

# Neural Health Predictor

## Early Detection of Cardiac Arrest in Newborn Babies

*A Comprehensive Deep Learning System  
with 12-Model Ensemble Architecture*

Contributors:

V. Bhavana | S. Roshini | D. Sanjana

Project Guide: Ms. M.N. Sailaja

CMR College of Engineering & Technology, Hyderabad

## Table of Contents

---

- \* 1. Executive Summary
- \* 2. Problem Statement & Motivation
- \* 3. System Architecture Overview
- \* 4. Input Features & Data Pipeline
- \* 5. The 12 Neural Network Models (Detailed)
- \* 6. BERT-Based Medical Text Processing
- \* 7. Ensemble Weighted Voting Mechanism
- \* 8. Training Configuration & Hyperparameters
- \* 9. Installation & Usage Guide
- \* 10. Expected Performance Metrics
- \* 11. Technology Stack
- \* 12. Future Improvements
- \* 13. References

## 1. Executive Summary

---

The Neural Health Predictor is a state-of-the-art machine learning system designed to assist healthcare professionals in the early detection of cardiac arrest risk in newborn babies. By analyzing 10 critical physiological indicators, the system provides real-time risk assessment across three severity levels: Low, Medium, and High risk.

What makes this system unique is its ensemble approach - combining 12 different deep neural network architectures, each with its own specialization. This is analogous to consulting 12 different specialist doctors, each examining the patient from their unique perspective, and then combining their opinions for the most accurate diagnosis possible.

### Key Technical Innovations

- \* 12-Model Deep Learning Ensemble for robust predictions
- \* BioBERT integration for understanding medical context
- \* Clinical text generation from tabular data
- \* Weighted voting based on individual model performance
- \* Hyperparameter optimization with 10,000+ configurations

## 2. Problem Statement & Motivation

---

### The Clinical Challenge

Cardiac arrest in neonates (newborn babies) is a life-threatening emergency that requires immediate medical intervention. The challenge lies in the fact that symptoms can be extremely subtle and easy to miss, especially in the chaotic environment of a neonatal intensive care unit (NICU).

Traditional monitoring systems often fail to detect the early warning signs that precede cardiac arrest. By the time obvious symptoms appear, the window for effective intervention may have significantly narrowed, leading to poor outcomes including permanent brain damage or death.

