HandyLabels: Real-Time Annotation Tool Using Hand Gesture Recognition

SACHIN KUMAR SINGH, RPTU Kaiserslautern-Landau & DFKI GmbH, Germany
KO WATANABE, RPTU Kaiserslautern-Landau & DFKI GmbH, Germany
BRIAN B. MOSER, RPTU Kaiserslautern-Landau & DFKI GmbH, Germany
ANDREAS DENGEL, RPTU Kaiserslautern-Landau & DFKI GmbH, Germany

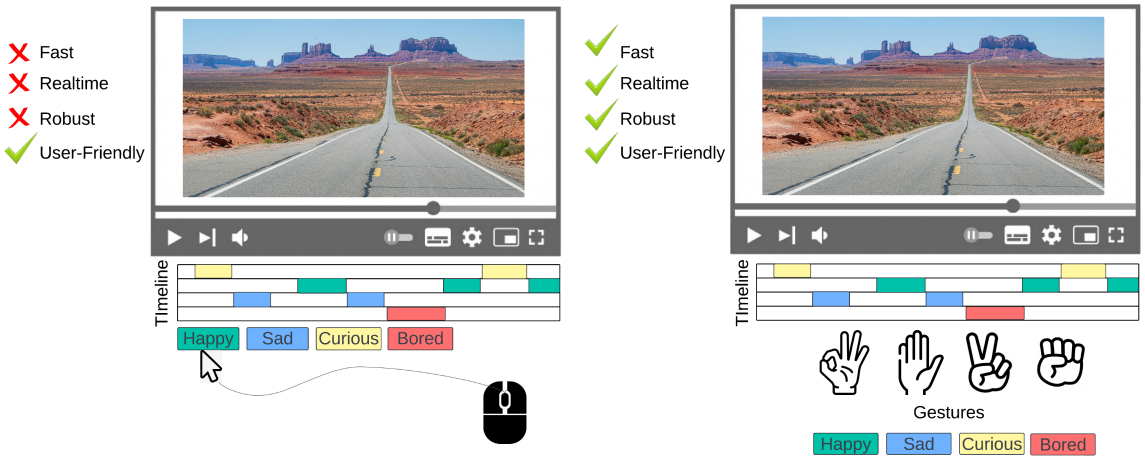


Fig. 1. A comparison between a traditional label annotation tool **(left)** and our HandyLabels system **(right)**. The traditional method requires manual inputs, such as mouse interactions, to label data, making it slow, inefficient, and lacking in real-time capabilities. In contrast, HandyLabels leverages real-time gesture-based annotations, significantly enhancing speed, robustness, and user experience. By enabling intuitive hand gestures for on-the-fly annotation, HandyLabels provides a more efficient and seamless approach to labeling, particularly suited for real-time applications.

The success of machine learning is deeply linked to the availability of high-quality training data, yet retrieving and manually labeling new data remains a time-consuming and error-prone process. Traditional annotation tools like LabelStudio often require post-processing, where users label data after recording. This cumbersome process leads to inefficiencies, particularly with large datasets. In this work, we introduce HandyLabels, a real-time annotation tool that leverages hand gesture recognition to map hand signs to labels during live recordings. The application allows users to customize gesture mappings through a web-based interface, enabling real-time annotations. To validate the usability of HandyLabels, a user study was conducted with over 50 participants. The results suggest that HandyLabels is preferred by the majority of participants compared to traditional annotation tools. To ensure the robustness of HandyLabels, we also evaluated several hand gesture recognition models on a custom dataset, with and without skeleton-based pre-processing. These models were evaluated under challenging conditions (various backgrounds, distances, and devices) to stress-test their robustness, with the goal of selecting the most reliable backbone for HandyLabels. Our experiments revealed that models incorporating skeletal data, particularly the transformer-based ViT model (the backbone of HandyLabels), exhibited the highest robustness.

CCS Concepts: • **Human-centered computing** → **User interface toolkits**; **Gestural input**; • **Computing methodologies** → *Tracking*.

Additional Key Words and Phrases: Hand gesture recognition, annotation tool, intelligent system, labeling.

1 INTRODUCTION

Data annotation is the most important factor in the development of learning-based systems and is at the root of the success of AI applications, such as ChatGPT, Stable Diffusion and object detection models [24, 26, 27, 39]. The importance of annotated datasets is highlighted in foundational works like BERT for language understanding by Devlin et al. [4], and the extensive survey on object detection by Zou et al. [39], while OpenAI's GPT-4 Technical Report emphasizes the reliance on large-scale data annotation for advancing AI capabilities [24]. Often referred to as the "new oil," data is the driving force behind introducing disruptive innovations with AI across various fields, such as autonomous driving, healthcare, and creative industries [3, 18, 19, 35, 39]. In autonomous driving, for instance, annotated datasets are critical for training object detection models that enable cars to identify pedestrians, traffic signs, and other vehicles [39]. Similarly, in healthcare, AI models rely on vast amounts of labeled medical images to detect diseases early, improving diagnostics and treatment outcomes [6, 19]. The impact of data is particularly evident in creative industries, where models like Stable Diffusion leverage extensive text image datasets to generate high-quality visual content, driving innovation in design and digital art [1, 20, 27]. These applications underscore the pivotal role of data annotation in pushing the boundaries of AI technologies across diverse sectors [8, 22, 36].

Yet, well-annotated data does not come for free and require a significant workforce in the first place [7, 11, 25, 31]. Despite the widespread availability of annotation tools, most solutions remain limited to post-processing workflows [2, 14, 32, 34]. Video data as an example, traditional methods require users to label the footage after it has been recorded, often involving tedious tasks such as pausing, labeling specific frames, and repeating the process multiple times. This approach is time-consuming and inefficient, particularly when dealing with large datasets. As the demand for high-quality labeled data grows, there is a pressing need for more intuitive, real-time tools that streamline the annotation process to improve speed and accuracy.

In the field of human-computer interaction, hand gestures offer a natural and intuitive method for interacting with software systems [16, 23]. Building upon this observation, we developed HandyLabels, a real-time gesture-based video annotation tool that simplifies the data labeling process. Unlike traditional tools, which rely on manual, post-event labeling, HandyLabels enables live annotations triggered by predefined hand gestures. This not only speeds up the annotation process but also makes it more efficient. HandyLabels also offers users the flexibility to create custom gestures that can be used to label events or emotions in real-time, making it adaptable to a wide range of scenarios. Figure 1 provides a visualization of the concept behind HandyLabels, demonstrating how users can effortlessly annotate data in real-time by simply waving hand gestures.

The usability of HandyLabels and ease of use was further validated through a comprehensive user study comparing it to a widely-used platform, Label Studio. The results of this study demonstrated that HandyLabels is not only faster and more intuitive but also requires significantly less effort from users, with most asked participants recommending it over traditional tools. The study's findings underscore the importance of real-time interaction and ease of use in data annotation tools, particularly in dynamic environments. As the demand for labeled data grows, tools like HandyLabels provide a practical solution by combining efficiency, accuracy, and user-friendly design. This makes HandyLabels well-suited for a variety of real-world applications where speed and accuracy are critical.

While the use of HandyLabels is intuitive and user-friendly, its impact extends across a wide range of real-world applications. For instance, imagine a group of users watching a movie and tracking their emotional reactions in real-time. In this scenario, assigning emotional labels would require participants to put in extra effort to communicate their feelings, as emotions are not always easily observable from external cues. In contrast, HandyLabels enables users to

define custom labels, such as “happy,” “sad,” or “excited,” and assign these labels through simple hand gestures as they are recorded. As the recording progresses, the annotations are logged automatically, generating a timeline of events that can be exported as a CSV file for further analysis (a feature provided by HandyLabels). This example showcases the versatility of HandyLabels in managing real-time annotation tasks, offering significant advantages over traditional post-processing methods.

To ensure high annotation accuracy, we rigorously tested several hand gesture recognition models, including ResNet [10], ResNeXt [37], MobileNet [13], and ViT [5], all trained on the HaGRID dataset [15], which comprises over 550,000 FullHD images covering 18 gesture categories across diverse environments. To further enhance recognition performance, we incorporated skeleton-based hand tracking through the MediaPipe library [9]. The skeletal data, extracted from 21 key points on the hand, significantly improved the accuracy of gesture recognition, enabling the models - especially the ViT-based approach - to consistently outperform those relying solely on raw hand images.

