American Sign Language Detection using YOLOv5 and YOLOv8

Shobhit Tyagi
Department of Computer Science & Engineering, School of Engineering & Technology, Sharda University, India

Prashant Upadhyay (prashant.upadhyay@sharda.ac.in)
Department of Computer Science & Engineering, School of Engineering & Technology, Sharda University, India

Hoor Fatima
Department of Computer Science & Engineering, School of Engineering & Technology, Sharda University, India

Sachin Jain
Department of Computer Science & Engineering, Ajay Kumar Garg Engineering College, Ghaziabad, India

Avinash Kumar Sharma
Department of Computer Science & Engineering, ABES Institute of Technology Ghaziabad

Research Article

Keywords: CNNs, Sign Language Recognition, YOLOv8, Transfer Learning, Direct Location Prediction

Posted Date: July 6th, 2023

DOI: https://doi.org/10.21203/rs.3.rs-3126918/v1

Abstract

In the modern world, culture and religion are diverse and widespread. Sign language culture had grown since its emergence in the American School for the Deaf (ASD) in 1817. In a world where computers are now solving real-time applications and issues using deep learning, Sign language (SL) is one of those. YOLO is an object detection and classification algorithm that uses Convolutional neural network (CNN) to achieve high performance and accuracy. The paper aims to detect American sign language using YOLO models and compare different YOLO algorithms by implementing a custom model for recognizing sign language. The experiments show that the latest YOLOv8 gave better results than other YOLO versions regarding precision and mAP, while YOLOv7 has a higher recall value during testing than YOLOv8. The proposed model is lightweight, fast and uses the American sign language letters dataset for training and testing. The custom model achieved 95% precision, 97% recall, and 96% mAP @0.5, showing the model capabilities in real-time hand gesture recognition.

1. Introduction

According to the WHO [1], over 1.5 billion people suffer from hearing loss around the globe. Additionally, about one billion teenagers are at risk of developing hearing loss due to the improper use of earbuds and headphones. Such disabilities severely affect children's physical and mental health, education, and employment opportunities. Older adults often experience hearing loss, social isolation, loneliness, and frustration. Children with hearing loss may face delayed language development and communication difficulties. Unfortunately, hearing loss is often not adequately accommodated in private and government settings, which negatively impacts employment opportunities and academic achievement. Children with hearing impairment have a hard time understanding others and receive little to no education. There is no universal sign language that is used by all deaf individuals. Sign language detection poses several challenges due to its unique characteristics and the complexity of capturing and interpreting sign language. Although there exists some noticeable similarities, each country has its own unique way of using sign language. Some of the key challenges in sign language detection include:

1. **Variations in sign languages:** SL vary across different regions and countries. Each sign language has its own vocabulary, grammar, and syntax. Detecting and understanding different sign languages require language-specific models and datasets, making developing a universal sign language detection system challenging.
2. **Gesture recognition:** Sign languages involve a combination of simultaneous hand movements, facial expressions, body postures, and other non-manual markers. Capturing and recognizing these subtle and dynamic gestures accurately is a complex task.
3. **Data scarcity:** SL data is relatively scarce compared to spoken languages. Building accurate sign language detection models requires large amounts of annotated data, which is often limited, especially for certain sign languages or specific sign variations.

4. **Background noise and occlusion:** Sign language is often performed in real-world environments, which can introduce background noise, occlusions, and cluttered backgrounds. These factors can interfere with the visibility of the signer's hands and facial expressions, making it difficult to detect and interpret signs accurately.
5. **Ambiguity and context dependency:** Sign languages rely on facial expressions, context, and body movements to express meaning. Isolated signs may have multiple interpretations depending on the surrounding signs or the speaker's intentions.
6. **Real-time detection and latency:** Low latency is crucial in applications where sign language detection needs to be performed in real time, such as sign language interpretation systems or assistive technologies.