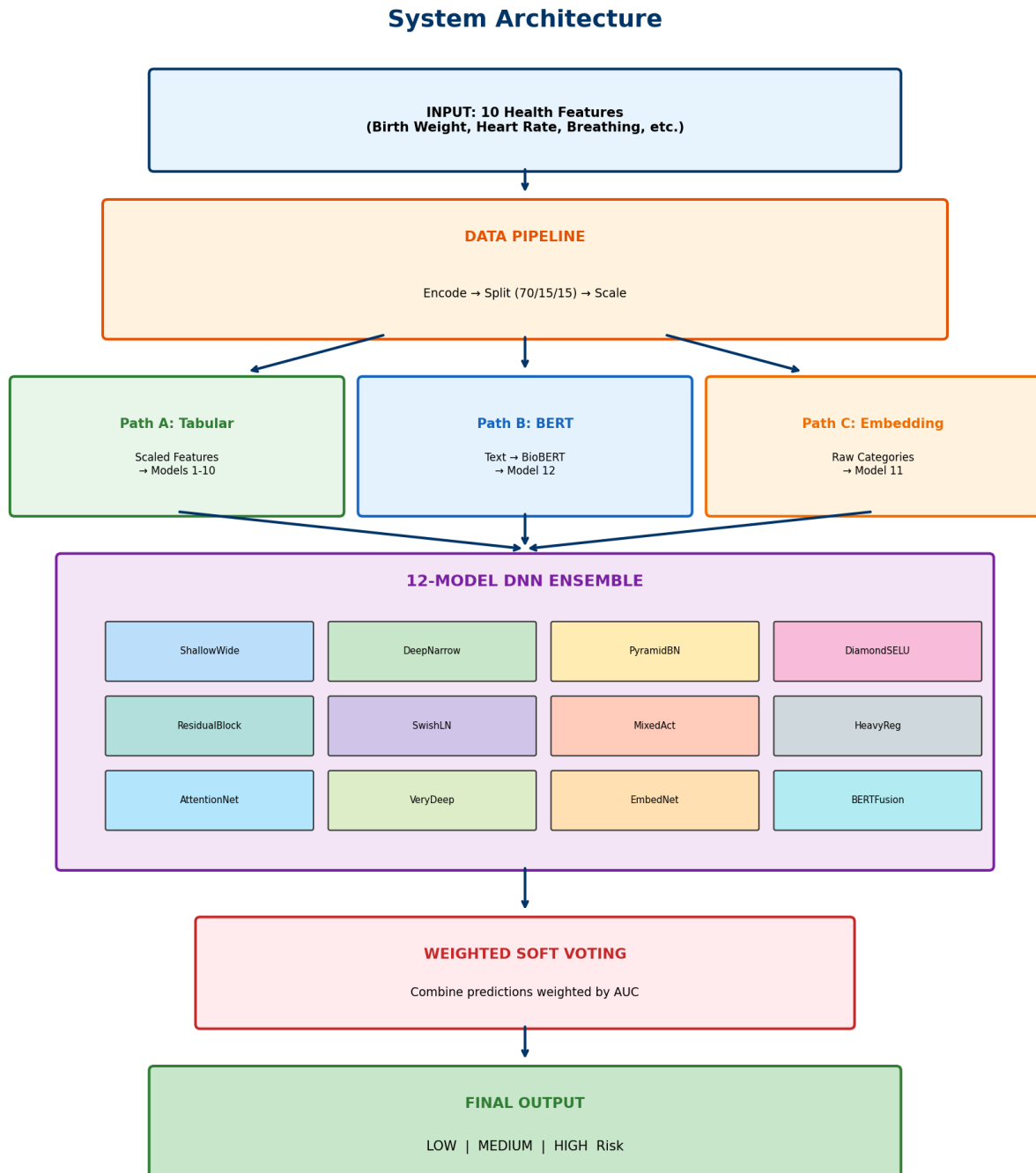
### Our Solution Approach

Our system addresses this challenge by continuously analyzing multiple physiological parameters and combining predictions from 12 different neural network architectures. This multi-model approach provides several advantages:

- \* Diversity of perspectives catches patterns any single model might miss
- \* Ensemble averaging reduces prediction variance and increases reliability
- \* BERT integration understands the contextual meaning of symptoms
- \* Real-time processing enables immediate alerting

### 3. System Architecture Overview

The following diagram illustrates the complete data flow through our system, from input features to final risk prediction:

