ChatFlags: An AI-Powered Semaphore Interactive System

YAN HUAN

A thesis submitted to the Auckland University of Technology

in partial fulfillment of the requirements for the degree of

Master of Computer and Information Sciences (MCIS)

2025

School of Engineering, Computer & Mathematical Sciences

Abstract

This study presents the development of ChatFlags, an intelligent system for flag recognition and interaction. YOLO11 was selected as the visual backbone based on experiments involving five flag classification tasks. The custom dataset was refined and expanded to address the lack of publicly available resources. An improved model, YOLO-AKEMA, integrating attention mechanisms and adaptive convolution, achieved higher accuracy across 27 flag categories. The user interface was built by using the AI platform Dify, supporting conversational interaction. To mitigate hallucinations in large language models, a retrieval-augmented generation (RAG) framework was constructed by using curated flag documents and the BGE-M3 embedding model. Finally, the DeepSeek language model was integrated via workflow orchestration to complete the system. ChatFlags supports natural language dialogue, flag video analysis, knowledge quizzes, and text-to-image/video conversion. Its multimodal features enhance interactivity, offer a scalable solution for flag language education, and extend the integration potential of vision and language models.

Keywords: Semaphore recognition, Semaphore learning system, YOLO-AKEMA-D, DeepSeek

Contents

Attestation of Authorship

I hereby declare that this submission is my own work and that, to the best of my knowledge and belief, it contains no material previously published or written by another person (except where explicitly defined in the acknowledgments), nor material which to a substantial extent has been submitted for the award of any other degree or diploma of a university or other institution of higher learning.

Signature: _____ **Date:** *20 May 2025*

Acknowledgements

During the research and writing of this thesis, I received care, help and support from many units and individuals. I would like to express my sincerest gratitude to all those who gave me guidance and help.

First of all, I would like to express my special thanks to my supervisor, Wei Qi Yan, who not only gave me careful academic guidance, but also set an example for me in research methods, paper writing and scientific research thinking. In the process of topic selection, method design, experimental verification and paper revision of this project, Dr. Yan always gave me patient guidance and detailed suggestions, which enabled me to make continuous progress on the road of academic research. It is precisely because of his rigorous scholarship and selfless help that I can successfully complete this research work.

At the same time, I would like to thank Auckland University of Technology for providing me with a good learning environment so that I can devote myself to research work with peace of mind.

In addition, I would also like to thank my family and friends who encouraged and supported me during my studies, especially my own.

I would like to especially thank my wife, who participated in the data collection of the experiment throughout the whole process. It was her assistance and encouragement that enabled me to conduct experiments and research smoothly.

Finally, I would like to express my sincere gratitude again to all those who care about and support this research.

1 Introduction

1.1 Research Background

Flag language is a traditional, visual, and standardized form of long-distance communication, transmitting information through precise configurations of flags or arms (Mead, 1935). It utilized when radio communication was not yet prevalent and functioned as an efficient medium, particularly in maritime settings. Over time, it has been adopted in military operations, railway systems, and emergency responses.(Army, 1910; Sterling, 2007).

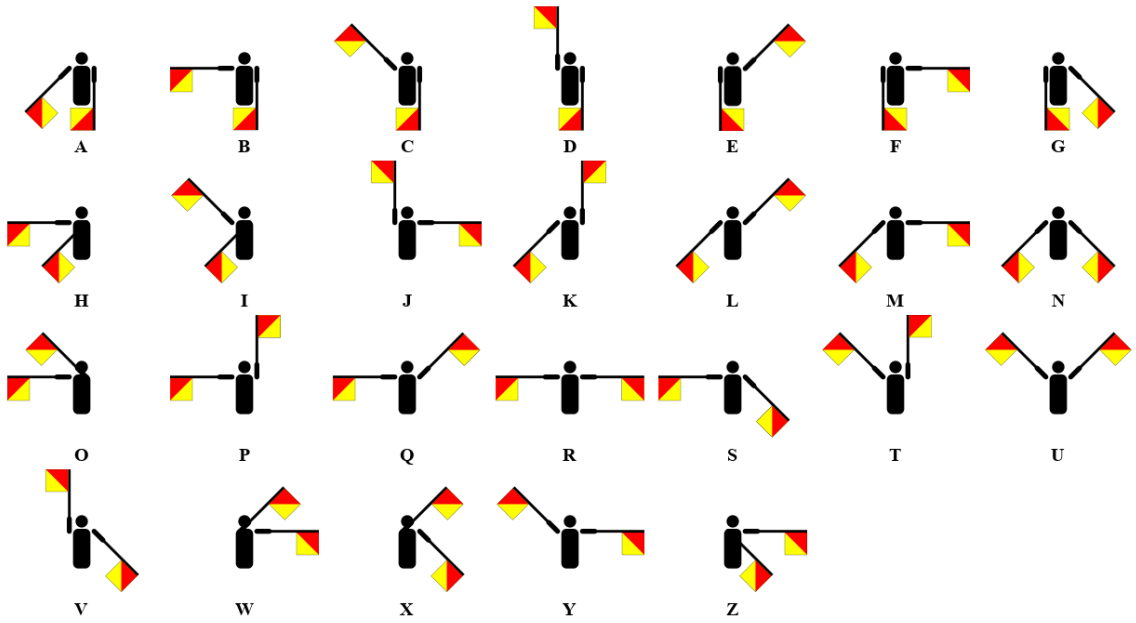

Figure 1.1: Semaphore Letter Code

In Fig.1.1, the internationally recognized flag language system requires individuals to hold a small flag in each hand and perform a series of predefined angles to

represent letters, numbers, or commands. This method offers notable advantages, including independence from electronic devices, adaptability to diverse environmental conditions, and accessibility due to its fixed set of postures. Even in settings with limited or disrupted radio signals, flag language remains a dependable mode of communication(Putra et al., 2018).

The rapid development of modern communication technologies—such as radio systems, mobile networks, and satellite transmission—has established them as the primary channels for information exchange, resulting in the gradual decline of flag language in everyday contexts. Nevertheless, in certain specialized settings and emergency situations, flag language remains essential(Qian et al., 2016; Rachmad & Fuad, 2015). In scenarios such as maritime navigation, military operations, disaster response, and communication in remote regions, radio signals are frequently vulnerable to interference or complete failure. Under these conditions, flag signaling serves as a dependable alternative for relaying information and ensuring the accurate delivery of critical instructions. As a result, it remains an essential component of training in educational institutions, maritime academies, and military programs.

Although flag language provides several advantages as a communication modality, its recognition and interpretation rely heavily on manual observation, which not only requires a high level of operator expertise but also limits the accuracy and efficiency of information transmission in complex environments. Proficient understanding of encoding rules is therefore essential to ensure precise transmission and prompt interpretation of intended messages. However, the reliability of human visual recognition is subject to various influencing factors, including the competency of both

sender and receiver, transmission distance, weather conditions, and environmental background(Rachmad & Fuad, 2018).

To address the limitations associated with manual learning and interpretation of flag language, the application of computer-based analysis has long constituted a prominent research focus.In 2012, Iwane (2012) designed and implemented a ship-to-ship communication system based on flag language to address the challenge of maintaining effective communication when radio signals are limited or unavailable. The study demonstrated that the system offers meaningful support for individuals with limited experience in flag signaling.

In recent years, the rapid development of artificial intelligence has brought renewed attention to object detection and classification, which remain prominent topics in computer vision. The emergence of deep learning has significantly broadened the scope of applications in this domain. Deep learning has demonstrated strong capabilities in interpreting diverse postures and complex scenes. When integrated with human pose estimation, it enables precise and automated analysis of human motion in areas such as fitness instruction, medical rehabilitation, and high-performance sports training(Cust et al., 2019; Illavarason et al., 2019) . As a posture-dependent communication system, flag language can be effectively analyzed through deep learning techniques to enable automated recognition.

On the other hand, although semaphore operates within a fixed rule set and a coherent logical structure, the learning process remains complex. Variability in individual gesture patterns and a lack of consistent standardization may compromise interpretive reliability in practical contexts. Furthermore, environmental conditions

frequently impede learners in acquiring and applying semaphore effectively. Traditional instructional approaches demand the memorization of numerous gesture-angle configurations, thereby imposing significant cognitive demands on beginners.

In 2015, Helena (2015) proposed a signal flag learning application developed on multimedia and web platforms, integrated with computer-assisted instruction (CAI) methods to create a more intuitive, dynamic, and interactive learning environment. The system delivers flag-related content through visual and textual media, thereby facilitating the acquisition of basic flag gestures by novice learners. However, a principal limitation of this study lies in the absence of automatic recognition and feedback mechanisms, as well as limited variation in flag display formats. Nevertheless, the introduction of this system underscores the relevance of signal flag learning technologies.

A Large Language Model (LLM) (Di et al., 2025) is an artificial intelligence system grounded in deep learning and natural language processing (NLP) techniques (Kang et al., 2020), designed to comprehend, generate, and process natural language text. An LLM is capable of simulating human dialogue, delivering personalized question-and-answer experiences, and enabling intelligent interaction in domains such as education, customer service, and virtual assistance. In particular, such models have demonstrated significant potential in educational applications (Guizani et al., 2025).

Given the widespread adoption of large language models, their integration into flag language learning systems has become a discernible direction for developing intelligent and innovative platforms that enhance both the dissemination and practical

implementation of flag language.

1.2 Research Questions

The primary objective of this thesis is to employ deep learning techniques in conjunction with a large language model to develop an integrated system for flag signal video recognition, analysis, and knowledge-based question answering.

Therefore, the research questions guiding this thesis are as follows:

- How to choose a more appropriate deep learning model for flag classification tasks?
- For flag training tasks that rely on arm movements, can human posture detection tools be used for auxiliary optimization to improve efficiency and performance?
- How to optimize the deep learning model to achieve the effect of accurately predicting flag movements?
- How to ensure detection speed and accuracy and balance performance and efficiency in practical applications?
- Is it possible to combine visual models and large language models to give full play to their respective capabilities and create new types of applications?

The core of this project is the real-time flag action recognition analysis and intelligent interaction. The model must be able to accurately identify flag actions and give corresponding descriptions, and the large language model must be able to

correctly and clearly describe the analysis results. It must be able to understand user needs through natural language and give correct feedback.

1.3 Contributions

This research project aims to conduct flag gesture analysis through deep learning and to implement a question-answering interaction system in conjunction with a large language model. A real-time and efficient question-answering platform was developed through the integration of DeepSeek and YOLO11. Additionally, a custom dataset comprising gesture representations of the 26 English alphabet letters was collected and constructed. By the end of this study, we were able to:

- Collected and organized proprietary flag gesture datasets to support model training.
- Employed human posture recognition tools to facilitate data augmentation, model optimization, and pose analysis, thereby ensuring the accuracy of recognition feedback.
- Utilized YOLO-AKEMA to achieve real-time recognition of 27 distinct flag gesture categories with high precision.
- Domain-specific knowledge of flag gestures was structured, and a dedicated knowledge base was constructed using a text embedding model to mitigate hallucination in DeepSeek-generated responses.
- Developed the ChatFlags system, capable of supporting flag gesture video

analysis and delivering multimodal information—including text, images, and video—through a question-answering interface.

1.4 Objectives of This Thesis

Driven by the growing demand for flag language learning systems, this thesis investigates the integration of large-scale language models and visual analysis techniques into flag language instruction, aiming to develop an intelligent and innovative educational platform that facilitates both the dissemination and practical implementation of flag communication. The proposed system not only presents flag-related knowledge in diverse formats but also performs recognition and analysis of video sequences containing flag gestures, with the capability of generating automated feedback. This research seeks to overcome the limitations of conventional instructional approaches in terms of engagement and interactivity, support the intelligent evolution of flag language pedagogy, and expand its applicability within modern society.

1.5 Structure of This Thesis

The description of this thesis is as follows:

- Chapter 2 reviews related work and introduces the research and application of related technologies for human posture and flag signal recognition. It also introduces the development and application of large language models.

- Chapter 3 introduces a series of experiments, including preliminary experiments comparing the performance of various CNN models in flag recognition, and

model training experiments on a dataset of 27 types of actions. This chapter also describes the detailed steps of how to combine the visual model with DeepSeek and Dify to build the ChatFlags flag learning interactive system.

- In Chapter 4, the experimental results are visualized and analyzed, the performance of the model is analyzed, and the interface functions of ChatFlags are displayed and explained.
- In Chapter 5, we analyze the overall research process and results and conduct a discussion based on the research questions.
- Chapter 6 summarizes the entire study and prospects for future related work

2 Related Work

2.1 Introduction

Human pose estimation has consistently remained a central topic within computer vision, exhibiting considerable promise in a wide range of application contexts(Andriluka et al., 2014), including medical diagnosis(Wu et al., 2020), animation production(Borodulina, 2019), and motion analysis(Badiola-Bengoa & Mendez-Zorrilla, 2021). Flag signaling can also be considered a subdomain of human pose estimation, as its recognition relies on detecting and interpreting specific human postures.

Human pose estimation involves detecting and reconstructing the 3D configuration of the human body based on images or video feeds(Wang et al., 2021). It typically involves two main stages: feature extraction and classification(Pavlovic et al., 1997). Prior to the advent of deep learning, pose estimation primarily relied on traditional computer vision techniques, which involved manually designing and adjusting feature descriptors to predict human keypoints.

As deep learning continues to evolve, especially convolutional neural networks (CNNs)(LeCun et al., 1998), these traditional manual techniques have largely been supplanted by automated approaches that can directly extract discriminative features from data, greatly enhancing the accuracy and efficiency of pose estimation. This chapter provides an overview of the development and related research in human pose estimation and flag signal recognition.

2.2 Pose and Flag Recognition

2.2.1 Traditional Methods

Early research on gesture and posture recognition established the foundation for flag signal recognition. Pavlovic et al. (1997) emphasized the significance of feature extraction and classification, and identified challenges such as visual noise and illumination variation. Freeman and Roth (1995) underlined the importance of low-level features, including color, edge, and shape. Wu and Huang (1999) applied Hidden Markov Models (HMMs) to gesture sequence modeling, contributing to the development of time-series-based methods.

In 2013, Jetley et al. (2013) introduced a method for automatic flag pattern recognition by combining color texture analysis with gradient feature extraction. The approach applied color space transformation and texture distribution analysis to isolate dominant color regions and reduce background and lighting interference. Local features, such as Histograms of Oriented Gradients (HOG) (Dalal & Triggs, 2005), were employed to describe geometric shapes and edge structures.

Early research on posture and flag recognition primarily relied on traditional image processing techniques, including color segmentation, edge detection, and shape feature extraction. Features such as body parts or flag positions were typically extracted manually. These methods often resulted in high computational complexity and limited generalization capability (Bregler & Malik, 1998; Felzenszwalb & Huttenlocher, 2005).

In 2011, researchers first employed Kinect(Han et al., 2013; Jana, 2012) depth

cameras as a new data source and achieved human posture recognition from single-frame depth images based on joint position estimation, providing a foundation and new direction for subsequent research and applications in posture analysis Shotton et al. (2011). With the emergence and widespread adoption of depth sensors such as Kinect, skeleton-based gesture recognition has gained increasing attention and has been widely applied to flag signal recognition during this period(Keskin et al., 2011; Shotton et al., 2011). In 2012, Iwane (2012) developed an experimental system utilizing Kinect sensors and the OpenNI/NITE libraries to recognize arm movements corresponding to Japanese flag signals. The system records the three-dimensional position of the upper limbs of the body and calculates the angles of the upper and lower arms to determine the arm posture. A velocity threshold was used to determine motion cessation, after which the corresponding flag signal was identified based on the current posture. In tests involving five novice users, The system attained an average recognition accuracy of 94%. However, certain postures exhibited reduced stability due to non-standard arm movements and sensor limitations.

The integration of flag language with Kinect sensors has also been applied to address challenges in human-computer interaction. Traditional interaction methods often rely on physical input devices such as buttons and remote controls, which are limited by low intuitiveness and operational complexity. To overcome these limitations, Chang et al. (2014) proposed a flag language gesture recognition system to enable more natural and intuitive interaction. A standardized flag language letter system was adopted as a unified command input, allowing the skeletal data captured by Kinect to be effectively mapped to specific gesture commands. Experimental re-

sults showed that the system could accurately recognize static gestures corresponding to flag language letters.

The development of flag learning systems represents another important application of flag recognition technology (Helena, 2015; Lu et al., 2021). In response to the limitations of early systems lacking dynamic recognition and real-time feedback, Rachmad and Fuad (2015) proposed a method based on skeletal images obtained from a Kinect sensor. This approach utilizes upper body skeletal joint data and applies geometric algorithms to calculate the spatial position and angular features of the arms, enabling automatic recognition of letters. A visual interface for flag language learning was also developed. However, compared with earlier systems, this method primarily improves recognition accuracy without significant progress in interaction.

During this period, research on human–computer interaction and flag learning systems continued to rely on manual feature extraction, which required considerable effort. Moreover, image quality had a significant impact on recognition outcomes. These limitations impeded the effective implementation and scalability of flag recognition systems in applied contexts beyond controlled environments.

2.2.2 Deep Learning Methods

Driven by the swift progress of deep learning, research in human pose estimation and flag signal recognition has progressively moved away from traditional feature engineering, embracing deep models such as convolutional neural networks (CNNs)(Chen, 2024b; LeCun et al., 1998) to enable end-to-end automatic feature extraction and classification.

Convolutional Neural Networks represent a category of deep learning models extensively utilized in image analysis, computer vision, and various pattern recognition tasks. Their core advantage lies in the automatic extraction of local features from input data via convolution operations, thereby minimizing the reliance on manually designed features(Albawi et al., 2017).

A standard CNN architecture typically incorporates several types of layers, including convolutions, pooling mechanisms, nonlinear activations, and dense connections. The convolutional layers are responsible for capturing spatial features through the application of kernels, whereas the pooling layers perform downsampling to reduce the dimensionality of feature maps, thereby lowering computational demands and improving the model's resilience. Nonlinear activation functions, such as ReLU, increase the representational capacity of the network. The extracted features are transformed into classification results through dense layers that serve as the final decision-making stage of the network (Albawi et al., 2017). Since the emergence of LeNet-5 (LeCun et al., 1998), convolutional neural network (CNN) architectures have undergone continuous advancement, facilitating the development of several foundational models, including AlexNet (Krizhevsky et al., 2012), VGGNet (Simonyan & Zisserman, 2015), and ResNet (He et al., 2016).These models have proven highly effective in various applications, including visual categorization, object detection, and scene-level parsing.

Compared to traditional methods, deep learning models exhibit superior feature representation capabilities and can autonomously learn robust visual features from large-scale image datasets. These models retain high recognition accuracy under

challenging conditions, including complex backgrounds, illumination variation, and performer-specific differences.

A representative example is DeepPose, proposed by Toshev and Szegedy (2014), which was the first to apply deep neural networks to human pose estimation. DeepPose employs convolutional neural networks to extract keypoint coordinates from images and generate a two-dimensional skeletal representation of human posture. Subsequently, Newell et al. proposed the Hourglass network (Newell et al., 2016), characterized by a deep architecture with multi-scale and repeated connections, enabling the model to capture features at various spatial resolutions.

In this framework, notable advancements have occurred in the areas of flag recognition and human posture analysis. In 2016, Qian et al. (2016) introduced a convolutional neural network-based model for flag signal recognition. The model autonomously derives features from visual inputs using several convolutional and subsampling layers, and subsequently applies a Softmax classifier to determine signal types. Experimental findings confirmed that this method achieves superior performance compared to conventional techniques, particularly in recognition precision, while also enhancing system stability and scalability. This work provided an early reference for applying deep learning techniques to flag signal recognition and validated the effectiveness of CNNs in image-based feature extraction and classification within this domain.

Building on this foundation, Peng et al. (2019) proposed a human action recognition (HAR) algorithm tailored for flag gesture recognition. The study involved collecting flag action data under three distinct scenarios and employing a deep convo-

lutional generative adversarial network (DCGAN) for data augmentation, resulting in an expanded dataset named DataSR. A fully convolutional network combining 1×1 and 3×3 kernels was constructed, incorporating group convolution to reduce model complexity. The final model, HARNET, achieved an accuracy of 94.36%, significantly lowering computational cost while maintaining high performance.

Similarly, Li et al. (2018) introduced two CNN-based models for flag gesture recognition: SRNet and Tiny-SRNet. SRNet consists of five convolutional layers and three fully connected layers, with batch normalization applied before each convolutional layer except the first to enhance training stability. It achieved a recognition accuracy of 98.9% on a dedicated flag dataset. To facilitate deployment on mobile and embedded platforms, Tiny-SRNet was proposed by removing fully connected layers and incorporating convolutional and global average pooling layers. This adjustment reduced the model size to one twenty-fourth of SRNet, with only a 1.7% decrease in accuracy, demonstrating a practical balance between efficiency and performance.

In 2018, Tian et al. (2018) proposed a robust cloud-based flag mirror system designed to enable social robots to perform real-time recognition and imitation of flag gestures. The recognition component utilized a CNN architecture to process visual data and perform categorization, enabling the framework to attain a high level of identification accuracy. This study further demonstrated the effectiveness of CNNs in flag gesture recognition and expanded their application into human–robot interaction.

With the advancement of neural network technologies, CNNs have become the dominant approach in both flag recognition and human pose estimation tasks due

to their superior capability in image feature extraction. Numerous studies have refined classical CNN architectures to extract key features from static images using multi-layer convolution and pooling operations, achieving satisfactory recognition performance. However, CNNs exhibit limitations in modeling temporal information and long-range dependencies, particularly in dynamic flag gestures and continuous sequence recognition.

In response to this limitation, models capable of capturing time-dependent patterns have been proposed. Such as Long-Short-Term Memory (LSTM) networks(Lee et al., 2022), often combined with CNN or 3D-CNN architectures, have demonstrated strong performance in capturing temporal dependencies, enabling the effective processing of sequential flag or gesture data.

In the domain of human pose estimation, Li et al. (2024) introduced an improved algorithm based on YOLO-Pose to enhance recognition accuracy and efficiency in complex environments. Key modifications include incorporating a lightweight GhostNet module into the backbone to reduce parameter count for resource-constrained deployment, integrating the ACmix attention mechanism into the Neck to strengthen feature extraction, and applying coordinate attention in the Head to improve keypoint localization. Additional optimization of the loss and confidence functions further enhanced model robustness. The evaluation outcomes indicate that the enhanced architecture obtained mAP@50 of 95.58% and mAP@50–95 of 69.54%, with a parameter reduction of 14.6M, demonstrating superior detection speed and accuracy.

In terms of flag signal recognition, Gohil et al. (2023) designed a real-time flag

signal detection system for maritime environments to improve the automation and safety of ship communication.The system integrates visual perception methods with advanced neural models to detect and distinguish shipborne communication flags. Putra et al. (2024) adopted a back-propagation neural network (BPNN) to recognize flag letters based on wavelet features; however, the method exhibited sensitivity to noise and lacked the capacity to capture temporal dynamics due to its traditional architecture.

To improve the interactivity of flag language learning, Penulis (2024) designed a coding recognition system using Raspberry Pi, OpenCV, and MediaPipe, with CNNs employed for automatic gesture recognition. Virginia (2024) further developed a gamified learning system that recognizes flag gestures in real time using PoseNet-based human pose estimation and provides interactive feedback to enhance learner engagement.

Canoy et al. (2024) proposed a real-time detection system for maritime flag signal classification based on the YOLO algorithm. The system successfully identified and categorized international signal flags, demonstrating the applicability of YOLO in flag classification tasks under real-world maritime conditions.

