GSoC 2026 Proposal
(NeoSmartML Foundation Model for Neonatal ICU Monitor Data)

Applicant: Anagha Pradeep
Organization: LibreHealth
Project Size: Large (~350 hours)
Mentors: Saptarshi Purkayastha, Sherri Bucher
Gitlab : https://gitlab.com/anaghapradeep404
University: Indiana University Indianpolis
Location: Indiana, USA(EDT (UTC-4))
Email: anaghapradeep404@gmail.com

ABOUT ME

My name is Anagha Pradeep. I am a Master's student in computer science who is currently interning in IntelinAir. My current project I am working on is weather and crop disease prediction systems where I am developing pipelines that integrates sensor and satellites data to provide when farmers should use fungicides. Prior to pursuing my current degree, I was working as Software engineering for 2 years on HMI systems, middleware systems and infrastructure. Now I am shifting my career to AI/ML. When I saw NeoSmartML project, I wanted to contribute in this project as it combines edge deployment, multimodal learning, and real clinical impact. Contributing to an open-source project like NeoSmartML, under supervision of experienced mentors, will help me in learning new skills and topics which can be useful in my career/future. I approached this project by surfing and reading documentation and codebase to create mental modal of this system, to know what code does but also why it is built, where are the risks, gaps and what is missing. I am still honing my ML skill but I can use my software engineer experience, familiarity in embedded systems and firmware, writing unit tests and maintainability in addition to model accuracy.

MOTIVATION

WHY GSOC?

I want to experience how can I work in real world project (focusing AIML) not just course work. This project is an intersection of my interest Edge ML and Social impact. So, working under mentor who is already experienced in this field will help me grow and learn new things. Also, this is my first experience with open-source contribution even though I worked with a team in my previous company. By participating in Gsoc, will give me structured and accountability to see it through properly.

WHY LIBRA HEALTH ?

This organisation caught my attention because of their mission as this directly aligned to my interest social impact, which is providing quality medical care to everyone regardless of race, status or location. They also focus on how technology can be used for real world health impact. This made me want to join this organisation and contributing to this org means the work I do actually matters beyond my grade.

WHY THIS PROJECT AND WHAT I LEARNT FROM IT?

This project NeoSmartML project captures NICU monitor images using an ESP32 CAM and extracts vital through OCR. The goal is to build a foundational model that can handle both image and time series data for tasks like risk classification, apnea/bradycardia prediction and treatment response. There are two challenges I faced. One is multimodal architecture fusing visual and structured data meaningfully and other one is edge deployment. From my quantization work on CheXNet, I learned how ESP32 has very limited memory and processing prower. Quantizing only Linear layers gave me a result less than 0.2% size reduction because DenseNet is Conv2d-dominated. This kind of issues is what makes this problem genuinely hard and interesting.

LEARNING EXPECTATION

1. Learn how to design and train a multimodal architecture
2. Understand edge deployment limitation deeply (not just theory but on real world hardware)
3. Learn how to collaborate with domain experts (ML researchers)

CONTRIBUTIONS

I went through the entire codebase for security and reliability issues, and found few issues below:

1. Unsafe strcpy in main.c
2. Risk in the FastAPI upload handler
3. WiFi credentials accidentally committed to the repo
4. Unauthenticated Bluetooth Low Energy configuration
5. Outdated pandas method

	PULL REQUEST	DESCRIPTION
1.	**!23 - OCR Code Refactor** (closes #9)	Replaced 8 nearly identical extract_* and plot_* functions with a single generic extract_vital_value() and plot_vital_results(), driven by a VITAL_CONFIGS dictionary. Reduces 300 lines to ~100 and makes adding new vital signs require only one dict entry. Includes passing tests.
2.	**!24 - CI Pipeline Fix** (closes #10)	Fixed a pre-existing yaml invalid error in .gitlab-ci.yml that was causing 0 jobs to run across all MRs.

		The sast job was defined twice manually and inside the included SAST template causing a conflict. Removing the redundant definition restored the pipeline for all contributors.

PROJECT UNDERSTANDING

In this project, Code uses ESP32 CAM to capture NICU monitor images and OCR for vital signs extraction.

The goal is building foundational modal for handling both image and temporal data for multiple prediction tasks such as risk classification, apnea/bradycardia prediction, treatment response, and assessment.

Primary Model difficulty is to develop a multimodal architecture that integrates:

1. Vision encoder – raw monitor images
2. OCR extraction layer – structured vital signs
3. Temporal modal – to analyse trends for multiple time increments
4. Multi – task prediction

Secondary difficulty is edge deployment – ESP32 has very limited memory and processing power So, it needed quantization, pruning, distillation to make it run.

