Indonesian Journal of Data and Science

Volume 6 Issue 2
ISSN 2715-9936
https://doi.org/10.56705/ijodas.v6i2.276

Research Article

YOLOv8 Implementation on British Sign Language System with Edge Detection Extraction

Muhammad Rizqi Romadlon [1], Anik Nur Handayani [2,*]

[1] Universitas Negeri Malang, Malang, Indonesia, muhammad.rizqi.2105356@students.um.ac.id
[2] Universitas Negeri Malang, Malang, Indonesia, aniknur.ft@um.ac.id

Correspondence should be addressed to Anik Nur Handayani; aniknur.ft@um.ac.id

Received 18 January 2025; Accepted 29 April 2025; Published 31 July 2025

Abstract:

This study presents the development and implementation of a deep learning-based system for recognizing static hand gestures in British Sign Language (BSL). The system utilizes the YOLOv8 model in conjunction with edge detection extraction techniques. The objective of this study is to enhance the accuracy of recognition and facilitate communication for individuals with hearing impairments. The dataset was obtained from Kaggle and comprises images of various BSL hand signs captured against a uniform green background under consistent lighting conditions. The preprocessing steps entailed resizing the images to 640×640 pixels, implementing pixel normalization, filtering out low-quality images, and employing data augmentation techniques such as horizontal flipping, rotation, shear, and brightness adjustments to enhance robustness. Edge detection was implemented to accentuate the contours of the hand, thereby facilitating more precise gesture identification. Manual annotation was performed to generate both bounding boxes and segmentation masks, allowing for the training of two model variants: The first is YOLOv8 (non-segmentation), and the second is YOLOv8-seg (segmentation). Both models underwent training over a period of 100 epochs, employing the Adam optimizer and binary cross-entropy loss. The training-to-testing data splits utilized were 50:50, 60:40, 70:30, and 80:20. The evaluation metrics employed included mAP@50, precision, recall, and F1-score. The YOLOv8-seg model with an 80:20 split demonstrated the optimal performance, exhibiting a precision of 0.974, a recall of 0.968, and mAP@50 of 0.979. These metrics signify the model's capacity for robust detection and localization. Despite requiring greater computational resources, the segmentation model offers enhanced contour recognition, rendering it well-suited for high-precision applications. However, the generalizability of the model is constrained due to the employment of static gestures and controlled backgrounds. In the future, researchers should consider incorporating dynamic gestures, varied backgrounds, and uncontrolled lighting to enhance real-world performance.

Keywords: YOLOv8, British Sign Language (BSL), Hand Gesture Detection, Deep Learning, Segmentation Model.
Dataset link: https://www.kaggle.com/datasets/tr1gg3rtrash/yoga-posture-dataset

1. Introduction

In today's digital age, the ability to recognize and translate sign language is becoming increasingly important, especially for facilitating communication between individuals with hearing loss and the general public [1]. As technology evolves, tools that can detect and translate sign language gestures with precision are needed, as sign language is generally based on hand gestures and facial expressions. One of the main challenges is recognizing hand gestures in the context of communication without also recognizing the facial expressions that are often integral to understanding sign language [2].

This research focuses on static hand gesture recognition in British Sign Language (BSL). BSL has a unique structure and grammar that differs from other verbal languages, such as English, and it is an important communication system for the deaf community in the UK [3]. Therefore, developing a hand gesture recognition system for BSL is highly relevant, especially for supporting inclusive social interaction.

The application of the YOLOv8 (You Only Look Once) model for static hand gesture recognition in BSL offers great potential in improving the accuracy and speed of real-time object detection [4]. YOLOv8 is a recent object detection model that has been shown to offer significant performance improvements in terms of speed and accuracy when compared to previous versions [5]. This implementation of YOLOv8 in sign language recognition, which relies on edge detection extraction techniques, aims to clarify the contours of hand movements and facilitate the model's recognition of hand movement shape and direction with greater precision [6].

Edge detection extraction, a pivotal technique in this process, facilitates the model's identification of the outlines and contours of the hand, a critical component in enhancing detection efficiency and accuracy in static hand gesture recognition [7]. This approach is expected to enhance the quality of hand gesture detection utilized in BSL and to facilitate the development of a more effective and efficient sign language interpreter system [8], [9].