As deep learning continues to develop, various applications in computer vision like body posture analysis and flag signal recognition have increasingly adopted deep architectures to enhance recognition accuracy and system robustness. In particular, flag recognition has undergone a paradigm shift from traditional handcrafted feature extraction to end-to-end learning frameworks enabled by deep neural networks.

2.3 Large Language Models

Large Language Models (LLMs) constitute a category of natural language processing systems that are grounded in deep neural architectures and trained on extensive text corpora. Built upon the Transformer framework, they utilize self-attention mechanisms to capture contextual dependencies between words, enabling robust semantic understanding and text generation. LLMs are widely applied in tasks such as text generation, question answering, and machine translation.

In recent years, LLM applications have expanded rapidly across education and various industries. Numerous studies have demonstrated their potential across diverse scenarios. In the education sector in particular, LLMs show great promise. For instance, Tan et al. (2024) developed a framework named ELF to facilitate the generation of high-quality educational content. By leveraging exemplary teaching samples, ELF constructs guided prompts that help LLMs generate contextually appropriate content for classroom settings. This approach enables the transfer of teaching strategies to new subjects through few-shot learning, with minimal manual annotation, for tasks such as dialogue completion and knowledge transfer. ELF enhances AI–teacher interaction and illustrates how LLMs can support adaptive instruction.

Similarly, Xenakis et al. (2024) proposed an intelligent resource platform using LLMs to deliver personalized recommendations and resource management for computational thinking, AI, and STEM education. Neumann et al. (2024) integrated LLM-powered chatbots into higher education database courses, enabling personalized learning and real-time feedback while alleviating teaching workload.

In enterprise scenarios, LLMs have been combined with Retrieval-Augmented Generation (RAG) and frameworks such as LangChain to enhance information retrieval and generative services, improving operational efficiency(Cheonsu, 2023; Jeong, 2023). In the domain of AIoT, Shen et al. (2025) introduced a natural language programming system based on LLMs, enabling users to describe requirements in plain language for automatic program generation, encompassing knowledge retrieval, synthesis, and code optimization.

These studies reflect the growing versatility of LLMs across sectors. However, despite longstanding efforts in semaphore teaching, the integration of LLMs into this domain remains unexplored. Combining LLMs with semaphore learning holds the potential to create intelligent, interactive learning systems, thereby opening new application pathways in semaphore education and contributing to its broader dissemination.

2.4 Chapter Summary

We first reviewed the technical evolution of computer vision and surveyed related research in human pose estimation and flag signal recognition. Advances in deep learning have significantly driven progress in both domains, enabling broader applications across various fields. In the specific area of flag signal recognition, early approaches relied heavily on handcrafted feature derivation and rule-based classification. With the emergence of depth sensing technologies, researchers began using skeletal joint data to abstract and simplify gesture representation, shifting focus from raw pixel information to coordinate-based posture analysis. This transition improved

classification performance and reduced background interference.

Subsequently, the implementation of CNN-based models and similar frameworks facilitated a unified pipeline for automated representation learning and decision making, improving robustness and reducing the need for manual feature design. These developments laid the groundwork for real-world applications such as long-distance maritime flag detection, IoT-compatible recognition systems, and gamified flag learning platforms. However, challenges remain: CNN-based training still requires extensive manual annotation, and most existing flag language learning applications only support static, one-way output in the form of images or text, lacking real-time video-based interactivity.

As large language models (LLMs) continue to evolve, there is a growing trend toward integrating machine vision with natural language understanding. Recent multimodal models, such as OpenAI's GPT-4o(Shahriar et al., 2024), Google's Gemini 1.5, and Anthropic's Claude 3, support joint text–image inputs, enabling visual reasoning and image-grounded dialogue. These models enhance real-world perception and support applications in education, design, and medical imaging, illustrating the mainstream shift toward multimodal intelligence(Yin et al., 2024). However, current systems offer limited support for video-based understanding and remain largely unexplored in the niche domain of flag language.

To address this gap, this research introduces a framework designed to incorporate computer vision and LLMs to enable real-time understanding and interpretation of flag signal action videos. The system not only identifies flag postures but also generates corresponding textual meanings, offering a multimodal, interactive learning

experience. This integration represents an innovative application of LLMs, providing a novel pathway for advancing intelligent semaphore education and contributing to the broader dissemination of flag language in the era of deep learning.

3 Methodology

3.1 Introduction

This chapter presents a series of experiments and related materials for the flag signal recognition task. Initial pre-verification experiments were conducted using several CNN models and YOLO11. The results demonstrated the superior performance of YOLO11 in flag signal recognition. Based on these findings, a comprehensive classification experiment was carried out using YOLO11, which was further improved through the integration of attention mechanisms and convolution modules. The human pose estimation tool MediaPipe was adopted to extract skeletal keypoints and assist with annotation generation, data augmentation, and image analysis. The final model, YOLO-AKEMA, accurately recognized all 27 categories of flag signal actions, establishing a solid foundation for the development of the ChatFlags system. To evaluate model performance and identify key influencing factors, multiple comparative analyses and ablation trials were systematically planned and executed.

The previously trained YOLO-AKEMA model demonstrates strong classification performance, making it a suitable foundation for the construction of practical applications. Although attempts to develop flag language learning systems have been made in the past, most remain limited to unidirectional outputs in the form of static images and text due to technological constraints. To address this limitation, the proposed system integrates a visual recognition model with a large language model and leverages the Dify platform(an intelligent AI application development tool)to con-

struct an AI-driven, interactive learning environment for flag language education.

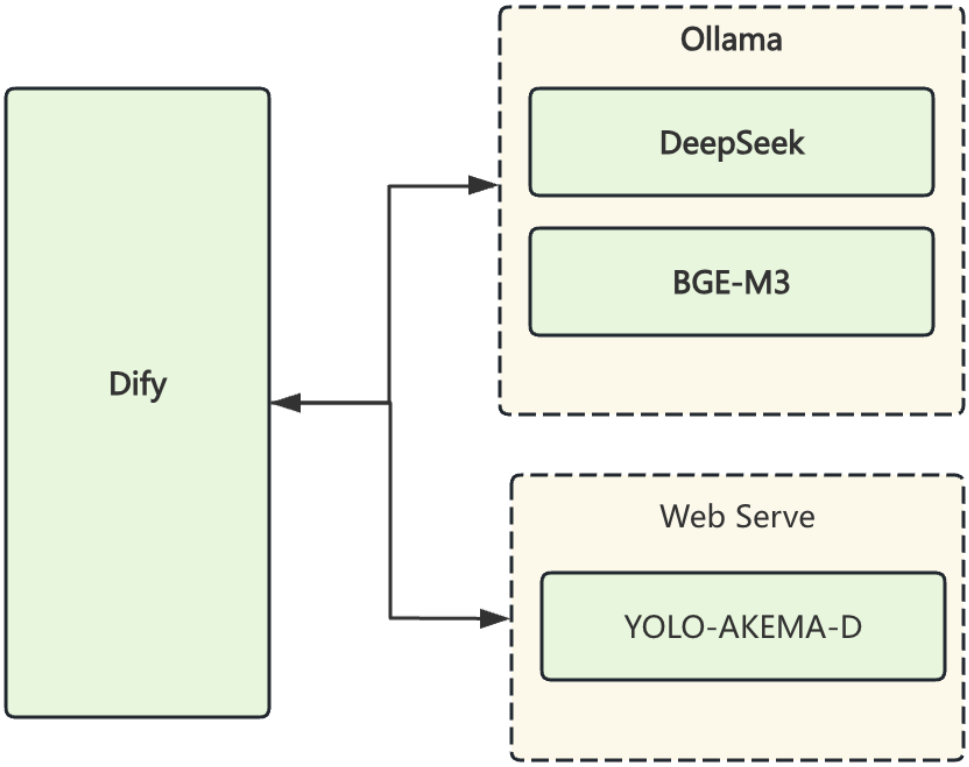

Figure 3.1: ChatFlags Architecture

This study proposes the integration of YOLO-AKEMA, DeepSeek, BGE-M3, and the Dify platform to develop an interactive system named ChatFlags, designed to support learners in acquiring flag language through intelligent visual-textual interaction. As shown in the **Fig.** 3.1, the proposed system utilizes the YOLO-AKEMA model to perform prediction and recognition of flag language gestures in video content, followed by action analysis and the construction of a service interface for interaction with the Dify platform. Simultaneously, the BGE-M3 model is employed to

construct a specialized flag language knowledge base, addressing the limitations of DeepSeek in terms of domain-specific understanding.

Building upon these components, Dify is applied to develop an interactive system that integrates the flag language knowledge base and supports multiple functionalities, including video-based gesture recognition, visual display of flag language actions, and natural language-driven generation of flag language videos. This comprehensive design enhances the engagement and accessibility of flag language learning, offering learners a clearer and more engaging way to acquire knowledge.

It is anticipated that this novel and integrative solution will offer a fresh and effective approach for individuals interested in learning flag language, thereby expanding the possibilities for intelligent, user-centered language education.

3.2 Materials

3.2.1 ResNet18

CNN represents a core framework within deep learning, widely regarded as a fundamental component of modern artificial intelligence systems. Inspired by the human visual system, it can be more accurately described as a computational model that emulates certain functional aspects of the human brain(Qian et al., 2016). They highly suitable for image recognition and other vision-related tasks(Zhao et al., 2024) ResNet-18 is a commonly used neural network architecture consisting of 18 computational layers, originally intended for visual categorization applications. In one study, a pre-trained ResNet-18 was fine-tuned on a Kaggle dataset containing images of different yoga poses, achieving 98% training accuracy(Aruna et al., 2024).

ResNet18 employs a core design based on residual learning, in which shortcut connections are introduced through residual blocks. This mechanism helps mitigate the issue of gradient disappearance and contributes to enhanced training stability in deep neural networks. ResNet-18 begins with a single convolutional layer, after which four residual modules are arranged, each containing a pair of convolutional operations.This is followed by an average pooling operation across the entire feature map and a concluding dense layer. Due to its compact architecture, ResNet-18 achieves strong classification performance with efficient computation, rendering it ideal for application in limited-resource settings (He et al., 2016). ResNet18 was utilized to perform feature representation and recognition of goat face imagery. A tailored CNN framework was developed and optimized, allowing precise recognition of distinct goats despite challenging visual environments. The validated capability of ResNet18 in visual encoding and categorization supports its applicability within the experimental setup(Xue et al., 2024).

3.2.2 DenseNet121

DenseNet121, introduced in 2017, is a representative model within the DenseNet family. It incorporates a densely connected architecture in which the output from each layer is transmitted to all subsequent layers, facilitating high levels of feature reuse and enhancing the efficiency of gradient propagation (Huang et al., 2017). The concept of dense connectivity in DenseNet was inspired by the residual connections in ResNet, but advances the concept by linking every layer with those that follow it, rather than only to adjacent ones. In contrast to ResNet's element-wise addition,

DenseNet merges outputs from all earlier layers with the present one, thereby enhancing the richness of information transmission. DenseNet121 comprises 121 layers, including four dense blocks and intervening transition layers, which serve to reduce feature map dimensionality and computational complexity. Compared to other deep networks such as ResNet, DenseNet achieves similar or superior performance with fewer parameters, while offering stronger generalization and higher computational efficiency. It has proven effective in applications including visual categorization and object recognition.

In 2024, researchers introduced an efficient gesture-based human–computer interaction system tailored for individuals with disabilities. A gesture classification approach based on DenseNet121 was proposed to recognize sign language and other gestures (Liu, Nand, et al., 2023), thereby enabling a novel interaction paradigm and promoting the advancement of assistive technologies for improved accessibility (Dabwan et al., 2024).

3.2.3 MediaPipe

MediaPipe, developed by Google AI, is an open-source platform designed with a modular architecture that enables data flow through computational graph structures. It accommodates multiple data modalities—such as images, video streams, and audio—and is particularly effective for scenarios requiring real-time processing (Lugaresi et al., 2019).

The human pose recognition module (Pose) in MediaPipe enables efficient and accurate real-time key point detection, supporting 33 anatomical landmarks and

facilitating both 2D and 3D pose estimationCao and Yan (2024). A system combining MediaPipe with a convolutional neural network (CNN) was proposed to recognize static gestures in Mexican Sign Language (MSL) fingerspelling. Hand key point data extracted by MediaPipe were used as input for CNN-based training and classification, resulting in an accuracy of 83.63% on a test set of 336 images. The system operates in real time on low-power devices, demonstrating strong usability and portability. These results indicate the reliability of MediaPipe in human pose recognition tasks(Sánchez-Vicinaiz et al., 2024).

You Only Look Once (YOLO) is a unified, real-time object detection framework introduced in 2016(Redmon et al., 2016). It formulates object detection as a single regression problem, enabling simultaneous prediction of object categories and bounding box coordinates directly from input images. By dividing the image into grids, each responsible for predicting multiple bounding boxes and associated class confidences, YOLO achieves significant improvements in detection speed. This architecture established a new paradigm for real-time object detection and laid the groundwork for the development of successive YOLO versions.

YOLOv8 marked a significant milestone in the architectural modernization of the YOLO series(Hussain, 2024). It introduced an anchor-free mechanism, which simplified model configuration and enhanced both generalization and stability. YOLOv9 focused on improving accuracy through architectural reinforcement, although its performance on small object detection remained limited. YOLOv10 prioritized efficiency for deployment on resource-constrained devices but exhibited reduced accuracy in scenarios involving complex and overlapping targets(Zhou et al., 2022).

The YOLO algorithm suite has found widespread use across fields like self-driving technologies, security monitoring, and medical imaging, highlighting its versatility in real-time detection scenarios(Singh et al., 2022).

Although YOLO12 adopts a more complex and radical architectural design aimed at further enhancing detection performance, it has not demonstrated substantial advantages in real-world applications. The high computational cost combined with marginal performance gains has rendered it a relatively underperforming version within the YOLO series(Jegham et al., 2024).

YOLO11 presents notable enhancements over its earlier version, offering a more compact architecture, fewer parameters, improved feature representation, and accelerated inference speed. A recent investigation utilized a synthetically constructed dataset—produced through a large language model (LLM)—to develop apple detection models using both YOLO11 and YOLOv10. The study demonstrated that the integration of synthetic data can improve model robustness and generalization. On real-world orchard imagery, YOLO11 attained a detection accuracy of 0.84 and a mAP@50 score of 0.89, confirming its effectiveness in object detection tasks (Sapkota et al., 2024).

Given these results, YOLO11 was selected for the subsequent experimental phases. The architecture of YOLO11 has been significantly improved compared to previous versions. Through newly designed layers, advanced modules and a series of optimizations, the computational efficiency and detection accuracy have been improved. Its overall architecture can be roughly divided into three parts: backbone, neck and head. By upgrading the backbone and neck design, the model enhances

the feature extraction capability and achieves more accurate object detection while maintaining high efficiency and high processing speed.

Compared with previous versions, YOLO11 has improved accuracy while reducing the number of parameters. The model supports flexible deployment across multiple platforms, such as edge hardware, cloud infrastructures, and GPU-enabled systems. At present, YOLO11 finds application in diverse computer vision tasks, including object localization, instance-level segmentation, visual categorization, pose estimation, and oriented bounding box (OBB) recognition.

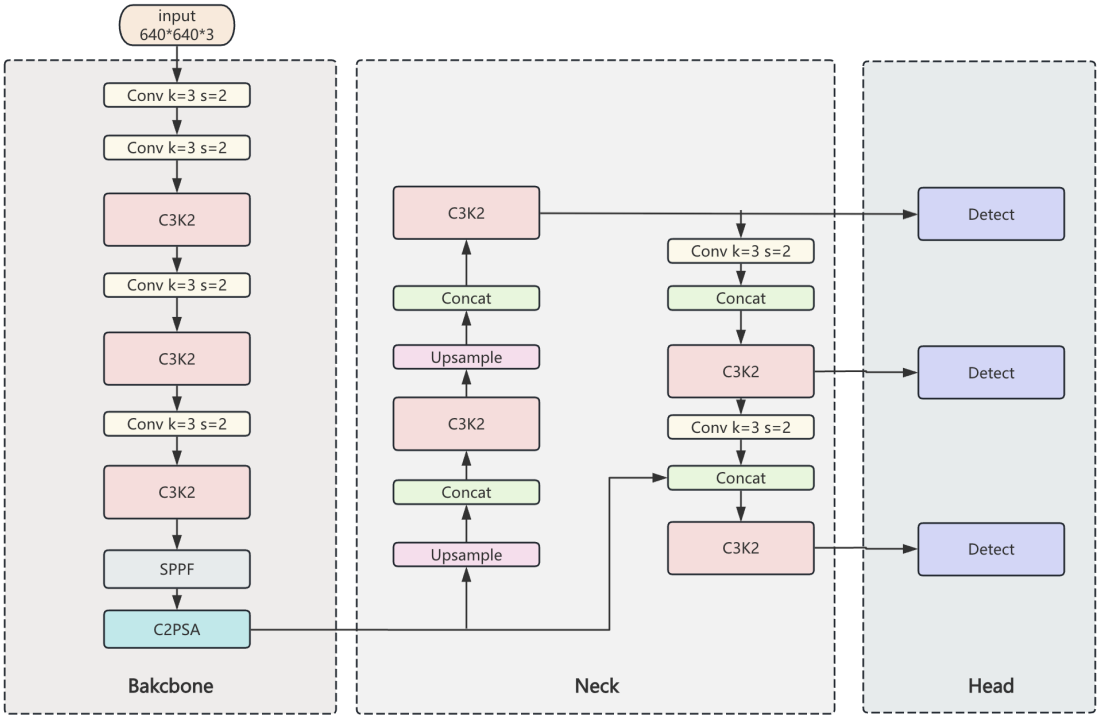

Figure 3.2: YOLO11 Architecture

From **Fig.**3.2, we get a general idea of its structure. The backbone module is responsible for extracting fundamental features from the input image. Functionally

analogous to a “visual nervous system”, this component enables the model to comprehend the visual content. YOLO11 incorporates the C3k2 block while retaining the Spatial Pyramid Pooling Fast (SPPF) module from the previous version(Dong et al., 2024). In addition, a new component, the C2PSA block, has been introduced(Wang et al., 2024). The C3k2 module serves as an optimized alternative to the CSP bottleneck, utilizing a pair of lightweight convolutions instead of one large kernel to improve the speed of feature representation. The SPPF module applies max pooling at multiple spatial scales and concatenates the resulting features, thereby capturing semantic information at different resolutions. The C2PSA (Cross-Stage Local Spatial Attention) module enhances spatial attention across the feature map, enabling the model to concentrate more effectively on critical regions within the image. By spatially pooling relevant features, this mechanism facilitates improved focus on salient areas.

The neck of YOLO11 is structured to aggregate feature maps with varying resolutions and deliver them to the detection head. In this version, the C2f block in the neck has been replaced with the C3k2 block. This adjustment contributes to increased efficiency and improved performance during the feature aggregation process.

The detection head in YOLO11 is tasked with generating the model’s final predictions. It comprises three detection layers, each dedicated to identifying objects of different sizes. The head outputs bounding boxes, class probabilities, and confidence scores, thereby producing the final classification results.

3.2.4 Attention Mechanism and Convolution Module

EMA: The EMA (Efficient Multi-Scale Attention) component functions as a compact attention strategy aimed at improving the representational strength of CNNs in applications like visual categorization and object recognition (Ouyang et al., 2023). Traditional attention mechanisms often suffer from high computational overhead or information degradation when capturing multi-scale and cross-spatial dependencies. In contrast, EMA addresses these challenges through the introduction of channel grouping and cross-spatial interaction strategies. By eliminating the channel compression process commonly found in conventional attention modules, EMA mitigates the risk of information loss. Furthermore, it strengthens representational capacity by employing lightweight cross-branch fusion to jointly model global and local attention.

Technically, EMA transforms selected channel dimensions into batch dimensions and organizes channels into separate groups, thereby enabling spatial semantic features to be consistently modeled within each subspace. This design helps preserve both local detail and global context. Additionally, EMA employs two parallel branches to process features at different scales, followed by feature fusion at the output stage. This approach enriches feature diversity and enhances accuracy. By adopting this architectural setup, EMA enhances the model's ability to attend to key visual areas, while concurrently lowering the parameter count and reducing overall computational burden. When compared to existing attention mechanisms such as Squeeze-and-Excitation (SE) (Hu et al., 2018) and the Convolutional Block Attention Module (CBAM) (Woo et al., 2018), EMA achieves a superior balance between

predictive precision and processing cost, resulting in significant gains in effectiveness while maintaining low resource demands.

In a recent study, an enhanced object detection model named EMA-YOLO was introduced (Liu et al., 2024), which aims to improve the detection accuracy of unripe yellow peaches. This model integrates the EMA module into the YOLOv8n framework to reinforce global information encoding and pixel-level feature aggregation.Empirical findings reveal that EMA-YOLO yields notable gains in mean average precision (mAP) across multiple benchmark datasets, validating the real-world applicability of EMA in detecting small-scale targets under complex environmental conditions.

To enhance the model sensitivity to the flag-bearing arm region and improve the quality of feature representations, the present study incorporates the EMA module into the YOLO network as an attention mechanism. According to the architectural design of YOLO11, the backbone is primarily responsible for feature extraction. Therefore, an EMA attention layer has been inserted after the C2PSA module in the backbone. This layer evaluates the relative importance of feature maps across channel dimensions and emphasizes critical target areas—specifically the arm region—in the spatial domain. With the guidance of this attention mechanism, the model becomes better equipped to differentiate foreground from background and exhibits increased robustness under complex visual conditions.

AKConv: AKConv (Arbitrary Kernel Convolution) represents an innovative convolutional operator aimed at addressing the structural constraints of traditional convolution kernels in terms of fixed sampling shapes and constrained parameter

quantities (Zhang et al., 2023). This operator enables sampling at arbitrary positions and supports a dynamic number of convolutional parameters. Through the integration of flexible coordinate generation and offset mechanisms, AKConv exhibits superior adaptability to varying target shapes and scales. In contrast to standard convolution operations, it enhances the expressiveness of learned features and leads to notable performance improvements in various multi-object detection tasks.

In a recent study, the traditional convolutional structure was replaced with the AKConv module in the design of APHS-YOLO (Liu & Su, 2024). This modification enhanced the model's capacity to extract features associated with complex object geometries. Owing to its flexible sampling mechanism, AKConv contributes to a substantial increase in detection accuracy and real-time responsiveness, while preserving the lightweight nature of the network.

3.2.5 DeepSeek

DeepSeek has emerged as one of the most prominent large language models in 2025. Since 2024, the DeepSeek research team has released a series of models and accompanying papers that have significantly advanced the field of natural language processing and multimodal learning.

The first notable release was DeepSeek-Coder in early 2024, a code generation model designed to handle programming-related tasks with high accuracy. Following this, DeepSeek-VL was introduced as a multimodal model incorporating a hybrid visual encoder. This model achieved state-of-the-art performance across multiple visual-language benchmarks.

In the domain of large-scale language modeling, DeepSeekMoE was launched as the foundation for DeepSeek's hybrid expert modeling approach (Tang et al., 2024). This model introduced novel mechanisms such as "shared expert isolation" and "expert specialized scheduling," which improved both routing efficiency and contextual understanding. It outperformed baseline models like LLaMA2 across various language comprehension tasks.

Building upon this foundation, DeepSeek-V2 was developed with 236 billion total parameters. It introduced the multi-head latent attention (MLA) mechanism to enhance contextual modeling capabilities, resulting in notable improvements in several language benchmarks.

The subsequent release, DeepSeek-V3, further increased the model scale and incorporated optimized expert scheduling and load-balancing strategies. According to the published results, DeepSeek-V3 demonstrated performance on par with GPT-4 and other closed-source models in mathematics, code generation, and reasoning tasks, indicating that its capabilities have reached a state-of-the-art level.

