SYNTHEIA

Real-Time Emotion Recognition on the Edge

Illinois Institute of Technology

Authors: Sanggil Lee and Gowtham Kumar Kamuni

ECE 501 – Artificial Intelligence and Edge Computing

Instructor: Dr. Jafar Saniie
Summer 2025, June

Table of Content:

Group Members:

Sanggil Lee
Graduate Student, Electrical and Computer Engineering
Illinois Institute of Technology, Chicago

Graduated from SeoulTech in South Korea with a Bachelor's in Electrical and Information Engineering. Currently pursuing a Master's in Electrical and Computer Engineering at Illinois Tech, specializing in computer networks and working under the guidance of Dr. Lin Cai. Passionate about building reliable systems that connect people and machines seamlessly.

Gowtham Kumar Kamuni
Graduate Student, Artificial Intelligence
Illinois Institute of Technology, Chicago

Completed my Bachelor's in Computer Science back in India, where I built my foundation in programming and problem-solving. Now diving deep into the world of AI at Illinois Tech, blending logic with intuition to build systems that don't just compute — they understand. SYNTHEIA is a reflection of that journey.

1. Introduction

1.1 Problem Statement

Human communication is layered, what we say is often just the surface. Beneath that are facial expressions, micro-reactions, and emotional cues that shape how we truly feel. Most machines today miss that entirely.

Existing emotion recognition systems are either cloud-based, overly complex, or rely on multi-modal inputs like audio and physiological signals, making them impractical for real-time use in sensitive environments. For example, a therapist or a teacher might benefit from emotionally aware systems, but not if they require an internet connection or invasive sensors.

There's a clear gap: a lightweight, privacy-respecting, real-time solution that can read emotions through facial expressions alone, running entirely on an edge device.

1.2 Objective

This project aims to develop SYNTHEIA, a real-time, embedded AI system designed to detect and interpret human emotions using facial expressions alone. It will be deployed on the Jetson Orin Developer Kit, without relying on cloud services or storing personal data.

Our goal is to:

- Use pretrained deep learning models like MobileNet or Mini-Xception, optimized for edge deployment
- Deliver live emotion classification with confidence scores
- Maintain low latency, high responsiveness, and a privacy-first design
- Provide a clean foundation for future multimodal integration (voice, biometrics)

SYNTHEIA is not built to judge or detect deception. It's built to understand, and to quietly recognize the emotions that often go unspoken.

1.3 Motivation

The idea behind SYNTHEIA didn't come from a research paper. It came from real life.

Whether it's someone masking their feelings in a conversation, or a person struggling silently in a classroom, emotions often go unnoticed. We wanted to build something that could give people a better chance of being understood, even when they don't have the words to express what they feel.

This project is more than an AI tool. It's a step toward building emotionally aware technology that functions in real time, respects privacy, and can work in real-world environments without compromise.

Emotion is data, and SYNTHEIA is designed to listen to it, without needing you to say a word.

2. Methodology & Model Architecture

The goal of SYNTHEIA is to process a live video feed, detect facial expressions, and classify them into emotional states in real time, entirely on an edge device (Jetson Orin). To achieve this, we're using a modular pipeline composed of standard computer vision and deep learning components, optimized for embedded deployment.

2.1 System Workflow

SYNTHEIA follows a five-stage processing flow:

1. Video Capture
   A USB or CSI camera captures continuous video frames in real time.
2. Face Detection
   Each frame is scanned using lightweight face detection algorithms (OpenCV's Haar cascades or Dlib CNN face detector) to isolate the face region.
3. Preprocessing
   Detected face images are resized (48x48 or 64x64), normalized, and converted to grayscale or RGB based on the model requirement.
4. Emotion Classification
   The preprocessed face image is passed through a trained CNN model (initially pretrained, later fine-tuned) to classify it into one of seven emotion categories:
   Happy, Sad, Angry, Fearful, Surprised, Disgusted, Neutral
5. Result Display & Logging
   The predicted emotion and its confidence score are displayed on a local interface (LCD or GUI), and optionally logged with timestamps.

2.2 Model Selection

We're using a hybrid model strategy to balance performance and real-time capability.

