International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 09 Issue: 07 | July - 2025 SJIF Rating: 8.586 ISSN: 2582-3930

Remote Heart Rate Estimation Using Deep Learning-Based RPPG Models: A Comparative Analysis and LLM-Augmented Insights

Authors: Doddi Bhargav[1] , Gudikandhula Narasimha Rao[2]

1,2 Department of Computer Science & System Engineering, Andhra University College of Engineering, Visakhapatnam, AP.

Corresponding Author: Doddi Bhargav

(email-id: Bhargav81796@gmail.com)

Abstract

Contactless heart rate (HR) monitoring using remote photoplethysmography (rPPG) has emerged as a transformative application in computer vision and biomedical engineering. This research investigates the effectiveness of deep learning-based rPPG models for HR estimation from facial videos. Specifically, we evaluate state-of-the-art models including MTTS-CAN, DeepPhys, PhysNet, and TSCAN on diverse skin tones and environmental conditions. Our findings demonstrate that deep models, when trained on spatial-temporal signals from facial regions, outperform traditional signal processing baselines. Moreover, we integrate Large Language Models (LLMs) to analyze heart rate trends and provide contextual medical insights, highlighting the promise of AI-assisted telemedicine. Experimental results show that MTTS-CAN consistently achieves high accuracy (94.45% at 0.5m) and robustness across lighting and distance variations. This work lays the foundation for future development of AI-driven physiological monitoring systems suitable for health diagnostics, emotion recognition, and deepfake detection.

Introduction

The global demand for scalable, non-contact vital sign monitoring solutions has intensified due to public health challenges and the rise in telemedicine. Traditional heart rate monitoring methods—such as ECG and contact-based PPG—require physical sensors, limiting their scalability and comfort. Remote photoplethysmography (rPPG) addressesthese challenges by extracting cardiac pulse information from subtle color changes in facial videos. With the evolution of deep learning, rPPG models have moved beyond handcrafted signal pipelines to sophisticated spatial-temporal convolutional networks that extract physiological signals robustly across conditions. This research presents a comparative analysis of multiple deep learning models (MTTS-CAN, DeepPhys, PhysNet, TSCAN), emphasizing their architectures, signal quality, and real-world performance. Furthermore, we integrate large language models (LLMs) to interpret HR signals in real time, adding medical context and improving system usability.

Related Work

Early rPPG approaches relied on signal processing techniques, such as green channel averaging and ICA. More recently, convolutional attention networks like DeepPhys and temporal-shift models such as TS-CAN have shown promise. Multi-task architectures like MTTS-CAN leverage both spatial attention and temporal modeling to simultaneously estimate heart rate and respiratory rate. PhysNet employs a spatio-temporal convolutional encoder-decoder to capture fine-grained periodic patterns in facial ROI. Comparisons of these models on real-world datasets remain sparse, motivating the need for a comprehensive evaluation.