2. Method:

In this paper, we implement the YOLOv8 model to identify British Sign Language (BSL) signs using an edge detection extraction technique. The objective of this technology is to enable interaction between sign language users and visual-based systems [10]. The implementation process commences with the collection and processing of labeled BSL image data, followed by data augmentation and splitting stages to enhance model accuracy. The selection of YOLOv8 was predicated on its capacity to detect objects in real-time with a high degree of accuracy and its adaptability for customization in hand signal recognition applications, a finding corroborated by several recent international studies [11]. For instance, Madyanto et al. attained a maximum accuracy of 96% by employing YOLOv8 to detect Indonesian Sign Language (SIBI) letters with an average inference time of 4.6 milliseconds per image [11]. Another study by Alsharif et al. reported 98% precision and 99% F1-score in American Sign Language (ASL) alphabet recognition using YOLOv8 with transfer learning [12]. The implementation process commences with the collection and processing of BSL image data that has been labeled, followed by data augmentation and splitting stages to enhance model accuracy. The selection of YOLOv8 was predicated on its capacity to detect objects in real-time with a high degree of accuracy, in addition to its adaptability for hand gesture recognition applications [13]. The edge detection extraction method is employed to identify significant patterns in the image, thereby enhancing the efficacy of the detection process. It is hypothesized that this approach will yield an efficient and effective model for recognizing various BSL gestures with minimal error rate.

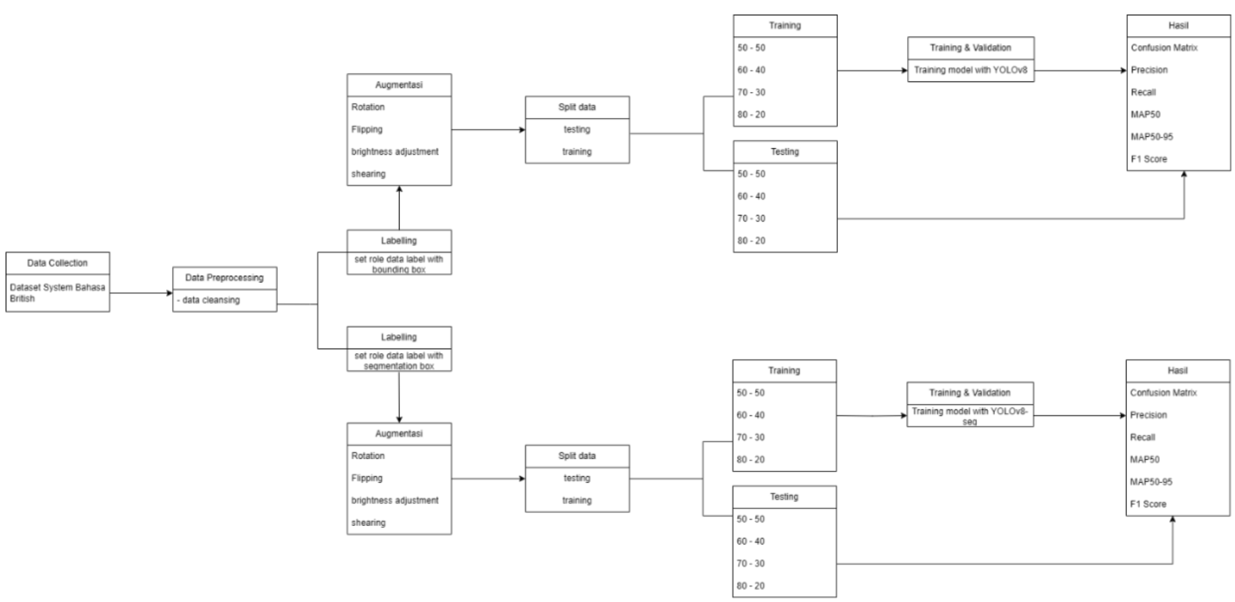

Figure 1. Research Flowchart

Data Collection

In the data collection stage for this research, we employed a dataset obtained from the Kaggle platform that focuses on British Sign Language (BSL) recognition. The dataset under consideration is composed of a variety of hand drawings that depict BSL symbols in different variations, including various viewing angles, lighting conditions, and skin colors. This data collection encompasses a substantial array of images representing diverse BSL signs, which are subsequently utilized to train and evaluate the sign detection model. A similar approach has been used in several previous studies that utilize image-based datasets for BSL recognition using deep learning [14], [15]. The acquired data undergoes subsequent processing through various stages, including pre-processing and labeling, to ensure the quality and accuracy of the constructed models. It has been demonstrated in several studies that variations in background and lighting conditions within the dataset can enhance model generalization [15], [16]. The utilization of this extensive dataset is anticipated to yield a more robust and accurate gesture recognition system.

Data Pre-processing

In the data pre-processing stage, the data that has been collected through the British Sign Language (BSL) dataset is processed to ensure the quality and readiness of the data to train the model. This process entails several critical steps, including image resizing for consistency, pixel normalization to circumvent scale discrepancies that could compromise model performance, and data augmentation techniques to diversify image variations through rotation, flipping, and altered lighting [17]. Augmentation is imperative for enhancing model generalization, particularly in the context of hand gesture recognition, a field that is profoundly influenced by factors such as shape and viewing angle [18]. Furthermore, images characterized by low quality or high noise are subjected to filtration and thresholding procedures to enhance the precision of the segmentation and classification processes [19]. Edge detection is also applied as part of the pre-processing to highlight hand contours that are important to the gesture detection system [20]. The objective of the pre-processing stage is to enhance the precision and efficacy of the model in recognizing BSL signs under diverse real-world conditions.