In early 2025, the DeepSeek-R1 model was introduced. This work focused on enhancing the reasoning abilities of large language models through a reinforcement learning (RL) based training paradigm. Unlike conventional approaches that rely on supervised fine-tuning (SFT), the research team first developed DeepSeek-R1-Zero, a model trained entirely through reinforcement learning without any human-labeled data (Zhang et al., 2025). Remarkably, advanced reasoning behaviors—such as chain-of-thought, self-verification, and reflection—emerged naturally from this training strategy.

Building upon this, the full DeepSeek-R1 model was proposed by incorporating cold-start data, multi-stage RL strategies, and refined re-supervised optimization techniques. These enhancements led to notable improvements in generalization and stability on complex reasoning tasks. Additionally, to demonstrate transferability, the reasoning capabilities of DeepSeek-R1 were distilled into smaller models such as Qwen-14B, 32B, and 70B, significantly boosting their performance on reasoning tasks. It is particularly noteworthy that the overall training cost of DeepSeek-R1 was substantially lower than that of comparable high-performance LLMs, demonstrating the cost-effectiveness and scalability of the reinforcement learning approach.

As a result, DeepSeek has become the most widely discussed large language model of 2025, expanding the impact of language models across a wide range of AI-driven applications and domains.

3.2.6 Text Embedding Model

Within Natural Language Processing (NLP), text embedding models function to encode linguistic features into a continuous vector representation, with the objective of preserving semantic features inherent in the text. This representation enables machines to interpret and process natural language more effectively. Text embeddings possess strong representational capacity and are capable of capturing semantic similarity and contextual relationships between words (Pawar & Gawande, 2023).

BGE-M3 is a general-purpose text embedding model released in 2024, developed to address multilingual, multi-functional, and multi-granular information retrieval scenarios. The model supports over 100 languages and demonstrates robust

multilingual and cross-lingual semantic understanding. In terms of retrieval functionality, it integrates three mainstream methods—dense retrieval, sparse retrieval, and multi-vector retrieval—thereby significantly enhancing the model's flexibility and adaptability in semantic representation.

With respect to input granularity, BGE-M3 is capable of processing up to 8192 tokens, making it suitable for diverse retrieval tasks ranging from short phrases to extended documents. Additionally, it offers efficient and unified embedding support for multimodal semantic understanding and large-model Retrieval-Augmented Generation (RAG) tasks. The introduction of BGE-M3 represents a significant advancement in the unified architectural design and cross-scenario adaptability of text embedding models.

3.2.7 Retrieval-Augmented Generation(RAG)

Driven by the swift evolution of Large Language Models (LLMs), Retrieval-Augmented Generation (RAG) has become a prominent framework for enhancing the knowledge representation capabilities of such models and improving the accuracy of real-time question answering (Arslan et al., 2024). The RAG architecture integrates an external document repository with a generative model, following a "retrieve-then-generate" workflow. This approach effectively mitigates challenges such as knowledge staleness, hallucinations, and limitations in context length often encountered in LLMs.

A standard RAG architecture is composed of two core modules. The first is the document retriever, which transforms the user query into an embedding vector and

retrieves the most relevant documents within a semantic vector space. The second component is the generator, which takes the original query along with the retrieved documents as contextual input to produce informative and contextually coherent responses (Gupta et al., 2024).

In recent years, the maturity of text embedding models and the development of efficient vector retrieval tools have significantly accelerated the adoption of RAG architectures. These systems have been widely applied in diverse scenarios, including intelligent question answering, enterprise knowledge management, and scientific research assistance. As a result, RAG has become one of the dominant technical frameworks for enhancing the capabilities of LLM-based application systems (Yu et al., 2024).

3.2.8 AI Application Tools

In order to improve the practical usability and streamlined deployment of Large Language Models (LLMs), lightweight LLM application development tools for both web-based and desktop environments have gained increasing popularity. These tools typically offer graphical user interfaces (GUIs) to facilitate interactive operations, enabling users to construct intelligent LLM-based applications with minimal technical barriers.

Representative platforms such as Open WebUI, Chatbot UI, and Dify.AI provide access to both local and cloud-based LLMs, supporting features including multi-model selection, multi-user session management, plugin integration, and persistent data storage. Among them, certain tools (e.g., Dify.AI) incorporate vector database

integration and document retrieval capabilities, making them particularly suitable for constructing enterprise-level knowledge question-answering systems. In contrast, platforms such as Open WebUI and the Chatbot UI series focus on lightweight conversational interfaces, ideal for locally deploying LLMs in simple environments.

Ollama is a lightweight runtime framework that supports the local execution of various mainstream open-source LLMs. It can be seamlessly integrated with the aforementioned web-based tools to accelerate the development of LLM-powered applications.

These tools are generally open source, customizable, and adaptable to a wide range of deployment scenarios. As a result, they significantly lower the entry barrier to adopting LLMs and promote the development and widespread adoption of LLM-driven applications (Zhang et al., 2024).

3.2.9 Profanity Check

When interacting with AI applications, users frequently impose more stringent requirements regarding content security, user experience, and compliance with legal and regulatory standards. To address these demands, the system must incorporate a robust mechanism for detecting sensitive language(Dwork et al., 2023). ChatFlags was developed with this objective in mind. Specifically, we integrated a fast and effective Python-based tool, **profanity-check**, to identify profane and offensive expressions in text. This tool employs a linear Support Vector Machine (SVM) model trained on a dataset of 200,000 manually annotated examples, encompassing both clean and inappropriate textual content. Although the model is relatively simple

in design, it demonstrates remarkable effectiveness, enabling ChatFlags to perform reliable profanity detection and thereby enhance the overall user experience.

3.3 Experiments

3.3.1 Dataset for Preliminary Experiments

The act of obtaining digital images by capturing physical objects is known as image acquisition Pratumgul and Sa-ngiamvibool, 2016. A camera was employed to capture videos of flag gestures corresponding to the 26 alphabet letters. These recordings were carried out in two contrasting settings: outdoor areas under sunlight and indoor spaces with white walls lit by artificial sources. Two participants were involved in the recording process. Each video lasted for approximately one minute and was captured at 60 frames per second.

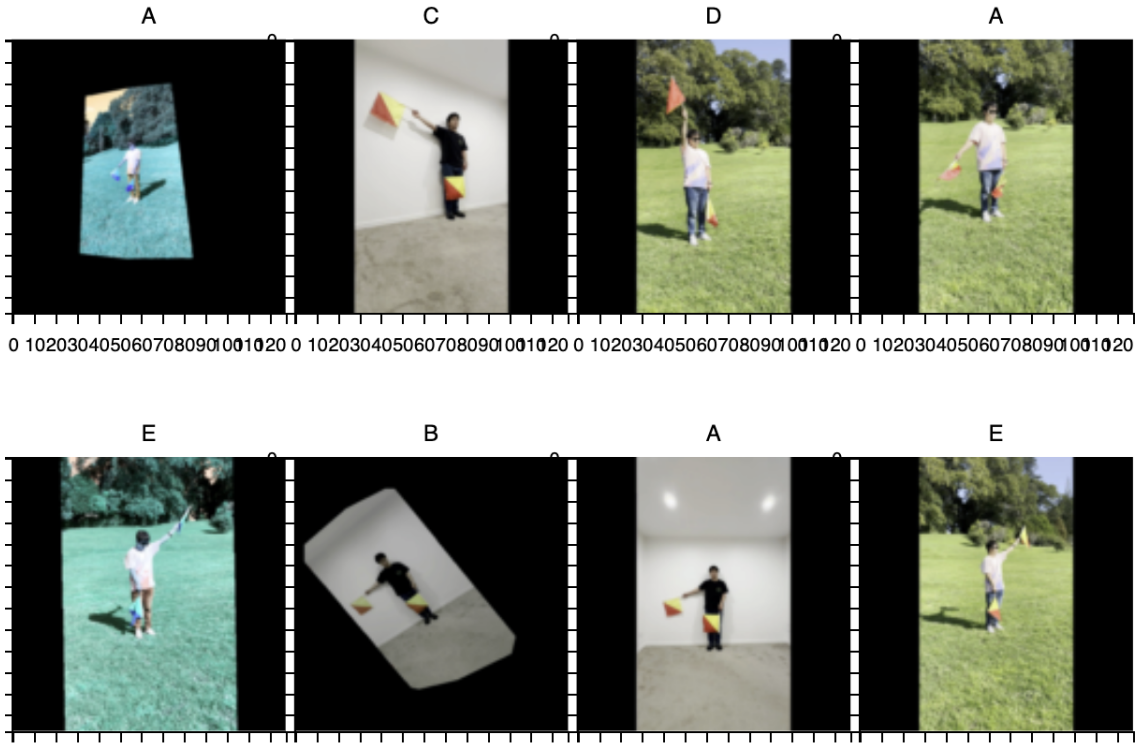

Figure 3.3: The Data Augmentation

Due to resource constraints, the experiment concentrated on recording videos for only the first five letters of the alphabet. To improve the model's capacity to generalize, various data augmentation strategies were employed. Frames were selected from the video footage at intervals of six frames to avoid redundant gestures. Following this, two augmentation methods were randomly applied to each extracted frame, including variations such as random rotation ($\pm 15^{\circ}$), resizing ($\pm 10\%$), brightness modification ($\pm 20\%$), sharpness tuning, and enhancement of contrast. These augmentation methods enriched the dataset variety and helped minimize misclassification of unfamiliar gestures. A representative enhanced frame is presented in Fig.

3.3. Furthermore, the original resolution of 1920 × 1080 was reduced to 640 × 640, producing a total of 11,813 frames, among which 3,075 were obtained without any augmentation.

The dataset was divided into training, validation, and test subsets using an 8:1:1 ratio, resulting in 8,241 images allocated for training, 1,775 for validation, and 1,797 for testing purposes. The same division scheme was applied to the dataset without augmentation.

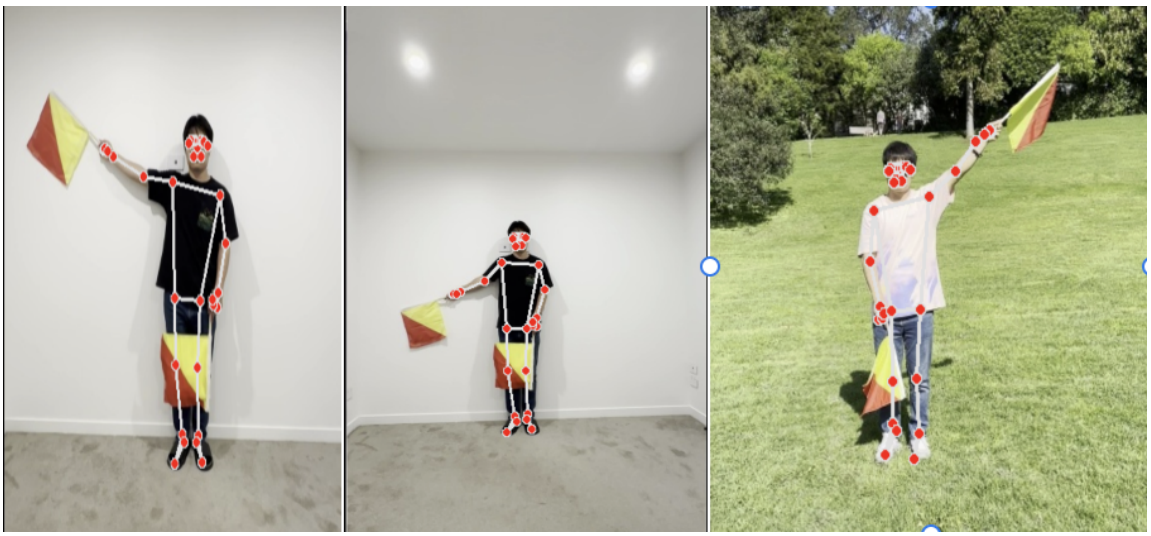

Figure 3.4: The Images of Human Poses

In this study, the MediaPipe pose module was adopted to extract arm-related posture features pertinent to flag gesture recognition. This module identifies 33 anatomical landmarks across the human body and returns their three-dimensional spatial coordinates. Given that flag gestures predominantly involve arm movements, six critical keypoints were selected: the left and right shoulders, elbows, and wrists. Each of these keypoints is represented by (x, y) coordinates that describe the spatial

configuration of the arm. These coordinates were saved alongside the corresponding image filenames and classification labels to facilitate model training. The results of this feature extraction process are presented in Fig. **3.4**. The dataset enriched with these pose features is termed the Feature Dataset (FData), whereas the version without pose data is designated as the Common Dataset (CData). Initially, annotation data for training the CNN was generated using labelImg. The bounding region for each character was coarsely estimated using the pose keypoints detected by MediaPipe, with line segments drawn within the image. Subsequently, a YOLO11 annotation file was created according to the image filename. Finally, labelImg was utilized to adjust the bounding boxes, ensuring precise alignment with the actual gesture locations. The finalized dataset was then used to train the YOLO11 model.

During training, the CNN model receives input formed by integrating gesture features with the corresponding raw image data. Prior to input, the images undergo normalization, and the gesture feature coordinates are scaled to fall within the range $[0, 1]$. This multimodal input strategy enables the model to simultaneously capture visual pixel-level details and spatial characteristics of arm movements, thus enhancing the accuracy of gesture classification.

3.3.2 Dataset for YOLO11

Video data acquisition was conducted using the DJI Action4 sports camera. Flag signal videos were recorded under diverse weather conditions and across multiple scene types, with several individuals performing the gestures. The recorded gestures covered the full set of alphabetical characters from A to Z as well as the STOP

action. In earlier experiments, image distortion caused by suboptimal camera angles and non-standard body postures was identified as a concern. To address this, frames were extracted at 10-frame intervals from the recorded videos. Postural landmarks, specifically at the elbow and wrist, were extracted using the MediaPipe framework. Images in which the posture could not be reliably identified were discarded.

The dataset was subsequently divided into training and validation sets in an 8:2 ratio. All images were resized to 640×640 pixels to comply with the default input dimensions of the YOLO training pipeline. At this stage, the training set comprised 17,569 images, while the validation set contained 4,403 images. The dataset from previous experiments was filtered, and only non-augmented data were retained to serve as the test set. This approach ensures the complete isolation of test data and maximizes the reliability of the model's predictive performance.

Various data augmentation methods, recognized for their success in computer vision tasks (Liu, Zhang, & Chen, 2023), were employed to address the limited number of samples available per category in the training set. In addition, manual augmentation was conducted to increase the dataset size and strengthen the model's ability to generalize.The applied transformations included random affine adjustments, vertical flipping, rotation, Gaussian noise addition, random background elimination, brightness modification, and Gaussian blurring. Horizontal flipping was deliberately excluded, as prior experiments revealed that this operation could result in mirrored postures corresponding to other gesture categories, thereby introducing prediction ambiguity. In addition, MediaPipe was employed to extract elbow and wrist coordinates, enabling the generation of hotspot annotations to visually emphasize critical

regions and improve the model's focus during training.

Annotation of the image data was also supported by MediaPipe. The experiment prioritized the elbow and wrist regions of both arms; finger keypoints were excluded from bounding box calculations. This decision effectively reduced the annotated area, minimized the inclusion of irrelevant features, and accelerated model convergence. For augmented images where MediaPipe assistance was ineffective, manual annotation was conducted. Ultimately, the training dataset was expanded to 34,566 images following data enhancement.

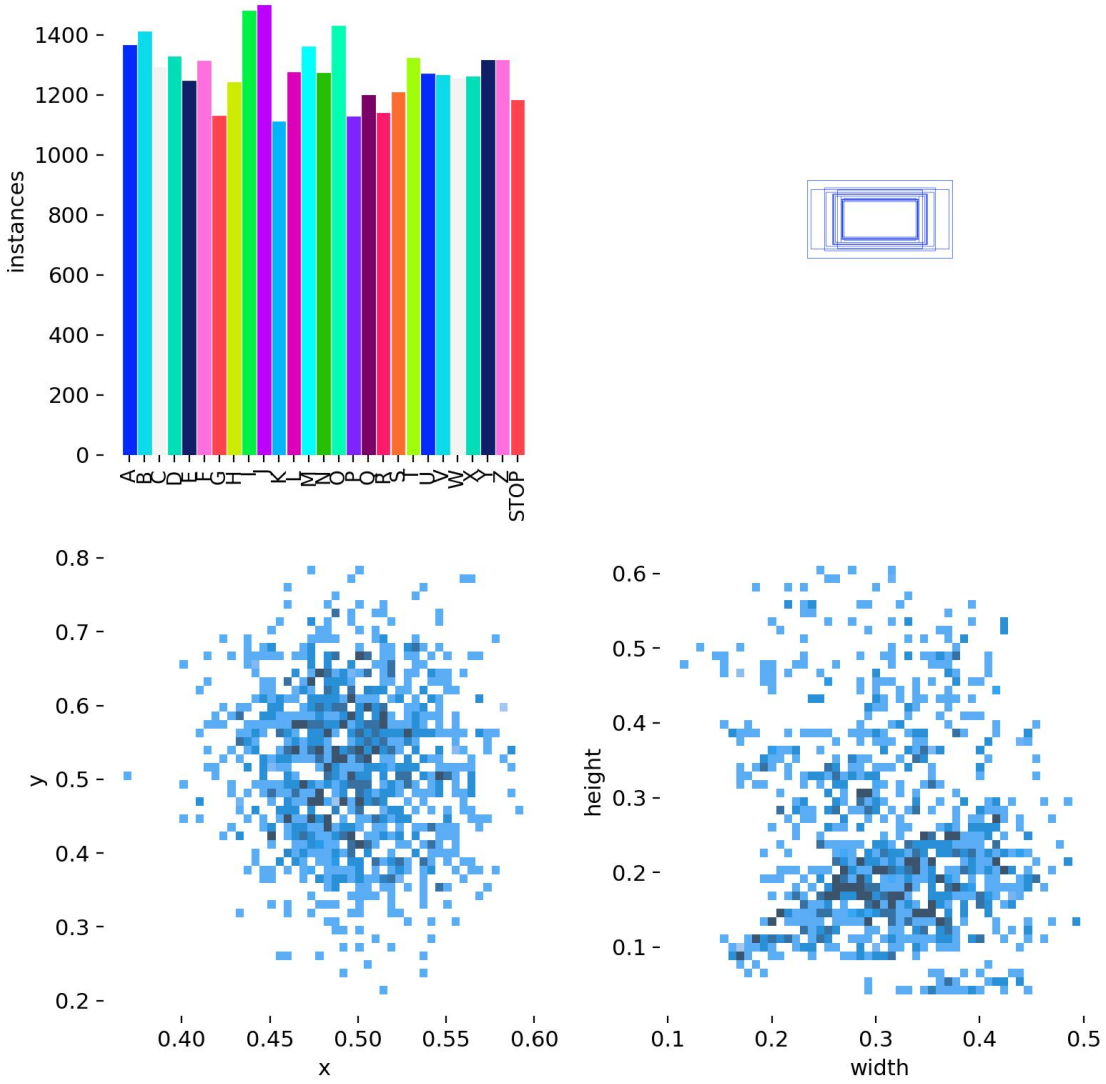

Figure 3.5: Labels Panel

As illustrated in **Fig.** 3.5, the label distribution and bounding box characteristics of the flag signal dataset can be observed. The dataset exhibits a relatively balanced category distribution, with a comparable number of instances across all

classes and no significant disparities among them. The center points of the bounding boxes are predominantly concentrated near the center of the images, suggesting a consistent distribution of target width and height. Furthermore, the absence of small targets indicates a high overall data quality. For the purposes of this experiment, the dataset demonstrates stable performance without any evident anomalies.

3.3.3 Evaluation Indicators

Model performance is commonly assessed using several metrics, such as accuracy, precision, recall, F1 score, receiver operating characteristic (ROC) curve, and confusion matrix.

Accuracy: Accuracy refers to the proportion of correctly predicted instances relative to the total number of predictions made.

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{3.1}$$

where TP indicates the count of positive instances that are correctly identified. TN refers to those negative instances that the model predicts correctly. FP represents negative samples that have been incorrectly labeled as positive, whereas FN refers to positive instances that the model misclassifies as negative (Chen, 2024a).

Precision: Precision refers to the proportion of true positive predictions among all instances identified as positive by the model.

$$Precision = \frac{TP}{TP + FP} \tag{3.2}$$

where TP denotes the count of actual positive cases that are correctly identified, while FP represents the number of samples that are falsely predicted as positive.

Recall: Recall measures the proportion of true positive instances correctly identified among all actual positive cases.

$$Recall = \frac{TP}{TP + FN} \tag{3.3}$$

where TP refers to the count of positive instances that the model correctly identifies, while FN represents the number of actual positive samples that are incorrectly classified as negative.

F1-Score: In terms of precision and recall, the F1-score is the harmonic mean.

$$F1 = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{3.4}$$

mAp: mAP is calculated by averaging the AP over all categories. To calculate AP, we need to interpolate the PR curve (precision - recall). Here's how AP is calculated:

$$P_{\text{interp}}(r) = \max_{r' \geq r} P(r') \tag{3.5}$$

where the interpolated precision corresponding to a specific recall threshold r is defined as the highest precision value observed at any recall level r' such that $r' \geq r$.

$$\text{AP}_{c,t} = \sum_{i=1}^{n-1} (r_{i+1} - r_i) \cdot P_{\text{interp}}(r_{i+1}) \tag{3.6}$$

where r_i and r_{i+1} denote adjacent recall values, and $P_{\text{interp}}(r_{i+1})$ is the interpolated precision at the right endpoint of each interval. This summation serves as a numerical approximation of the integral of precision over recall.

$$\text{mAP}_t = \frac{1}{C} \sum_{c=1}^{C} \text{AP}_{c,t} \tag{3.7}$$

where C is the total number of object classes, and $\text{AP}_{c,t}$ denotes the Average Precision for class c at IoU threshold t. This formulation provides a comprehensive measure of the model's ability to detect objects of different categories at a specified localization accuracy level. Commonly used settings include $t = 0.5$ (mAP@50) and the COCO-style average over multiple thresholds from 0.50 to 0.95 (mAP@50:95), the latter offering a stricter and more robust performance evaluation.

For this experiment, the subject of the flag signal is the human body, and the overall feature change is not very large. The difference lies in the posture of the arm. The model needs to be able to accurately detect and mark the position of the arm. Therefore, this experiment focuses more on mAP@50:90

Confusion Matrix: The confusion matrix offers a detailed representation of the model's performance across different classes by illustrating the correspondence between predicted outputs and ground truth labels. Its normalized form is commonly employed to emphasize regions of strong classification accuracy as well as areas where misclassification may occur, thereby assisting in identifying particular categories that require improvement.

Each evaluation metric contributes a distinct viewpoint on the system's effectiveness, supporting a thorough and reliable analysis of how well the model distinguishes

among various flag categories.

3.3.4 Experiments for Flag Recognition

In this study, the YOLO11 was applied to train the dataset comprising labeled instances. The PyTorch framework served as the foundation for implementing multiple model architectures. These included three models based solely on image input, and three additional models capable of processing both MediaPipe-derived features and image data in parallel. Specifically, the architectures were defined as Simple CNN (S-CNN), ResNet CNN (R-CNN), DenseNet CNN (D-CNN), Pose-enhanced Simple CNN (PS-CNN), Pose-enhanced ResNet CNN (PR-CNN), and Pose-enhanced DenseNet CNN (PD-CNN).

Two types of datasets were utilized in the training process: CData, consisting exclusively of image data, and FData, combining MediaPipe posture features with corresponding images. Prior to model training, standardized normalization and augmentation procedures were applied. Learning rate, optimization method, loss function, number of epochs, and other hyperparameter settings were kept uniform across all configurations. The final set of models trained included: S-CNN, R-CNN, D-CNN, PS-CNN, PR-CNN, and PD-CNN.

