Quick guide for AWS

- Sign-in process
- Creation of a Lambda function

Nodes (Version 1)

- **ingestion**: Get buddy's input video and make it normal to MP4 + WAV for video and audio separate, get some meta data like your duration stuff and even FPS, and upload them to S3 bucket.
- **perception (ASR)**: load the audio, convert speech to text, return the time stamped words or segments and also the confidence scores for each stuff.
- **perception (frame caption)**: basically "groups" consecutive frames basically man its like a DSU but only 2 consecutive frames can be in a group, so how do u define a group? a group is basically a "scene". for example the video has a news reporter talking some bs and suddenly we shift to some animal kingdome stuff, so one "group" is the news reporter talking and another "group" is the animal kingdome shit.
- **perception (key frame)**: this stuff essentially runs after the frame caption. it basically intelligently picks one representative frame of the "group" that we made.
- **perception (OCR)**: Reads on-screen text from frames (lower thirds, banners, dates, URLs) and time-ranges stable text spans.
- **claim extraction**: Converts ASR/OCR/captions into concise, normalized claims (who/what/where/when), deduplicated and ID'd and stuff like thta.
- **evidence retrieval**: this guy basically runs on claim extraction ka result. it basically takes the claims, queries the internet, checks the crazy results and basically maps each claim with some resources and stuff... basically what GPT does when you enable "web search" (atleast that is what i think it does)
- **source reliability**: basically gives some score of how "reliable" the source is... for example, say that the claim guy got shit from say, the hindu news channel or something, then it is very reliable (NOT BEING POLITICAL HERE 😭) compared to something like, say, wikipedia or something.
- **forensics (lip speech sync)**: need to make sure if the lips sync or not man, just store some sort of a "sync score" in S3 so later we can do something with it.
- **forensics (gesture to narration alignment)**: Measures how much "same-ness" is there between narrated emphasis/action words and visible hand/arm motion peaks over time.

- **forensics (frame-level artifact detector)**: Flags visual inconsistencies (edge/skin/eye-blink anomalies, texture/boundary artifacts) across faces/frames.
- **consistency & scoring**: this guy should be there, one suggestion is to take all the “scores” as features and run it as a logistic regression. so one video makes one “row” in the data… we need to manually label stuff unless someone finds a better way 😟
- **explanation**: maybe have this to see the results and here slap a LLM call and ask that guy to make a damn report guys istg cant run this shit locally (no money)

Nodes (Version 2)

Nomenclature of the nodes and features are given bellow, and '*' can be replaced by decimal numbers.

A* - Audio nodes

V* - Video nodes

E* - Evidence nodes

f* - actual features generated for logistic regression

LR - single logistic regression node that takes input from J node

LRU - single logistic regression coeff updater when label is provided

Nodes

A* nodes

- [ ] A1 = Demux + AudioExtract
- [ ] A2 = VAD + ASR
- [ ] A3 = Audio spike TIME SERIES

V* nodes

- [ ] V1 = Keyframes + FaceTrack
- [ ] V2 = OCR Overlays
- [ ] V3 = Mouth Landmarks TIME SERIES
- [ ] V4 = Blink rate + Head-pose dynamics
- [ ] V5 = Texture / ELA (compression/texture anomaly)

C* nodes

- [ ] C1 = Lip Sync Score
- [ ] C2 = Gesture and Narration Check
- [ ] C3 = Claim Extraction

