216.85**	**51714**
	192.168.216.234	9198
	192.168.216.22	8533
	192.168.216.223	8034
	192.168.216.1	3465
	239.255.255.250	32
	224.0.0.251	5
	192.168.216.255	2
3	**192.168.216.85**	**59475**
	192.168.216.234	10010
	192.168.216.223	9289
	192.168.216.22	9135
	192.168.216.1	1933
	239.255.255.250	16
	192.168.216.255	1
4	**192.168.216.85**	**48294**
	192.168.216.234	11266
	192.168.216.223	10961
	192.168.216.22	10871
	192.168.216.1	369
5	**192.168.216.85**	**35980**
	192.168.216.223	9992
	192.168.216.22	9197
	192.168.216.234	8871
	192.168.216.1	622
	224.0.0.251	6
	192.168.216.255	4
6	**192.168.216.85**	**65804**
	192.168.216.223	16990
	192.168.216.22	16713
	192.168.216.1	1550
	224.0.0.251	6
	192.168.216.255	1

Submitted By Zia Md Galib Ul Alim

It can clearly be seen 192.168.216.85 received significantly more packets, thus identifying our server.

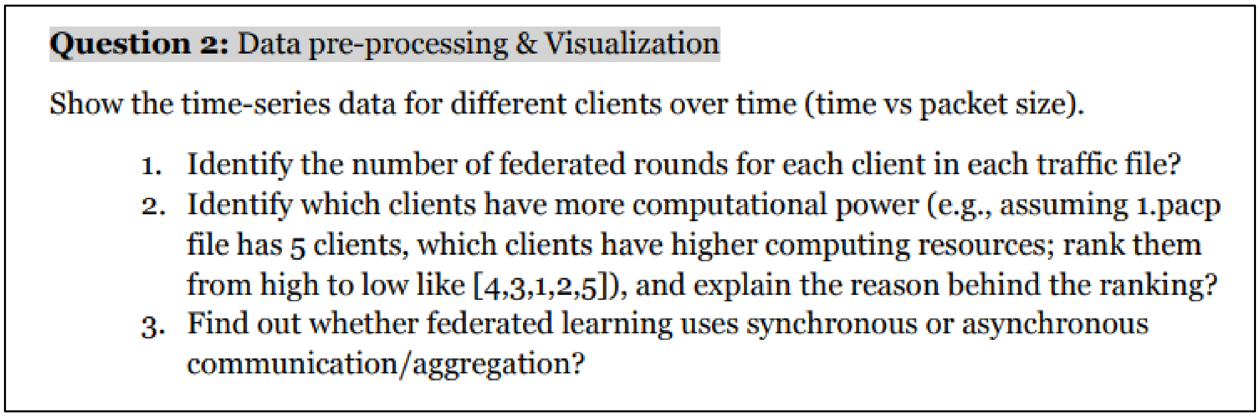
The CSV File gave an in-depth view of the IP addresses by sorting through them, also my code shows the unique IP addresses based on source IPs. **Table 2** shows the clients per pcap file.

Table – 2: Identifying the Clients

pcap File	All Source IPs	Actual Clients	Other Sources	Remarks
1	192.168.216.223	192.168.216.223		
	192.168.216.22	192.168.216.22		
	192.168.216.1		192.168.216.1	Not FL Behaviour
	192.168.216.85		192.168.216.85	Server
2	192.168.216.223	192.168.216.223		
	192.168.216.234	192.168.216.234		
	192.168.216.22	192.168.216.22		
	192.168.216.1		192.168.216.1	Not FL Behaviour
	192.168.216.85		192.168.216.85	Server
3	192.168.216.223	192.168.216.223		
	192.168.216.234	192.168.216.234		
	192.168.216.22	192.168.216.22		
	192.168.216.1		192.168.216.1	Not FL Behaviour
	192.168.216.85		192.168.216.85	Server
4	192.168.216.223	192.168.216.223		
	192.168.216.234	192.168.216.234		
	192.168.216.22	192.168.216.22		
	192.168.216.1		192.168.216.1	Not FL Behaviour
	192.168.216.85		192.168.216.85	Server
5	192.168.216.223	192.168.216.223		
	192.168.216.234	192.168.216.234		
	192.168.216.22	192.168.216.22		
	192.168.216.1		192.168.216.1	Not FL Behaviour
	192.168.216.85		192.168.216.85	Server
6	192.168.216.223	192.168.216.223		
	192.168.216.22	192.168.216.22		
	192.168.216.1		192.168.216.1	Not FL Behaviour
	192.168.216.85		192.168.216.85	Server