Labelling

In the labeling stage, each image in the British Sign Language (BSL) dataset is labeled according to the gesture shown by the hand in the image. The labeling process is conducted manually to ensure accuracy and consistency between the image and the label. Proper labeling is imperative as the model acquires the capacity to discern visual patterns based on the provided annotations [21]. The efficacy of consistent and accurate labeling in enhancing the accuracy of hand gesture recognition has been demonstrated in prior studies [22]. In the context of sign language recognition, meticulous labeling is paramount to differentiate between hand gestures that bear similarities yet possess divergent meanings [23]. A number of studies have demonstrated that a combination of manual labeling and automatic validation can enhance efficiency without compromising accuracy [24].

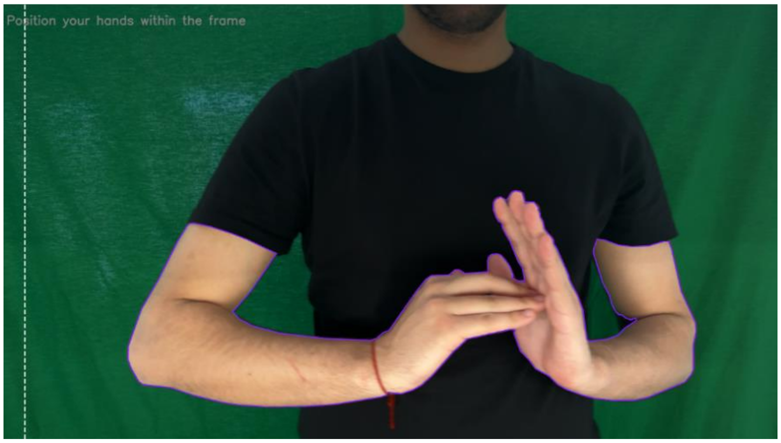

Figure 2. Example of Labeling Segmentation

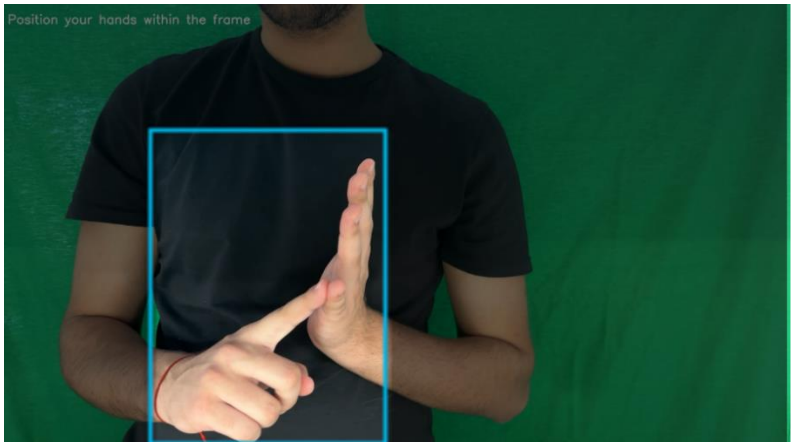

Figure 3. Example of Labeling Bounding Box

Augmentation

The specific data augmentation techniques that were implemented include horizontal flip, rotation, shear, and brightness adjustment [25]. The horizontal flip is employed to generate variations of mirror symmetry in the image, thereby simulating different hand perspectives. The rotation function enables the adjustment of images at various angles, thereby facilitating the model's development of an understanding of cues presented in diverse orientations. The shear technique was employed to tilt the image, thereby introducing realistic geometric distortions. In conclusion, the adjustment of brightness is executed to emulate the fluctuations in lighting conditions [26]. The integration of these augmentation techniques is intended to enhance the robustness of the model to input variations, while concurrently preventing overfitting by substantially augmenting the dataset's variability [27].

Figure 4. Example of Augmentation Image

Split Data

In the data sharing stage, this research utilizes the Roboflow platform to separate the dataset into a training set and a testing set with various ratio variations, such as 50:50, 60:40, 70:30, and 80:20, where the first number represents the percentage of data allocated for training and the second number for testing. The selection of the data sharing ratio exerts a substantial influence on the performance of the model, as it can impact the accuracy and generalization ability of the model. As demonstrated by Nguyen et al. [28], a ratio of 70:30 has been shown to yield optimal results in the prediction of soil shear strength. Furthermore, Nazarkar et al. [29] discovered that a ratio of 80:20 achieved the highest level of accuracy in their research on lung cancer classification. Therefore, the objective of this study is to evaluate the impact of the data sharing configuration on the accuracy and robustness of the BSL gesture recognition model.