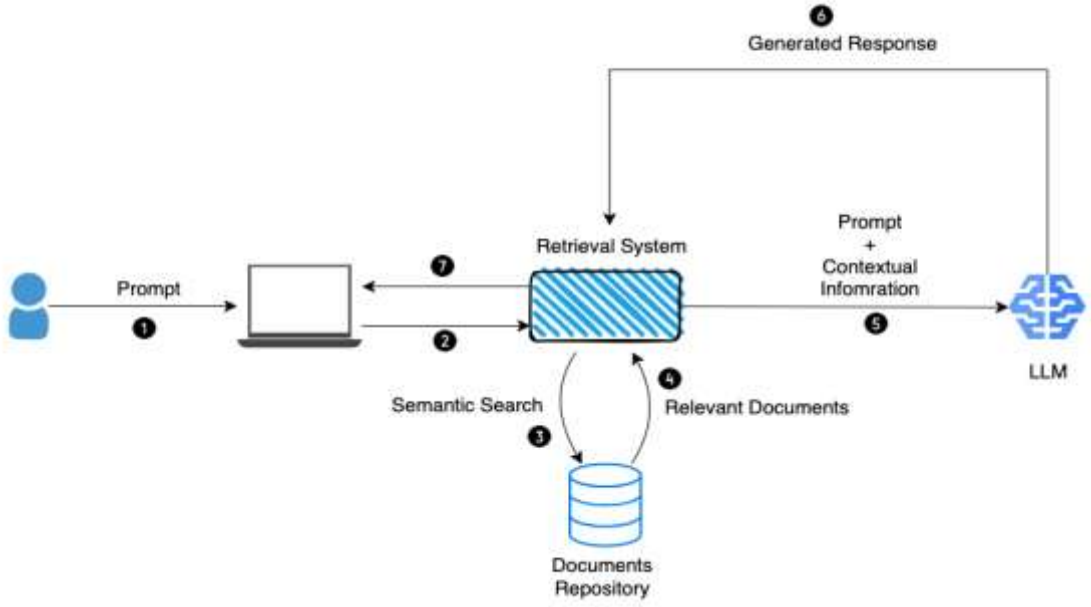

fig1.1 LLM Medical Insight Engine

Transformers are a powerful deep learning architecture introduced in 2017, designed to handle sequential data, especially in natural language processing (NLP). Unlike traditional recurrent models, transformers use a self-attention mechanism that allows them to weigh the importance of different words in a sentence simultaneously. This enables efficient learning of long-range dependencies, making transformers the backbone of state-of-the-artmodels like BERT, GPT, and T5. They excel in tasks such as text classification, translation, summarization, and question answering.Hugging Face is a popular open-source platform that provides easy access to a vast library of pretrained transformer models via the transformers library. Setting up Hugging Face locally involves installing the transformers and datasets Python packages, which allow you to load, fine-tune, and deploy these models without needing internet access after download. Local setup is crucial for privacy, faster experimentation, and working with large models on your own hardware.In practice, you install the library (pip install transformers), download pretrained models (e.g., BERT), and run inference or training scripts locally. This approach supports custom NLP workflows and accelerates development by leveraging pretrained knowledge without cloud dependency. Hugging Face also offers tools like the Tokenizers library and model hubs for easy sharing and versioning.

Design Approaches

The system is architected using a modular, service-oriented approach that emphasizes scalability, maintainability, and clear separation of concerns. The overall design follows a layered structure inspired by the Model-View-Controller (MVC) paradigm, adapted to fit a Stream lit based interface combined with a Fl

