THESIS

INTENTIONAL MICROGESTURE RECOGNITION FOR EXTENDED HUMAN-COMPUTER INTERACTION

Submitted by

Chirag Kandoi

Department of Computer Science

In partial fulfillment of the requirements

For the Degree of Master of Science

Colorado State University

Fort Collins, Colorado

Summer 2023

Master's Committee:

Advisor: Nathaniel Blanchard

Nikhil Krishnaswamy
Hortensia Soto

ABSTRACT

INTENTIONAL MICROGESTURE RECOGNITION FOR EXTENDED HUMAN-COMPUTER INTERACTION

As extended reality becomes more ubiquitous, people will more frequently interact with computer systems using gestures instead of peripheral devices. However, previous works have shown that using traditional gestures (pointing, swiping, etc.) in mid-air causes fatigue, rendering them largely unsuitable for long-term use. Some of the same researchers have promoted "microgestures"—smaller gestures requiring less gross motion—as a solution, but to date there is no dataset of intentional microgestures available to train computer vision algorithms for use in downstream interactions with computer systems such as agents deployed on XR headsets. As a step toward addressing this challenge, I present a novel video dataset of microgestures, classification results from a variety of ML models showcasing the feasibility (and difficulty) of detecting these fine-grained movements, and discuss the challenges in developing robust recognition of microgestures for human-computer interaction.

ACKNOWLEDGEMENTS

I would like to express my deepest appreciation and gratitude to the following people and organizations who have contributed to the completion of this thesis:

My supervisor, Dr. Nathaniel Blanchard, for their guidance, support, and invaluable insights throughout the research process. Their expertise and mentorship have been instrumental in shaping this work.

The members of my thesis committee, Professor Nikhil Krishnaswamy and Professor Hortensia Soto for their valuable feedback, suggestions, and constructive criticism that helped improve the quality of this thesis.

I am deeply grateful to my co-authors, Changsoo Jung, Sheikh Mannan, Hannah VanderHoeven and, Quincy Meisman. Their expertise, collaboration, and tireless efforts in the research process have been invaluable. Together, we have navigated challenges, brainstormed ideas, and co-authored several publications that have laid the foundation for this thesis.

I am indebted to the participants who took part in this study and generously shared their time, experiences, and perspectives. Their contributions were fundamental to the data collection process and greatly enriched the findings of this research.

I would like to acknowledge Computer Science Department of Colorado State University, for providing the necessary resources and facilities that enabled the successful completion of this thesis.

I extend my heartfelt gratitude to my family, for their unwavering support, encouragement, and understanding throughout this academic journey. Their love and belief in me have been my constant source of motivation.

I would like to express my appreciation to my friends and colleagues who provided valuable assistance, engaged in insightful discussions, and offered encouragement during challenging times. Their presence and camaraderie have made this research endeavor all the more rewarding.

Lastly, I want to acknowledge all the authors, researchers, and scholars whose work has been referenced in this thesis. Their contributions have formed the foundation upon which my research in build, and I am grateful for their valuable insights.

DEDICATION

I would like to dedicate this thesis to my teachers and parents.

TABLE OF CONTENTS

ABSTRACT ii
ACKNOWLEDGEMENTS iii
DEDICATION v
LIST OF TABLES vii
LIST OF FIGURES viii

Chapter 1 Introduction 1

Chapter 2 Related Works 3
2.1 Datasets 3
2.2 Algorithms 4

Chapter 3 Methodology 7
3.1 Data Collection 7
3.2 Data Statistics 10
3.3 Dataset Characteristics 13

Chapter 4 Model Training and Evaluation 16
4.1 Random Classification 16
4.2 Landmark-Based Model 17
4.3 Computer Vision Models 17
4.4 Training Details 18
4.4.1 Train-Test Splits 19
4.4.2 Metrics 19

Chapter 5 Results 20
5.1 Results on Gesture-wise split 20
5.2 Results on Participant-wise split 20

Chapter 6 Discussion 23
6.1 Misclassfications 23
6.2 Key Frame Selection for Landmarks 24

Chapter 7 Future Work 28

Chapter 8 Conclusion 29

Bibliography 30

LIST OF TABLES

3.1 Comparison of existing gesture datasets, including ours. 11
3.2 Comprehensive table of all 49 gestures. The second column lists the 17 groups of gestures, while the third column lists all 49 gestures. The second and third columns are named as level 1 and level 2 gestures, respectively, in the two-level gesture hierarchy. 13

5.1 Reported metrics: precision, recall, F1, and top-*k* accuracy for all evaluated model architectures. The level column indicated higher (1 - 17 classes) or lower (2 - 49 classes) levels of abstraction for the defined classes. 21
5.2 Metrics for the 17 level 1 classes (participant-wise train-test split) 22

6.1 Average reported metrics: precision, recall, F1, and top-*k* accuracy. 10 frames were gathered for both methods. Static collection started at frame 20 of each video. Dynamic started at the unique key frame location for each individual video. 27
6.2 Standard deviation of reported metrics: precision, recall, F1, and top-*k* accuracy. 27

LIST OF FIGURES

3.1 The dataset includes the following specific types of data: (A) Sequences of cropped real-world images capturing the gesture from beginning to end; (B) Sequences of synthetic images depicting the same gesture against a black background; (C) Sequence of synthetic images against a HDRI (High Dynamic Range Image) scene; and (D) Sequence of synthetic images featuring various HDRI backgrounds, including nature, night, urban, indoor, and outdoor settings, as well as different angles around the Z-axis. 7
3.2 This image represents the three Microsoft Kinect cameras with a green screen setup for real data collection. 9
3.3 The image illustrates the Unity 3D software, which was used as a creative tool to generate a varied set of 49 random gestures intended to engage participants. 9
3.4 The image showcases the Blender software, utilized to generate synthetic data by employing a 3D hand model and animations. 11

4.1 Extracted landmarks superimposed on video still. 17

6.1 Confusion matrices for level 1 gesture classes for the traditional split. From top left: C3D, MViT, 3D ResNet and VideoMAE. 24
6.2 Confusion matrices for level 2 gesture classes for the traditional split. From top left: C3D, MViT, 3D ResNet and VideoMAE. 25
6.3 Confusion matrices for level 1 gesture classes for the participant-wise split. From top left: C3D, MViT, 3D ResNet and VideoMAE. 26

Chapter 1

Introduction

Gesture recognition is a current focus of extensive ongoing research and development in HCI and computer vision. As extended reality technology becomes increasingly prevalent, it is anticipated that people will increasingly use gestures as a means to interact with computer systems, rather than traditional peripheral devices. Previous research has shown that the use of hand gestures, such as pointing and swiping, in mid-air can result in fatigue ("gorilla arm") [1], making them unsuitable for extended use. In order to address this issue, Way *et al.* [2] have proposed the use of smaller hand motions, known as "microgestures," which require less movement. These microgestures are intended to mitigate the issue of fatigue and are adaptable to multiple situations such as human-object interaction and driving a car [3–7]. In addition, microgestures are suitable replacements for general gestures if there are physical space constraints while interacting with XR systems. Beginning with simple microgestures for communicating between human and computer, the study of microgestures has potential to facilitate delivery of complex information using microgesture sequences. Despite the potential benefits of utilizing microgestures for human-computer interaction, there exists no dataset of intentional[1] microgestures for the purpose of training computer vision algorithms for downstream interactions with computer systems, such as agents deployed on XR headsets. In order to address this challenge, this research introduces a novel video dataset of microgestures and investigates the performance of various machine learning models in classifying these gestures. Additionally, I discuss challenges and considerations in developing robust recognition of microgestures for human-computer interaction.

My research aims to better understand the challenges posed to recognition algorithms by microgestures, which are characterized by their subtle and fast nature. These properties make the task of

[1]On occasion, research in the computer vision community has used "microgesture" to refer to unconscious movements that indicate emotional state (e.g., [8]). I use "microgesture" in the HCI sense, referring to some *intentional* movement intended to convey information to a system.

gesture classification difficult, creating a challenge for those working in the field of vision-based gesture recognition for use in HCI. In response to this challenge, I created the "Microgesture" dataset, a novel dataset that combines both real and synthetic microgestures, providing a valuable resource for gesture classification. The Microgesture dataset is the largest dataset of its kind, containing both real and synthetically-rendered videos for the task of hand gesture recognition. The dataset includes 3,234 RGB-D videos captured in real-world scenarios from 10 different people, as well as 3,920 RGB videos generated synthetically. In addition, I developed a taxonomy of 49 semantically-distinct gestures with the goal of eventually improving human-computer interaction inputs. I anticipate that the Microgestures dataset will serve as a benchmark for future research efforts, providing a valuable resource for the academic and wider research community.

