

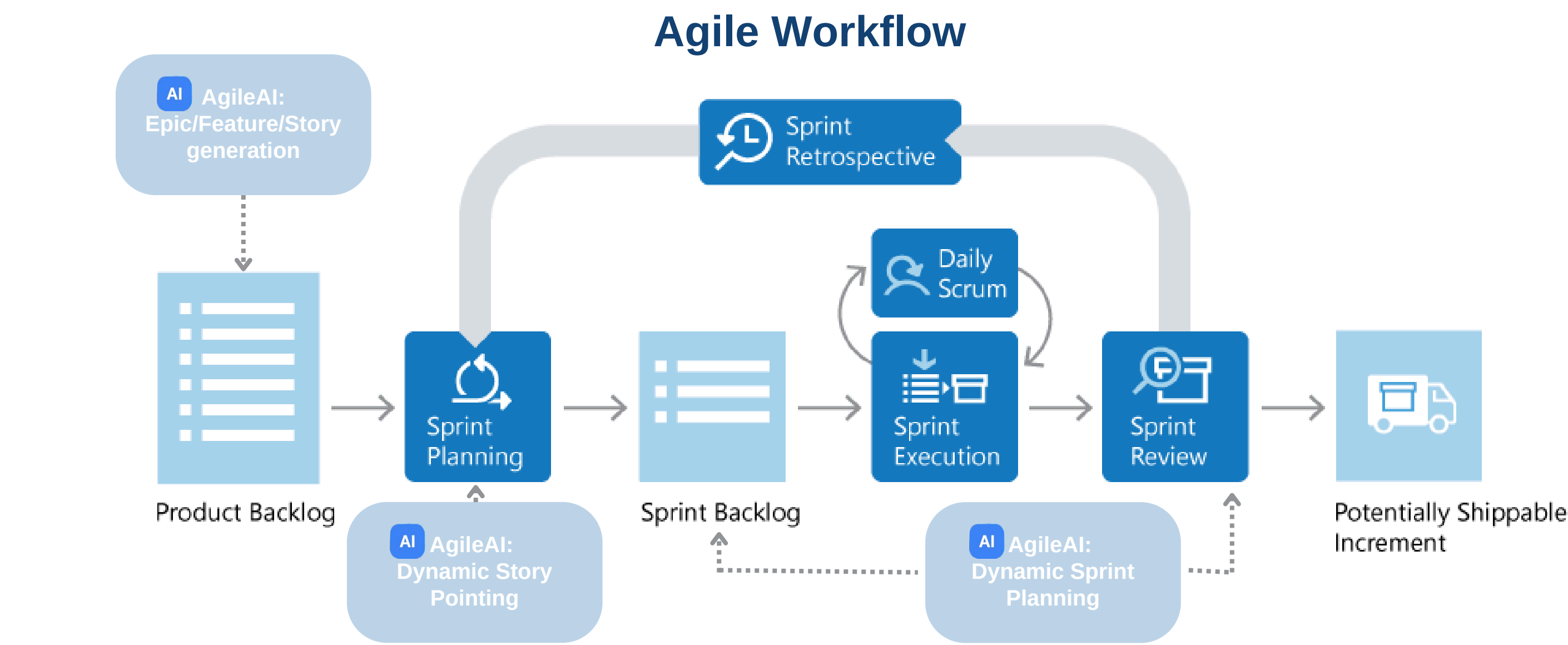
About AGILE-AI

Agile is the backbone of modern software development, yet breaking requirements into **Epics, Features, and User Stories** is still a slow, manual task. Teams often spend hours refining scope before any real development begins. This creates delays, inconsistencies, and a heavy cognitive load on project owners.

AgileAI eliminates that bottleneck by instantly converting project descriptions into structured Agile artifacts. It acts as a smart assistant to project manager, speeding up planning, improving clarity, and reducing manual effort. With this automation, organizations can move closer to the **5x development vision** by efficient and faster project management.

Architecture & Training

- **Objective:** Convert unstructured project PDFs into structured Agile artifacts (Epic → Features → User Stories) using an efficient LLM pipeline tailored for real-world project design automation.
- **Model Choice:**
 - Fine-tuned **TinyLLaMA-1.1B** using **Supervised Fine-Tuning (SFT)** with **QLoRA** and **8-bit quantization**.
 - Chosen for its low inference cost, fast training cycles, and edge-deployable footprint.
- **Dataset:**
 - 25 diverse project documents (**web, mobile, IoT, AI, data science**). This ensures broad generalization and minimizes domain bias.
- **Preprocessing:**
 - Extracted only requirement-critical text (scope, workflow, objectives).
 - Removed filler content (tables, references, UI mockups).
 - Benefits: focused context, lower token load, cleaner supervision signals.
- **Why Multi-Stage Prediction?**
 - Tiny models struggle with large outputs → high **hallucination** risk.
 - Breaking tasks into Epic → Features → Stories reduces complexity, improves reliability, and yields stable JSON outputs.
- **Prompt Engineering:**
 - Task-specific JSON templates for each stage.
 - Minimal, high-signal prompts using one-shot examples.
 - Improves structural reliability and reduces format errors.



Benefits of AgileAI:

- **50–70% Reduction in Backlog Creation Time:** Stakeholders confirmed AgileAI would significantly reduce manual effort, especially for large, multi-product teams. This translates to months of cumulative time saved per project cycle.
- **2–3x More Consistent & Audit-Ready Documentation:** Industry stakeholder emphasized that standardized epics, features, and stories help eliminate team-to-team variability. This consistency directly benefits SRE reviews, compliance audits, and cross-project knowledge transfer.
- **30–40% Improvement in Planning Insight via Intelligent AI Features:** Dynamic Story Pointing + AI-assisted Sprint Planning provide performance signals (e.g., workload patterns, velocity drift) would make the perfect recorded statistics for the team's performance evaluation.

AgileAI Flow Diagram



Evaluation & Results

Consistent Structured Generation

- Epic Confidence: 0.45
- Feature Confidence: 0.36
- Story Confidence: 0.33



High Coherence & Low Redundancy

- Evaluation metrics show strong linguistic stability with Pseudo-Perplexity ≈ 8.8

Bigram Repetition 22%

Trigram Repetition 9%

This indicates **low hallucination** and clean, non-repetitive outputs.

Integrated Feedback & Monitoring

- Real-time thumbs-up/down feedback + Prometheus/Grafana logging track model reliability, enabling measurable improvement through count-ratio accuracy of 1.0 for both features and stories.

Future Work

Azure DevOps Integration

Automate creation of Epics, Features, and User Stories directly into ADO using Azure REST APIs for a seamless end-to-end workflow.

Model Scalability & Accuracy Improvements

Expand training with larger, multi-domain datasets and explore more capable foundation models to increase generalization and reduce hallucination.

Dynamic Story Point Estimation

Incorporate historical sprint data to generate adaptive, data-driven story point predictions tailored to each team's velocity.

ML-Driven Sprint Planning

Build a dynamic sprint planning engine that recommends optimal sprint scopes, capacity balancing, and workload distribution.

Acknowledgements

I would like to express my sincere gratitude to **Dr. Andrea Ramirez Salgado** for her guidance and academic support throughout this project. I am also grateful to my peers for their constructive feedback and collaboration during the development process. Finally, I extend my appreciation to **Mr. Alex Karpman (BlackRock)** for providing valuable industry insights that informed and strengthened the design of AgileAI.

References

- <https://learn.microsoft.com/en-us/devops/plan/what-is-scrum>
- <https://github.com/bradtraversy/50projects50days>



ECG Analysis
Built to help Doctors!

AI-Powered ECG Signal Processing and Arrhythmia Detection System

Honey Reddy Daram



github



demo

GAP

Doctors currently rely on slow, manual, and fragmented ECG review processes, which limits speed and consistency in clinical decision-making.

SOLUTION:

A unified automated system that processes, analyzes, and classifies ECG signals to support faster and more consistent clinical decisions.

01. ABSTRACT

This project builds an end-to-end system for ECG signal processing and arrhythmia risk detection. It preprocesses raw ECG data, detects PQRST peaks, extracts features, and classifies Normal, Moderate Risk, or High Risk using a CNN model. The system achieves 98.84% accuracy, 98.84% weighted F1, 0.9826 MCC, and 0.9993 ROC-AUC, and is deployed through a Streamlit app for easy visualization and clinician review.

02. Objective

To build an automated ECG processing and risk classification system that helps doctors make faster and more accurate clinical decisions.

PhysioNet

<https://physionet.org/content/mitdb/1.0.0/>

Database Open Access

MIT-BIH Arrhythmia Database

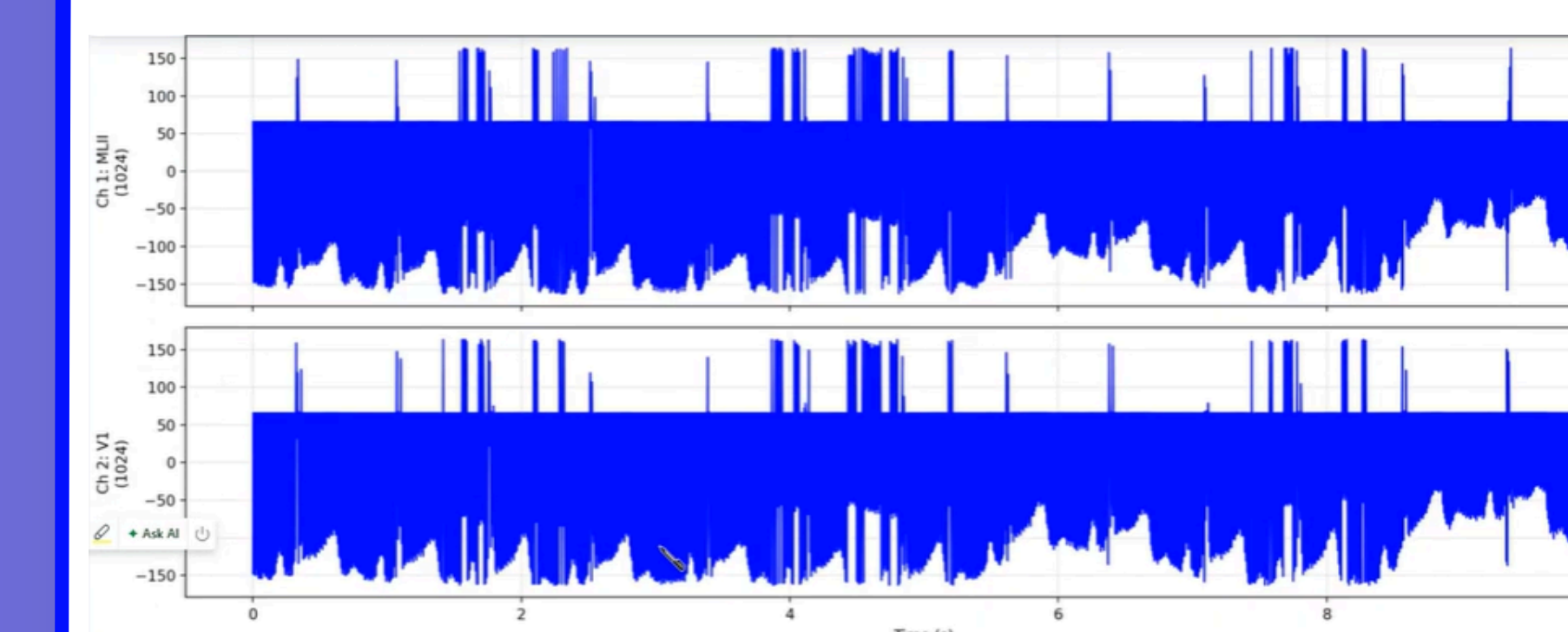
George Moody, Roger Mark

Published: Feb. 24, 2005. Version: 1.0.0

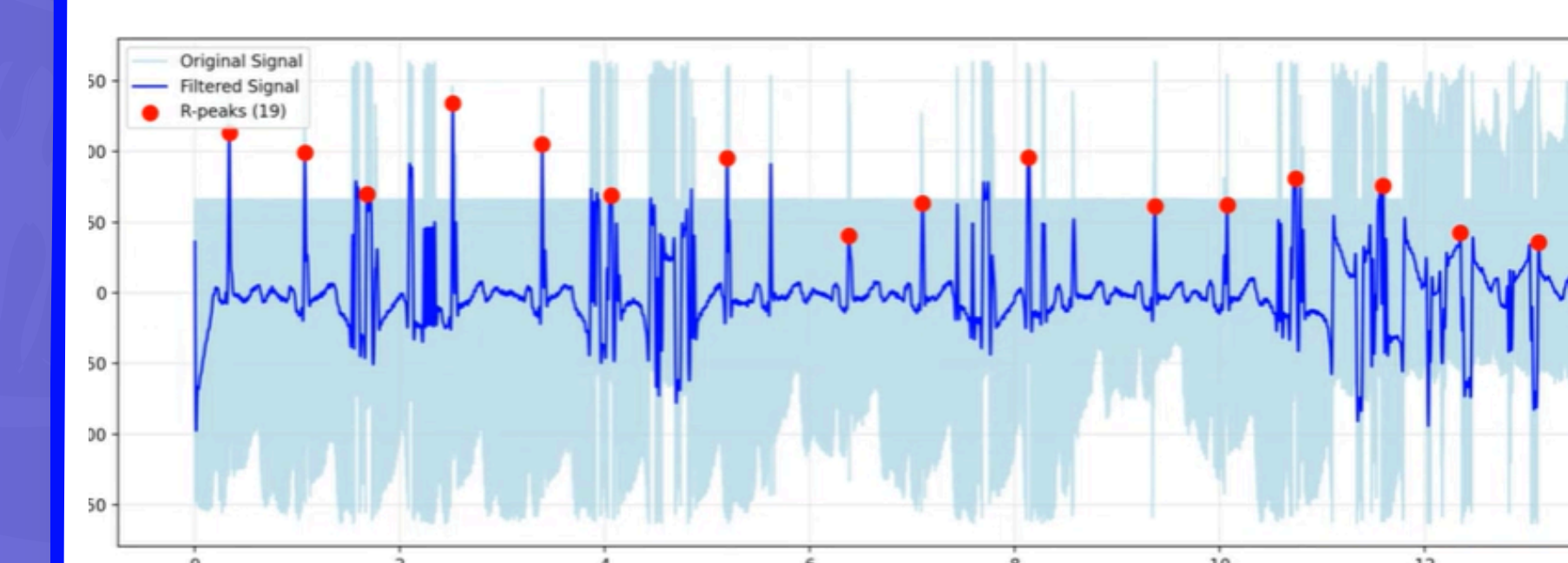
Workflow

Data → Preprocess Segment →
Label/Prepare Train → Calibrate
Evaluate → Deploy

03. Methodology

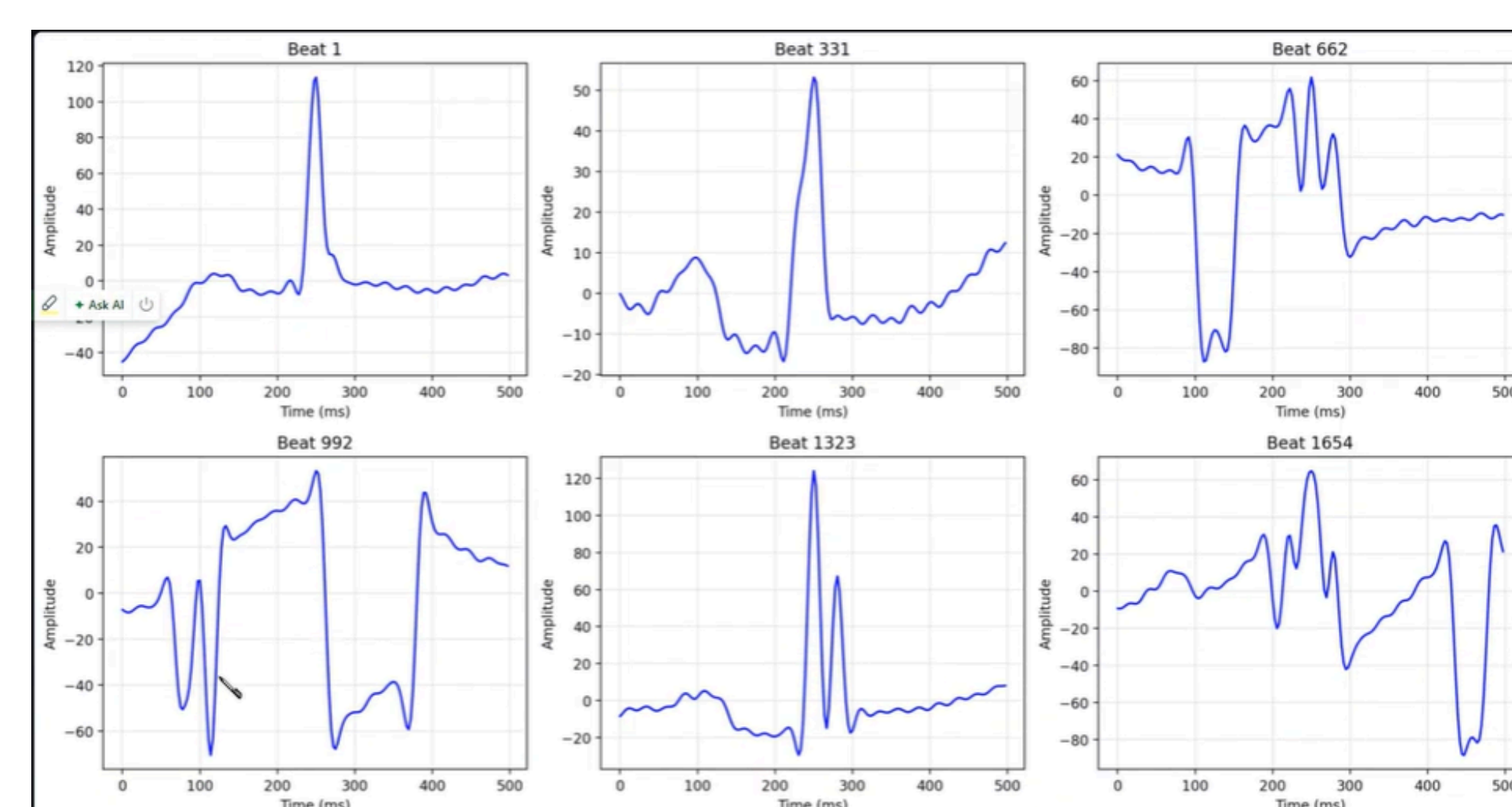


Raw ECG from two leads (10 s) showing noise spikes and baseline drift—motivating filtering before reliable R-peak detection and analysis.

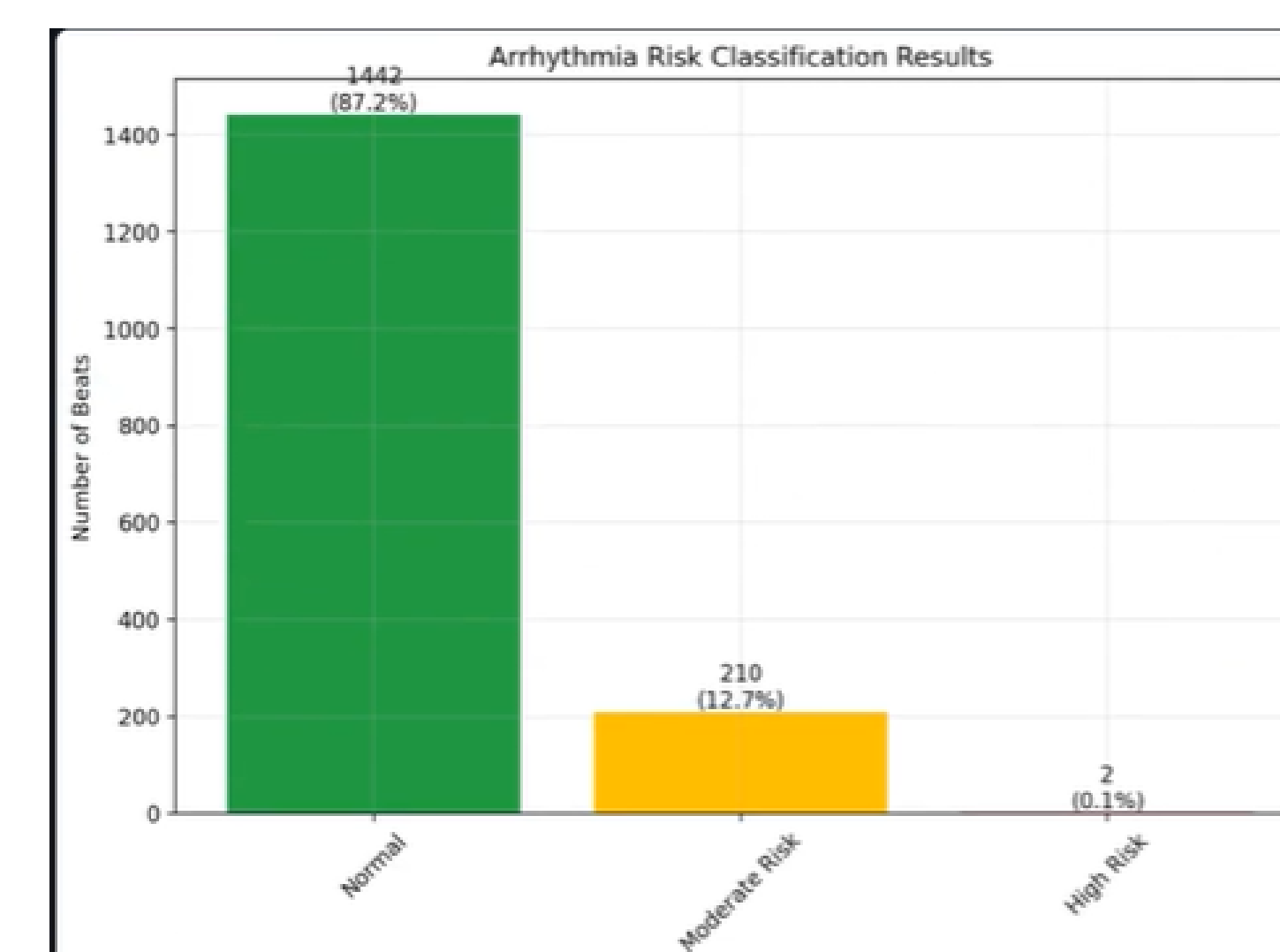


Enhanced ECG analysis showing the bandpass-filtered waveform (blue) with detected R-peaks (red) overlaid on the original signal, highlighting clearer beat detection and more reliable interval measurement.

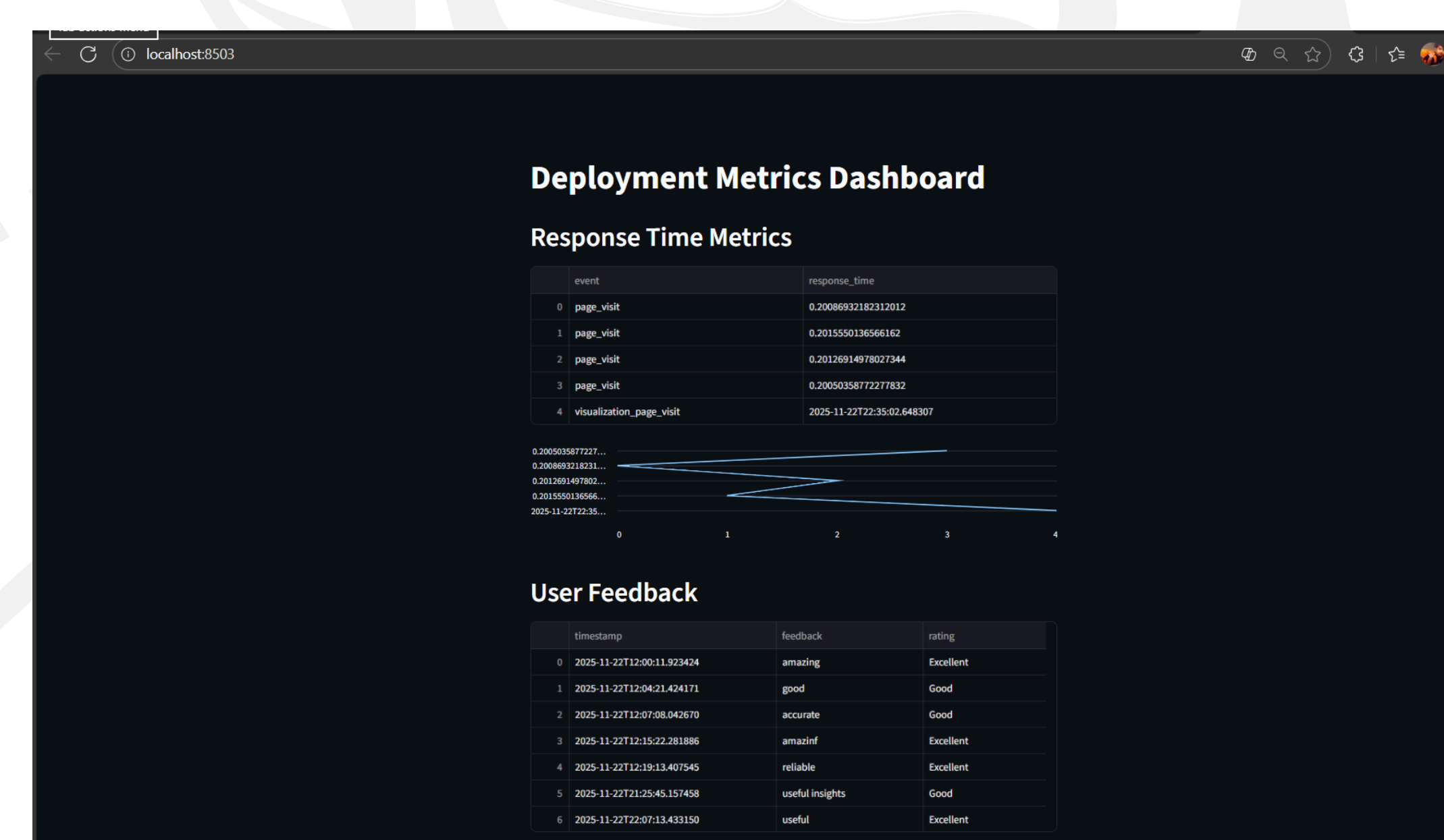
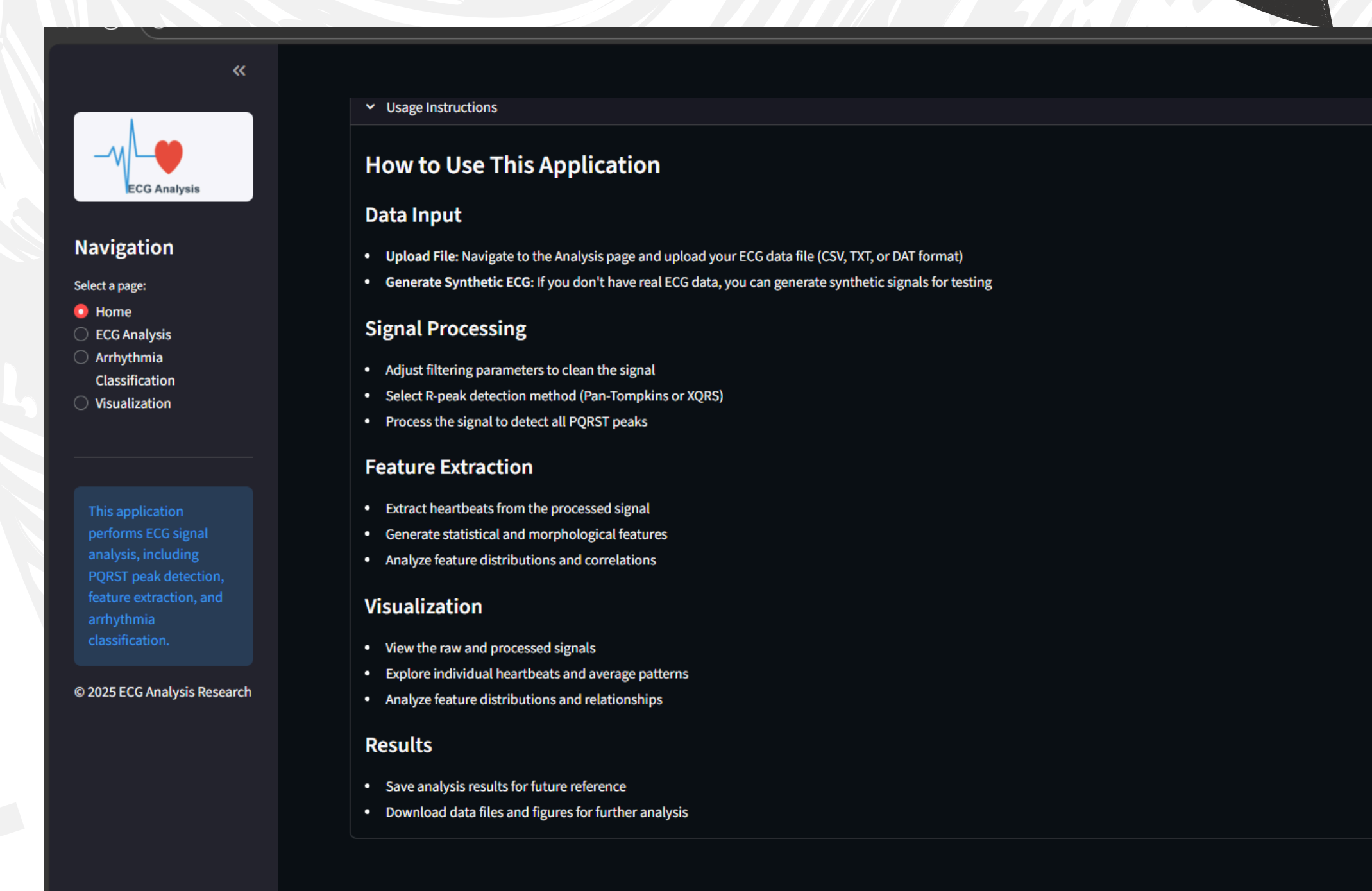
04. RESULTS



- Model Accuracy: 0.988
- F1 Score (Macro): 0.9884
- ROC AUC: 0.9993
- Balanced Accuracy: 0.9884
- Matthews Correlation: 0.9826
- Cohen's Kappa: 0.9825



bar chart summarizes model outputs with Normal dominant (~87%), Moderate risk (~13%), and High risk rare (<1%), reflecting calibrated, conservative risk labeling.