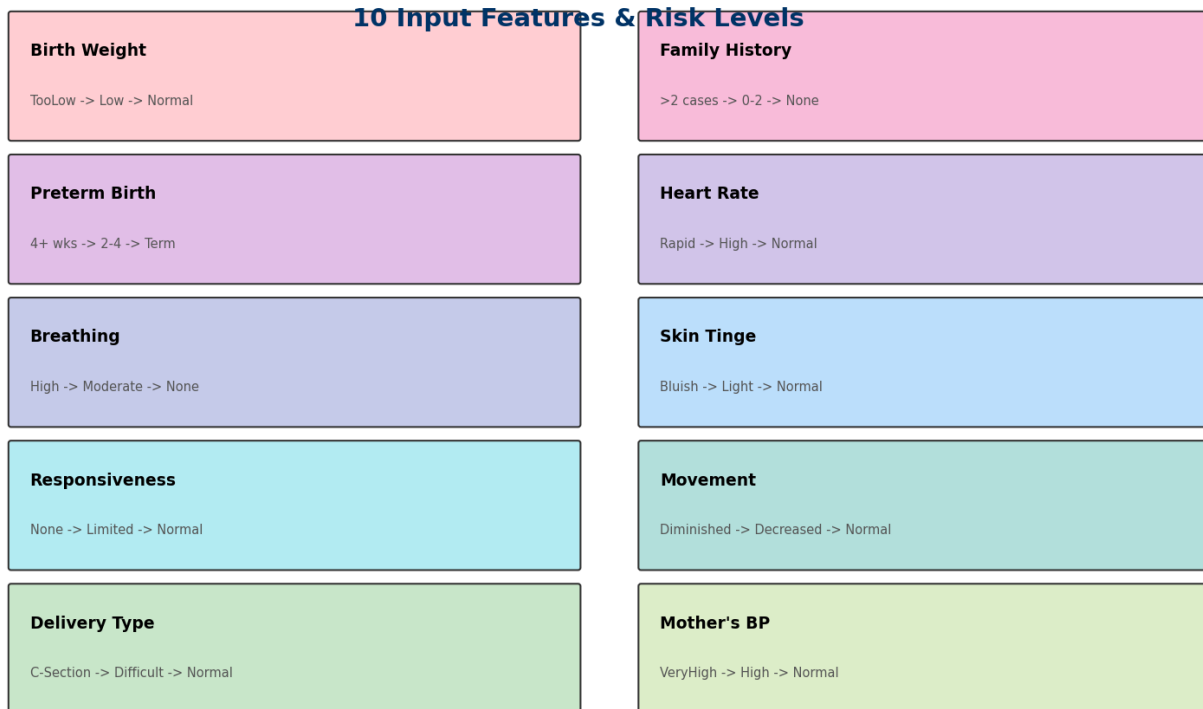


The architecture consists of three parallel processing paths that feed into the 12-model ensemble:

- \* Path A (Tabular): Scaled numerical features for standard DNN models 1-10
- \* Path B (BERT): Clinical text converted to BioBERT embeddings for Model 12
- \* Path C (Embedding): Raw categorical encodings for entity embedding Model 11

## 4. Input Features & Data Pipeline

The system analyzes 10 carefully selected physiological indicators that are clinically significant for predicting cardiac arrest risk in newborns:



### Feature Encoding

Each feature is categorical with 3 severity levels. Features are encoded to integers (1=lowest risk, 3=highest risk) for numerical processing. The system uses stratified splitting (70% train, 15% validation, 15% test) to maintain class balance across all subsets.

### Data Pipeline Steps

- \* Step 1: Load CSV data with 10 features per patient
- \* Step 2: Encode categorical values to integers using predefined mappings
- \* Step 3: Stratified split maintaining class proportions
- \* Step 4: StandardScaler normalization (fit on train, transform all)
- \* Step 5: Generate clinical text narratives for BERT processing
- \* Step 6: Extract BioBERT embeddings (768-dimensional vectors)

## 5. The 12 Neural Network Models (Detailed)

---

Each model in our ensemble has unique architectural characteristics designed to capture different aspects of the input data. Here is a comprehensive breakdown of each model:

### Model 1: ShallowWide - 'The Quick Generalist'

**Architecture:** 2 layers (256 to 128 neurons)

**Activation:** ReLU

This model uses wide layers with few depths, making it excellent at capturing broad, obvious patterns. Like a general practitioner who can quickly assess overall health status. Its simplicity also makes it highly interpretable and serves as a reliable baseline for the ensemble.

### Model 2: DeepNarrow - 'The Detailed Analyzer'

**Architecture:** 6 layers (64 neurons each)

**Activation:** ELU

With 6 layers of narrow width, this model examines data step-by-step, capable of finding subtle, complex patterns that shallower models miss. The ELU activation helps with learning faster and avoiding dying neuron problems.

### Model 3: PyramidBN - 'The Organized Thinker'

**Architecture:** 4 layers (256 to 32 pyramid)

**Activation:** GELU + BatchNorm

Gradually compresses information from 256 to 32 neurons, forcing the network to learn the most important features at each level. BatchNormalization ensures stable training and faster convergence.

### Model 4: DiamondSELU - 'The Self-Balancing Expert'

**Architecture:** 5 layers (diamond shape)

**Activation:** SELU

Diamond shape expands then contracts, allowing the network to explore a higher dimensional representation before compressing. SELU activation provides self-normalizing properties, eliminating the need for BatchNorm.