My specific contributions include:

- A novel video dataset consisting of real and synthetically-generated microgestures.
- Classification results from a variety of computer vision algorithms with this dataset, showcasing the feasibility of automatically classifying microgestures using RGB.

Chapter 2

Related Works

In this section, I summarize some of the earlier, relevant work in modeling and recognizing gestures, including key datasets and algorithms, and describe how my methods differ from previously-existing methods.

2.1 Datasets

There are a number of publicly-available datasets in the field of vision-based gesture recognition, including ChaLearn ISO/ConGD [9], Jester [10], EgoGesture [11] IPN Hand [12], nvGesture [13], and HaGRID [14]. It is important to keep in mind that these datasets lack synthetic data.

The Jester dataset [10] is the largest dataset containing 148,092 videos that were collected from 1,300 different human subjects, covering 27 distinct actions, totaling over 5 million frames. The authors propose that larger datasets are necessary to recognize complex and subtle gesture features. I have the same motivation for my dataset and create both synthetic and real data to give a wider sample to recognition algorithms. The Jester authors also note the elimination of the requirement for any external or wearable devices in their study, which I also adopted in my research.

The IPN hand dataset [12] is a continuous gesture dataset which contains 4,000 gesture instances with more than thirty different representative scenarios at 640 $\times$ 480 pixels at 30 frames per second. The IPN dataset has been enhanced by incorporating more features of real data to effectively train large deep learning networks. Building on this concept, I generated synthetic data using HDR (High Dynamic Range) images to add further diversity to the dataset.

EgoGesture [11] presents a dataset primarily focused on a first-person perspective, with over 24,000 gesture samples from 50 subjects, including 83 static and dynamic gesture classes. The authors propose that hand gestures are intuitive and natural for communicating with computers, and a first-person perspective in XR technology offers a unique human-centered viewpoint. I

believe such approaches have the potential to revolutionize human-computer interactions through the integration of microgestures.

The HaGRID dataset [14] aims to improve hand gesture recognition systems for various industries through device-human interaction. The dataset consists of 552,992 Full HD RGB images including 18 hand gestures and a "no gesture" class, with at least 34,730 unique scenes. While HaGRID focused on static hand gestures, my study emphasized the examination of dynamic hand gestures.

The iMiGUE dataset [15] for emotional AI research focuses on nonverbal microgestures, with 32 gesture categories, 2 emotions, and 18,499 samples from 72 subjects, obtained from online video interviews. iMiGUE assesses a model's ability to identify emotions by considering microgestures as an integrated whole, not just isolated prototypes in a sequence. This holistic approach matches my method for recognizing hand microgestures (Sec. 4).

The nvGesture dataset [13] contains 25 gesture classes (1,532 samples) of dynamic gestures from 20 subjects. The authors introduced nvGesture to address the challenge of detecting and classifying hand gestures in real-world human-computer interaction systems.

Finally, Wolf et al. [16] proposed a taxonomy for categorizing microgestures based on usability and scenarios, which provides a useful framework for design and evaluation. This taxonomy is used in my research to design microgestures.

2.2 Algorithms

The Jester dataset [10] has been used to show the capabilities of gestures in human-computer interaction and their potential applications in a wide range of industries, such as automotive, gaming, home automation, and consumer electronics. In constructing their networks [10], the authors employed a methodology using spatio-temporal filters as it effectively represented spatio-temporal data in previous approaches, e.g., [17]. Their model was trained using a stochastic gradient descent (SGD) algorithm, with a learning rate of 0.001, for a total of 100 epochs, and with no data

augmentation being employed during the training process. The final model achieved a top-level accuracy of 93.81%.

The IPN dataset [12] was designed to effectively detect and categorize the input stream; to accomplish this, they employed two hierarchical model structures, incorporating multimodal (RGB+depth) 3D CNN models with HarDNet (Harmonic Dense Networks) to achieve state-of-the-art results. The video sequences were segmented into isolated gestures using manually annotated beginning and ending frames. For the real data used in my study, I used the same methodology and manually annotated the beginning and ending frames.

In their study, the EgoGesture authors [11] adopted a multimodal approach, utilizing both handcrafted and deep-learned features to address two key tasks: classifying gestures in separated data and identifying gestures in continuous data. The authors demonstrated a high level of performance, achieving an 89.7% classification accuracy for segmented ego gestures in RGB-D data and a 0.718 Jaccard index using the LSTM-C3D-LL6s8 method for spotting and recognition in continuous data. For my research, I have chosen to focus on segmented data, where I have extracted frames from both synthetic and real datasets.

The HaGRID authors [14] used SSDLite with MobileNetV3 for hand detection, with ResNeXt-101 as the best for gesture classification and ResNet-152 for leading hand classification.

The iMiGUE authors [15] present a Seq2Seq-based unsupervised encoder-decoder model for microgesture recognition without labeled data. TSM [18], a supervised 2DCNN RGB modality, was used with a top 1 accuracy of 61.10% and top 5 accuracy of 91.24%.

The nvGesture dataset [13] was collected in both real and simulated environments using a head-mounted camera in both RGB and depth modalities. The proposed method, which combines color, depth, and optical flow, achieved 98.2% accuracy and a Jaccard score of 0.98.

In this study, I aim to investigate the classification of microgestures, which is a challenging task due to the rapid and intricate nature of the hand movements involved. Therefore, accurate classification of microgestures can have important implications for various fields, such as human-computer interaction, psychology, and linguistics. To achieve this objective, I propose a novel

Microgesture dataset, which is the first and most extensive dataset of its kind for microgesture classification, to the best of my knowledge.

Chapter 3

Methodology

Fig. 3.1 presents a visual overview of the different components of the Microgesture dataset: real and synthetic images against different backgrounds and in different orientations.

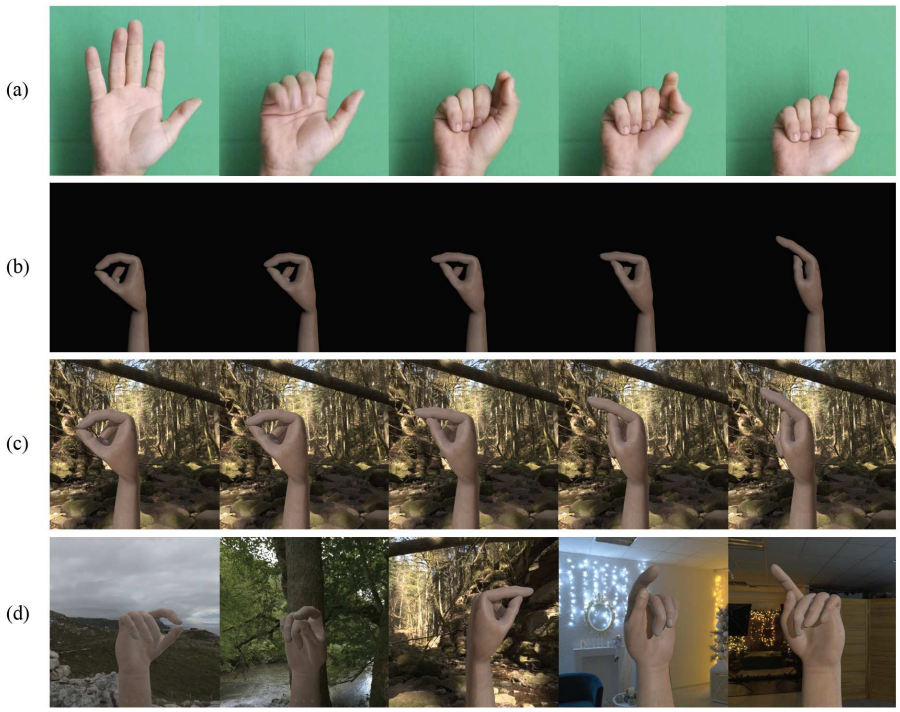

Figure 3.1: The dataset includes the following specific types of data: (A) Sequences of cropped real-world images capturing the gesture from beginning to end; (B) Sequences of synthetic images depicting the same gesture against a black background; (C) Sequence of synthetic images against a HDRI (High Dynamic Range Image) scene; and (D) Sequence of synthetic images featuring various HDRI backgrounds, including nature, night, urban, indoor, and outdoor settings, as well as different angles around the Z-axis.