REFERENCES

- [1] "A real-time QRS detection algorithm," IEEE Transactions on Biomedical Engineering.
- [2] PhysioBank, PhysioToolkit, and PhysioNet: Components of a new research resource for complex physiologic signals
- [3] "The impact of the MIT-BIH Arrhythmia Database," IEEE Engineering in Medicine and Biology Magazine
- [4] SMOTE: Synthetic minority over-sampling technique
- [5] Adam: A method for stochastic optimization
- [6] Batch normalization: Accelerating deep network training by reducing internal covariate shift
- [7] Keras
- [8] Streamlit: A faster way to build data apps

AUTOMATED COMPONENT INSPECTION SYSTEM (ACIS)

End-to-End AI Lifecycle Management for Automotive Quality Control

Authors

Rudra Patel

Affiliations

University of Florida



Scan to view
GitHub
Repo

Introduction

The Challenge High-variation automotive assembly lines currently rely heavily on manual inspection. This dependence creates significant bottlenecks, increases safety risks, and leads to costly human errors in verifying complex dashboard components.

The Solution: End-to-End Lifecycle Management

To tackle this issue, I created the Automated Component Inspection System (ACIS). This system manages the entire AI lifecycle, from data collection to deployment. It uses YOLOv8n for real-time inference, integrates Docker containerization for easy scaling, and employs k-anonymity to protect worker privacy. ACIS replaces subjective manual checks with objective computer vision. This ensures high throughput and reliable industrial performance.

Methodology

- Model: Selected YOLOv8n (Nano) for its lightweight architecture and real-time inference capabilities, suitable for edge deployment.
- Training: Trained on Google Colab using an NVIDIA A100 GPU for 100 epochs.
- Data Processing: Utilized a custom dataset of ~2,000 images, augmented with rotations ($\pm 15^\circ$) and brightness changes ($\pm 20\%$) to mimic factory conditions.
- Privacy: Applied k-anonymity to metadata to remove operator IDs and timestamps.

Results/Findings

- Accuracy: Achieved 99.5% mAP@0.5 on the validation set.
- Speed: Inference latency of 0.3ms, far exceeding the <300ms target.
- Reliability: 99.4% Recall, ensuring safety-critical incorrect parts are effectively caught.

IMPORTANT!

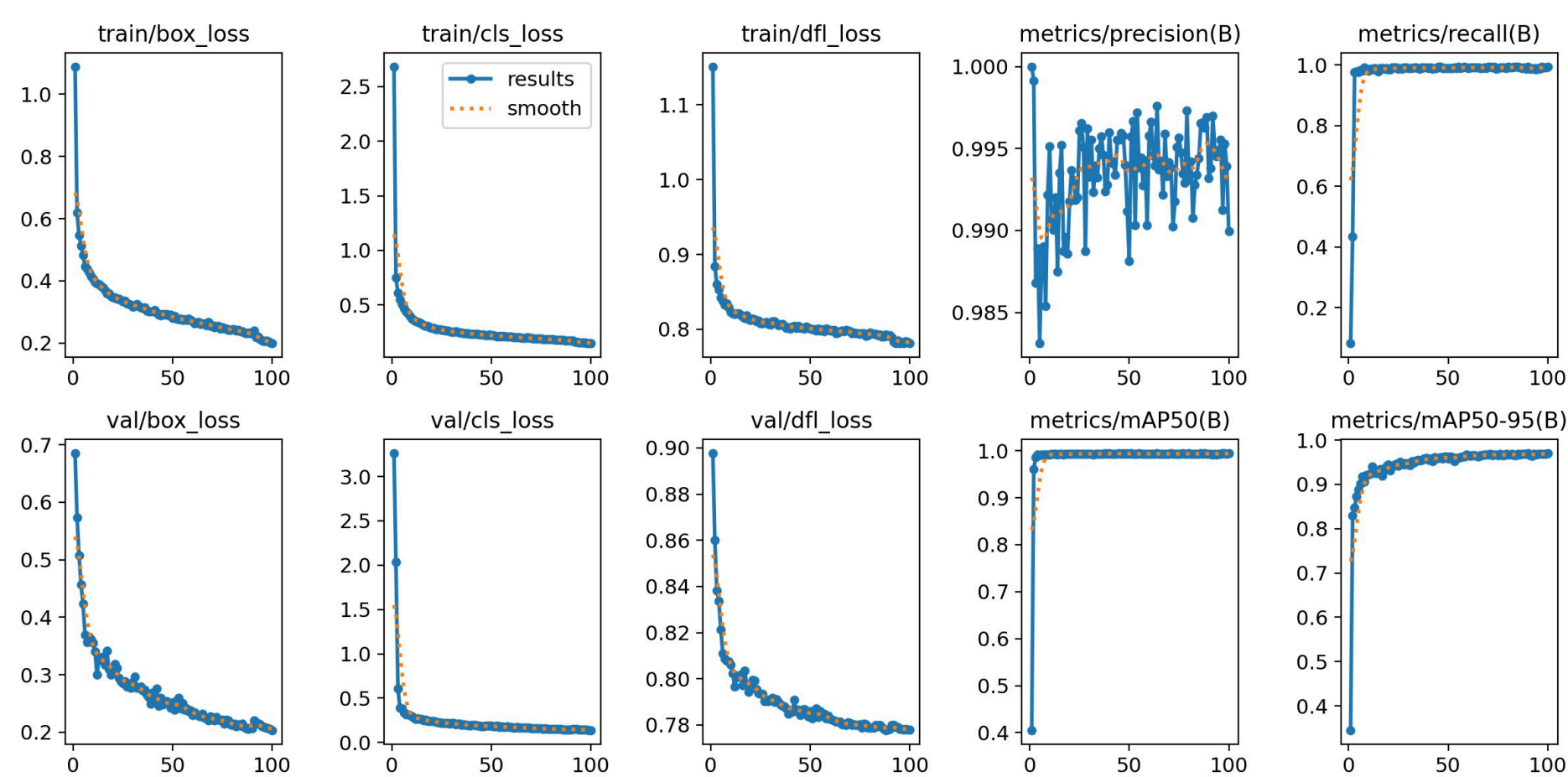
Training was highly efficient, completing in just ~13 minutes on the A100 GPU.

Objective

- Goal: To instantly confirm the correct part is installed, eliminating manual errors.
- Benchmarks:-
- Accuracy: $\geq 95\%$
- Latency: 200–300 ms
- Throughput: 3–5 images/sec

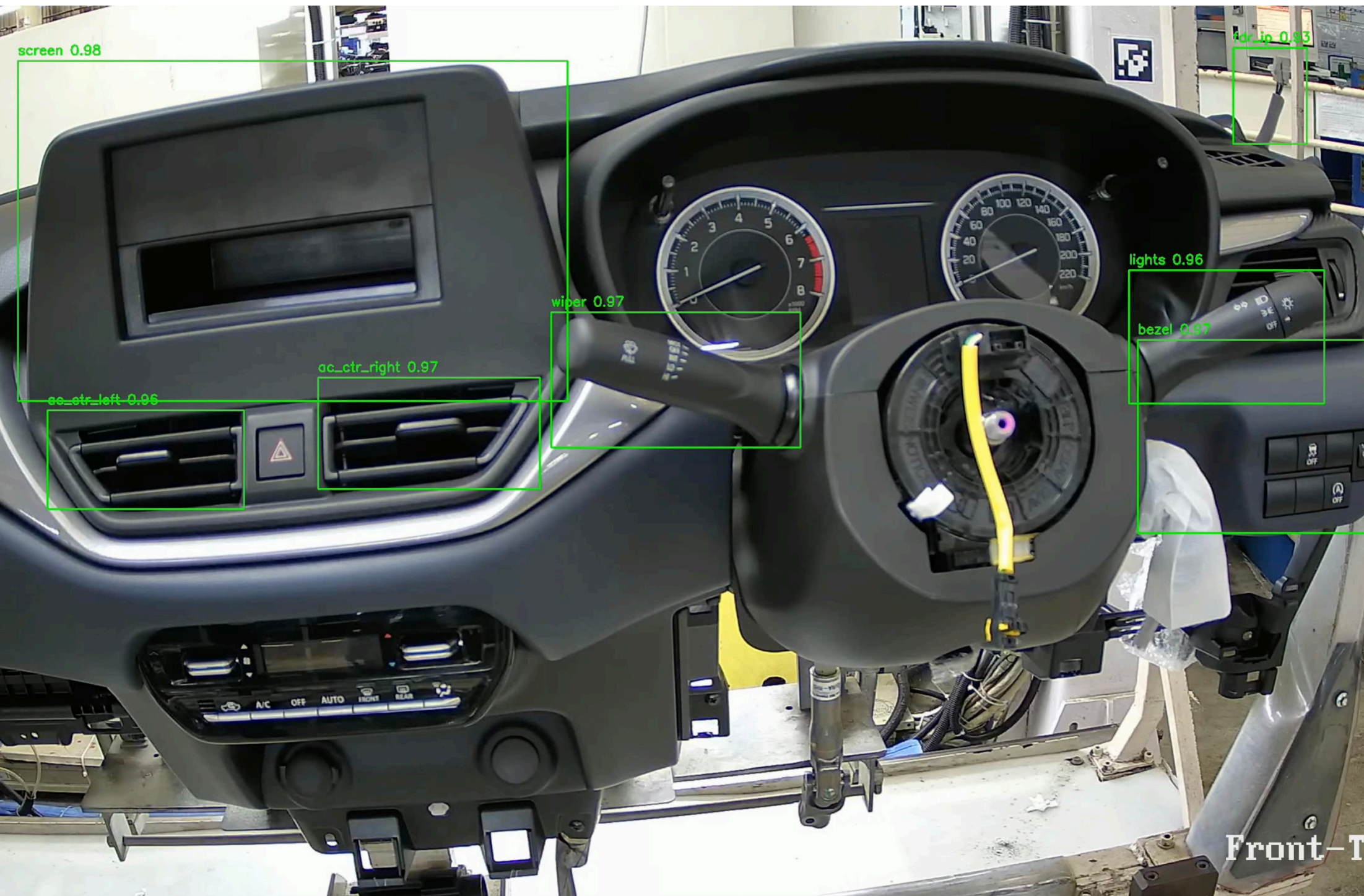
Analysis

Performance evaluation against strict automotive benchmarks confirmed the system's industrial viability. The model significantly outperformed baseline targets, ensuring both high accuracy and operational reliability in a production environment.



Training metrics over 100 epochs. Rapid convergence seen in loss curves (left) and consistently high mAP@0.5 scores (bottom right) validate model stability and accuracy

Real-time localization of dashboard components. Bounding boxes confirm the correct presence of Screens, Wiper,Lights, and Bezel with >95% confidence during the assembly process.



Conclusion

- Summary: ACIS successfully automates verification with >99% accuracy.
- Impact: The system increases throughput and lowers quality control costs.
- Future Work: Plan to transition from cloud deployment to Edge devices to reduce network dependence and latency.

FINE-TUNING QWEN2-VL-2B FOR ORAL CANCER HISTOPATHOLOGICAL IMAGE CLASSIFICATION

Anshul Patil



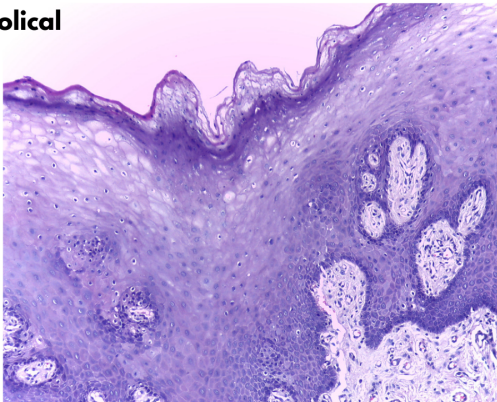
Department of Engineering Education

Master of Science in Applied Data Science

Introduction

The objective of this project is to fine-tune a multimodal vision-language model, Qwen2-VL-2B, for the classification of oral histopathological images into Normal and Oral Squamous Cell Carcinoma (OSCC) categories. Unlike traditional convolutional neural networks (CNNs), Qwen2-VL2B integrates both visual and textual modalities, allowing for joint reasoning over image content and associated medical context. The model is fine-tuned using a curated dataset of histopathological slides, enabling it to understand complex tumor morphology and associate it with descriptive textual cues. This work aims to improve diagnostic accuracy and interpretability in digital pathology applications, contributing toward clinically reliable multimodal AI systems.

Normal Histopathological Slide



Data processing and Image captioning

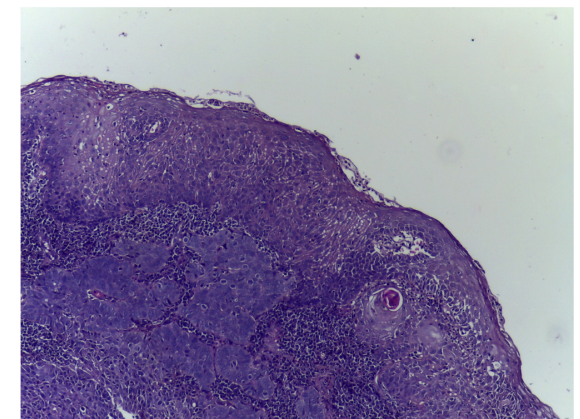
- Used the BLIP model to generate initial image captions.
- Cleaned the raw BLIP captions by fixing spacing issues and removing unwanted text.
- Split each caption into separate question and answer fields using string parsing.
- Removed entries where the split failed or the caption format was incorrect.
- Verified that every image had a valid caption and removed mismatched or missing pairs.
- Saved the filtered dataset in a clean CSV format for model training and evaluation.

Model Finetuning

- 01 >> Loaded the Qwen2-VL-2B variant via Unsloth's FastVisionModel API (the exact checkpoint used during experiments was the Unsloth Qwen2-VL instruct build). The model was run in quantized mode (4-bit) to fit GPU memory constraints.
- 02 >> Converted the dataset to the Unsloth chat format required for training: each example became a messages entry where the user content is "\n" and images were passed separately (no nested {type:image} in the text). This ensured token ↔ image alignment for the custom processor.
- 03 >> Applied the same image preprocessing used during caption creation and training: convert to RGB, reduce resolution (resize), and load as PIL images before passing to the processor. Captions used BLIP-2 outputs framed as QA prompts (e.g., "Question: Explain why this image is classified as ... Answer:") and were cleaned/split into question/answer pairs prior to fine-tuning.
- 04 >> Enabled LoRA-based PEFT with the same LoRA configuration used in experiments: $r=16$, $\text{lo_ra_alpha}=16$, $\text{lo_ra_dropout}=0$, $\text{bias}=\text{"none"}$, and $\text{use_rslora}=\text{False}$. Only LoRA layers (attention/MLP projection targets) were updated; the base model parameters remained frozen. Random seed and reproducibility parameters were set (seed 3407).
- 05 >> Switched the model into training mode using `FastVisionModel.for_training(...)` and used Unsloth's `UnslothVisionDataCollator + trL.SFTTrainer` for the optimization loop.

Results

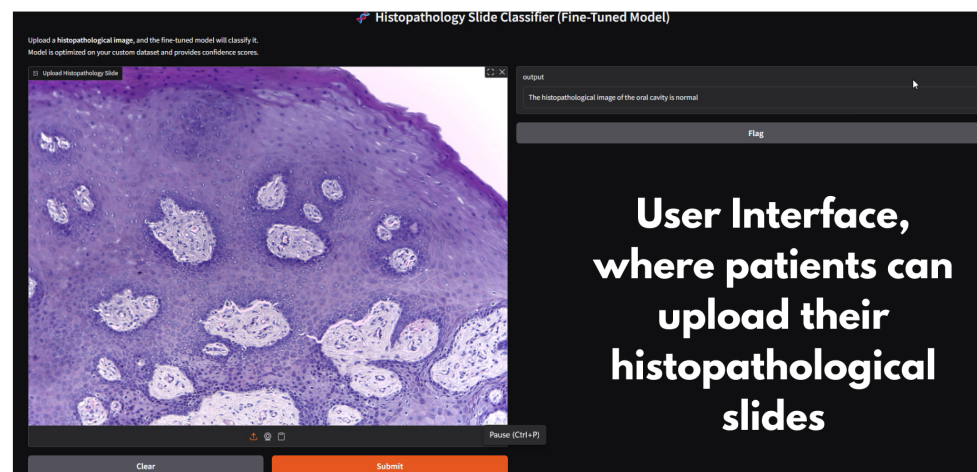
- The model was evaluated before and after fine-tuning using BERTScore and validation accuracy. Pre-fine-tuning, it produced reasonable explanations but showed inconsistent alignment with ground-truth captions (lower precision, recall, and F1). After fine-tuning on the BLIP-generated QA dataset, all BERTScore metrics rose sharply to ~0.96, indicating much more accurate and contextually aligned explanations.
- Validation accuracy was 100% both before and after fine-tuning, showing that label classification was already easy for the pre-trained model. Fine-tuning primarily improved the quality, specificity, and consistency of textual explanations while reducing output variability and better aligning reasoning with domain-specific patterns in oral histopathology images.



Oral squamous cell carcinoma

Conclusion

This project presents a full multimodal pipeline for oral histopathology image classification (Normal vs. OSCC) using a fine-tuned Qwen2-VL-2B model. The pipeline covers dataset cleaning, BLIP captioning, QA-pair generation, efficient LoRA fine-tuning with Unsloth, and a Gradio demo for real-time inference. Fine-tuning boosted explanation quality (BERTScore F1: 0.88 → 0.96) while keeping 100% validation accuracy. The model now delivers accurate class predictions with clearer, more precise, domain-specific diagnostic reasoning. A simple web interface enables direct image uploads and instant explainable results, highlighting how fine-tuned vision-language models can provide both high accuracy and interpretability in medical imaging tasks.



Introduction

The integration of voice recognition into drone control systems could open up an entirely new frontier of autonomous machine-human interaction. This experimental prototype combines Wav2Vec2, a transformer-based deep learning architecture with microcontrollers to create the core of **AURA** (**A**udio-**U**nderstanding **R**obot for **A**erial navigation). The goal of this prototype is to showcase the conceptual viability of drone automation using speech commands as well as demonstrate a clear path of evolution for future real world application.

Model Training

AURA was trained on the Google Speech Commands dataset—a widely used benchmark for keyword spotting—to classify short one-word speech commands such as “up”, “down”, “left”, and “right” as well as numbers 1 - 9. For model initialization, using the pretrained weights associated with Wav2Vec2 expediated the training process due to their already high latent representation of language. The end-goal output of the custom-trained model was to be able to extract the correct command from 1 second audio clips. Since the original Wav2Vec2 model does not match with the desired output, a linear layer head is attached to the output of the Wav2Vec2 model to allow the model to output the appropriate classes for this system.

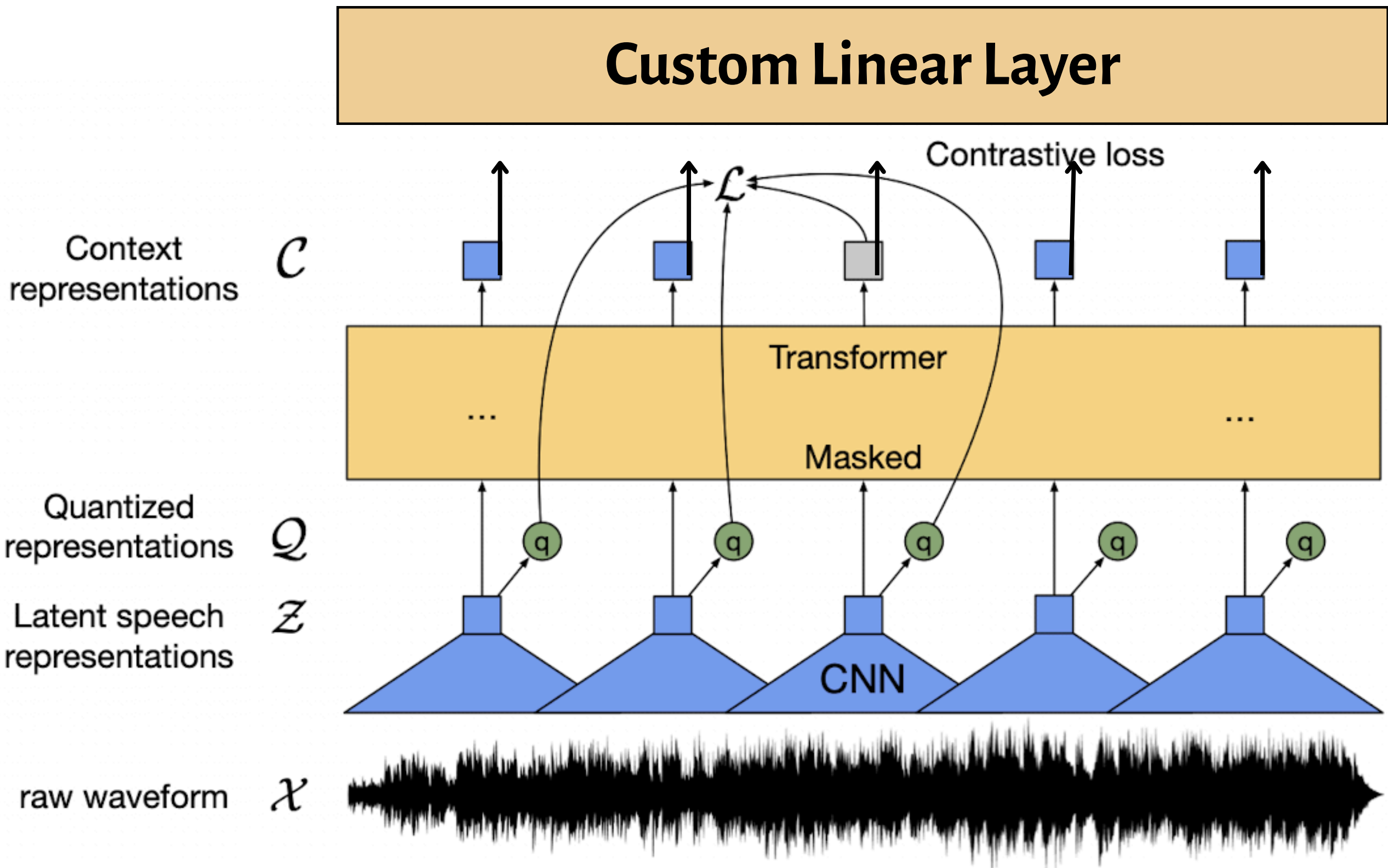
Results/Evalutation

AURA was able to successfully takeoff. The demo video demonstrates the drone’s ability to maneuver through a simple obstacle course. Due to the limited constraints of what is outputted from the voice recognition model, the drone was controlled by utilizing two keywords for each movement: direction and time. The input was configured so that the drone could know which direction to move in and for how long to move in that direction.