Another major issue faced by the deaf community is that some languages are officially recognized while others are not recognized by the government or the institutions. The National Association of Deaf [2] reports that there are 18 million individuals with hearing impairments in India. Innovative Artificial Intelligence approaches [3] could be useful for a sign language interpretation application. The field of artificial intelligence (AI) involves developing smart computers that can learn from raw data. These machines are capable of making decisions even in unfamiliar situations. There are many systems and methods that have been developed to tackle such problems. This research aims to compare two algorithms of the same object detection family and also discuss the characteristics and advantages of each algorithm in their own respective ways [4].

It is observed that many institutions tend to give priority to American Sign Language (ASL) over Indian Sign Language (ISL). Nevertheless, ASL is chosen in this project as it is suitable for training the models using alphabets and numbers. This paper introduces a custom CNN model to predict the gestures performed in real-time at high accuracy. The experiment-based functional model is capable of recognizing hand gestures that can be deployed in any real-life situation.

2. Related Work

Sign language is unique due to its nature of using different kinds of physical gestures instead of using specific sounds. A substantial amount of work had been done in facial and gesture recognition. Based on our research, the existing methods can be divided into two categories

In the first method, an external device is used for sign recognition; while another method for detecting and recognizing hand gestures [10] is to use deep learning. Several researchers have developed novel techniques for SL recognition with solutions to different aspects such as cost, latency, performance and portability. A. Bhattacharya [5] talks about training a classifier on a dataset of 24 gestures that can be easily spelt on fingers using the "bag of visual words approach".

In order to describe each image as a graph (particularly a histogram) indicating the frequency of observed gestures in that image, this method first identifies the features in the images that are then

clustered to construct a book containing those gestures and their frequency. T. Starner [6] showed two real-time hidden Markov model-based systems that only require a camera to track the user's user's to recognize intermediate-level continuous ASL.

The original system had a desk-mounted camera to watch the user and had a 92% accuracy rate. The second device, which reaches 98% accuracy, mounts the camera in the user's hat. L. Aziz [7] surveyed the most recent developments in visual object detection with deep learning, covering around 300 methods, including region-based object detection methods such as SPPnet [30], Faster R-CNN [32], Classification and regression-based object detection methods such as YOLO [15, 16], etc. The authors also researched and analyzed publicly available benchmark datasets based on their origin, usage, advantages and limitations along with their evaluation metrics. D. Naglot [8] recognized several signs using a Multi-layer Perceptron neural network on 520 samples contained in a dataset. Table 1. shows the recent state-of-the-art methods for sign language detection.

Table 1
Related work

AUTHOR	METHOD	ALGORITHM USED	DATASET	ACCURACY
O. Vedak et al. [2]	SL interpreter utilizing machine learning and image processing	HOG & SVM	6000 images of 26 English language alphabets.	88%
A. Bhattacharya et al. [5]	Training a SL classifier using the bag of visual words approach	SURF and BRISK features are used for automatic feature identification. Algorithms such as KNN, SVM, etc, are used for evaluation.	4972 images of 24 static single-handed fingerspelling gestures on a plain background	**SURF Features** SVM – 91.35 KNN – 86.97 **BRISK Features** SVM – 91.15 KNN – 87.38
D. Naglot et al. [8]	In Real-time SL detection using Leap Motion Controller	MLP and Back propagation (An idea for a categorization model that accepts a set of features as input.)	26 different alphabets of ASL of 520 samples (consisting of 20 samples of each alphabet)	96.15%
C. Chuan et al. [9]	SL recognition using Leap Motion Sensor	K-NN & SVM (Leap motion sensors and a webcam have a lot of potential to advance SL learning techniques.)	Four data sets were collected from the two signers with two sets from each individual.	KNN – 7.78% SVM – 79.83%
Z. wang et al. [11]	End-to-end SL detection in real time.	DeepSLR technology" is used to record the coarse and finger movements along with several sensors.	5586 sign words and 26 alphabet signs	-
M. Fernando et al. [12]	Low-cost approach for Real-time SL recognition	Active shape models	50 signs, 5 signs from 10 users (A, B, C, D, V Signs)	76%
K. Dutta et al. [13]	Machine learning methods for ISL detection	ISL	220 images of double handed ISL alphabets and 800 images of single handed Indian SL alphabets	92%