E* nodes

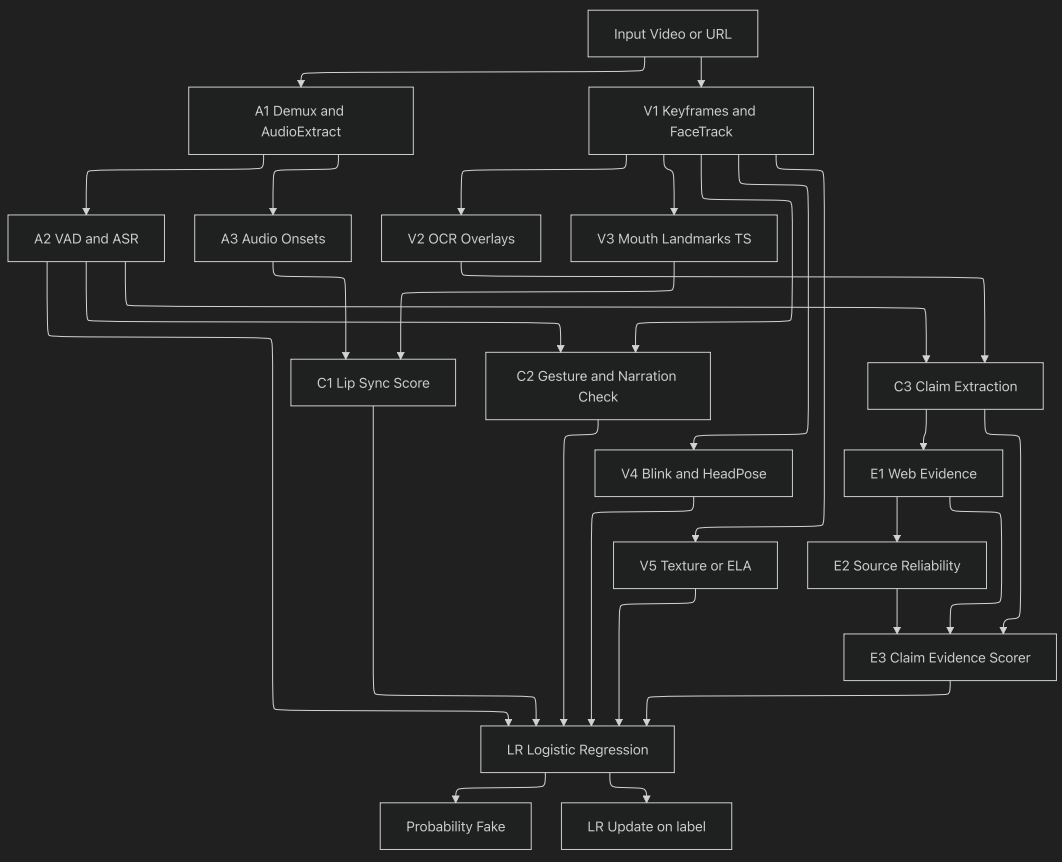
- [ ] E1 = Web Evidence
- [ ] E2 = Source Reliability

- [ ] E3 = Claim Evidence Scorer

f* features:

- A2 - f1 (speech_rate)
- A2 - f2 (pause_ratio)
- C1 - f3 (lip_sync_score)
- C2 - f4 (gesture_congruence)
- V4 - f5 (blink_anomaly)
- V4 - f6 (headpose_jerk)
- V5 - f7 (texture_anom)
- E3 - f8 (claim_support_ratio)
- E3 - f9 (median_source_reliability)
- E3 - f10 (asr_ocr_consistency)

```
flowchart TD IN[Input Video or URL] A1[A1 Demux and
AudioExtract] A2[A2 VAD and ASR] A3[A3 Audio Onsets] V1[V1
Keyframes and FaceTrack] V2[V2 OCR Overlays] V3[V3 Mouth
Landmarks TS] V4[V4 Blink and HeadPose] V5[V5 Texture or ELA]
C1[C1 Lip Sync Score] C2[C2 Gesture and Narration Check] C3[C3
Claim Extraction] E1[E1 Web Evidence] E2[E2 Source Reliability]
E3[E3 Claim Evidence Scorer] LR[LR Logistic Regression]
OUT[Probability Fake] UPDATE[LR Update on label] IN --> A1 IN --
> V1 A1 --> A2 A1 --> A3 V1 --> V2 V1 --> V3 V1 --> V4 V1 --> V5
A3 --> C1 V3 --> C1 A2 --> C3 V2 --> C3 C3 --> E1 E1 --> E2 C3 -
-> E3 E1 --> E3 E2 --> E3 A2 --> C2 V1 --> C2 A2 --> LR C1 -->
LR V4 --> LR V5 --> LR E3 --> LR C2 --> LR LR --> OUT LR -->
UPDATE
```



This should be the agentic graph...

Node description

▼ A1 (Demux + AudioExtract)

need: get the audio from the full video

input: the whole video in S3 bucket (maybe bytes format in S3?, we will see that)

otuput: 16 kHz audio with some simple metadata like sample rate/duration etc

how?:

we can simply use ffmpeg, ez (hopefully)

▼ A2 (VAD + ASR)

need: find where a person is actually speaking in the audio, and turn that speech into rough text with timings. this gives us “whos talking when” and a basic transcript

input: the output of A1 (the audio)

output: a kind of json with which word was spoken from what time to what time...

example:

if that guy says “hello world!”
the words are ‘hello’ and ‘world’

we will get stuff like → ‘hello’ was told from 12pm to 12pm 1 second and ‘world’ was told from 12pm 2 second to 12pm 3 second...

and also ‘segments’ of when this guy spoke....

so → this guy spoke from 12pm to 12 pm 1second, and stayed silent for 1 second and spoke again for 1 second

why are we doing all this bs?
we need the 2 features:

one is: speech rate, another is pause ratio

speech rate is intuitive, i dont have to explain

pause ratio is silent time vs total time

how to do it?