This extensive evaluation formed the foundation for selecting the most robust backbone model, ViT, for HandyLabels. Importantly, rather than aiming for perfect scores across all conditions, our primary goal was to identify the model that could perform most reliably under hard real-world constraints. To ensure robustness, we subjected the gesture recognition models to challenging scenarios, including varying backgrounds, devices, and distances. The final setup, particularly the integration of skeletal data, was the outcome of a rigorous evaluation designed to stress-test diverse model candidates in these varied environments and pinpoint the backbone model that could maintain high accuracy across all scenarios for HandyLabels (ViT).

To achieve this, we collected a comprehensive dataset that captures a wide range of variables, including diverse backgrounds, lighting conditions, devices, and distances (1m, 2m, and 3m), with contributions from 19 participants. The primary objective of this dataset is to stress-test recognition models and to simulate the kinds of real-world challenges a system would face, to ensure that the final chosen model, ViT, would be highly resilient and effective in live annotation applications like HandyLabels and future developments. As a result, our work not only introduces HandyLabels but also presents an automated annotation tool whose backbone model was rigorously tested and selected across a variety of real-world conditions. In summary, our contributions are as follows:

- C1 HandyLabels: A fast and user-friendly real-time annotation tool using a hand gesture recognition.
- C2 A dataset including varying conditions for testing the robustness of hand gesture recognition models.
- C3 Field-tested application robust under different backgrounds, hardware devices, and recording distances.

2 RELATED WORK

In this section, we examine the most influential annotation tools and evaluate their performance in handling real-time data annotation, particularly in comparison to HandyLabels, which was designed from the ground up to address these challenges. Moreover, we briefly review commonly used hand gesture recognition models.

2.1 Existing Annotation Systems

One of the most popular open-source annotation tools is Label Studio [32], known for its flexibility in handling a variety of data types, from text to images and videos. For video annotation specifically, Label Studio enables users to label frames after loading a media in the app. It features a timeline-based interface, which makes it easy to annotate segments over time, and it supports customizable labels to suit different projects. Although Label Studio is versatile and well-suited for projects that need manual or semi-automated annotation, it does not fully address the demands of

real-time interaction and immediate feedback. In contrast, HandyLabels focuses on live gesture recognition, allowing annotations to happen on-the-fly, which removes the need for post-event processing and improves efficiency.

ELAN [2] is another powerful tool that is widely used for multi-tier annotation tasks. Originally created for linguistic annotation, ELAN allows users to label several layers of information in a single video, making it ideal for tasks like gesture recognition that require detailed temporal segmentation. However, ELAN's strength lies in its timeline-based capabilities, which are more suited for post-recording analysis. It lacks the ability to provide real-time feedback, which is where HandyLabels excels by allowing users to label gestures during live sessions, thus enhancing both accuracy and workflow speed.

Similarly, CVAT [14], developed by Intel, is an open-source tool designed for video and image annotation, particularly for computer vision tasks. CVAT supports tracking and labeling objects across video frames and offers useful features like bounding box interpolation to minimize manual work. However, like ELAN, CVAT is built around a post-processing model, making it less ideal for real-time applications. HandyLabels addresses this limitation by focusing on gesture-triggered real-time annotations, allowing users to interact and label data during live recording sessions.

The VIA tool [34], developed by Oxford's Visual Geometry Group, is a lightweight, browser-based tool designed for straightforward manual annotation. While it does support video annotation, VIA is ideal for smaller projects where ease of use is a priority. Its minimal setup makes it efficient for basic tasks, but it falls short when it comes to handling more complex, real-time applications like deep learning-based gesture recognition. Unlike advanced systems that automate gesture detection, VIA requires manual input, making it less suited for large-scale, dynamic tasks that demand real-time feedback. HandyLabels, by contrast, combines real-time gesture recognition with machine learning to offer a robust solution that scales to diverse use cases, all while maintaining real-time performance.

Despite the aforementioned capabilities, most of these tools focus on traditional post-event annotation workflows, where data is labeled after the video has been recorded. This method can be cumbersome for users who need immediate feedback or are working in environments where real-time interaction is essential. HandyLabels fills this gap by enabling real-time annotation through hand gestures, which are captured and labeled instantly as the video is being recorded.

2.2 Hand Gesture Recognition

Hand gesture recognition has become a key technology for developing more natural and intuitive human-computer interactions [16, 23]. Over the years, various approaches have been proposed to tackle the challenge of recognizing gestures in real-time, particularly for tasks such as virtual reality interactions, sign language interpretation, gaming, robotic control, and touchless navigation systems [29, 30, 33, 38]. Many of these approaches focus on balancing accuracy, speed, and hardware simplicity to ensure seamless user experiences across different devices and environments [17, 21]. Recent advancements in deep learning have further improved the accuracy and real-time processing capabilities of gesture recognition systems, making them more flexible and widely applicable. In the following, we explore some of the key developments in gesture recognition and how they relate to real-world applications.

Herbaz et al. [12] developed a comprehensive hand gesture recognition system using the YOLO (You Only Look Once) object detection framework, focusing specifically on its application in real-time scenarios. The study explored various YOLO models, including YOLOv5, YOLOv6, and YOLOv8, to optimize the accuracy and speed of gesture recognition. Among the models tested, YOLOv8 demonstrated the highest accuracy, reaching 99.5% on the customized dataset. This work not only highlights the robustness and adaptability of YOLO models in complex environments but also underscores their potential for enhancing the interaction between humans and machines, particularly in real-time applications where speed and precision are critical. However, while YOLO excels in object detection, HandyLabels enhances gesture

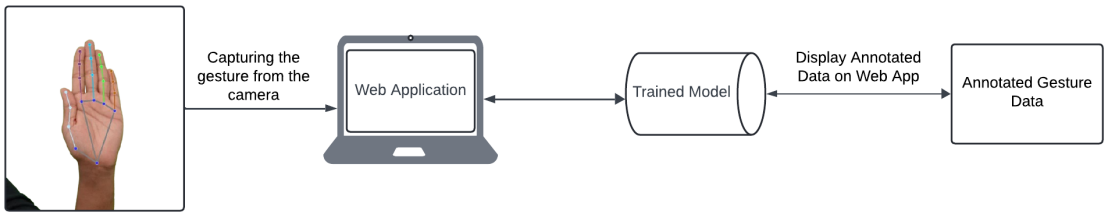


Fig. 2. Overview of the real-time gesture-based annotation system workflow. The process begins with capturing hand gestures through a camera, which are then processed by a web application. The captured gestures are sent to a trained machine learning model (skeleton-based ViT) that recognizes the gestures and returns annotated data. The annotated gestures are displayed on the web application in real-time, allowing for seamless and efficient data labeling.

recognition by combining machine learning models with skeleton data. This approach allows HandyLabels to achieve high accuracy in gesture recognition without relying solely on object detection models, making it more adaptable to different tasks.

S. and V. developed a state-of-the-art hand gesture recognition system by combining the Faster R-CNN model with the Inception V2 architecture [28]. Their approach was specifically designed to function effectively in real-time and unconstrained environments, where traditional systems often struggle. The system was tested under various challenging conditions, such as varying lighting and complex backgrounds, achieving an average precision of 0.991 and a prediction time of just 137 milliseconds. These results demonstrate the system's suitability for dynamic gesture recognition tasks, particularly in scenarios where both accuracy and response time are critical. However, due to the complexity of these models, their use in real-time applications is limited. HandyLabels, on the other hand, simplifies the process by focusing on real-time recognition, combining skeleton-based pre-processing to improve both accuracy and processing speed.

Advancements in hand gesture recognition have made significant strides, with systems like YOLO and MediaPipe pushing the boundaries of what is possible in real-time interaction. However, many of these systems either require specialized hardware or have limitations in terms of speed or complexity. HandyLabels addresses these gaps by offering a real-time, accessible solution that leverages skeleton-based pre-processing to improve accuracy, making it an ideal tool for diverse, dynamic environments where immediate feedback is critical.