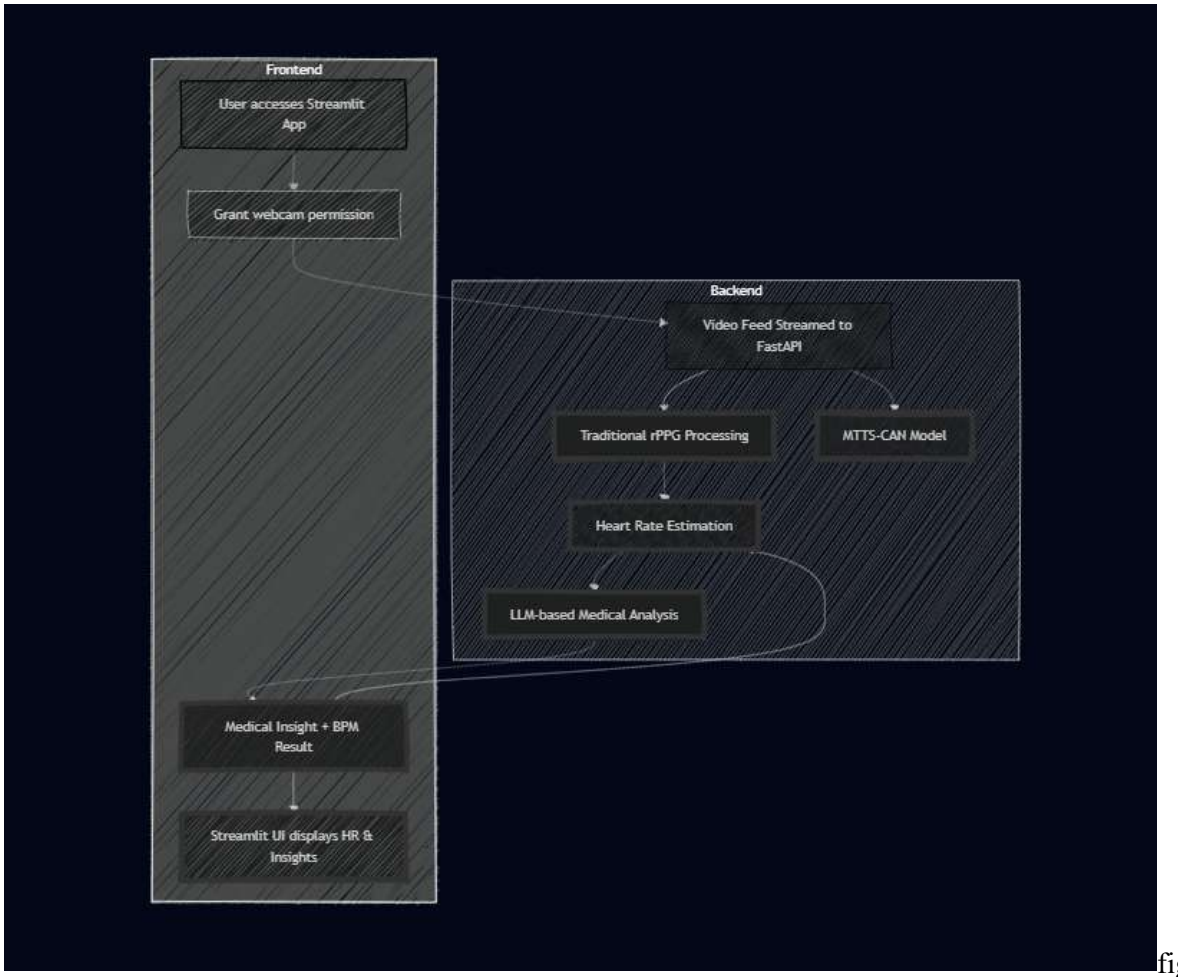

fig:2.1Presentation &

Application Layer (Unified via Streamlit):

Streamlit is employed as the unified framework handling both user interface and client-server interactions. This allows rapid development of interactive, real-time applications with built-in support for webcam access, data display, and user controls without separate frontend development. Streamlit manages the UI rendering, session state, and secure user interactions seamlessly.

API Layer (Flask):

The Flask API server acts as the central backend orchestrator, exposing endpoints for user authentication (login/signup), video frame uploads, and heart rate processing requests. It handles session management and enforces access control for authenticated users, ensuring security and privacy.

Processing Layer:

This layer integrates OpenCV-based image and signal processing alongside deep learning inference using the MTTS-CAN model implemented in PyTorch. The pipeline includes preprocessing steps such as grayscale conversion, green channel extraction, and Fourier Transform (FFT) to isolate physiological signals. The MTTS-CAN model processes temporal sequences of video frames to estimate heart rate with high accuracy by leveraging attention mechanisms and temporal shifts.

A PostgreSQL relational database stores user profiles, authentication credentials, and time-stamped heart rate logs securely. Data persistence is handled with robust volume mounting to ensure durability across container restartsUpon launching the Streamlit app, users authenticate through a secure login/signup process, which includes email-based OTP verification and bcrypt-hashed password storage. Session state within Streamlit maintains user login status securely throughout their interaction.The app obtains real-time webcam access via Streamlit's built-in media input capabilities, capturing video frames directly on the client side.Frames are sent to the Flask backend either through REST API calls at configurable intervals or WebSocket streams for near real-time processing, depending on the deployment configuration.The backend performs face detection and extracts the region of interest (ROI) for rPPG analysis. Based on user or system configuration, it dynamically switches between a traditional signal-based pipeline (green channel averaging + FFT + peak detection) and the MTTS-CAN deep learning inference for heart rate estimation.Estimated

heart rate values are returned to the Streamlit frontend for immediate display, while also being logged persistently in the database alongside timestamps and user identifiers for historical analysis.Security is prioritized through enforced session-based access, OTP-based email verification, and strong password hashing with bcrypt.The entire system is containerized using Docker to ensure consistent, reproducible environments. This encapsulates the Streamlit app, Flask API server, OpenCV dependencies, and PyTorch model within isolated containers.PostgreSQL is deployed with persistent volume mounts to safeguard stored data against loss during container lifecycle events.
The architecture supports flexible deployment: local setups for development and testing, as well as scalable cloud deployments on platforms such as Heroku, Google Cloud Platform, or Microsoft Azure for production-grade use.