Real World Applications

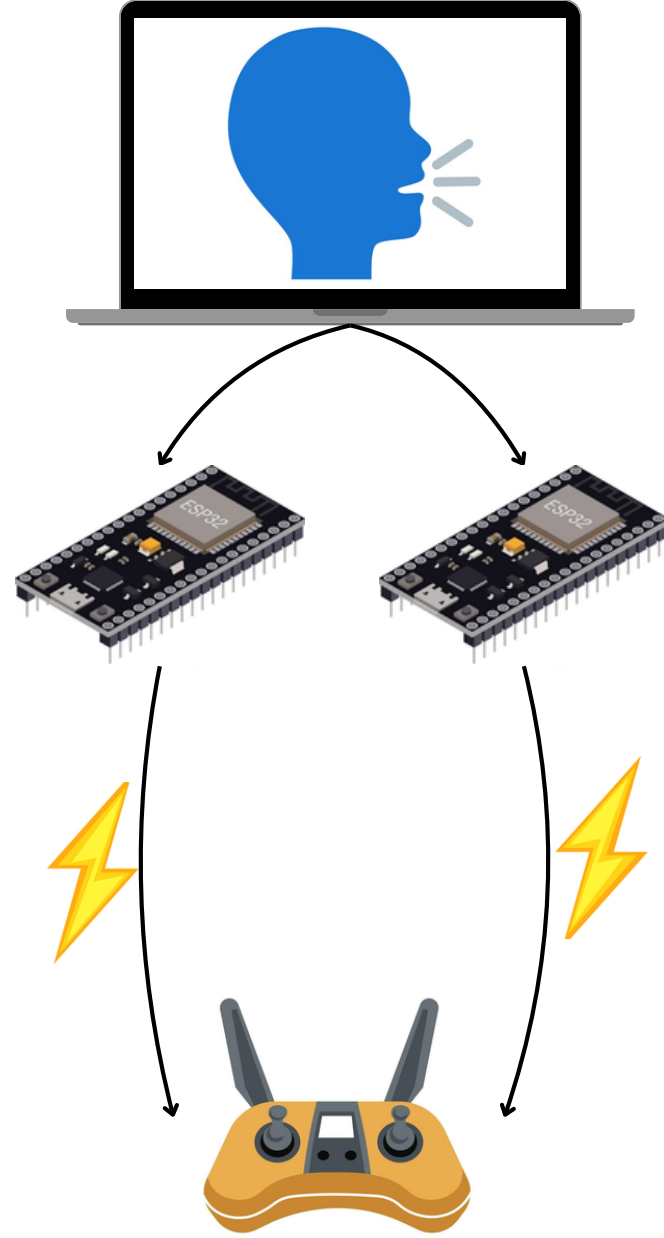
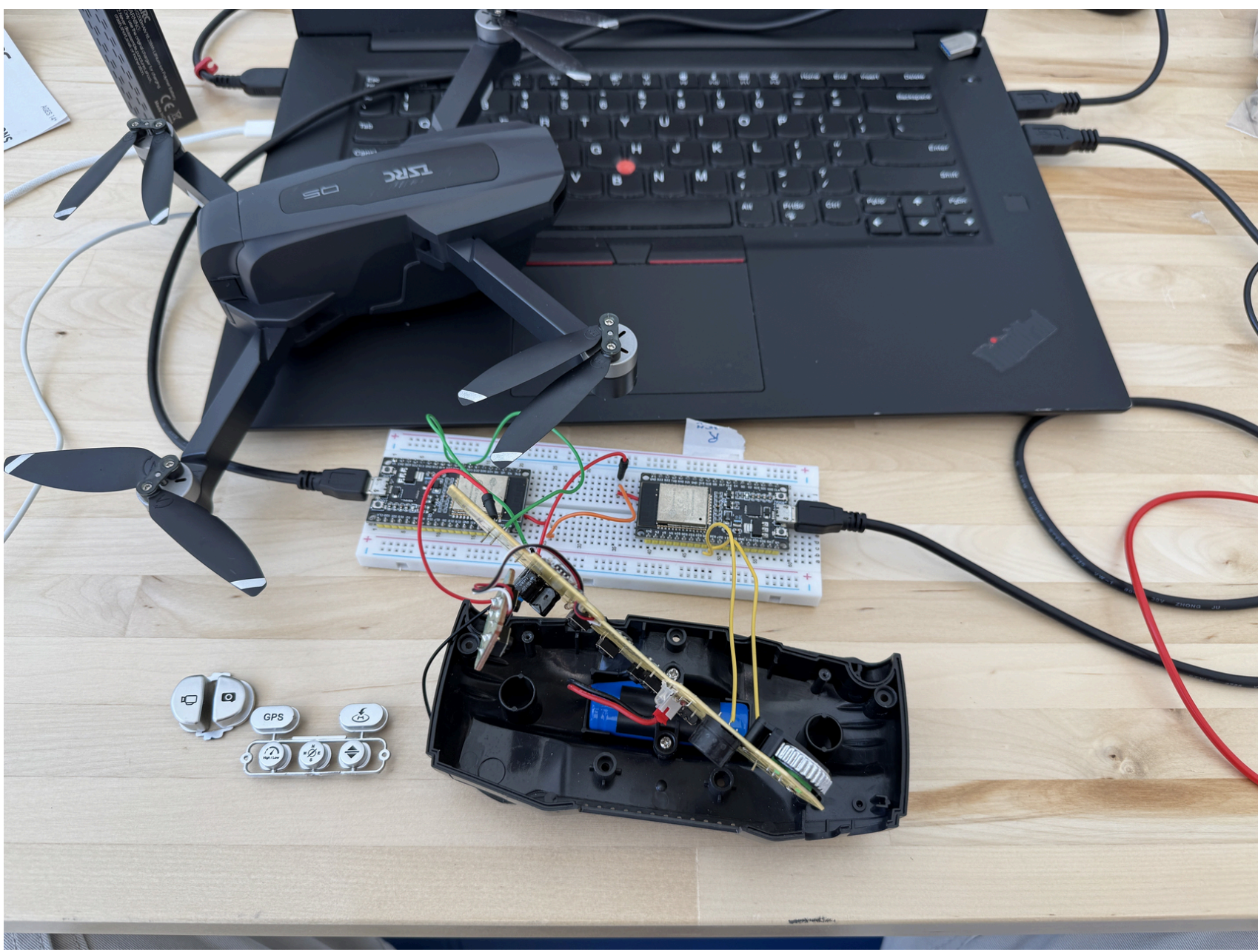
AI drones are the future. The next evolution for **AURA** would be the incorporation of LLM’s. This would make **AURA** much more versatile in what it can understand and expand its ability to complete tasks from the user. Many industries that utilize drones such as agriculture and security would be able much more efficiently deliberate tasks to drones without having to manually control the drones themselves such as crop fire detection or area surveillance.

Wav2Vec2 Architecture



There are 2 main components that make up the backbone of Wav2Vec2, its convolutional encoder layer and its deep transformer layer. The CNN layer transforms the raw audio data input into latent speech representations. These latent representations are then processed by the transformer to create contextualized representations of the entire sequence. These are then mean pooled to get a single average vector which is then put into a feed-forward layer where each logit (class) is given a probability determining which class of words the audio is.

Hardware Architecture



The image on the left is what the hardware component of the **AURA** system looks like while the diagram on the right is a more simplified version of **AURA**. Based on the command selected by the user, both microcontrollers were flashed to be able to handle the varying inputs. Each ESP32 was responsible for controlling the left and right joystick of the drone. Since the RC measured the position of each joy stick through a range of voltages, depending on the command sent by the model, the voltages outputted by each microcontroller ranged between 0 to 3.3 V. As for why there were two ESP32’s included within this system, it is because each ESP32 has only two Digital to Analog Converters (DACs). A DAC pin is responsible for turning a digital signal from the ESP32 to an analog signal for the remote control to read. Each DAC pin controlled the voltage of either the horizontal or vertical component of the joystick.

References

[1] Baevski, Alexei, et al. “WAV2VEC 2.0: A Framework for Self-Supervised Learning of Speech Representations.” arXiv.Org, 22 Oct. 2020, arxiv.org/abs/2006.11477.
[2] n.a. “Digital to Analog Converter (DAC).” Espressif, 2022, docs.espressif.com/projects/esp-idf/en/stable/esp32/api-reference/peripherals/dac.html.

YouHue: AI-Powered Personalized Lipstick Recommendations via Two-Stage Complexion Matching

¹Davis Carson
Master of Science in AI Systems
daviscarson@ufl.edu



The Problem

Industry Context:

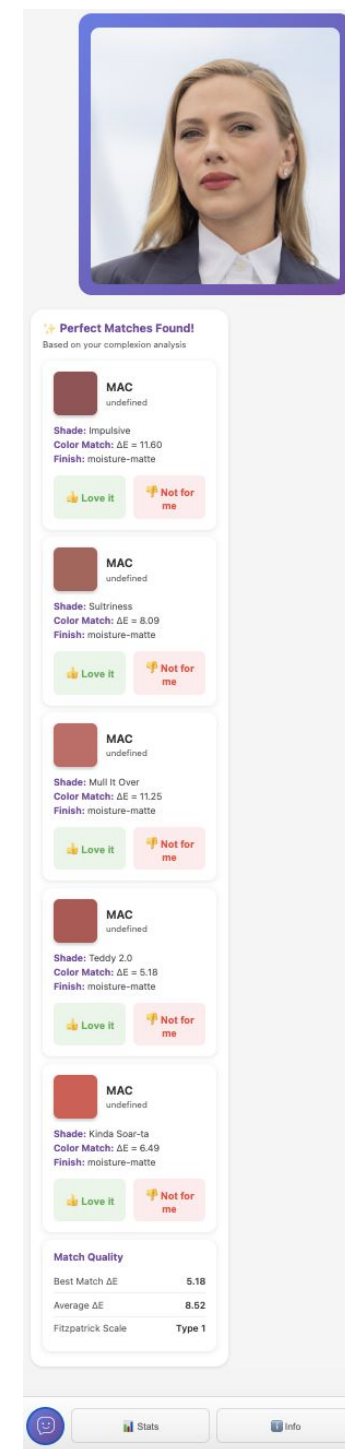
- Global cosmetics market: \$500B+ with rapid digital transformation
- 73% of consumers dissatisfied with online makeup recommendations
- Documented racial bias in beauty tech algorithms

Technical Challenges:

- Traditional recommendations ignore individual complexion variations, lighting, and geometry
- Existing solutions lack scientific color matching (RGB vs. perceptual color spaces)
- Privacy concerns with face data retention in commercial apps
- Need: Sub-500ms latency for real-time user experience

Solution - Full System Lifecycle:

- Research → Development → Production Deployment → Real-time Monitoring
- Privacy-first, fairness-aware AI system for personalized cosmetics
- Mobile-first web interface on FastAPI backend (port 8001)
- Comprehensive evaluation: performance, fairness, accuracy metrics



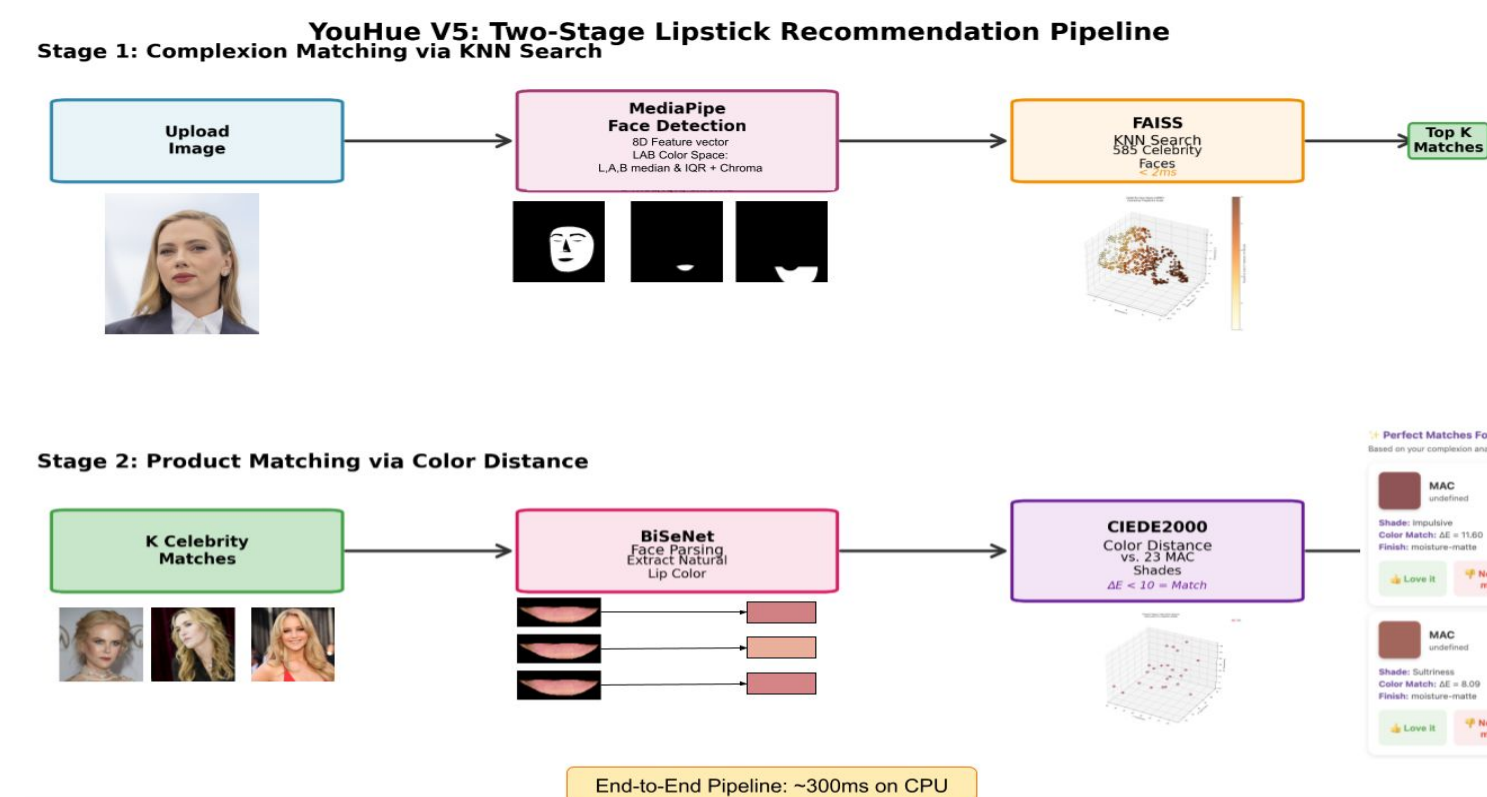
Approach & Technical Architecture

Stage 1: Celebrity Face Matching

- MediaPipe face detection
- 8D complexion features (LAB color space: L_med, a_med, b_med + IQR)
- FAISS K-NN search (~2ms for 595 celebrities)

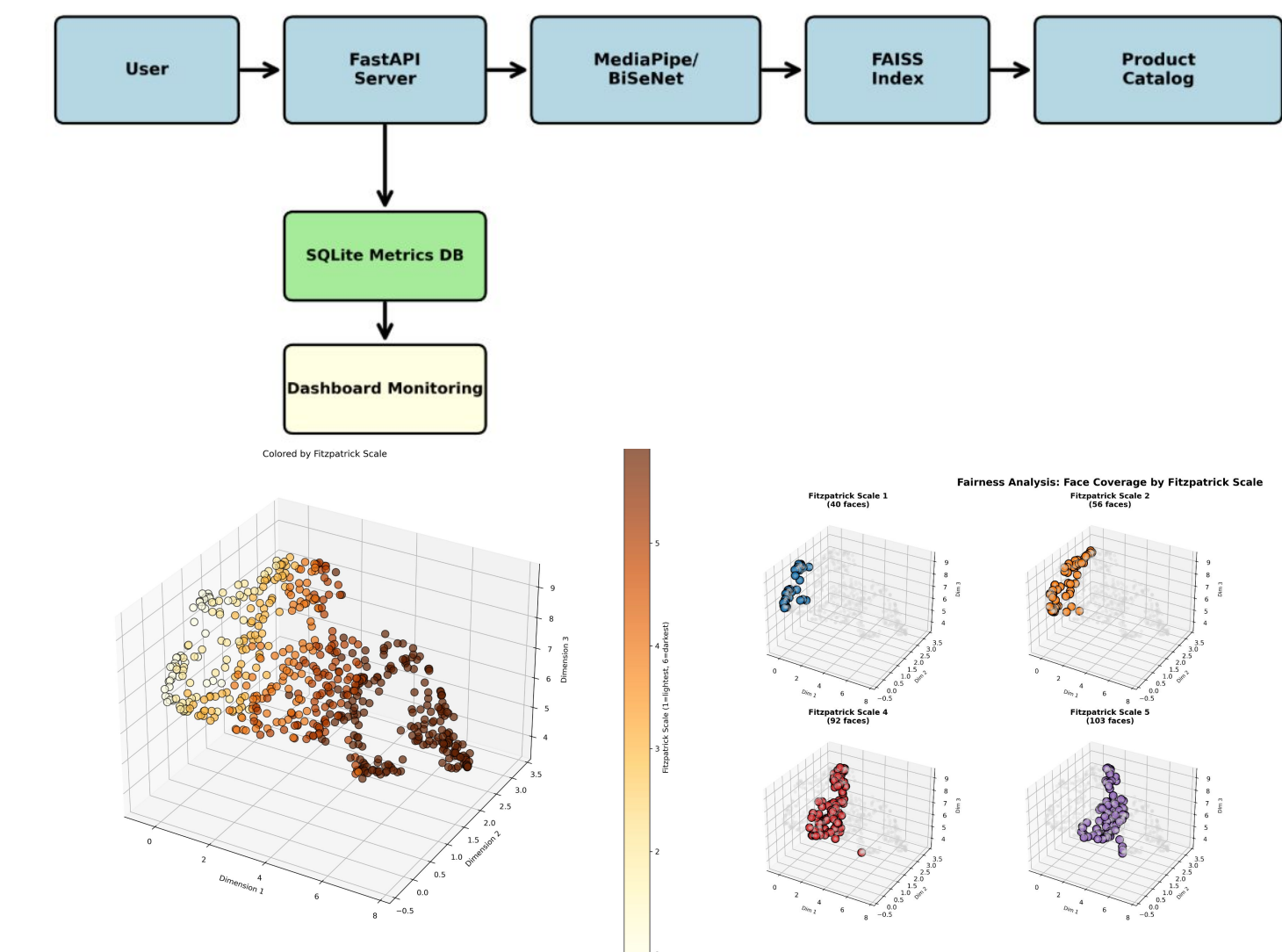
Stage 2: Product Matching

- BiSeNet lip segmentation
- Natural lip color extraction (LAB)
- CIEDE2000 color distance ($\Delta E < 10$)
- Smart deduplication for K unique products

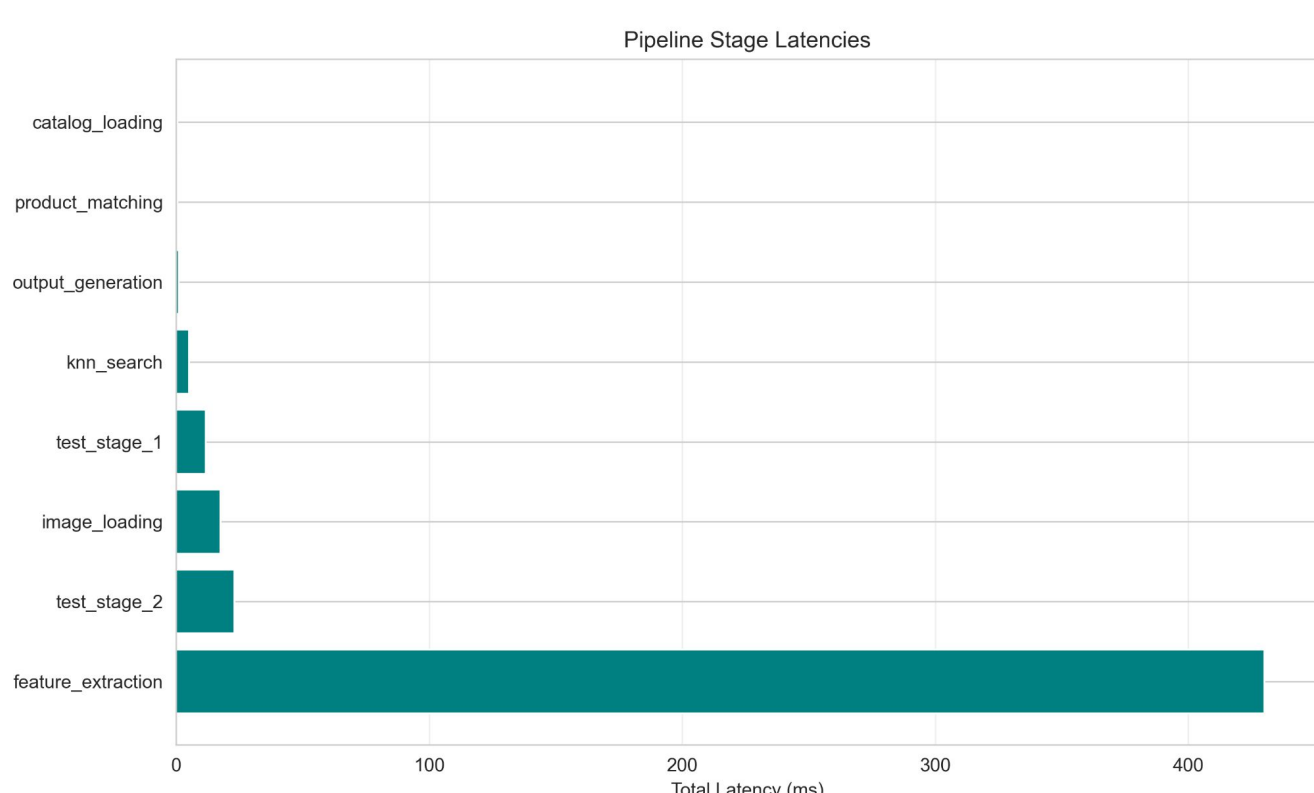


Data Management

- FastAPI server (port 8001)
- SQLite metrics database
- Privacy-by-design: immediate image deletion, EXIF stripping
- Real-time dashboard for monitoring

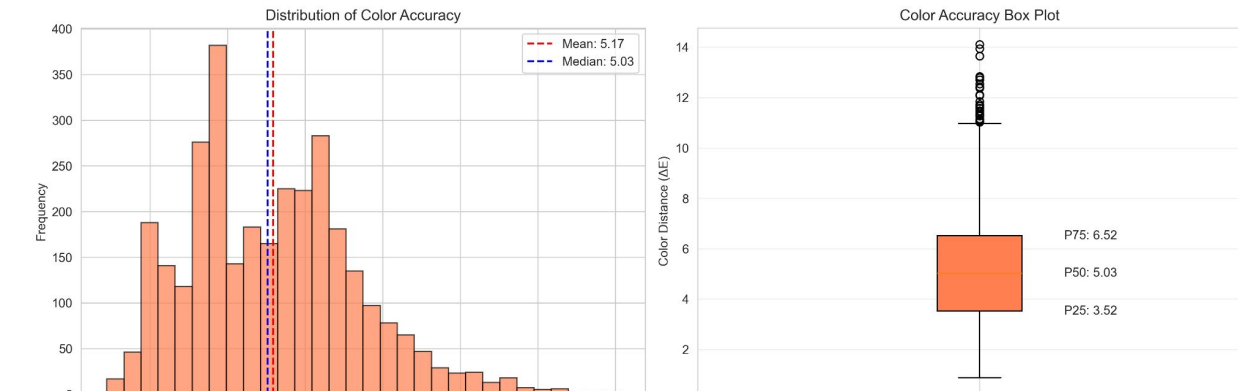


Results & Evaluation



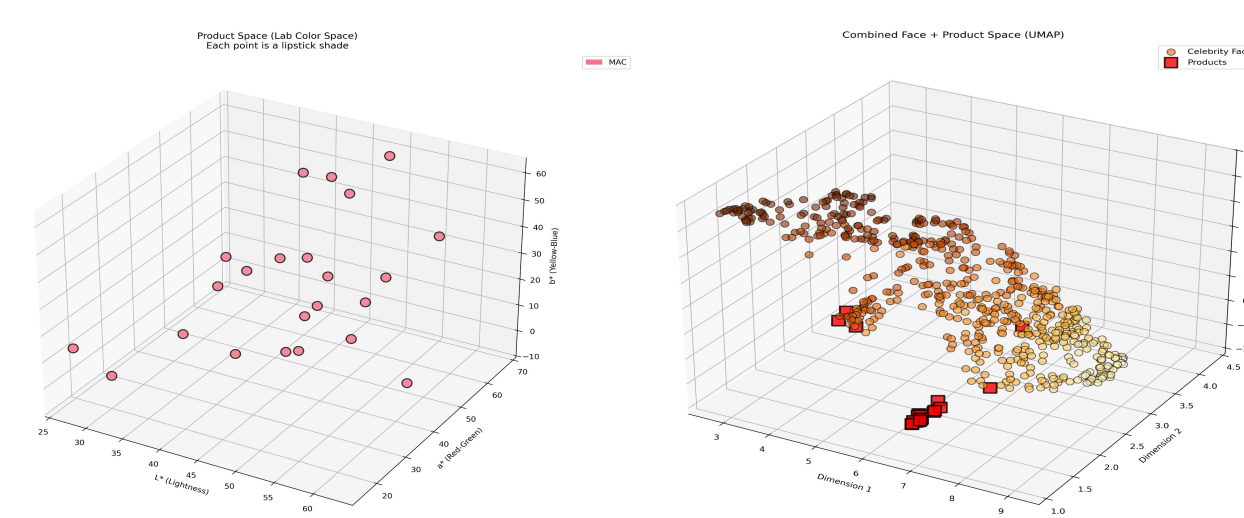
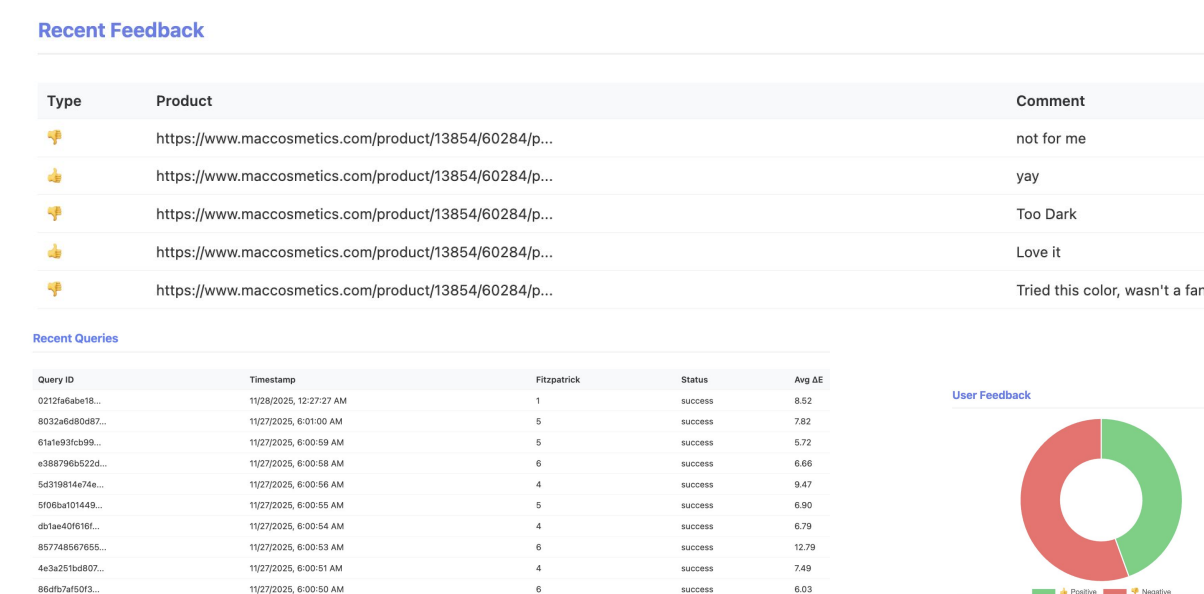
Accuracy:

- Color matching: ΔE threshold < 10 (perceptually similar)
- Database: 585 celebrity faces, 28+ MAC lipstick products



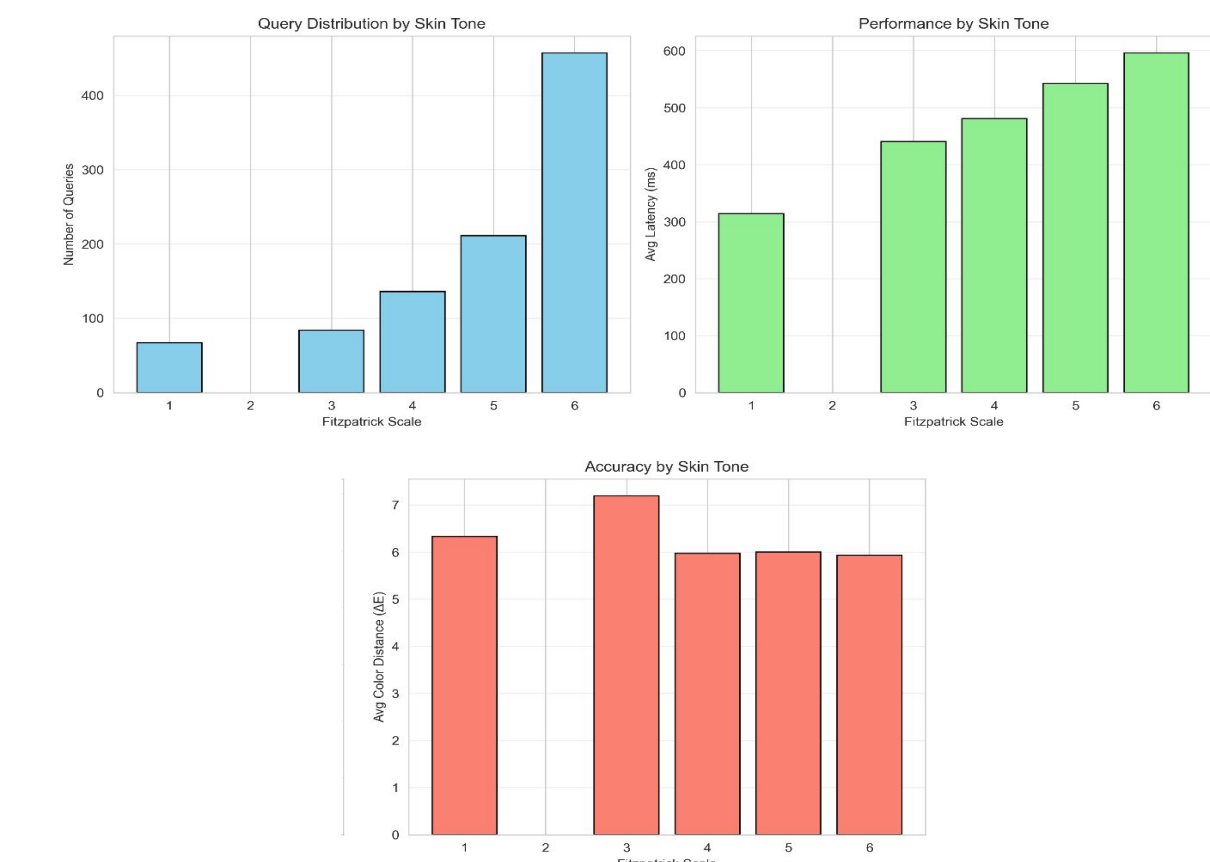
Fairness & Ethics

- Fitzpatrick scale tracking (1-6)
- Demographic fairness metrics across skin tones
- Privacy-first architecture (no data retention)



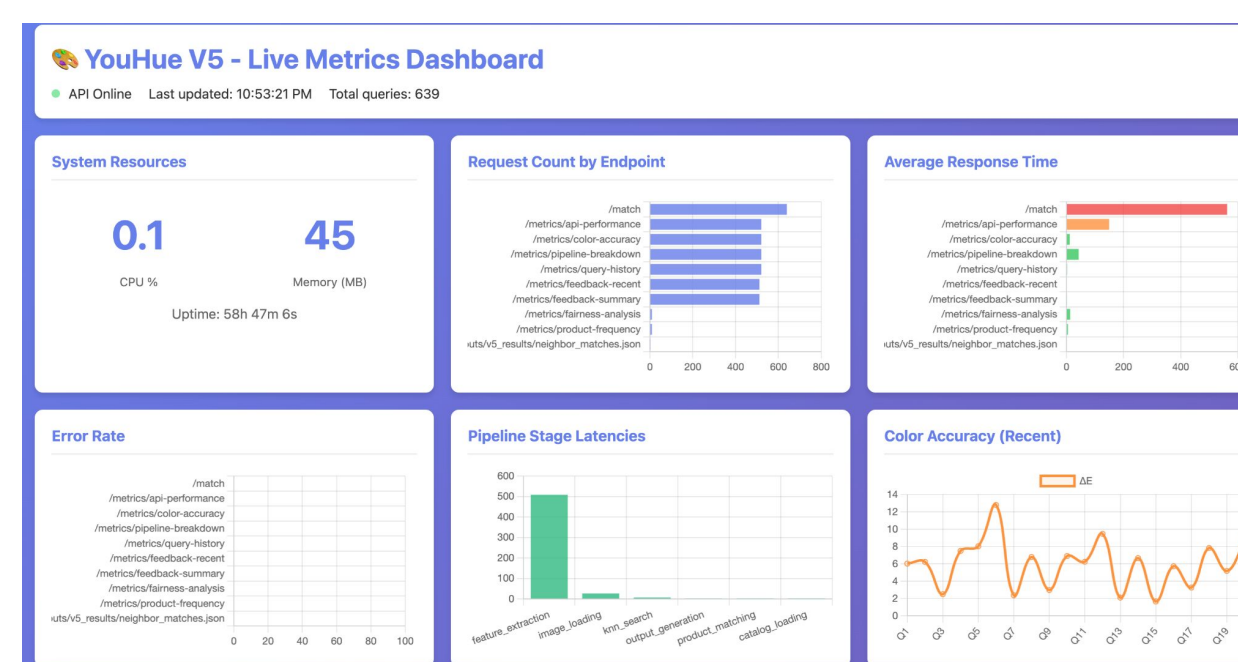
Key Takeaways

- Two-stage architecture outperforms single-stage matching
- FAISS enables real-time search at scale (585→infinity scalable)
- Color science (CIEDE2000) > RGB distance
- Privacy-by-design enables ethical AI deployment
- Production-ready system with comprehensive monitoring



Performance Metrics:

- Total pipeline latency: ~300ms (CPU)
- Face detection + feature extraction: ~150ms
- KNN search: <2ms (585 faces)
- Product matching: ~50ms per celebrity
- API throughput: ~200 req/min sustained (1000 req/min burst limit)



References



LinkedIn



Portfolio



Github



Download



Lightweight Speech-to-Image Framework for Rapid Facial Reconstruction in Forensic Settings

Joshua Lamb, M.S. Artificial Intelligence Systems

BACKGROUND

- Traditional crime suspect sketches rely on forensic artists, who are rare and often unavailable early in investigations.
- Software like FACES and EFIT-V can generate sketches, but require manual input, trained operators, and aren't easily accessible.
- Witnesses usually provide only verbal descriptions and quickly forget face details, making rapid reconstruction essential for suspect identification.
- Recent advancements in lightweight speech-to-text and text-to-image models (i.e., Wav2Vec2, Stable Diffusion v1.5) enable:
 - Fast, automated generation of suspect faces from speech.
 - Lightweight, deployable systems for real-time, offline crime scene use.
 - Downstream applications like matching via CLIP and search online databases and public records to assist in identification
- The Problem:** Can we build a lightweight, offline-capable pipeline that rapidly converts witness speech into realistic suspect imagery, without trained artists or GUIs?
- The Solution:** Tune a lightweight, multimodal speech-to-text-to-image pipeline using pretrained Wav2Vec2 and Stable Diffusion v1.5, finetuned on Librispeech and FaceCaption-15M respectively.

RESULTS

Architecture	Train Performance	Test Performance	CPU Efficiency (4-8 cores)	GPU Efficiency (16GB VRAM)
Wav2Vec2	10% WER	5% WER	~2s/10s input	<1s/10s input
Diffusion	0.14 MSE	0.13 MSE	~3m/img	~3s/img



Fig. 3. Ground Truth Images and Generated Images

- Wav2Vec2 tends to use complex words from the training data, i.e., "knight" over "night"
- Diffusion model captures facial nuance, but is heavily limited by poor captioning in the FaceCaption-15M dataset

KEY TAKEAWAYS

- Fast, Lightweight System:** Diffusion is the computational bottleneck of the pipeline yet tuned Stable Diffusion v1.5 still runs reasonably with just the CPU.
- Impact Potential:** Rapid, on-site suspect identification, minimizing witness memory decay and enabling downstream database searches.
- Modularity:** Components (i.e., speech and diffusion architectures) can be swapped to suit different use cases or constraints.
- Offline Deployment:** The pipeline runs without an internet connection, making it ideal for field use even in remote locations.
- Resource-Intensive Tuning:** Stable Diffusion v1.5 is fast in inference but requires ~20GB VRAM minimum for tuning, even with memory-saving optimizations.
- Quality Data:** Wav2Vec2 currently uses complex words from the training data and Stable Diffusion v1.5 is limited by the poor captions from the training data, highlighting the need for high-quality data.

PIPELINE WORKFLOW

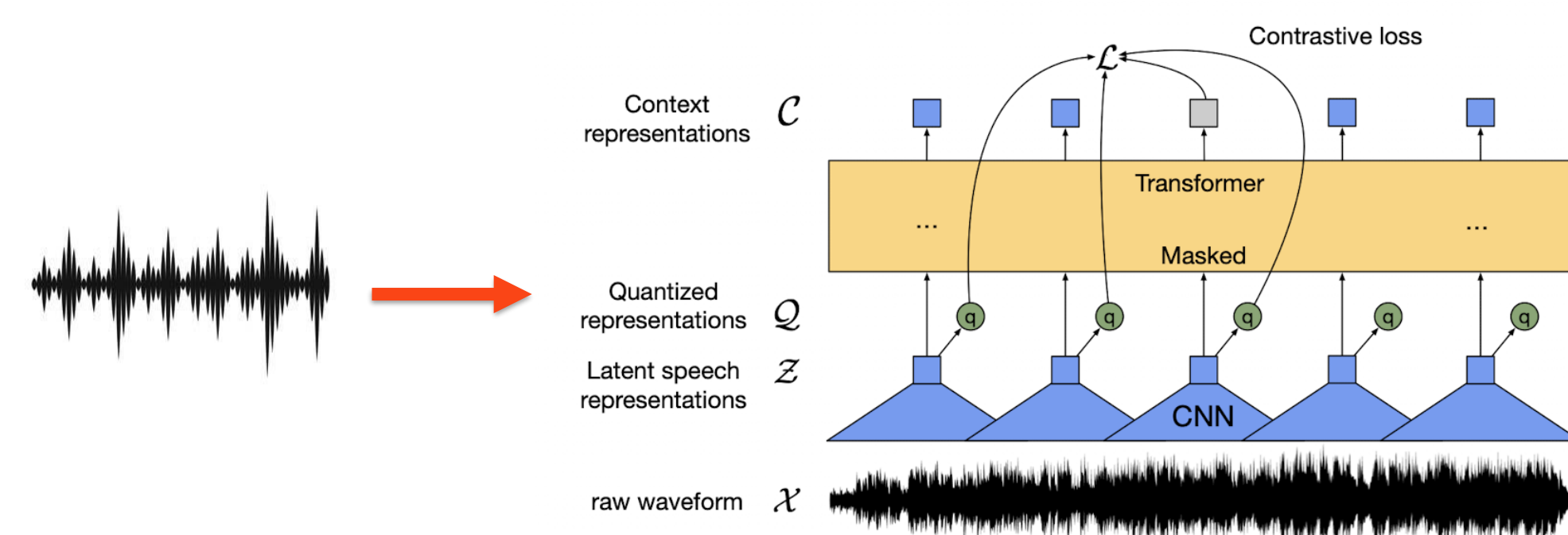


Fig. 1. Wav2Vec2 Architecture

1. Wav2Vec2 takes a raw audio input, extracts latent features using a Convolutional Neural Network (CNN) and Transformer, and uses a Connectionist Temporal Classification (CTC) decoder to output text.

2. Stable Diffusion v1.5 starts with text input, tokenizes and embeds it using a CLIP encoder, and iteratively denoises Gaussian noise into a latent space under the guidance of the embedded text.

"A man with dark hair, glasses, and a long face."

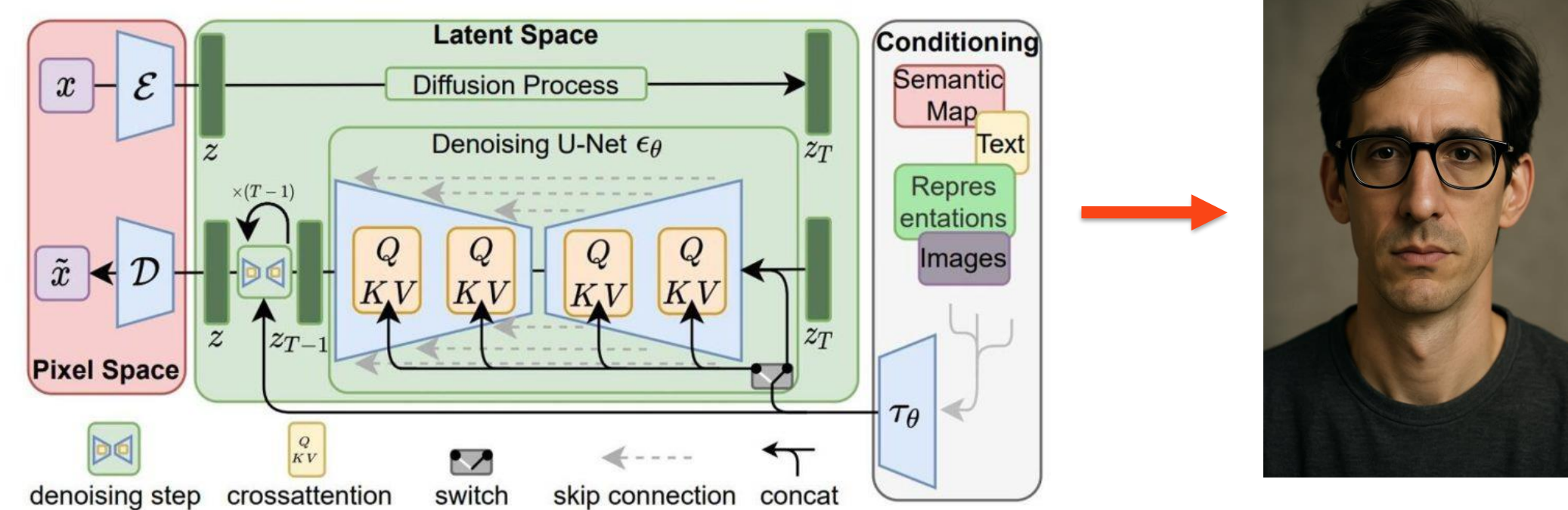


Fig. 2. Diffusion Architecture

3. Lastly, Stable Diffusion v1.5 uses a Variational Autoencoder (VAE) to transform the refined latent into an outputted image.

WHAT'S NEXT

- Use noisier, more realistic speech to tune Wav2Vec2 for field conditions.
- Use richer image captions to better tune Stable Diffusion v1.5.
- Scale up training data for both models to boost robustness and diversity.
- Tune hyperparameters with validation sets for both models.
 - Try "guidance_scale" and "num_inference_steps" to control caption strictness and image quality of the diffusion model.
- Limit the diffusion model's noise initialization to yield more deterministic outputs.
- Add LoRA modules when tuning the diffusion model for better efficiency and lower memory use.



My LinkedIn



References

Emotion-Aware Multilingual Audio-to-Image Generation Using CLIP-Guided Diffusion

¹Anany Sharma
Master of Science in AI Systems
anany.sharma@ufl.edu



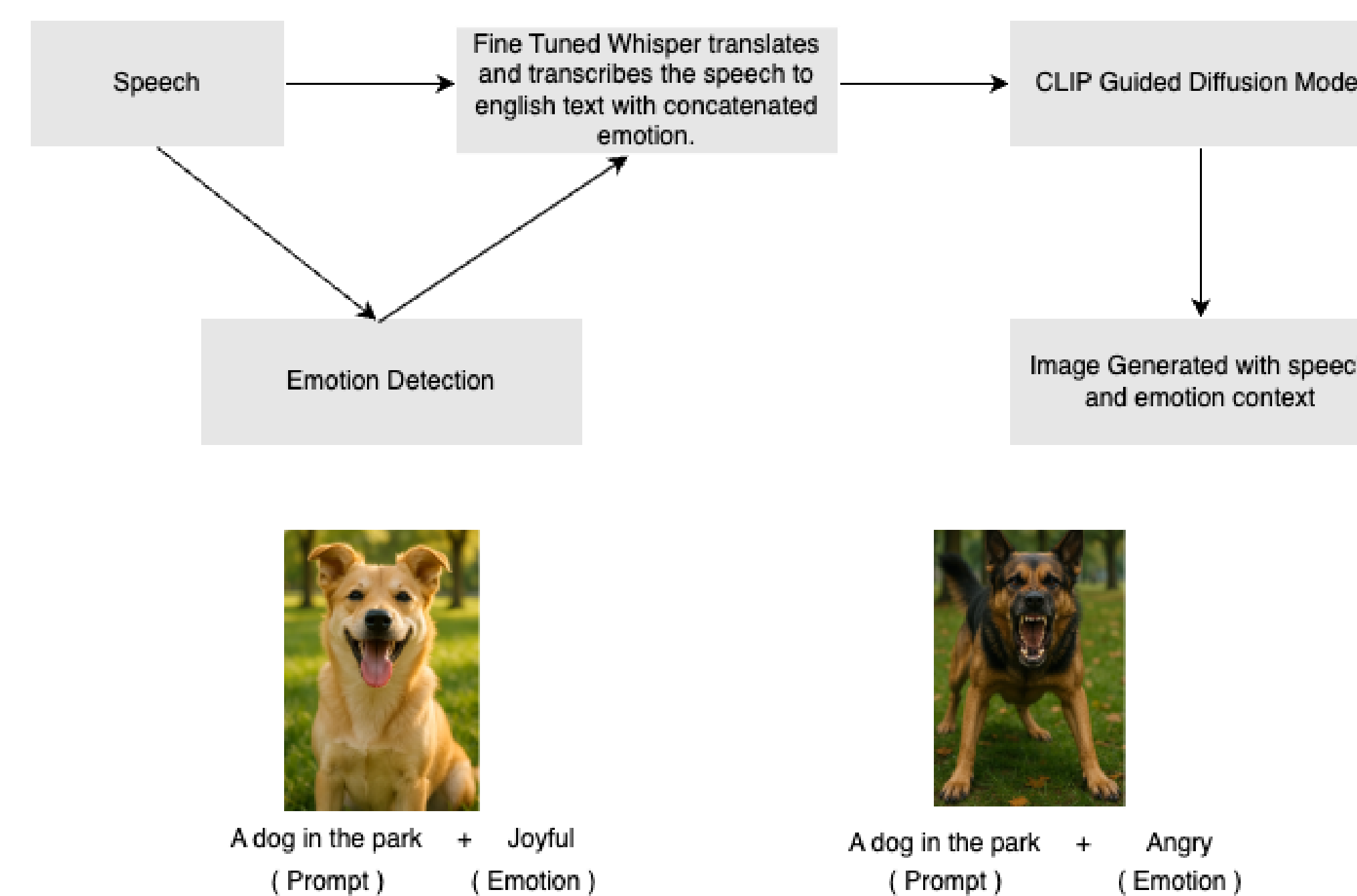
Background

- Text-to-image synthesis has emerged as a powerful research direction at the intersection of computer vision and natural language processing.
- However, most existing models are trained predominantly on English datasets and lack emotional context, limiting their applicability for low-resource languages and expressive tasks.
- In this work, we propose a novel pipeline that enables emotion-aware image generation from Hindi speech prompts.
- We fine-tune OpenAI's Whisper model to transcribe and translate Hindi audio into English, while simultaneously applying a speech emotion recognition module to extract the emotional tone of the input.

Problem to Solve?

- How can we enable speech-to-image generation for multilinguals (Hindi), using existing state-of-the-art models?
- Can we make these generated images emotionally aware incorporating speech emotion recognition into the generation pipeline?

Model Pipeline



- User's Hindi speech initiates the pipeline and is analyzed for sentiment using an emotion detection module.
- The sentiment is attached to the translated and transcribed text output from a fine-tuned Whisper model.
- A CLIP-guided diffusion model then generates emotion-aware, contextually relevant images based on the enriched text.

Implementation Results

- We fine-tuned Whisper on Hindi from Common Voice, trained an emotion classifier using RAVDESS, and generated images with a CLIP-guided Stable Diffusion model fine-tuned on Flickr8K using translated text and inferred emotions.
- Fine-Tuning Hyperparameters (Whisper):

Parameters	Values
Learning_rate	1e-5 - fine tuning learning rate
max_steps	4000 - total_training_steps
gradient_accumulation_step	4 - for every 4 steps gradients are updated
per_device_train_batch_size	4 - training_batch_size_per_device
Evaluation_strategy	Word Error Rate

- Results

Component	Metric	Results
Whisper Fine Tuned	WER	32.0%
Emotion Classifier	F1-Score	[Pending]
Diffusion Model	CLIP Score, MSE	0.023 MSE before CUDA OOM

Architectural Diagram / Workflow

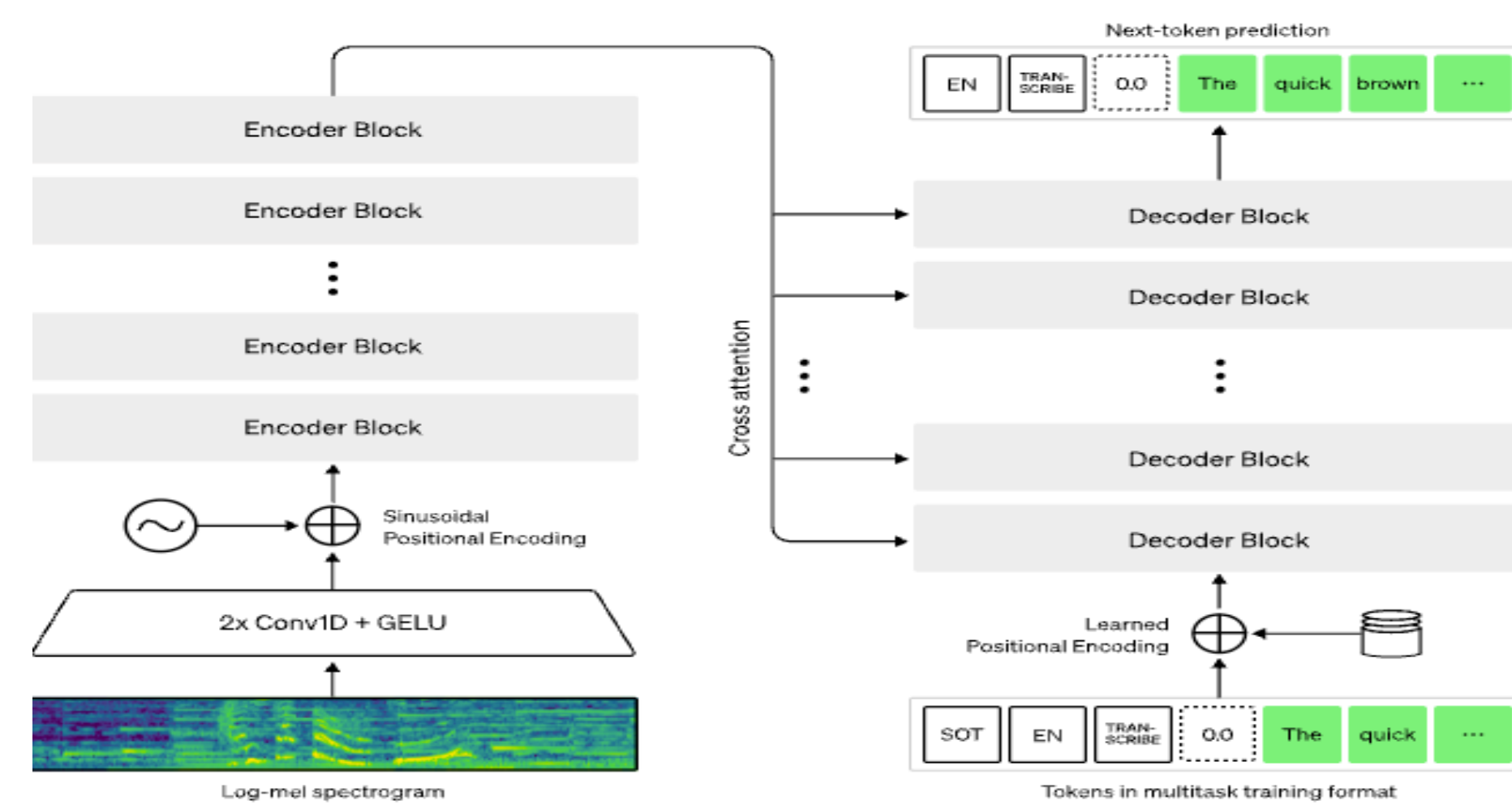


Fig 1. Whisper Model Architecture

As per Fig1, the raw audio inputs are converted to a log-Mel spectrogram by action of the feature extractor. The Transformer encoder then encodes the spectrogram to form a sequence of encoder hidden states. Finally, the decoder autoregressively predicts text tokens, conditional on both the

Fig 2 depicts the implementation of how CLIP converts text and images into a shared latent space using trained encoders for both modalities.