Evaluation Matrix

In evaluating the performance of the British Sign Language (BSL) recognition model, this research utilizes several comprehensive evaluation metrics to ensure the accuracy and robustness of the model. The primary metric employed is Mean Average Precision (mAP), specifically mAP@50, which is particularly pertinent for object detection tasks such as hand gesture recognition [30]. mAP quantifies the model's efficacy in detecting hand gestures across various levels of Intersection over Union (IoU), which is a metric of the overlap between the prediction bounding box and the ground truth bounding box [31].

a) IoU (Intersection over Union)

The term "IoU" is used to denote the degree of similarity between two sets, that is, the extent to which model predictions align with ground truth. The IoU value is calculated by the following formula:

$$IoU(A,B) = \frac{|A \cap B|}{|A \cap B|} \tag{1}$$

b) Precision and Recall

Precision and recall are used to measure the effectiveness of the model in detecting objects, calculated with the following formula:

$$Precision = \frac{True\ Positives}{True\ Positives + False\ Positive} \tag{2}$$

$$Recall = \frac{True\ Positives}{True\ Positives + False\ Negative} \tag{3}$$

c) Average Precision (AP)

The calculation of Average Precision entails the ranking of predictions based on model confidence scores, followed by the calculation of the area under the precision-recall curve. The following formula is employed to calculate AP for a given class:

$$AP = \sum_{n=1}^{N} (Recall_n - Recall_{n-1}) \times Precision_n \tag{4}$$

d) Mean Average Precision (mAP)

The mean average precision (mAP) is determined by calculating the mean of the average precision (AP) values for all classes. In this study, mAP@50 is calculated with an IoU threshold of 0.5, which means the prediction will be considered correct if the IoU between the prediction and ground truth is greater than 0.5.

$$mAP = \frac{AP_{IoU=0.5} + AP_{IoU=0.55} + \cdots + AP_{IoU=0.95}}{k} \tag{5}$$

e) F1-Score

The F1-score is a metric that integrates precision and recall into a single value, thereby providing a more balanced assessment of the model's performance. The F1-score is calculated using the following formula:

$$F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{6}$$

The F1-score has been demonstrated to provide a more balanced assessment of model performance, particularly in datasets characterized by imbalanced classes [32]. Within the framework of BSL recognition, the F1-score metric is employed to evaluate the model's capacity to detect all pertinent hand signals with high recall, while concurrently minimizing detection errors with high precision [33].

3. Results and Discussion

Environment and Parameters

This research was conducted in a development environment designed to support intensive computing, especially in the context of training deep learning models. The hardware utilized in this study included an NVIDIA GeForce RTX 3060 GPU with 12GB of VRAM memory. This configuration significantly accelerated the training process of the YOLOv8 and YOLOv8-seg models. The selected software development environment is Visual Studio Code, which offers flexibility and effective integration for deep learning development utilizing Python.

With regard to the training parameters, the images in the dataset were resized to a resolution of 640×640 pixels to ensure consistency of input to the model. The YOLOv8 and YOLOv8-seg models were trained for 100 epochs to enable the models to learn relevant features from the BSL data. The selection of these parameters was informed by a comprehensive review of preliminary experiments and relevant literature, with the objective of attaining a balance between model accuracy and efficient training time. The selection of an RTX 3060 GPU and 640×640 pixels of image resolution was made with consideration for the available resources and computational requirements of the complex BSL dataset.

The evaluation of models is conducted employing Mean Average Precision (mAP) metrics, namely mAP@50, F1-score, and Intersection over Union (IoU), to assess the efficacy in detecting and classifying British Sign Language (BSL) hand gestures. In this section, an analysis of the experimental results will be conducted, followed by a comparison of the performance of the two models.

Training Method

In this study, YOLOv8 and YOLOv8-seg models are trained using Adam's optimization algorithm, which has been demonstrated to effectively address convergence issues in deep learning models. The loss function employed is binary cross-entropy, a suitable choice for object detection and segmentation tasks. The model has been meticulously trained from the ground up using BSL datasets that have been augmented via the Roboflow platform, with augmentation techniques such as horizontal flip, rotation, shear, and brightness adjustment. In order to prevent overfitting during the training process, dropout regularization and batch normalization techniques were applied.

The training process was executed for a total of 100 epochs, with the batch size being adjusted to correspond to the capacity of the 12GB NVIDIA GeForce RTX 3060 GPU. The data was segmented into training and testing sets with various proportionate variations, specifically 50:50, 60:40, 70:30, and 80:20, to assess the influence of disparate data proportions on model efficacy. The efficacy of the model was appraised during the training phase by employing the Mean Average Precision (mAP), F1-score, and Intersection over Union (IoU) metrics on the testing set. The training was implemented using the PyTorch library, which integrates seamlessly within the Visual Studio Code development environment

Results

In this study, YOLOv8 and YOLOv8-seg models are trained using Adam's optimization algorithm, which has been demonstrated to effectively address convergence issues in deep learning models. The loss function employed is binary cross-entropy, a suitable choice for object detection and segmentation tasks. The model has been meticulously trained from the ground up using BSL datasets that have been augmented via the Roboflow platform, with augmentation techniques such as horizontal flip, rotation, shear, and brightness adjustment. In order to prevent overfitting during the training process, dropout regularization and batch normalization techniques were applied.