Submitted By Zia Md Galib Ul Alim

I could not derive an exact UDI, rather used IP addresses (as ports are varied and not a good way to identify devices), I recognize that neither are IP addresses, but at an instance, each device has only one IP attached to it, therefore IP addresses can be used to detect the device (like UDI). **So, number of clients participating are 2, 3, 3, 3, 3 and 2 for pcap files 1-6 respectively.**



Question 2: Data pre-processing & Visualization

Show the time-series data for different clients over time (time vs packet size).

1. Identify the number of federated rounds for each client in each traffic file?
2. Identify which clients have more computational power (e.g., assuming 1.pacp file has 5 clients, which clients have higher computing resources; rank them from high to low like [4,3,1,2,5]), and explain the reason behind the ranking?
3. Find out whether federated learning uses synchronous or asynchronous communication/aggregation?

<u>Answer to Question 2 (Part 1):</u>

I have written a code with the following logic –

i. The code iterates through each of the pcap files
ii. Checks if the destination port is the server (that is, 192.168.216.85) and discards any other destination IPs from consideration (as that is not relevant for the FL, as the followed-up questions suggest)
iii. Plot the unique source IPs that are connecting/sending packets to the server IP by length on Y axis against timestamps on X axis.

The result is as follows –

PCAP 1

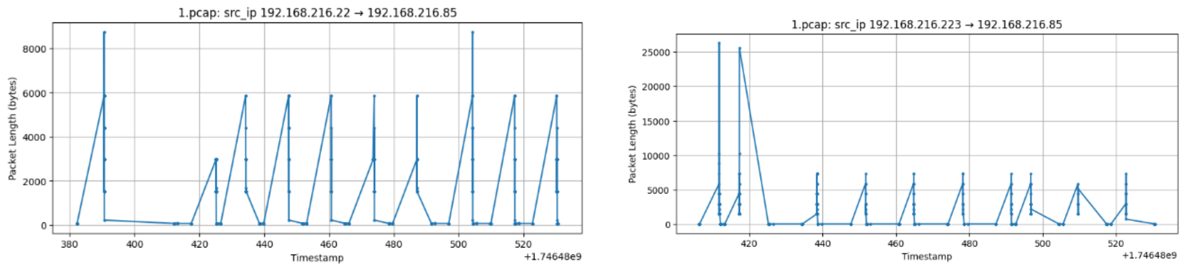
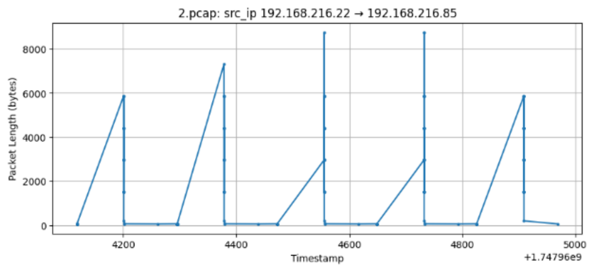
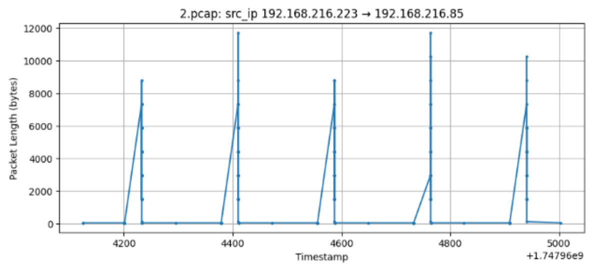