3.1 Data Collection

I followed the gesture semantics proposed by Kendon [19] and elaborated by Lascarides and Stone [20], among others, and capture the pre-stroke, stroke (semantic head), and post-stroke of each gesture in the microgesture dataset.

Data collection consisted of two segments: recording participants making the 49 microgestures for the real dataset, and creating and rendering animations of the 49 microgestures for the synthetic dataset.

To record participants, I used the Microsoft Kinect Azure camera to record both RGB and depth (Fig. 3.2) at a resolution of 1,920 $\times$ 1,080 for 30 frames per second. Three Kinects were syncronized and positioned with a backdrop of a green screen approximately 30 - 40 cm away from the participant's hand. The Kinect system consists of three cameras: a primary camera and two sub-module cameras (referred to as camera 1 and camera 2). Each of these cameras has a different delay in capturing images or data. The primary camera is the main camera of the Kinect system, and it starts capturing images or data without any delay. In other words, it initiates its operation immediately. However, the sub-module cameras (camera 1 and camera 2) experience delays before they start capturing images or data. Camera 1 has a delay of 180 milliseconds, which means it starts capturing 180 milliseconds after the primary camera. Similarly, camera 2 has a delay of 360 milliseconds, so it starts capturing 360 milliseconds after the primary camera. These delays between the cameras can be important to consider in certain applications or scenarios. For example, if the Kinect system is used for motion tracking or depth sensing, the time difference between camera captures can impact the accuracy of tracking or measurements. It's essential to account for these delays when analyzing or synchronizing data from multiple Kinect cameras.

Each Kinect was angled at the central point where participants made the microgestures — this was done to maximize the amount of data being captured. Every Kinect was between 30 cm and 40 cm away from the participants' hand. I included a 10 second recording of a checkerboard at the beginning of each recording session in the event anyone wanted to perform their own depth calibrations. To facilitate participant interaction and imitation using synthetic data, I placed a screen in front of the camera setup. This allows participants to observe all the actions displayed on the screen and replicate them. To accomplish this, I utilized Unity 3D to develop software capable of generating 49 random movements (Fig. 3.3). The software generates these movements randomly and records relevant information such as the movement's name, start time, and end time.

Figure 3.2: This image represents the three Microsoft Kinect cameras with a green screen setup for real data collection.

This data is crucial for later analysis and retrieval of the dataset, ensuring a clear and organized record of the generated movements.

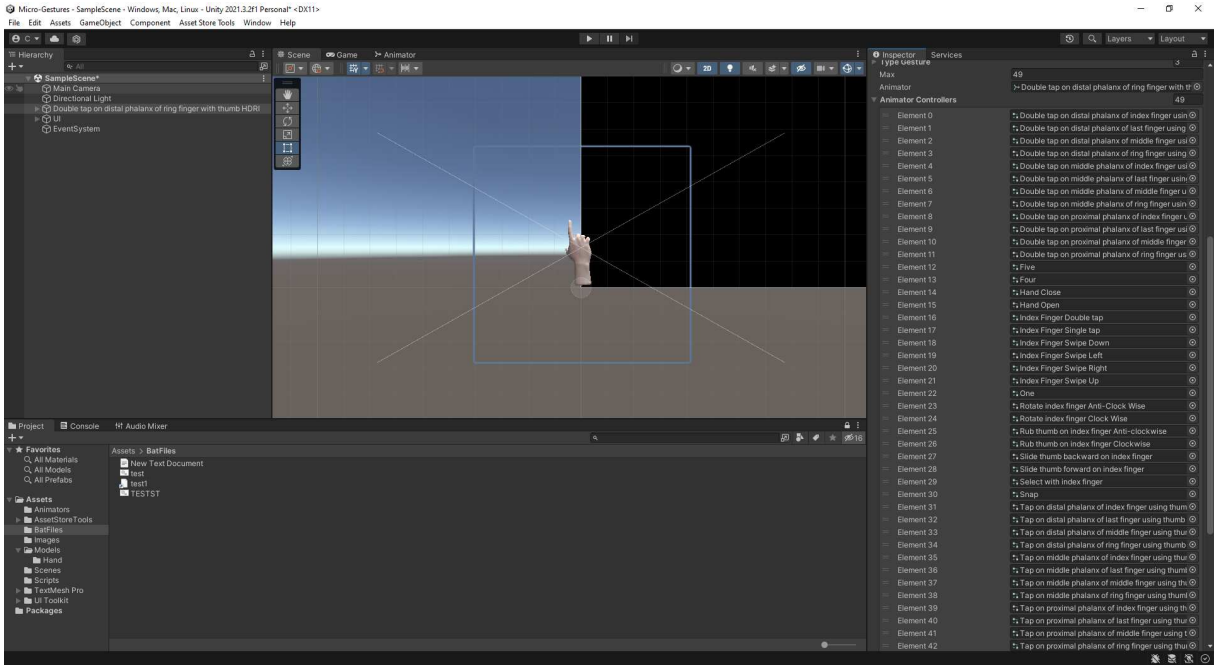

Figure 3.3: The image illustrates the Unity 3D software, which was used as a creative tool to generate a varied set of 49 random gestures intended to engage participants.

Prior to recording, participants were informed of the procedure to follow for making the microgestures. Emphasis was placed on explaining the differences between a microgesture over a gesture. Moving the arm or significantly moving the wrist would not qualify as a microgesture. For instruction on how to make each microgesture, participants were shown the synthetic data and the researcher performing the microgesture. Participants were allowed to practice making each microgesture as the recording proceeded. When they were ready to perform they would enter the starting position which was an open palm facing the center camera. After the participant made a microgesture, We marked if it was made correctly [21]. The use of an "incorrect label" to remove gesture frames that are performed incorrectly ensures that the final dataset only contains gestures that are performed correctly. The frequency of incorrect gestures indicated the difficulty performing that gesture caused participants. Overall the data collection, *Index finger swipe right*, *Index finger swipe down*, and *Index finger swipe left* had 21, 15, and 15 mistakes respectively. Each participant was recorded for about 45 minutes which allowed for 1-4 rounds of making each microgesture. Participants were randomly assigned to start with their left or right hand and would switch between rounds.

To create the synthetic data, a 3D model of a hand was created using Blender 3D software (Fig. 3.4). All 49 gesture animations were then played on the rigged hand, and the videos were generated with 2,000 $\times$ 2,000 resolution and 30 frames per second. The background was created using a black image and five High Dynamic Range Image (HDRI) scenes that were randomly provided to the software as a background of hand gestures. These five HDRI scenes were broken down into five categories: night, urban, indoor, and outdoor, and each category has four images for that scenario. The angle of the hand in each video was randomly chosen around the Z-axis (in the coordinate system used, the Z-axis is up-down).

3.2 Data Statistics

For the real dataset I had 10 participants (60% were male and 40% were female). I collected a total of 66 videos containing 49 gestures each. This resulted in 66 instances of each microgesture

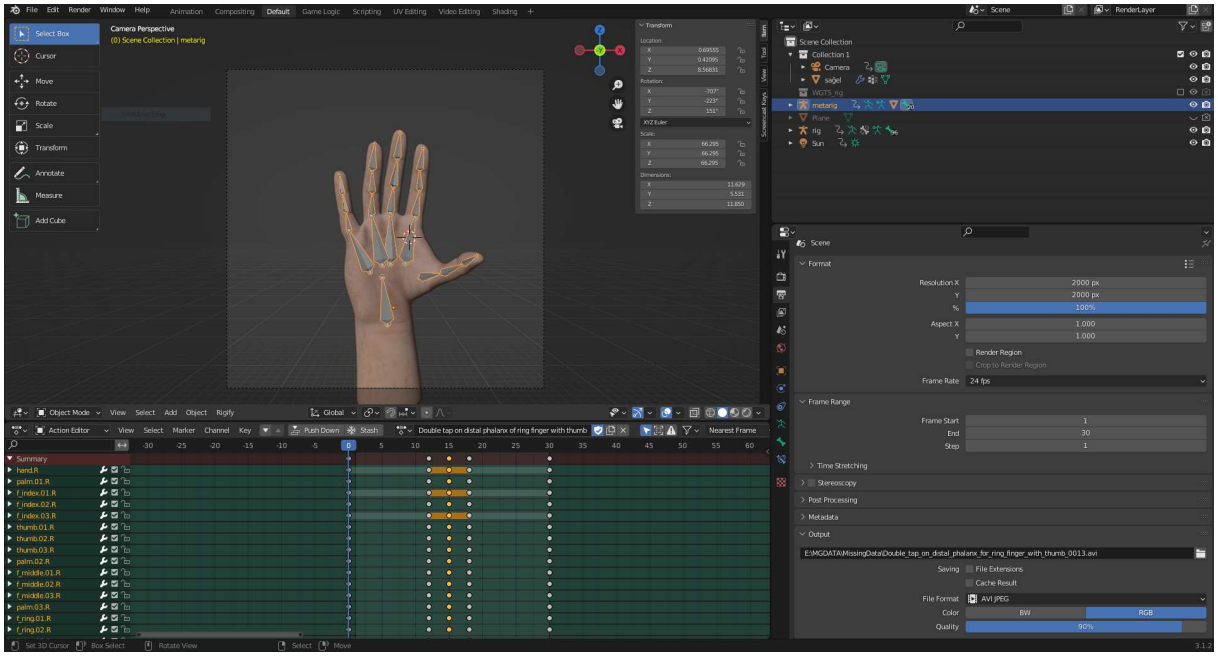

Figure 3.4: The image showcases the Blender software, utilized to generate synthetic data by employing a 3D hand model and animations.