- Base Model:
  - Mini-Xception or MobileNetV2
  - Lightweight, proven performance on FER-2013 dataset
  - Pretrained weights available, compatible with edge deployment
- Optional Fine-Tuning:
  - If time permits, we will fine-tune on a small, self-collected dataset to adapt the model to real-world lighting, camera angles, and facial diversity.
- Model Format & Optimization:
  - Trained in TensorFlow or PyTorch
  - Converted to ONNX and optimized using TensorRT for Jetson deployment

2.3 Dataset

- FER-2013
    - Public dataset with ~35,000 labeled grayscale facial expression images
    - 7 emotions, 48x48 pixel images
    - Used for initial training, benchmarking, and validation
- Custom Dataset (Optional)
    - Collected using team member expressions or volunteer input
    - Used for fine-tuning or overfitting robustness tests

2.4 Output & Feedback System

- Display: Emotion prediction + confidence shown on-screen
- Logging: Optionally stores emotion + timestamp for analysis
- No data is stored permanently, ensuring privacy by design

2.5 Real-Time Constraints

To support smooth performance on the Jetson Orin:

- Target frame rate: 10–20 FPS
- Average inference time per frame: <100ms
- Batch size: 1 (real-time streaming)
- Minimal RAM footprint and energy-efficient design

This pipeline ensures that SYNTHEIA is not only effective in recognizing emotions but also responsive enough to run in real-world embedded environments.

3. Hardware Components

SYNTHEIA is designed to run entirely on embedded hardware with no external dependencies. The selected components offer a balance between performance, portability, and real-time processing capability.

3.1 Jetson Orin Developer Kit

The NVIDIA Jetson Orin Dev Kit is the heart of SYNTHEIA. It includes:

- An NVIDIA Ampere architecture GPU with Tensor Cores, optimized for deep learning
- A 64-bit ARM Cortex CPU for efficient general processing
- CUDA, cuDNN, TensorRT support for accelerated inference
- Onboard interfaces for USB, camera input, and display output

The Jetson Orin enables the entire pipeline, from video capture to emotion classification, to run locally and in real-time, without internet access.

Fig: NVIDIA Jetson Orin Nano Developer Kit

3.2 Camera Module

- A USB or CSI camera is connected to capture live video frames.
- It supports at least 30 FPS at 720p, ensuring that facial micro-expressions can be detected without motion blur.
- The camera feed is processed directly by the Jetson board.

3.3 Optional Display (LCD or Monitor)

- We plan to use a connected display to visualize:
  - Detected face
  - Emotion label
  - Confidence score
- The display also provides a user-friendly interface for testing and showcasing the system in demos.

3.4 Power Supply & Housing

- The system will be powered via standard USB-C or DC input.
- For portability and safety, we aim to encase the Jetson and components in a custom enclosure (3D-printed or modular case).

Fig: Nvidia Jetson Orin Nano Super Developer Kit - ARM Cortex A78AE 6x 1.5GHz, Nvidia Ampere + 8GB RAM

4. Software Components

SYNTHEIA is powered by a lightweight, modular software stack designed specifically for embedded AI on the NVIDIA Jetson Orin. The entire pipeline is optimized for real-time processing, efficient memory use, flexibility and compatibility with edge devices.

4.1 Operating System & SDK

- Ubuntu (Jetson flavor): The Jetson Orin runs a custom version of Ubuntu provided through NVIDIA's JetPack SDK.
- JetPack SDK:
    - Includes essential drivers, libraries, and runtime environments
    - Comes pre-loaded with CUDA, cuDNN, TensorRT, and DeepStream support
    - Enables hardware-accelerated AI inference using the onboard GPU

4.2 Deep Learning Frameworks

- TensorFlow / Keras or PyTorch:
    - Used for training and prototyping the emotion classification model
    - Initial experiments will use pretrained models like MobileNet or Mini-Xception
- ONNX (Open Neural Network Exchange):
    - Model format used to convert trained models to be inference-ready on the Jetson
- TensorRT:
    - NVIDIA's inference engine that speeds up model performance
    - Reduces latency and improves frame rate during real-time classification

4.3 Image Processing Libraries

- OpenCV:
    - Used for capturing frames from the camera, detecting faces, and preprocessing the image before feeding it into the classifier
    - Also used to draw bounding boxes, display results, and handle basic GUI interactions
- NumPy:
    - Handles matrix operations, image array manipulation, and feature scaling