Fig: PCAP 1 Clients Packet Length vs Timestamp graphs

Submitted By Zia Md Galib Ul Alim

PCAP 2





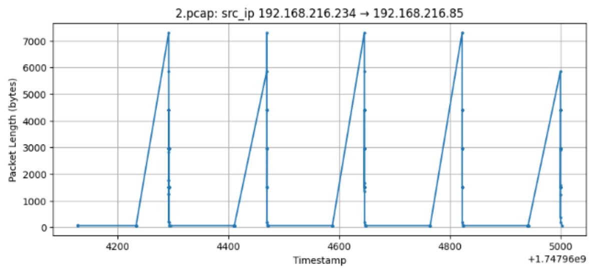
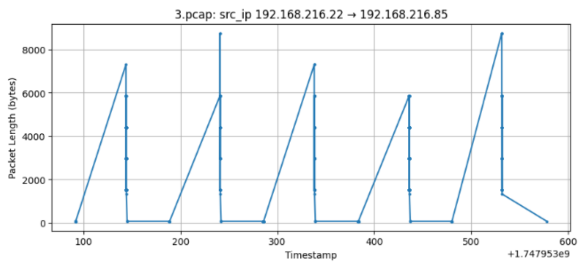
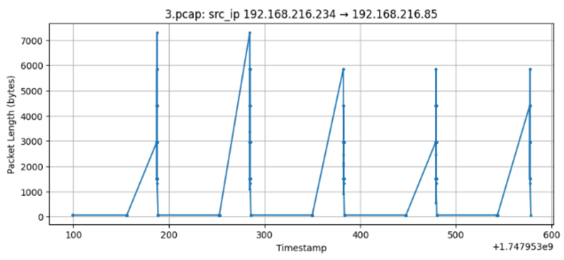


Fig: PCAP 2 Clients Packet Length vs Timestamp graphs

PCAP 3





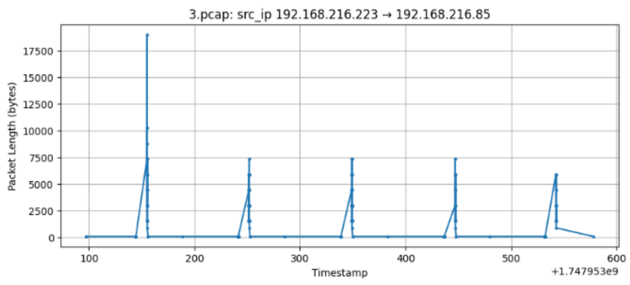
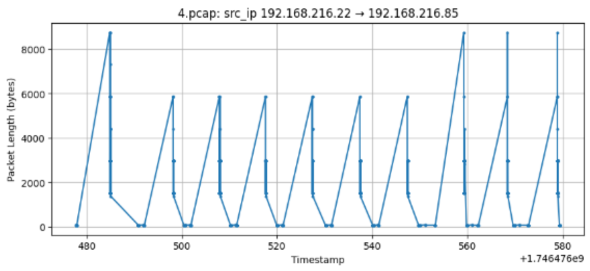
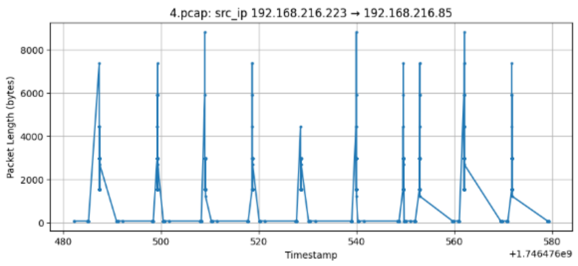


