ZeroLag RPS: Real-Time Anticipation of Hand Gestures via Temporal Micromovement Analysis

Final Project Report

Group: Computer Visionaries
Parsa Bakhtiari, Gaia Iori, Ilia Koldyshev*

Abstract

Real-time gesture recognition systems often suffer from latency due to the classification of completed motions. This report presents ZeroLag RPS, an AI agent designed to defeat a human opponent in Rock-Paper-Scissors by predicting moves before gesture completion. We developed a hybrid architecture combining a ResNet-18 spatial encoder with a Temporal Convolutional Network (TCN) head, trained on a custom dataset of gesture sequences. The TCN's dilated causal convolutions efficiently model temporal dependencies within a 64-frame window, enabling the detection of subtle "wind-up" micromovements.

The pipeline integrates MediaPipe for skeletal tracking and utilizes a sigmoid time-weighted loss function to address early-phase ambiguity. Our model achieves a test set accuracy of 70.6% and demonstrates the ability to infer the user's move as early as frame 48 (approx. 200ms before completion). Deployed in a live environment with a rolling buffer, the system delivers zero-latency counter-moves for a seamless user experience.

1. Introduction

The core mission objective of this project was to create an Artificial Intelligence agent capable of **reading human intention** during the execution of a Rock-Paper-Scissors (RPS) gesture. Standard recognition approaches, which classify a static, final hand shape, are not applicable to this task. Our goal was to build a system that could **predict the opponent's move before it fully happens**, thus having a nearly 100% win rate.

The task is formulated as a video-based gesture classification problem, but with a critical temporal constraint: **early prediction**.

*The code for this project is available at https://github.com/Coldilger/rock-paper-scissors-project/tree/main

1.1. Importance and Relevance

- **Temporal Dynamics:** A gesture such as forming 'Paper' starts from a closed fist, hence it is fundamental that it is understood as a process, not a static image. The relevant information is the **evolution of the hand movement over time**.
- **Pre-emptive Reaction:** The agent must infer the gesture during the **'wind-up' phase** (early timestamps) to allow time for the counter-move to be executed faster than human perception can register the final gesture. This demonstrates the capacity of neural networks to model **intent** instead of just a state.
- **Spatial Focus:** Effective prediction requires isolating the subject. The input video contains noise (face, background, clothing), hence needing real-time **hand detection, tracking, and cropping** to feed only relevant data to the model.

2. Related Works

The problem of recognizing hand gestures from a video stream is fundamentally a sequence modeling task. The objective is to map a sequence of inputs $X = \{x_1, x_2, \ldots, x_T\}$, where each x_t represents the spatial features of a frame at time t, to a sequence of class probabilities or a single label. In this section, we review three dominant paradigms for sequence modeling: Recurrent Neural Networks, Transformers, and Temporal Convolutional Networks.

2.1. Recurrent Architectures (LSTM and GRU)

For a significant period, Recurrent Neural Networks (RNNs) served as the de facto standard for sequence modeling. Specifically, Long Short-Term Memory (LSTM) networks [2] were engineered to mitigate the vanishing gradient problem inherent in standard RNNs by introducing gating mechanisms (input, output, and forget gates) that control information flow.

Inputs and Outputs: An LSTM processes data sequentially. At each time step t, the cell takes the current input

feature vector x_t and the hidden state from the previous step h_{t-1}. It outputs an updated hidden state h_t, which carries the memory of the sequence up to that point.

Limitations: Despite their success, recurrent architectures suffer from a fundamental bottleneck: lack of parallelism. The calculation of h_t depends explicitly on h_{t-1}, forcing the network to process the sequence step-by-step. This sequential dependency limits training throughput on GPUs and can introduce latency during inference, which is suboptimal for real-time applications like the "ZeroLag" agent.

2.2. Transformers and Self-Attention

Following the publication of "Attention Is All You Need" [4], Transformers supplanted RNNs in Natural Language Processing and have been adapted for vision tasks (Video Vision Transformers). Transformers rely entirely on the self-attention mechanism to draw global dependencies between input and output.

Inputs and Outputs: The input to a Transformer is the entire sequence X embedded into vectors, combined with positional encodings to retain order information. The self-attention layer computes a weighted sum of all values in the sequence based on the compatibility of queries and keys.

Limitations for this Project: While Transformers are highly parallelizable, they have a quadratic computational complexity $O(T^2)$ with respect to sequence length T. Furthermore, they lack the *inductive bias* of locality that convolutions possess. Consequently, Transformers typically require massive datasets to generalize well. Given the custom, relatively small and very task-specific dataset collected for this project, a Transformer would be prone to overfitting compared to CNN-based approaches.