AUTHOR	METHOD	ALGORITHM USED	DATASET	ACCURACY
T. Mangamuri et al. [14]	Two-handed ISL dataset for comparing machine learning classification models	HOG	THISL dataset with 26 motions, each of which represents a letter of the English alphabet	87.67%

3. Methodology

The most commonly used datasets for object detection [20, 21] and segmentation are Pascal VOC 2007 [28] and Microsoft COCO [17, 29]. This research focuses on YOLO version 5 and 8, as these models have better performance in recognizing sign language with high accuracy. The proposed methodology is shown in Fig. 1.

3.1 YOLOv5

YOLOv5 is the deep learning-based architecture that is used to conduct this experiment. YOLOv5 [18] is lightweight and fast and needs less computational power than the other current state-of-the-art architecture model while keeping the accuracy near the current state-of-the-art detection models. The first model to be published without a supporting paper was Glenn Jocher's YOLOv5, which was also marked as being under "ongoing development" on its repo. Glenn Jocher is a researcher at Ultralystics LLC. The Github repository for YOLOv5 is available : Releases · ultralytics/yolov5 (github.com). The publication date was June 2020. Python was used to construct YOLOv5 on IoT devices [26], which simplifies installation and integration, as opposed to C as in past versions. Also, the PyTorch community was larger than the Darknet community, giving it more possibility for growth and expansion in the future. [19]. It is much faster than the other YOLO models. YOLOv5 uses CSPNET [29] as the backbone to extract the feature map from the image. It also uses Path Aggregation Network (PANet) [31] to boost the information flow. Figure 3 shows the architecture of YOLOv5. We are using YOLOv5 for the following reasons:

1) The YOLOv5 has SOTA features such as an activation function, hyperparameter, data augmentation technique and an easy-to-use manual

2) The model has a simple architecture, which makes it computationally easy to train even with small resources.

3) The small and lightweight nature of YOLOv5 makes it useful for mobile devices and embedded applications.

3.2 YOLOv8

Also authored by Glenn Jocher and launched on January 23rd 2023, YOLOv8 is the latest in the family of algorithms and is still in development, along with adding many new features such as Anchor free

detection and mosaic augmentation. A CLI that is included with YOLOv8 makes training a model easier to understand. Moreover, there is a Python package [27] that offers a smoother development experience than the previous model. The Github repository for the YOLOv8 is available : GitHub - ultralytics/ultralytics: NEW - YOLOv8 in PyTorch > ONNX > CoreML > TFLite. Predictions for both the bounding boxes as well as the classes are produced when the input image has been evaluated once. The algorithm is as since both the predictions – bounding box and classification, are performed simultaneously. The provided image is initially converted into a grid of equal lengths (S x S). Next, confidence scores are defined for each grid cell's "b" bounding boxes as shown in equation (i). [22]. Confidence is the probability that an object exists in every bounding box.

*Confidence (C) = P(object) ** IOU_{pred}^{target} –(i)

Where, IOU = Intersection over union

IOU [23] stands for a fractional value in the range of 0 and 1. Union is the total area between the predicted and the target areas, whereas intersection is the overlap between the predicted bounding box and the target area. The ideal value is close to 1, which denotes that the estimated bounding box is near the target region. Along with this, every grid cell also predicts the Confidence conditional class probability as shown in equation (ii) and (iii).

C = P (Class $_i$ *| object) * P(object) ** $I.O.U_{pred}^{target}$ –(ii)

C = P(Class $_i$*) ** $I.O.U_{pred}^{target}$ –(iii)