Table 1. Performance Evaluation of Segmentation vs. Non-Segmentation Models on Various Data Ratios

Model	Scenario	Precision	Recall	mAP50	mAP50-95	F1-score
Non-segmentation	50:50	0.914	0.95	0.957	0.789	0.930
Non-segmentation	60:40	0.937	0.955	0.958	0.81	0.943

Model	Scenario	Precision	Recall	mAP50	mAP50-95	F1-score
Non-segmentation	70:30	0.95	0.949	0.96	0.829	0.950
Non-segmentation	80:20	0.948	0.942	0.961	0.82	0.941
segmentation	50:50	0.96	0.949	0.971	0.931	0.954
segmentation	60:40	0.973	0.956	0.974	0.947	0.964
segmentation	70:30	0.965	0.959	0.969	0.93	0.958
segmentation	80:20	**0.974**	**0.968**	**0.979**	**0.947**	**0.971**

The training process was executed for a total of 100 epochs, with the batch size being adjusted to correspond to the capacity of the 12GB NVIDIA GeForce RTX 3060 GPU. The data was segmented into training and testing sets with various proportionate variations, specifically 50:50, 60:40, 70:30, and 80:20, to assess the influence of disparate data proportions on model efficacy. The efficacy of the model was appraised during the training phase by employing the Mean Average Precision (mAP), F1-score, and Intersection over Union (IoU) metrics on the testing set. The training was implemented using the PyTorch library, which integrates seamlessly within the Visual Studio Code development environment.

To bolster the quantitative analysis, a visualization of the training results of the Segmentation model on the optimal 80:20 scheme was conducted. This visualization employs metrics and loss graphs during the training process, thereby offering a more profound understanding of the model's stability, convergence, and learning effectiveness. The subsequent figure illustrates the model's consistent reduction in loss and its steady enhancement in pivotal evaluation metrics, including Precision, Recall, mAP50, and mAP50-95.

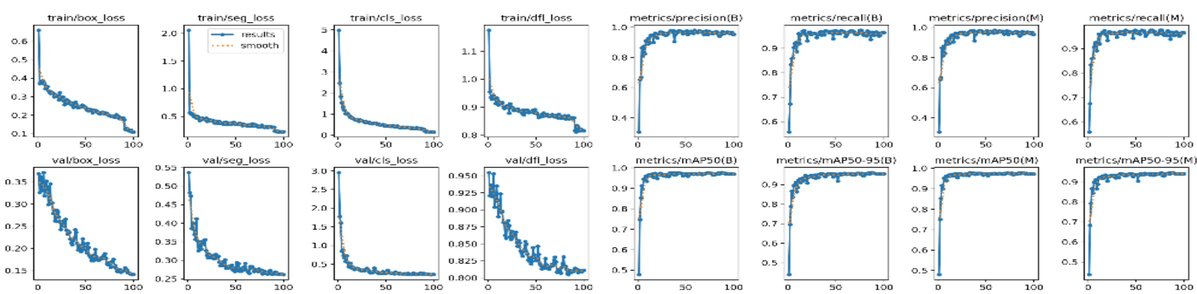

Figure 5. Result of 80:20 Segmentation

As demonstrated in **Figure 5**, it is evident that the training loss and validation loss values undergo a substantial decrease, approaching a convergence point, thereby signifying that the model learns efficiently without overfitting. Concurrently, evaluation metrics such as Precision, Recall, mAP50, and mAP50-95 demonstrate a consistent upward trend, approaching values close to 1.0. This finding indicates that the model demonstrates a high degree of accuracy in recognizing and localizing hand signals, exhibiting effective generalization capabilities when applied to test data. The consistent enhancement in the mAP50-95 metric signifies the model's capacity to detect objects of diverse sizes and complexity levels. Consequently, the integration of the Segmentation architecture and a substantial proportion of training data in the 80:20 scheme was identified as the most optimal configuration to generate a precise, consistent, and dependable model.

Testing

A series of experiments were conducted to assess the efficacy of detection models that rely on segmentation and those that do not utilize it in recognizing static hand gestures as representations of Filipino gestures. The initial figure presents the detection outcomes utilizing the Non-Segmentation model, which successfully identifies hand position through the implementation of a bounding box and attains a confidence score of 0.96. This model demonstrates efficacy in swiftly and accurately identifying hand-objects; however, it lacks the capability to discern hand-objects with high precision against their background. Conversely, the second image presents the outcome of the Segmentation model, which demonstrates an ability to meticulously delineate the hand contour from the background. The model in question produces a confidence score of 0.99, with the hand area accurately highlighted through the prediction mask. While segmentation necessitates greater computational demands, the resulting data offers a more informative depiction of the object and can be employed in subsequent, more intricate phases of gesture recognition.