2.3. Temporal Convolutional Networks (TCN)

In response to the limitations of RNNs and the data-hunger of Transformers, [1] demonstrated that a generic convolutional architecture, called the Temporal Convolutional Network (TCN) (Figure 1), could outperform canonical recurrent networks across diverse sequence modeling tasks. TCNs treat time steps as spatial dimensions, applying 1D convolutions across the sequence.

Inputs and Outputs: The TCN takes an input tensor of shape (N, C_{in}, L), where N is batch size, C_{in} is the number of input features (from the spatial encoder), and L is the sequence length. It produces an output of the same length L, mapping features to hidden representations or class probabilities at every time step simultaneously.

The TCN architecture is defined by three primary characteristics:

2.3.1. Causal Convolutions

Unlike standard convolutions which may access future information via padding, TCNs strictly employ causal convolutions. The output at time t is convolved only with elements from time t and earlier. This prevents information leakage from the future to the past, a critical requirement for real-time forecasting where future frames are unavailable.

2.3.2. Dilated Convolutions

To capture long-range dependencies without an explosion in parameters, TCNs utilize dilated (expanded) convolutions. The receptive field is expanded by introducing a fixed step d between every two adjacent filter taps. The dilation factor typically increases exponentially ($d = 2^i$ for layer i) with network depth. This allows the network to effectively "look back" at a history size that grows exponentially with the number of layers, enabling the modeling of the gesture's "wind-up" phase [3].

2.3.3. Residual Connections

To facilitate the training of deep architectures, TCNs incorporate residual blocks. Formally, the output o is defined as $o = \text{Activation}(x + \mathcal{F}(x))$, which allows gradients to flow through the network depth effectively, stabilizing training.

2.4. Comparative Analysis and Architecture Selection

We selected the TCN over LSTMs and Transformers for three distinct reasons relevant to the "Live Battle" requirements:

1. **Parallelism (vs. LSTM):** Unlike RNNs, TCNs perform convolutions across the entire buffer in parallel, maximizing GPU utilization for zero-latency inference.
2. **Data Efficiency (vs. Transformer):** TCNs possess a strong inductive bias towards local temporal patterns. This allows them to learn effective motion dynamics from our custom, smaller-scale dataset without the massive pre-training required by Transformers.
3. **Flexible Memory:** By adjusting the kernel size k and dilation d, we could explicitly engineer the receptive field to cover exactly 64 frames (approx. 2 seconds), ensuring the model sees the entire gesture evolution without wasting computation on irrelevant history.

3. Main Contribution

Our main contribution is the successful design and implementation of a video processing and deep learning pipeline that solves the problem of predicting RPS moves preemptively, which we would later refer to as the **Cognitive Timing** challenge.

- **Temporal Model:** Integration of a **Temporal Convolutional Network (TCN)** to process motion dynamics extracted by a CNN.
- **Optimization Strategy:** Development and application of a **Sigmoid Time-Weighted Loss** function to enforce correct learning by penalizing much less the prediction on

the starting frames and focusing on the critical movement and completion phases.
- **Real-Time Performance:** The system achieves prediction trigger times at **3.0s** during the countdown, designed to exploit human reaction time.

Other contributions are related to solving some of the aforementioned issues, such as what we call **The Rock Paradox**. Visually, the start of every move (Rock, Paper, or Scissors) looks identical to the 'Rock' gesture (a closed fist), which represents a problem that arises with more simple models.

4. Data

The project utilized raw video sessions of RPS gesture performances collected by the team.
- **Source Material:** Raw video sessions featuring a single subject performing isolated one-handed gestures.
- **Dataset Size:** We collected 70 videos for each move, resulting in approximately 5,300 frames per class.
- **Preprocessing/Filtering:** Audio was removed, and frames were extracted at ~ 20 frames per second.
- **Splitting Strategy:** A stratified, video-level split was performed to create distinct Training, Validation, and Test sets, preventing any single video's frames from contaminating multiple sets. The Test Set was treated as a "Black Box" for final, unbiased metrics.