4.4 GUI & Output Display

- Flask or Tkinter:
    - Flask (for web-based GUI) or Tkinter (for on-device window GUI)
    - Will display:
        - Real-time video
        - Predicted emotion label
        - Confidence percentage

4.5 Supporting Tools

- Jupyter Notebooks / VS Code: For model experimentation and debugging

- TensorBoard: For visualizing training logs and performance metrics

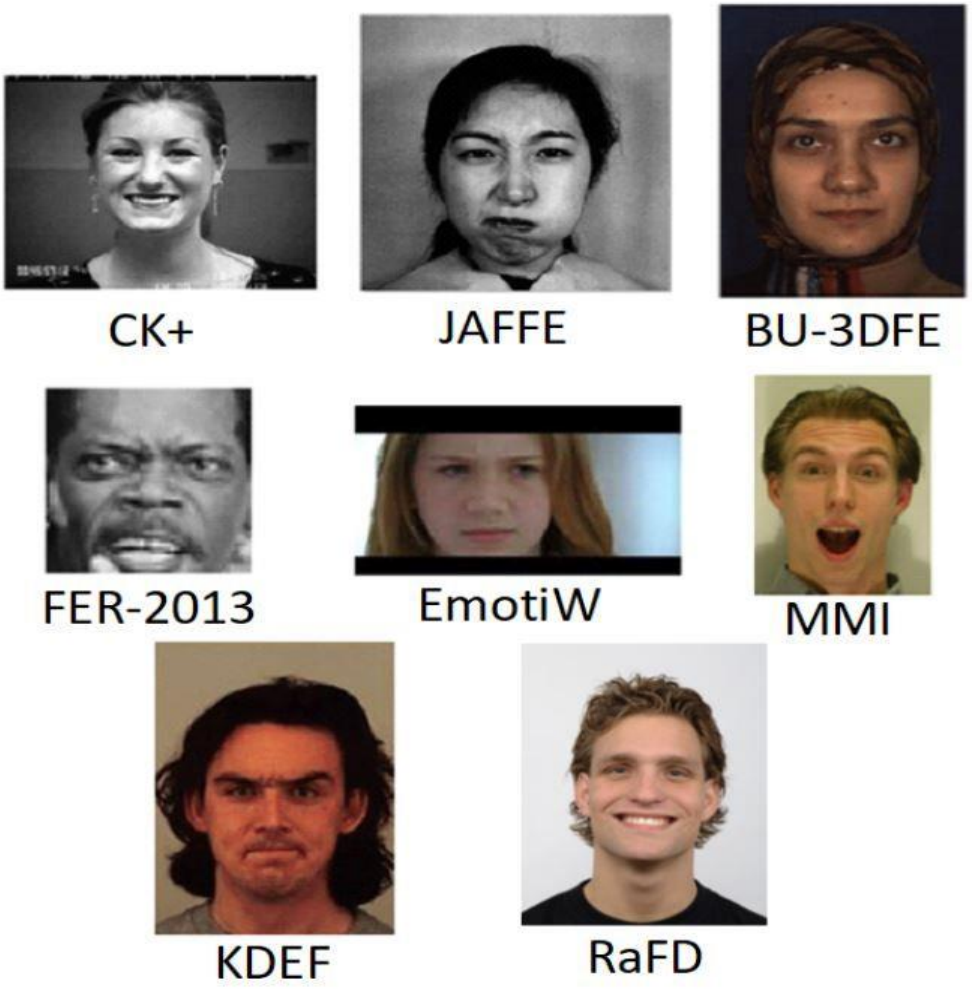

Fig: ***Example images of facial emotion recognition datasets***

In our project, we are primarily using the FER-2013 dataset for model training and evaluation due to its public availability and real-world facial diversity. However, we acknowledge other established datasets like CK+, JAFFE, MMI, and EmotiW, which can be explored in the future for cross-dataset validation or fine-tuning. These datasets represent different demographic groups, image qualities, and expression intensities, and highlight the breadth of research in facial emotion recognition.

Understanding these datasets helps us better align our training strategy and anticipate challenges in generalizing the model to real-world conditions.