### Model 5: ResidualBlock - 'The Memory Keeper'

**Architecture:** 4 blocks with skip connections

**Activation:** ReLU + BatchNorm

Uses skip connections that allow gradients to flow directly through the network, preventing vanishing gradient problems. Even if deep layers learn poorly, the original information still reaches the output.



### Model 6: SwishLayerNorm - 'The Smooth Operator'

**Architecture:** 4 layers (128 neurons each)

**Activation:** Swish + LayerNorm

Swish activation ( $x * \text{sigmoid}(x)$ ) provides smoother gradients than ReLU, often leading to better optimization. LayerNorm normalizes each sample independently, making it robust to batch size variations.

### Model 7: MixedActivation - 'The Versatile Specialist'

**Architecture:** 4 layers with different activations

**Activation:** ReLU, ELU, GELU, Swish

Each layer uses a different activation function, allowing the network to leverage the unique strengths of each. ReLU for speed, ELU for negative values, GELU for probabilistic behavior, and Swish for smoothness.

### Model 8: HeavyRegularization - 'The Cautious One'

**Architecture:** 4 layers with L1+L2 and 50% Dropout

**Activation:** ReLU

Heavy regularization prevents the model from memorizing training data. L1 encourages sparsity (some weights become exactly zero), L2 prevents any single weight from becoming too large, and 50% dropout forces the network to learn redundant representations.

### Model 9: AttentionNet - 'The Focused Expert'

**Architecture:** Attention layer + 3 dense layers

**Activation:** ReLU

Inspired by TabNet, this model learns to dynamically focus on the most relevant features for each specific input. Not all features are equally important for every patient, and this model adapts its attention accordingly.

### Model 10: VeryDeep - 'The Exhaustive Researcher'

**Architecture:** 8 layers (varying widths)

**Activation:** PReLU

The deepest model in the ensemble with 8 hidden layers. PReLU (Parametric ReLU) learns the optimal slope for negative inputs during training. Can model extremely complex non-linear relationships.

### Model 11: EmbeddingNet - 'The Category Expert'

**Architecture:** 10 embedding layers + dense

**Activation:** ReLU

Instead of using simple integer encodings, learns 4-dimensional embedding vectors for each category value. These embeddings capture semantic similarity - for example, 'LowWeight' and 'WeightTooLow' would have similar embeddings.

## **Model 12: BERTFusion - 'The Medical Language Expert'**

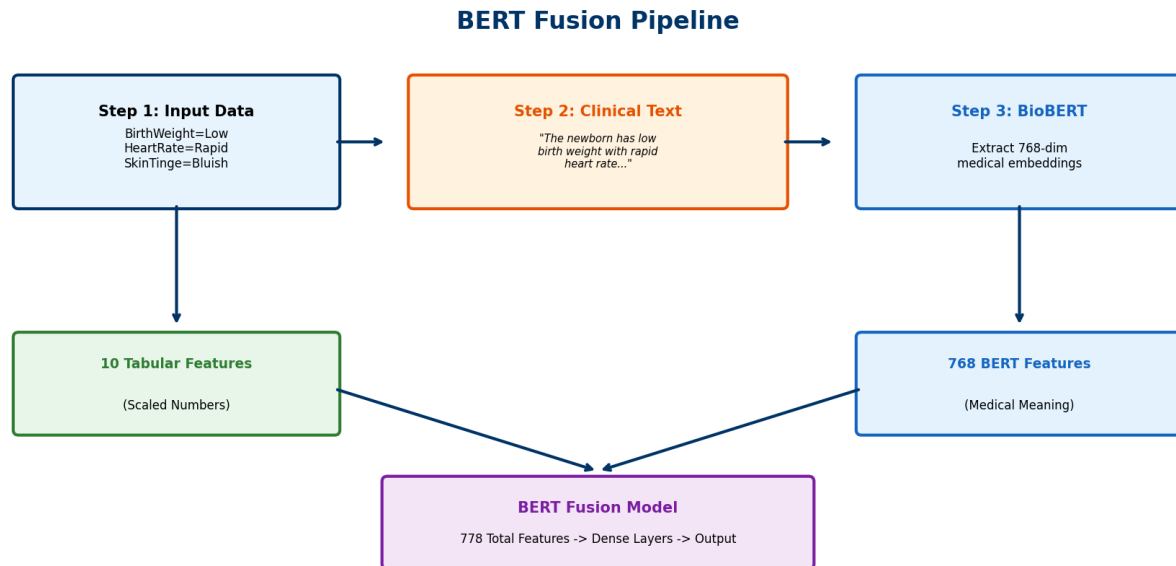
**Architecture:** 768 BERT + 10 tabular to dense

**Activation:** GELU

The star model that combines tabular features with BioBERT embeddings. Natural language understanding from millions of medical papers helps interpret clinical context that pure numerical models cannot capture.