1. Contrastive pre-training

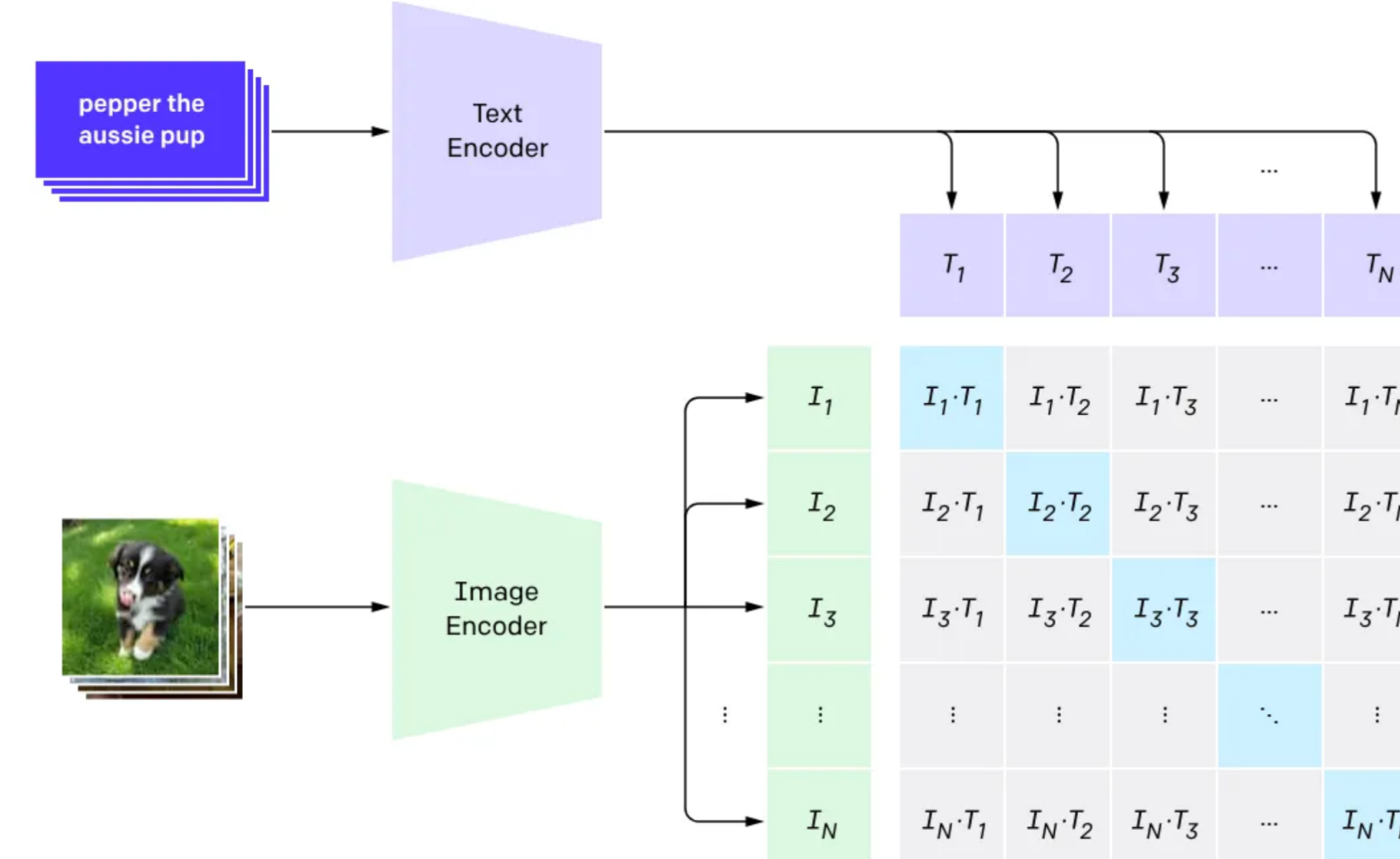


Fig 2. CLIP shared embedding space

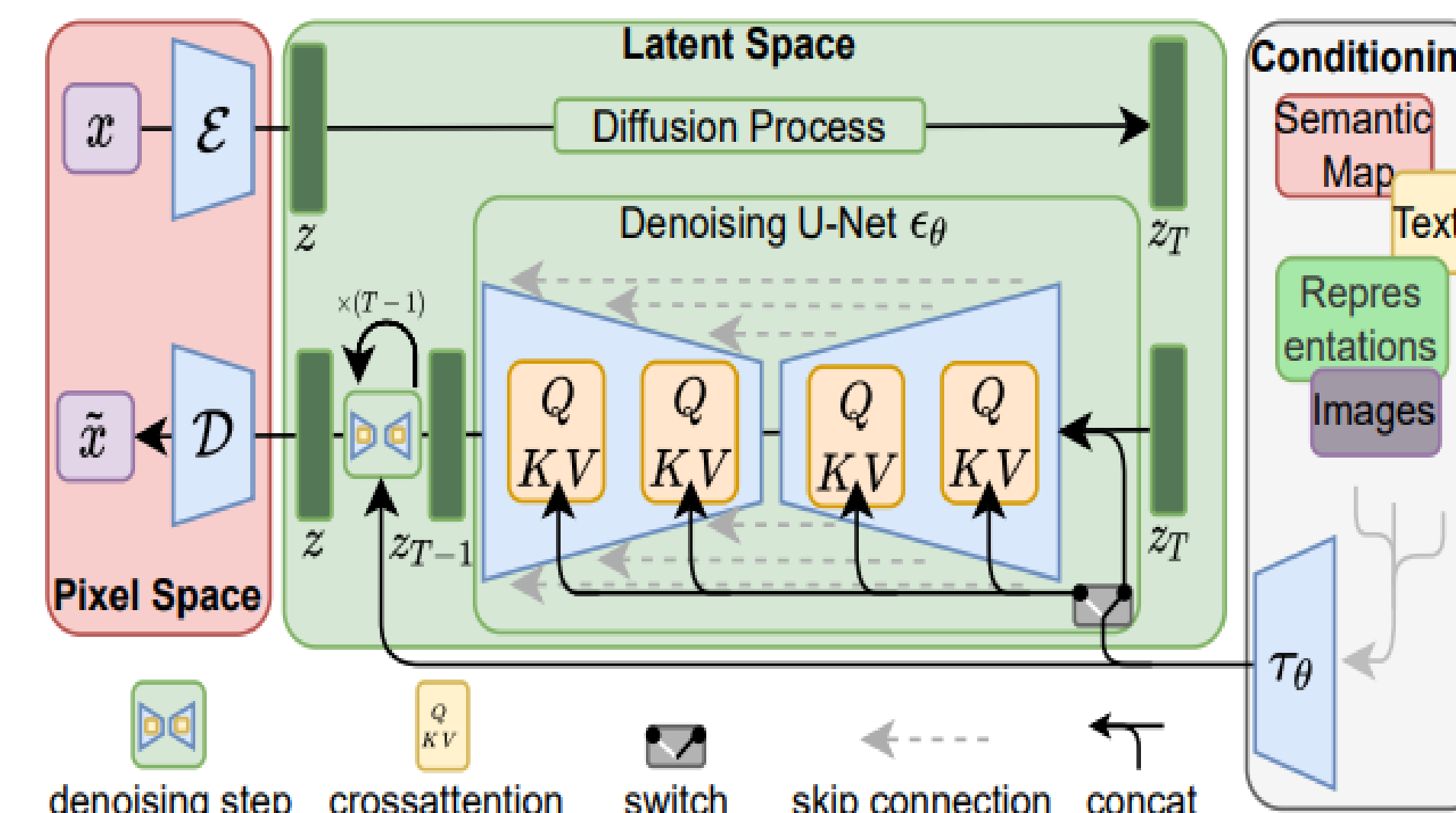


Fig 3. Diffusion Model Architecture

For Fig 3, the CLIP embeddings are used as cross attention while training the diffusion model to generate semantically similar images with emotional context.

Key Takeaways

- Multimodal Integration:** Building a pipeline that connects speech, text, emotion, and image modalities.
- Fine-Tuning Large Models:** A major learning milestone was fine-tuning state-of-the-art models like Whisper and Stable Diffusion.
- Compute Limitations:** Without access to high-end GPUs and cloud compute resources, implementing this project would have been impractical or extremely time-consuming.
- Transfer Learning:** Leveraging pretrained models significantly reduced training costs and allowed experimentation within constrained environments.
- Resource Management & Optimization:** I gained valuable insight into how GPU memory, batch sizes, and learning rates impact training.

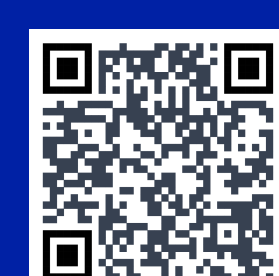
References



LinkedIn



Portfolio



Github



Kaggle

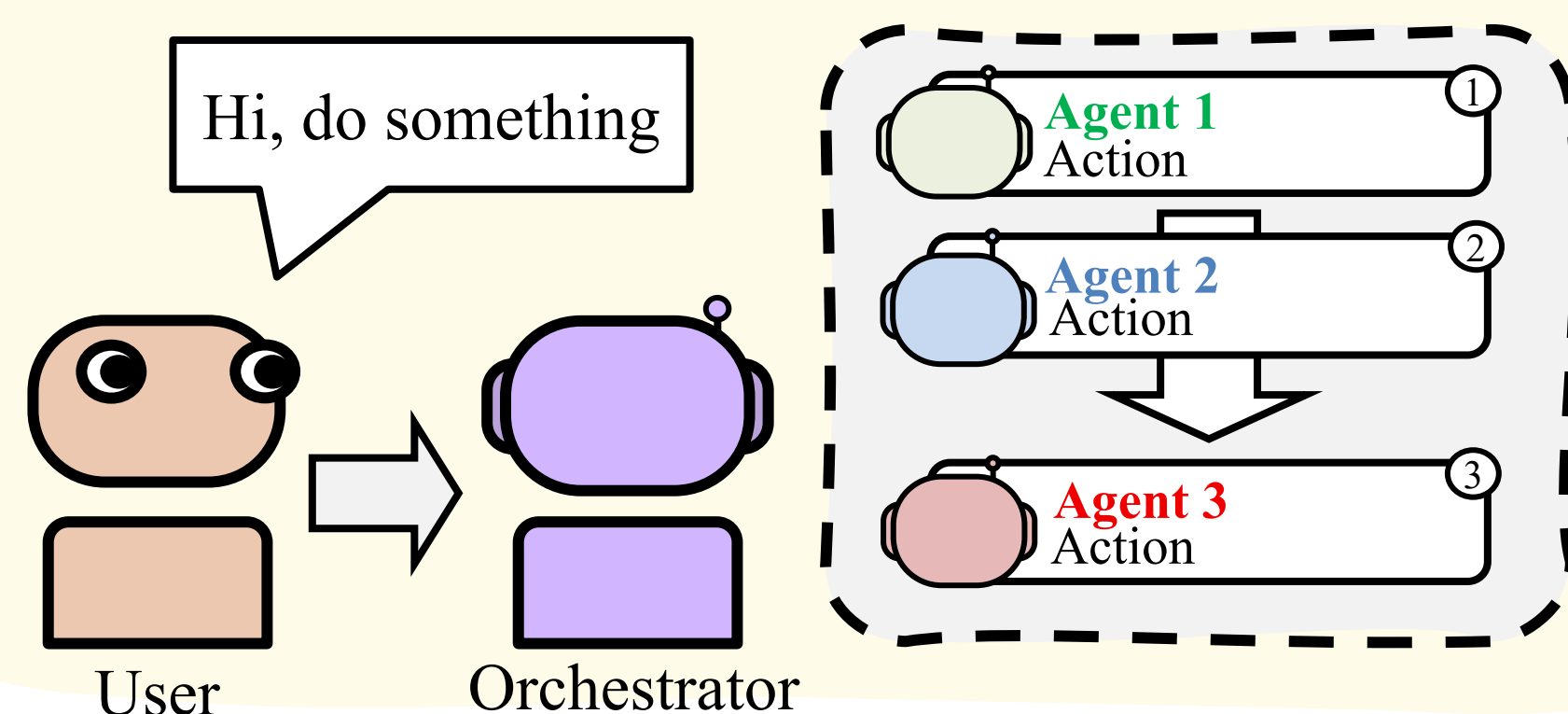


Introduction

Current AI agent systems excel in short-term tasks but continue to fail on longer ones. This work proposes “SentinelSteps” to enable agents to successfully complete longer tasks and introduces “SentinelBench” as a novel benchmark of simulated website environments.

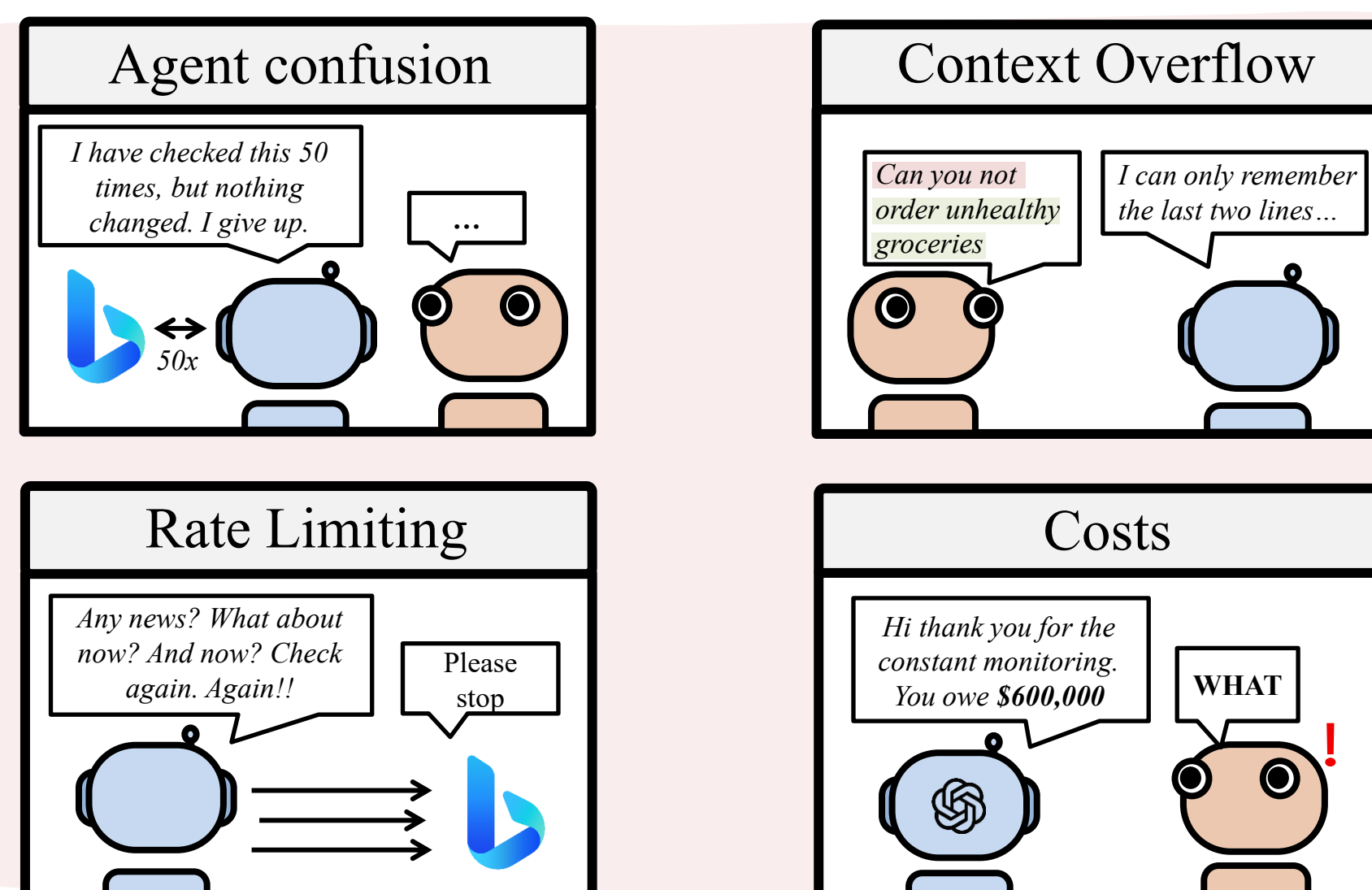
Current Approaches

(1) User makes a request (2) Orchestrator splits it into steps (3) Steps are assigned to specialized agents (4) Steps are executed sequentially.



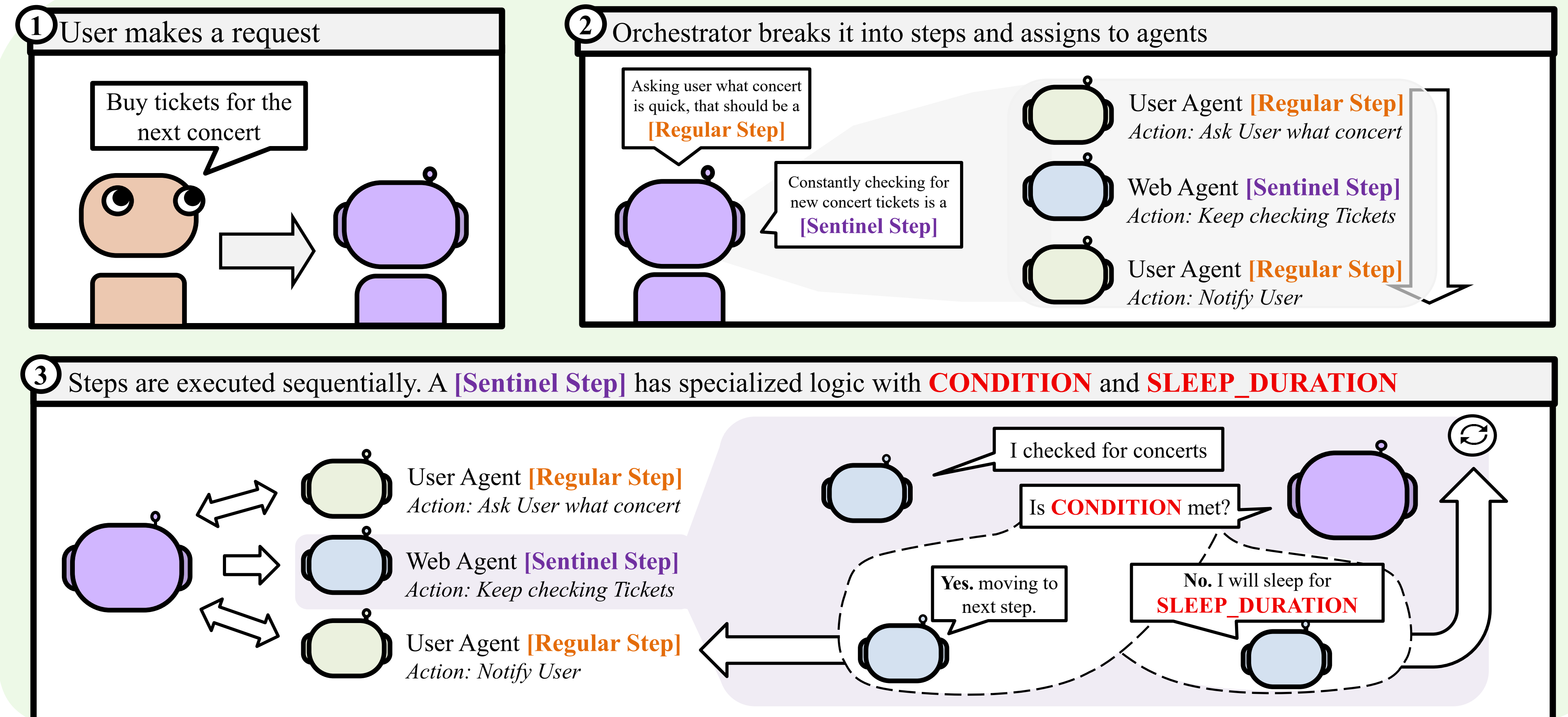
Limitations of Current Approaches

While excelling in many short tasks, **current systems may run into problems** when performing longer tasks:



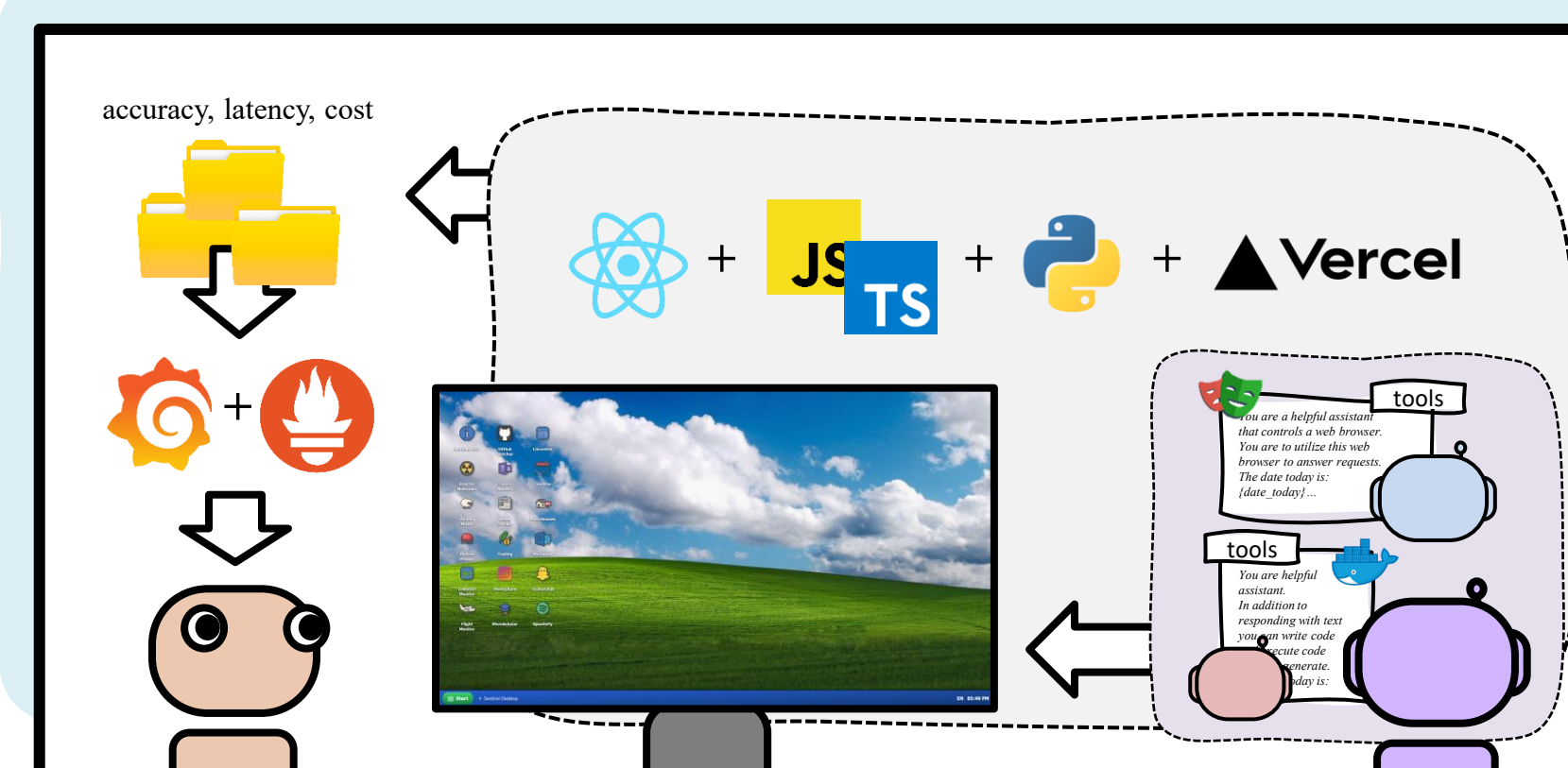
Addressing the Limitations

To address these limitations, we add a new type of step for long running, periodic or conditional tasks: **SentinelSteps**



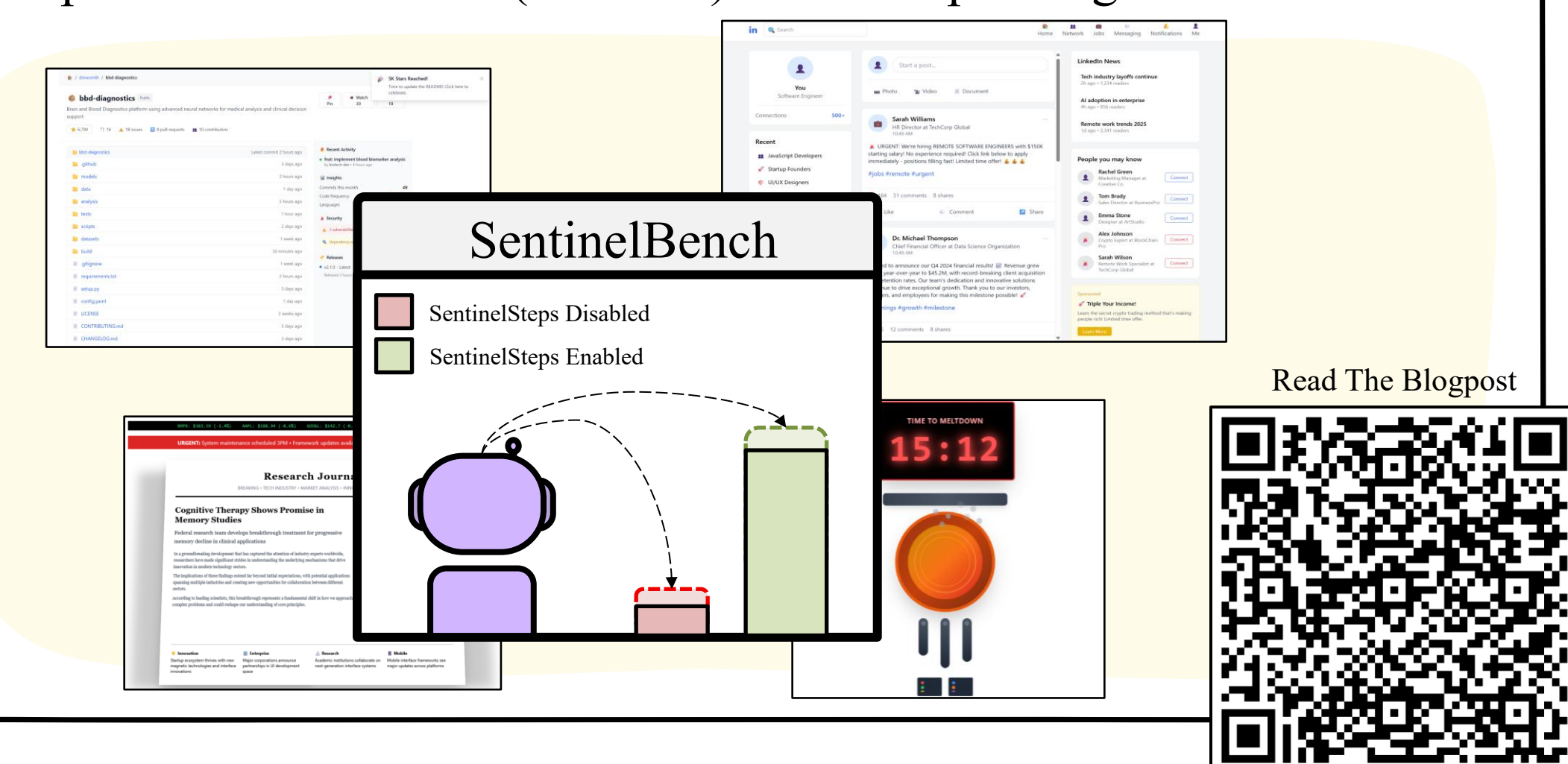
Benchmark

To then test our approaches, we create SentinelBench, a set of synthetic environments that mimic real websites