and 3,234 total instances. From this total, I excluded gestures that were incorrectly made by participants, leaving us with 3,054 instances from which 184,107 frames were extracted. The dataset of gestures is characterized by its uniqueness, primarily stemming from the inclusion of participants from various age groups and racial backgrounds, as well as the noticeable differences in the speed at which gestures were executed. By encompassing participants of different ages and races, the dataset reflects a broader representation of the population and ensures a more comprehensive understanding of how gestures vary across demographic groups. This inclusivity allows for insights

Table 3.1: Comparison of existing gesture datasets, including ours.

Datasets	Samples	Labels	Subjects	Scenes	Task
ChaLearn ISO/ConGD [9]	47,9339	249	21	1	classification, detection
IPN Hand	4218	13	50	28	classification
Jester Dataset	148,092	27	1376	N/A	classification
EgoGesture	24,161	83	50	6	classification, detection
nvGesture	1532	25	20	1	classification, detection
HaGRID	552,992	19	34,730	34,730	classification
iMiGUE	18,499	32	72	N/A	classification
Microgesture	301,707	49	10	21	classification

into potential differences in gesture styles, preferences, or cultural influences. Furthermore, the variations in gesture speed within the dataset add another layer of richness and complexity. The diverse pacing of gestures provides valuable information about individual motor skills, reaction times, and expressive nuances. It enables researchers or developers to explore and analyze the impact of timing and rhythm on gesture recognition systems or other related applications. Overall, the unique collection of gestures within this dataset offers a valuable

For the synthetic dataset, 80 videos of each microgesture were created, all evenly split between the left and right hands. The background for both the left and right hands were split such that both had 20 videos with the black background and 20 videos with the HDRI background. 3,920 total videos were created from which 117,600 frames were extracted.

In the real data collection, the number of frames recorded for each gesture varies between 60 and 81 frames. This variation can be attributed to the different speeds at which participants perform the gestures. Some participants may execute the gestures quickly, resulting in fewer frames captured, while others may perform the gestures at a slower pace, leading to a greater number of frames being recorded. This variability in frame count provides a more accurate representation of how gestures are naturally performed by individuals in real-world scenarios. On the other hand, in the synthetic data, each gesture is captured with a consistent speed, resulting in a fixed duration of 30 frames for each gesture. This uniformity ensures that all synthetic gestures are standardized in terms of timing and duration, allowing for easier comparison and analysis. While the synthetic data may not encompass the same level of variability as the real data, it provides a controlled and consistent basis for studying gesture patterns and developing gesture recognition algorithms. By comparing the real data with the synthetic data, researchers and developers can gain insights into the impact of participant speed variations on gesture recognition systems. Additionally, the consistent speed of synthetic data allows for the evaluation and optimization of gesture recognition algorithms in a controlled setting. The data statistics of my dataset are represented in Table 3.1 — I also compare my gestures with other gesture datasets that are currently accessible online.

3.3 Dataset Characteristics

I designed all 49 gestures in this dataset to be easy, fast, and low-effort (and hence low-fatigue over the long term), since these are essential qualities for microgestures to have when considering HCI applications. I classified the gestures into 17 distinct groups by features of the gesture. I called this gesture categorization a two-level hierarchy. Level 1 of the hierarchy consists of the gesture groups and level 2 includes the individual gesture types. Table 3.2 describes the taxonomy of all 49 gesture classes.

Table 3.2: Comprehensive table of all 49 gestures. The second column lists the 17 groups of gestures, while the third column lists all 49 gestures. The second and third columns are named as level 1 and level 2 gestures, respectively, in the two-level gesture hierarchy.

	Level 1	Level 2
1	Single tap index	Tap on distal phalanx of index finger w/ thumb
2		Tap on middle phalanx of index finger w/ thumb
3		Tap on proximal phalanx of index finger w/ thumb
4	Single tap middle	Tap on distal phalanx of middle finger w/ thumb
5		Tap on middle phalanx of middle finger w/ thumb
6		Tap on proximal phalanx of middle finger w/ thumb
7	Single tap ring	Tap on distal phalanx of ring finger w/ thumb
8		Tap on middle phalanx of ring finger w/ thumb
9		Tap on proximal phalanx of ring finger w/ thumb
10	Single tap last	Tap on distal phalanx of last finger w/ thumb
11		Tap on middle phalanx of last finger w/ thumb
12		Tap on proximal phalanx of last finger w/ thumb
13	Double tap index	2x tap on distal phalanx of index finger w/ thumb
14		2x tap on middle phalanx of index finger w/ thumb
15		2x tap on proximal phalanx of index finger w/ thumb
16	Double tap middle	2x tap on distal phalanx of middle finger w/ thumb

Table 3.2 – continued from previous page

17		2x tap on middle phalanx of middle finger w/ thumb
18		2x tap on proximal phalanx of middle finger w/ thumb
19	Double tap ring	2x tap on distal phalanx of ring finger w/ thumb
20		2x tap on middle phalanx of ring finger w/ thumb
21		2x tap on proximal phalanx of ring finger w/ thumb
22	Double tap last	2x tap on distal phalanx of last finger w/ thumb
23		2x tap on middle phalanx of last finger w/ thumb
24		2x tap on proximal phalanx of last finger w/ thumb
25	Tap once	Index finger single tap
26	Tap twice	Index finger double tap
27	Move	Index finger swipe up
28		Index finger swipe down
29		Index finger swipe right
30		Index finger swipe left
31		Select with index finger
32	Numbers	One
33		Two
34		Three
35		Four
36		Five
37	Rotate (In air)	Rotate index finger anti-clockwise
38		Rotate index finger clockwise
39	Rotate (Rub)	Rub thumb on index finger anti-clockwise
40		Rub thumb on index finger clockwise
41	Slide	Slide thumb backward on index finger

Table 3.2 – continued from previous page

42		Slide thumb forward on index finger
43	Open/close	Hand open
44		Hand close
45	Zoom	Zoom out using palm
46		Zoom out with index finger and thumb
47		Zoom in using palm
48		Zoom in with index finger and thumb
49	Snap	Snap

Chapter 4

Model Training and Evaluation

To train models on the real dataset, using standard machine learning practice, I collaborated with my co-authors, benefiting from their valuable assistance. [21] We carefully considered two different ways of dividing the data into training, validation, and test segments (splits): a split where all data from individual participants were grouped together (a "participant-wise" split), and a split where they were not (hereafter, a "traditional" or "gesture-wise" split). For traditional gesture-based distribution, 80%, 10%, and 10% of gestures were allocated to the train, validation, and test sets, respectively. Since the allocation was done by level 2 gestures, the dataset contained an equal portion of level 1 gestures for the train, validation, and test sets. The dataset of the traditional split was used for fine-tuning various methods including SOTA models (see Table 5.1). With the participant-wise data split, our goal was to show where and using which models the trained visual features would applicable to general cases where the individual person has never been seen before by the model. We assigned gestures belonging to a male and female participant to the validation and test sets respectively, and the gestures of the remaining participants were allocated to the training set. The training on the participant-wise split was done for level 1 gestures only (see Table 5.2).

4.1 Random Classification

We first established a random chance classification baseline. For each video in the test set, we assigned the predicted class randomly, and computed classification accuracy and other metrics for these random predictions. We then averaged these metrics across 10,000 iterations of randomly guessing to minimize any noise from the random labeling.