## 6. BERT-Based Medical Text Processing

One of the most innovative aspects of our system is the integration of BERT (Bidirectional Encoder Representations from Transformers) for understanding medical context. Here's how it works:



### Clinical Text Generation

The ClinicalTextGenerator converts each patient's 10 categorical values into a coherent medical narrative. This is not simple concatenation - it uses domain-specific templates to generate clinically meaningful sentences.

Example transformation:

*Input: BirthWeight=LowWeight, HeartRate=RapidHeartRate, SkinTinge=Bluish*

*Output: "The newborn has a low birth weight, with a significant family history of cardiac conditions. The infant is presenting with a rapid heart rate indicating tachycardia, with cyanotic skin coloration suggesting poor oxygenation..."*

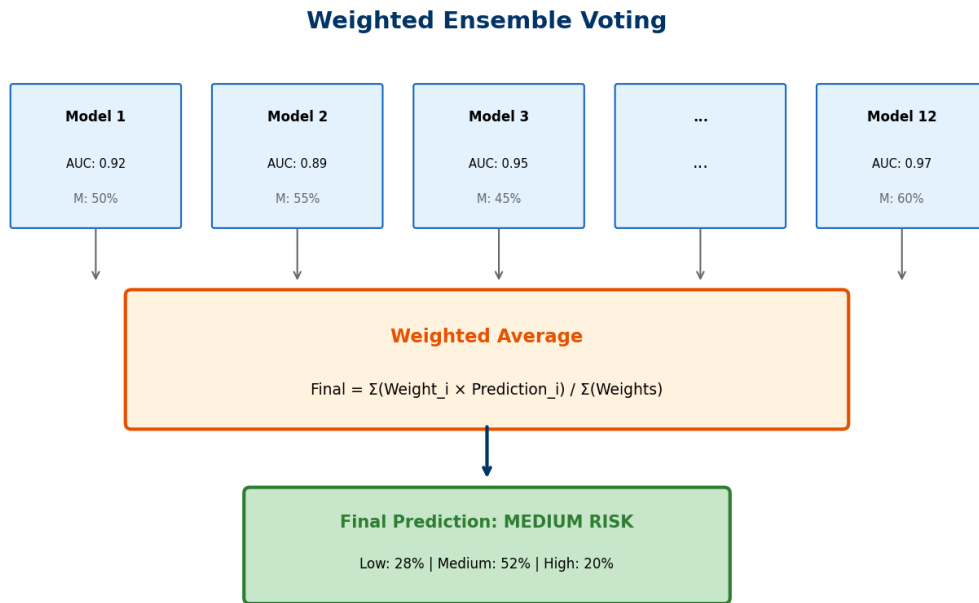
### BioBERT Embeddings

BioBERT is a domain-specific BERT model pre-trained on large-scale biomedical corpora including PubMed abstracts and PMC full-text articles. It understands that 'tachycardia' and 'rapid heart rate' are the same concept, and that 'cyanotic' indicates a serious oxygen deprivation concern.

- \* Model: dmis-lab/biobert-v1.1
- \* Output: 768-dimensional embedding per text
- \* Token limit: 128 tokens per text

## 7. Ensemble Weighted Voting Mechanism

The ensemble combines predictions from all 12 models using a weighted soft voting scheme. This is more sophisticated than simple majority voting:



### How Weights Are Determined

Each model's weight is proportional to its AUC score on the validation set. Models that performed better during training get more influence in the final prediction. This adaptive weighting ensures that the ensemble benefits more from its stronger members while still considering diverse perspectives.