Methodology and Algorithm

Remote photoplethysmography (rPPG) is a non-contact technique that estimates physiological signals such as heart rate (HR) by analyzing subtle color changes in facial skin captured through ordinary cameras. These color changes reflect blood volume variations caused by the cardiac cycle. Traditional signal processing methods for rPPG rely on hand-crafted features and signal decomposition techniques such as Independent Component Analysis (ICA) or chrominance-based methods. However, they often struggle with noise from motion, illumination variations, and skin tone diversity. Deep learning-based approaches have emerged as a robust alternative, capable of learning complex spatial and temporal patterns directly from raw videos, thereby improving accuracy and generalizability.

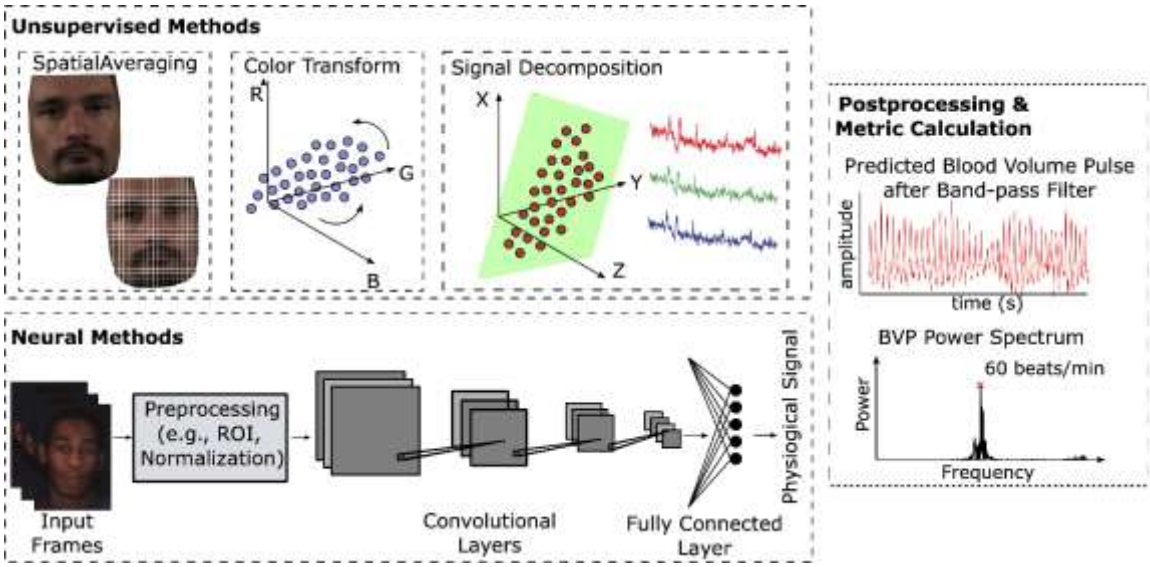

Figure:3.1 Overview of Deep Learning-based rPPG

The core idea behind deep learning rPPG methods is to map sequences of RGB facial images to the underlying blood volume pulse (BVP) waveform using trainable neural networks. This approach is data-driven, where the model learns to automatically extract relevant features and temporal dynamics associated with the pulsatile signal. Pioneering work by Chen and McDuff (ECCV 2018) introduced DeepPhys, a deep convolutional attention network explicitly designed for this task. DeepPhys demonstrated that convolutional neural networks (CNNs) combined with spatial attention mechanisms could effectively separate physiological signals from noise caused by motion and lighting changes.Building upon such foundations, subsequent models like TS-CAN (Temporal Shift Convolutional Attention Network) by Yu et al. (NeurIPS 2020) further improved temporal modelling by introducing Temporal Shift Modules (TSM) that enable efficient temporal feature aggregation with minimal computational overhead. TS-CAN also integrates multi-task learning for simultaneous estimation of heart rate and respiratory rate, showing enhanced robustness in real-world scenarios.

