RGB-Only Real-Time Gesture Recognition for Screen Control via Temporal Convolutional Networks

Myles Robinson, Nick Chermak, Abdulrahman Aleid, Inseung Kang
Carnegie Mellon University
Pittsburgh, PA, USA
{mylesr, nchermak, aaleid, inseung}@andrew.cmu.edu

***Abstract*—Hand gesture recognition (HGR) has a wide range of applications, from gaming and AR/VR to human behavior analysis. Real-time gesture recognition has grown significantly as a field, driven by advances in deep learning and on-device machine learning. While the usefulness of existing gesture recognition models is evident, their effectiveness—particularly in practical, real-time scenarios—is what we aim to challenge in this study. Many current models are constrained by specific environments or require advanced imaging modalities to achieve high performance.**

In response, we propose a real-time gesture recognition system that uses only RGB imaging and generalizes well across individuals and environments by leveraging few-shot learning. To evaluate our approach, we conducted experiments on both the IPN Hand dataset and a custom gesture dataset designed for screen control tasks. Our model achieved 88.84% accuracy on IPN and 98.85% on our noisy dataset, outperforming the IPN benchmark in isolated gesture recognition. By showing that an RGB-only, lightweight model can deliver high accuracy with low latency, our work contributes a practical solution for real-time HGR in everyday human-computer interaction and resource-constrained environments.

***Index Terms*—Hand gesture recognition, real-time classification, computer vision, few-shot learning, temporal convolutional network, RGB imaging, human-computer interaction, landmark detection, gesture-based control, deep learning, MediaPipe**

I. INTRODUCTION

Computer vision techniques have become indispensable to robotics, automation, and human–computer interaction (HCI). Among the most intuitive interfaces is hand gesture recognition (HGR), which lets users issue commands without touching a device, supporting applications that range from virtual/augmented reality and smart-home control to assistive technologies [1], [2].

Capturing dynamic gestures—continuous hand motions that unfold over time—enables richer, more natural interaction than static pose recognition. Yet many state-of-the-art (SOTA) systems still depend on depth cameras, infrared imaging, optical flow, or confined capture ranges, and their accuracy often deteriorates when hands are partly occluded or moving unpredictably [2], [3]. These extra sensors raise hardware costs, while heavy 3D convolutional or attention-based architectures can exceed the latency budgets of edge or mobile devices. A practical real-time solution must therefore operate on RGB-only input, remain robust under everyday lighting and clutter, and run efficiently on commodity hardware.

Early HGR research relied on hand-crafted pipelines: Starner and Pentland segmented skin-colored regions and used Hidden Markov Models to recognize American Sign Language letters [4]; Davis and Shah accumulated motion-history images and classified them with Hu moments and an SVM [5]; Wang *et al.* extracted geometric fingertip descriptors for HMM classification [6]. Although these systems achieved promising accuracies in controlled settings, they were brittle to illumination variation, camera viewpoint changes, and background clutter, requiring painstaking parameter tuning.

Deep learning removed the need for manual feature design. Molchanov *et al.*'s 3D CNN processed 16-frame clips to detect gestures online, but incurred high compute on embedded GPUs [7]. Two-stream architectures fused RGB appearance with optical-flow motion to boost accuracy [8], trading speed for a costly flow-estimation step. More recently, transformer variants such as MVTN [9] and GestFormer [10] captured long-range dependencies via self-attention and set new RGB-only benchmarks, yet their tens of millions of parameters and quadratic attention hinder real-time deployment.

Another line of work represents motions with skeletal key points obtained from pose estimators; ST-GCN built spatiotemporal graphs over 2D/3D joints to exceed 85% accuracy on hand pose datasets [11]. However, performance collapses when key point detection (e.g., OpenPose [12]) fails under occlusion, and many top scores rely on multimodal fusion (RGB + depth + IR) that demands specialized sensors [9]. Collectively, the literature leaves a gap: accurate, real-time, RGB-only gesture recognition that scales to unconstrained environments.

To bridge this gap, we develop a lightweight pipeline that couples MediaPipe Hands for fast landmark extraction [13] with a compact Slow–Fast CNN designed to balance spatial precision against temporal context. Few-shot learning techniques further improve generalization across users and backgrounds while keeping the network footprint modest.