5. Methods

5.1. Data Preprocessing and Augmentation

The data preparation was crucial for isolating the signal:
- **Tracking Evolution:** Our approach initially relied on simple static detection that reused the last bounding box during failures. This evolved into a physics-based tracker using spatial proximity rather than confidence scores to predict motion and filter background objects. We further refined this with a 5-frame "smart drop" limit and larger crop padding to minimize drift. Ultimately, this custom stack was replaced by MediaPipe Hands to achieve superior robustness and real-time performance.
- **Hand Detection/Tracking: MediaPipe Hands** was used to locate the skeletal landmarks of the hand.
- **Cropping and Centering:** The input frames were **tightly cropped and centered** around the detected hand, ensuring the model focused purely on the gesture and not on background noise or large-scale movements.
- **Leakage Prevention: Stratified Video-Level Splitting** was implemented to group files by their original Video ID before partitioning the dataset (70% Training, 15% Validation, 15% Test), preventing **data leakage** and ensuring unbiased metrics.
- **Augmentation:** To improve robustness, the training data underwent augmentation, including **random rotations, horizontal flips, and color jitter**.

5.2. Model Configuration and Training

The final model used a **ResNet-18 Encoder** and a **TCN Decoder**.
- **Sequence Length:** $SEQ\_LEN = 64$ frames were used as the rolling buffer for the TCN.
- **TCN Hyperparameters:** Optimized parameters included a **Kernel Size** of $K = 6$ and 4 layers of channels.
- **Sigmoid Time-Weighted Loss (Optimization):** Instead of training only on the final frame's output, the TCN was trained to predict at every timestep. The loss incorporates a temporal sigmoid weighting:

$$\mathcal{L}_{weighted} = \sum_{t=1}^{T} w(t) \cdot \mathcal{L}_{CE}(y_t, \hat{y}_t)$$

The weighting function $w(t)$ is defined as:

$$w(t) = \frac{1}{1 + e^{-\alpha(t-t_0)}}$$

Here, $\mathcal{L}_{CE}(y_t, \hat{y}_t)$ denotes the **Cross-Entropy loss** between the ground truth y_t and the predicted probability distribution $\hat{y}_t$ at time t. The parameter t_0 marks the transition midpoint where the action becomes less ambiguous, and α controls the steepness of the weighting curve. This mechanism allows the model to **downweight the ambiguous start** (mitigating the Rock Paradox) while heavily penalizing errors in the later, definitive frames.

6. Experiments

6.1. Quantitative Results

After optimization, the model achieved the following performance on the "Black Box" test set:
- **Final Test Accuracy (at frame 64):** $\mathbf{70.59\%}$.
- **Validation Peak Accuracy:** Up to 85.19% during training.

6.2. Early Prediction Analysis

The key metric was **Accuracy per frame timestep**. The results confirmed the hypothesis that the model's confidence increases over time(Figure 2).
- **Confidence Rise:** The accuracy line begins a steep rise around **Frame 20-25** , indicating the TCN is successfully capturing the motion signal and moving beyond the initial ambiguity (the Rock Paradox).
- **Cognitive Timing:** The prediction logic successfully triggered the counter-move at $t = 3.0$ seconds, guaranteeing a time advantage over the human player, whose move is usually finalized around $t = 3.2$ seconds.

6.3. Per-Class Performance

Analysis of the individual class performance revealed strong differentiation capabilities, particularly for moves requiring finger articulation:

- **Scissors:** Showed the highest and fastest convergence to high accuracy, reaching $\sim 90\%$ accuracy before Frame 52 (Figure 3).
- **Rock:** As the model expects the fist to evolve into a different gesture (Paper or Scissors), for rock we have lower and more fluctuating early accuracy, reflecting the Rock Paradox and the difficulty in modelling this situation as accurately as for the other moves (Figure 4).
- **Paper:** Contrary to scissors, but similarly to rock, the paper (open hand) is not an innatural shape for the hands to be, hence the difficulty in accurately modelling this move (Figure 5).

7. Conclusion

We successfully implemented a sophisticated video classification pipeline for real-time gesture prediction. By combining a **ResNet Encoder** for spatial features and a **TCN Decoder** for temporal analysis, coupled with the innovative **Sigmoid Time-Weighted Loss** function, we built a system that actively anticipates human intention. The system achieved the "Cognitive Timing" goal by triggering the counter-move before the opponent's gesture was fully executed.

Future work will focus on expanding the dataset for greater diversity and exploring **Adaptive Difficulty** where the model learns specific opponent patterns over multiple rounds.