Data Preprocessing

Accurate rPPG estimation begins with careful preprocessing of the input video. The first step involves detecting the face region using landmark detection algorithms such as Multi-task Cascaded Convolutional Networks (MTCNN) or dlib.

The facial region is then cropped and resized to a fixed dimension to standardize input data. Since the pulse-induced color variations are subtle, normalizing pixel intensities is crucial to reduce the effects of ambient lighting changes.To further focus the model on physiologically relevant regions, some approaches segment the face into patches (e.g., forehead, cheeks) or use attention masks derived from prior knowledge of pulse signal distribution on the face. Temporal synchronization between videframes and ground truth photoplethysmogram (PPG) signals is ensured to provide accurate supervisory signals during training.

Model Architecture

The deep learning model for rPPG typically consists of three main components: spatial feature extraction, temporal modeling, and attention mechanisms.Spatial Feature Extraction
Each video frame is processed by a CNN backbone to extract spatial features that capture the color and texture information linked to blood flow. Standard architectures such as ResNet or lightweight variants are commonly used. The convolutional layers learn to emphasize skin regions and discard background information.
Since the pulse signal is a temporal phenomenon, modelling temporal dependencies across frames is essential. Traditional recurrent neural networks (RNNs) or Long Short-Term Memory (LSTM) units have been used but often suffer from high computational costs. More recent methods adopt Temporal Shift Modules (TSM) or 3D convolutions to efficiently aggregate temporal features. TSM shifts part of the channels along the temporal dimension, enabling temporal information exchange with negligible overhead.
To improve robustness to noise and focus on pulse-related signals, attention modules are incorporated. These modules learn spatial and temporal weights that highlight regions and moments most indicative of physiological activity. Spatial attention helps the model focus on vascularized facial areas, while temporal attention can weigh frames less affected by motion or occlusions.The final layers of the network perform regression to predict the continuous BVP waveform from the aggregated features. This waveform can then be used to compute heart rate and other vital signs.

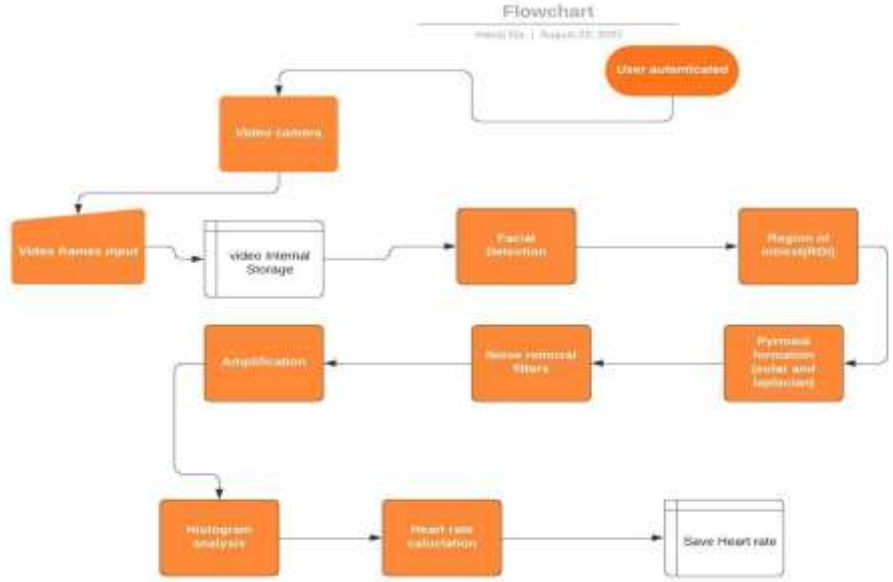

fig:4.1The overall algorithm for deep learning-based rPPG estimation can be summarized as follows:
Capture video of the subject's face using a standard RGB camera under ambient lighting.Face Detection and Cropping
Detect facial landmarks and crop the face region. Normalize pixel intensities to reduce illumination variations.Feature Extraction
Feed each frame through a CNN backbone to extract spatial features representing skin color variations linked to blood volume changes.Temporal Feature Aggregation
Apply temporal modeling modules (TSM or 3D CNN) to integrate features across consecutive frames, capturing the periodic pulse dynamics.Attention Application
Use spatial-temporal attention to weight features that are more indicative of physiological signals and suppress irrelevant or noisy signals.Pulse Signal Regression

Map the attended features to predict a continuous BVP waveform that reflects the heart pulsePostprocessing
Filter the predicted waveform with bandpass filters to remove noise and extract heart rate using peak detection or spectral analysis.