Fig: PCAP 3 Clients Packet Length vs Timestamp graphs

Submitted By Zia Md Galib Ul Alim

PCAP 4





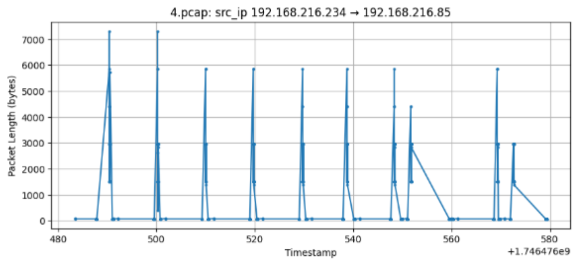
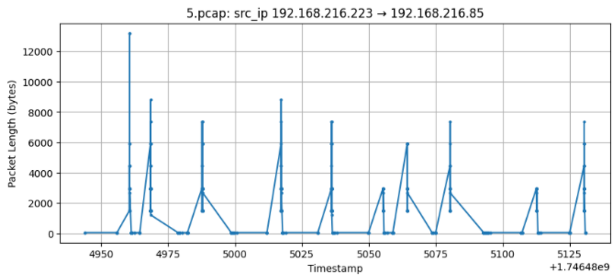
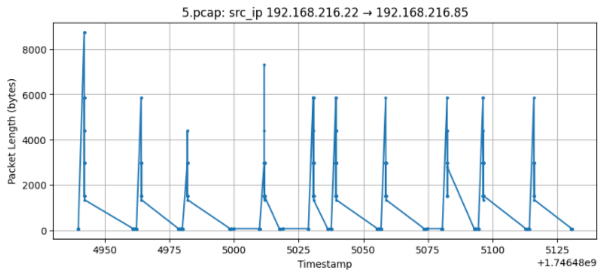


Fig: PCAP 4 Clients Packet Length vs Timestamp graphs

PCAP 5





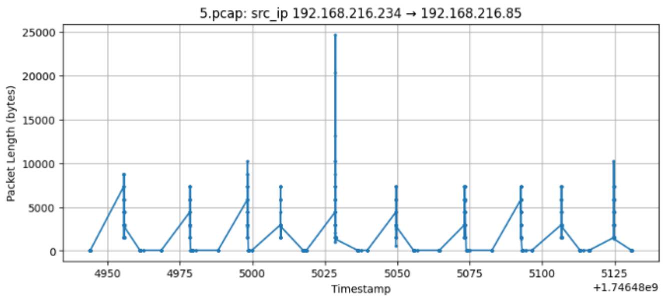
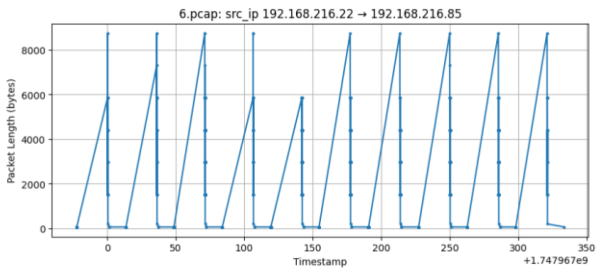


Fig: PCAP 2 Clients Packet Length vs Timestamp graphs

Submitted By Zia Md Galib Ul Alim

PCAP 6



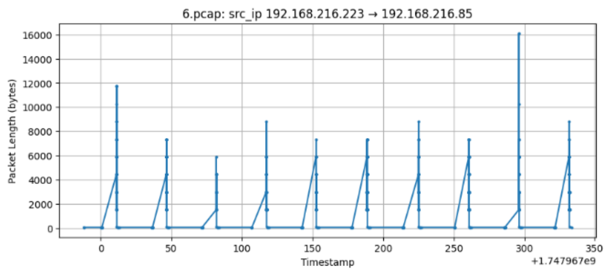


Fig: PCAP 2 Clients Packet Length vs Timestamp graphs

PCAP 1

- 192.168.216.22 – 10 FL Rounds
- 192.168.216.223 – 10 FL rounds

PCAP 2

- 192.168.216.22 – 5 FL Rounds
- 192.168.216.223 – 5 FL Rounds
- 192.168.216.234 – 5 FL Rounds