3 HANDYLABELS

HandyLabels is a real-time hand gesture recognition tool designed to provide a seamless and intuitive user experience for annotating data on-the-fly. By leveraging hand gesture recognition, it simplifies the typically cumbersome task of data annotation, allowing users to quickly label events, emotions, or other interactions in real-time. This Chapter details the architecture of HandyLabels and discusses how we designed our experiments to find the most robust backbone model capable of handling diverse scenarios.

3.1 Architecture

The architecture of HandyLabels has been optimized for ease of use and high accuracy as well as fast runtime in real-time applications. The system is designed to be accessible to both novice and experienced users, ensuring that the interface is simple yet powerful, as depicted in Figure 4 The key components include:

Fig. 3. The five hand gestures used in our experiment to evaluate different backbone model candidates: Fist, Ok, Stop, Two-Up, and Peace (signs are displayed in the same order from left to right).

3.1.1 Custom Label Setting. After launching the application, users can define custom labels for up to five gestures, adapting the system to specific use cases, such as emotional annotation during a movie or real-time feedback in a presentation. These labels can be easily assigned to predefined hand gestures like "Fist," "Ok," "Peace," "Two-Up," and "Stop," or any custom gesture (as shown in Figure 1 and Figure 3).

3.1.2 Skeleton-Based Gesture Tracking. To improve both gesture recognition accuracy and visual feedback, HandyLabels leverages a skeleton-based hand tracking system using the MediaPipe library [9]. This method tracks 21 key points of the hand, allowing for precise gesture detection while providing clear, explainable visual cues. For instance, when a user performs a gesture linked to an emotion like "excited" while watching a movie, the system instantly recognizes and logs the gesture in real-time, ensuring immediate, accurate annotation with minimal lag. Additionally, it visually highlights the detected hand joints and finger positions. As we will demonstrate in our experiments, this skeleton-based processing also significantly enhances recognition accuracy compared to models relying solely on raw hand images.

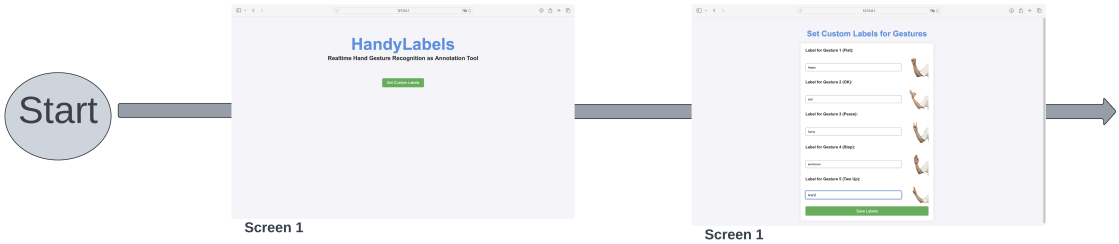
3.1.3 Back-End Processing. The front-end of HandyLabels communicates with a back-end server built on Flask, where the computational heavy-lifting happens. Frames from the user's camera are processed in real-time to detect gestures, with models trained on the HaGRID dataset [15]. The skeleton-based structure ensures high efficiency and accuracy, allowing the system to focus on essential hand movements even in visually complex environments.

3.1.4 Real-Time Data Annotation. One of the standout features of HandyLabels is its ability to annotate ongoing activities as they happen. The system logs recognized gestures along with corresponding timestamps.

3.1.5 Data Storage and Visualization. After gestures are performed and recognized, users can save the recorded gestures and their corresponding timestamps. This feature is particularly useful for post-event analysis, such as analysing the trajectory of emotions during a movie. The data is saved in a CSV file, which can be downloaded. After storage, HandyLabels allows users to visualize the collected data through a breakdown of gestures, including timelines and pie charts that represent the frequency and distribution of gestures during the recording session. This provides valuable insight, whether the system is used for emotional analysis, feedback tracking, or other real-time applications.

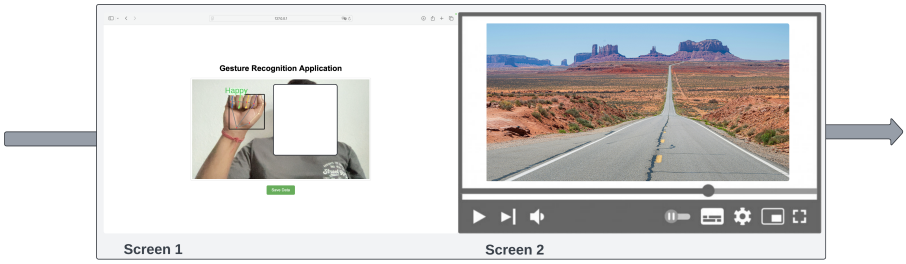
3.2 Perception & Usability

To assess the perception and usability of HandyLabels, we conducted a comprehensive user study comparing it with the established annotation tool Label Studio. The primary goals of the study were to evaluate the intuitiveness, ease of use, and suitability of both tools for annotation tasks.

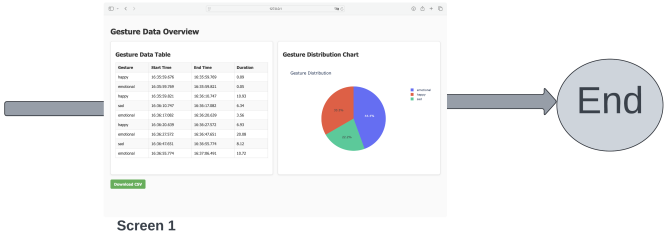


(1) HandyLabels Landing Page

(2) Custom Labels for Gestures



(3) Real-time Video Annotation in HandyLabels



(4) Gesture Data Overview

Fig. 4. This diagram illustrates the complete workflow of the HandyLabels application. (1) The process begins with the landing page (Screen 1), where users can set custom labels for hand gestures. (2) Once users proceed to customize their labels, they are presented with an interface to define unique gestures such as "happy" or "sad". (3) After labels are set, users can start the real-time gesture recognition process, where gestures performed on Screen 1 are annotated as the video plays on Screen 2. (4) The workflow concludes with a comprehensive overview of the recorded data, including a gesture data table and a distribution chart, allowing users to visualize and export their results.

3.2.1 Study Setup. The study involved 46 participants from diverse backgrounds, including various nationalities, genders, and occupations (with the majority being students). The age ranges from 19 to 46, with an average age of 27. Each participant was presented with videos demonstrating the labeling process using both HandyLabels and LabelStudio.

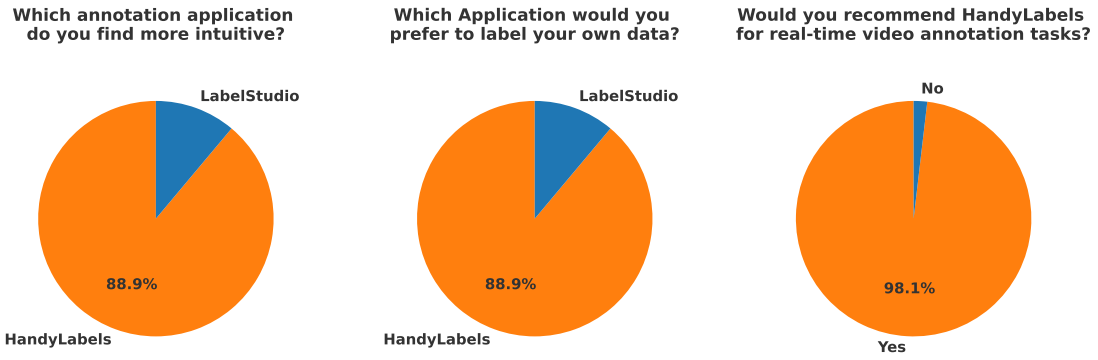


Fig. 5. User preferences for HandyLabels compared to Label Studio. The first pie chart illustrates that the vast majority of users found HandyLabels more intuitive than Label Studio, with only a small percentage favoring Label Studio. The second chart reflects user preferences for data annotation, with 88.9% of participants indicating they would rather use HandyLabels for annotation tasks. The third chart shows that the majority would recommend HandyLabels, highlighting a strong overall preference for HandyLabels.