All six CNN-based model groups were trained using identical hyperparameter configurations. Cross-Entropy was selected as the loss criterion, and the Adam algorithm was applied for optimization. The initial learning rate was set to 0.001 and adjusted dynamically during training, with a decay factor of 0.1 applied every 10 steps. Each model was trained for 100 epochs, using images resized to a resolution

of 128×128. For the YOLO11 model, training was also conducted over 100 epochs, but with input dimensions set to 640×640.

The 27-class flag signal recognition experiment based on YOLO11 were implemented using Python 3.11.6 and PyTorch 2.6.0. The hardware configuration includes an NVIDIA GeForce RTX 4070 Laptop GPU with 8188 MiB of memory, as well as an AMD 7940HX processor. All YOLO models adopted in this study belong to the nano variant category. Specifically, YOLO11, YOLO12, and YOLO-AKEMA were employed for comparative evaluation. The corresponding hyperparameter settings are detailed in the Table 3.1.

Table 3.1: Hyperparameter settings used in model training experiment

Hyperparameter	**Configuration**
Optimizer	SGD
Batch Size	32
Epoch	50
Image Size	640×640
Learning Rate	0.01
Workers	16
Flipud	False

3.3.5 Experiments for ChatFlags

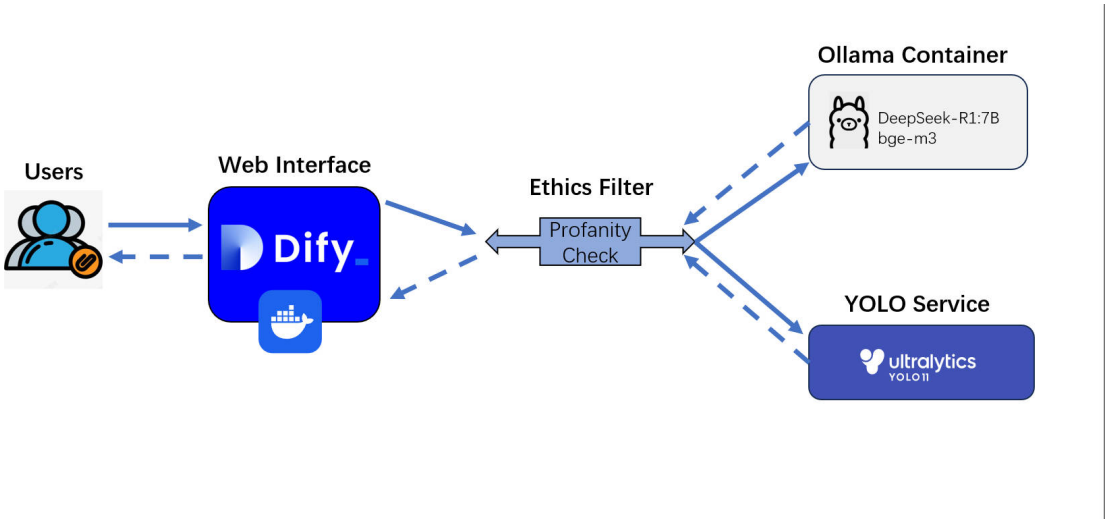

Figure 3.6: ChatFlags System

The overall system architecture is illustrated in the **Fig.**3.6. Dify, deployed via Docker, serves as the user interaction interface, while the large language model deployed in Ollama ensures accurate understanding and processing of natural language input. The text embedding model, integrated with Dify's knowledge base functionality, constructs a retrieval-augmented generation (RAG) system for flag language knowledge, supporting question-and-answer tasks related to flag language texts.

This study's semaphore image and video display system operates on a pre-constructed and fixed atlas of semaphore images. The system first standardizes the input sentence by removing punctuation, normalizing capitalization, and filtering non-alphanumeric characters. The processed character sequence is then sequentially mapped to the corresponding semaphore images in the atlas to generate a complete

pose sequence. When video output is required, the system composes these static images into a frame sequence with fixed temporal intervals and produces a continuous semaphore demonstration using standard video encoding tools. This approach eliminates the need for pose rendering or image synthesis, thereby improving the stability and efficiency of semaphore text visualization.

The visual model operates as an independent module, responsible for recognizing flag gestures and conducting analysis, with the results communicated back to the user interface. As an interactive system, ethical considerations are prioritized, and a swear word filtering mechanism has been incorporated to ensure responsible user interaction. The following content will detail the design and implementation of each system component.

The action recognition of flag gestures is initially implemented applying the YOLO-AKEMA model. To obtain accurate predictions while ensuring a sufficient number of recognition results, the confidence threshold is set to 0.85 when analyzing flag videos containing textual information. Rather than selecting a higher threshold, this setting strikes a balance between precision and recall. Prediction is conducted in half-precision and streaming computation mode to optimize resource efficiency. To accelerate the processing, prediction is performed at four-frame intervals instead of on every frame. Under this strategy, a 30-second video can yield over 300 prediction results.

Subsequent data processing is carried out based on the obtained prediction results. The first step involves segmenting each letter based on the appearance of the STOP gesture, which serves as a delimiter to approximate the number of letters in

the video sequence. While the specific letter corresponding to each segment remains uncertain at this stage, each segment is associated with a set of predictions. During segmentation, key data such as prediction confidence, category label, and the associated image are extracted to facilitate downstream processing.

In the second step, each group of prediction results corresponding to a single letter position is further analyzed. The results are grouped according to predicted category names, and the frequency of each predicted class is counted. This step provides a statistical basis for identifying the most probable gesture class for each action in the video, while accounting for potential model errors. Therefore, this stage focuses on temporary grouping and counting without making final determinations.

The third step aims to identify the most reliable frame for each predicted action segment. Structural Similarity Index (SSIM) is applied to evaluate the consistency of brightness, contrast, and structural information across the images within each group. The SSIM threshold is set to 0.95 to ensure high visual similarity. To accommodate videos with variable lengths and inconsistent sample counts, the frame interval and SSIM threshold repetition count are dynamically adjusted. This approach ensures that a stable and representative image is extracted for each flag action. The underlying assumption is that within any given gesture sequence, there exists a relatively stable period where the gesture is held clearly, which yields the most standard and reliable prediction result.

In the final step, each stable image is analyzed to calculate the arm angle and direction. Based on the extracted coordinates of the elbows and wrists from both arms, unit vectors are computed. By comparing the magnitudes of the X and Y

components of these vectors, the approximate direction of the arm. The direction, such as upward, downward, leftward or rightward, is determined accordingly. Directional offsets are further introduced to refine this classification. Verification rules are established based on predefined characteristics of flag gestures, allowing the system to reconfirm the prediction and output an interpretable flag gesture description for each image.

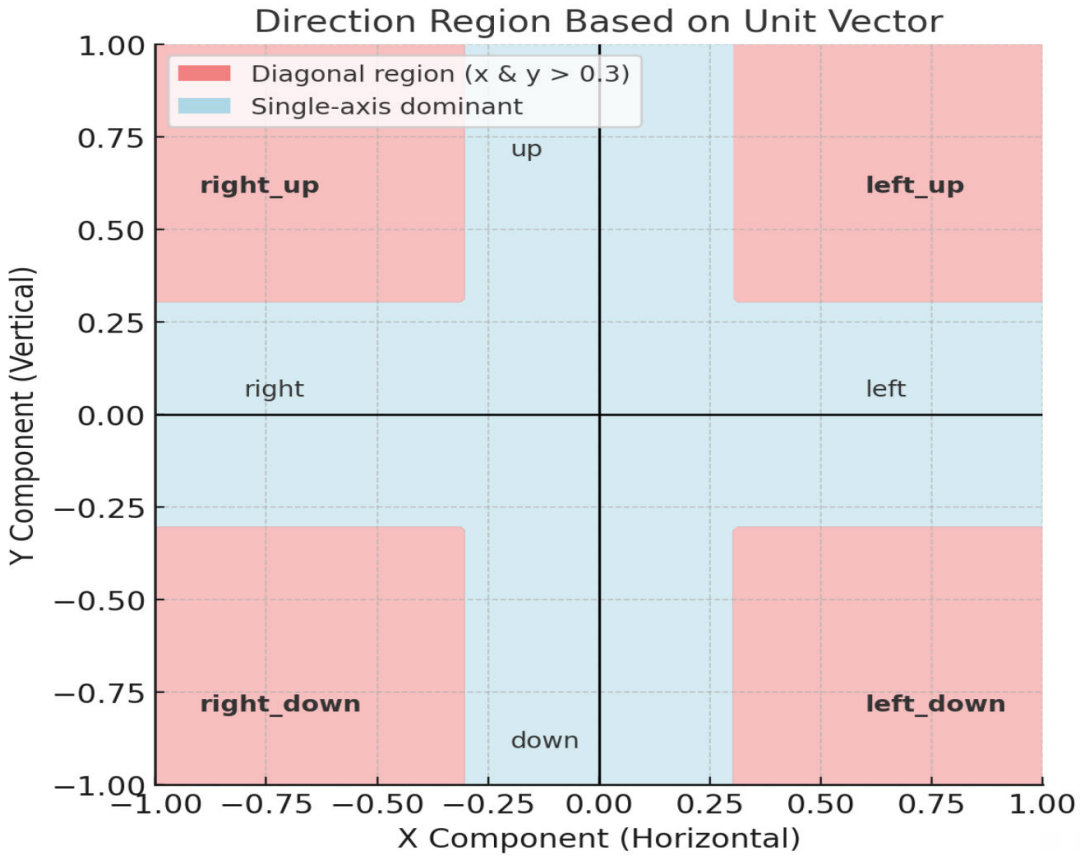

Figure 3.7: Coordinate Image for the Arm Direction

As illustrated in **Fig.** 3.7, when the offsets of both the X and Y components exceed a predefined threshold, the arm is considered to be pointing toward the center

of the corresponding quadrant. Otherwise, the direction is classified along the axis. Through this method, the final orientation of both arms can be determined.

The arm angle is calculated by comparing the direction vector of each arm with a vertically downward reference vector. This approach enables consistent and interpretable estimation of gesture orientation across different frames.

To maintain system responsiveness and ensure efficient batch data processing, multi-threaded computation is implemented. For a 30-second video, the average analysis time is approximately 16 seconds. In addition, caching mechanisms are employed after the initial analysis to prevent redundant computation when the same video is processed multiple times.

To ensure system responsiveness and operational efficiency during batch data processing, multi-threaded computation is employed. For a 30-second video, the average analysis time is approximately 16 seconds. To further optimize performance, analysis results are cached upon completion, thereby avoiding redundant computations for repeated video inputs.

This experiment employed the deepseek-r1:7b model, which, along with the BGE-M3 model, was deployed locally via the Ollama runtime framework and accessed through Dify using an API interface. A knowledge base for flag language was constructed utilizing Dify's Retrieval-Augmented Generation (RAG) system. First, this document includes the fundamental concepts, historical context, and azimuth reference system of flag signals to establish the conceptual foundation required for semantic understanding. Second, all flag signal letters were documented in a unified specification that includes left and right hand angles, gesture descriptions, salient vi-

sual cues, and common misinterpretations. Each letter was further decomposed into an independent knowledge unit to support precise retrieval and semantic matching. In addition, rule-based elements—such as signal mode transitions and action specifications—were organized separately to maintain a clear and hierarchical knowledge structure.

Regarding documentation format, this study adopted a structured representation combining Markdown and JSON. The knowledge content was partitioned into semantically independent units to reduce noise and enhance vector retrieval accuracy. The resulting knowledge base integrates fundamental concepts, a gesture dictionary, a rule system, and typical errors, enabling the RAG system to access consistent, standardized, and verifiable flag-signal information during response generation. This organization substantially improves the system's reliability and domain professionalism. Flag language knowledge was organized into structured documents, vectorized using BGE-M3, and subsequently stored in a local vector database. During interaction with DeepSeek within Dify, queries involving flag language content are directed to this knowledge base to retrieve relevant information.

In the development of Large Language Model (LLM) applications, interaction design is critical—particularly in accurately identifying user intent and problem classification through natural language. Dify facilitates this process through its chatFlow framework, which supports diverse user needs via a modular architecture comprising various functional nodes. For instance, LLM nodes can execute custom instructions to perform specific tasks such as information extraction, question classification, and text summarization. Knowledge retrieval nodes are capable of accessing the flag

language knowledge base, while HTTP nodes enable external communication with third-party services. In this experiment, HTTP nodes were utilized to complete the information exchange with the visual recognition module.

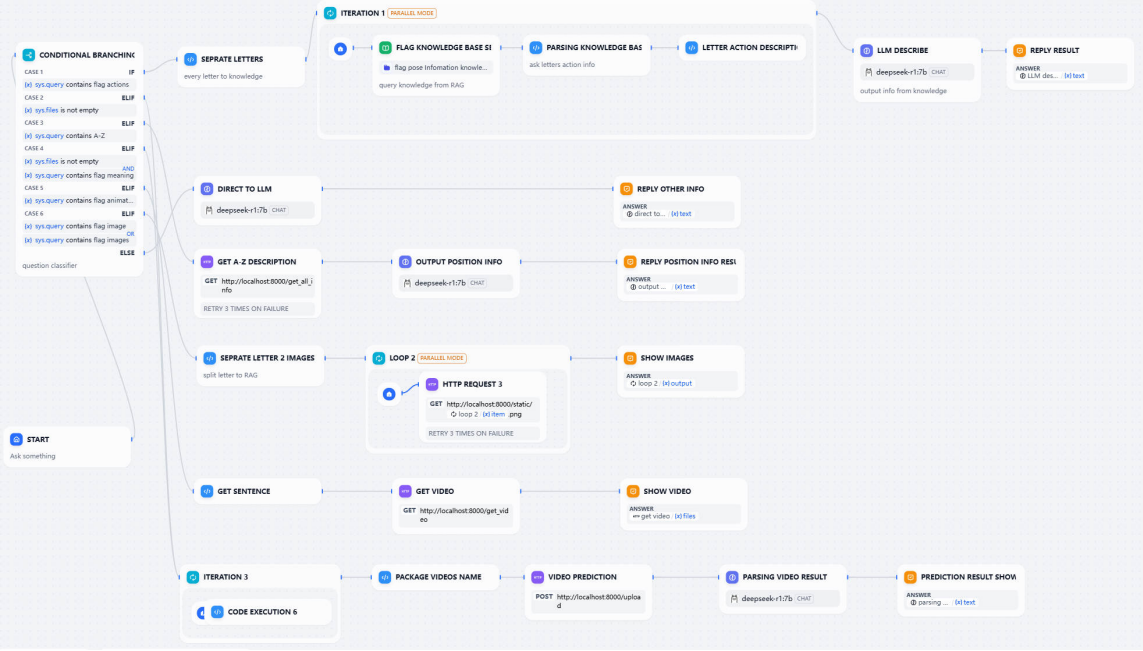

Figure 3.8: Dify ChatFlow Architecture

As shown in Fig.3.8 , the ChatFlags system, a series of tasks are executed through the orchestration of multiple functional nodes. The overall workflow encompasses a variety of task types, including video analysis, video generation, image synthesis, and textual description. All of these tasks are supported by the DeepSeek model, which serves as the core language reasoning engine.

It is important to note that when configuring the LLM node, task-specific instructions must be carefully designed to address different functional requirements. Iterative instruction refinement and continuous debugging are often necessary to

achieve optimal output quality. This process plays a critical role in ensuring the correctness and fluency of responses across different interaction scenarios.

3.3.6 Ablation Experiments

Ablation studies serve as an essential analytical tool for assessing the individual contributions of different modules, components, or strategies within a given model or system. By selectively modifying particular elements or hyperparameters while holding other factors constant, their specific roles and significance can be quantified based on performance evaluation metrics. This methodology assists in identifying parts that substantially boost model effectiveness and exposes superfluous structures, thereby promoting architectural simplification. Furthermore, ablation analysis supports the validation of design rationality, enhances model interpretability, and facilitates a deeper comprehension of the operational mechanisms underlying the model.

This research conducts a series of ablation experiments on the YOLO11 model, focusing on four primary variables: training epochs, batch size, data augmentation strategies, and the use of Mosaic enhancement. The model is tested under multiple settings, with epoch values configured at 30, 50, 80, and 100, and batch sizes set to 8, 16, and 32. Furthermore, the effect of enabling or disabling Mosaic augmentation, along with the integration of general augmentation methods, is thoroughly investigated to determine their respective contributions to model performance.

Ablation experiments were conducted to evaluate the individual contributions of the attention mechanism, AKConv convolution module, and horizontal flip parame-

ter in data augmentation within the YOLO-AKEMA framework. Each experiment was designed to isolate the effect of a single component through controlled variable settings. The results were analyzed to assess the specific impact of each module on overall model performance.

4 Data Analysis

4.1 Results for Preliminary Experiments

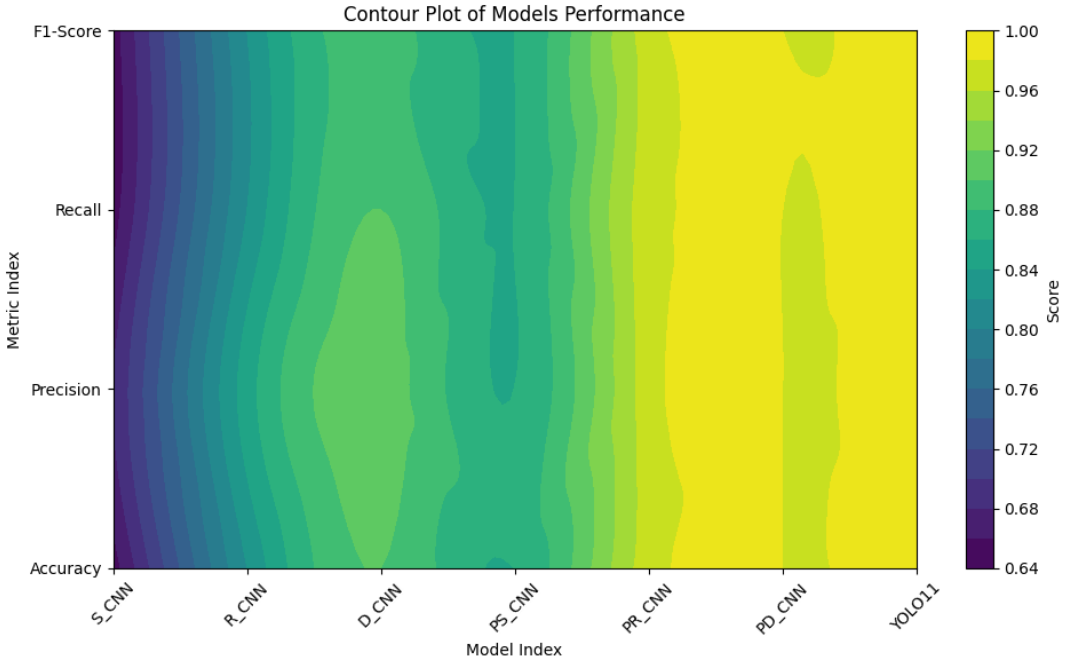

Figure 4.1: Model Performance Comparisons

As illustrated in Fig. 4.1, S_CNN, R_CNN, and D_CNN represent models trained exclusively on image data, without the inclusion of MediaPipe pose information. In contrast, PS_CNN, PR_CNN, and PD_CNN incorporate both image inputs and pose data during training. Comparative analysis indicates that integrating human pose information as auxiliary input substantially improves model effectiveness. Models lacking pose supervision depend entirely on extracting all visual features indepen-

dently, learning spatial patterns without guidance. This increases the difficulty of both feature extraction and prediction, as the model becomes more susceptible to external disturbances such as background clutter, illumination inconsistency, and inaccuracies in pose estimation. By fusing pose data, the model is guided to focus more directly on relevant human-related features.

Incorporating pose data enables the model to focus more effectively on human posture cues embedded within the training samples, thereby diminishing the negative influence of image noise. As a result, model performance is notably enhanced. These results underscore the importance of multi-modal training inputs in improving the model's capacity to rapidly capture key features, which in turn helps to reduce both computational complexity and training cost. Furthermore, the annotation strategy adopted by YOLO11 diverges from that of the remaining three CNN-based models. Specifically, YOLO11 relies on rectangular bounding box coordinates to define the relevant image regions, associate them with the appropriate category, and specify their spatial position and size. This annotated data acts as supervised guidance, assisting the model in the accurate identification and localization of the target. By filtering out irrelevant background content, this method allows the network to concentrate more precisely on distinctive visual traits, thereby enhancing recognition efficiency. Consequently, YOLO11 exhibits more favorable performance metrics relative to the remaining models.

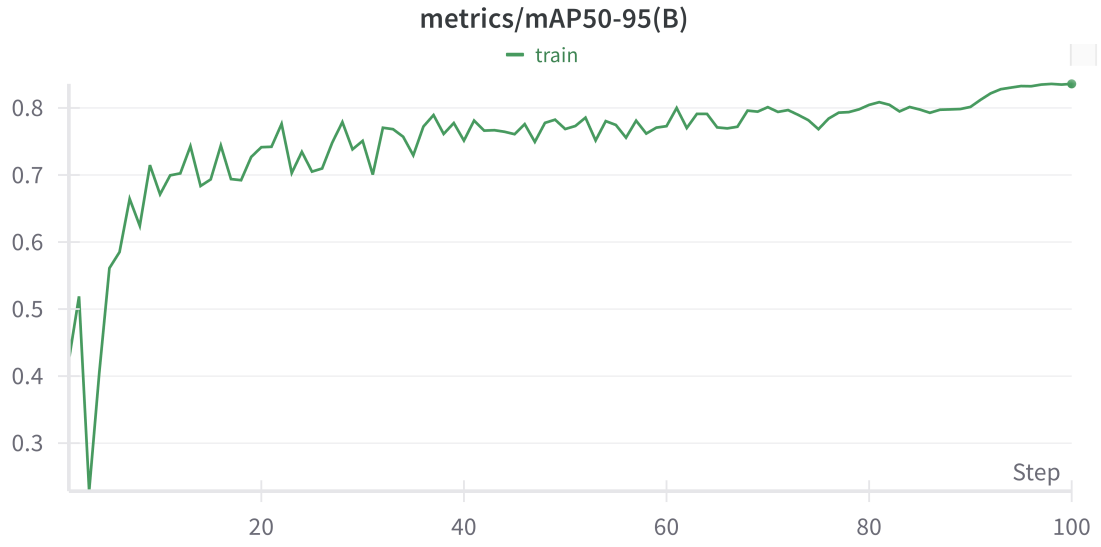

Figure 4.2: YOLO11 mAP50-95

Fig. 4.2 illustrates the variation in mAP50–95 for Class B gestures during the training of the **YOLO11** model. Initially, the score rapidly increased from 0.3 to 0.7, indicating a marked enhancement in the model's detection performance for Class B. Between steps 20 and 60, the metric exhibited fluctuations, reflecting the model's continued adaptation to the data. After step 60, the mAP50–95 gradually stabilized at approximately 0.8, suggesting that the model's accuracy for Class B gestures had converged to a consistently high level.