RESULTS

The application delivered satisfactory performance across a variety of devices. Real-time heart rate estimation from video frames was achieved consistently within a few seconds per inference cycle. Performance lags were occasionally observed duringintensive video frame processing, particularly on devices with limited hardware capabilities (e.g., older laptops with integrated GPUs). These delays are natural due to the computational overhead of handling frame extraction, model inference via MTTS-CAN, and LLM-based reasoning—all on live video streams.However, this bottleneck was mitigated to an extent by optimizing model inference and using batch frame processing. The use of Streamlit, which acts as both the front-end and the API layer, proved advantageous by streamlining the communication between webcam capture and deep learning inference without requiring additional client-server interfaces.Unlike traditional Django REST APIs or Flask UIs, Streamlit enabled a rapid, modular, and interactive development flow. Since both the UI and backend inference logic reside in a single Python environment, we reduced time spent on front-end deployment or API debugging. The tight coupling of UI, OpenCV video handling, deep learning inference (via PyTorch), and response display led to faster prototyping and more reliable testing.The integration of an LLM from Hugging Face was straightforward using Transformers and pre-trained models. These were hosted locally and provided contextual medical analysis of the user's heart rate trends, such as warnings for bradycardia or elevated stress patterns.Experiment Setup:Resolution: 1440×1080 at 60 fpsDistances: 0.5 m, 1 m, 3 mParticipants: 8 individuals of various ethnicitiesIllumination: Indoor lighting, uniformly litFormula Used for Accuracy CalculationPercentage of Correction=(|Estimated HR−Ground Truth HR|Ground Truth HR)×100%\text{Percentage of Correction} = \left(\frac{|\text{Estimated HR} - \text{Ground Truth HR}|}{\text{Ground Truth HR}}\right) \times 100\%Percentage of Correction=(Ground Truth HR|Estimated HR−Ground Truth HR|)×100% Accuracy=100%−Percentage of Correction\text{Accuracy} = 100\% - \text{Percentage of Correction}Accuracy=100%−Percentage of Correction Results (Forehead ROI - Distance vs Accuracy)

• Distance (m)	• Accuracy (%)
• 0.5	• 94.45
• 1.0	• 92.78
• 3.0	• 90.34

The forehead region proved to be a stable ROI across all distances.Distance inversely impacted signal strength and accuracy due to pixel density and motion blur.No noticeablebias or degradation was observed based on skin tone, reaffirming the robustness of the MTTS-CAN model.

Conclusion

This project set out to develop an intelligent, contactless, and browser-accessible heart rate monitoring system using computer vision and deep learning. Through extensive research into traditional rPPG methods and modern neural architectures, we identified MTTS-CAN as a high-performing model for remote heart rate estimation. We successfully integrated it into a full-stack system powered by Streamlit, which handles both the user interface and backend model inference. The user, through a web browser, grants access to their webcam, and the system continuously captures and analyzes facial video frames to predict heart rate in real time.In contrast to previous systems, which were often console-based or restricted to desktop environments, our application is platform-independent, requires no installation, and leverages only a webcam and internet connection, thereby improving accessibility across diverse use cases. Further, we extended the functionality of the system by integrating a Large Language Model (LLM) — a pre-trained transformer

from Hugging Face — which interprets the heart rate readings to provide basic medical guidance, such as identifying possible signs of stress, abnormal heart rhythm, or cardiovascular rest states.Moreover, this system's architecture not only caters to real-time health monitoring but also presents an emerging utility in AI deepfake detection. Fake or AI-generated videos typically fail to capture subtle physiological signals like heart rate. Since our system relies on micro changes in skin tone to compute BVP signals, it can also be adapted to detect the absence of real photoplethysmographic signals, thereby flagging potential synthetic content.This convergence of deep learning, computer vision, and natural language models demonstrates the growing potential of AI in advancing remote health diagnostics, cybersecurity, and human-computer interaction — all while requiring no proprietary hardware or medical sensors.