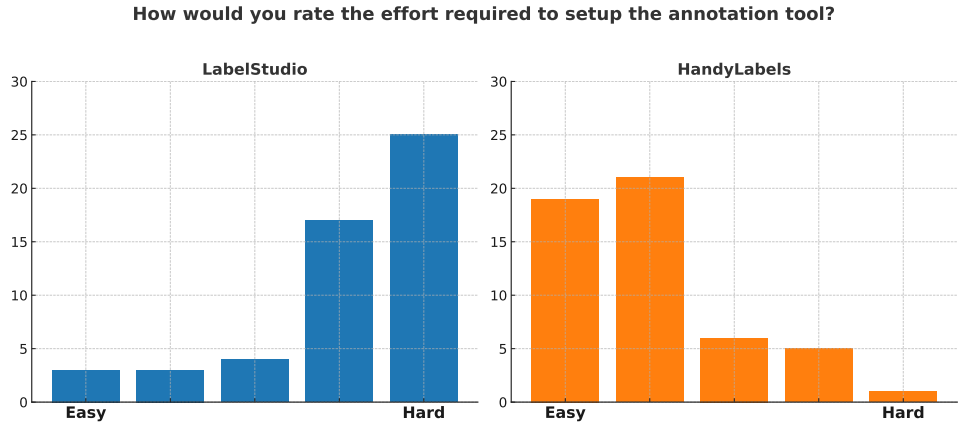


Fig. 6. Comparison of user ratings for setup effort between HandyLabels and Label Studio. The left bar chart shows that most participants found Label Studio more challenging to setup, with the majority giving it a difficulty rating of 4 or 5 (on a scale of 1 to 5, where 5 is the hardest). In contrast, the right bar chart demonstrates that HandyLabels is much easier to setup, with most participants rating the effort as 1 or 2, indicating its user-friendly setup process.

After watching the videos, participants answered a series of questions assessing their experience with each tool. This included how intuitive they found each tool, the amount of effort required for setup, and which tool they preferred for labeling tasks.

3.2.2 Study Results. The findings were conclusive and overwhelmingly in favor of HandyLabels. The majority of participants expressed a preference for HandyLabels when asked which tool they would use for annotating their own data. Moreover, HandyLabels was rated as the more intuitive tool, with users noting its real-time gesture recognition capabilities and interactive, user-friendly interface. These features were particularly valued in dynamic environments

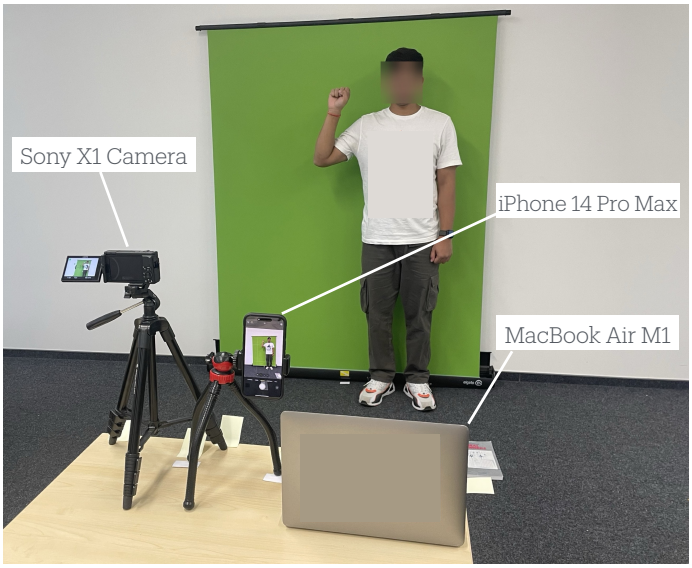


Fig. 7. Experimental setup for evaluating hand gesture recognition models: A participant is performing gestures in front of a green screen at three different distances (one meter, two meters, and three meters) from the camera with different devices (Sony X1 Camera, iPhone 14 Pro Max, MacBook Air M1).

where speed and responsiveness are crucial. Also, most participants said they would recommend HandyLabels for real-time video annotation tasks, reinforcing its perceived effectiveness and ease of use. All these three results are visualized in Figure 5. Finally, participants found HandyLabels significantly easier to set up, rating it 2.07 on a 5-point scale (1 = Easy, 5 = Hard), compared to 4.22 for Label Studio, which was perceived as more manual and labor-intensive (as shown in Figure 6). In summary, these results demonstrate that HandyLabels not only outshines Label Studio in terms of usability and setup effort but is also highly favored by users for annotation tasks.

3.3 Evaluating Robustness of HandyLabels

To rigorously assess the robustness and reliability of possible backbone models for HandyLabels, we conducted a controlled data collection study designed to simulate diverse real-world conditions. Participants were instructed to perform five predefined hand gestures - Fist, Ok, Peace, Two-Up, and Stop (as illustrated in Figure 3) - under varying conditions to evaluate a model's performance across multiple scenarios. For this purpose, we selected three axis under which the backbone models were tested: Background complexity, device variability, and distance variations. In the following, we outline how we achieved such an evaluation.

3.3.1 Experimental Setup. Gestures were performed at three distances: one meter, two meters, and three meters from the camera. To ensure robustness across different hardware, we captured gestures using a professional-grade camera, an iPhone 14 Pro Max, and a MacBook Air (see appendix for details). This multi-device approach simulated real-world hardware variability, ensuring HandyLabels performs well on both high-end and consumer-grade equipment (Figure 7 illustrates the setup). The data collection took place in a closed room with consistent lighting to minimize external light influences. A green screen ensured uniformity and facilitated future testing in simulated backgrounds, allowing us to evaluate system performance in varied real-world environments.

3.3.2 Real-World Backgrounds. Following the initial data collection, the green screen background was digitally replaced with various real-world environments to further validate the robustness of possible backbone model candidates for HandyLabels. This allowed us to evaluate the model's ability to maintain high recognition accuracy even in the presence of distractions, such as varying lighting, cluttered backgrounds, and different levels of visual noise.

3.3.3 Privacy and Anonymization. To protect participants' privacy, a custom anonymization script was applied to automatically remove facial features from all captured images. This process adhered to stringent data privacy protocols, ensuring that no Personally Identifiable Information (PII) was stored. Additionally, demographic information, such as age and gender, was collected in an anonymized format. The participants were informed that the collected data will be used exclusively for research purposes and may be presented at conferences or published in academic journals.

3.3.4 Participant Demographics and Ethical Guidelines. Nineteen participants, aged between 24 and 32 years, of diverse genders and ethnicities, were recruited from Germany to participate in the study. Participation was entirely voluntary, and all participants were thoroughly briefed on the study's objectives and their rights, including the option to withdraw at any time without repercussions. Informed consent was obtained from each participant in line with institutional ethical guidelines, and no compensation was provided for participation.

4 EXPERIMENTS

We selected several deep learning models as backbone candidates for HandyLabels: ResNet18 [10], MobileNetV3 (both small and large versions) [13], ResNeXt50 [37], ResNet50 [10], and ViT-B16 [5]. These models were chosen for their strong performance in image classification tasks and their adaptability to gesture recognition scenarios. ResNet models were included due to their deep architecture, which excels at learning complex visual features, while MobileNetV3 was selected for its efficiency and lightweight design, making it ideal for real-time applications. ViT-B16, a transformer-based model, was also tested to evaluate the potential of transformer architectures in hand gesture recognition tasks.

Training Details: We used standardized hyperparameters to ensure consistency across all models, including a learning rate of 10^{-3}, a batch size of 64, and cross-entropy loss as the objective function. The models were trained on the HaGRID dataset for up to 100 epochs, with early stopping applied if validation loss plateaued. These consistent settings allowed for a fair comparison of the performance between models.

Evaluation Details: We used the F1 score, which balances precision and recall, offering a more comprehensive view of model performance by accounting for both false positives and false negatives. While the F1 scores may appear lower than expected, it is important to note that these tests were conducted in particularly challenging environments, including diverse backgrounds and various distances, where many real-time recognition systems would struggle. The goal was to identify which models maintain high reliability under these conditions rather than optimizing for ideal setups and finally explaining the rationale behind choosing ViT as backbone for HandyLabels.

Skeleton-Data: Moreover, we tested two pre-processing configurations in all our following experiments:

- **Raw Hand Images**: Images from the dataset were fed directly into the model without any pre-processing.
- **Skeleton-Based Overlay**: We extracted skeletal structures using the MediaPipe library, which detects 21 key hand-knuckle coordinates. This skeleton data was overlaid on the original images, providing additional input features for the models.