DELIVERABLES

1. A working multimodal foundation model (vision + OCR + temporal)
2. Trained and evaluated on NICU monitor data with documented metrics
3. Compressed version (quantized + pruned) that runs on or near ESP32 hardware
4. Multi-task prediction heads for at least the core clinical tasks
5. Documented codebase with tests, merged into the main repo
6. A write-up or report explaining architecture decisions

PRELIMINARY WORK

I completely both preliminary tasks mentioned in project forum.

Task 1: Dynamic Quantization of CheXNet

Used: Pytorch dynamic quantization (torch.ao.quantization.quantize.dynamic) on model targeting both nn.Linear and nn.Conv2d Layers.

Findings: quantizing only on nn.Linear layers didn’t give accurate compression for DenseNet-121

Because model was domainated by Conv2d layers and final classifier is single Linear layer.So linear layer quantization without including Conv2d reduced the size by 0.2%.

The results:

- Original checkpoint (model.pth.tar): **84 MB**
- Quantized model (chexnet_dynamic_int8.pth): **approx. 28 MB**
- Verification: max |float − quantized| = **0.39** (expected for int8 dynamic quantization)

The script can handle two different model formats and figures out which one it is so you don't have to tell model manually.

Task 2: Rotation Sensor Integration + Bulk Download Endpoint

Rotation sensor (ESP32 firmware)

I have added MPU6050 gyroscope sensor to support esp camera (esp_camera_ble_control.ino) instead of using Adafruit library because library was crashing (due to IRAM overflow when combined with BLE + Camera + WiFi on the AI Thinker ESP32-CAM) My approach aimed usage of less memory and adding fallback so it can work even if no sensor is connected.

Bulk download endpoint (app_httpd.cpp)

I also tried to resolve issue #4 (endpoint to download multiple images from the ESP32 at once instead of one by one).

/download?count=N endpoint - added to read N images from SD card and stream them as Gzip file.While working on this, I found and tried to fix a bug where it shows 404 when there is no count.i just added default to 10 images to avoid that.

I am still working on and testing this change before submitting as merge request.

PROPOSED ARCHITECTURE

I am proposing this architecture for foundational model after reviewing project description and codebase. Goal here is to achieve project target and refine it if needed.

Multimodal Encoder

Vision branch:

I used lightweight AI model (MobileNetV3 - small) – to process raw images of NICU monitor.

Why this modal? – I chose this over denseNet-121 because it is lighter. It uses depthwise separable convolutions that takes only limited memory IRAM on small chip ESP32.

Also, Quantization friendly – model is easier to quantize. So, it can run faster on limited resource without any accuracy issue.

OCR pipeline:

Here I used EasyOCR to get accurate numbers of Heart rate (HR), Oxygen (SpO2), RR, temperature etc from screen.

Projection layer: We are using vision branch as eyes to see pictures on display and use OCR as reader to get number on screen. I am using small layer to translate whatever result or number I get into standard digital format. So, I can use this part to combine later stages.

Fusion:

This where we use cross attention fusion layer instead of vision and ocr separately

Smart matching: This will act as Spotlight. It uses the number to search the image (visual feature map) for specific area that match.

How to solve timing problem: video and data streams cannot sync perfectly . so we can use attention method which allows ai to learn exactly which part of image aligned / important for which number, making systems reliable than guessing.

Temporal Sequence Modelling

Transformer Encoder:

This is for recognising patient's vitals while watching watch 30 sec highlight reel.

Pattern recognition: Here this wont look at numbers, it looks at trajectory (why heart rate trending down?) and oscillations (is the oxygen level is changing?)

Early warning: By analysing this pattern, we can spot signs of apnea (breathing stops) or bradycardia (slow heart rate), before any risk occurs.

Structure: transformer encoder is very lightweight (4 layers, 128 hidden dims) which means fast but also can process complex math to understand patterns in that 30 sec reel.

GRU (Gated Recurrent Unit):

Smaller and quicker: It uses two "Gates" (Update and Reset). Simple math filters that decide (Keep this new heart rate info, or ignore it?). It requires only less processing power and memory than transformer for edge deployment.

Multi-Task Prediction Heads

Instead of using one giant layer, we assign small networks called Adaptor Heads for each task.

Freeze the backbone: Heavy lifters like vison encoder (mobileNetV3) and temporal Encoder (GRU) should be frozen after initial training. Save their weights in esp32" s read only memory and never change.

Adaptor Heads: 2-layer MLP (Multilayer perceptron)

Structure:

Input -> Linear Layer (128 -> 64 dims) -> ReLU -> Linear Layer (64 -> Output dims).

These heads are tiny. We can switch between tasks without reloading the massive backbone.

Mutli-task loss function:

We use combined loss function to train foundational model so that it works for all task at once. This then force backbone to learn feature which is useful for everything, not specific.