We evaluate on (i) IPN Hand, a 14-class public benchmark, and (ii) a custom seven-gesture dataset (clean + messy splits) designed for screen-control commands. Metrics include classification accuracy and end-to-end inference latency on

a laptop-class GPU. The proposed model attains 88.84% accuracy with 18.85ms latency on IPN—beating the RGB-only baseline by 5.2pp while matching its speed—and 98.85% on the messy subset of our dataset, demonstrating strong generalization.

Our contributions lie in demonstrating that accurate, real-time gesture recognition is possible using only RGB input, without the need for auxiliary data such as depth or optical flow. We show that a compact Slow–Fast CNN paired with off-the-shelf landmark detection can match or surpass heavier models while maintaining practical inference speeds. Additionally, we present a comprehensive evaluation across public and custom datasets to highlight generalization and real-world usability. Finally, we release the full pipeline—including code, trained weights, and annotated data—to support future research in accessible, deployable gesture-based interfaces.

II. DATA COLLECTION & DESCRIPTION

A. Benchmark Dataset

For our benchmark dataset, we use the IPN Hand Dataset [2], a widely used resource for real-time continuous hand gesture recognition. It contains a total of 4,218 labeled gesture instances across more than 800,000 RGB video frames collected from 50 different subjects. The dataset consists of 13 dynamic and static gesture classes, in addition to a non-gesture (null) class with 1,431 instances. Videos are recorded at 30 frames per second with a resolution of 640×480, providing sufficient temporal and spatial resolution for gesture modeling.

We selected the IPN Hand dataset because the recording setup closely resembles our target application environment: a front-facing camera capturing upper-body interactions for screen control. Moreover, its large variety of gestures, subjects, and recording conditions enables robust evaluation of generalization capabilities. In addition to its realistic setting, the dataset's creators provide an open-source gesture recognition model, which we adopt as a performance benchmark to compare the effectiveness of our proposed system.

A summary of the gesture class distribution is provided in Table I.

TABLE I
DISTRIBUTION OF GESTURE CLASSES IN THE IPN HAND DATASET

Class Type	Label(s)	Instances	Mean Duration (frames)
Non-gesture	D0X	1,431	147
Pointing	B0A, B0B	2,017	∼222
Other gestures	G01–G11 (11 classes)	2,201	∼63
Total	All (14 classes)	**5,649**	∼142

B. Collected Dataset

To fine-tune our model for screen control applications, we collected a custom dataset focused on seven specific gestures: left, right, up, down, select, back, and null. Each gesture was recorded for 2 seconds at 30 frames per second by two subjects, yielding a total of 1,400 instances in two separate subsets. The first, referred to as the "clean" dataset, consists of well-executed gestures performed orthogonally to the camera with minimal noise—each gesture was repeated 100 times per subject under controlled conditions. In this clean set, all gestures were performed using the right hand with the wrist held stationary. The gestures were designed using a consistent anatomical logic: left was a lateral swipe to the left, with the hand oriented vertically (thumb up, pinky down), and the motion directed in the direction the open palm was facing (i.e., away from the subject). Conversely, right was a lateral swipe to the right in the direction the back of the hand was facing. For up and down, the hand rotated about the wrist in a vertical arc—up involved swiping upward with the palm facing up, and down involved swiping downward with the palm facing down. The select gesture was defined as touching the thumb to the middle finger, while back involved forming a closed fist.

The second subset, the "messy" dataset, introduced greater variability: gestures were performed more casually, at varying angles, occasionally moving in and out of frame, and not always orthogonal to the camera. Like the clean set, it includes 1,400 total instances. A detailed breakdown of our custom dataset is provided in Table II

TABLE II
DISTRIBUTION OF CUSTOM COLLECTED DATASET

Class Type	# Labels	Subset	Instances	Mean Duration (frames)
Gestures	6	Clean	1,200	60
Null	1	Clean	200	60
Gestures	6	Messy	1,200	60
Null	1	Messy	200	60
Total	7	–	**2,800**	60

In both subsets, the distance between the subject and the camera was intentionally varied to simulate realistic usage scenarios. The null class encompassed a range of natural, non-gesture hand behaviors such as holding a phone, scratching the face, or resting the hand passively in view—capturing common activity that a robust gesture recognition model must ignore. This dual-dataset strategy was intended to promote generalization and enable effective real-time screen control in both ideal and uncontrolled environments.