4.2 Landmark-Based Model

Our second baseline approach uses 3D joint positions extracted from the videos rather than raw visual features. Videos were preprocessed to extract landmarks on the hands using MediaPipe [22]. MediaPipe is a two-stage pipeline that tracks hands using 21 landmark points made up of X, Y, and Z coordinates. MediaPipe processes each frame into an array of landmarks normalized relative to the image dimensions. Fig. 4.1 shows extracted landmarks superimposed on a video frame. MediaPipe performs well for detecting hands on a frame by frame basis, and has been used in many projects to aid in static single frame gesture classification, however, not much work has been done to classify gestures that span multiple frames, like exist in this dataset, and no works have explored microgestures.

We established a model baseline using this landmark-based approach by selecting 10 frames starting at the 20th frame for every video in the dataset. Extracted landmarks from the collected frames were then fed into a neural network classifier (details in Sec. 4.4).

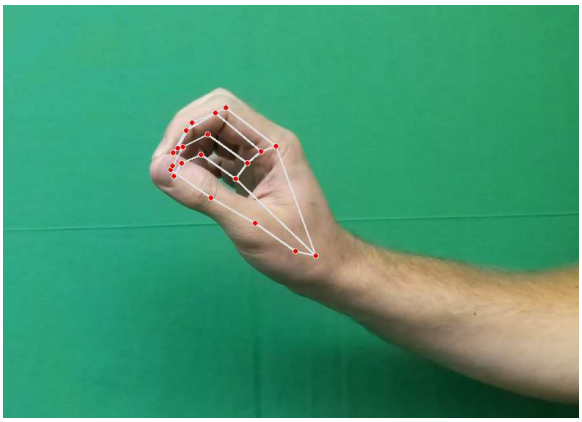

Figure 4.1: Extracted landmarks superimposed on video still.

4.3 Computer Vision Models

We evaluated various computer vision models to determine the feasibility of detecting micro-gestures with a computer vision method. Models were evaluated on a both level 1 and level 2

of my gesture taxonomy. Following general practice in gesture recognition, we focused on action recognition and gesture recognition models which weI fine-tuned for my dataset. The action recognition models we evaluated were VideoMAE [23], Multiscale Vision Transformers (MViT) [24], 3D ResNet [25], and C3D [26]. Note that we used pretrained weights for VideoMAE and 3D ResNet whereas MViT and C3D were initialized with random weights (see 4.4). The gesture recognition models, based on Köpüklü et al. [27] used a ResNeXt [28] architecture. These models were pretrained on two different datasets: EgoGesture [11] and nvGesture [13], following the training details of each respective work.

4.4 Training Details

we trained all models on the gesture-wise and participant-wise splits of my dataset. To train the action recognition models, we used 1,000 epochs to fine-tune/train VideoMAE, MViT (base), 3D ResNet, and C3D with a batch size of 8 for VideoMAE and 16 for the others. For all these models, we used an SGD optimizer with a learning rate of $1e^{-4}$. The input images were resized to $I \in R^{3\times224\times224}$, except for when fine-tuning the gesture recognition models, where the input size was $I \in R^{3\times112\times112}$ [11, 13]. These models were checkpointed every 5 epochs and the best-performing checkpoints were used for evaluation.

In addition, 480 epochs were used to fine tune the state of the art gesture recognition models (batch size of 16 and a learning rate of $1e^{-4}$ were used). 480 epochs was chosen to make a direct comparison to the best-performing checkpoint of VideoMAE.

For the landmark-based model, we used a 2-layer feed-forward neural network with 20 and 10 units, respectively, all with ReLU activation, followed by a final softmax classification layer. The model was trained for 100 epochs with an Adam optimizer, a learning rate of 0.001, sparse categorical cross-entropy loss, and a batch size of 16.

4.4.1 Train-Test Splits

For this dataset, we created 2 different train-test splits for evaluation. The first is a traditional **gesture-wise** split using an 80:10:10 ratio for train, validation, and test sets respectively. The second is a **participant-wise** split with the validation and test sets each having gestures from 1 human subject and the rest make up the train set. For each split (gesture and participant), we fine-tune/train our models for both levels in the hierarchy of gestures, the higher abstract classes and the lower fine-grained classes.

4.4.2 Metrics

We used the F1 score and top K accuracy to evaluate the classification task. To compare how each model predicts the exact class of gesture, we measured F1 scores along with precision and recall. We took advantage of top K accuracy to see the predicted candidates are in the K number of options although a model missed the first predictions.

Chapter 5

Results

My results present precision, recall, macroaveraged F1, and top-*k* classification accuracy for multiple models, including the landmarks-based classifier. I also contextualize the results by presenting the “random chance" baseline (Sec. 4.1). Metrics were calculated using the Scikit-learn package, which first calculates macroaveraged F1 over each class and then averages the F1 scores for all classes.[2]

5.1 Results on Gesture-wise split

Table 5.1 compares all classification results. Ultimately, VideoMAE exhibited the best performance over both vision and landmark-based models. The top-*k* accuracy of vision models indicate that the visual features of microgestures are learnable with neural networks, indicating the potential to utilize microgestures in HCI applications that use such models. Interestingly, the models that were pretrained on EgoGesture and nvGesture exhibited similar performance to 3D ResNet, which is based on action recognition, indicating that pre-training on gesture may not provide particular benefits. For gesture-wise classification at the more fine-grained level (level 2 in my taxonomy), C3D barely performed better than random classification and the landmark-based model was on par with or slightly worse than random, showing that classification at this level of granularity is difficult enough to likely require a custom deep learning model fine-tuned on the dataset.

5.2 Results on Participant-wise split

As shown in Table 5.2, the performance of most models on the participant-wise split was lower than on the level 1 gesture-wise split. The exception is 3D ResNet, which showed comparable performance. In addition, MViT and C3D models showed a little performance gap from the random

[2]As a result, classes with lower than average sample support and lower than average F1 may cause overall macroaveraged F1 for a model to fall below both average precision and recall.

chance model, indicating these models are poor fits for microgesture classification. Generally, the drop in accuracy when training and evaluating on different participants is expected — such evaluations approximate how well a trained model could be expected to generalize to an unseen participant.

Table 5.1: Reported metrics: precision, recall, F1, and top-*k* accuracy for all evaluated model architectures. The level column indicated higher (1 - 17 classes) or lower (2 - 49 classes) levels of abstraction for the defined classes.

Level	Model	Precision	Recall	F1	Top-1	Top-3	Top-5
1	Random	9.70	9.03	9.27	10.92	-	-
	Landmarks	14.85	14.86	11.59	19.37	54.93	72.54
	VideoMAE	74.38	73.67	72.72	75.74	94.49	97.79
	MViT[1]	52.67	52.41	51.60	52.86	83.57	92.50
	3D ResNet	63.68	58.81	60.36	61.79	92.50	98.21
	C3D[1]	31.77	32.27	31.70	35.71	69.29	83.21
	Köpüklü et al. [27][2]	54.57	53.99	53.03	54.93	85.92	92.61
	Köpüklü et al. [27][3]	57.75	56.38	56.03	59.15	88.03	92.96
2	Random	6.91	6.33	6.27	6.34	-	-
	Landmarks	3.79	7.21	3.63	7.40	20.78	33.45
	VideoMAE	67.58	64.90	64.79	65.07	90.44	96.32
	MViT*	41.27	39.39	38.16	39.29	67.50	77.14
	3D ResNet	44.88	43.03	41.35	43.57	78.93	91.43
	C3D*	8.73	9.86	8.87	9.64	25.00	41.07
	Köpüklü et al. [27][2]	44.78	41.02	40.45	41.20	73.24	82.04
	Köpüklü et al. [27][3]	55.18	48.67	48.19	48.59	72.89	82.04

Table 5.2: Metrics for the 17 level 1 classes (participant-wise train-test split)

Model	Precision	Recall	F1	Top-1	Top-3	Top-5
Random	14.84	17.48	15.11	14.18	-	-
VideoMAE	25.05	25.87	20.25	30.15	66.18	81.61
MViT[1]	5.35	11.16	6.69	15.44	29.41	44.12
3D ResNet	63.24	55.14	56.27	57.35	89.71	95.59
C3D[1]	12.37	18.18	13.05	18.38	43.38	57.35
Köpüklü et al. [27][2]	30.03	32.03	27.67	34.04	68.09	78.01
Köpüklü et al. [27][3]	26.42	34.97	26.09	31.21	61.70	74.47

[1]Trained from scratch.

[2]Pretrained on EgoGesture.

[3]PreTrained on nvGesture.