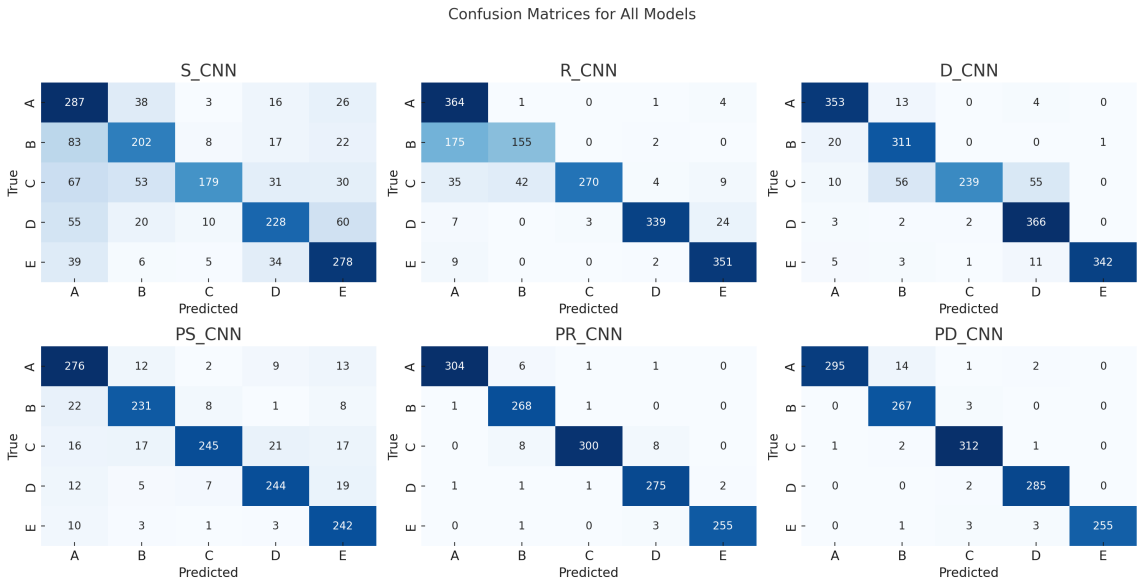

Figure 4.3: Confusion Matrix for All Models

As shown in Fig.4.3, both the S_CNN and PS_CNN models exhibit substantial confusion in distinguishing between Class B and Class C gestures. The S_CNN model frequently misclassifies Class B gestures as either Class A or Class C, and often identifies Class C gestures as Class B. This misclassification primarily results from the visual similarity between these two gesture categories, a problem further exacerbated by variations in camera angles. The confusion matrix analysis of the R_CNN and PR_CNN models indicates that PR_CNN achieves markedly better classification performance than R_CNN, particularly for Class B and Class C gestures. Similarly, the PD_CNN model outperforms the D_CNN model and other baseline models in gesture classification tasks. Incorporating MediaPipe-derived posture features significantly enhances classification accuracy by providing additional geometric positional information.

Consistent improvements are observed across these three groups of models, with notable reductions in misclassification rates. Specifically, for the D_CNN and PD_CNN models, the number of misclassifications for Class B decreased from 20 to 0, and for Class C from 56 to 2, underscoring the critical role of posture features in accurately distinguishing complex gestures. Furthermore, the PD_CNN model exhibits greater stability across other gesture categories, particularly for Classes D and E, where misclassifications are minimal.

Overall, these findings demonstrate that the integration of human posture information substantially improves model accuracy and robustness in gesture classification.

Table 4.1: Confusion Matrix for YOLO11 and PD_CNN

	YOLO11					**PD_CNN**				
	A	**B**	**C**	**D**	**E**	**A**	**B**	**C**	**D**	**E**
A	354	0	0	0	0	295	14	1	2	0
B	0	302	0	0	0	0	267	3	0	0
C	0	1	342	0	0	1	2	312	1	0
D	0	0	0	340	0	0	0	2	285	0
E	0	0	0	0	309	0	0	1	3	255

As shown in Table 4.1, **YOLO11** achieves strong performance on the test set, with only one instance of category B being misclassified as category C. In contrast, the PD_CNN model exhibits slightly higher confusion, particularly in cases where category A is misclassified as category B, and category C is incorrectly identified

as other adjacent categories. The relatively greater number of misclassifications observed in PD_CNN suggests that it encounters more difficulty in distinguishing categories with overlapping or visually similar features. **YOLO11**, on the other hand, demonstrates superior generalization capability and reduced inter-class confusion, making it a more reliable candidate for flag gesture classification tasks.

Furthermore, ablation experiments conducted on the YOLO11 model provide valuable insights into the effects of different training variables, including the number of epochs, batch size, data augmentation strategies, and mosaic enhancement. These factors are found to have a significant influence on the model's overall performance.

Table 4.2: YOLO11 Ablation Experiments Results

Ep	**Batch**	**Mos**	**Aug**	**Acc**	**P**	**R**	**F1**	**mAP0.5**	**mAP0.5-0.95**
100	16	✓	✓	0.9994	0.9994	0.9994	0.9994	0.9914	0.8348
80	16	✓	✓	0.9912	0.9912	0.9912	0.9912	0.9912	0.8353
50	16	✓	✓	0.9988	0.9988	0.9988	0.9988	0.9901	0.8305
30	16	✓	✓	0.8816	0.8841	0.8816	0.8816	0.8411	0.5911
50	16	✓	✗	0.8044	0.8393	0.8044	0.8101	0.7603	0.5453
50	16	✗	✓	0.9951	0.9952	0.9951	0.9951	0.9904	0.8318
50	32	✓	✓	0.9939	0.9942	0.9939	0.9939	0.9888	0.8329
50	8	✓	✓	0.9971	0.9971	0.9972	0.9971	0.9893	0.8261

As presented in Table 4.2, the model achieves an accuracy and precision of 0.9994 and a mAP@50 of 0.9914 when trained for 100 epochs. The performance

difference between 50 and 100 epochs is relatively small. However, a notable decline in performance is observed when the number of epochs is reduced to 30, indicating that 30 epochs lead to underfitting. Therefore, training the model for 50 epochs is sufficient, as further increases to 80 or 100 offer only marginal performance gains.

In the batch size experiments, a batch size of 16 yields the highest performance. Although batch sizes of 8 and 32 result in slightly lower accuracy, the overall effect of batch size on the model's performance remains limited. This suggests that batch size is not a dominant factor in this experimental setting.

The mosaic augmentation method, which merges four images into one to incorporate multiple targets, is generally effective for detecting small objects. However, in this study, enabling mosaic augmentation results in only marginal performance gains. This limited impact is likely attributable to the relatively large size of the flag targets, which diminishes the utility of mosaic augmentation.

In contrast, conventional data augmentation techniques, including rotation and brightness adjustment, substantially improve model performance. Experimental results indicate that training with the enhanced dataset yields significantly better outcomes, with accuracy decreasing from 0.9988 to 0.8044 and mAP@0.5 dropping considerably when augmentation is disabled. These findings underscore the importance of data augmentation in promoting model generalization and maintaining dataset diversity.

Based on the results of the four groups of experiments, it can be concluded that, for the flag classification task, 50 training epochs are sufficient, and a batch size of 16 yields the optimal performance. The mosaic augmentation technique exhibits

limited effectiveness, which is likely due to the relatively large size of flag targets. In contrast, the absence of conventional data augmentation methods, such as rotation and brightness adjustment, leads to a substantial decline in model performance, emphasizing their critical role in promoting data diversity and improving generalization ability.

Furthermore, although the highest mAP@50–95 exceeds 83%, the overall gain is modest. Unlike mAP@0.5, which primarily evaluates detection accuracy, mAP@0.5–0.95 introduces stricter requirements on both detection quality and localization precision. The limited improvement observed may be attributed to annotation inconsistencies or background interference, which could hinder precise object localization. These issues will be prioritized in future optimization efforts.

From the analysis of the experimental results in this chapter, YOLO11 has the highest accuracy in the five types of flag signal actions. At the same time, mediaPipe can not only provide assistance in experimental data processing, but its extracted posture data can also be provided as multi-modal data to model training to help the model learn and improve performance. Through the ablation experiment, we can see the obvious role of the data augmentation dataset, which is of great help for subsequent experiments. At the same time, some problems were also found. Through the analysis of the confusion matrix, there is confusion caused by the distortion of B and C due to the shooting angle, and there are errors between C and D that may be caused by changes in arm posture due to horizontal flipping. For these misclassifications, we need to pay special attention to them in the subsequent experiments on the dataset of all letters, and we also need to improve the quality of the dataset

to ensure the smooth progress of subsequent experiments.

4.2 Results for YOLO11

Based on the experimental results, the model that achieved the most favorable performance was YOLO-AKEMA. A qualitative analysis will be carried out from multiple perspectives to further interpret and evaluate the outcomes of the experiment.

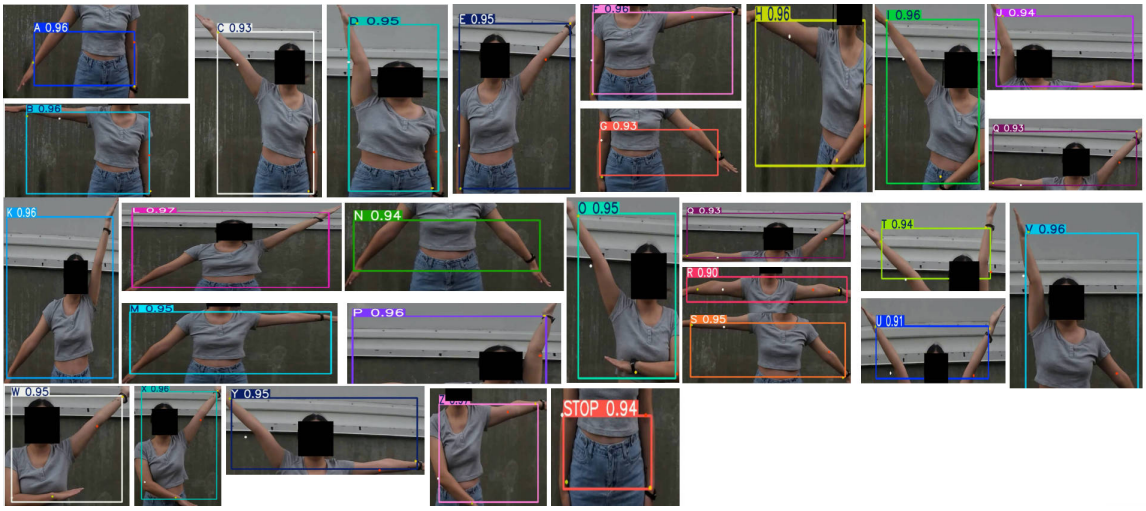

Figure 4.4: Predictions for 27 categories

From Fig. 4.4, the image illustrates the results obtained from applying the model to a flag action video. It is evident that all categories have been accurately identified, with no instances of multiple predictions corresponding to a single gesture. This level of real-time prediction capability is both effective and essential. The results demonstrate that the model possesses sufficient reliability for practical deployment in flag recognition tasks. High predictive accuracy serves as a fundamental prerequisite

for subsequent flag video analysis.

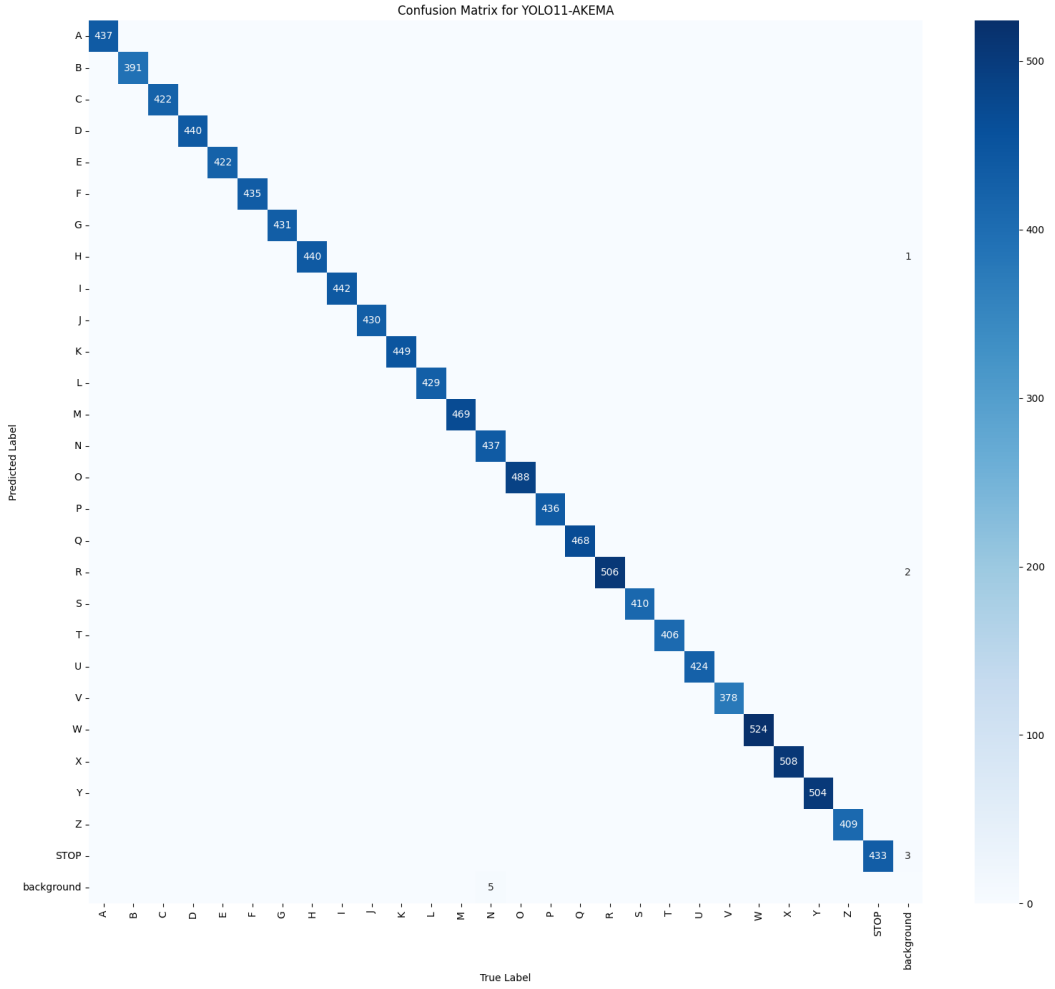

Figure 4.5: Confusion Matrix for YOLO-AKEMA

From Fig.4.5, the confusion matrix shows that the model has an overall excellent classification performance on 28 categories (including A-Z, STOP and background). The prediction results of most categories are concentrated on the diagonal, indicating that the model has a high accuracy rate. There are several classification errors in H, R, and N, which may be caused by image quality, but the overall performance is

good and the model has good classification ability.

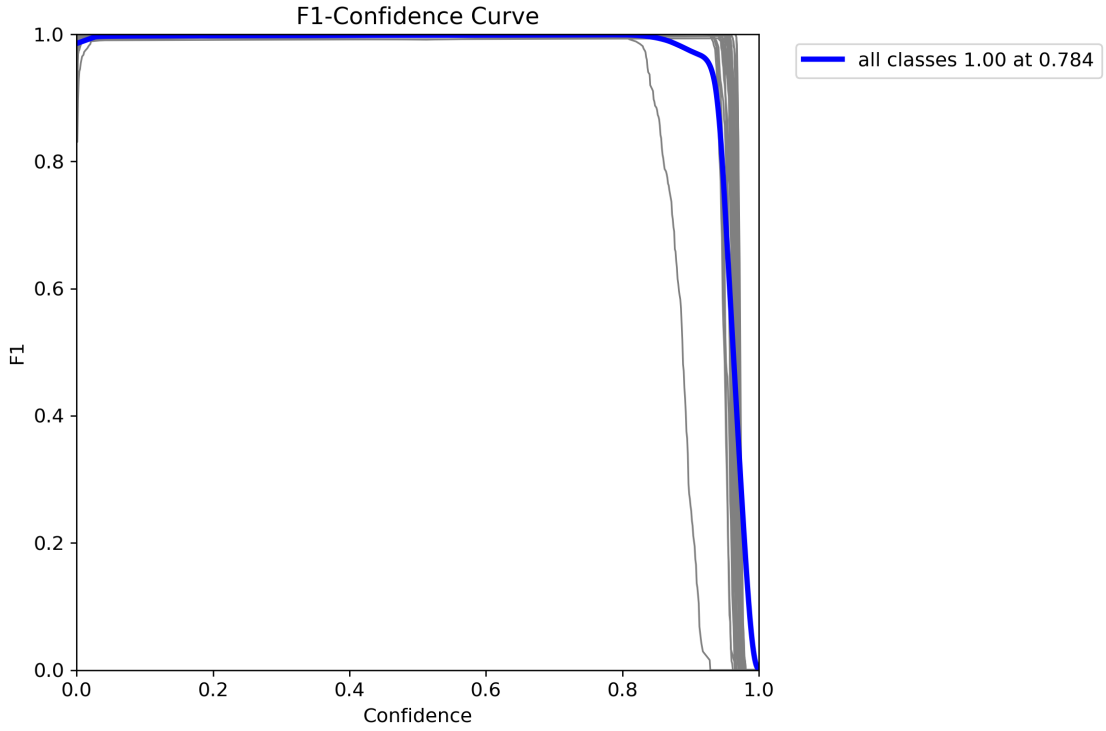

Figure 4.6: F1-Confidence Curve for for YOLO-AKEMA

As shown in Fig. 4.6, the F1-Confidence curve illustrates that the model consistently maintains an F1 score close to 1.00 across a wide range of confidence thresholds, demonstrating exceptional classification stability and generalization capability. When the confidence threshold is set to 0.784, the overall F1 score reaches the optimal value of 1.00, indicating that the model achieves an ideal balance between precision and recall under this threshold. This characteristic is particularly important for real-world deployment, as it enables effective suppression of low-confidence false positives while maximizing the model's responsiveness to correctly identified

instances.

The gray curves in the figure represent the F1 score variations of each individual category under different confidence thresholds. These curves appear densely concentrated and generally maintain high F1 values without any notable decline. This observation reflects the model's balanced performance across all categories, suggesting an absence of category-specific bias or performance degradation. Consequently, the model demonstrates strong generalization across the entire classification task.

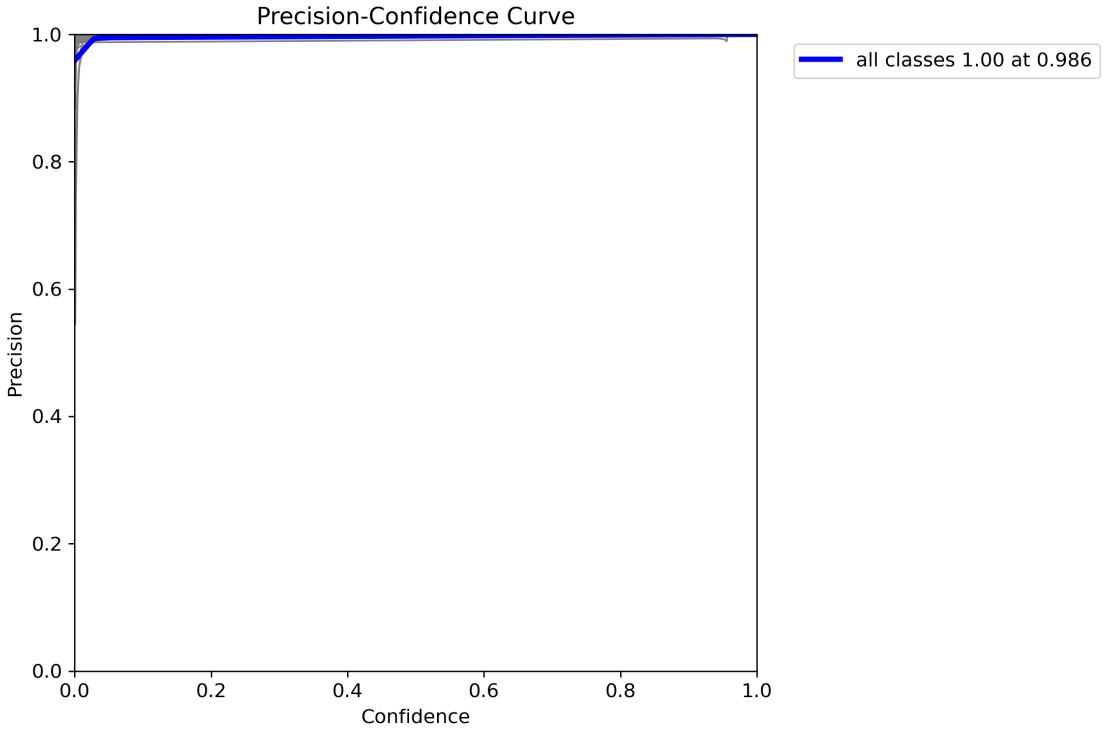

Figure 4.7: Precision-Confidence Curve for for YOLO-AKEMA

As illustrated in **Fig.** 4.7, the Precision-Confidence curve demonstrates that the model consistently maintains a precision value close to 1.00 across most confi-

dence thresholds, reflecting a high degree of prediction accuracy and stability. The primary blue curve in the figure represents the overall precision performance across all categories, which remains at or near the maximum level throughout the entire confidence interval. This observation indicates that the model produces almost no incorrect predictions within the evaluated range.

Additionally, the gray curves depict the variation in precision for each individual category. These curves exhibit a highly concentrated distribution with no evident decline, further confirming the model's consistency and robustness in recognizing a wide variety of flag gestures. Notably, when the confidence threshold reaches 0.986, the overall precision attains a value of 1.00, suggesting that all accepted predictions at this threshold correspond to correct classifications. Such performance underscores the model's strong potential for real-world application.

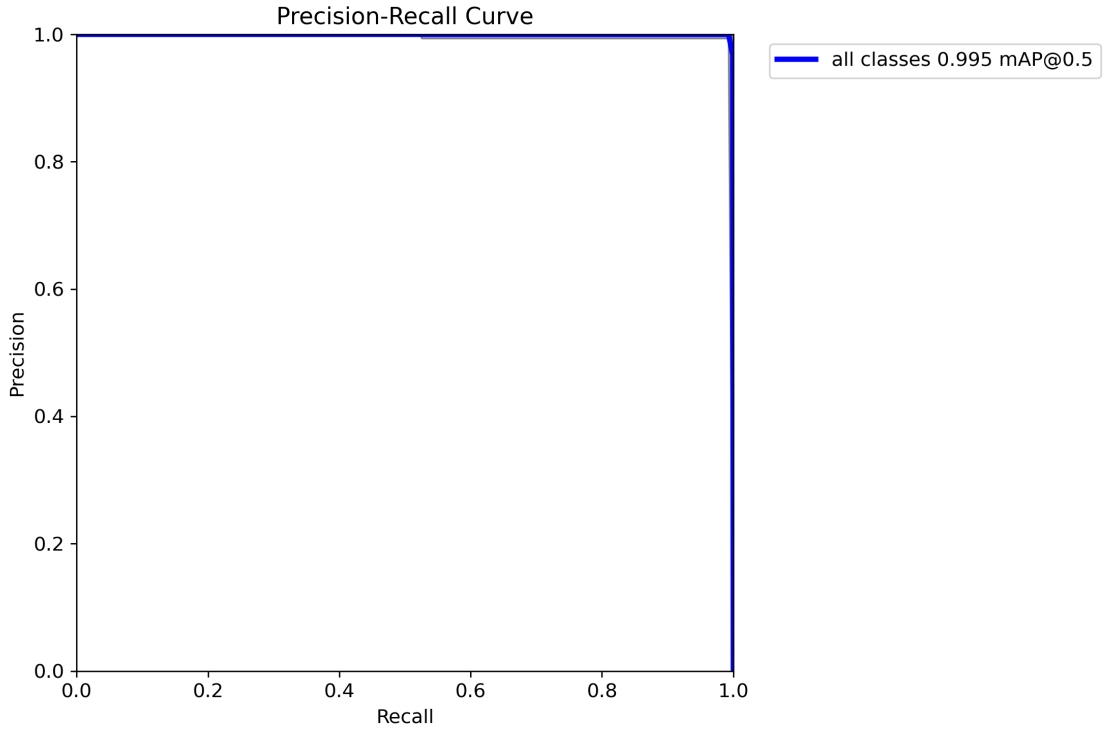

Figure 4.8: Precision-Recall Curve for for YOLO-AKEMA

As presented in **Fig.** 4.8, the blue curve maintains a precision level close to 1.0 across the entire recall range, forming an almost linear high-value trend. This observation indicates that regardless of the number of targets recalled, the model consistently delivers highly accurate predictions with an exceptionally low misclassification rate. The mAP@0.5 value displayed beneath the curve is 0.995, further validating the superior performance of the model in the multi-class flag recognition task. This result suggests that when the Intersection over Union (IoU) threshold is set to 0.5, the model achieves near-optimal average detection accuracy across all categories.