III. METHODS

A. Benchmark Model Architecture

To evaluate the performance of our proposed system, we compare it against the benchmark model provided with the IPN Hand dataset. This benchmark follows a two-stage architecture: a gesture detector based on ResNet-50, followed by a ResNeXt-101 classifier. Both components operate over a sliding window of video frames, allowing the system to capture temporal context during inference. The architecture of the benchmark IPN Hand model is illustrated in Fig. 1. The authors of the IPN Hand dataset experimented with various input modalities, including raw RGB frames, RGB combined with optical flow, and RGB with image segmentation [2].

Because our target use case is real-time screen-control, the entire pipeline must stay below **20 ms** end-to-end latency on commodity hardware; this requirement drives every architectural and training choice described below.

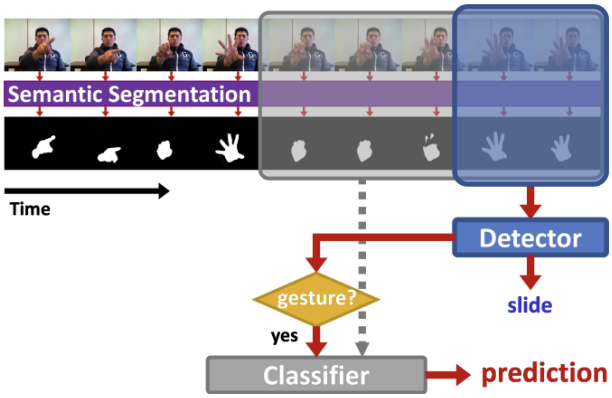

Fig. 1. Model pipeline of the IPN Hand benchmark, consisting of a ResNet-50 detector followed by a ResNeXt-101 classifier operating over a sliding windows. Adapted from [2].

B. Our Model Architecture

Using the pre-trained Mediapipe Hands model as a feature extractor, we first extract the hand keypoints from each frame. Those keypoints then feed our Slow-Fast CNN network. The slow and fast pathways run in parallel, taking in frame keypoints at different sample rates according to an alpha parameter. Both our slow and fast pathways use three convolution blocks in series. Each convolution block has the same structure: a 1D convolution followed by batch norm, relu activation, max pooling, and dropout layers, respectively. The output channels for the slow convolution blocks are 256, 512, and 768, respectively (the fast pathway's convolution block output channels are determined by a beta parameter). The outputs of the slow and fast pathways then feed global pool layers before being concatenated and fed into a two-layer MLP with output sizes 128 and num_classes, respectively. For a visualization of our model architecture, refer to Figure 2. For information on our hyper-parameters, reference Table III.

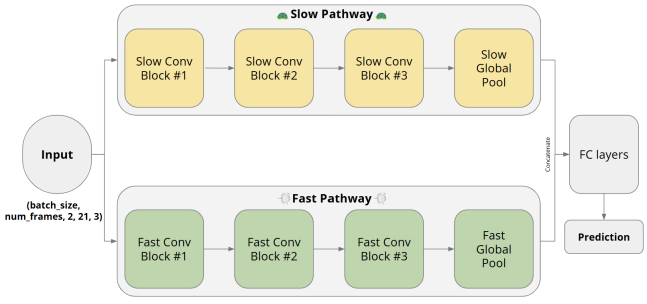

Fig. 2. Architecture of our proposed model, showing the slow and fast pathways with temporal convolution blocks.

C. Training

To validate our model's performance and compare to the IPN Hand model results, the Slow-Fast CNN was first trained on the IPN Hand dataset. We trained on all 5,649 gesture instances just as they did. We then fine-tuned this model on our collected dataset of clean gestures comprised of 1,400 gesture instances. In doing so, we had a two-stage training process where we first re-trained the classifier before fine-tuning the entire model (Slow-Fast CNN pathways in addition to the classifier). In all training loops, the learning rate was reduced upon plateau with a patience of 2. Initial learning rates varied depending on whether the model was being trained from scratch or fine-tuned, but learning rates never exceeded 0.0015.

