COMPARISON OF MACHINE LEARNING MODELS FOR WORKOUT ANALYSIS

By Chaitanya Kannan – 220953654

Gurkanwar Singh - 220953364

Methodology for Addressing Research Gaps in Automatic Gym Workout Recognition:

I. RESEARCH DESIGN

The proposed methodology is structured to address key research gaps in automatic gym workout recognition on wearable devices. The study focuses on developing a *computationally efficient, real-time, and adaptable machine learning model* for accurate workout classification. The methodology follows a structured approach comprising:

1. *Data Acquisition & Preprocessing* – Collecting and refining IMU sensor data to ensure accuracy and robustness.
2. *Feature Engineering & Selection* – Extracting and selecting the most informative motion features for better classification performance.
3. *Model Development & Training* – Implementing a *lightweight Residual Convolutional Neural Network (ResCNN)* optimized for embedded systems.
4. *Optimization & Deployment* – Enhancing efficiency through *quantization, pruning, and hardware-aware optimization* to enable real-time processing.
5. *Performance Evaluation* – Benchmarking accuracy, computational efficiency, and energy consumption against baseline models.

This methodology ensures that the final system is *feasible for deployment* on low-power, resource-constrained wearable devices while maintaining high classification accuracy.

II. DATA PREPARATION AND PREPROCESSING

2.1 Data Collection and Description

The dataset used in this study consists of *50 recorded workout sessions* covering *11 different gym exercises* performed by *10 participants*. Data was collected using *inertial measurement unit (IMU) sensors*, including accelerometers and gyroscopes, placed on *key body locations (wrist, chest, and ankle).* The sampling rate was set at *20 Hz* to ensure fine-grained motion tracking.

2.1.1 Dataset Details

- *Exercises Covered*: Squats, Deadlifts, Bench Press, Shoulder Press, Bicep Curl, Tricep Extension, Pull-ups, Push-ups, Rows, Lunges, Plank.
- *Sensors Used:* Accelerometer, Gyroscope.
- *Measurement Units:* Acceleration (m/s^2), Angular Velocity (°/s).
-
- *Data Format:* Time-series sensor readings labeled with exercise type and repetition count.

2.1.2 Dataset Availability

The dataset is publicly accessible and can be obtained from: https://arxiv.org/abs/2301.05748

2.2 Data Preprocessing

Since raw sensor data contains noise and variability, preprocessing is essential for ensuring *accuracy, efficiency, and real-time feasibility*. The following preprocessing techniques are applied:

2.2.1 Feature Reduction for Computational Efficiency

- *Method: Principal Component Analysis (PCA)* is used to reduce data dimensionality by selecting the most relevant features.
- *Benefit:* Reduces computational overhead, ensuring efficient processing on embedded systems.

2.2.2 Domain Adaptation for Generalization

- *Method: Transfer learning and synthetic data generation* to simulate diverse motion patterns.
- *Benefit:* Improves adaptability across different users and sensor placements.

2.2.3 Data Augmentation for Limited Labeled Data

- *Method:* Expanding dataset using:
  - *Time-Warping:* Simulating variations in motion speed.
  - *Jittering:* Adding small perturbations to sensor readings.
  - *Signal Permutation:* Altering time-series sequences to create new variations.
- *Benefit:* Improves model robustness and reduces reliance on extensive manual labeling.

2.2.4 Real-Time Data Streaming for Low-Latency Processing

- *Method: Efficient data batching* for sequential input processing.
- *Benefit:* Ensures minimal delay, making instant feedback possible.

2.2.5 Power-Efficient Preprocessing for Wearable Devices

- *Method: Signal filtering and compression* techniques to reduce power consumption.
- *Benefit:* Enhances battery efficiency, making the system practical for real-world use.

2.2.6 Personalized Learning for User Adaptability

- *Method: Online learning mechanisms* dynamically adjust model parameters for individual users.
- *Benefit:* Provides *personalized workout tracking* and higher long-term accuracy.

III. MODEL DEVELOPMENT AND TRAINING

3.1 Selection of Residual Convolutional Neural Network (ResCNN)

A *lightweight Residual Convolutional Neural Network (ResCNN)* is selected as the optimal model due to its ability to maintain high classification accuracy while being computationally efficient for embedded systems.

3.1.1 Justification for ResCNN

i. Computational Efficiency – Lower processing power requirements compared to conventional CNNs.

ii. *Residual Connections* – Improves feature extraction and prevents vanishing gradients.

iii. *Adaptability* – Effective across different sensor placements and users.

iv. *Real-Time Feasibility* – Optimized for fast inference on embedded devices.