PCAP 3

- 192.168.216.22 – 5 FL Rounds
- 192.168.216.223 – 5 FL Rounds
- 192.168.216.234 – 5 FL Rounds

PCAP 4

- 192.168.216.22 – 10 FL Rounds
- 192.168.216.223 – 10 FL Rounds
- 192.168.216.234 – 10 FL Rounds

PCAP 5

- 192.168.216.22 – 10 FL Rounds
- 192.168.216.234 – 10 FL Rounds
- 192.168.216.223 – 10 FL Rounds

PCAP 6

- 192.168.216.22 – 10 FL Rounds
- 192.168.216.223 – 10 FL Round

Submitted By Zia Md Galib Ul Alim

Answer to Question 2 (Part 2):

It would be much easier to rank the clients based on the computation time against data uploaded as the time taken for one full FL round indicated by 'download global model - compute – upload model – idle time' pipeline indicates the computation power much better. However, **I have seen that the clients have minuscule round time differences**, which is understandable given a **synchronous FL model**. So, I have ranked the models based on the data upload size and upload time.

From the figures, we can rank the following –

PCAP 1 – 223 is slightly more computationally efficient than 22 given its longer idle time and slightly sharper peaks. Except one or two rounds at the start for initialization, all of them have similar payload lengths.

Ranking – 223>22

PCAP 2 – 223 once again shows much less upload times, and a higher payload length this time, 234 shows the next sharper peaks followed by 22.

Ranking - 223>234>22

PCAP 3 – Once again, similar to PCAP 2, 223 showed better upload times and structure.

Ranking – 223>234>22

PCAP 4 – For this file, although 223 and 224 shows almost similar peaks, similar round time and similar upload times, the bytes uploaded are actually different. 223 has uploaded an average of 7000 bytes payloads whereas 224 has uploaded close to 6000 bytes payloads in the same time, meaning 223 has better computing power than that of 224.

Ranking – 223>224>22

PCAP 5 – 22 seems to have a more uniform graph except lagging in the end of each upload, 224 also has a higher average payload sent than the others, meaning 224 processed much higher amount of data.

Ranking – 22>224>223

PCAP 6 – Although 22 has consistent uploads, 223 shows much better peaks and payload delivery.

Ranking – 223>22

Submitted By Zia Md Galib Ul Alim

Answer to Question 2 (Part 3):

The model seems to be synchronous given a tolerance of a few ms. The number of rounds are same, as well as the difference between rounds starting their upload is very minute, which is understandable given that the server 192.168.216.85 is not broadcasting instantly, rather sending one-to-one, so it cannot send the global update to all the model at once (like OpenFL). This works more like Flower.

Note: Briefly explain your data cleaning methodology. How did you handle non-FL-related packets to ensure your data represents only FL system behavior?

The data is cleaned by ignoring the noise and non-FL packets. Any non-ports and non-IPs are noise, whereas communications noticed to have different upload/download/payload length patterns are considered non-FL, which were detected from sources such as 192.168.216.1 and some server uploads to 224.0.0.251 and 239.255.255.250. As the IPs have been identified, I know my clients and server in every pcap file and thus anything outside of these have been dumped.

Question 3: Traffic Analysis

These traffic data files contain training traffic for different Deep learning models. We are not specifying the specific model structure. However, they are either CNN-based model or RNN-based models

1. Identify how many different types of deep learning models there might be. You can use a clustering technique if needed. Explain through analysis and visualization (For example, there might be four vision models and two time-series analysis models). For each file, explain which device is running which model?
    a. Explain your work. How are you preprocessing the PCAP files and why? What are the features you are considering, and why are those important?
    b. What techniques (statistical/ML/Deep Learning/others) are you going to apply to differentiate between those, and why?