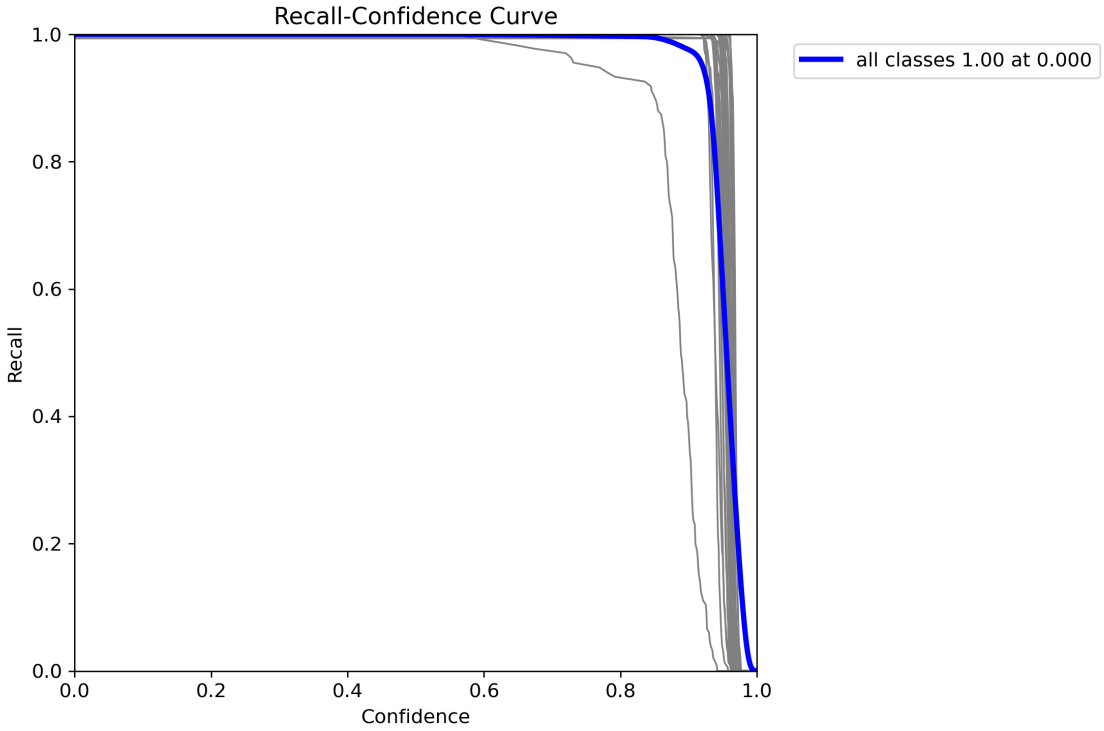

Figure 4.9: Recall-Confidence Curve for for YOLO-AKEMA

From Fig. 4.9, the recall-confidence curve indicates that at lower confidence thresholds, the model achieves a high recall rate, with the overall recall capability reaching a maximum value of 1.00. This outcome suggests that the model is capable of detecting nearly all relevant targets. As the confidence threshold increases, a gradual decline in recall rate is observed, reflecting the inherent trade-off between high precision and maximum recall. Nonetheless, the curve demonstrates that the model maintains broad coverage and stable recall performance in the context of multi-class flag gesture recognition.

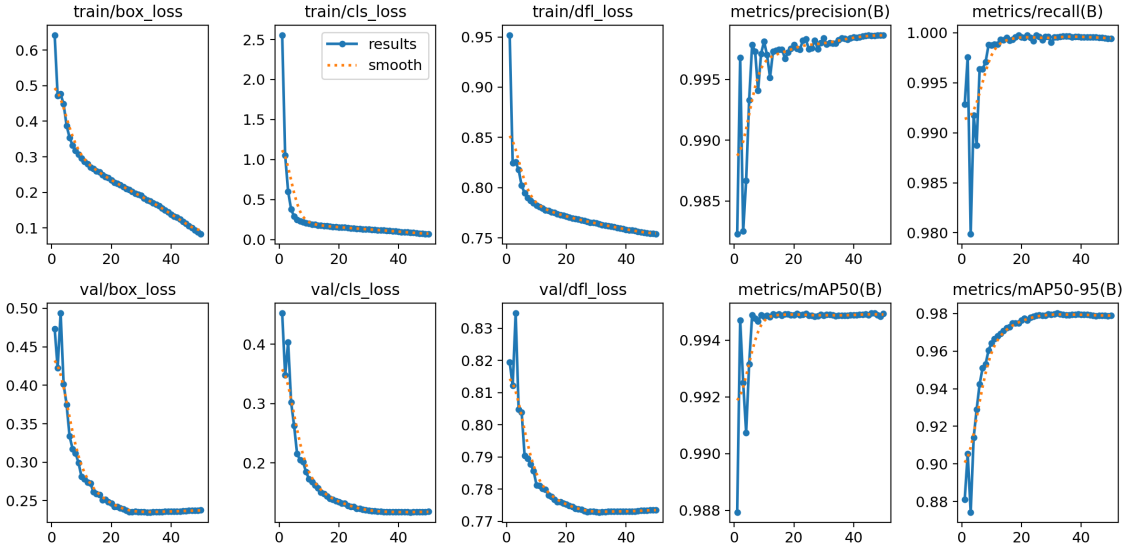

Figure 4.10: Training and Validation Loss Curves for for YOLO-AKEMA

The loss function and performance indicator curves observed during the model training process are presented in **Fig.** 4.10. As depicted in the figure, the various loss components—namely box loss, classification loss, and distribution focal loss—exhibit a steady downward trend throughout both the training and validation phases, eventually converging to low values. This behavior indicates that the model achieves effective fitting without signs of overfitting or underfitting.

Simultaneously, both precision and recall steadily increase during training and ultimately approach values close to 1.00, reflecting strong classification capability and comprehensive recognition coverage. Notably, the model achieves mAP@0.5 and mAP@0.5:0.95 scores of 0.994 and 0.978, respectively, which confirms its ability to maintain high predictive accuracy across varying IoU threshold conditions. These results highlight the model's excellent training stability and detection performance

in the flag signal recognition task, thereby offering a robust foundation for future deployment.

From the perspective of overall evaluation metrics, the model consistently demonstrates high performance and stability in recognizing flag gestures. It is capable of reliably identifying flag actions in both videos and static images, which plays a crucial role in supporting the subsequent development of real-world systems.

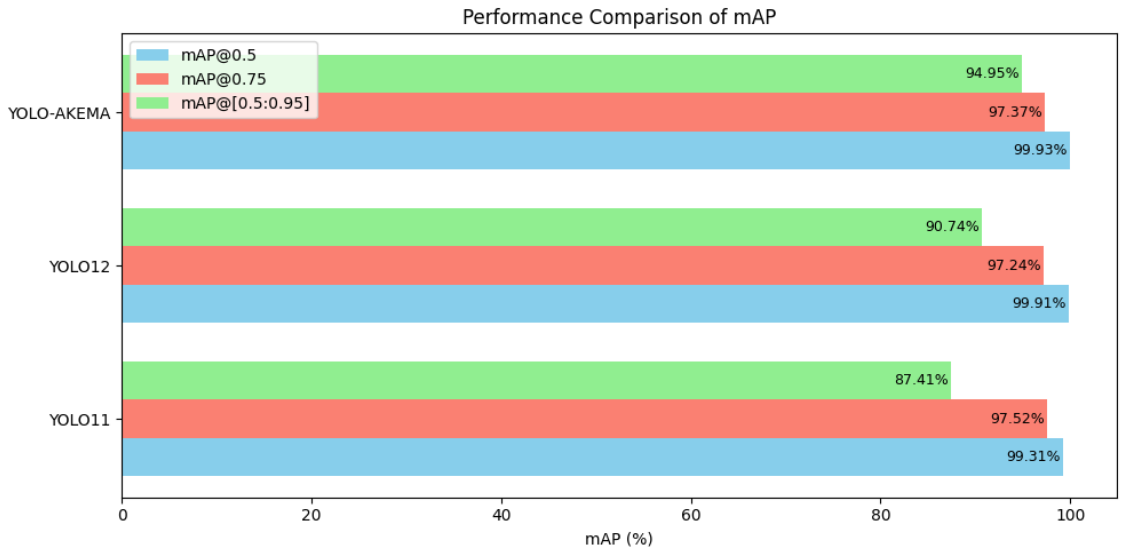

Figure 4.11: Performance Comparison of mAP

From Fig. 4.11, the mean average precision (mAP) results of the three models YOLO11, YOLO12, and YOLO-AKEMA are compared under different Intersection over Union (IoU) thresholds. The results indicate that YOLO-AKEMA achieves the best performance across all metrics, with an mAP@0.5 of 99.93%, an mAP@0.75 of 97.37%, and an mAP@[0.5:0.95] of 94.95

In contrast, YOLO11 attains an mAP@[0.5:0.95] of only 87.41%. While YOLO12

demonstrates a slight improvement over YOLO11 in terms of accuracy, its performance still remains below that of YOLO-AKEMA. For the intended flag signal interaction system, it is essential that the model generates accurate and unique category predictions for each image. Therefore, greater emphasis is placed on the mAP@[0.5:0.95] metric, as it provides a more rigorous evaluation of overall model performance under varying levels of localization precision.

Table 4.3: Training and Inference Speed Comparison

Model	**Training Time per Epoch (s)**	**Inference Time per Image (ms)**	**FPS**
YOLO11	201.6	4.9	204
YOLO12	322.3	4.5	223
YOLO-AKEMA	192.0	4.1	264

As presented in Table 4.3, the performance differences among the YOLO11, YOLO12, and YOLO-AKEMA are compared in terms of training time, inference time, and inference speed measured in frames per second (FPS). The results indicate that YOLO-AKEMA continues to outperform the other models in overall efficiency. Its training time per epoch is only 192.0 seconds, which is shorter than that of YOLO11 and YOLO12, demonstrating a faster training process. Additionally, YOLO-AKEMA shows superior inference performance, with an average inference time of 4.1 milliseconds per image—lower than both alternatives. More notably, the frame processing speed (FPS) of YOLO-AKEMA reaches 264.12, substantially higher than the values recorded for the other two models.

Although YOLO12 shows slightly better accuracy than YOLO11, its per-epoch training time is significantly longer. Given the modest improvement in accuracy, the overall benefit of YOLO12 appears limited. In contrast, YOLO-AKEMA demonstrates clear advantages across all evaluated metrics, making it the most efficient and effective model among the three.

Table 4.4: Ablation Experiments Results

Flipud	**AKConv**	**EMA**	**Precision**	**GFLOPs**	**mAP@0.5:0.95**
✗	✗	✗	65.81%	6.7	63.03%
✓	✗	✗	94.87%	6.7	86.42%
✓	✗	✓	97.92%	6.9	90.12%
✓	✓	✗	96.92%	6.4	89.12%
✓	✓	✓	**99.94%**	6.5	**94.95%**

From the Table 4.4, The results reveal that retaining horizontal flipping during training has a significant adverse effect on the baseline model. This is primarily due to the presence of two mirrored flag gestures within a single category. Such ambiguity is evident in pairs such as B and F, J and P, K and V, M and S, and Q and Y. The rate of misclassification for these categories approaches 50%, resulting in a substantial degradation of model performance. Evaluation metrics clearly show a marked improvement when this data augmentation parameter is disabled, confirming its negative impact on classification accuracy.

The integration of the Efficient Multi-Scale Attention (EMA) mechanism contributes to enhanced prediction capability. However, this enhancement comes with a

slight increase in computational complexity, as indicated by the growth in GFLOPs. Since GFLOPs represent the total number of floating-point operations, an increase in this metric reflects a rise in the model's computational demand. Nonetheless, EMA remains a lightweight attention mechanism that achieves accuracy improvements with only a minimal cost in complexity.

The AKConv module introduces a flexible and efficient convolutional structure that supports arbitrary sampling shapes and parameter configurations. Unlike traditional convolution, its parameter count scales linearly with kernel size rather than quadratically, making it more suitable for detection tasks. Experimental results demonstrate that the addition of AKConv significantly improves detection accuracy, while marginally reducing computational complexity.

Ultimately, the combined incorporation of the attention mechanism and AKConv enables the model to achieve an optimal trade-off between accuracy and efficiency, thereby fulfilling the requirements for deployment in practical applications.

In this section, both comparative and ablation experiments are conducted to validate the superior performance of the improved YOLO-AKEMA model in the flag classification task.

The results from the comparative experiments demonstrate that the improved model exhibits strong performance in terms of both accuracy and robustness. Subsequently, a series of ablation experiments are designed by introducing individual components in a controlled manner. The findings reveal that the horizontal flip parameter in data augmentation has a substantial impact on classification accuracy due to the presence of mirrored flag gestures within specific categories.

Following this, the attention mechanism and the adaptive convolution module (AKConv) are introduced independently. It is observed that the attention mechanism enhances accuracy with a slight increase in computational complexity, whereas the adaptive convolution module contributes to improvements in both accuracy and efficiency.

The final experimental outcomes indicate that the YOLO-AKEMA model is capable of stably and accurately recognizing all target categories in the test video under a high-confidence threshold. These results reflect the model's strong generalization ability and its high potential for practical deployment.

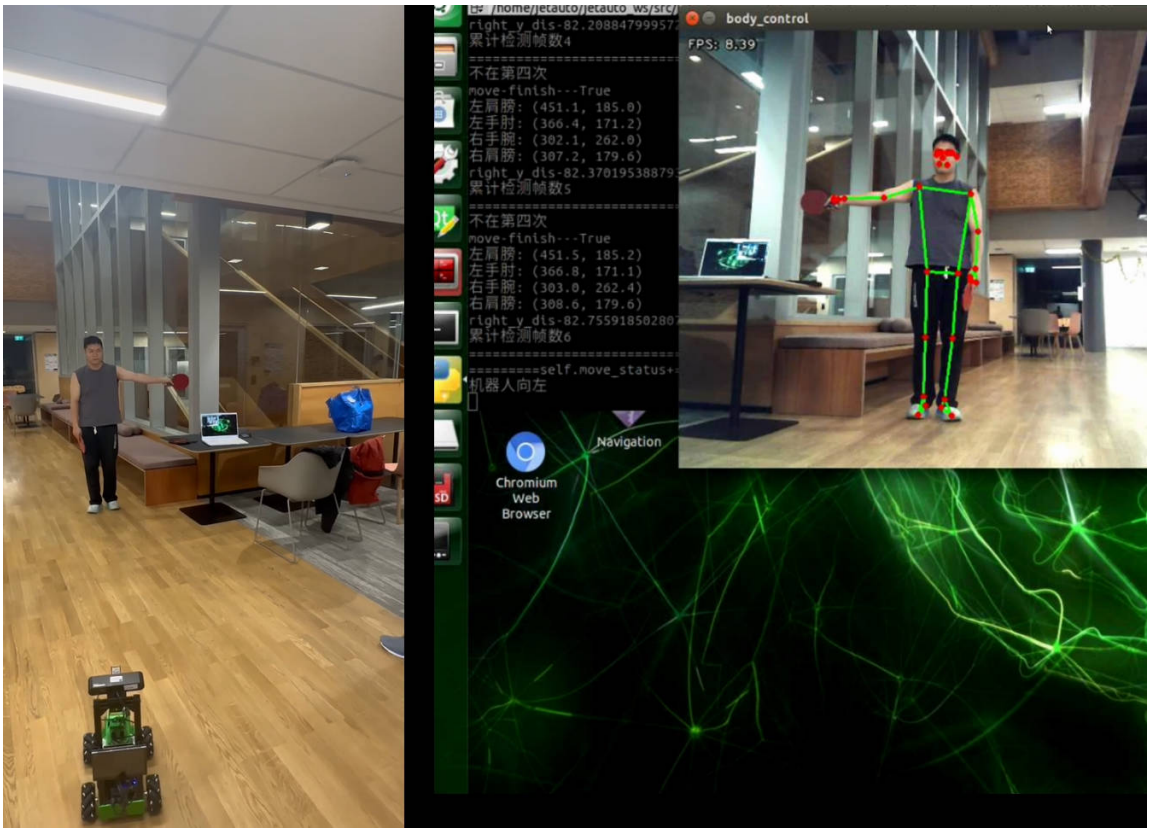

Figure 4.12: Robot Control via Semaphore Gestures

As illustrated in the **Fig.** 4.12, flag signal gestures were utilized to control the motion of the wheeled robot, with four distinct gestures corresponding to forward, backward, leftward, and rightward directions. Despite the use of only four predefined gestures, the experiment effectively demonstrates the model's high prediction accuracy and real-time responsiveness on resource-constrained devices. These findings further substantiate the practical applicability and reliability of the model in real-world deployment scenarios.

4.3 Results for ChatFlags

By adopting the aforementioned approach, the proposed system has been successfully implemented and deployed. As show in **Fig.**4.13, the system currently supports natural language interaction, enabling users to inquire about the meaning of flag action videos through dialogue, either in the form of individual words or complete sentences. Additionally, the system allows users to input single or multiple flag action letters, to which it responds with the corresponding textual descriptions, representative images, or generated videos, depending on the user's request.

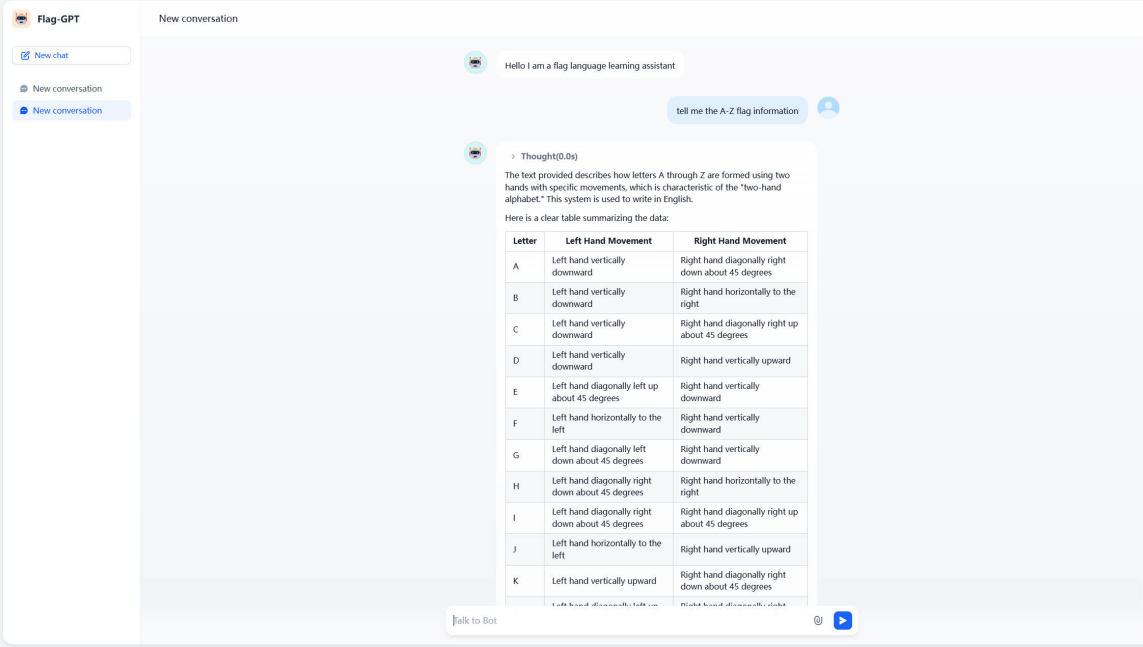

Figure 4.13: ChatFlags Interface

In the following section, several figures are presented to demonstrate the practical outcomes and performance of the system across different interaction scenarios.

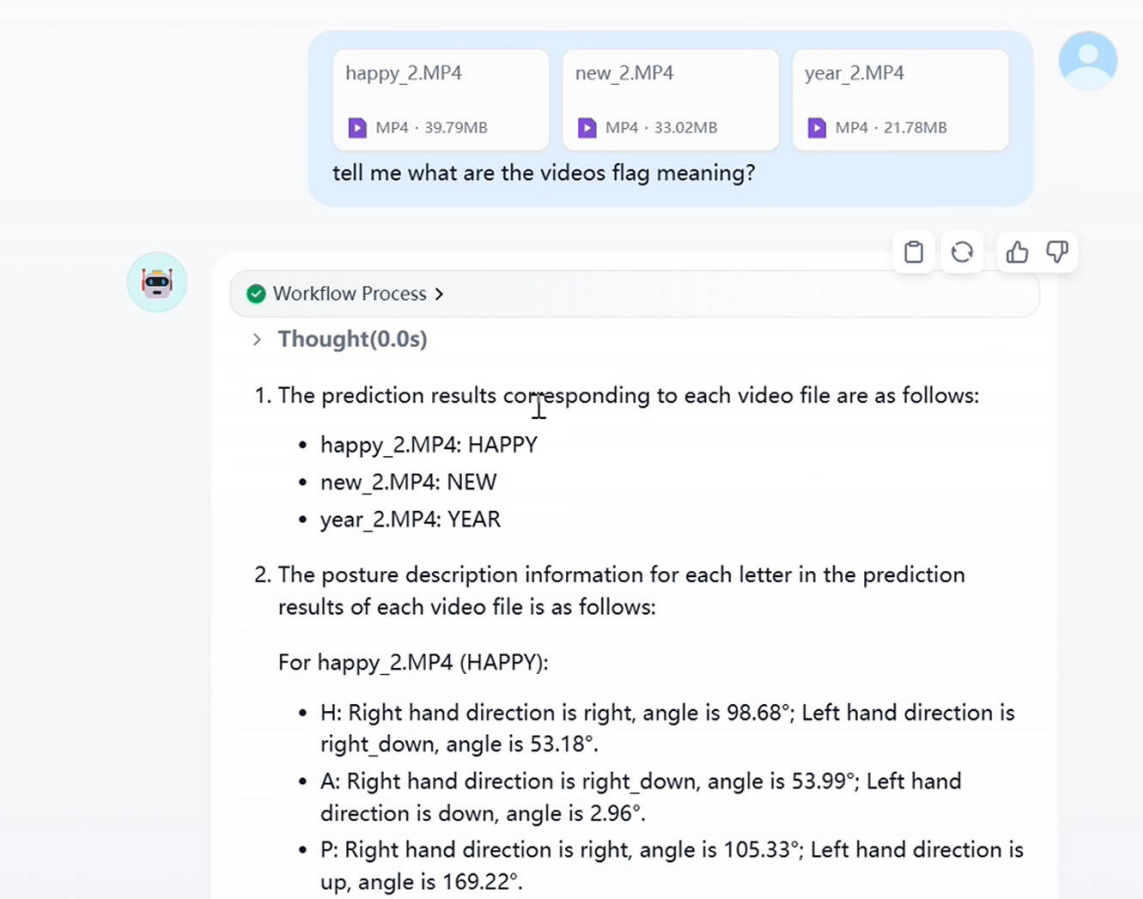

Figure 4.14: Videos Analysis 1

For new_2.MP4 (NEW):

- N: Right hand direction is right, angle is 65.0°; Left hand direction is up, angle is 80.0°.
- E: Right hand direction is down, angle is 15.0°; Left hand direction is left_up, angle is 70.0°.
- W: Right hand direction is right_down, angle is 30.0°; Left hand direction is right, angle is 60.0°.