Now coming to the loss function, it is calculated by summing all the bounding box parameter's loss function result as shown in equation (iv),

$$\lambda_{coord}\sum_{i=0}^{S^2}\sum_{j=0}^{b}1_{ij}^{obj}\left[(x_i-\widehat{x}_i)^2+(y_i-\hat{y}_i)^2\right]$$

$$+\lambda_{coord}\sum_{i=0}^{S^2}\sum_{j=0}^{b}1_{ij}^{obj}\left[\left(\sqrt{w_i}-\sqrt{\widehat{w}_i}\right)^2+\left(\sqrt{h_i}-\sqrt{\widehat{h}_i}\right)^2\right]$$

$$+\sum_{i=0}^{S^2}\sum_{j=0}^{b}1_{ij}^{obj}\left(C_i-\widehat{C}_i\right)^2+\lambda_{noobj}\sum_{i=0}^{S^2}\sum_{j=0}^{b}1_{ij}^{obj}\left(C_i-\widehat{C}_i\right)^2$$

$$+\sum_{i=0}^{S^2}1_i^{obj}\sum_{c\epsilon classes}\left(p_i(c)-\hat{p}_i(c)\right)^2$$ –(iv)

The given equation carries five important terminologies defined as follows [23]:

- *(x* $_i$*, y*$_i$*)* – coordinates of target center (bounding box)
- *(*$\widehat{x}$ $_i$*,* $\hat{y}$ $_i$*)* – coordinates of predicted center (bounding box)

- (w_i, h_i) – dimensions of target
- $(\widehat{w}_i, \widehat{h}_i)$ – dimensions of predicted

Equation's first step computes the loss associated with the bounding box using coordinates (x_i,y_i). If an object is present inside the j^{th} forecasted bounding box inside the i^{th} cell, 1_{ij}^{obj} is defined as 1, and if it is not true then 0 as shown in equation (v). The prediction with the highest current IOU with the target region will be considered "responsible" for predicting an object by the predicted bounding box [19].

$$\lambda_{coord}\sum_{i=0}^{S^2}\sum_{j=0}^{b}1_{ij}^{obj}\left[(x_i-\widehat{x}_i)^2+(y_i-\hat{y}_i)^2\right] \text{–(v)}$$

The next portion is responsible for calculating the error in the prediction of the dimensions of the bounding box. The scale of the inaccuracy in the large boxes, however, is less impactful on the equation itself than it is in the small boxes. The square roots of width and height, which are both normalized in the range of 0 and 1, make the discrepancies between smaller values bigger than those between larger ones. As a result, rather than using the dimension values directly, the bounding box's square root is employed as shown in equation (vi).

$$\lambda_{coord}\sum_{i=0}^{S^2}\sum_{j=0}^{b}1_{ij}^{obj}\left[\left(\sqrt{w_i}-\sqrt{\widehat{w}_i}\right)^2+\left(\sqrt{h_i}-\sqrt{\widehat{h}_i}\right)^2\right] \text{–(vi)}$$

The loss value of confidence is calculated in the next section for both circumstances, regardless of whether the object is present inside the bounding box or not. However, if that predictor is in charge of the target box, only then the loss function shall penalize the object confidence mistake as shown in equation (vii). If there is an object in the cell, 1_{ij}^{obj} equals 1, else results 0.

$$\sum_{i=0}^{S^2}\sum_{j=0}^{b}1_{ij}^{obj}\left(C_i-\widehat{C}_i\right)^2+\lambda_{noobj}\sum_{i=0}^{S^2}\sum_{j=0}^{b}1_{ij}^{obj}\left(C_i-\widehat{C}_i\right)^2 \text{–(vii)}$$

With the exception of 1_{ij}^{obj}, which is needed because the algorithm does not penalize classification errors if there are no objects present in the cell, the last portion computes the loss of class probability [19] as shown in equation (viii) .

$$\sum_{i=0}^{S^2}1_i^{obj}\sum_{cc classes}\left(p_i\left(c\right)-\hat{p}_i\left(c\right)\right)^2 \text{–(viii)}$$

The first step is to install and initialize both the algorithms – YOLOv5 and YOLOv8, both the algorithms will be running simultaneously on different devices with similar specifications all so it is easier to compare both results and also less time consuming. The data set that is chosen for this work is “American Sign Language letters” [23] from the publicly available datasets on Roboflow. Along with that, we will use PyTorch which is based on the well-known Torch library. Moreover, a Python-based library that is more frequently used for computer vision and natural language processing. When downloading a

dataset from Roboflow, many methods are provided as to how that dataset should be implemented in the model. One of those methods is to apply the pytorch code that installs a roboflow package and also downloads the dataset directly into the directory of the program. Another major package that is downloaded is Ultralytics, this package provides all the versions of the YOLO algorithm hence making it optimal for this particular program.