Table 1. Comparison of F1 scores across different backgrounds (B1 to B5) for models with and without skeleton-based pre-processing. The backgrounds range from simple (B1: grey background) to more complex scenes (B5). Models utilizing skeleton-based pre-processing consistently outperform those without it, particularly in more complex environments (e.g., B5). The ViT-B16 model achieves the highest F1 scores with particularly strong improvements seen in ResNet50 and ViT-B16 models, demonstrating superior performance. Best performances are marked in dark blue, second best in light blue.

	Without Skeleton					**With Skeleton**				
Model	B1	B2	B3	B4	B5	B1	B2	B3	B4	B5
ResNet18	0.1997	0.1048	0.1115	0.1283	0.0860	0.1549	0.2724	0.2701	0.2332	0.2281
MobileNetV3 (small)	0.0858	0.1949	0.1451	0.0665	0.1067	0.1478	0.1670	0.2846	0.0909	0.1259
MobileNetV3 (large)	0.1311	0.1687	0.1409	0.0752	0.1176	0.2704	0.1549	0.1666	0.1472	0.1702
ResNeXt50	0.2103	0.1435	0.0985	0.1213	0.0926	0.1469	0.1488	0.1303	0.1425	0.0918
ResNet50	0.1927	0.1024	0.0823	0.1164	0.0966	0.2139	0.1895	0.2300	0.1319	0.1781
ViT-B16	0.2093	0.1958	0.1336	0.1737	0.2219	0.3693	0.2996	0.2708	0.3122	0.3293

4.1 Influence of Different Backgrounds

To better understand how background complexity affects model performance, we tested the models both with and without skeleton-based pre-processing across five distinct backgrounds (B1-B5). Table 1 shows the results of this comparison. B1, a simple grey background, served as a baseline, while B2 through B5 introduced more complex and varied environments, ranging from natural scenes to urban landscapes. These tests were conducted using an iPhone 14 Pro Max, with backgrounds altered using green-screening techniques. Figure 8 offers a visual overview of these different backgrounds alongside the hand gestures.

Results without Skeleton: When tested without skeleton-based pre-processing, the models struggled significantly as the complexity of the background increased. For example, ResNet18, a relatively simple model, achieved an F1 score of 0.1997 on B1 (grey background), but its performance dropped sharply to 0.0860 on B5, the most complex background. This pattern was common across most models, suggesting that without skeleton data, the models found it difficult to isolate hand gestures from the visual noise and distractions in more intricate environments. Similarly, ResNet50 - a more complex model than ResNet18 - also faced challenges in handling complex backgrounds. While it achieved an F1 score of 0.1927 in B1, its performance dropped to 0.0966 in B5, further highlighting how challenging visually rich environments can be for gesture recognition models when raw images are used without skeletal guidance.

Results with Skeleton: In contrast, when skeleton-based pre-processing was applied, the models showed a much more consistent performance across all backgrounds. This approach allowed the models to focus on the structure of the hand, effectively filtering out the background complexities. For instance, ResNet18 with skeleton-based pre-processing maintained an F1 score of 0.2724 in B1 and performed considerably better than its non-skeleton counterpart in B5, with an F1 score of 0.2281. ResNet50, which also performed well with skeleton pre-processing, showed a clear improvement, achieving an F1 score of 0.2139 in B1 and 0.1781 in B5. The skeleton data helped the model maintain stronger performance in complex backgrounds, making it more adaptable to challenging environments.

Summary: Among all the tested models, ViT-B16 - a transformer-based architecture - was the most robust. With skeleton pre-processing, ViT-B16 achieved the highest overall scores, including an F1 score of 0.3693 in B1 and a still impressive 0.3293 in B5, showing that even in highly complex environments, this model performed reliably. ViT-B16's ability to consistently deliver high accuracy across diverse backgrounds underscores the advantage of combining transformer architectures with skeleton-based features, especially in scenarios where environmental factors vary widely. Overall, the inclusion of skeleton-based pre-processing dramatically enhanced model performance, particularly in

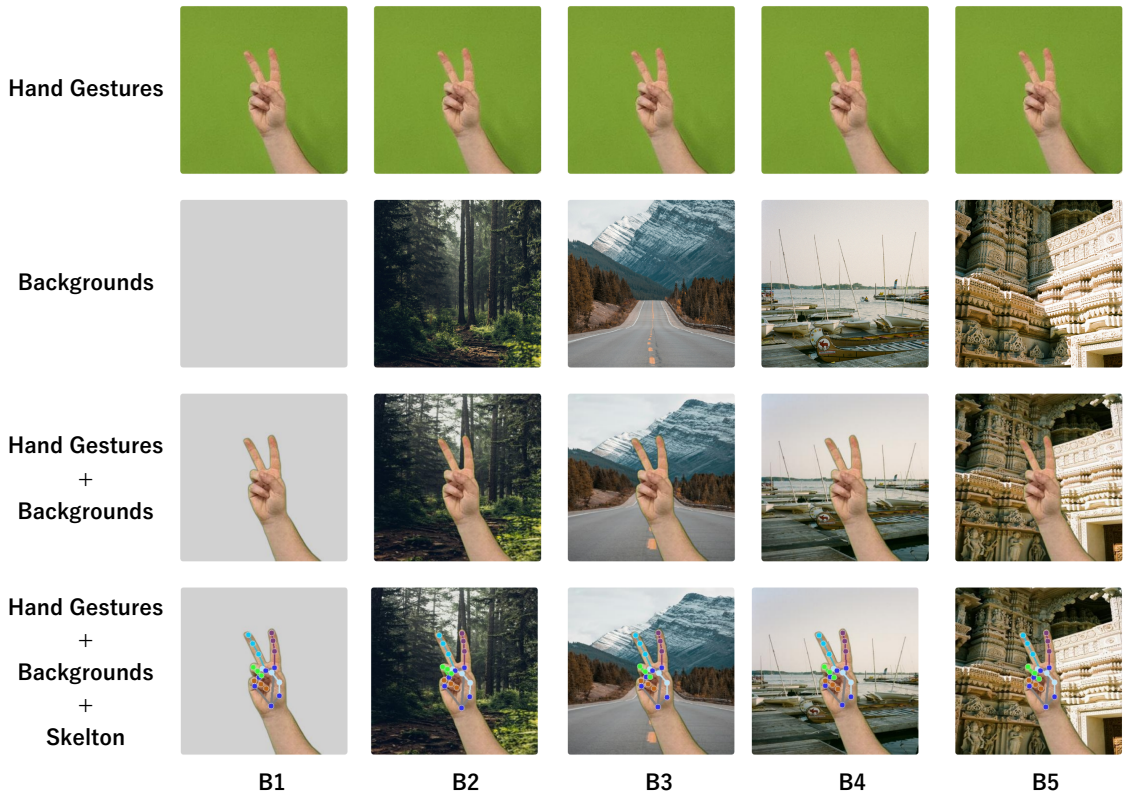


Fig. 8. The first row shows the "peace" hand gesture against a green screen, followed by the backgrounds alone in the second row. The third row combines the gesture with different backgrounds, while the fourth row applies skeleton-based pre-processing, mapping hand landmarks to improve gesture recognition across varied environments.

Table 2. Comparison of F1 scores across different devices (MacBook M1, Sony X1, iPhone 14 Pro Max) for models with and without skeleton-based pre-processing. The skeleton-based approach generally improves performance across all devices, with particularly strong improvements seen in ResNet50 and ViT-B16 models. Best performances are marked in dark blue, second best in light blue.

	Without Skeleton			**With Skeleton**		
Model	MacBook	Sony X1	iPhone	MacBook	Sony X1	iPhone
ResNet18	0.2290	0.1351	0.1997	0.2224	0.1941	0.1549
MobileNetV3 (small)	0.1101	0.0952	0.0858	0.2486	0.1754	0.1478
MobileNetV3 (large)	0.1986	0.1475	0.1311	0.2464	0.2478	0.2704
ResNeXt50	0.2546	0.1863	0.2103	0.2338	0.1593	0.1469
ResNet50	0.2256	0.1795	0.1927	0.3468	0.2274	0.2139
ViT-B16	0.1941	0.2070	0.2093	0.3669	0.3322	0.3693

complex visual environments. Models like ResNet50 and ViT-B16 benefitted most from this approach, showing that skeletal data helps improve recognition accuracy when dealing with intricate backgrounds. These results suggest that incorporating skeletal data into gesture recognition systems can significantly improve the robustness of HandyLabels, making it more effective in real-world applications where background complexity can vary.