Note: Explain with any applicable math, theory, and visualization.

Submitted By Zia Md Galib Ul Alim

Answer to Question 3 (Part a)

I have tried to answer the main question of Question 3 with the provided information directly, however it is much easier to do if I extract a few more information from the data provided by some common formulae we use after data extraction by Wireshark, such as mean packet size, packet size variability etc.

I have decided to calculate the following features:

- Mean Packet Size (mean frame and standard frame size)
- Packet Size Variability
- Inter-arrival Time
- Uplink Ratio

Mean Packet Size was chosen as CNN and RNN will have different average packet sizes, although due to varying peaks, may have exceptions.

Packet Size Variability is also important to detect CNN/RNN models as RNNs are generally much more consistent from my experience with their algorithms.

Inter-arrival time was a test case basis as we are probably dealing with synchronous data but it is still an important factor from server download to compute and upload so I decided to include this as well.

Uplink ratio was to just complement the packet size computed before as the size of the uploads (local models) to the server downloads (global model) gives an idea about the computation size.

Formula used:

$$mean\ packet\ size = \frac{1}{N}\sum_{i=1}^{N} L_i$$

Where, L_i = Packet length in bytes and N = number of packets.

$$standard\ frame\ size = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(L_i - \mu)^2}$$

$$Inter\ Arrival\ Times\ \Delta t = t_{i+1} - t_i$$

Thus, the mean inter-arrival time:

$$mean\ inter\ attival\ time = \frac{1}{N-1}\sum \Delta t$$

$$standard\ inter\ arrivals = \sqrt{\frac{1}{N-1}\sum (\Delta t - \frac{\Delta t}{N})^2}$$

$$uplink\ ratio = \frac{packets\ sent\ to\ server}{total\ packets}$$

Next, I have decided to use PCA to divide the extracted data into two axes, so that I can cluster the devices better. This also shows uncorrelated data much better. As I do not have any labels, the clusters are labeled 0 and 1 now.