VAD: fullform is Voice Activity Detection!

basically this guy is looks at the audio energy and patterns and marks each short chunk as “speech” or “not speech.”

this shit is ready made (google meet kind of platforms use this to know when you’re yapping and stuff)

ASR: Automatic Speech Recognition

nothing to explain about this node

ready made af

we can chose to break this node down to 2 or 3 nodes depending on the load of this node

Why these features matter?
Fakes (like dubs or something or even fakes) can have odd pacing too fast, too slow, or strange gaps

▼ A3 (Audio onsets)

okay so to explain this guy we need to understand what we are doing with the node C1 and V3...
in V3, we are deriving a time series data that shows how open that guy's mouth is over time

and in C1, we are checking with the audio's spike over time and see if that shit matches with the mouth or not...

so this node basically does that spike part

it makes a time series data that creates a "spike over time" time series data...

A small time-series, e.g. a list like `[(t0, strength0), (t1, strength1), ...]`, where each strength is between 0 and 1 or something like that man we will see..

we should try to match the FPS of the video? question that we can discuss later

How do we do it??

so firstly, CUT DOWN the high and low freq sounds, maybe maintain the rough speech range, according to internet it is 300-3400 hz, cut off everything else

considering we are matching the FPS, there is a time gap between each frame, say 't'

so for each t seconds, we capture the whole audio stuff and do a loudness measure...

this should be a ready made stuff, since this type of forensics is common...

maybe also look at how the current 't' window is changing from the previous 't' window to capture the syllables

maybe normalize that shit to 0 to 1 and store in bucket

▼ V1 (Keyframes + FaceTrack)

Grab a few important frames from the video and keep track of the main face across those frames... tahts all!!

input is the actual video itself (not the audio)

convert this stuff to maybe a 1 FPS video (for light weight computation) or even 10 FPS, we will discuss... let the FPS be x FPS

so for each Frame in the x FPS video, calculate the region of interest (the guy's face basically) and crop it to that, we dont need any distractions...

how do we do this?

Either sample x frames per second or use scene change detection so we dont store too many images...

Run a lightweight face detector on each picked frame

this is also kind of ready made... we can run a CPU only framework

▼ V2 (OCR Overlays)

this guy basically reads on screen texts...

takes inputs from the reduced FPS video from V1

outputs a JSON of text that appeared on screen with the time stamp, confidence and the box of where it appeared (optional)

how can we do this?

maybe edit each image (pre processing stuff) to make the “text” STANDOUT in the image...

maybe use tesseract or something that can read on screen texts..

maybe we can clean the text and then store it in the json

Tesseract is ready made.. so not a problem

▼ V3 (Mouth Landmarks TIME SERIES)

Track how much the mouth opens and closes over time

this is what i was talking about in A3

takes inputs from V1 after face crop

now these cropped images, we can mesh the face, and measure the distance between the upper lip and the lower lip and normalize that

outputs the time series of this distance between lips (normalized)

for each frame we can store the openness of the mouth

▼ V4 (Blink rate + Head-pose dynamics)

basically get the oddness of blink rate and if the head movement is abnormal

take inputs from V1

need to use same mesh and landmarking algorithm to spot eyes

need to generate the 3 axis rotation values from this mesh, and then some sort of anomoly detection

this node generates 2 features, blink rate anomoly and headpose jerk

now how do we work with this node?

For each frame, compute EAR (Eye Aspect Ratio) from eye landmarks (a tiny formula that shrinks when the eyelids close).

When EAR drops below a threshold for a few frames, that's a blink. Count blinks and divide by clip minutes to get our blink rate

Turn that into a blink anomaly score by comparing to a typical range that we can check on the internet...

Use a few stable face points (nose tip, eye corners, mouth corners) run the standard solvepnp to estimate head orientation per frame

from this maybe compute frame to frame change of position, and from the "change" get a high percentile value, something between 90-95%le of abs change and and normalize by FPS

▼ V5 (Texture / ELA (compression/texture anomaly))

basicallt tell if the video has a a tampered face or not

takes inputs from V1 (cropped stuff)

outputs a feature ‘texture anom.’ , basically more tha value more sus the video is.

how do we do it?