4.2 Influence of Different Devices

To evaluate how different devices impact gesture recognition performance, we tested three distinct devices: MacBook Air M1, Sony X1, and iPhone 14 Pro Max. The tests were conducted from a fixed distance of three meters for all devices. The results of these tests are summarized in Table 2.

Results without Skeleton: Without skeleton-based pre-processing, models showed varying performance across devices. For instance, ResNeXt50 achieved its best result on the MacBook Air, with an F1 score of 0.2546, slightly outperforming its result on the iPhone (0.2103). However, ResNet18 performed consistently across all devices, with its highest score on the MacBook Air (0.2290) and slightly lower on the iPhone (0.1997). MobileNetV3 (small), on the other hand, struggled on all devices, with the lowest score recorded on the iPhone (0.0858).

Results with Skeleton: When skeleton-based pre-processing was applied, performance across all devices improved significantly, with the most notable gains seen on the iPhone. ResNet50, which performed well without pre-processing, saw a considerable boost in its score on the MacBook Air, increasing from 0.2256 to 0.3468. Similarly, ViT-B16 showed the highest overall scores with skeleton pre-processing, achieving an F1 score of 0.3693 on the iPhone, and maintaining strong performance across the MacBook Air (0.3669) and Sony X1 (0.3322). The MobileNetV3 (large) model also saw significant improvement, with its score on the iPhone rising from 0.1311 without skeletons to 0.2704 with skeleton data. This highlights the effectiveness of skeleton-based pre-processing, particularly for models with lightweight architectures.

Overall Device Performance: While all devices showed improved performance with skeleton-based pre-processing, the iPhone 14 Pro Max consistently led in terms of F1 scores, likely due to its superior imaging hardware (48 MP sensor, optical image stabilization, and sensor-shift technology). However, the MacBook Air M1 also performed well, especially with models like ResNet50 and ViT-B16, demonstrating that high-quality hardware combined with pre-processing techniques can lead to robust gesture recognition even on consumer-grade devices. The Sony X1 generally has the worst performance across all devices, likely due to its older camera technology, which may not have captured gestures as clearly from a distance.

Summary: These findings suggest that a combination of high-quality hardware and skeleton-based pre-processing is essential for achieving reliable gesture recognition. In particular, HandyLabels excels on devices like the iPhone 14 Pro Max, where advanced camera capabilities enhance the model's ability to accurately capture and recognize gestures. Nonetheless, the system remains adaptable to a range of devices, ensuring strong performance across varying hardware configurations, with ViT+skeleton again achieving the best results.

4.3 Influence of Different Distances

In addition to testing across various backgrounds and devices, we also evaluated the models' performance at three distinct distances: 1m, 2m, and 3m, using the Sony X1 camera as the capturing device.

Results without Skeleton: As expected, model performance declined as the distance from the camera increased when skeleton-based pre-processing was not applied. ResNet18, for example, showed a sharp drop in accuracy, achieving an F1 score of 0.2012 at one meter, which dropped significantly to 0.1351 at three meters. This trend was consistent across most models, where the increased distance introduced additional challenges, such as reduced image clarity and gesture size, making it more difficult for the models to accurately detect and classify hand gestures. Similarly, MobileNetV3 (large) scored 0.1527 at one meter, with a slight decrease to 0.1475 at three meters, further highlighting the challenges of maintaining recognition accuracy as distance increases without additional pre-processing support.

Table 3. Comparison of F1 scores across different distances (1m, 2m, 3m) for models with and without skeleton-based pre-processing. Skeleton pre-processing consistently improves performance across most distances, especially for models like ViT-B16 and ResNet50. Best performances are marked in dark blue, second best in light blue.

	Without Skeleton			**With Skeleton**		
Model	1m	2m	3m	1m	2m	3m
ResNet18	0.2012	0.2511	0.1351	0.1272	0.0945	0.1941
MobileNetV3 (small)	0.0962	0.0666	0.0952	0.1572	0.1429	0.1754
MobileNetV3 (large)	0.1527	0.1179	0.1475	0.1576	0.1669	0.2478
ResNeXt50	0.2287	0.2783	0.1863	0.0967	0.0946	0.1593
ResNet50	0.1803	0.2345	0.1795	0.1807	0.1175	0.2274
ViT-B16	0.2306	0.2104	0.2070	0.3000	0.3475	0.3322

Results with Skeleton: The use of skeleton-based pre-processing significantly improved model performance across all distances. For instance, ResNet18, which had seen a marked decline in accuracy without pre-processing, managed to maintain more consistent performance, with an F1 score of 0.1941 at thre meters after pre-processing. This improvement suggests that skeleton-based pre-processing helps the models focus on essential hand movements, mitigating the difficulties posed by increased distance. A similar improvement was observed with MobileNetV3 (large), where the F1 score increased from 0.1475 to 0.2478 at three meters when skeleton-based pre-processing was applied. This demonstrates the ability of skeleton data to filter out irrelevant background information and enhance gesture detection at greater distances.

Standout Performer – ViT-B16: Among all tested models, ViT-B16 consistently outperformed others across varying distances. Even without skeleton-based pre-processing, ViT-B16 delivered strong results, achieving an F1 score of 0.2070 at three meters. However, with skeleton-based pre-processing, its performance improved substantially, reaching 0.3322 at three meters. The transformer-based architecture of ViT-B16, when combined with skeleton data, proved highly effective in adapting to distance variability, maintaining accuracy even at greater distances.

Notable Observation – ResNeXt50: Interestingly, ResNeXt50 exhibited better performance without skeleton-based pre-processing across all distances, achieving its highest F1 score of 0.2783 at two meters. This suggests that certain models, particularly those with deeper architectures like ResNeXt50, might be less reliant on skeleton data at shorter distances but still benefit from pre-processing at longer distances. The model's deep architecture could potentially allow it to capture more detailed features, reducing its dependence on pre-processing for short-distance recognition.

Summary: The results from these experiments emphasize the importance of skeleton-based pre-processing in maintaining accuracy as the distance between the camera and the subject increases. While models generally performed better at closer distances, skeleton-based pre-processing helped reduce the performance degradation observed at greater distances. Models like ViT-B16 and MobileNetV3 (large) were particularly effective at handling distance variability, demonstrating the benefits of skeleton data in ensuring reliable gesture recognition across a wide range of distances. Given its ability to improve model robustness in distance-varying scenarios, HandyLabels incorporates skeleton-based pre-processing as a core feature, ensuring consistent performance even when users are positioned at different distances from the camera. This makes the system particularly suitable for real-time annotation tasks, where camera distance may not always be controlled, further enhancing the tool's adaptability and reliability.

4.4 Overall Results

Our experiments demonstrate that skeleton-based pre-processing significantly improves hand gesture recognition accuracy across diverse backgrounds, devices, and distances. The incorporation of skeletal data, using the MediaPipe library, was especially helpful in visually complex environments, at greater distances, and with devices featuring less advanced camera technology. With varying backgrounds, models like ViT-B16 and ResNet18 showed marked improvements with skeleton data, handling complex scenes more effectively. For distances, skeleton pre-processing mitigated performance drops at longer ranges, with models like ViT-B16 and MobileNetV3 (large) maintaining accuracy even at three meters, compared to notable declines without pre-processing. In terms of devices, the iPhone 14 Pro Max delivered the highest scores, particularly when paired with skeleton data. These findings align with previous research on skeleton-based data's effectiveness in real-world, dynamic conditions [15].

For HandyLabels, we chose ViT-B16 as the primary model due to its consistent and superior performance across various test scenarios, including complex backgrounds, different devices, and varying distances. ViT-B16, a transformer-based model, demonstrated robust accuracy, particularly when combined with skeleton-based pre-processing. Its ability to capture long-range dependencies and global context from images made it highly effective in recognizing hand gestures even in challenging real-world conditions. This combination of flexibility and accuracy makes ViT-B16 ideal for HandyLabels, where reliable and adaptable gesture recognition is critical for real-time annotation tasks.