4. Results

The dataset used contains images for testing, training and validation in the ratio of 1: 21: 2 respectively (72 for testing, 1512 for training, 144 for validation) and the new model has been trained for 80 epochs. Figure 2. shows the confusion matrix of v5 and v8 on the ASL dataset. It is worth noting that as compared to YOLOv5, YOLOv8's confusion matrix shows more discrepancies, which makes v8 vulnerable to unseen images and real life applications. There are 12 occurrences, other than background in YOLOv5 where the predicted value is not the same as the true values.

While in YOLOv8's case, those occurrences are 13. In hindsight, this comparison may seem redundant but after observing the two matrices, a conclusion is reached that background prediction for both cases are highly differentiable. One difference between v5 and v8 is the number of iterations needed to reach the minimum, v8 requires less number of iterations as compared to the v5. This may have happened because YOLOv8 is still in its first few stages of development as compared to its predecessor.

Next noticeable difference is the significance of each wrong prediction, the other model presents 7 of those specific cases. Figure 3 shows the comparison of the loss reduction of both YOLO versions, which shows that v5 has descended faster than the v8. Almost at both bounding box and classification loss, YOLOv8 gives a quicker minimum point which is around the 40th epoch mark; contrasting that, YOLOv5 results give that point around after the 50th epoch, presumably around the 60th epoch or higher as shown in Table 2. The custom model achieved 95% precision, 97% recall, and 96% mAP @0.5, showing the model capabilities in real-time hand gesture recognition. The bounding box loss in v5 and v8 base models is 0.326 and 0.312,while the classification loss of v5 and v8 models is 0.191 and 0.175. This shows that YOLOv8 is much better at detecting objects, while YOLOv8 has much classification accuracy. Overall, v8 is more accurate and faster at detecting objects and recognizing hand gestures. YOLOv5 shows more crowded points and also high stretches between the initial few points. After all the results of the training were obtained, a small batch from the validation subset was run and tested and comparing both results from the surface would indicate that both models can perform well when the gesture is much more distinct and identifiable through naked eyes alone. However, the same cannot be stated for those gestures which are less distinct and can normally be mistaken for another letter entirely. YOLOv8 gives more accurate and correct results and also provides predictions of certain gestures that weren't given as well. properly predicted by the YOLOv5 model.

Table 2
Initial and Final loss

	BOUNDING BOX LOSS		CLASSIFICATION LOSS		mAP
	1st epoch	80th epoch	1st epoch	80th epoch	
YOLOv5	1.733	0.3265	4.251	0.191	93.6%
YOLOv8	1.444	0.312	4.352	0.1735	96%

Both neural networks were trained for 80 epochs on web IDE “Google collab” with a Tesla T4 at P(8) and CUDA version 12.0. Time taken for training was approximately 6 hours, excluding the remaining 1–2 hours for preparing the dataset and testing. Validation results of both are shown in Fig. 4a and Fig. 4b. Also the Precision Curve of YOLOv5 and YOLOv8 is shown in Fig. 5a and Fig. 5b. Both performances are excellent and do not show any major change in between themselves. The difference between the final confidence level of these is observed to be around 0.01 and therefore no proper census can be made on whether a particular model is better than the other. However, the situation is different during practical testing, while images provided from the dataset [24] for testing both the models show no difference in between themselves, better performance is observed in YOLOv5 for majority of its time when testing an intermediate quality recorded video. Correct and Incorrect prediction is shown using both versions in Fig. 6a and Fig. 6b.

5. Conclusion

While YOLOv8 performs noticeably better during training and validation to some extent, even in testing. However, the model is not ready for any practical applications that involve constant frame movement, which makes YOLOv8 a more conceptual model at its current stage. On the other hand, YOLOv5 is easily able to predict the right gesture and even fills the prediction, which was left blank by the other model.