Preliminary Evaluation

Preliminary evaluation showcases that an agent using SentinelSteps performs 33% better (38%/5%) on tasks spanning over 2hours



Bilingual Handwriting Translation Between English and Telugu Using CycleGAN



¹Venkata Jyothi Priya Mulaka
Master of Science in Applied Data Science
mulakav@ufl.edu

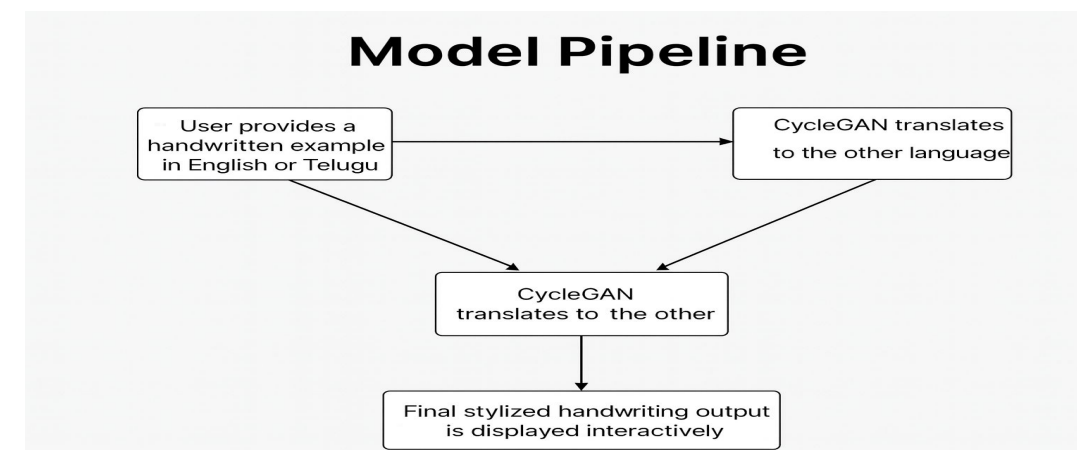
Background

- Handwriting reflects a person's style, culture, and identity, making it an important part of visual communication.
- Existing translation systems focus on text meaning but ignore handwriting appearance – the strokes, slant, and texture that make writing unique.
- This limits creativity and multilingual applications, especially when translating across different scripts such as English and Telugu.
- In this work, we propose a CycleGAN-based handwriting translation model that learns to convert handwriting styles between English and Telugu.
- The model does not need paired samples; it learns style transformation through unpaired image-to-image translation.
- It preserves word structure while adapting the visual handwriting style of the target script, enabling realistic bilingual handwriting synthesis.

Problem to Solve?

- How can we enable handwriting style translation between English and Telugu, using existing state-of-the-art generative models?
- Can we make the translated handwriting look natural and human-like, preserving both word structure and visual style across scripts?

Model Pipeline



- The user uploads a handwritten image in English or Telugu to start the translation process.
- The image undergoes preprocessing with brightness normalization and handwriting style augmentations (blur, dilation, erosion, rotation).
- A CycleGAN model with two generators and two discriminators learns to translate handwriting styles between English and Telugu using unpaired data.
- The translated handwriting is evaluated using SSIM and PSNR and visualized interactively through a Gradio interface..

Architectural Diagram / Workflow

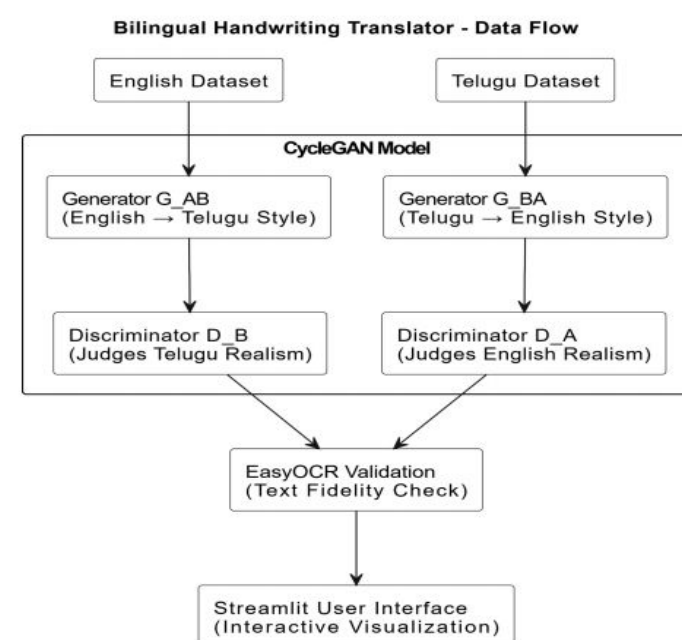


Fig 1. Data Flow of the Bilingual Handwriting Translator using CycleGAN

As shown in Fig. 1, the English and Telugu handwriting datasets are passed through a CycleGAN with dual generators and discriminators for cross-style translation. The model learns mappings without paired data, guided by cycle-consistency, identity, and adversarial losses. Generated outputs are evaluated using SSIM and PSNR to ensure clarity and realism. The final translated handwriting is visualized interactively through a Gradio interface.

Fig. 2 depicts the data augmentation process applied to both English and Telugu handwriting datasets to enhance style diversity during training. OpenCV-based transformations such as Gaussian blur, dilation, erosion, rotation, and inversion simulate natural handwriting variations like cursive, bold, and calligraphic styles. These augmentations enable the CycleGAN to generalize across diverse stroke patterns and writing intensities, resulting in more realistic and human-like translations.



Fig 2. Data Augmentation Samples

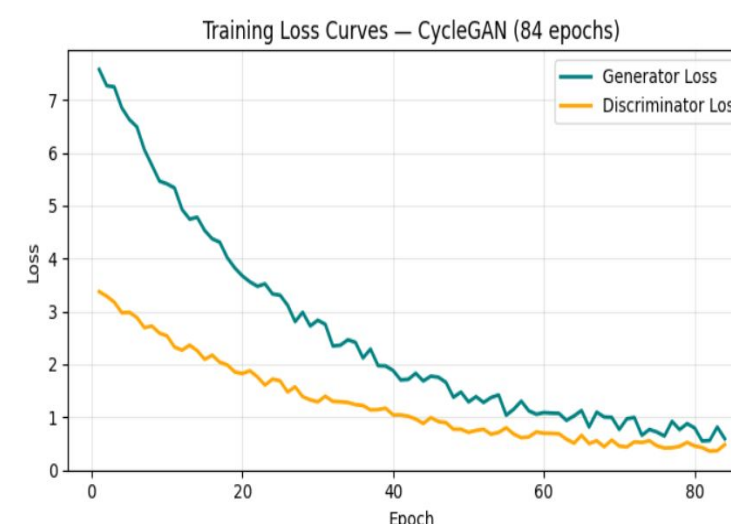


Fig 3. Training Loss Curves - CycleGAN(84 Epochs)

Fig. 3 shows the training behavior of the CycleGAN, where two generators and two discriminators are trained together. The total generator loss combines both directions (English → Telugu and Telugu → English) using adversarial, cycle-consistency, and identity losses. These losses help the model translate handwriting styles without needing paired samples while keeping strokes realistic and consistent. The smooth convergence of both losses shows stable training and balanced learning between the generators and discriminators.

Implementation Results

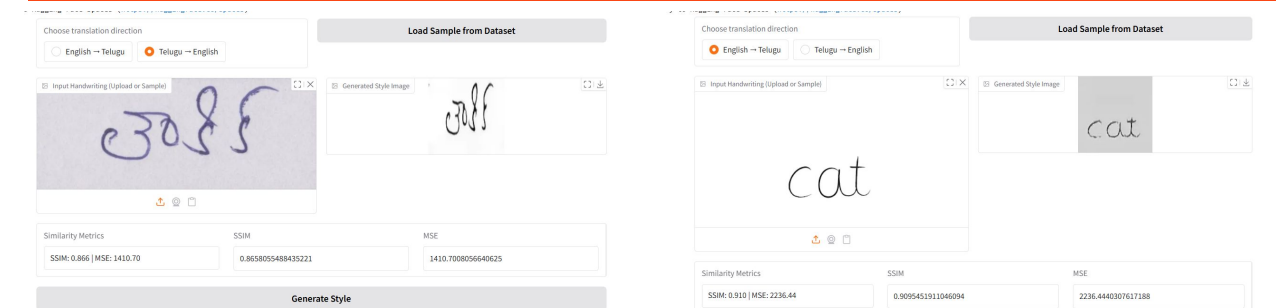


Fig. 4. Gradio interface of bilingual handwriting translation. Left: English → Telugu. Right: Telugu → English. SSIM (0.86 – 0.91) confirms strong structural similarity with variation only in handwriting style.

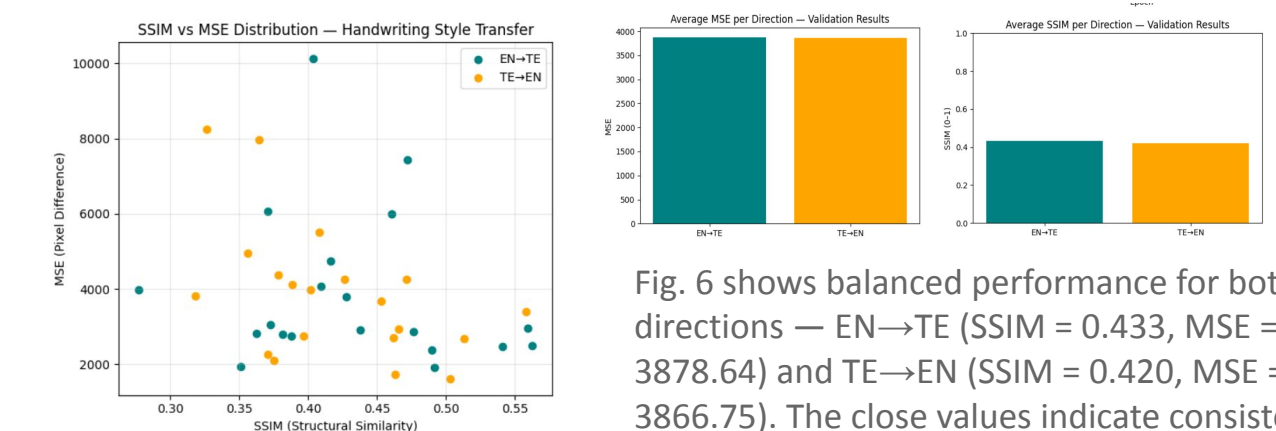


Fig 5. SSIM-MSE Distribution Plot

Fig. 6 shows balanced performance for both directions — EN→TE (SSIM = 0.433, MSE = 3878.64) and TE→EN (SSIM = 0.420, MSE = 3866.75). The close values indicate consistent handwriting translation with minimal pixel distortion and good structure preservation.

Key Takeaways

- CycleGAN successfully performs unpaired handwriting translation between English and Telugu.
- Achieved realistic style adaptation while maintaining word structure and clarity.
- SSIM (0.5–0.8) confirms style change without content loss.
- OpenCV augmentations improved robustness across diverse handwriting forms.
- A Gradio interface enables real-time visualization and interactive testing.

References

- [1] J.-Y. Zhu *et al.*, "Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks," *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. Available: <https://arxiv.org/abs/1703.10593>
- [2] *IAM Handwriting Word Database*, Kaggle, 2023. Available: <https://www.kaggle.com/datasets/nibinv23/iam-handwriting-word-database>
- [3] *Indic Handwritten Data (Telugu Script)*, IIIT Hyderabad CVIT Research Group, 2023. Available: <https://cvit.iiit.ac.in/research/projects/cvit-projects/indic-hw-data>
- [4] *Gradio Documentation*, Hugging Face, 2024. Available: <https://www.gradio.app/docs>
- [5] V. J. P. Mulaka, "Bilingual Handwriting Translator - CycleGAN Project Repository," University of Florida, 2025. Available: <https://github.com/MVJyothiPriya/Bilingual-Handwriting-translator>

LinkedIn



Portfolio



Github

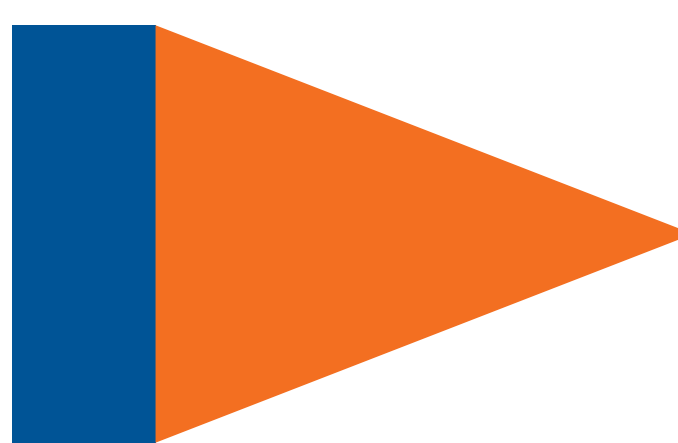


Kaggle



**UF**

Herbert Wertheim
College of Engineering
Department of Engineering Education
UNIVERSITY of FLORIDA



Cloudy with a Chance of Labels

Understanding Satellite Images with Advanced Deep Learning

Norman Bukingolts

nbukingolts@ufl.edu

Introduction

This research explores two complementary approaches to satellite image understanding in scenarios like disaster response, climate monitoring, and land use classification:

- **Multi-label classification** using *CNN* and *CycleGAN* architectures,
- **Image captioning** and **semantic search** using *CLIP*-based models that bridge visual and textual representations via multimodal learning.

Research Question

How effectively can deep learning algorithms interpret satellite imagery, and how can their validity for automated remote sensing analysis be demonstrated?

Datasets



The "Planet: Understanding the Amazon from Space" dataset [1, *left*] contains 40,478 Amazon rainforest satellite images with 17 labels. The RSICD dataset [2, *right*] includes 10,921 satellite images, each with five descriptive sentences.

Key Takeaways

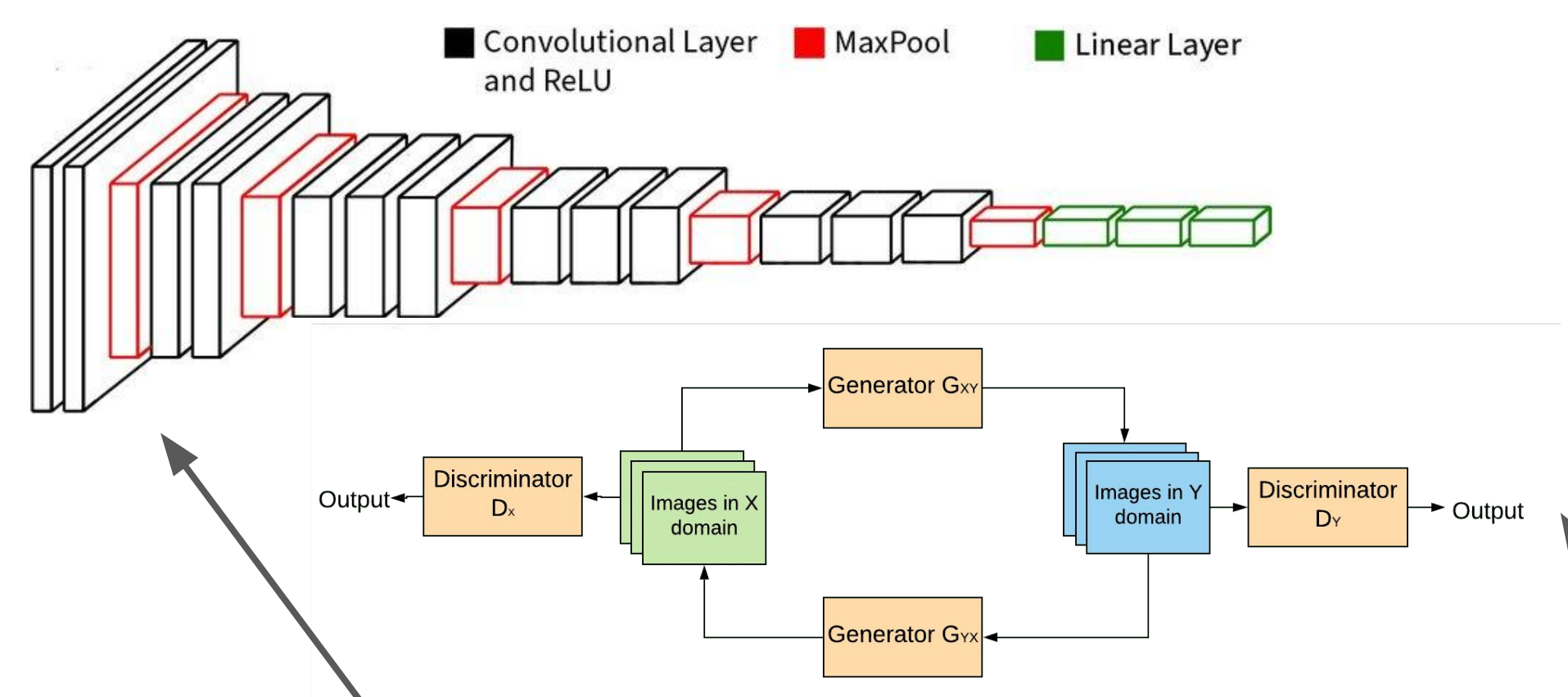
1. Deep learning models can be beneficial in efforts to automate satellite imagery analysis.
2. Generative and multimodal approaches can help address data imbalance and search vast datasets.
3. Future work could explore training with *multi-spectral* satellite imagery (4+ bands instead of only RGB) to enhance model performance.

References & Demo

Scan the QR code to see the full list of the references used in this poster and links to associated reports and a demo.



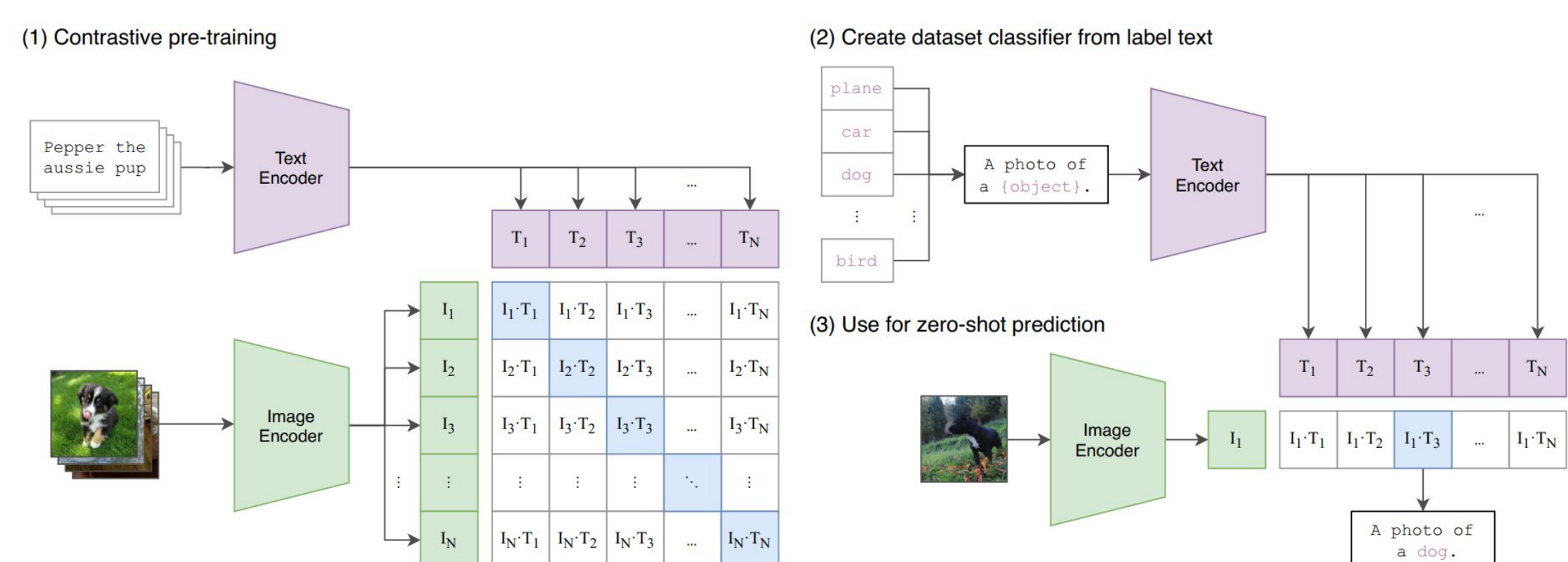
Approach 1: CNN & CycleGAN



A VGG-style *CNN* architecture [3, 4] was implemented for multi-label classification of satellite imagery, featuring repeating blocks of convolutional layers with small 3×3 filters followed by max pooling.

To address class imbalance, a *CycleGAN* was developed to augment the dataset by generating synthetic images through unpaired image-to-image translation [5], particularly for the "cloudy" label.

Approach 2: CLIP



A contrastive image-language pre-training (*CLIP*) architecture [6] was implemented for image captioning, initially fine-tuning OpenAI's pre-trained encoders. This foundation was enhanced by replacing the original encoders with domain-specific alternatives [7, 8, 9] along with a learnable temperature parameter to optimize contrast between image-text pairs, improving captioning and enabling semantic search through a vector database.

Results & Evaluation

The *CNN* achieved strong performance with a **validation accuracy of 0.95** and an **F2 score of 0.81** after 15 epochs. Saliency maps revealed that the model correctly focused on relevant image features (e.g., edges of clouds).

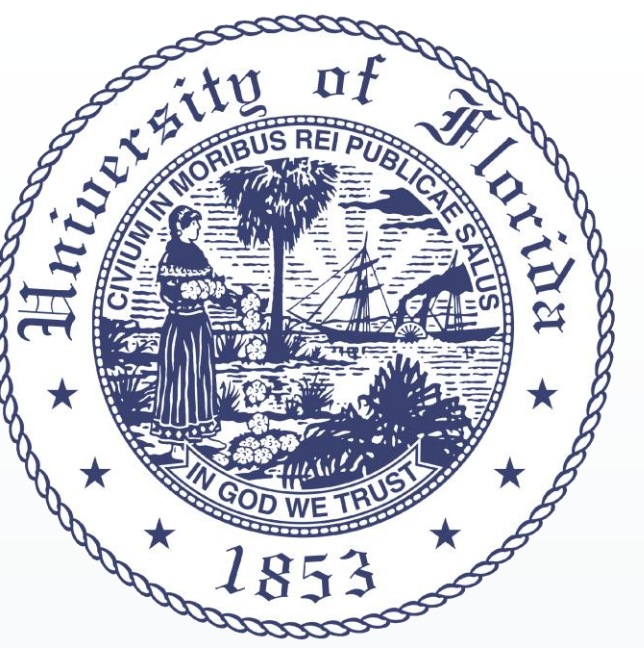
The *CLIP* model initially achieved modest validation metrics of BLEU: 0.40 and CIDEr: 2.70, while the enhanced architecture more than doubled performance to **CIDEr: 5.79 and BLEU: 0.60**, with strong test results confirming better generalization.

Uncertainty-Guided Generative Adversarial Networks

Luke Saleh

University of Florida

Department of Electrical and Computer Engineering



Abstract

This project introduces an **uncertainty-guided WGAN-GP** to synthesize **realistic chest X-rays** from the NIH dataset and visualize results in an **interactive Tkinter app**. The generator loss is augmented with a **critic-based uncertainty term**, and models are trained in both a **two-stage fine-tuning regime** and **single-stage baselines**. We sweep uncertainty weights λ and evaluate on a held-out test set using **FID** and **Inception Score**, while the visualizer lets users explore samples, play a “Real vs. Fake” game, and inspect critic scores.