5. System Features and Functionality

SYNTHEIA isn't just a system that runs, it's a system that responds. Every decision in this project is built around making the system not only functional but fluid, fast, and aware, all while running completely offline.
Here's what SYNTHEIA is designed to actually do once deployed on the Jetson Orin.

5.1 Real-Time Emotion Recognition

- SYNTHEIA continuously captures video input from the connected camera.
- It detects and isolates facial regions in real time using OpenCV-based algorithms.
- Each face is passed through a CNN model (e.g., Mini-Xception or MobileNet), which predicts the user's emotional state.
- Supported emotions: Happy, Sad, Angry, Fearful, Surprised, Disgusted, Neutral

The system provides a live, frame-by-frame classification, with an average target of 10–20 frames per second.

5.2 Confidence Score Display

- For every prediction, the model outputs a confidence score — a percentage showing how certain the system is about the detected emotion.
- This score will be shown along with the emotion label on-screen, providing transparency and trust for the user.

5.3 On-Device Output Interface

- The results will be displayed on a local interface:
    - Either a connected monitor, or
    - A simple GUI built using Flask or Tkinter
- This interface will show:
    - The video feed
    - A bounding box around the detected face
    - The predicted emotion and confidence level, live, without lag

There's no need for internet access, no data uploads, everything runs and stays on the device.

5.4 Privacy by Design

- SYNTHEIA does not store video or audio recordings.
- If logging is enabled, it will only save:
    - Timestamps
    - Predicted emotion labels
    - Confidence scores

This makes the system suitable for environments like:

- Therapy sessions
- Classrooms
- Research labs

- Human-computer interaction studies

5.5 Optional Logging & Analysis

- The system can optionally log the emotional timeline, storing predictions and timestamps for offline analysis.
- This feature is useful for researchers who want to analyze emotional patterns over time, without compromising user privacy.

5.6 Real-World Testing and Robustness

- SYNTHEIA is being developed with edge-case behavior in mind, dealing with:
    - Varying lighting conditions
    - Slight head movements
    - Partial occlusion
- The goal is to keep performance stable even when the input isn't perfect.

5.7 Future Expandability (Built-In)

Even though this version focuses on facial expressions, SYNTHEIA is built to evolve. The software and hardware stack are ready for:

- Voice-based emotion detection
- Biometric fusion (e.g., heart rate sensors)
- Multi-face detection
- Context-aware emotion analysis

The modular codebase and scalable model architecture make it easy to plug in new capabilities when the time comes.

SYNTHEIA doesn't just classify emotions. It creates a moment, a brief, intelligent acknowledgment that someone is being seen. That's what it means to be emotionally aware. That's what this system is all about.

6. Milestones and Deliverables

The project will be completed over the course of four weeks. Each week is focused on a specific phase of development, ensuring a structured, goal-oriented progression toward a functional prototype of SYNTHEIA.

Week 1 – Setup and Dataset Preparation

- Jetson Orin hardware and camera setup
- Install software stack (JetPack SDK, OpenCV, TensorFlow/PyTorch)
- Collect and organize the FER-2013 dataset
- Perform initial model testing on desktop for benchmarking

Week 2 – Model Training and Inference Integration

- Fine-tune pretrained models (MobileNet / Mini-Xception) using FER-2013
- Convert and optimize model using ONNX and TensorRT
- Begin integration of model into Jetson pipeline
- Set up basic real-time inference loop on test video feed

Week 3 – Interface and Display System

- Build GUI using Flask or Tkinter to display live predictions
- Overlay emotion labels and confidence scores on video stream
- Implement optional logging for timestamped emotion data
- Test system in varied lighting and facial orientation conditions

Week 4 – Testing, Evaluation, and Finalization

- Measure system performance: FPS, latency, prediction accuracy
- Debug any bottlenecks or edge cases
- Finalize all documentation and prepare demo materials
- Submit final report and deliverable package

Final Deliverables

- Real-time emotion recognition system running on Jetson Orin
- GUI displaying predictions with confidence
- Logged emotion data (if enabled)
- Final project report and codebase
- Live or recorded demo video

7. Challenges

While the core idea behind SYNTHEIA is straightforward, building a real-time, reliable, and privacy-conscious emotion recognition system on embedded hardware introduces several technical and practical challenges. We've identified the following key areas where difficulties are expected:

7.1 Real-Time Performance Constraints

Running deep learning models on edge devices like the Jetson Orin, especially with live video feeds, can lead to frame drops or latency issues if not optimized properly. Ensuring consistent FPS and smooth processing will require efficient model tuning and use of tools like TensorRT.