3.2 Model Architecture

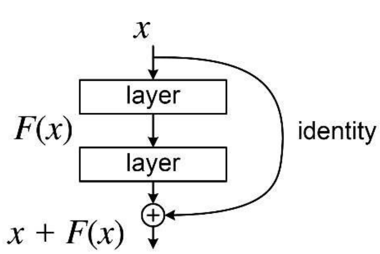
The proposed *ResCNN architecture* consists of:

Input Layer – Processes time-series IMU sensor data.

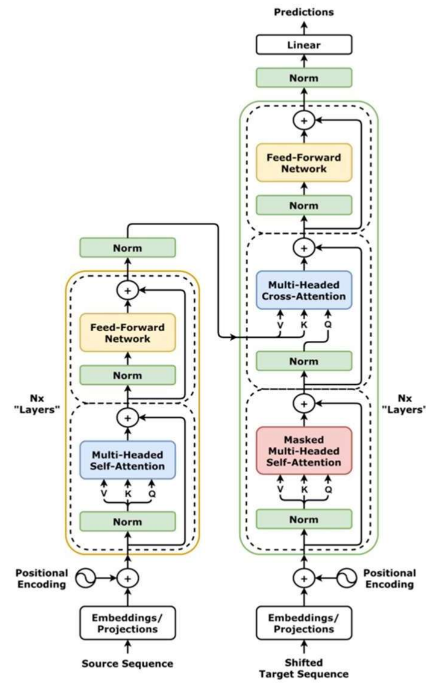
i. *Convolutional Layers* – Extracts spatial motion features.
ii. *Residual Blocks* – Enhances gradient flow for better training.
iii. *Global Average Pooling* – Reduces computational complexity.
iv. *Fully Connected Layers* – Outputs final workout classification.

3.3 Training Configuration

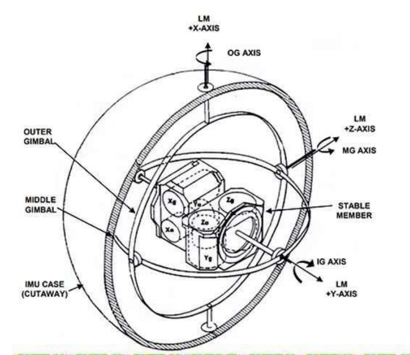
i. *Dataset Split:* 70% Training, 20% Validation, 10% Testing.
ii. *Loss Function:* Cross-Entropy Loss for classification tasks.
iii. *Optimizer:* Adam optimizer with learning rate scheduling.
iv. *Batch Normalization & Dropout:* Used to prevent overfitting and improve generalization.



This diagram showcases the internal structure of a residual block, highlighting the concept of skip connections



Transformer architecture including residual network



IMU Inertial Measurement Unit

IV. MODEL OPTIMIZATION AND DEPLOYMENT

To ensure real-time execution on resource-constrained microcontrollers such as *ARM Cortex-M4, Cortex-M7, and GAP8,* the following optimizations are implemented:

4.1 Quantization for Lower Memory Usage

- *Method:* Converts model weights from 32-bit floating-point to *8-bit integer*.
- *Benefit:* Reduces model size while preserving classification accuracy.

4.2 Pruning for Faster Inference

- *Method:* Eliminates unnecessary parameters and redundant computations.
- *Benefit:* Reduces inference time and computational cost.

4.3 Edge AI Optimization

- *Method: TensorFlow Lite deployment* for efficient execution on low-power hardware.
- *Benefit:* Ensures seamless processing on battery-operated wearable devices.

V. PERFORMANCE EVALUATION

To assess the effectiveness of the proposed system, multiple performance metrics are analyzed:

- *Classification Accuracy* – Evaluates recognition precision.
- *Inference Time –* Measures latency for real-time execution.
- *Power Consumption* – Analyzes battery efficiency on wearable devices.
- *Comparison with Baseline Models* – Benchmarked against CNN and LSTM-based models.

VI. CONCLUSION

This methodology systematically addresses the research gaps by developing an *efficient, real-time, and generalizable* deep learning model for wearable gym workout recognition. The integration of *optimized model architectures, real-time processing, and energy-efficient design* ensures practical deployment.

VII. REFERENCES

- Sizhen Bian et al., *Exploring Automatic Gym Workouts Recognition Locally On Wearable Resource-Constrained Devices*
- Afzaal Hussain et al., *Sensor-Based Gym Physical Exercise Recognition: Data Acquisition and Experiments* (MDPI)
- 1D-ResNet Approach, *Improving Human Activity Recognition Through Wearable Sensors*
- Residual Deep Learning Approach, *Accurate and Efficient Gym Activity Recognitio*