Problem Statement/Research Question

GANs for medical image synthesis often struggle to balance realism and diversity, and their training dynamics can be brittle [1]. This work asks:

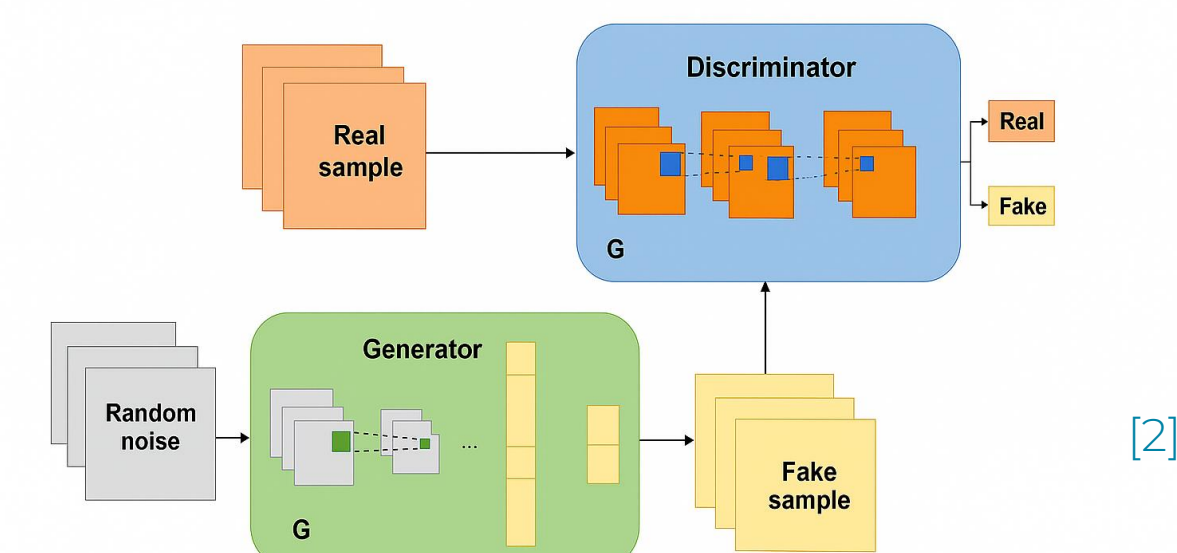
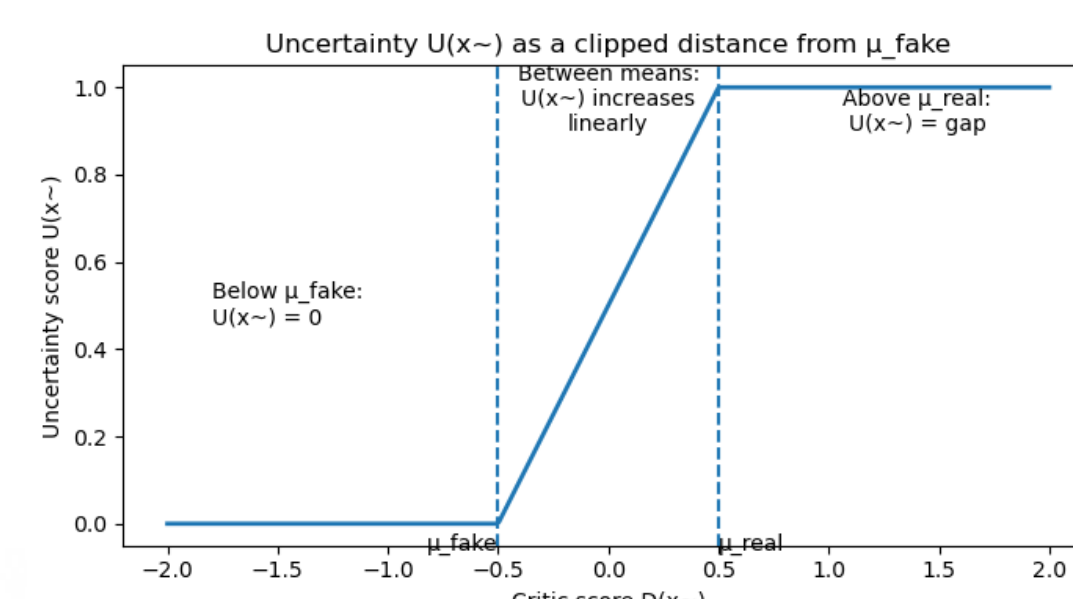
- Can critic-derived uncertainty be used to guide WGAN-GP training to improve sample quality and diversity for chest X-ray generation?
- How does the strength of this uncertainty weighting (λ) interact with data augmentation and a two-stage fine-tuning regime compared to single-stage non-augmented baselines?

Dataset Info

- Source: NIH Chest X-Ray Dataset
- Scale & Split: **86,524** chest radiographs split into **72% train**, **13% validation**, **15% test**
- Preprocessing: Images converted to grayscale, resized to 128x128, normalized to range of [-1,1]
- Training Transforms:
 - **Stage 1** (Fine tuned augmented models): Random horizontal flips, small rotations, affine transforms, random crops/zoom, brightness/contrast jitter
 - **Stage 2 & baselines** (non-augmented): Resize->tensor-> normalization only.

Model/Approach

- Core model: WGAN-GP with uncertainty-aware generator loss
- Loss becomes $L_g = -E[D(x)] - \lambda * E[U(x)]$
 - $U(x) = \text{clip}(D(x) - \mu_{\text{fake}}, 0, \mu_{\text{real}} - \mu_{\text{fake}})$



- Finetuned models: **augmented stage first**, then retrain on nonaugmented
- Baseline models train only on **nonaugmented data**
- Both trained with DistributedDataParallel

Results

- Experimental design for both model types:
 - **Systematic sweep of λ** over: {0.0, 0.001, 0.1, 0.25, 0.5, 0.75, 1.0, 1.5, 2.0}

Table 1: Finetuned Model Results

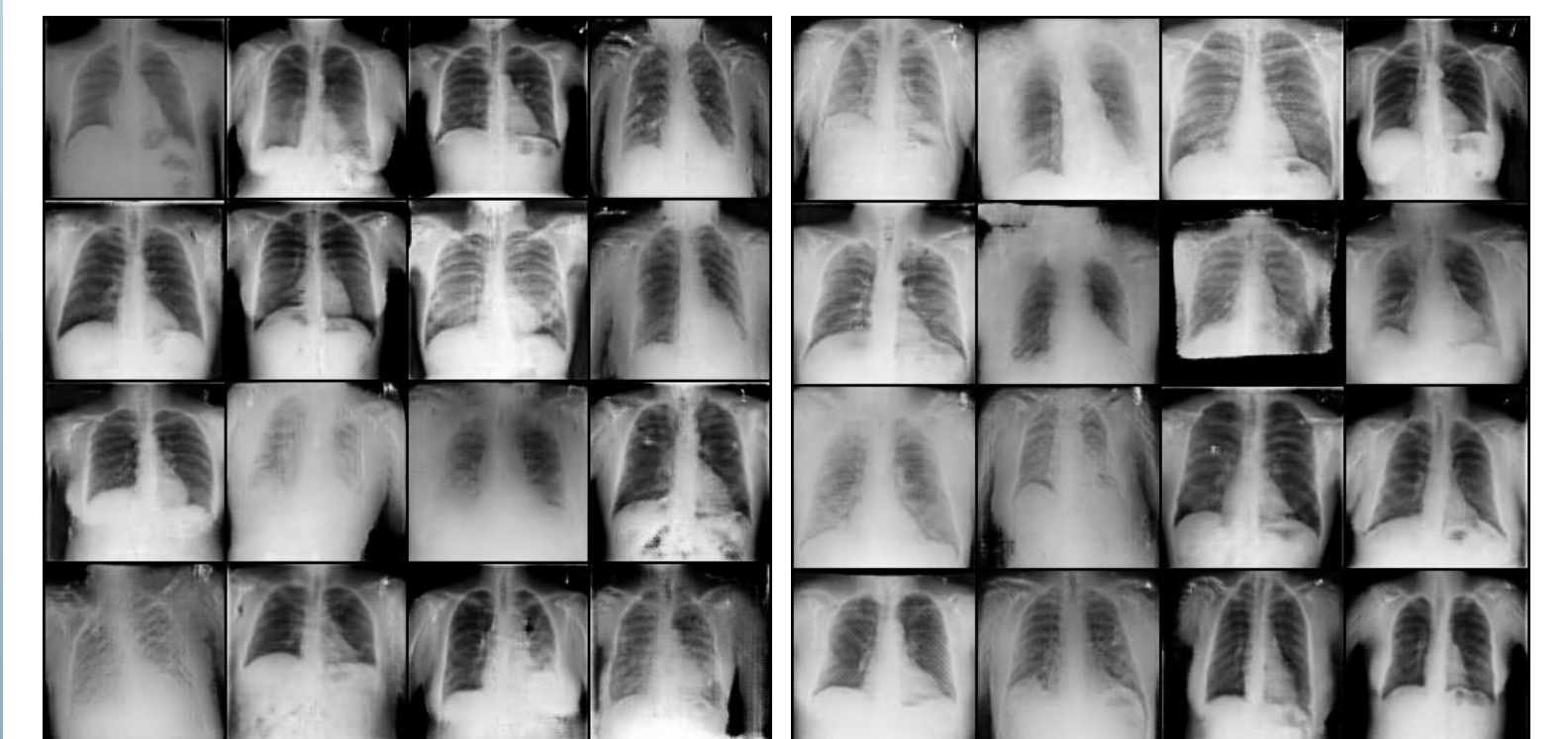
λ	Validation FID	Test FID	Test Inception Score (mean $\pm$ std)
0.0	75.55	87.68	3.24 $\pm$ 0.12
0.001	74.47	84.93	3.24 $\pm$ 0.11
0.01	70.27	82.48	3.34 $\pm$ 0.13
0.1	71.01	83.40	3.27 $\pm$ 0.14
0.25	79.29	91.28	3.26 $\pm$ 0.16
0.5	81.90	91.92	3.12 $\pm$ 0.12
0.75	78.89	89.91	3.27 $\pm$ 0.11
1.0	69.69	82.79	3.40 $\pm$ 0.12
1.5	74.56	86.35	3.31 $\pm$ 0.09
2.0	79.83	91.61	3.25 $\pm$ 0.10

Table 2: Baseline Model Results

λ	Validation FID	Test FID	Test Inception Score (mean $\pm$ std)
0.0	75.82	87.42	3.28 $\pm$ 0.10
0.001	75.29	87.90	3.37 $\pm$ 0.07
0.01	84.11	94.14	3.25 $\pm$ 0.16
0.1	81.56	93.06	3.38 $\pm$ 0.20
0.25	78.16	88.57	3.26 $\pm$ 0.17
0.5	71.68	81.49	3.23 $\pm$ 0.16
0.75	75.03	86.25	3.44 $\pm$ 0.16
1.0	72.38	84.00	3.31 $\pm$ 0.15
1.5	78.76	90.39	3.37 $\pm$ 0.16
2.0	74.35	86.55	3.30 $\pm$ 0.07

Conclusions

- **Uncertainty helps in moderation**
 - Both training regimes show improved FID and IS compared to $\lambda=0$
- **Two stage fine-tuning is competitive, but not dominant**
 - Do not consistently outperform baseline
 - Reach slightly better FID and similar IS
- Finetuning vs baseline models qualitatively have different issues
- Future work would include modifications to the uncertainty term, starting with different noise profiles, and comparing to other models like diffusion



Baseline model with $\lambda = 1.0$

Fine tuned model with $\lambda = 1.0$

References

- [1] X. Yi, E. Walia, and P. Babyn, “Generative Adversarial Network in medical imaging: A Review,” *Medical Image Analysis*, vol. 58, p. 101552, Dec. 2019. doi:10.1016/j.media.2019.101552
- [2] L. Yuan, Y. Ma, and Y. Liu, “Protein secondary structure prediction based on Wasserstein generative adversarial networks and temporal convolutional networks with convolutional block attention modules,” *Mathematical Biosciences and Engineering*, vol. 20, no. 2, pp. 2203–2218, 2022. doi:10.3934/mbe.2023102

Acknowledgements

Thank you to Dr. Ramirez Salgado for guiding me in the project

Stop Generating, Start Ideating: A Socratic AI Design Tool

Shaun Jose
Master in Applied Data Science
University of Florida

BACKGROUND

How might we use AI to guide designers rather than doing the work for them

Novice designers often get lost and confused in the ideation process. While standard AI tools provide direct answers, they fail to teach the actual process of creative problem-solving. **SocraDesign** bridges this gap by acting as an intelligent Socratic coach.

- **Challenge:** Users struggle with unstructured ideation, and existing chatbots suffer from statelessness (no memory) and high latency.
- **Solution:** Integrated a **Double Diamond framework** (discover→deliver) with a **Hybrid Evaluation Strategy** and **cloud persistence**. This helped to reduce the latency by ~50% and act as a seamless, real-time design partner.

RESULTS & UI

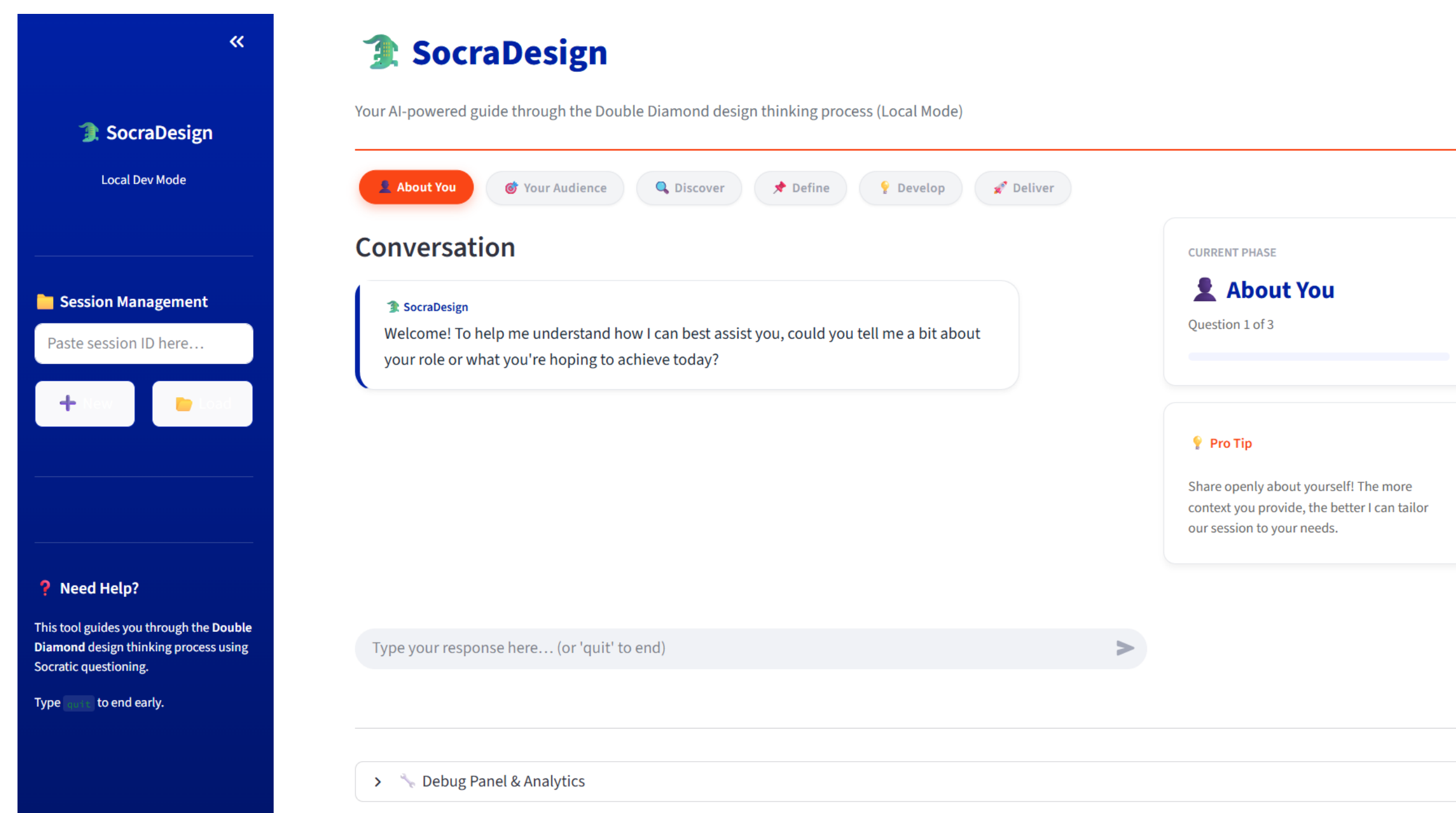


Fig 1: SocraDesign Interface. UI has gamified "Phase Pills" for progress tracking

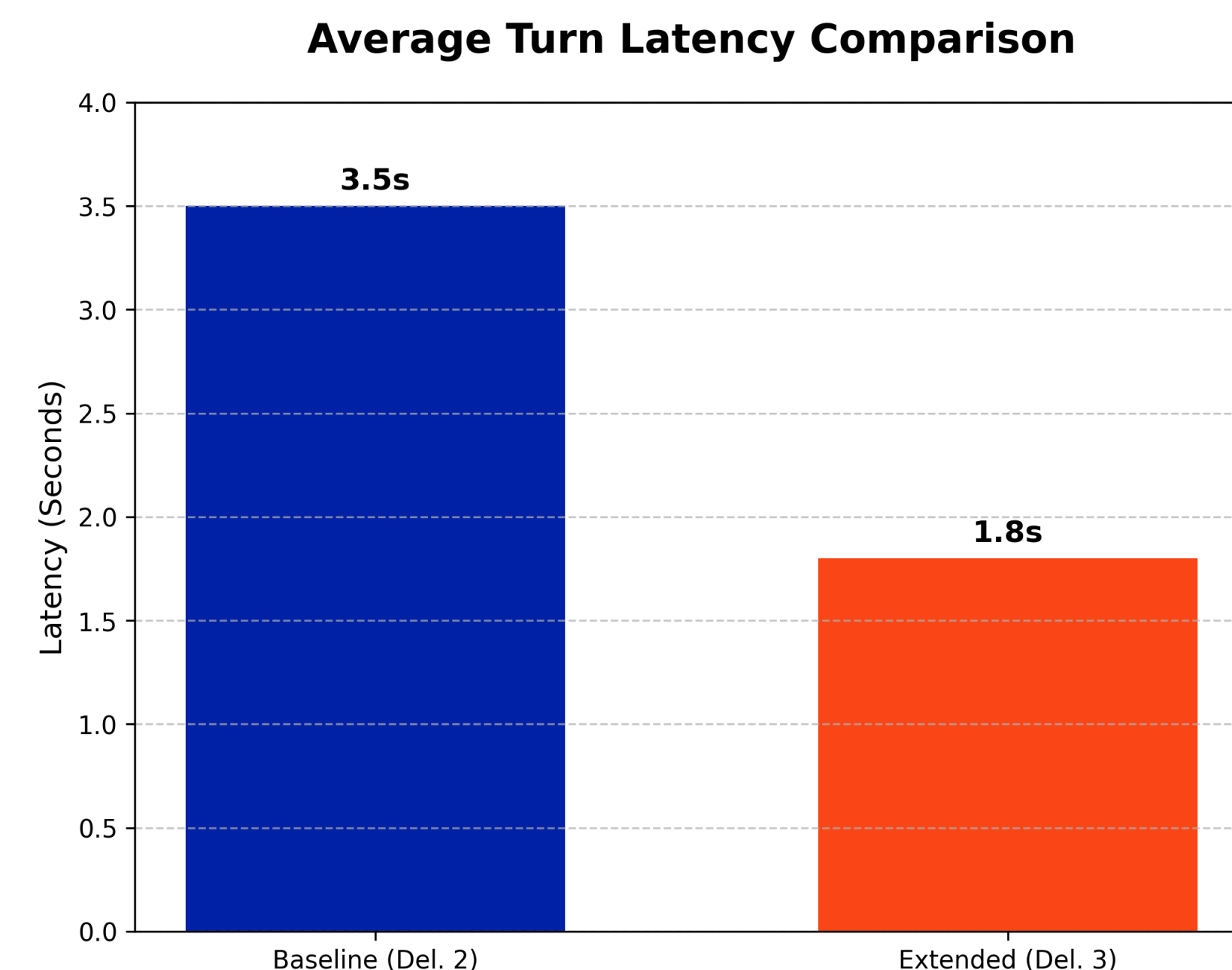


Fig 2: Latency Performance Comparison. Before and after using the hybrid AI architecture

METHODS & ARCHITECTURE

The system utilizes a Client-Server Architecture combining **Streamlit** and **LangGraph**.

- **Framework:** The agent moves through 4 phases: Discover Define, Develop, and Deliver
- **Hybrid Evaluation Node:** Local SLM evaluates user inputs for quality. High confidence (>0.7) bypasses the **Gemini API**, significantly reducing latency.
- **Self-Correction Loop:** Internal validation verifies every generated question is strictly Socratic before it reaches the user.
- **Persistence Layer:** Real time serialization to Google Firestore allows users to pause and resume sessions seamlessly across devices.
- **Interface:** Responsive UI featuring "**Phase Pills**" designed to support dynamic workshop settings.
- **Output:** Generates an **Actionable Strategic Roadmap**, synthesizing the user's ideas into a concrete project plan.

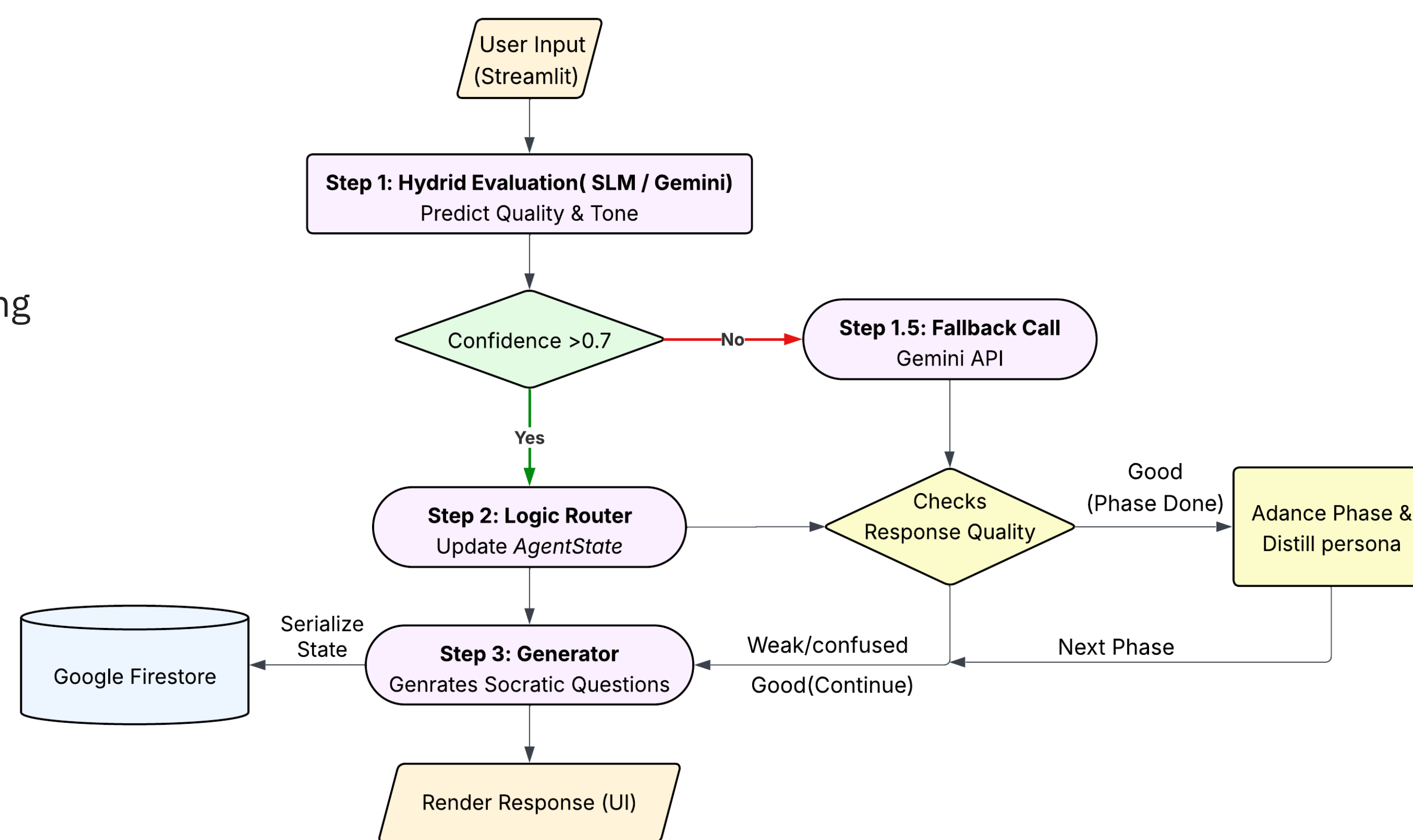


Fig 3: System Architecture. Hybrid Evaluation Node (Pink) acts as a gatekeeper to prevent unnecessary and costly calls to the Gemini API

CONCLUSION & KEY TAKEAWAYS

SocraDesign successfully validated the **Hybrid AI architecture**, eliminating the latency and memory bottlenecks of its prototype.

- **Hybrid Efficiency:** Offloading evaluation to local SLMs reduced latency by ~50%.
- **Seamless UX:** Firestore persistence enables true long-running design sessions.

With the core architecture stabilized, **future work** will be:

- **DeepEval Integration:** Incorporating **Answer Relevancy** and **Socratic Depth** metrics for evaluation reproducibility.
- **Token Optimization (TOON):** Serializing historical context into a custom, compact format to minimize token overhead and **achieve latency reduction** during context injection.

MORE INFORMATION



Demo



GitHub



References



Hugging Face Space



RetinAI (Diabetic Retinopathy Recognition and Recommendation System)

Soroush Saririan (MSADS)

SCAN ME



Introduction

Diabetic Retinopathy is a leading cause behind preventable blindness. This is a significant challenge for public health, and one of the main factors is the very subtle early onset indicators. This is a major issue in areas where healthcare is limited or areas without proper medical facilities. This can lead to delayed diagnosis, resulting in unrepairable eye damage. To solve this issue, the goal of this project is to use VLMs, specifically the use of LLaVA, which is specifically tailored for medical image datasets. This method will allow for medical professionals to gain a fast and secondary opinion for their initial classification, providing an additional line of defense against preventable blindness.

