

AI PROMPTS FORM – SEMESTER TWO

This document details the application of generative AI in the research and development of the Network Traffic Profiler application. Each table represents a prompt sent to an AI.

Context:	Action classification machine learning implementation research
Team Member:	Tim: <input checked="" type="checkbox"/> Tomiris: <input type="checkbox"/> Amelia: <input type="checkbox"/> Sophia: <input type="checkbox"/>
Prompt	<p>My team have developed a network traffic profiler tool that categories network packets from a PCAP file into network flows, and gives basic insights via an easy to use interface using python and libraries Streamlit, Pandas, Numpy and Scapy. We want to implement ML to analyse the PCAP file and try to determine the type of action the user is doing. For now, we just want the ML to be able to predict actions on Youtube.</p> <p>Our client has given us 150 pcap files of each action on Youtube: Like, Play, Subscribe, Comment & Search. Each is labelled though we do not know the exact time at which the action occurs within the file. Each file is under 10 seconds long.</p> <p>How should we proceed?</p>
Response	<p>This is a classic Traffic Analysis problem. Since YouTube traffic is encrypted (HTTPS/QUIC), you can't look at the packet payloads to see a "Like" button click. Instead, you have to rely on Side-Channel Information—the patterns of packet sizes and timing.</p> <p>Here is a roadmap to transition your profiler from a basic statistics tool to an ML-powered classifier.</p> <hr/> <p>1. Feature Engineering (The Most Critical Step)</p> <p>Since you don't have exact timestamps, treat each 10-second PCAP as a single observation. You need to convert the raw Scapy packets into a numerical "fingerprint."</p> <p>Flow-Based Features</p> <p>Group packets by their 5-tuple (Source IP, Dest IP, Source Port, Dest Port, Protocol) and calculate:</p> <ul style="list-style-type: none">• Total Bytes/Packets: Sent vs. Received.• Packet Size Stats: Mean, Median, StDev, and Max packet sizes.• Inter-Arrival Time (IAT): The time between packets. Actions like "Search" often have a burst of small

	<p>packets, while "Play" has sustained large downstream packets.</p> <p>Time-Series Features (The "Shape" of the Action) Divide the 10 seconds into small windows (e.g., 500ms bins) and count bytes per bin. This captures the "rhythm" of the action.</p> <ul style="list-style-type: none"> • Burstiness: How many spikes occur? A "Comment" might have a distinct upload spike compared to a "Like." 								
	<p>2. Data Preprocessing Pipeline Since you're already using Pandas and Scapy, your workflow should look like this:</p> <ol style="list-style-type: none"> 1. Parse: Use Scapy to extract (timestamp, size, direction) for every packet. 2. Label: Map the filename (e.g., like_01.pcap) to a numerical target (e.g., Like = 0, Play = 1). 3. Aggregate: Convert the list of packets into a single row in a Pandas DataFrame. 4. Normalize: Network data varies wildly (a 1500-byte packet vs. a 40-byte ACK). Use StandardScaler from Scikit-Learn so the ML model doesn't get overwhelmed by large numbers. 								
	<p>3. Selecting a Model Given you have 750 files (150 per class), you are in the "small to medium" data range.</p> <table border="0"> <thead> <tr> <th>Model</th><th>Why use it?</th></tr> </thead> <tbody> <tr> <td>Random Forest</td><td>Great baseline. It handles non-linear relationships and tells you which features (like packet size) are important.</td></tr> <tr> <td>XGBoost / LightGBM</td><td>Generally the gold standard for tabular data and highly accurate.</td></tr> <tr> <td>1D-CNN</td><td>If you want to treat the packet sequence as a signal. Higher complexity, but good at finding timing.</td></tr> </tbody> </table> <p>Export to Sheets</p> <p>Recommendation: Start with a Random Forest. It's robust, hard to overfit, and integrates perfectly with your existing Python stack.</p>	Model	Why use it?	Random Forest	Great baseline. It handles non-linear relationships and tells you which features (like packet size) are important.	XGBoost / LightGBM	Generally the gold standard for tabular data and highly accurate.	1D-CNN	If you want to treat the packet sequence as a signal. Higher complexity, but good at finding timing.
Model	Why use it?								
Random Forest	Great baseline. It handles non-linear relationships and tells you which features (like packet size) are important.								
XGBoost / LightGBM	Generally the gold standard for tabular data and highly accurate.								
1D-CNN	If you want to treat the packet sequence as a signal. Higher complexity, but good at finding timing.								
	<p>4. Implementation Strategy Since you are using Streamlit, you can make the transition seamless:</p> <ul style="list-style-type: none"> • Training Phase: Create a separate script to process the 750 PCAPs and save a trained model using pickle or joblib. • Inference Phase: In your Streamlit app, allow the user to upload a <i>new</i> PCAP. Your code will: <ol style="list-style-type: none"> 1. Run the Scapy profiler you already built. 2. Extract the same features used during training. 3. Pass them to model.predict(). 4. Display the result (e.g., "Detected Action: Search with 85% confidence"). 								
	<p>5. Potential Pitfalls</p> <ul style="list-style-type: none"> • Background Noise: Did the PCAPs capture <i>only</i> YouTube traffic, or is there background OS telemetry? You 								

	<p>may need to filter by the YouTube server IP ranges or DNS queries first.</p> <ul style="list-style-type: none"> • TCP vs. QUIC: YouTube heavily uses the QUIC protocol (UDP port 443). Ensure your Scapy logic accounts for UDP traffic, not just TCP streams. • The "Play" Overlap: Almost every action involves some "Play" activity (video buffering). Your features need to focus on the <i>delta</i>—the specific burst of data that happens when a "Subscribe" or "Comment" API call is made.
--	--

Context:	
Team Member:	Tim: <input type="checkbox"/> Tomiris: <input type="checkbox"/> Amelia: <input type="checkbox"/> Sophia: <input type="checkbox"/>
Prompt	
Response	

Context:	
Team Member:	Tim: <input type="checkbox"/> Tomiris: <input type="checkbox"/> Amelia: <input type="checkbox"/> Sophia: <input type="checkbox"/>
Prompt	
Response	