The user study results strongly support our findings. In a head-to-head comparison with Label Studio, 46 participants from various backgrounds and professions overwhelmingly favored HandyLabels. The study emphasized HandyLabels's real-time gesture recognition and its intuitive, user-friendly design, with 87% of participants recommending it. Notably, users found HandyLabels much easier to set up and use, scoring it at 2.07 for effort compared to Label Studio's 4.22 (on a scale where 1 is "Easy" and 5 is "Hard"). Furthermore, 40 out of 46 participants expressed a preference for using HandyLabels for their own data labeling needs. These results highlight HandyLabels 's effectiveness in real-world settings, showcasing its superior speed, simplicity, and overall user satisfaction.

5 DISCUSSION & FUTURE WORK

In this section, we want to discuss and summarizes the key insights from this work.

Skeleton-Based Pre-Processing Enhances Gesture Recognition: Our experiments revealed that integrating skeleton-based pre-processing significantly improves gesture recognition accuracy across a variety of scenarios. By focusing on 21 key points of the hand, we were able to improve model robustness, especially in complex environments.

High Adaptability: One valuable outcome is the adaptability of HandyLabels to real-world scenarios. By testing our backbone model ViT under diverse conditions, we ensured that the system can perform reliably in practical, dynamic environments. The real-time nature of our system, coupled with customizable hand gestures, positions HandyLabels as a versatile tool for data annotation across domains. Whether it is emotional tracking during a movie, real-time feedback in presentations, or annotating live events, HandyLabels can handle a wide variety of real-world applications efficiently.

Hardware Quality Affects Performance, but Skeleton Data Mitigates the Influence: Our tests across different devices revealed a clear relationship between hardware quality and model performance. The iPhone 14 Pro Max, with its advanced camera capabilities, consistently achieved higher scores, highlighting the impact of high-resolution imaging on gesture recognition. However, we observed that skeleton-based pre-processing helped offset the performance differences between devices. Even with lower-end devices like the MacBook Air M1, models using skeleton-based pre-processing performed comparably well, ensuring high accuracy and reliability across different hardware setups.

ViT as the Optimal Model for Robustness: Our evaluation identified ViT as the most robust model for HandyLabels, consistently outperforming other architectures across all test scenarios. The transformer-based architecture of ViT excels at capturing global dependencies within images, making it particularly effective when combined with skeleton data. This robustness makes ViT well-suited for real-time annotation tasks, ensuring high accuracy even under challenging real-world conditions. However, with our custom dataset that tests recognition models under hard conditions, our application is also adaptable to future advancements in gesture recognition.

Limitations and Opportunities for Future Work: While HandyLabels demonstrated strong performance, there are still several opportunities for future enhancements. Expanding the range of gestures, including multi-gesture recognition, would allow for broader applications and improve scalability. Additionally, further optimizing the tool for low-latency, high-speed environments - such as sports or live performances - could enhance its versatility. To this end, building a custom dataset that includes more gestures and diverse scenarios will be crucial for continued improvement.

Potential for Broader Application Domains: With its real-time capabilities and customizable gesture recognition, the concept of HandyLabels has the potential to be applied in a wide range of industries beyond data annotation. The system could be used in assistive technologies, education, entertainment, and human-computer interaction, offering intuitive control mechanisms through gesture-based inputs. By refining its real-time object tracking and extending its detection techniques, HandyLabels could become a central tool for gesture-based interfaces in numerous domains.

6 CONCLUSION

In this paper, we introduced HandyLabels, a real-time hand gesture recognition tool designed to ease the data annotation process. By leveraging hand gestures for live video annotation, HandyLabels eliminates the inefficiencies associated with traditional post-processed methods, providing a fast, accurate, and intuitive solution for real-time applications. Our system, which integrates a transformer-based model (ViT-B16) and incorporates skeleton-based preprocessing, was rigorously tested across a wide range of conditions, including diverse backgrounds, devices, and distances, to ensure robustness and adaptability in real-world scenarios. The system's flexibility ensures improved performance across all hardware setups tested, with every device showing significant gains when skeleton-based pre-processing is applied.

The significance of HandyLabels was further reinforced through a comprehensive user study, where it was compared to Label Studio. The users clearly favored HandyLabels and were recommending it due to its real-time gesture recognition capabilities and ease of use. Participants consistently highlighted its lower effort requirements and faster, more intuitive interface. These findings demonstrate that HandyLabels is not only technically superior but also preferred by users for its efficiency, usability, and practicality in real-world annotation tasks.

ACKNOWLEDGMENTS

TODO: Write acknowledgement.

REFERENCES

[1] Omer Bar-Tal, Lior Yariv, Yaron Lipman, and Tali Dekel. 2023. Multidiffusion: Fusing diffusion paths for controlled image generation. (2023).

[2] Hennie Brugman and Albert Russel. 2004. Annotating Multi-media/Multi-modal Resources with ELAN. In *Proceedings of the Fourth International Conference on Language Resources and Evaluation (LREC'04)*, Maria Teresa Lino, Maria Francisca Xavier, Fátima Ferreira, Rute Costa, and Raquel Silva (Eds.). European Language Resources Association (ELRA), Lisbon, Portugal. http://www.lrec-conf.org/proceedings/lrec2004/pdf/480.pdf

[3] Bálint Csanády, Lajos Muzsai, Péter Vedres, Zoltán Nádasdy, and András Lukács. 2024. LlamBERT: Large-scale low-cost data annotation in NLP. *arXiv preprint arXiv:2403.15938* (2024).

[4] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *NAACL-HLT* (2019).

[5] Alexey Dosovitskiy. 2020. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929* (2020).

[6] Christoffer Edlund, Timothy R Jackson, Nabeel Khalid, Nicola Bevan, Timothy Dale, Andreas Dengel, Sheraz Ahmed, Johan Trygg, and Rickard Sjögren. 2021. LIVECell—A large-scale dataset for label-free live cell segmentation. *Nature methods* 18, 9 (2021), 1038–1045.

[7] Teodor Fredriksson, David Issa Mattos, Jan Bosch, and Helena Holmström Olsson. 2020. Data labeling: An empirical investigation into industrial challenges and mitigation strategies. In *International Conference on Product-Focused Software Process Improvement*. Springer, 202–216.

[8] Stanislav Frolov, Brian B Moser, and Andreas Dengel. 2024. SpotDiffusion: A Fast Approach For Seamless Panorama Generation Over Time. *arXiv preprint arXiv:2407.15507* (2024).

[9] Google. 2019. MediaPipe: A Framework for Building Perception Pipelines. https://mediapipe.dev/.

[10] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 770–778.

[11] Xingwei He, Zhenghao Lin, Yeyun Gong, Alex Jin, Hang Zhang, Chen Lin, Jian Jiao, Siu Ming Yiu, Nan Duan, Weizhu Chen, et al. 2023. Annollm: Making large language models to be better crowdsourced annotators. *arXiv preprint arXiv:2303.16854* (2023).

[12] Nourdine Herbaz, Hassan El Idrissi, and Abdelmajid Badri. 2023. Deep Learning Empowered Hand Gesture Recognition: using YOLO Techniques. In *2023 14th International Conference on Intelligent Systems: Theories and Applications (SITA)*. 1–7. https://doi.org/10.1109/SITA60746.2023.10373734

[13] Andrew G Howard. 2017. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861* (2017).

[14] Intel. 2021. CVAT: Computer Vision Annotation Tool. https://github.com/openvinotoolkit/cvat.

[15] Alexander Kapitanov, Karina Kvanchiani, Alexander Nagaev, Roman Kraynov, and Andrei Makhliarchuk. 2024. HaGRID – HAnd Gesture Recognition Image Dataset. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*. 4572–4581.

[16] Unseok Lee and Jiro Tanaka. 2013. Finger identification and hand gesture recognition techniques for natural user interface. In *Proceedings of the 11th Asia Pacific Conference on Computer Human Interaction* (Bangalore, India) *(APCHI '13)*. Association for Computing Machinery, New York, NY, USA, 274–279. https://doi.org/10.1145/2525194.2525296

[17] Yifan Li, Yukun Wen, Shibin Qiu, and Anfeng Hao. 2019. Deep learning based hand gesture recognition in virtual reality applications. *IEEE Access* 7 (2019), 131019–131029.