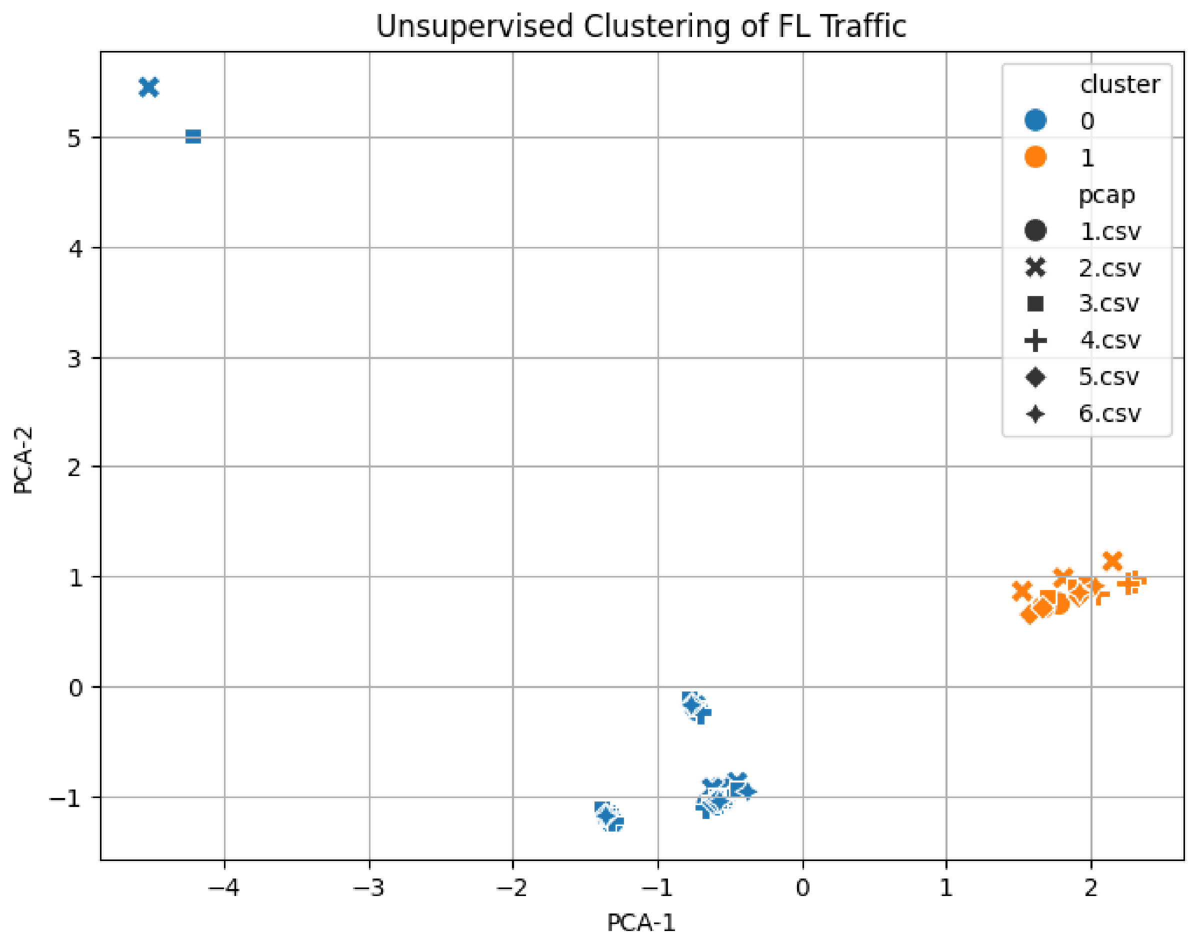


Fig: PCA Plot

Submitted By Zia Md Galib Ul Alim

The clusters made it clear that there are two main types of models (and two other models with much higher variance, that I believe are Vision models). The problem I have run into is not knowing the labels/classes, so although I can ensure that there are two different models, **it is intuitive to decide which cluster is CNN and which one is RNN**.

I have created a few KL plots to understand how the extracted data can be used for clustering (mainly to visualize). As expected, the inter-time arrival was not of much help but the others have been quite helpful.