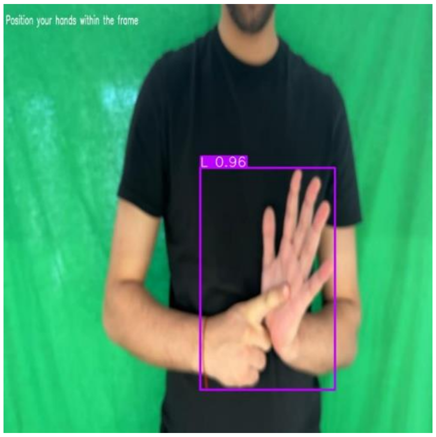

Figure 6. Example of Non-Segmentation Testing Result

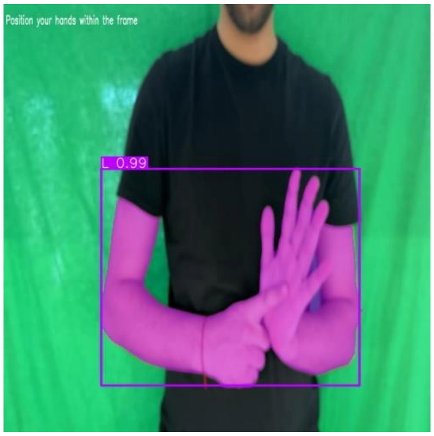

Figure 7. Example of Non-Segmentation Testing Result

Each approach has its own advantages. The Non-Segmentation model demonstrates a superior capacity for time and resource efficiency, as it relies solely on the bounding box as an indicator of object presence. However, the Segmentation model offers certain advantages in terms of spatial precision, a critical consideration in hand gesture systems where the position and shape of the fingers play a significant role. These results suggest that the Segmentation model is more suitable for applications that require in-depth analysis of object shapes.

However, it must be acknowledged that this study is not without significant limitations. The dataset under consideration exclusively encompasses static hand gestures set against a uniform green background and characterized by relatively consistent lighting conditions. This limitation renders the model susceptible to overfitting under specific conditions and has not been demonstrated to be reliable for application to complex backgrounds, uneven lighting conditions, or dynamic hand gestures that are prevalent in natural sign communication.

4. Conclusion

However, it must be acknowledged that this study is not without significant limitations. The dataset under consideration exclusively encompasses static hand gestures set against a uniform green background and characterized by relatively consistent lighting conditions. This limitation renders the model susceptible to overfitting under specific conditions and has not been demonstrated to be reliable for application to complex backgrounds, uneven lighting conditions, or dynamic hand gestures that are prevalent in natural sign communication.

This research successfully implements the YOLOv8 model for static hand gesture recognition in British Sign Language (BSL) using edge detection extraction techniques. The present study utilizes both segmentation and non-segmentation models to assess the efficacy of the model in detecting hand gestures, considering a range of data sharing ratios. The findings indicate that the segmentation model with an 80:20 data ratio exhibits optimal performance, characterized by high precision, recall, and mAP50 values of 0.974, 0.968, and 0.979, respectively. This model has been demonstrated to achieve a high degree of precision in the delineation of hand contours, thereby enhancing its efficacy in the detection of gestures with greater accuracy. However, it should be noted that this enhanced performance is accompanied by an increase in the computational demands, resulting in a need for additional time to complete the processing tasks.

This research makes a substantial contribution to the development of an image-based sign language recognition system. The implementation of this system has the potential to facilitate communication between individuals with hearing disabilities and the general public. It should be noted that the model has not yet been tested under real-world conditions involving complex backgrounds, uncontrolled lighting, or dynamic hand movements. Consequently, to enhance the model's performance and its capacity for generalization, it is advised that subsequent research endeavors focus on augmenting the dataset with variations in background, lighting, and dynamic hand movements.

References:

[1] C. Arya, A. Gusain, Kunal, M. Diwakar, I. Gupta, and N. K. Pandey, “A Lightweight Solution for Real-Time Indian Sign Language Recognition,” in *2024 International Conference on Artificial Intelligence and Emerging Technology (Global AI Summit)*, IEEE, Sep. 2024, pp. 1359–1364. doi: 10.1109/GlobalAISummit62156.2024.10947786.

[2] A. N. Handayani, M. I. Akbar, H. Ar-Rosyid, M. Ilham, R. A. Asmara, and O. Fukuda, “Design of SIBI Sign Language Recognition Using Artificial Neural Network Backpropagation,” in *2022 2nd International Conference on Intelligent Cybernetics Technology & Applications (ICICyTA)*, IEEE, Dec. 2022, pp. 192–197. doi: 10.1109/ICICyTA57421.2022.10038205.