7.2 Lighting and Environmental Variability

Facial expressions can vary significantly depending on lighting, camera angle, shadows, and even facial obstructions like glasses or hair. Making the system robust enough to handle real-world conditions is a significant challenge.

7.3 Model Generalization

Pretrained models trained on datasets like FER-2013 may not always generalize well to real-world inputs. Expression intensity, facial structure, and cultural variations can lead to misclassification if the model isn't fine-tuned carefully.

7.4 Integration and Compatibility Issues

Merging multiple components, camera input, model inference, GUI, and logging, into a single pipeline on a low-power device often results in compatibility problems, driver issues, or memory bottlenecks.

7.5 Privacy and Ethical Considerations

Even though SYNTHEIA doesn't store data, there's still a responsibility to design the system in a way that respects user privacy and transparency. Avoiding unintentional data capture or unnecessary logging is a key challenge.

By identifying these challenges early, we've built flexibility into our timeline and system design. Our plan includes fallback options, modular testing, and performance benchmarking to ensure a stable, useful final system.

8. Division of Work

SYNTHEIA is being developed as a collaborative effort between two team members. Responsibilities are divided equally to ensure balanced contributions while allowing both members to support each other across different phases of the project. Each task area has been chosen based on mutual agreement, workload fairness, and opportunity to learn.

Gowtham Kumar Kamuni

- Hardware Setup: Configuring Jetson Orin, camera integration, and system deployment
- Model Optimization: Handling ONNX conversion, TensorRT integration, and real-time performance tuning
- Interface Design: Assisting with GUI layout and visual feedback output
- Testing & Final Debugging: Evaluating latency, FPS, accuracy, and stability of the system
- Documentation: Writing core sections of the report and final deliverables

Sanggil Lee

- Dataset Management: Handling FER-2013 preprocessing, augmentation, and formatting
- Model Training: Running training cycles on pretrained models and experimenting with tuning
- GUI Implementation: Setting up the real-time interface using Flask or Tkinter for emotion output
- Evaluation: Assisting in system testing, confusion matrix, and performance logging
- Presentation & Submission: Creating slides, reviewing the final report, and co-presenting

Both team members will jointly contribute to:

- Idea development and discussion
- Integration and system testing
- Final polishing, review, and oral presentation

9. Conclusion

SYNTHEIA isn't just a project about AI or hardware, it's about building something that pays attention to what most people miss: emotion that isn't spoken, moments that are silent, and expressions that pass too quickly to explain.
We started this project not just to meet a requirement, but because we believe technology should be able to understand people better, without asking them to say everything out loud. SYNTHEIA reflects that belief. It's designed to work in real-time, with privacy in mind, and without any fancy cloud setups, just quiet, efficient intelligence at the edge.
We know it's not perfect yet. But it's real. It's working. And it's the start of something that can grow, into voice recognition, biometric sensing, and beyond. But even in this first version, it proves one simple idea: machines don't have to speak to understand. Sometimes, all they need to do is see you, and listen to what you didn't say.
This is what SYNTHEIA is made for.

10. References

1. Goodfellow, I., Bengio, Y., & Courville, A. (2016). Deep Learning. MIT Press.
2. Barsoum, E., Zhang, C., Ferrer, C. C., & Zhang, Z. (2016). FER-2013: A Dataset for Facial Expression Recognition.
3. Arriaga, O., Valdenegro-Toro, M., & Plöger, P. (2017). Real-time Convolutional Neural Networks for Emotion and Gender Classification. arXiv:1710.07557
4. OpenCV Documentation – https://docs.opencv.org/
5. TensorFlow Lite – https://www.tensorflow.org/lite
6. PyTorch – https://pytorch.org/
7. NVIDIA JetPack SDK – https://developer.nvidia.com/embedded/jetpack
8. ONNX Runtime – https://onnxruntime.ai/
9. FER-2013 Dataset – https://www.kaggle.com/c/challenges-in-representation-learning-facial-expression-recognition-challenge/data
10. Jetson Orin Developer Kit Documentation – https://developer.nvidia.com/embedded/jetson-orin