Chapter 6

Discussion

Figs. 6.1 and 6.2 show confusion matrices for all four action recognition models trained on the gesture-wise level 1 and level 2 classes, respectively. Classifiers generally performed similarly across both levels — for example, C3D (top-left) made the most mistakes on both levels while VideoMAE (bottom-right) performed best. Although both C3D and VideoMAE show similar results on the UCF101 dataset (90.4% and 91.3%, respectively), VideoMAE substantially outperforms C3D on my real dataset for both levels, indicating there are specific design decisions that successful microgesture models will need to address [23, 26, 29].

Fig. 6.3 shows confusion matrices plots for all four action recognition models trained on the participant-wise split. We can see that all models had similar and better performance for the *single tap middle* and *number* gesture classes, with 3D ResNet (bottom-left) making the fewest mistakes. In addition, 3D ResNet was robust to *Double tap index*, *Double tap middle*, *double tap ring*, and *double tap last*.

6.1 Misclassfications

Comparing the best-performing models, VideoMAE and 3D ResNet, for both data splits, we can see what common microgestures are difficult for neural network models to distinguish. For the level 1 gesture-wise split, both models have trouble identifying the *tap twice* and *tap once* classes. Looking at the fine-grained microgestures from level 2, VideoMAE is unable to predict the *slide thumb forward on index finger* (ID 42 — see Table 3.2) microgestures, and 3D ResNet is unable to predict the similar *slide thumb backward on index finger* (ID 41) microgesture, perhaps indicating that these subtle distinctions, even down to the finger level, are difficult for recognition models in general.

Looking at the confusion matrices from the participant-wise split (Fig. 6.3) we can also see that 3D ResNet and VideoMAE have difficulty in identifying the *tap twice* and *slide* gesture classes.

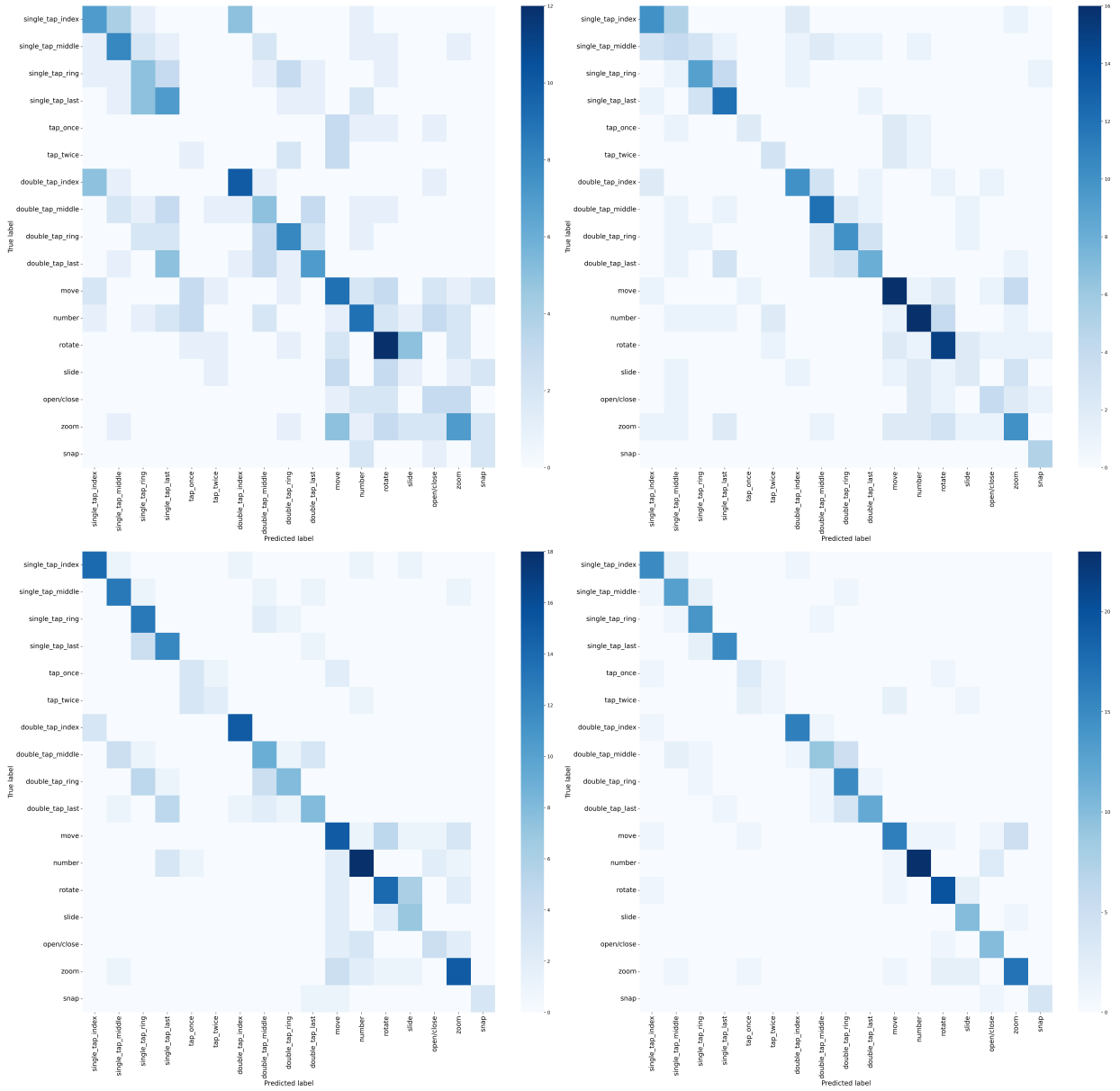

Figure 6.1: Confusion matrices for level 1 gesture classes for the traditional split. From top left: C3D, MViT, 3D ResNet and VideoMAE.

6.2 Key Frame Selection for Landmarks

Recognizing more complex gestures is substantially easier if precise key frames are identified prior to recognition, since this preemptively filters excess noise. Since the landmarks-based approach performed poorly yet is fast to train, but also used a constant frame selection across all videos, my collaborators and I wanted to investigate if smarter selection of frames could improve