Total loss – Weighted Sum of individual task losses

As an alternative strategy, instead of updating weights, we can prepend few learnable tokens to the temporal input similar to prefix tuning. We are training/ updating tokens for new task, the backbone remains unchanged. This is useful when even adapter is too costly on the edge.

Edge Compression Pipeline

Post-Training Quantization:

Model use float32 numbers (long decimals) as it can take more memory. So, we can convert them to int8 (Linear + Conv2d) which reduce 3x of model size instantly because whole number only take up way less than decimals.

Now when I run NICU photos through model, we will get result like heart rate is always between these 2 shades. Then we can lock those result into model. so that ESP32 doesn't haven't guess it, can just calculate which save battery and space.

Structured Pruning:

Here we check whether any attentions heads are idle, meaning their activation magnitude is low. If head is not contributing, we remove it completely. This makes the model smaller and faster. Fewer calculations mean faster predictions; we get results in real time.

Knowledge Distillation:

we will train a smaller version of model to mimic the original model. So this smaller system will learn pattern lighter than original(probability). Target size will be 5mb because it can be fits in ESP32 memory.

BENEFITS TO COMMUNITY:

Also, this project improves neonatal care by automatically predicting ICU monitor data and alerting real time event like apnea and bradycardia before increase in risk. It also runs without needing powerful computing resources which won't be available in every hospitals. Making this open source can be utilized and improvised by people who is in need.

VALIDATION AND DOCUMENTATION

Validation:

1. Evaluate each prediction head (accuracy, F1, AUC per task)
2. Test smaller system vs full model – for accuracy drop after quantization
3. Edge case testing
4. Test on real ESP32 hardware

Documentation:

1. Explaination for every architecture and decision and why

2. Setup guide so in future people can refer to the document
3. Documentation for compression – original size vs ccompressed and accuracy before vs after
4. Create unit tests
5. Proper description for MRs
6. Also write weekly blog about my work.

TIMELINE:

Period - week	Work
Community bonding	Study codebase, set up environment, discuss plan with mentors
1–2	Vision encoder + OCR embedding layer
3-4	Fusion layer + unit tests
5-6	Temporal sequence model
7-8	Prediction heads + training loop
9-10	Evaluation and metrics
11	Model compression (quantization + pruning)
12	ESP32 integration + documentation
13	Buffer week - feedback and fixes

WHY I'M THE RIGHT CANDIDATE

During preliminary work, quantizing only Linear layers gave me a result less than 0.2% size reduction because DenseNet is Conv2d-dominated by running few experiments. For esp32, I avoided Adafruit library because it was causing memory(IRAM) overflow when combining Bluetooth, camera and WIFI. Instead, I used the sensor directly via raw I2C register access which used less memory than before.

Apart from the tasks, I re-factored the OCR pipeline reducing over 200 lines and also fixed a CI issue which blocked every contributor from running pipelines and reviewed a full codebase by finding security issues with the required file and references. I have 2 years of Software experience and I always think about what works with the real word rather than just wwhat's written in a notebook. I work best with async communication which fits the remote GSoC format well. I am someone who takes full ownership of the work I submit. I may not have years of ML experience but I have strong software engineering foundations, I ask good questions, and I push through problems until I understand them properly. I think that mindset matters especially in a project that handles real clinical data from neonatal patients.

REFERENCE

[1] Tariq, A., Celi, L.A., Newsome, J.M., Purkayastha, S., Bhatia, N.K., Trivedi, H., Gichoya, J.W., Banerjee, I. (2021). Patient-specific COVID-19 resource utilization prediction using fusion AI model. *NPJ Digital Medicine*, 4(1), 1-9.

[2] Mahajan, Y., Pinnamraju, J., Burns, J.L., Gichoya, J.W., Purkayastha, S. (2022). Using Machine Learning Approaches to Identify Exercise Activities from a Triple-Synchronous Biomedical Sensor. *Lecture Notes in Networks and Systems*, vol 418. Springer, Cham.

[3] Sinha, P., Gichoya, J.W., Purkayastha, S. (2022). Leapfrogging Medical AI in Low-Resource Contexts Using Edge Tensor Processing Unit. *2022 IEEE Healthcare Innovations and Point of Care Technologies (HI-POCT)*. IEEE.

[4] Sinha, P., Tummala, S.S., Purkayastha, S., Gichoya, J. (2022). Energy Efficiency of Quantized Neural Networks in Medical Imaging. *Medical Imaging with Deep Learning*.

[5] LibreHealth Community Forum. NeoSmartML Foundation Model for Neonatal ICU Monitor Data. https://forums.librehealth.io

[6] GitLab: https://gitlab.com/librehealth/incubating-projects/mhbs/neosmartml

[7] Arduino/ESP32 MPU6050 Datasheet and I2C Register Map. Retrieved from ESP32 firmware implementation during preliminary task 2.