[3] J. J. Bird, A. Ekárt, and D. R. Faria, “British Sign Language Recognition via Late Fusion of Computer Vision and Leap Motion with Transfer Learning to American Sign Language,” *Sensors*, vol. 20, no. 18, p. 5151, Sep. 2020, doi: 10.3390/s20185151.

[4] Y. Zhang and X. Jiang, “Recent Advances on Deep Learning for Sign Language Recognition,” *Computer Modeling in Engineering & Sciences*, vol. 139, no. 3, pp. 2399–2450, 2024, doi: 10.32604/cmes.2023.045731.

[5] B. Alsharif, E. Alalwany, A. Ibrahim, I. Mahgoub, and M. Ilyas, “Real-Time American Sign Language Interpretation Using Deep Learning and Keypoint Tracking,” *Sensors*, vol. 25, no. 7, p. 2138, Mar. 2025, doi: 10.3390/s25072138.

[6] M. M. Alnfiai, “Deep Learning-Based Sign Language Recognition for Hearing and Speaking Impaired People,” *Intelligent Automation & Soft Computing*, vol. 36, no. 2, pp. 1653–1669, 2023, doi: 10.32604/iasc.2023.033577.

[7] M. D. Montefalcon, J. R. Padilla, and R. Rodriguez, “Filipino Sign Language Recognition Using Long Short-Term Memory and Residual Network Architecture,” 2023, pp. 489–497. doi: 10.1007/978-981-19-2397-5_45.

[8] I. Tupal, M. Cabatuan, and M. Manguerra, “Recognizing Filipino Sign Language with InceptionV3, LSTM, and GRU,” in *2022 IEEE 14th International Conference on Humanoid, Nanotechnology, Information Technology, Communication and Control, Environment, and Management (HNICEM)*, IEEE, Dec. 2022, pp. 1–5. doi: 10.1109/HNICEM57413.2022.10109284.

[9] C. B. Jetomo and M. L. D. De Lara, “Filipino sign language alphabet recognition using Persistent Homology Classification algorithm,” *PeerJ Comput Sci*, vol. 11, p. e2720, Feb. 2025, doi: 10.7717/peerj-cs.2720.

[10] A. N. Handayani, T. Andriyanto, D. F. Azizah, M. Z. Wiryawan, and H. A. Rosyid, "Comparison of ResNet-50 and EfficientNet-B0 Method for Classification of Indonesian Sign Language System (SIBI) Using Multi Background Dataset," in *2024 International Conference on Computer Engineering, Network, and Intelligent Multimedia (CENIM)*, IEEE, Nov. 2024, pp. 1–6. doi: 10.1109/CENIM64038.2024.10882836.

[11] H. A. AbdElghfar *et al.*, "QSLRS-CNN: Qur'anic sign language recognition system based on convolutional neural networks," *The Imaging Science Journal*, vol. 72, no. 2, pp. 254–266, Feb. 2024, doi: 10.1080/13682199.2023.2202576.

[12] B. Alsharif, E. Alalwany, and M. Ilyas, "Transfer learning with YOLOV8 for real-time recognition system of American Sign Language Alphabet," *Franklin Open*, vol. 8, p. 100165, Sep. 2024, doi: 10.1016/j.fraope.2024.100165.

[13] T.-H. Nguyen, B.-V. Ngo, and T.-N. Nguyen, "Vision-Based Hand Gesture Recognition Using a YOLOv8n Model for the Navigation of a Smart Wheelchair," *Electronics (Basel)*, vol. 14, no. 4, p. 734, Feb. 2025, doi: 10.3390/electronics14040734.

[14] A. S. G. Raharjo and E. Sugiharti, "Alphabet Classification of Sign System Using Convolutional Neural Network with Contrast Limited Adaptive Histogram Equalization and Canny Edge Detection," *Scientific Journal of Informatics*, vol. 10, no. 3, pp. 239–250, Jun. 2023, doi: 10.15294/sji.v10i3.44137.

[15] M. Mahyoub, F. Natalia, S. Sudirman, and J. Mustafina, "Sign Language Recognition using Deep Learning," in *2023 15th International Conference on Developments in eSystems Engineering (DeSE)*, IEEE, Jan. 2023, pp. 184–189. doi: 10.1109/DeSE58274.2023.10100055.

[16] Y. Wang, Z. Hao, X. Dang, Z. Zhang, and M. Li, "UltrasonicGS: A Highly Robust Gesture and Sign Language Recognition Method Based on Ultrasonic Signals," *Sensors*, vol. 23, no. 4, p. 1790, Feb. 2023, doi: 10.3390/s23041790.

[17] C. Shorten and T. M. Khoshgoftaar, "A survey on Image Data Augmentation for Deep Learning," *J Big Data*, vol. 6, no. 1, p. 60, Dec. 2019, doi: 10.1186/s40537-019-0197-0.