TABLE III
MODEL HYPERPARAMETERS FOR SLOWFAST PATHWAY

Model Parameter	Description	Optimal Value(s)
alpha	ratio between slow and fast pathway sample rates	8
beta	ratio between slow and fast pathway conv channels	1/8
slow dropout	dropout rate for slow path	0.2
fast dropout	dropout rate for fast pathway	0.2
slow kernel size	kernel size for slow pathway	3
fast kernel size	kernel size for fast pathway	5
slow out channels	list of output channels for slow pathway	[128, 256, 512]

IV. EVALUATION & RESULTS

To evaluate our model's performance, we adopted a two-stage approach. First, we tested our model on the IPN Hand benchmark dataset, allowing for a direct comparison with the baseline models provided by the dataset authors. This comparison was conducted under standardized conditions, focusing specifically on RGB input to align with the constraints of our proposed system.

Table IV summarizes the results across different input configurations. The IPN Hand benchmark model achieves 86.32% accuracy when using RGB combined with optical flow, but this comes at a significant cost in latency (50.8 ms). When limited to RGB input alone, its accuracy drops to 83.59% with improved latency of 18.2 ms. In contrast, our RGB-only model achieves the highest accuracy of 88.84% while maintaining a comparable latency of 18.85 ms, demonstrating that high performance can be achieved without relying on multi-modal input.

TABLE IV
ACCURACY AND INFERENCE LATENCY COMPARISON ON IPN HAND DATASET

Model (Input Type)	Accuracy (%)	Latency (ms)
IPN Hand (RGB + Flow)	86.32	50.8
IPN Hand (RGB only)	83.59	18.2
Ours (RGB only)	**88.84**	**18.85**

Next, to adapt the model for our intended screen control use case, we fine-tuned it on our custom-collected dataset consisting of seven specific gestures. After fine-tuning on our custom dataset, the model achieved an accuracy of 98.85% on the "messy" subset. This result reflects performance on data containing variability in gesture execution, hand positioning, and distance from the camera. The corresponding confusion matrix is shown in Figure 3.

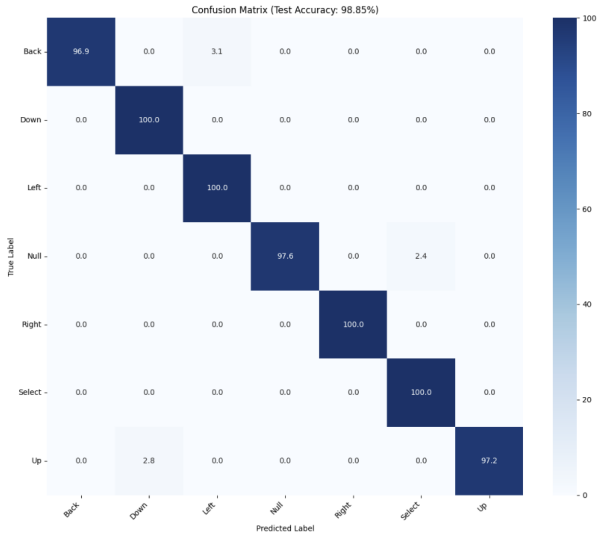

Fig. 3. Confusion matrix showing classification performance on the messy dataset after fine-tuning.

V. DISCUSSION & CONCLUSION

Our RGB-only pipeline substantially narrows the gap to multimodal or flow-based systems while adhering to real-time constraints. On the IPN Hand benchmark [3], we achieve 88.84% accuracy at 18.85 ms end-to-end latency. By comparison, the RGB+flow baseline reports 86.32% at 50.8 ms, and the RGB-only baseline reaches 83.59% at 27.7 ms. On our custom seven-gesture screen-control dataset, the model attains 98.85% accuracy even under "messy" conditions.

Several factors likely contribute to this performance. First, MediaPipe Hands supplies stable landmarks under moderate occlusion, allowing the Slow–Fast CNN to focus on temporal dynamics rather than raw-pixel variability. Second, the dual-path design balances short-term motion cues with longer temporal context, a property traditionally obtained with optical flow [8]. Finally, few-shot fine-tuning compensates for limited training data by adapting feature scales per subject.

Despite these gains, four key limitations remain:

- **Handedness coverage:** Data were collected from only two right-handed users. To generalize to left-handed interactions, the dataset must be horizontally flipped and relabeled to capture mirrored motions.
- **Null-class imbalance:** Idle frames (no gesture) are underrepresented, increasing false positives during inactivity. In practice, it is preferable to miss a gesture than to misclassify background as a command (e.g., inadvertently closing an application).
- **Landmark dependency:** Severe occlusion or poor lighting can cause MediaPipe key-point dropout, propagating errors into the classifier.
- **Hardware scope:** All experiments ran on a laptop-class GPU; performance and energy consumption on mobile CPUs remain to be evaluated.