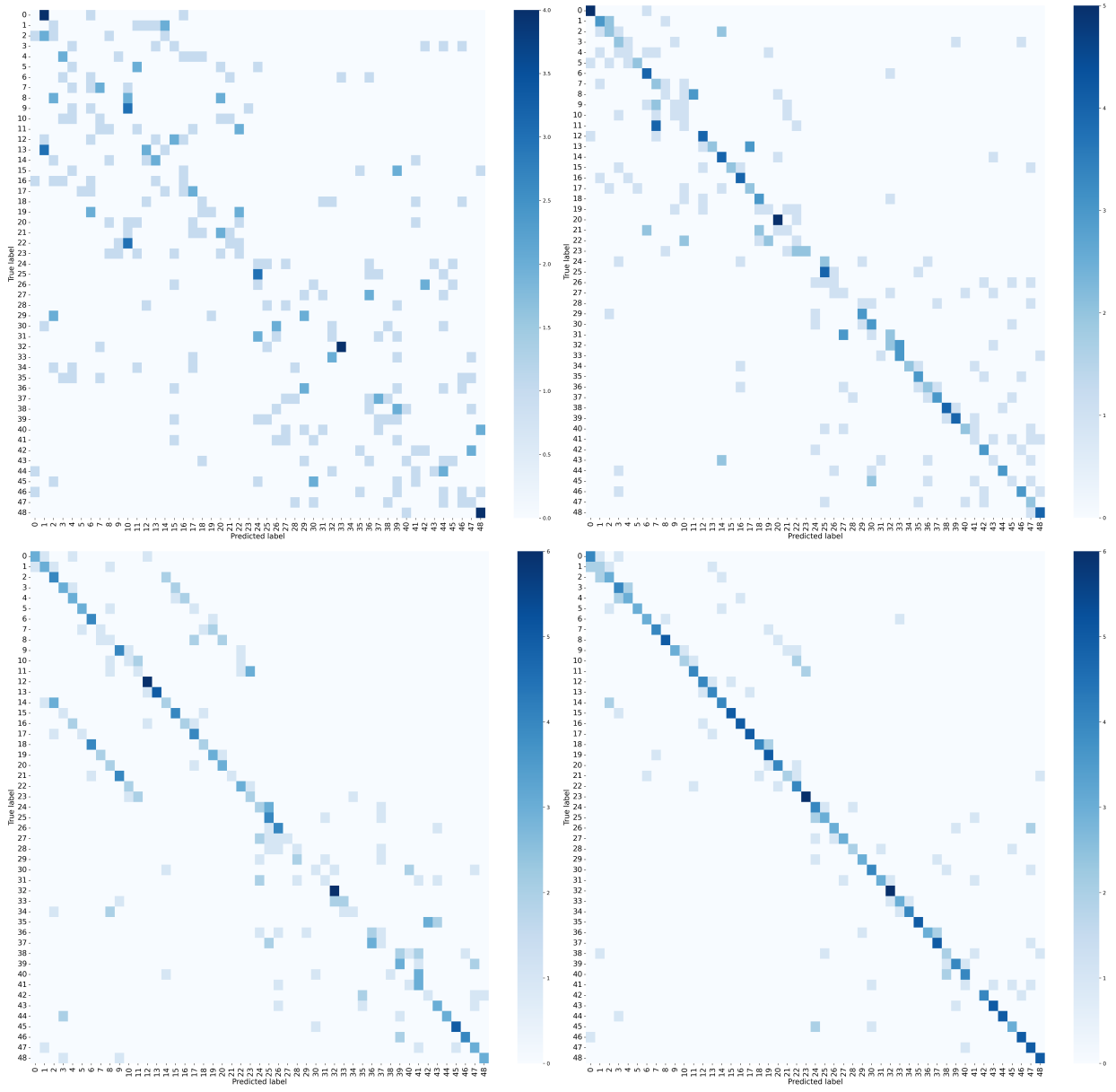

Figure 6.2: Confusion matrices for level 2 gesture classes for the traditional split. From top left: C3D, MViT, 3D ResNet and VideoMAE.

this model's performance. [30] developed a key frame annotation solution that locates key frames using a three stage pipeline. First, a simple binary classifier recognizes the general static shape of a gesture of interest (a "hold"). Next frames in a gesture video are grouped by relative changes in motion to create "segments." Finally, we identify segments with some percentage of frames in "hold" using another binary classifier. Using this data, we can identify the start and end of the key frames, along with the start and end of the "peak" segment, which contain the most still frames in

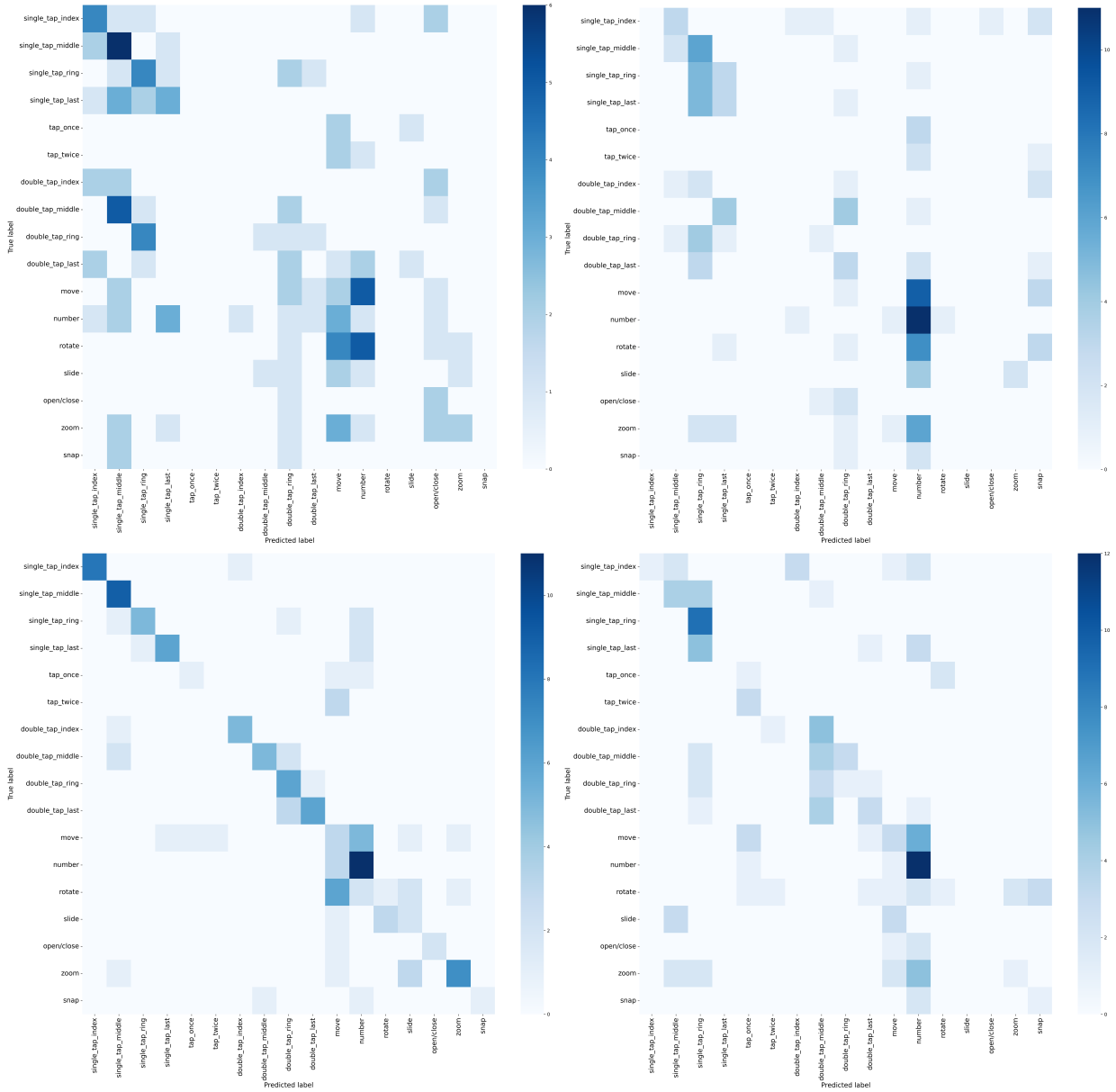

Figure 6.3: Confusion matrices for level 1 gesture classes for the participant-wise split. From top left: C3D, MViT, 3D ResNet and VideoMAE.

"hold" and could be considered the peak of the key frames [30]. We hypothesized that we could improve the overall performance of the landmark-based classifier over multiple frames using these annotated values to select dynamic features on a per-gesture basis.

To test this procedure, an additional experiment was run on a subset of the Microgesture data. The gestures included in the subset were *snap*, *two*, *index finger swipe right*, *hand close*, and *zoom in with palm*. We trained a feedforward neutral network on the subset to retrieve key frames us-

Table 6.1: Average reported metrics: precision, recall, F1, and top-*k* accuracy. 10 frames were gathered for both methods. Static collection started at frame 20 of each video. Dynamic started at the unique key frame location for each individual video.

Method	**Precision**	**Recall**	**F1**	**Top-1**	**Top-3**
Static	35.28	39.16	33.86	41.48	83.49
Dynamic	66.35	66.48	62.78	69.10	89.10

Table 6.2: Standard deviation of reported metrics: precision, recall, F1, and top-*k* accuracy.

Method	**Precision**	**Recall**	**F1**	**Top-1**	**Top-3**
Static	23.99	20.36	21.97	21.16	9.78
Dynamic	22.80	19.00	22.00	18.28	9.86

ing the same hyperparameters defined in Sec. 4.4, using a leave-one-out split for each of the 10 participants. Tables 6.1 and 6.2 show the average performance and standard deviation of classification using the dynamic key frame selection compared to the static method, run on the same 5 gestures [30]. The increase in performance using the dynamic method shows promise as a solution to improve the overall performance of the landmark based model if trained using the dynamic key frames across the entire dataset. This makes sense, since the microgesture movements are quite small and the key frame identification eliminates excess noise.

Chapter 7

Future Work

Here, I presented a novel microgesture dataset to serve as a benchmark and to encourage research into the use of microgestures interactive systems. However, the current dataset has only minimal egocentric data (Sec. 3.1) and thus limits the ability to adopt microgestures in XR contexts — microgestures are particularly important for XR contexts where the user could conceivably interact with the system throughout the day.

From an HCI perspective, future work should evaluate how effective my set of microgestures are from both a fatigue estimation perspective and from a usability perspective. In parallel, computer vision research should focus on identifying which gesture types are most easily recognized by trained systems. In this work, I showcase the feasibility of detecting various microgestures, but future work can likely surpass these baselines since I did not explore custom models.