Declarations

On behalf of all authors, I state that there is no conflict of interest.

Ethical Considerations I declare that all ethical guidelines and principles have been followed during the course of this research.

Contributors: 1,2 wrote the manuscript and 3,4,5 have designed the experiments of the proposed model.

Availability of Data: All the links for the external sources are included in the manuscipt.

Funding: No funding is received.

References

1. World Health Organization. (2023, February 27) Deafness and hearing loss. Retreived from: https://www.who.int/news-room/fact-sheets/detail/deafness-and-hearing-loss.
2. Vedak, Omkar, et al. "Sign language interpreter using image processing and machine learning." International Research Journal of Engineering and Technology (IRJET) (2019).
3. Halvardsson, G., Peterson, J., Soto-Valero, C., & Baudry, B. (2021). Interpretation of swedish sign language using convolutional neural networks and transfer learning. SN Computer Science, 2(3), 207.
4. Daniels, S., Suciati, N., & Fathichah, C. (2021, February). Indonesian sign language recognition using yolo method. In IOP Conference Series: Materials Science and Engineering (Vol. 1077, No. 1, p. 012029). IOP Publishing.
5. Bhattacharya, Abhiruchi, Vidya Zope, Kasturi Kumbhar, Padmaja Borwankar, and Ariscia Mendes. "Classification of sign language gestures using machine learning." Image 8, no. 12 (2019).
6. Starner, J. Weaver and A. Pentland, "Real-time American sign language recognition using desk and wearable computer based video," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 20, no. 12, pp. 1371-1375, Dec. 1998, doi: 10.1109/34.735811.
7. Aziz, M. S. B. Haji Salam, U. U. Sheikh and S. Ayub, "Exploring Deep Learning-Based Architecture, Strategies, Applications and Current Trends in Generic Object Detection: A Comprehensive Review," in IEEE Access, vol. 8, pp. 170461-170495, 2020, doi: 10.1109/ACCESS.2020.3021508.
8. Naglot and M. Kulkarni, "Real time sign language recognition using the leap motion controller," 2016 International Conference on Inventive Computation Technologies (ICICT), 2016, pp. 1-5, doi: 10.1109/INVENTIVE.2016.7830097.
9. C. -H. Chuan, E. Regina and C. Guardino, "American Sign Language Recognition Using Leap Motion Sensor," 2014 13th International Conference on Machine Learning and Applications, 2014, pp. 541-544, doi: 10.1109/ICMLA.2014.110.
10. Imagawa, Shan Lu and S. Igi, "Color-based hands tracking system for sign language recognition," Proceedings Third IEEE International Conference on Automatic Face and Gesture Recognition, 1998, pp. 462-467, doi: 10.1109/AFGR.1998.670991.
11. Wang et al., "Hear Sign Language: A Real-Time End-to-End Sign Language Recognition System," in IEEE Transactions on Mobile Computing, vol. 21, no. 7, pp. 2398-2410, 1 July 2022, doi: 10.1109/TMC.2020.3038303.
12. Fernando and J. Wijayanayaka, "Low cost approach for real time sign language recognition," 2013 IEEE 8th International Conference on Industrial and Information Systems, 2013, pp. 637-642, doi: 10.1109/ICIInfS.2013.6732059.
13. K. Dutta and S. A. S. Bellary, "Machine Learning Techniques for Indian Sign Language Recognition," 2017 International Conference on Current Trends in Computer, Electrical, Electronics and Communication (CTCEEC), 2017, pp. 333-336, doi: 10.1109/CTCEEC.2017.8454988.