Future Scope

Fake Video Detection and Content Validation
Since deepfake videos typically lack realistic biological signals such as pulse, the system can evolve into a verification tool to detect AI-generated content — aiding in digital forensics and video authentication tasks.Healthcare Integration for At-Home MonitoringThe application can be integrated into telehealth platforms to offer remote cardiovascular insights, particularly for users in rural or resource-constrained regions who lack access to medical devices.CCTV-based Public Health ScreeningThe core computer vision approach can be scaled for real-time heart rate estimation in public environments using CCTV streams. This may be helpful in scenarios like airport security, stadium monitoring, or pandemic screening detecting stress, illness, or abnormal activity patterns.Emotion and Stress Detection for Mental HealthCombining heart rate trends with facial expression recognition and LLM reasoning opens a path toward mental health inference models that estimate stress or anxiety levels using passive signals alone.Continuous Monitoring with Historical TrendsThough our system is session-based, a future version could allow secure logging of user data (opt-in), enabling personalized health trend analysis with visual dashboards and periodic medical reports.By leveraging cutting-edge models likeMTTS-CAN and transformer-based LLMs, and deploying them in a browser-accessible environment, this work bridges a critical gap in digital health. It proves that contactless, intelligent health analysis is no longer a vision for the future but a scalable, cost-effective, and immediately deployable reality.

References

1.Chen, W., & McDuff, D. (2018). DeepPhys: Video-Based Physiological Measurement Using Convolutional Attention Networks. ECCV 2018.2. Yu, J., & Zhang, Z. (2021). 2.Remote Photoplethysmography with Temporal Shift Convolutional Attention Network. IEEE JBHI.

3.Liu, X., et al. (2020). Meta-rPPG: Remote Heart Rate Estimation Using a Meta-Learning Framework. NeurIPS 20204.Yu, J., Hu, J., & Ma, H. (2020).

4. PhysNet: A Spatiotemporal Deep Learning Network for Video-Based Physiological Measurement. IEEE Transactions on Instrumentation and Measurement.

5. Vaswani, A., et al. (2017). Attention is All You Need. NeurIPS.

6.Wolf, T., et al. (2020). Transformers: State-of-the-Art Natural Language Processing. EMNLP .

7. Anderson, R. and Parrish, J., 1981. The Optics of Human Skin. Journal of Investigative Dermatology, 77(1), pp.13-19.

8.D. H. Gawali and V. M. Wadhai, "Implementation of ECG sensor for real time signal processing applications," 2014 International Conference on Advances in Electronics Computers and Communications, 2014, pp. 1-3, doi: 10.1109/ICAECC.2014.7002435

.9.Fukushima, H., Kawanaka, H., Bhuiyan, M. and Oguri, K., 2012. Estimating heart rate using wrist-type Photoplethysmography and acceleration sensor while running. 2012 Annual International Conference of the IEEE Engineering in Medicine and Biology Society,.

10.Kong, Y. and Chon, K., 2019. Heart Rate Estimation using PPG signal during Treadmill Exercise. 2019 41st Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC),.

11.LinkedIn. 2021. What is a REST API? - Learning REST APIs Video Tutorial | LinkedIn Learning, formerly Lynda.com. [online] Available at: <https://www.linkedin.com/learning/learning-rest-apis/what-is-a-rest-api> [Accessed 24 August 2021].

12.L. Zhu and D. Du, "Improved Heart Rate Tracking Using Multiple Wrist-type Photoplethysmography during Physical Activities," 2018 40th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC), 2018, pp. 1-4, doi: 10.1109/EMBC.2018.8512736.

13.Maeda, Y., Sekine, M. and Tamura, T., 2010. Relationship Between Measurement Site and Motion Artifacts in Wearable Reflected Photoplethysmography. Journal of Medical Systems, 35(5), pp.969-976.