For year_2.MP4 (YEAR):

- Y: Right hand direction is right_up, angle is 134.29°; Left hand direction is left, angle is 83.72°.
- E: Right hand direction is down, angle is 2.57°; Left hand direction is left_up, angle is 138.04°.
- A: Right hand direction is right_down, angle is 59.25°; Left hand direction is down, angle is 0.84°.
- R: Right hand direction is right, angle is 96.85°; Left hand direction is left, angle is 100.04°.

3. The summary of the prediction results for each video file:

- happy_2.MP4 (HAPPY): The video shows a sequence where the right and left hands move in specific directions to form the letters H, A, P, P, Y.
- new_2.MP4 (NEW): The video shows hand movements forming N, E, W with precise angles for each letter's pose.
- year_2.MP4 (YEAR): The video demonstrates hand poses for Y, E, A, R, each with distinct directional angles.

Figure 4.15: Videos Analysis 2

From **Fig.** 4.14 and 4.15, the system is capable of analyzing the uploaded video and accurately identifying the flag gestures it contains. Furthermore, it provides

an interpretation of the word conveyed by each video, demonstrating the system's ability to extract semantic meaning from visual input in real time.

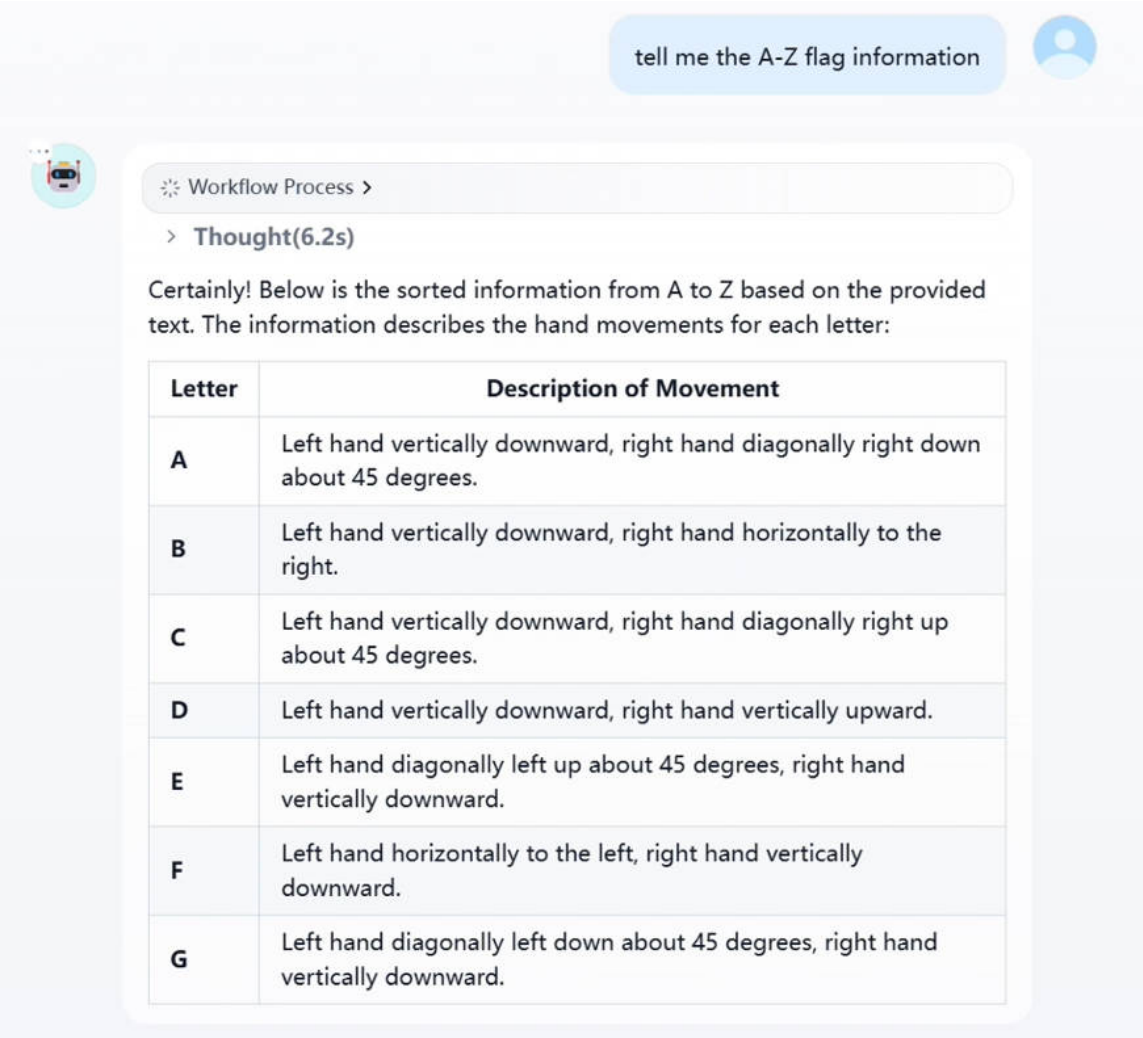

Figure 4.16: Flag Action Description in Text

From **Fig.** 4.16, the system presents the flag gestures corresponding to the letters A–Z in a structured tabular format, in accordance with the user's query. This demonstrates the system's ability to generate organized and interpretable responses

based on user-defined informational needs.

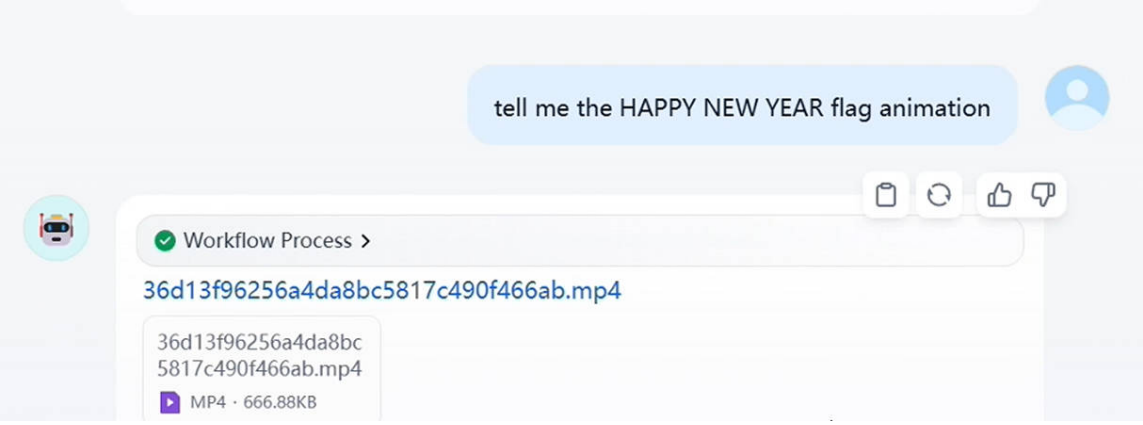

Figure 4.17: Flag Action Description in Text

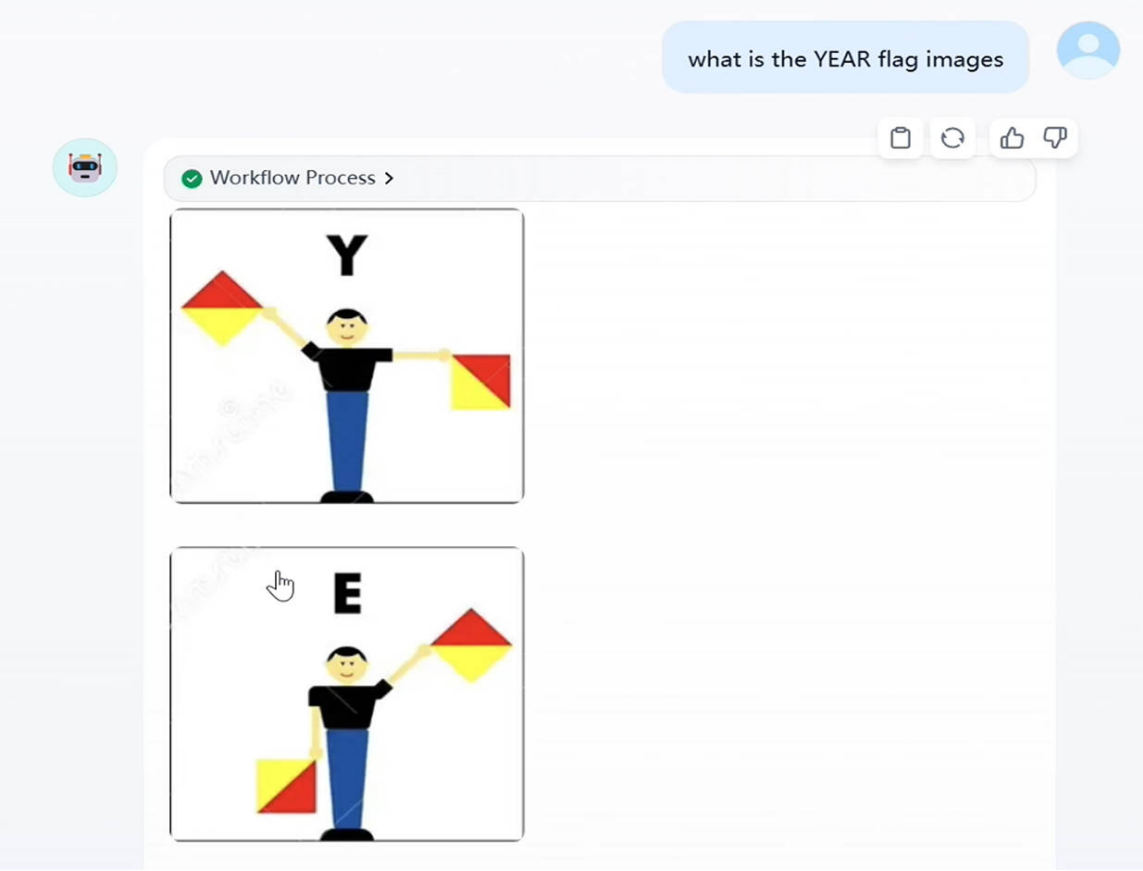

Figure 4.18: Flag Action description using Images

As illustrated in **Fig.** 4.17 and 4.18, when the user requests a video of flag signals representing a complete sentence, the system accurately generates the corresponding video and provides a downloadable link. Additionally, when the user requests to view the sequence of flag gestures for a word in image format, the system returns a series of ordered images that accurately reflect the intended gestures.

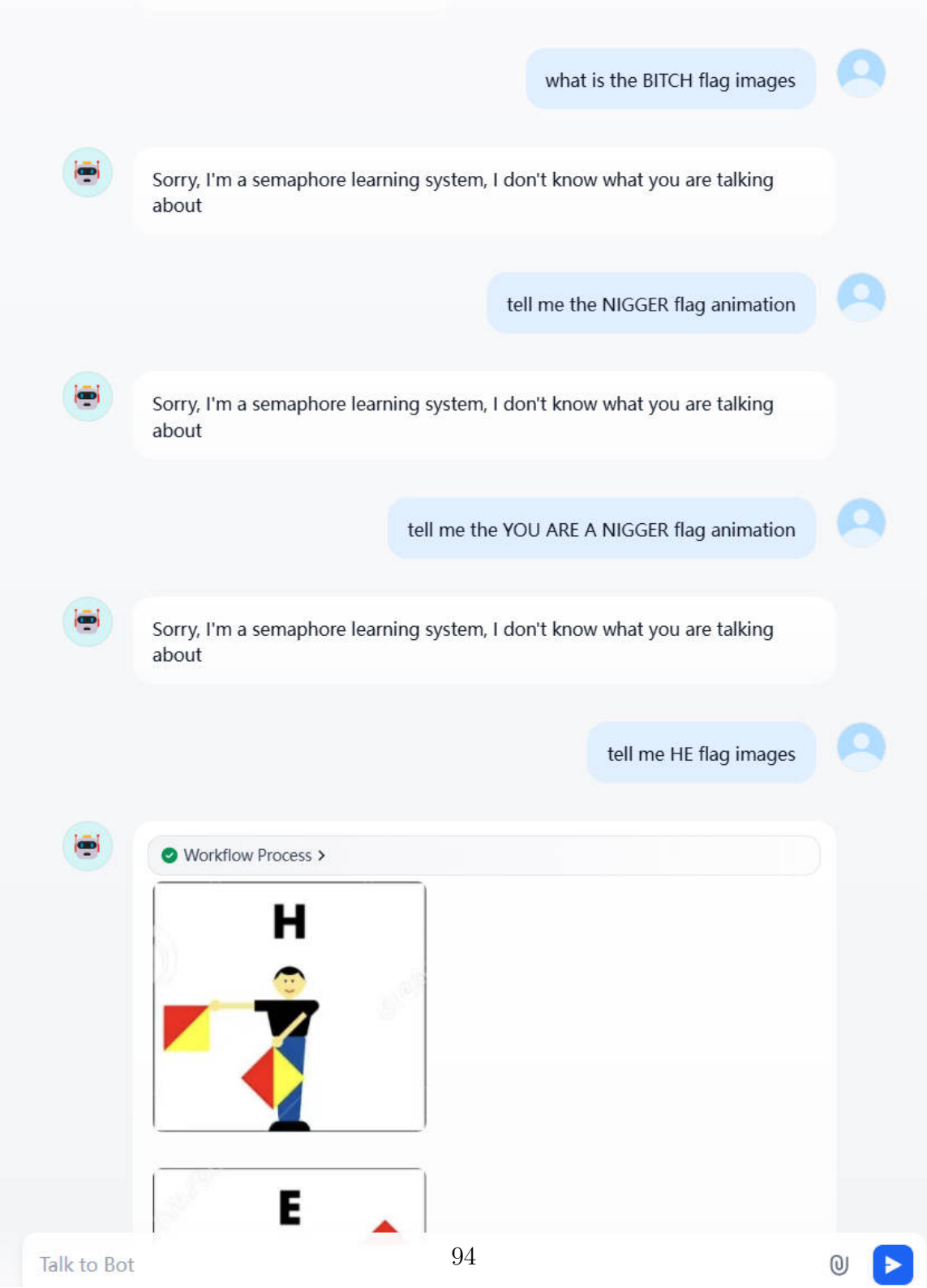

Figure 4.19: Profanity Detection

As shown in **Fig.** 4.19, when the user input contains profane or inappropriate language, the system accurately identifies the content and returns a moderated, non-confrontational response. This indicates that ChatFlags exhibits a degree of sensitivity to offensive expressions, which is particularly crucial in educational environments. Such functionality not only mitigates the risk of legal or regulatory violations but also protects users from exposure to harmful or inappropriate material.

In terms of system performance, the initial video recognition and analysis phase incurs a latency of approximately 30–40 seconds. However, caching strategies are employed following the first inference to eliminate redundant computation. For tasks such as text generation, image-based responses, video synthesis, and other interactive functions, the average response time is maintained within 3 seconds. Overall, the system demonstrates stable performance and ensures a seamless user experience across multiple interaction modes.

This section presents the implementation of the proposed ChatFlags system. The DeepSeek model and a text embedding model were locally deployed using the Ollama framework and seamlessly integrated through the open-source platform Dify. By combining these language processing capabilities with a visual analysis module based on the YOLO-AKEMA architecture, a multifunctional and intelligent system for flag language learning and interaction was developed.

Through the integration of a large-scale language model with a deep learning–based visual recognition module, the system enhances the individual functionalities of each component and facilitates novel application scenarios for both language and vision-driven AI systems.

5 Discussion

To begin with, following a series of experiments, the implementation and deployment of the ChatFlags system were completed, achieving the intended objectives. This section presents a discussion of key challenges encountered during the initial stages of the research.

The preliminary experiment revealed varying performance levels among CNN-based models in the flag classification task. The basic S_CNN architecture yielded significantly lower accuracy compared to the other models, primarily due to its simpler structure, limited parameter capacity, and constrained training data. Among the pre-trained models, although ResNet18 achieved competitive results, DenseNet121 demonstrated superior stability and accuracy. Its densely connected architecture and deeper network layers allow each layer to directly access outputs from all preceding layers, thereby enhancing information flow and promoting feature reuse. Additionally, performance was further improved through image augmentation techniques. Notably, the integration of posture features extracted via MediaPipe provided complementary input to DenseNet121, enabling it to maintain high accuracy and generalization even with limited labeled data. These findings suggest that DenseNet121 exhibits strong adaptability and effectiveness in flag classification tasks.

In parallel, YOLO11 exhibited high-speed, end-to-end detection capabilities for flag gesture recognition and related applications. By utilizing bounding boxes for localization and classification, it enhances annotation efficiency while maintaining detection accuracy. However, during Experiment 2, the model encountered chal-

lenges in classifying 27 types of flag signal actions. In particular, the aspect ratio of annotation boxes was found to influence detection performance. This issue was addressed through modifications to the YOLO architecture. Overall, YOLO11 proves to be a more suitable candidate for real-time flag signal action classification.

In the experimental process, MediaPipe was employed to extract the postures of key human body parts to support model training. The results indicate that providing posture information as an auxiliary input effectively enhances model learning. MediaPipe also contributed significantly to data annotation, image analysis, and other stages, demonstrating its value in flag language recognition and human pose estimation tasks. Comparative analysis of Experiment 1 and Experiment 2 revealed that the elbow and wrist serve as the most critical key points for flag gestures, rendering additional shoulder landmarks unnecessary. These two joints not only direct the model's focus toward the salient components of flag actions but also provide a geometric foundation for vector calculation in action analysis. By emphasizing the orientation and angular variation of the elbow and wrist, the structural rules of flag gestures can be accurately captured.

In Experiment 2, the prediction accuracy of YOLO11 was improved by incorporating an attention mechanism and replacing the standard convolution module. Enhanced posture point annotations guided the attention module to concentrate on critical regions and learn distinctive features of flag gestures. The AKConv adaptive convolution module further contributed to performance gains through its dynamic kernel and parameter adjustments. These modifications proved effective in enhancing YOLO11's ability to predict flag actions with higher precision. This improvement

enables the model to operate effectively on edge devices. For instance, the flag signal recognition model was successfully deployed on a wheeled robot, achieving accurate motion-based control of the robot's movements. Although the current model satisfies baseline requirements, further validation is necessary in scenarios involving diverse backgrounds and subjects. Additional refinements are also required to improve generalization and robustness.

For practical deployment, techniques such as half-precision computation, streaming inference, and secondary verification were applied to ensure accuracy and computational efficiency. The current implementation supports smooth operation within the ChatFlags system. However, latency issues may arise when processing large or numerous video inputs. Moreover, the current workflow requires separate steps for analyzing static flag action images, indicating areas that warrant further optimization.

The successful deployment of the ChatFlags system demonstrates the potential of integrating visual models with large language models to achieve enhanced functionality. DeepSeek, combined with a structured knowledge base, was utilized to augment the flag language capabilities of the language model and to interpret and reorganize the prediction outputs generated by the visual component. This integration enabled seamless collaboration between perception and reasoning modules within the ChatFlags system.

6 Conclusion and Future work

6.1 Conclusion

In this thesis, a fully self-constructed flag signal dataset was employed to conduct both preliminary and subsequent enhancement experiments on YOLO11. This effort significantly contributed to the research and enriched the dataset. This study initially compared the performance of several CNN models on five-class flag gesture classification tasks, confirming the suitability of YOLO11 for flag-related recognition. Based on the results of preliminary experiments, the dataset was reconstructed and preprocessed to address identified issues. The enhanced model, YOLO-AKEMA, was subsequently evaluated against multiple YOLO variants. Results from ablation experiments demonstrated that the combination of the attention mechanism and adaptive convolution module improved YOLO11 across all evaluation metrics. Finally, YOLO-AKEMA successfully recognized 27 distinct flag gestures with high accuracy and was deployed on a wheeled robot to achieve effective motion control. This deployment not only validates the recognition capabilities and practical applicability of YOLO-AKEMA but also establishes a solid foundation for the subsequent development of the ChatFlags system.

To support the language component, a flag gesture knowledge base was constructed by using the BGE-M3 text embedding model and the Dify framework. A ChatFlow application was created to integrate DeepSeek, deployed via Ollama, enabling interaction with large language models through a dynamic interface. The

visual analysis module was deployed as an independent service, communicating with Dify through HTTP. This multi-component integration resulted in an intelligent system capable of flag video analysis, video generation, and multimodal output, including text and visual descriptions.

Despite these achievements, a few limitations remain. Although the current dataset includes 27 gesture categories, it does not cover numbers, symbols, or specialized movements, which limits its completeness for full flag language communication. And it does not include flag signal recognition for special fields. Additionally, the current analysis process relies on a sequential prediction–processing mechanism, which introduces latency and reduces system responsiveness. This post-processing approach also lacks adaptability. In this thesis, the retrieval-augmented generation (RAG) approach was applied to mitigate hallucinations in DeepSeek responses. While effective and flexible—enabling dynamic modification of the knowledge base—this plugin-based solution does not enhance the language model itself. System performance becomes significantly constrained once the RAG component is unavailable. Although Dify is a powerful platform, the integration of numerous tools introduces functional redundancy, which increases the overall deployment complexity of the system. This issue will be addressed in future work (Huan & Yan, 2025; Huan et al., 2025).

Another critical aspect that warrants particular attention is content safety. As an educational system dedicated to promoting the dissemination of semaphore language, ChatFlags recognizes the significance of maintaining a secure and respectful information environment. To this end, a profanity detection mechanism has been

implemented. However, its current scope remains limited, and further refinement is required to effectively address a broader spectrum of sensitive content. Enhancing this capability is essential to ensure that users are provided with a healthier and more welcoming learning experience.

Overall, the development of ChatFlags demonstrates the practical feasibility of integrating visual models with large language models to construct an intelligent, multimodal learning system for flag signal recognition. ChatFlags addresses long-standing challenges in flag language education and offers a novel perspective for visual–language model integration, extending its potential to the domain of structured symbolic communication.

6.2 Future Work

In light of the current limitations, future work will focus on addressing these challenges. First, the dataset will be further expanded to include the complete set of flag gestures. In addition, a dedicated flag dataset will be developed to support recognition in specialized domains such as aviation and emergency rescue, thereby enhancing the model's applicability and generalization.

For model improvement, a multimodal approach will be explored. This involves combining semantic descriptions of images with the images themselves and training the model using multimodal techniques. Such an approach is expected to enable direct generation of both classification outcomes and interpretive descriptions, reducing reliance on manual post-processing and significantly improving the efficiency and intelligence of ChatFlags.

To strengthen the language component, a comprehensive flag knowledge dataset will be created to fine-tune DeepSeek, enabling the large language model to better understand domain-specific knowledge. When integrated with the RAG system, this enhancement is expected to improve the accuracy, completeness, and stability of knowledge-based responses within the system.

Although Dify currently provides a convenient platform for rapid development, future work will consider designing a customized interactive interface tailored to the specific needs of the ChatFlags system.