Future Work

To address these shortcomings, we plan to:

1) Expand the dataset to include left-handed users, diverse demographics, and richer null-action samples.
2) Enhance generalization via stronger data augmentation, domain adaptation, and few-shot learning across devices.
3) Optimize the model for embedded deployment through pruning, quantization, and TensorRT acceleration.
4) Develop online adaptation mechanisms to adjust decision thresholds dynamically and reduce false activations in continuous use.

Conclusion

We demonstrate that a compact Slow–Fast CNN, when paired with MediaPipe Hands landmark extraction, can match or exceed the accuracy of more complex RGB+flow and transformer-based models while maintaining sub-20 ms latency on commodity hardware. By open-sourcing our code, trained weights, and annotated datasets, we provide a practical foundation for future research into accessible, low-cost, real-time gesture interfaces.

ACKNOWLEDGMENT

We would like to sincerely thank Professor Inseung Kang for his valuable guidance, feedback, and support throughout the semester. His insights were instrumental in shaping the direction and rigor of our project.

REFERENCES

[1] M. Al-Hammadi, G. Muhammad, W. Abdul, M. Alsulaiman, M. A. Bencherif, and T. S. Alrayes, "Deep learning-based approach for sign language gesture recognition with efficient hand-gesture representation," *IEEE Access*, vol. 8, pp. 192527–192542, 2020.

[2] M. Oudah, A. Al-Naji, and J. Chahl, "Hand gesture recognition based on computer vision: A review of techniques," *Journal of Imaging*, vol. 6, no. 8, p. 73, 2020.

[3] G. Benitez-Garcia, J. Olivares-Mercado, G. Sanchez-Perez, and K. Yanai, "IPN Hand: A video dataset and benchmark for real-time continuous hand-gesture recognition," in *Proc. 25th Int. Conf. Pattern Recognit. (ICPR)*, 2020, pp. 4340–4347.

[4] T. Starner and A. Pentland, "Real-time American Sign Language recognition from video using hidden Markov models," in *Proc. IEEE Int. Conf. Automatic Face and Gesture Recognition (FG)*, 1998.

[5] J. Davis and M. Shah, "Visual gesture recognition," *IEE Proc. Vis. Image Signal Process.*, vol. 150, no. 2, pp. 87–96, 2003.

[6] J. Wang, X. Yuan, and Y. Zheng, "Hand-gesture recognition based on contour-point features and support vector machines," in *Proc. Int. Conf. Image Processing (ICIP)*, 2011.

[7] P. Molchanov, S. Gupta, K. Kim, and J. Kautz, "Online detection and classification of dynamic hand gestures with recurrent 3-D convolutional neural networks," in *Proc. IEEE/CVF Conf. Computer Vision and Pattern Recognition (CVPR)*, 2016.

[8] K. Simonyan and A. Zisserman, "Two-stream convolutional networks for action recognition in videos," in *Adv. Neural Inf. Process. Syst.*, 2014.
[9] X. Li, A. Hamdi, S. Giancola, and B. Ghanem, "MVTN: Multimodal vision transformer networks for robust gesture recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, 2023.
[10] Y. Chen, M. Garg, D. Ghosh, and P. M. Pradhan, "GestFormer: Multiscale wavelet pooling transformer network for dynamic hand gesture recognition," *arXiv:2405.11180*, 2024.
[11] S. Yan, Y. Xiong, and D. Lin, "Spatial temporal graph convolutional networks for skeleton-based action recognition," in *Proc. AAAI Conf. Artificial Intelligence (AAAI)*, 2018.
[12] Z. Cao, G. Hidalgo Martinez, T. Simon, S. Wei, and Y. Sheikh, "OpenPose: Realtime multi-person 2-D pose estimation using part affinity fields," *IEEE Trans. Pattern Anal. Mach. Intell.*, 2019.
[13] F. Zhang, V. Bazarevsky, A. Vakunov, A. Tkachenka, G. Sung, C.-L. Chang, and M. Grundmann, "MediaPipe Hands: On-device real-time hand tracking," 2020. [Online]. Available: https://google.github.io/mediapipe/solutions/hands