14. S. Teja Mangamuri, L. Jain and A. Sharmay, "Two Hand Indian Sign Language dataset for benchmarking classification models of Machine Learning," 2019 International Conference on Issues and Challenges in Intelligent Computing Techniques (ICICT), 2019, pp. 1-5, doi: 10.1109/ICICT46931.2019.8977713.
15. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You Only Look Once: Unified, Real-Time Object Detection”, in proceedings of the IEEE conference on computer vision and pattern recognition, pp. 779–788, 2016
16. Du, Juan. "Understanding object detection based on CNN family and YOLO." In Journal of Physics: Conference Series, vol. 1004, p. 012029. IOP Publishing, 2018
17. Redmon, J., & Farhadi, A. (2017). YOLO9000: better, faster, stronger. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 7263-7271)
18. Thuan, D. (2021). Evolution of Yolo algorithm and Yolov5: The State-of-the-Art coloobject detection algorithm.
19. D. T. Yung, W. K. Wong, F. H. Juwono and Z. A. Sim, "Safety Helmet Detection Using Deep Learning: Implementation and Comparative Study Using YOLOv5, YOLOv6, and YOLOv7," 2022 International Conference on Green Energy, Computing and Sustainable Technology (GECOST), Miri Sarawak, Malaysia, 2022, pp. 164-170, doi: 10.1109/GECOST55694.2022.10010490.
20. Huang, J. Pedoeem and C. Chen, "YOLO-LITE: A Real-Time Object Detection Algorithm Optimized for Non-GPU Computers," 2018 IEEE International Conference on Big Data (Big Data), Seattle, WA, USA, 2018, pp. 2503-2510, doi: 10.1109/BigData.2018.8621865
21. Diwan, T., Anirudh, G., & Tembhurne, J. V. (2022). Object detection using YOLO: Challenges, architectural successors, datasets and applications. Multimedia Tools and Applications, 1-33.
22. Thuan, D. (2021). Evolution of Yolo algorithm and Yolov5: The State-of-the-Art object detection algorithm.
23. lee, 2020, American Sign Language Letters Dataset v1, https://public.roboflow.com/object-detection/american-sign-language-letters
24. Noman, V. Stankovic and A. Tawfik, "Object Detection Techniques: Overview and Performance Comparison," 2019 IEEE International Symposium on Signal Processing and Information Technology (ISSPIT), 2019, pp. 1-5, doi: 10.1109/ISSPIT47144.2019.9001879.
25. Rastogi, A., & Ryuh, B. S. (2019). Teat detection algorithm: YOLO vs. Haar-cascade. Journal of Mechanical Science and Technology, 33, 1869-1874.
26. Wu, S., Li, Z., Li, S., Liu, Q., & Wu, W. (2023). Static Gesture Recognition Algorithm Based on Improved YOLOv5s. Electronics, 12(3), 596.
27. Karaman, A., Pacal, I., Basturk, A., Akay, B., Nalbantoglu, U., Coskun, S., ... & Karaboga, D. (2023). Robust real-time polyp detection system design based on YOLO algorithms by optimizing activation functions and hyper-parameters with artificial bee colony (ABC). Expert Systems with Applications, 119741.

28. Mark Everingham, Luc Gool, Christopher K. Williams, John Winn, and Andrew Zisserman. 2010. The Pascal Visual Object Classes (VOC) Challenge. Int. J. Comput. Vision 88, 2 (June 2010), 303–338.https://doi.org/10.1007/s11263-009-0275-4.
29. Lin, Tsung-Yi, et al. "Microsoft coco: Common objects in context." Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13. Springer International Publishing, 2014.
30. Wang CY, Liao HY, Wu YH, Chen PY, Hsieh JW, Yeh IH. CSPNet: A new backbone that can enhance learning capability. InProceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops 2020 (pp. 390-391).
31. Liu S, Qi L, Qin H, Shi J, Jia J. Path aggregation network for instance segmentation. InProceedings of the IEEE conference on computer vision and pattern recognition 2018 (pp. 8759-8768).
32. Ren, Shaoqing, et al. "Faster r-cnn: Towards real-time object detection with region proposal networks." Advances in neural information processing systems 28 (2015).