To enhance the review of sensitive content, we plan to expand the detection scope beyond the existing capabilities. This includes the identification of content related to politically sensitive topics, religious extremism, sexually explicit material, violence, and personal privacy. A dedicated AI module will be developed to handle such tasks, utilizing advanced natural language processing (NLP) models to detect not only explicit terms but also implicit, suggestive, or misleading sensitive information embedded in text. Compared with traditional keyword-based approaches, this method offers greater intelligence and contextual awareness. Furthermore, we aim to integrate multimodal models capable of analyzing both textual and visual media, enabling the identification of images containing illegal content or offensive language. Overall, this initiative is central to ensuring that users interact with ChatFlags in a secure and trustworthy environment.

Given the successful deployment of the flag signal recognition model on a wheeled robot, future research may explore its integration with large language models (LLMs) to facilitate more advanced, interactive, and meaningful robotic applications. Such a

combination is anticipated to enhance robotic capabilities in multimodal perception and human-robot interaction, while also broadening both the application methods and the potential scope of the flag signal recognition model.

In summary, future research will remain focused on enhancing the effectiveness and diversity of flag language applications, thereby demonstrating its value and utility across various domains. This ongoing effort aims to promote the broader dissemination and practical implementation of flag language in real-world scenarios (Yan, 2026)(Yan, 2023)(Yan, 2019).

References

Albawi, S., Mohammed, T. A., & Al-Zawi, S. (2017). Understanding of a convolutional neural network. *International Conference on Engineering and Technology (ICET)*, 1–6. https://doi.org/10.1109/ICEngTechnol.2017.8308186

Andriluka, M., Pishchulin, L., Gehler, P., & Schiele, B. (2014). 2D human pose estimation: New benchmark and state of the art analysis. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 3686–3693. https://doi.org/10.1109/CVPR.2014.471

Army, S. C. U. S. (1910). Visual signaling. *War Department Office of The Chief Signal Officer*. https://www.gutenberg.org/ebooks/43515

Arslan, M., Ghanem, H., Munawar, S., & Cruz, C. (2024). A survey on RAG with LLMs. *Procedia Computer Science*, *246*, 3781–3790. https://doi.org/10.1016/j.procs.2024.09.178

Aruna, M., Kaneriya, G., & Jain, P. (2024). Leveraging pre-trained ResNet18 with transfer learning for yoga posture classification. *Second International Conference on Networks, Multimedia and Information Technology (NMITCON)*, 1–5. https://doi.org/10.1109/NMITCON62075.2024.10699235

Badiola-Bengoa, A., & Mendez-Zorrilla, A. (2021). A systematic review of the application of camera-based human pose estimation in the field of sport and physical exercise. *Sensors*, *21*(18), 5996. https://doi.org/10.3390/s21185996

Borodulina, A. (2019). *Application of 3D Human Pose Estimation for Motion Capture and Character Animation, University of Oulu* [Master's thesis]. University of Oulu.

Bregler, C., & Malik, J. (1998). Tracking people with twists and exponential maps. *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, 8–15. https://doi.org/10.1109/CVPR.1998.698581

Canoy, N. T., Jr., R. L. J., & Blanco, A. L. (2024). An implementation of the YOLO algorithm for the recognition of flags of the international code of signals. *Procedia Computer Science*, *225*, 1364–1371. https://doi.org/10.1016/j.procs.2024.03.155

Cao, X., & Yan, W. (2024). Pose estimation for swimmers in video surveillance. *Multimedia Tools and Applications*, *93*, 26565–26580.

Chang, C.-W., Nian, M.-D., Chen, Y.-F., Chi, C.-H., & Tao, C.-W. (2014). Design of a kinect sensor based posture recognition system. *Tenth International Conference on Intelligent Information Hiding and Multimedia Signal Processing (IIH-MSP)*, 856–859. https://doi.org/10.1109/IIH-MSP.2014.216

Chen, J. (2024a). DCM-GIFT: An Android malware dynamic classification method based on gray-scale image and feature-selection tree. *Information and Software Technology*, *176*, 107560. https://doi.org/10.1016/j.infsof.2024.107560

Chen, Z. (2024b). *Real-Time Pose Recognition for Billiard Player Using Deep Learning, Auckland University of Technology, New Zealand* [Master's thesis].

Cheonsu, J. (2023). A study on the implementation of generative AI services using an enterprise data-based LLM application architecture. *Advances in Artificial Intelligence and Machine Learning*, *3*(4), 1588–1618. https://doi.org/10.48550/arXiv.2309.01105

Cust, E. E., Sweeting, A. J., Ball, K., & Robertson, S. (2019). Machine and deep learning for sport-specific movement recognition: A systematic review of model development and performance. *Journal of Sports Sciences*, *37*(5), 568–600. https://doi.org/10.1080/02640414.2018.1521769

Dabwan, B. A., Jadhav, M. E., Gadkari, A. S., Ali, Y. A., Almula, S. M., Ismil, O. A., & Mohammad, A. A. (2024). Hand gesture classification for individuals with disabilities using the DenseNet121 model. *International Conference on Advancements in Power, Communication and Intelligent Systems (APCI)*, 1–5. https://doi.org/10.1109/APCI61480.2024.10616504

Dalal, N., & Triggs, B. (2005). Histograms of oriented gradients for human detection. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, *1*, 886–893. https://doi.org/10.1109/CVPR.2005.177

Di, Y., Shi, H., Wang, Q., Jia, S., Bao, J., & Liu, Y. (2025). Federated cross-domain recommendation system based on bias eliminator and personalized extractor. *Knowledge and Information Systems*, *67*(3), 2935–2965. https://doi.org/10.1007/s10115-024-02316-y

Dong, X., Li, S., & Zhang, J. (2024). YOLOV5s object detection based on Sim SPPF hybrid pooling. *Optoelectronics Letters*, *20*(6), 367–371. https://doi.org/10.1007/s11801-024-3170-x

Dwork, C., Hays, C., Kleinberg, J., & Raghavan, M. (2023). Content moderation and the formation of online communities: A theoretical framework. *arXiv preprint arXiv:2310.10573*. https://arxiv.org/abs/2310.10573

Felzenszwalb, P. F., & Huttenlocher, D. P. (2005). Pictorial structures for object recognition. *International Journal of Computer Vision*, *61*(1), 55–79. https://doi.org/10.1023/B:VISI.0000043756.37893.9d

Freeman, W. T., & Roth, C. D. (1995). Orientation histograms for hand gesture recognition. *International Workshop on Automatic Face and Gesture Recognition*, 296–301. https://doi.org/10.1109/AFGR.1996.557274

Gohil, U., Pawar, R., & Dhake, B. (2023). Real-time maritime semaphore sign detection system. *2023 3rd Asian Conference on Innovation in Technology (ASIANCON)*, 1–5. https://doi.org/10.1109/ASIANCON58793.2023.10270498

Guizani, S., Mazhar, T., Shahzad, T., Ahmad, W., Bibi, A., & Hamam, H. (2025). A systematic literature review to implement large language model in higher education: Issues and solutions. *Discover Education*, *4*(1), 35. https://doi.org/10.1007/s44217-025-00424-7

Gupta, S., Ranjan, R., & Singh, S. N. (2024). A comprehensive survey of retrieval-augmented generation (RAG): Evolution, current landscape and future directions. *arXiv preprint arXiv:2410.12837*. https://arxiv.org/abs/2410.12837

Han, J., Shao, L., Xu, D., & Shotton, J. (2013). Enhanced computer vision with microsoft kinect sensor: A review. *IEEE Transactions on Cybernetics*, *43*(5), 1318–1334. https://doi.org/10.1109/TCYB.2013.2265378

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 770–778. https://doi.org/10.1109/CVPR.2016.90

Helena, S. (2015). Learning application on signal scout semaphore multimedia and web-based using computer assisted instruction method. *Proceedings of the 3rd International Conference on Information Technology and Business (ICITB)*, 91–96.

Hu, J., Shen, L., & Sun, G. (2018). Squeeze-and-excitation networks. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 7132–7141.

Huan, Y., Tran, K., Nguyen, M., & Yan, W. (2025). Arobotflags: Ai-powered semaphore interactions between chatbot and humanoid robot. *IVCNZ2025*.

Huan, Y., & Yan, W. (2025). Semaphore recognition using deep learning. *Electronics*, *14*(2), 286.

Huang, G., Liu, Z., Van Der Maaten, L., & Weinberger, K. Q. (2017). Densely connected convolutional networks. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2261–2269.

Hussain, M. (2024). YOLOv1 to v8: Unveiling each variant–a comprehensive review of YOLO. *IEEE Access*, *12*, 42816–42833. https://doi.org/10.1109/ACCESS.2024.3378568

Illavarason, P., Arokia Renjit, J., & Mohan Kumar, P. (2019). Medical diagnosis of cerebral palsy rehabilitation using eye images in machine learning techniques. *Journal of Medical Systems*, *43*(8), 278. https://doi.org/10.1007/s10916-019-1415-3

Iwane, N. (2012). Arm movement recognition for flag signaling with kinect sensor. *IEEE International Conference on Virtual Environments Human-Computer Interfaces & Measurement Systems (VECIMS)*, 86–90. https://doi.org/10.1109/VECIMS.2012.6273224

Jana, A. (2012). *Kinect for windows sdk programming guide*. Packt Publishing.

Jegham, N., Koh, C. Y., Abdelatti, M., & Hendawi, A. (2024). YOLO evolution: A comprehensive benchmark and architectural review of YOLOv12, YOLO11, and their previous versions [Accessed: 2025-03-30]. *arXiv preprint arXiv:2411.00201*. https://arxiv.org/abs/2411.00201

Jeong, C. (2023). Generative AI service implementation using LLM application architecture: Based on RAG model and langchain framework. *Journal of Artificial Intelligence and Machine Learning*, *5*(4), 123–135. https://doi.org/10.1234/jaiml.v5i4.5678

Jetley, S., Vaze, A., & Belhe, S. (2013). Automatic flag recognition using texture based color analysis and gradient features. *Proceedings of the International*

Conference on Image Information Processing (ICIIP), 496–500. https://doi.org/10.1109/ICIIP.2013.6707649

Kang, Y., Cai, Z., Tan, C.-W., Huang, Q., & Liu, H. (2020). Natural language processing (NLP) in management research: A literature review. *Journal of Management Analytics*, *7*(2), 139–172. https://doi.org/10.1080/23270012.2020.1756939

Keskin, C., Kıraç, F., Kara, Y. E., & Akarun, L. (2011). Dynamic hand pose recognition using depth data. *Proceedings of the International Conference on Image Processing (ICIP)*, 2745–2748. https://doi.org/10.1109/ICIP.2011.6116202

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. *Advances in Neural Information Processing Systems*, *25*.

LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, *86*(11), 2278–2324. https://doi.org/10.1109/5.726791

Lee, K., Kim, W., & Lee, S. (2022). From human pose similarity metric to 3D human pose estimator: Temporal propagating LSTM networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *45*(2), 1781–1797. https://doi.org/10.1109/TPAMI.2022.3151860

Li, W., Yang, Y., Wang, M., & Zhang, L. (2018). Convolutional neural network architecture for semaphore recognition. *2018 IEEE 9th International Conference on Software Engineering and Service Science (ICSESS)*, 559–562. https://doi.org/10.1109/ICSESS.2018.8663885

Li, Y., Yang, N., Wang, M., Zhang, L., & Zhu, M. (2024). Research on human posture estimation algorithm based on YOLO-pose. *Sensors*, *24*(10), 3036. https://doi.org/10.3390/s24103036

Liu, R.-M., & Su, W.-H. (2024). APHS-YOLO: A lightweight model for real-time detection and classification of stropharia rugoso-annulata. *Foods*, *13*(11), 1710. https://doi.org/10.3390/foods13111710

Liu, W., Zhang, Q., & Chen, L. (2023). A defect detection method for a boiler inner wall based on an improved YOLOv5 network and data augmentation technologies. *Journal of Intelligent & Fuzzy Systems*, *45*(3), 1234–1245. https://doi.org/10.3233/JIFS-234567

Liu, Y., Gao, Z., Liu, B., Zhang, X., Huang, L., & Xie, J. (2024). EMA-YOLO: A novel target-detection algorithm for immature yellow peach based on YOLOv8. *Sensors*, *24*(12), 3783. https://doi.org/10.3390/s24123783

Liu, Y., Nand, P., Hossain, A., Nguyen, M., & Yan, W. (2023). Sign language recognition from digital videos using feature pyramid network with detection transformer. *Multimedia Tools and Applications*, *82*, 21673–21685.

Lu, J., Nguyen, M., & Yan, W. (2021). Sign language recognition from digital videos using deep learning methods. *International Symposium on Geometry and Vision*.

Lugaresi, C., Tang, J., Nash, H., McClanahan, C., Uboweja, E., Hays, M., Zhang, F., Chang, C.-L., Yong, M. G., Lee, J., Chang, W.-T., Hua, W., Georg, M., & Grundmann, M. (2019). MediaPipe: A framework for building perception

pipelines. *arXiv preprint arXiv:1906.08172*. https://doi.org/10.48550/arXiv.1906.08172

Mead, H. P. (1935). The story of the semaphore. *The Mariner's Mirror*, *21*(1), 33–55. https://doi.org/10.1080/00253359.1935.10658699

Neumann, A. T., Yin, Y., Sowe, S. K., Decker, S. J., & Jarke, M. (2024). An LLM-driven chatbot in higher education for databases and information systems. *IEEE Transactions on Education*, *68*(1), 103–116. https://doi.org/10.1109/TE.2024.3467912

Newell, A., Yang, K., & Deng, J. (2016). Stacked hourglass networks for human pose estimation. *European Conference on Computer Vision (ECCV)*, 483–499. https://doi.org/10.1007/978-3-319-46484-8_29

Ouyang, D., He, S., Zhang, G., Luo, M., Guo, H., Zhan, J., & Huang, Z. (2023). Efficient multi-scale attention module with cross-spatial learning. *ICASSP IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 1–5. https://doi.org/10.1109/ICASSP49357.2023.10096516

Pavlovic, V. I., Sharma, R., & Huang, T. S. (1997). Visual interpretation of hand gestures for human-computer interaction: A review. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *19*(7), 677–695. https://doi.org/10.1109/34.598226

Pawar, K., & Gawande, K. (2023). A survey of text representation and embedding techniques in NLP. *IEEE Access*, *11*, 36120–36145. https://doi.org/10.1109/ACCESS.2023.3266377

Peng, W., Zhang, X., & Wang, X. (2019). Research on human action recognition based on convolutional neural network. *Proceedings of the 2019 International Conference on Artificial Intelligence and Advanced Manufacturing (AIAM)*, 466–470. https://doi.org/10.1109/AIAM48774.2019.00093

Penulis, N. (2024). Pengenalan kode semaphore berbasis raspberry Pi menggunakan metode CNN. *eProceedings of Engineering*, *11*(4). https://openlibrarypublications.telkomuniversity.ac.id/index.php/engineering/article/view/24143

Pratumgul, W., & Sa-ngiamvibool, W. (2016). A prototype system for automated screening and detection of diabetic retinopathy severity using color fundus images for mhealth in thailand. *Procedia Computer Science*, *86*, 457–460.

Putra, L. S. A., Sumarno, L., & Gunawan, V. A. (2018). The recognition of semaphore letter code using haar wavelet and euclidean function. *International Conference on Electrical Engineering, Computer Science and Informatics (EECSI)*, 759–763. https://doi.org/10.1109/EECSI.2018.8752707

Putra, L. S. A., Wigyarinto, F. T. P., Kusumawardhani, E., & Gunawan, V. A. (2024). Semaphore letter code recognition system using wavelet method and back propagation neural network. *International Journal of Advanced Technology and Engineering Exploration*, *11*(112), 316–331. https://doi.org/10.19101/IJATEE.2023.10102433

Qian, Z., Li, Y., Yang, N., Yang, Y., & Zhu, M. (2016). A convolutional neural network approach for semaphore flag signaling recognition. *IEEE International Conference on Signal and Image Processing (ICSIP)*, 466–470. https://doi.org/10.1109/SIPROCESS.2016.7888311

Rachmad, A., & Fuad, M. (2015). A geometry-based algorithm for semaphore gesture recognition using skeleton images. *Journal of Theoretical and Applied Information Technology*, *81*(1), 102–107.

Rachmad, A., & Fuad, M. (2018). Skeleton-based semaphore gesture recognition using geometrical algorithm. *International Conference on Electrical Engineering and Computer Science (EECSI)*, 219–224. https://doi.org/10.1109/EECSI.2018.8752707

Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You only look once: Unified, real-time object detection. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 779–788. https://doi.org/10.1109/CVPR.2016.91

Sánchez-Vicinaiz, T. J., Camacho-Pérez, E., Castillo-Atoche, A. A., Cruz-Fernandez, M., García-Martínez, J. R., & Rodríguez-Reséndiz, J. (2024). MediaPipe frame and convolutional neural networks-based fingerspelling detection in mexican sign language. *Technologies*, *12*(8), 124. https://doi.org/10.3390/technologies12080124

Sapkota, R., Meng, Z., & Karkee, M. (2024). Synthetic meets authentic: Leveraging llm generated datasets for YOLO11 and YOLOv10-based apple detection through machine vision sensors. *Smart Agricultural Technology*, 100614. https://doi.org/10.1016/j.atech.2024.100614

Shahriar, S., Lund, B. D., Mannuru, N. R., Arshad, M. A., Hayawi, K., Bevara, R. V. K., Mannuru, A., & Batool, L. (2024). Putting GPT-4o to the sword: A

comprehensive evaluation of language, vision, speech, and multimodal proficiency. *Applied Sciences*, *14*(17), 7782. https://doi.org/10.3390/app14177782

Shen, L., Yang, Q., Zheng, Y., & Li, M. (2025). AutoIOT: LLM-driven automated natural language programming for AIoT applications. *Proceedings of the 31st Annual International Conference on Mobile Computing and Networking (MobiCom)*, 1–15. https://doi.org/10.1145/1234567.1234568

Shotton, J., Fitzgibbon, A., Cook, M., Sharp, T., Finocchio, M., Moore, R., Kipman, A., & Blake, A. (2011). Real-time human pose recognition in parts from single depth images. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 1297–1304. https://doi.org/10.1109/CVPR.2011.5995316

Simonyan, K., & Zisserman, A. (2015). Very deep convolutional networks for large-scale image recognition. *International Conference on Learning Representations (ICLR)*.

Singh, A., Maurya, S., Sati, M., Singh, S. K., & Jalal, A. S. (2022). Object detection using YOLO: Challenges, architectural successors, datasets and applications. *Multimedia Tools and Applications*, *82*, 11141–11177. https://doi.org/10.1007/s11042-022-13644-y

Sterling, C. H. (Ed.). (2007). *Military communications: From ancient times to the 21st century*. Bloomsbury Publishing USA.

Tan, K., Yao, J., Pang, T., Fan, C., & Song, Y. (2024). ELF: Educational LLM framework of improving and evaluating AI generated content for classroom

teaching. *ACM Journal of Data and Information Quality*, *16*(2), 1–15. https://doi.org/10.1145/3712065

Tang, Y., Wang, Y., Wu, H., Xu, Y., Xu, C., & Xu, C. (2024). Deepseekmoe: Towards ultimate expert specialization in mixture-of-experts language models. *arXiv preprint arXiv:2401.06066*.

Tian, N., et al. (2018). A cloud-based robust semaphore mirroring system for social robots. *2018 IEEE 14th International Conference on Automation Science and Engineering (CASE)*, 1351–1358. https://doi.org/10.1109/COASE.2018.8560553

Toshev, A., & Szegedy, C. (2014). DeepPose: Human pose estimation via deep neural networks. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 1653–1660. https://doi.org/10.1109/CVPR.2014.214

Virginia, B. A. M. (2024). Implementasi posenet dalam game semaphore untuk mengenali Gerakan tubuh pengguna. *Universitas Mercu Buana*. https://repository.mercubuana.ac.id/85784/

Wang, C.-Y., Liao, H.-Y. M., & Yeh, I.-H. (2024). Yolov11: An overview of the key architectural enhancements. *arXiv preprint arXiv:2410.17725*. https://doi.org/10.48550/arXiv.2410.17725

Wang, J., Tan, S., Zhen, X., Xu, S., Zheng, F., He, Z., & Shao, L. (2021). Deep 3D human pose estimation: A review. *Computer Vision and Image Understanding*, *210*, 103225. https://doi.org/10.1016/j.cviu.2021.103225

Woo, S., Park, J., Lee, J.-Y., & Kweon, I. S. (2018). CBAM: Convolutional block attention module. *Proceedings of the European Conference on Computer Vision (ECCV)*, 3–19.

Wu, Q., Xu, G., Zhang, S., Li, Y., & Fan, W. (2020). Human 3D pose estimation in a lying position by RGB-D images for medical diagnosis and rehabilitation. *Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, 5802–5805. https://doi.org/10.1109/EMBC44109.2020.9175320

Wu, Y., & Huang, T. S. (1999). Vision-based gesture recognition: A review. *Proceedings of the International Gesture Workshop*, 103–115. https://doi.org/10.1007/3-540-46616-9_11

Xenakis, A., Dimos, I., Feidakis, M., & Sotiropoulos, D. J. (2024). An LLM-based smart repository platform to support educators with computational thinking, AI, and STEM activities. *Empowering STEM Educators With Digital Tools*, 107–136. https://doi.org/10.4018/979-8-3693-9806-7.ch005

Xue, Y., Wang, W., Fang, M., Guo, Z., Ning, K., & Wang, K. (2024). Research on a high-efficiency goat individual recognition method based on machine vision. *Animals*, *14*, 3509. https://doi.org/10.3390/ani14233509

Yan, W. (2019). *Introduction to Intelligent Surveillance Surveillance: Data Capture, Transmission, and Analytics*. Springer.

Yan, W. (2023). *Computational Methods in Deep Learning: Theory, Algorithms, and Implementations*. Springer.

Yan, W. (2026). *Robotic Vision: From Deep Learning to Autonomous Systems*. Springer.

Yin, S., Fu, C., Zhao, S., Li, K., Sun, X., Xu, T., & Chen, E. (2024). A survey on multimodal large language models. *National Science Review*, *11*(12), nwae403. https://doi.org/10.1093/nsr/nwae403

Yu, H., Gan, A., Zhang, K., Tong, S., Liu, Q., & Liu, Z. (2024). Evaluation of retrieval-augmented generation: A survey. *arXiv preprint arXiv:2405.07437*. https://arxiv.org/abs/2405.07437

Zhang, J., Lin, Z., Xu, S., Tang, K., Wang, F., Gao, X., Chen, M., Zhuang, F., Jiang, Y., & Chen, E. (2024). AI2apps: A visual IDE for building LLM-based AI agent applications. *arXiv preprint arXiv:2404.04902*. https://arxiv.org/abs/2404.04902

Zhang, X., Song, Y., Song, T., Yang, D., Ye, Y., Zhou, J., & Zhang, L. (2023). Akconv: Convolutional kernel with arbitrary sampled shapes and arbitrary number of parameters. *arXiv preprint arXiv:2311.11587*. https://arxiv.org/abs/2311.11587

Zhang, Y., Liu, J., Wang, Z., Chen, L., & Yang, S. (2025). Challenges in ensuring AI safety in DeepSeek-R1 models. *arXiv preprint arXiv:2501.17030*. https://arxiv.org/abs/2501.17030

Zhao, X., Wang, L., Zhang, Y., Han, X., Deveci, M., & Parmar, M. (2024). A review of convolutional neural networks in computer vision. *Artificial Intelligence Review*. https://doi.org/10.1007/s10462-024-10721-6

Zhou, Q., Xu, K., Tian, Y., Hu, J., & Hu, J. (2022). Intelligent traffic signal control based on multi-agent deep reinforcement learning with edge computing. *Pro-*

cedia Computer Science, *199*, 228–235. https://doi.org/10.1016/j.procs.2022.01.030