Our used novel technique for intelligent key frame selection [30] increases both performance and training efficiency — where training modern computer vision models as we used can each take between 1-5 days, the classifier portions of the key frame selection pipeline can be trained in a matter of minutes, and the entire procedure approaches the accuracy of some of the top-performing vision models. This technique is discussed in more detail in [30], but was evaluated only on a subset of the dataset. A larger evaluation is the subject of future work.

Chapter 8

Conclusion

Microgestures represent a naturalistic and low-effort strategy for human-computer interaction. However, prior to this work, microgesture datasets were limited at best, especially from the perspective of what would be needed to study them from an AI application perspective. My novel microgesture dataset includes video recordings of a hierarchy of microgesture types from a multitude of participants. Further, it contains synthetic videos of gestures being performed which can be augmented with various backgrounds or variations to improve the robustness of the trained model.

My experimental results, which compared random chance, landmark-based, and computer vision models, show how well micoregestures can be recognized by a computer vision system. While these results showcase the feasibility of detecting microgestures, they also highlight the difficulties that need to be overcome for an interactive system to capitalize on microgestures. While this is just the start of the work that needs to be accomplished to integrate microgestures into HCI applications, I believe it provides a solid foundation on which the HCI and AI research communities can build.

Bibliography

[1] Juan David Hincapié-Ramos, Xiang Guo, Paymahn Moghadasian, and Pourang Irani. Consumed endurance: a metric to quantify arm fatigue of mid-air interactions. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 1063–1072, 2014.

[2] David Way and Joseph Paradiso. A usability user study concerning free-hand microgesture and wrist-worn sensors. In *2014 11th International Conference on Wearable and Implantable Body Sensor Networks*, pages 138–142. IEEE, 2014.

[3] Euan Freeman, Gareth Griffiths, and Stephen A Brewster. Rhythmic micro-gestures: Discreet interaction on-the-go. In *Proceedings of the 19th ACM International Conference on Multimodal Interaction*, pages 115–119, 2017.

[4] Renate Häuslschmid, Benjamin Menrad, and Andreas Butz. Freehand vs. micro gestures in the car: Driving performance and user experience. In *2015 IEEE Symposium on 3D User Interfaces (3DUI)*, pages 159–160. IEEE, 2015.

[5] Robert Neßelrath, Mohammad Mehdi Moniri, and Michael Feld. Combining speech, gaze, and micro-gestures for the multimodal control of in-car functions. In *2016 12th International Conference on Intelligent Environments (IE)*, pages 190–193. IEEE, 2016.

[6] Adwait Sharma, Michael A Hedderich, Divyanshu Bhardwaj, Bruno Fruchard, Jess McIntosh, Aditya Shekhar Nittala, Dietrich Klakow, Daniel Ashbrook, and Jürgen Steimle. Solofinger: Robust microgestures while grasping everyday objects. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*, pages 1–15, 2021.

[7] Adwait Sharma, Joan Sol Roo, and Jürgen Steimle. Grasping microgestures: Eliciting single-hand microgestures for handheld objects. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, pages 1–13, 2019.

[8] Haoyu Chen, Xin Liu, Xiaobai Li, Henglin Shi, and Guoying Zhao. Analyze spontaneous gestures for emotional stress state recognition: A micro-gesture dataset and analysis with deep learning. In *2019 14th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2019)*, pages 1–8. IEEE, 2019.

[9] Jun Wan, Yibing Zhao, Shuai Zhou, Isabelle Guyon, Sergio Escalera, and Stan Z Li. Chalearn looking at people rgb-d isolated and continuous datasets for gesture recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 56–64, 2016.

[10] Joanna Materzynska, Guillaume Berger, Ingo Bax, and Roland Memisevic. The jester dataset: A large-scale video dataset of human gestures. In *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops*, pages 0–0, 2019.

[11] Yifan Zhang, Congqi Cao, Jian Cheng, and Hanqing Lu. Egogesture: a new dataset and benchmark for egocentric hand gesture recognition. *IEEE Transactions on Multimedia*, 20(5):1038–1050, 2018.

[12] Gibran Benitez-Garcia, Jesus Olivares-Mercado, Gabriel Sanchez-Perez, and Keiji Yanai. Ipn hand: A video dataset and benchmark for real-time continuous hand gesture recognition. In *2020 25th international conference on pattern recognition (ICPR)*, pages 4340–4347. IEEE, 2021.

[13] Pavlo Molchanov, Xiaodong Yang, Shalini Gupta, Kihwan Kim, Stephen Tyree, and Jan Kautz. Online detection and classification of dynamic hand gestures with recurrent 3d convolutional neural network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4207–4215, 2016.

[14] Alexander Kapitanov, Andrew Makhlyarchuk, and Karina Kvanchiani. Hagrid-hand gesture recognition image dataset. *arXiv preprint arXiv:2206.08219*, 2022.

[15] Xin Liu, Henglin Shi, Haoyu Chen, Zitong Yu, Xiaobai Li, and Guoying Zhao. imigue: An identity-free video dataset for micro-gesture understanding and emotion analysis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10631–10642, 2021.

[16] Katrin Wolf, Anja Naumann, Michael Rohs, and Jörg Müller. A taxonomy of microinteractions: Defining microgestures based on ergonomic and scenario-dependent requirements. In *13th International Conference on Human-Computer Interaction (INTERACT)*, number Part I, pages 559–575. Springer, 2011.

[17] Yale Song, David Demirdjian, and Randall Davis. Tracking body and hands for gesture recognition: Natops aircraft handling signals database. In *2011 IEEE International Conference on Automatic Face & Gesture Recognition (FG)*, pages 500–506. IEEE, 2011.

[18] Ji Lin, Chuang Gan, and Song Han. Tsm: Temporal shift module for efficient video understanding. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 7083–7093, 2019.

[19] Adam Kendon. *Gesture: Visible action as utterance*. Cambridge University Press, 2004.

[20] Alex Lascarides and Matthew Stone. A formal semantic analysis of gesture. *Journal of Semantics*, 26(4):393–449, 2009.

[21] Chirag Kandoi, Changsoo Jung, Sheikh Mannan, Hannah VanderHoeven, Quincy Meisman, Nikhil Krishnaswamy, and Nathaniel Blanchard. Intentional Microgesture Recognition for Extended Human-Computer Interaction. In *International Conference on Human-Computer Interaction (HCII)*. Springer, 2023.

[22] Camillo Lugaresi, Jiuqiang Tang, Hadon Nash, Chris McClanahan, Esha Uboweja, Michael Hays, Fan Zhang, Chuo-Ling Chang, Ming Yong, Juhyun Lee, et al. Mediapipe: A framework for perceiving and processing reality. In *Third Workshop on Computer Vision for AR/VR at IEEE Computer Vision and Pattern Recognition (CVPR)*, volume 2019, 2019.

[23] Zhan Tong, Yibing Song, Jue Wang, and Limin Wang. Videomae: Masked autoencoders are data-efficient learners for self-supervised video pre-training. *arXiv preprint arXiv:2203.12602*, 2022.

[24] Haoqi Fan, Bo Xiong, Karttikeya Mangalam, Yanghao Li, Zhicheng Yan, Jitendra Malik, and Christoph Feichtenhofer. Multiscale vision transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6824–6835, 2021.

[25] Kensho Hara, Hirokatsu Kataoka, and Yutaka Satoh. Can spatiotemporal 3d cnns retrace the history of 2d cnns and imagenet? In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 6546–6555, 2018.

[26] Du Tran, Lubomir Bourdev, Rob Fergus, Lorenzo Torresani, and Manohar Paluri. Learning spatiotemporal features with 3d convolutional networks. In *Proceedings of the IEEE international conference on computer vision*, pages 4489–4497, 2015.

[27] Okan Köpüklü, Ahmet Gunduz, Neslihan Kose, and Gerhard Rigoll. Real-time hand gesture detection and classification using convolutional neural networks. In *2019 14th IEEE international conference on automatic face & gesture recognition (FG 2019)*, pages 1–8. IEEE, 2019.

[28] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1492–1500, 2017.

[29] Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. Ucf101: A dataset of 101 human actions classes from videos in the wild. *arXiv preprint arXiv:1212.0402*, 2012.

[30] Hannah VanderHoeven, Nathaniel Blanchard, and Nikhil Krishnaswamy. Robust motion recognition using gesture phase annotation. In *Digital Human Modeling and Applications in Health, Safety, Ergonomics and Risk Management. Human Body, Motion and Behavior: 14th*

International Conference, DHM 2023, Held as Part of the 25th HCI International Conference, HCII 2023. Springer, 2023.