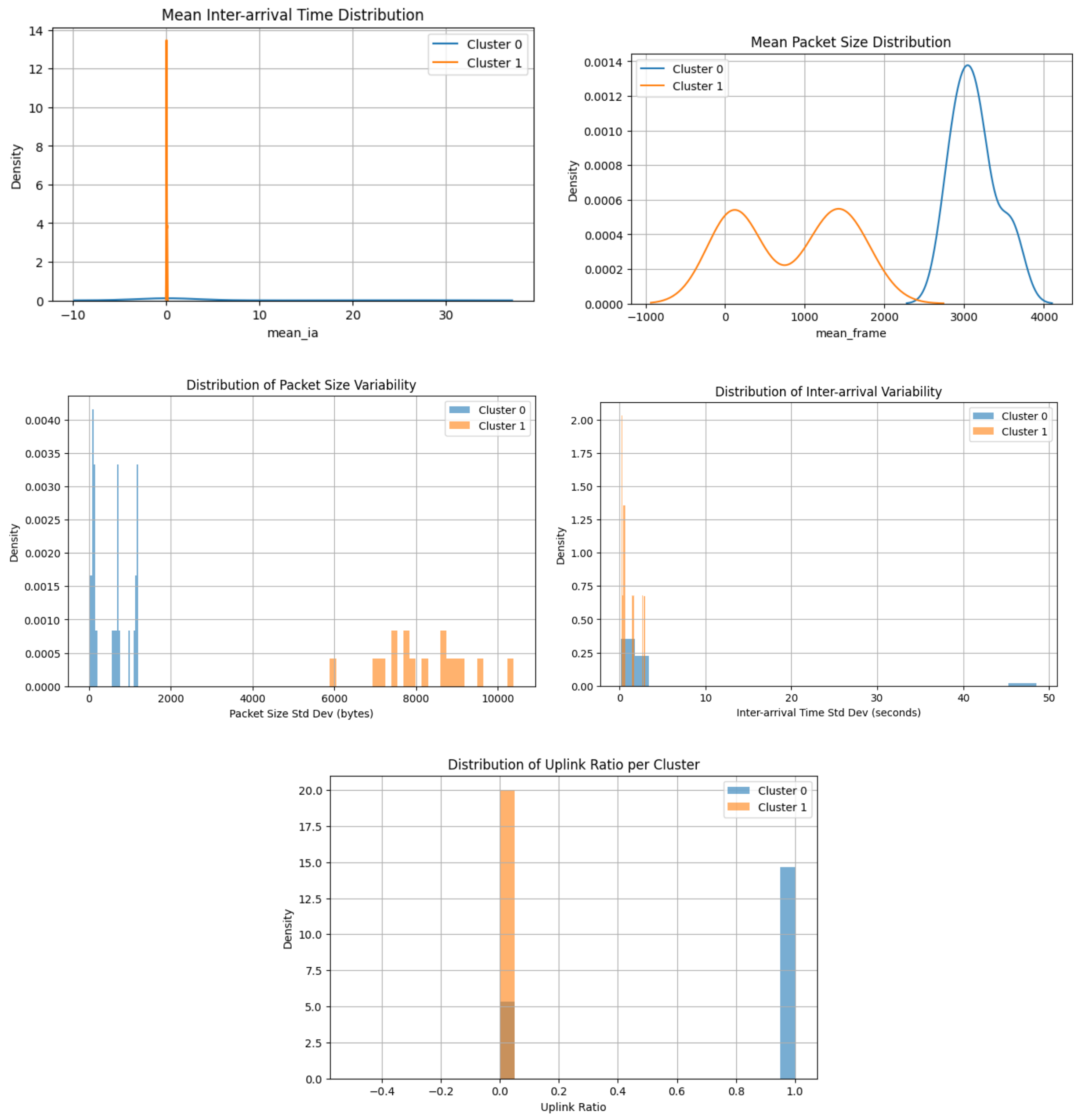


Fig: KL Plots of Density against various extracted parameters

Submitted By Zia Md Galib Ul Alim

After ensuring the two classes, as the data is unsupervised, I checked the graphs visually to have the following assumptions:

For 1.pcap file,

192.168.216.22 and 192.168.216.223 both seem to be running CNN models with Vision frameworks.

For 2.pcap file,

192.168.216.22 is assumed to be running an RNN model, whereas 223 and 234 are both running CNN models.

For 3.pcap file,

192.168.216.22 seems to be running an RNN again, and 223 a CNN model with vision and 234 a basic CNN model.

For 4.pcap file,

All three devices most likely running an RNN model as they show tails in the graphs.

For 5.pcap file,

22 and 234 seems to be running RNN models, whereas 223 is probably a CNN model.

For 6.pcap file,

Both seem to be running CNN models.

Answer to Question 3 (Part b):

I have applied K-Means KNN and also several data tree structures (such as Random Forest) as they are in my opinion, the best to detect cluster information, but classical ML for unsupervised data did not yield the best results. Given more information however, I believe it will be much easier to address using PCA to KNN. A DNN model also looks to be usable given the low covariance, but I did not have the time to implement a fine-tuned DNN model.

Therefore, if I am to apply, I would select traditional ML models such as KNN, Kernel SVMs and Data Trees such as Random Forests. I would also like to use DNN models if I can fine tune their parameters given the time.