References

[1] Shaojie Bai, J Zico Kolter, and Vladlen Koltun. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. *arXiv preprint arXiv:1803.01271*, 2018. 2

[2] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997. 1

[3] Aaron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu. Wavenet: A generative model for raw audio. *arXiv preprint arXiv:1609.03499*, 2016. 2

[4] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, 2017. 2

[5] Shuai Zuo, Zhongqi Song, Kexin Zhang, and Yuanliang Zhang. Strong robustness and high accuracy in predicting remaining useful life of supercapacitors. *APL Energy*, 1(1): 016105, 2022. 4

8. Appendix

A. TCN Schema

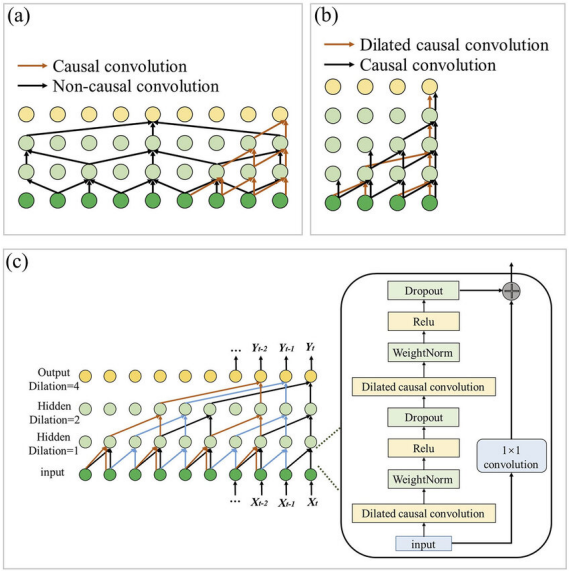

Figure 1. TCN model - schema, picture obtained from related work [5].

B. Early Prediction Accuracy

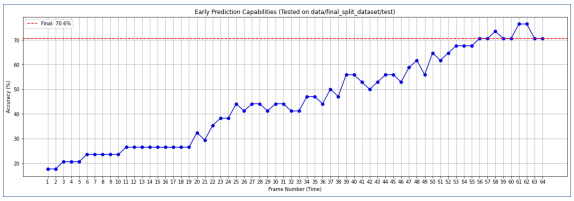

Figure 2. Early Prediction Accuracy

C. Prediction Accuracy per Gesture

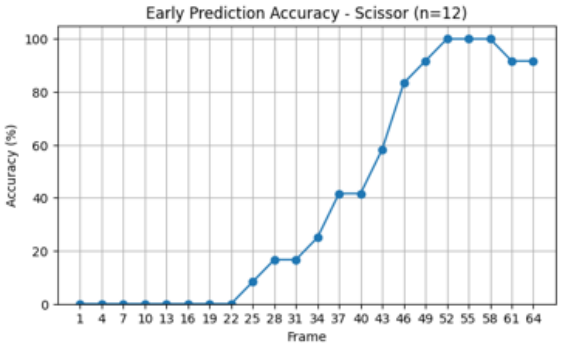

Figure 3. Gesture Prediction Accuracy - Scissors (n=12).

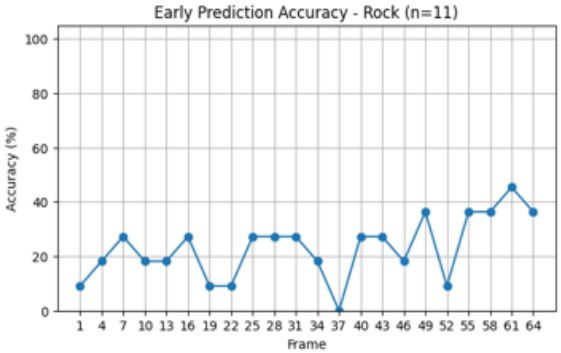

Figure 4. Gesture Prediction Accuracy - Rock (n=11).

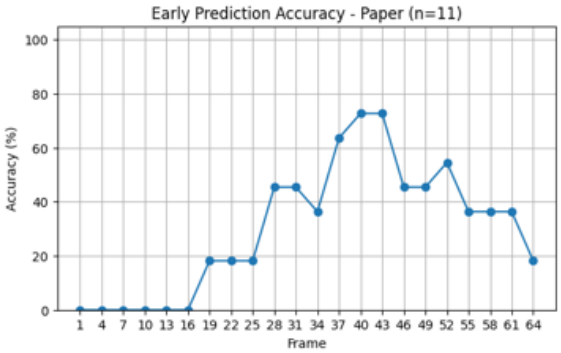

Figure 5. Gesture Prediction Accuracy - Paper (n=11).