Objective

The primary objective is to develop a cost-effective screening tool using open-source VLMs to provide medical professionals with a fast, secondary opinion

- **Develop:** A custom LLaVA model fine-tuned on diabetic retinopathy datasets
- **Explain:** Generate both a classification grade and a text explanation for the diagnosis
- **Support:** Act as an additional line of defense against preventable blindness

Datasets and Preprocessing

1. IDRID (Indian Diabetic Retinopathy Dataset)

- Smaller dataset (413 train images) with pixel level segmentation masks
- **Purpose:** Used for initial prototyping to reduce train time

2. APTOS 2019 Blindness Detection

- Large scale dataset (3,662 train images)
- **Purpose:** Diversity and size allow for better generalization on classification

Preprocessing Pipeline

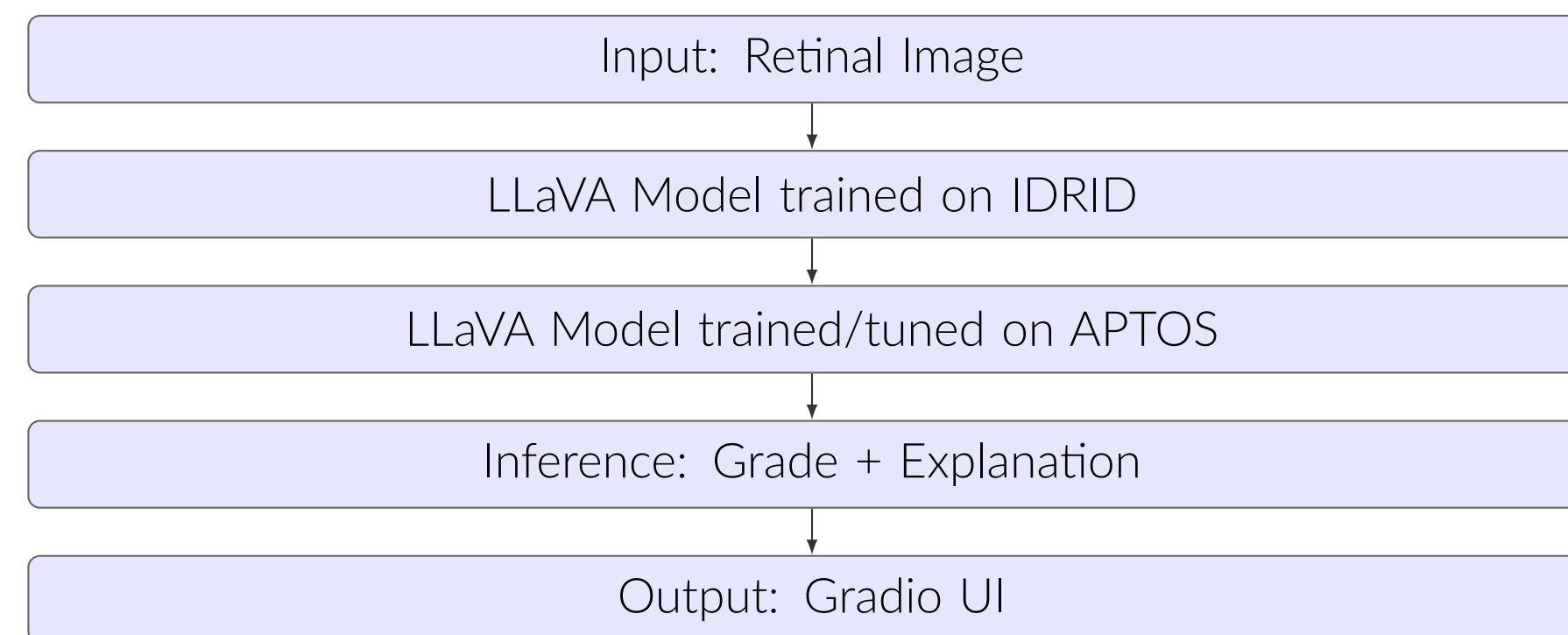
- Images resized to uniform dimensions and normalized.
- **Addressing Bias:** The large volume of Grade 0 (No DR) data was managed to prevent model bias against positive cases.

Architecture and Methodology

Model Framework

- **Core Model:** llava-hf/llava-1.5-7b-hf (Hybrid Vision Encoder and a LLM)
- **Hardware:** Trained on HiperGator via L4 GPUs
- **Optimization:** Used 4 bit quantization and LORA to reduce memory use

Two Step Training Strategy



Results

Overall Performance (APTOS Test Set)

- **Accuracy:** Achieved **81.08%** overall accuracy
- **Reliability:** 100% of test images generated valid diagnosis strings

Class	Precision	Recall	F1-Score
No DR	0.97	0.99	0.98
Mild DR	0.59	0.52	0.55
Moderate DR	0.68	0.86	0.76
Severe DR	0.50	0.01	0.01
Proliferative DR	0.57	0.46	0.51

Table 1. Model Performance on APTOS 2019 Test Set

Key Findings

- **Screening Success:** Excellent performance on No DR (Recall 0.99), making it reliable for identifying the healthy patients.
- **Severe DR Blind Spot:** The model struggled to define boundary features for Severe DR (Grade 3), often misclassifying them as moderate or proliferative

Gradio Interface

A user friendly web interface was developed for healthcare providers.

- **Input:** User uploads a retinal image
- **Output:**
 - Predicted Diagnosis Grade on a Scale 0-4
 - Generated text summary explaining detected features

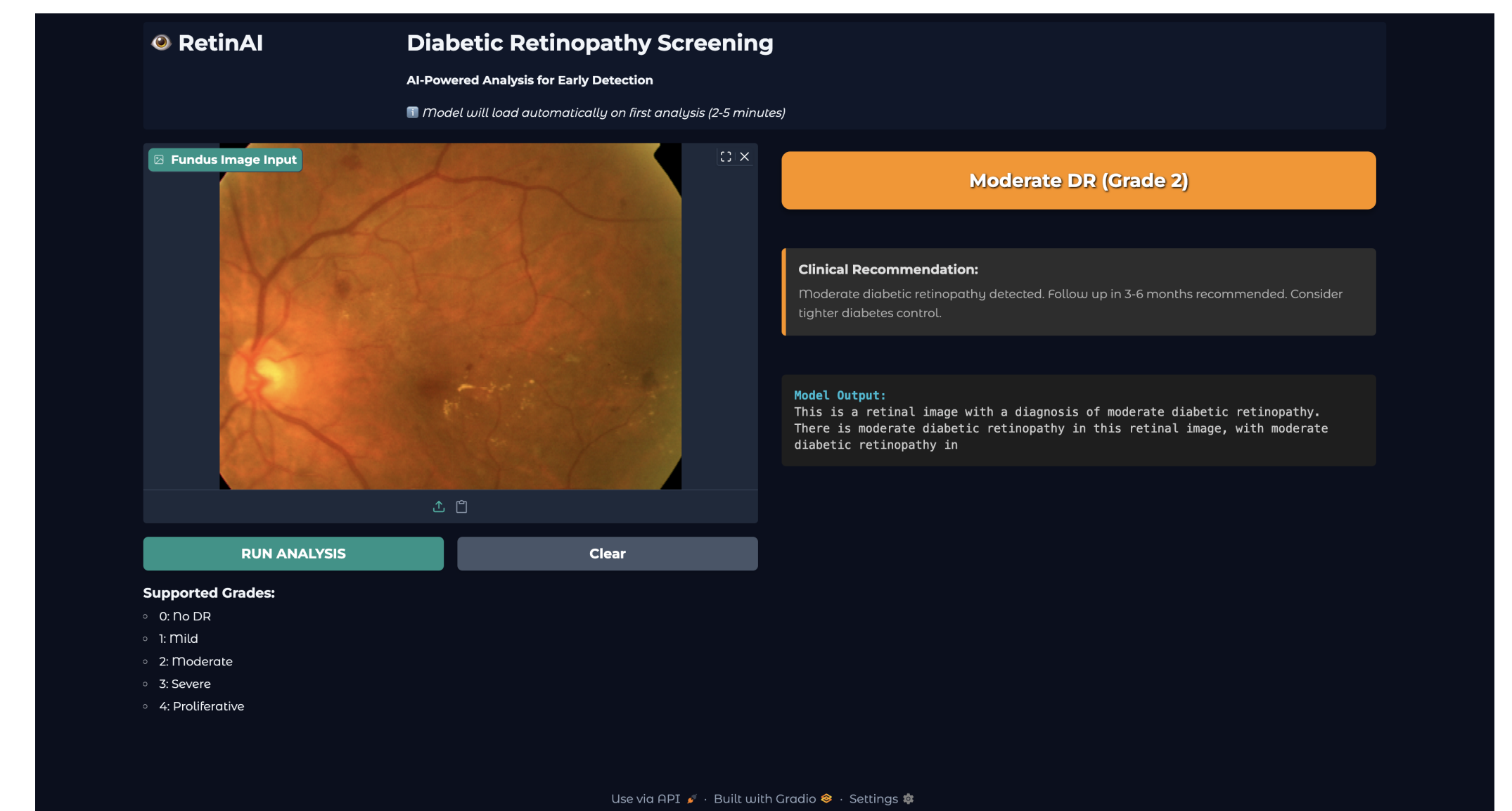


Figure 1. Web UI

Challenges and Ethical Concerns

- **Class Imbalance:** Significant imbalance in datasets requires augmentation to improve detection of Grade 3 (Severe DR).
- **Ethical Bias:** The IDRID dataset may not be fully representative of all ethnic groups, potentially affecting performance on diverse populations.
- **Medical Reliance:** This tool is a support system, not a replacement for human diagnosis.

Conclusion and Future Work

The RetinAI project successfully demonstrated that VLMs can effectively screen for Diabetic Retinopathy with high accuracy (81%).

Future Steps: Augmentation: Address the Severe DR blind spot by increasing data for Grade 3.

AGENTIC AI-POWERED DATA ENGINEERING FRAMEWORK

Automating Ingestion, Validation, Transformation, and Visualization Using Multi-Agent AI

Aniket Malpure

aniketmalpure@ufl.edu



M.S. Applied Data Science

University of Florida

Abstract

This project introduces an end-to-end Agentic AI-powered data engineering pipeline that automates ingestion, validation, transformation, storage, and visualization. By integrating LangGraph-based multi-agent orchestration with large language models, the system can interpret natural-language instructions and autonomously execute complex data engineering workflows. The framework significantly reduces manual workload, improves reliability, and accelerates analytical decision-making.

Problem Statement

Modern data engineering pipelines are manual, repetitive, and prone to human error.

Challenges include:

- Ingesting multi-source datasets
- Ensuring schema consistency and data quality
- Writing complex SQL/PySpark transformations
- Structuring Medallion-architecture layers
- Producing visualizations for rapid insights

This project aims to automate these tasks using AI-driven, autonomous agents.

Dataset

Olist Brazilian E-Commerce Dataset from Kaggle

- 100k+ customer transactions
- 8 relational tables
- Orders, reviews, payments, geolocation, sellers, products
- Real-world multi-table relational structure

Used to benchmark ingestion, joining, validation, transformation, and visualization.

Methodology

Ingestion Agent

01 Fetches GitHub CSVs, loads SQLite tables, converts raw data into the Bronze layer.

Validation Agent

02 Performs schema checks, missing value detection, data profiling, and drift analysis.

Transformation Agent

03 Uses LLM-guided column-aware logic to perform joins, aggregations, feature engineering, and filtering. Produces the Silver layer.

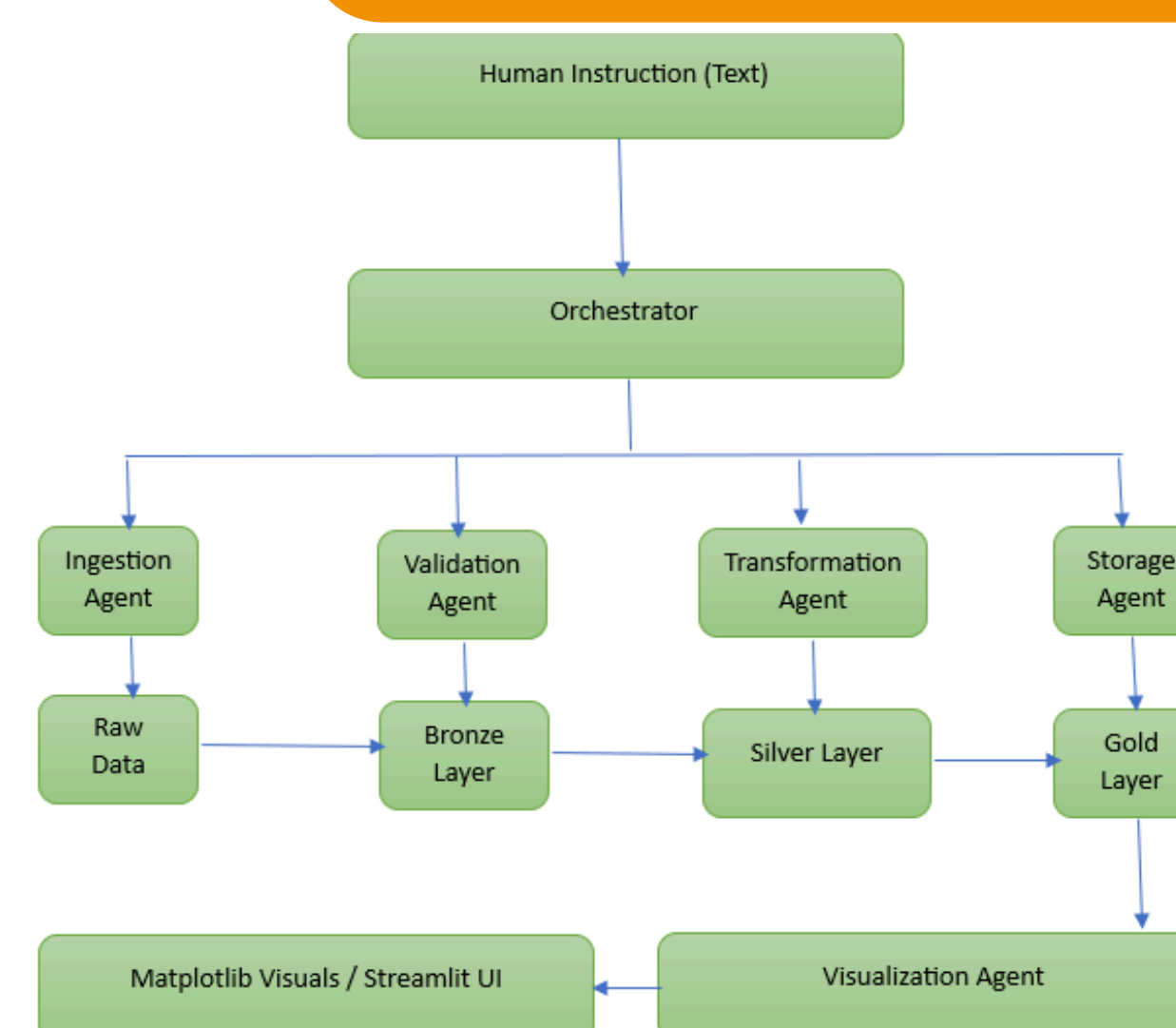
Storage Agent

04 Organizes data into Gold layer formats with metadata, versioning, and reproducibility.

Visualization Agent

05 Generates Matplotlib plots purely from user instructions, fully automated via AI.

System Architecture



Orchestrator

A LangGraph workflow that routes state between agents and ensures sequential execution.

The figure shows a Multi-agent architecture connected through a LangGraph Orchestrator, implementing Medallion (Bronze → Silver → Gold) Data Layers.

Results & Evaluation

Stage	Success Rate	Avg. Latency
Ingestion	100%	10.2 sec
Validation	100%	7.8 sec
Transformation	96%	80.5 sec
Storage	100%	8.5 sec
Visualization	92%	13.0 sec

Key Takeaways

- Agentic AI is highly effective for structured data workflows
- Multi-agent orchestration reduces errors and improves reliability
- Medallion architecture ensures clean data lineage
- The system accelerates analytics for data scientists and business teams
- LLM-guided transformations bridge the gap between natural language and SQL/PySpark logic

Demo Video



Scan this code to watch the full demo of the Agentic AI Data Engineering Framework.



Acknowledgements

I extend my sincere gratitude to Professor Andrea Ramirez Salgado and the Department of Engineering Education for their guidance and support throughout this project.



VisionARM: A Vision-Based Hand Gesture Controlled Robotic Arm System

University of Florida | Artificial Intelligence System Project

Shivansh Ade (shivansh.ade@ufl.edu) | Guide: Prof. Andrea Ramirez Salgado

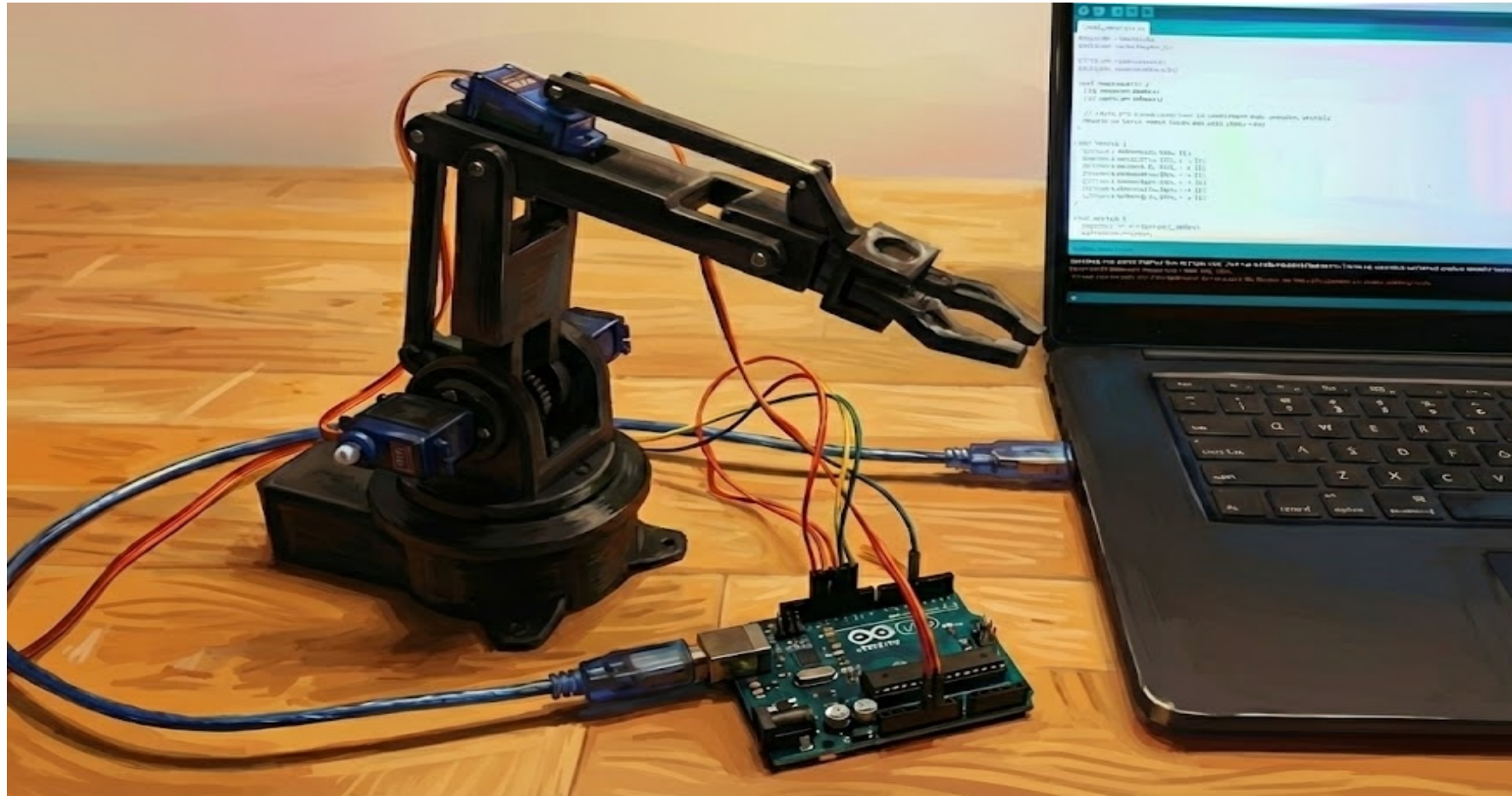
UF

Key Achievement: 98.5% gesture classification accuracy with approx. 20ms latency on a dataset of 2,000 hand gesture samples

PROBLEM STATEMENT

Traditional robotic systems rely on physical buttons, or controllers, which introduce several limitations in real-world environments. In hazardous settings, such as electrical maintenance zones, chemical labs, or high-heat industrial settings, operators must approach control panels that may become unsafe, contaminated or physically inaccessible. Mechanical controllers also suffer from stiffness, wear, and heat related degradation, causing frequent maintenance and reduced reliability. In sterile or cleanroom environments, every surface touched increases the risk of contamination and requires repeated sanitisation or PPE changes. Additionally traditional controllers restrict operators to fixed locations, limiting remote intervention and increasing exposure to risk.

VisionARM addresses these challenges by replacing physical touch points with a fully contactless, gesture based vision interface. By minimising physical mechanical interaction, it reduces contamination, decreases operators fatigue, and enhances safety by allowing remote operation from a safer distance.



INTRODUCTION

VisionARM is an intelligent hand gesture controlled robotic arm system that uses computer vision and machine learning to interpret hand gestures in real-time.

The system leverages Google MediaPipe for precise hand landmark detection and a custom-trained Support Vector Machine (SVM) classifier to recognize 8 distinct hand gesture labels.

Built with accessibility and usability in mind, VisionARM features a complete user interface powered by Gradio, secure user authentication, real-time performance monitoring with Prometheus, and seamless Arduino integration.

CORE FEATURES

- Real-Time Recognition:** 8 different hand gestures using MediaPipe + SVM
- High Accuracy:** 98.5% classification accuracy on test data
- Low Latency:** ~20ms average processing time per frame
- Web Interface:** Intuitive Gradio UI accessible from any browser
- User Management:** Secure authentication with SHA-256 encryption
- Hardware Integration:** Arduino serial communication for robotic arm
- Performance Monitoring:** Real-time FPS, latency, CPU/Memory tracking

DATASET

Total Samples: 2,000 | Features: 10 binary finger states | Classes: 8 gesture types

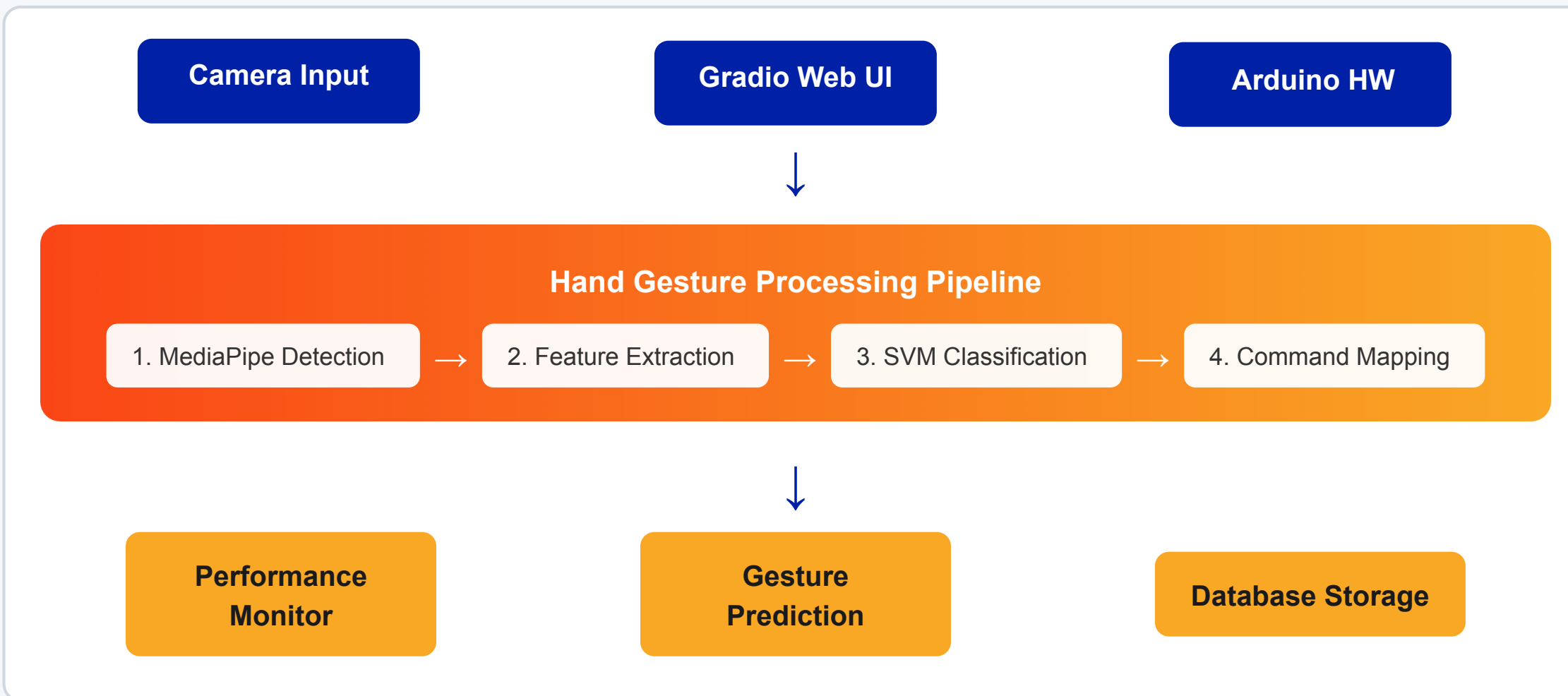
Class Distribution:

stop: 200 grab: 200 drop: 200 up: 200 down: 200 left: 200 right: 200

invalid: 600

Data Sources: Synthetic data (manually created binary patterns) + Real data (MediaPipe-encoded hand landmarks from real-time captures)

SYSTEM ARCHITECTURE



System Stack Layers:

Layer	Components
1. Input Processing	Webcam → OpenCV → MediaPipe
2. Feature Extraction	Hand Landmarks → 10D Binary Encoding
3. Classification	SVM (RBF Kernel)
4. Command Translation	Gesture → Arduino Commands
5. Monitoring	Prometheus Metrics
6. User Interface	Gradio Web UI

METHODOLOGY

Feature Description:

10 binary features representing finger states for both hands (thumb, index, middle, ring, pinky). Each feature is 1 if finger is raised, 0 if closed.

Model Training:

Support Vector Machine with RBF kernel trained on 2,000 samples. Model achieves 98.5% accuracy with cross-validation.

TECHNOLOGIES USED

Computer Vision and ML:

MediaPipe, OpenCV, scikit-learn, NumPy, Pandas

Web and Monitoring:

Gradio, Plotly, Prometheus, psutil

Hardware and Security:

PySerial, Arduino, hashlib (SHA-256)

RESULTS AND PERFORMANCE

98.5%

Classification Accuracy

~20ms

Average Latency

Metrics Tracked:

System Metrics:

- FPS (Frames Per Second)
- Latency per frame
- CPU/Memory Usage

Application Metrics:

- Total Frames Processed
- Gesture Distribution
- Session Duration

Real-time Visualizations:

FPS Over Time, Latency Distribution, Resource Usage graphs, and Gesture Frequency charts powered by Plotly

FUTURE ENHANCEMENTS

Planned Features:

- Drift Detection (NannyML)
- 20+ gestures support
- Multi-Hand Tracking
- Mobile Applications

Research Directions:

- Adaptive Learning
- Context-Aware Recognition
- Sub-10ms Latency
- Energy Efficiency

ACKNOWLEDGMENTS

Special thanks to **Professor Andrea Ramirez Salgado** for providing hardware resources and valuable guidance.

Framework support: Google MediaPipe Team | University of Florida

You can access the VisionARM's GitHub Repository by scanning this QR code-->