[18] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. 2014. Microsoft coco: Common objects in context. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*. Springer, 740–755.

[19] Geert Litjens, Thijs Kooi, Babak Ehteshami Bejnordi, Arnaud Arindra Adiyoso Setio, Francesco Ciompi, Mohsen Ghafoorian, Jeroen AW van der Laak, Bram van Ginneken, and Clara I Sánchez. 2017. A survey on deep learning in medical image analysis. *Medical image analysis* 42 (2017), 60–88.

[20] Andreas Lugmayr, Martin Danelljan, Andres Romero, Fisher Yu, Radu Timofte, and Luc Van Gool. 2022. Repaint: Inpainting using denoising diffusion probabilistic models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 11461–11471.

[21] Pavlo Molchanov, Shalini Gupta, Kihwan Kim, and Jan Kautz. 2015. Hand gesture recognition with 3D convolutional neural networks. In *2015 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*. 1–7. https://doi.org/10.1109/CVPRW.2015.7301342

[22] Brian B Moser, Arundhati S Shanbhag, Federico Raue, Stanislav Frolov, Sebastian Palacio, and Andreas Dengel. 2024. Diffusion models, image super-resolution and everything: A survey. *arXiv preprint arXiv:2401.00736* (2024).

[23] Thi Thanh Mai Nguyen, Ngoc Hai Pham, Van Thai Dong, Viet Son Nguyen, and Thi Thanh Hai Tran. 2011. A fully automatic hand gesture recognition system for human-robot interaction. In *Proceedings of the 2nd Symposium on Information and Communication Technology* (Hanoi, Vietnam) *(SoICT '11)*. Association for Computing Machinery, New York, NY, USA, 112–119. https://doi.org/10.1145/2069216.2069241

[24] OpenAI. 2023. GPT-4 Technical Report. In *arXiv preprint arXiv:2303.08774*.

[25] Christoffer Bøgelund Rasmussen, Kristian Kirk, and Thomas B Moeslund. 2022. The challenge of data annotation in deep learning—a case study on whole plant corn silage. *Sensors* 22, 4 (2022), 1596.

[26] Tal Ridnik, Emanuel Ben-Baruch, Asaf Noy, and Lihi Zelnik-Manor. 2021. Imagenet-21k pretraining for the masses. *arXiv preprint arXiv:2104.10972* (2021).

[27] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. 2022. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 10684–10695.

[28] Rubin Bose S. and Sathiesh Kumar V. 2020. Hand Gesture Recognition Using Faster R-CNN Inception V2 Model. In *Proceedings of the 2019 4th International Conference on Advances in Robotics* (Chennai, India) *(AIR '19)*. Association for Computing Machinery, New York, NY, USA, Article 19, 6 pages. https://doi.org/10.1145/3352593.3352613

[29] Sandeep Reddy Sabbella, Sara Kaszuba, Francesco Leotta, Pascal Serrarens, and Daniele Nardi. 2024. Evaluating Gesture Recognition in Virtual Reality. *arXiv preprint arXiv:2401.04545* (2024).

[30] Vaidehi Sharma, Abhishek Sharma, and Sandeep Saini. 2024. Real-time attention-based embedded LSTM for dynamic sign language recognition on edge devices. *Journal of Real-Time Image Processing* 21, 2 (2024), 53.

[31] Zhen Tan, Alimohammad Beigi, Song Wang, Ruocheng Guo, Amrita Bhattacharjee, Bohan Jiang, Mansooreh Karami, Jundong Li, Lu Cheng, and Huan Liu. 2024. Large language models for data annotation: A survey. *arXiv preprint arXiv:2402.13446* (2024).

[32] Maxim Tkachenko, Mikhail Malyuk, Andrey Holmanyuk, and Nikolai Liubimov. 2020-2022. Label Studio: Data labeling software. https://github.com/heartexlabs/label-studio Open source software available from https://github.com/heartexlabs/label-studio.

[33] Aurelijus Vaitkevičius, Mantas Taroza, Tomas Blažauskas, Robertas Damaševičius, Rytis Maskeliūnas, and Marcin Woźniak. 2019. Recognition of American sign language gestures in a virtual reality using leap motion. *Applied Sciences* 9, 3 (2019), 445.

[34] University of Oxford Visual Geometry Group. 2021. VGG Image Annotator (VIA). https://www.robots.ox.ac.uk/~vgg/software/via/.

[35] Ko Watanabe, Yusuke Soneda, Yuki Matsuda, Yugo Nakamura, Yutaka Arakawa, Andreas Dengel, and Shoya Ishimaru. 2021. DisCaaS: Micro Behavior Analysis on Discussion by Camera as a Sensor. *Sensors* 21, 17 (2021). https://doi.org/10.3390/s21175719

[36] Ko Watanabe, Seiya Tanaka, Andrew Vargo, Koichi Kise, and Andreas Dengel. 2024. Comparing Web Browsing Behaviors with High and Low Information Literacy: A Case Study for Fact Check Against GPT Generated Fake News. In *Companion of the 2024 on ACM International Joint Conference on Pervasive and Ubiquitous Computing*. 9–13.

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

[38] Yaseen, Oh-Jin Kwon, Jaeho Kim, Sonain Jamil, Jinhee Lee, and Faiz Ullah. 2024. Next-Gen Dynamic Hand Gesture Recognition: MediaPipe, Inception-v3 and LSTM-Based Enhanced Deep Learning Model. *Electronics* 13, 16 (2024), 3233.

[39] Zhengxia Zou, Zhenwei Shi, Yuhong Guo, and Jieping Ye. 2018. Object detection in 20 years: A survey. *International Journal of Computer Vision* 128, 2 (2018), 261–318.

APPENDIX

In Figure 9, we present the key points detected by the skeleton model provided by MediaPipe for hand tracking and gesture recognition. These landmarks, ranging from the wrist to the joints of each finger, form the foundation of our gesture recognition pipeline, allowing for precise motion tracking and accurate mapping of gestures.

In Table 4, we provide a detailed breakdown of the technical specifications for each device used in our experiments. These specifications play a critical role in understanding the variation in data quality and model performance across different hardware setups, from high-resolution cameras to more standard device captures.

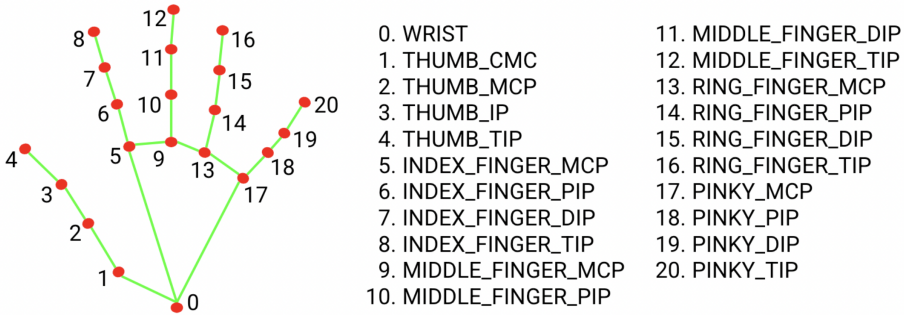


Fig. 9. Hand Landmarks: The image shows the key points and their associated labels used for hand tracking and gesture recognition. These include the wrist (0), thumb joints (1-4), index finger joints (5-8), middle finger joints (9-12), ring finger joints (13-16), and pinky finger joints (17-20) [9]

Table 4. Technical Specifications of Devices Used for Data Collection.

Device	Specifications
Sony X1 Camera	- 1/30 frames per second (fps) - F2.0 aperture - ISO 125 - Fine quality - Auto exposure - 3:2 aspect ratio - 20M resolution
iPhone 14 Pro Max	- 48 MP sensor - 24 mm equivalent f/1.78 aperture lens - Sensor-shift Optical Image Stabilization (OIS) - Dual Pixel autofocus (AF) - Auto exposure
MacBook Air M1	- Screenshots taken from display - MacBook screen resolution used for image capture - Screen was used for evaluation of gestures