### Mathematical Formulation

For each class  $c$  (Low, Medium, High), the ensemble probability is computed as:

$$P(\text{class}=c) = \frac{\sum(\text{weight}_i * P_i(\text{class}=c))}{\sum(\text{weight}_i)}$$

Where  $\text{weight}_i$  is the AUC of model  $i$ , and  $P_i$  is model  $i$ 's predicted probability.

## 8. Training Configuration & Hyperparameters

Parameter	Value	Description
Random Seed	42	For reproducibility
Train/Val/Test Split	70/15/15%	Stratified by class
Batch Size	64	Samples per gradient update
Epochs	200	Maximum training iterations
Early Stopping	20 epochs	Patience before stopping
LR Reduction	Factor 0.5	After 10 plateau epochs
Optimizer	Adam	Adaptive learning rate
Loss Function	Sparse Categorical CE	Multi-class classification

### Hyperparameter Optimization

The system supports extensive HPO using Optuna and Ray Tune backends. In combined mode, it explores 10,000+ configurations to find optimal hyperparameters for each model architecture.

- \* Optuna: Tree-structured Parzen Estimator (TPE) sampler
- \* Ray Tune: ASHA scheduler for early trial pruning
- \* Search space includes: learning rate, dropout, layer widths, regularization

## 9. Installation & Usage Guide

---

### Prerequisites

- \* Python 3.8 or higher
- \* CUDA-capable GPU (recommended for BERT extraction)
- \* At least 16GB RAM

### Installation

```
# Clone repository
git clone <repository-url>
cd cardiacarrestnewborn

# Create virtual environment
python -m venv venv
venv\Scripts\activate # Windows

# Install dependencies
pip install -r requirements.txt
```

### Running the Pipeline

Mode	Command	Description
Full	python main.py --mode full	Complete pipeline
BERT only	python main.py --mode bert	Extract embeddings
Ensemble only	python main.py --mode ensemble	Train 12 models
HPO only	python main.py --mode hpo	Hyperparameter search
Evaluate	python main.py --mode evaluate	Test saved models

## 10. Expected Performance Metrics

Metric	Expected Range	Description
Accuracy	92-95%	Overall correct predictions
AUC (macro)	0.94-0.97	Area under ROC curve
Precision	0.90-0.95	True positive rate
Recall	0.88-0.93	Sensitivity
F1-Score	0.89-0.94	Harmonic mean of P and R

### Performance Comparison

The ensemble consistently outperforms individual models due to the diversity of its components. The weighted voting mechanism typically improves AUC by 2-5% compared to the best individual model.

## 11. Technology Stack

Category	Technologies
Deep Learning	TensorFlow 2.12+, Keras
NLP/BERT	HuggingFace Transformers, PyTo
ML Utilities	scikit-learn, imbalanced-learn
Hyperparameter Tuning	Optuna 3.2+, Ray Tune 2.5+
Data Processing	Pandas, NumPy
Visualization	Matplotlib, Seaborn
GUI Application	PyQt5

## 12. Future Improvements

---

- \* Real-time monitoring integration with hospital systems
- \* Continuous learning from new patient data
- \* Explainability features using SHAP/LIME
- \* Mobile application for bedside predictions
- \* Integration with electronic health records (EHR)
- \* Multi-center validation studies

## 13. References

---

1. Gupta, K., et al. - "A Machine Learning Approach Using Statistical Models for Early Detection of Cardiac Arrest in Newborn Babies" - IEEE Access
2. Choi, E., et al. - "Using recurrent neural network models for early detection of heart failure onset" - J. Amer. Med. Inform. Assoc.
3. Rajkomar, A., et al. - "Scalable and accurate deep learning with electronic health records" - NPJ Digital Medicine
4. Lee, J., et al. - "BioBERT: a pre-trained biomedical language representation model" - Bioinformatics
5. Devlin, J., et al. - "BERT: Pre-training of Deep Bidirectional Transformers" - NAACL 2019

**DISCLAIMER: This tool is intended for research and educational purposes only. Clinical decisions should always be made by qualified healthcare professionals. This system is not a substitute for professional medical advice, diagnosis, or treatment.**