[18] B.-A. Awaluddin, C.-T. Chao, and J.-S. Chiou, "A Hybrid Image Augmentation Technique for User- and Environment-Independent Hand Gesture Recognition Based on Deep Learning," *Mathematics*, vol. 12, no. 9, p. 1393, May 2024, doi: 10.3390/math12091393.

[19] R. B. Shamshanovna *et al.*, "Combined-adaptive image preprocessing method based on noise detection," *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 15, no. 2, p. 1584, Apr. 2025, doi: 10.11591/ijece.v15i2.pp1584-1592.

[20] M. Arulmozhi, N. G. Iyer, and C. Amutha, "Edge Detection-Based Medibot and Adoption of Deep Learning for Hand Gesture Recognition," 2023, pp. 275–283. doi: 10.1007/978-981-99-2349-6_25.

[21] S. T. Abd Al-Latief, S. Yussof, A. Ahmad, and S. Khadim, "Deep Learning for Sign Language Recognition: A Comparative Review," *Journal of Smart Internet of Things*, vol. 2024, no. 1, pp. 77–116, Jun. 2024, doi: 10.2478/jsiot-2024-0006.

[22] N. Adaloglou *et al.*, "A Comprehensive Study on Deep Learning-Based Methods for Sign Language Recognition," *IEEE Trans Multimedia*, vol. 24, pp. 1750–1762, 2022, doi: 10.1109/TMM.2021.3070438.

[23] S. Sharma and S. Singh, "Vision-based hand gesture recognition using deep learning for the interpretation of sign language," *Expert Syst Appl*, vol. 182, p. 115657, Nov. 2021, doi: 10.1016/j.eswa.2021.115657.

[24] B. Emek Soylu, M. S. Guzel, G. E. Bostanci, F. Ekinci, T. Asuroglu, and K. Acici, "Deep-Learning-Based Approaches for Semantic Segmentation of Natural Scene Images: A Review," *Electronics (Basel)*, vol. 12, no. 12, p. 2730, Jun. 2023, doi: 10.3390/electronics12122730.

[25] L. Alzubaidi *et al.*, "Review of deep learning: concepts, CNN architectures, challenges, applications, future directions," *J Big Data*, vol. 8, no. 1, p. 53, Mar. 2021, doi: 10.1186/s40537-021-00444-8.

[26] M. Xu, S. Yoon, A. Fuentes, and D. S. Park, "A Comprehensive Survey of Image Augmentation Techniques for Deep Learning," *Pattern Recognit*, vol. 137, p. 109347, May 2023, doi: 10.1016/j.patcog.2023.109347.

[27] M. Xu, S. Yoon, A. Fuentes, and D. S. Park, "A Comprehensive Survey of Image Augmentation Techniques for Deep Learning," *Pattern Recognit*, vol. 137, p. 109347, May 2023, doi: 10.1016/j.patcog.2023.109347.

[28] Q. H. Nguyen *et al.*, "Influence of Data Splitting on Performance of Machine Learning Models in Prediction of Shear Strength of Soil," *Math Probl Eng*, vol. 2021, pp. 1–15, Feb. 2021, doi: 10.1155/2021/4832864.

[29] A. Nazarkar, H. Kuchulakanti, C. S. Paidimarry, and S. Kulkarni, "Impact of Various Data Splitting Ratios on the Performance of Machine Learning Models in the Classification of Lung Cancer," 2023, pp. 96–104. doi: 10.2991/978-94-6463-252-1_12.

[30] R. Padilla, W. L. Passos, T. L. B. Dias, S. L. Netto, and E. A. B. da Silva, "A Comparative Analysis of Object Detection Metrics with a Companion Open-Source Toolkit," *Electronics (Basel)*, vol. 10, no. 3, p. 279, Jan. 2021, doi: 10.3390/electronics10030279.

[31] T.-C. Lin, H.-H. Chen, K.-S. Chen, Y.-P. Chen, and S.-H. Chang, "Decision-Making Model of Performance Evaluation Matrix Based on Upper Confidence Limits," *Mathematics*, vol. 11, no. 16, p. 3499, Aug. 2023, doi: 10.3390/math11163499.

[32] M. S. Sarowar, N. E. J. Farjana, Md. A. I. Khan, M. A. Mutalib, S. Islam, and M. Islam, "Hand Gesture Recognition Systems: A Review of Methods, Datasets, and Emerging Trends," *Int J Comput Appl*, vol. 187, no. 2, pp. 1–33, May 2025, doi: 10.5120/ijca2025924776.

[33] A. F. AlShammari, "Implementation of Model Evaluation Using Confusion Matrix in Python," *Int J Comput Appl*, vol. 186, no. 50, pp. 42–48, Nov. 2024, doi: 10.5120/ijca2024924236.