Figures

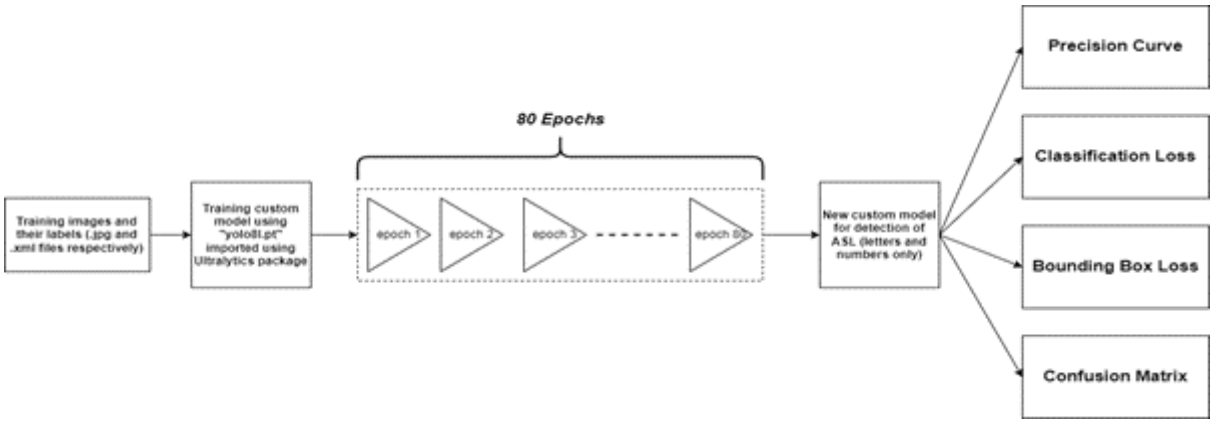

Figure 1

Methodology

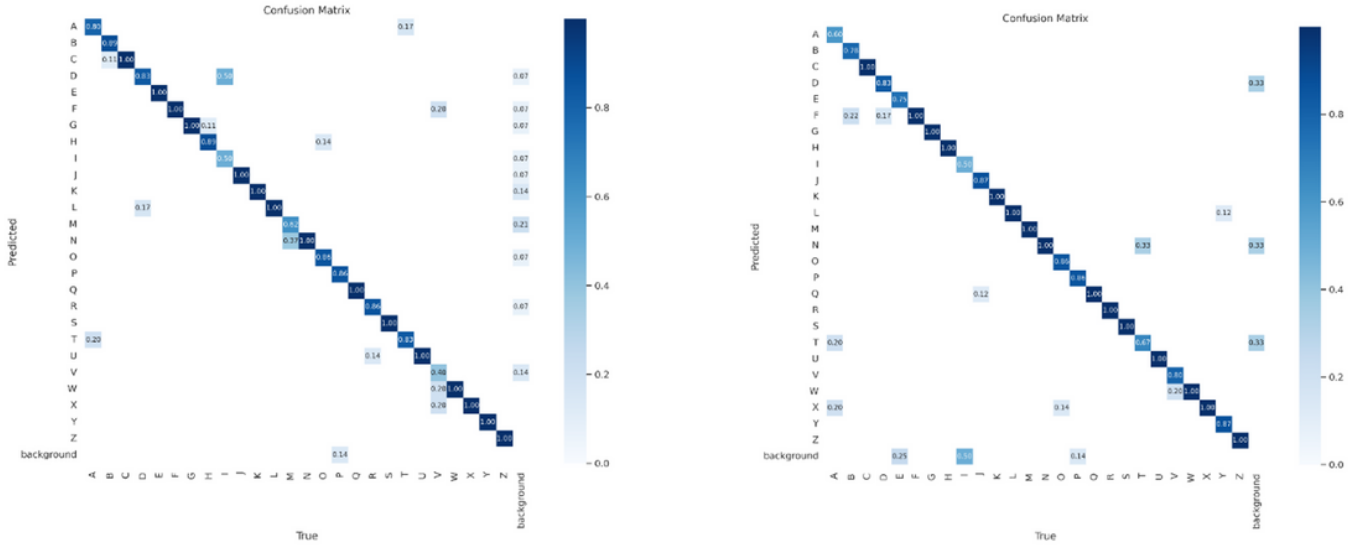

Figure 2

Confusion Matrix for YOLOv5 (above) and YOLOv8(below)

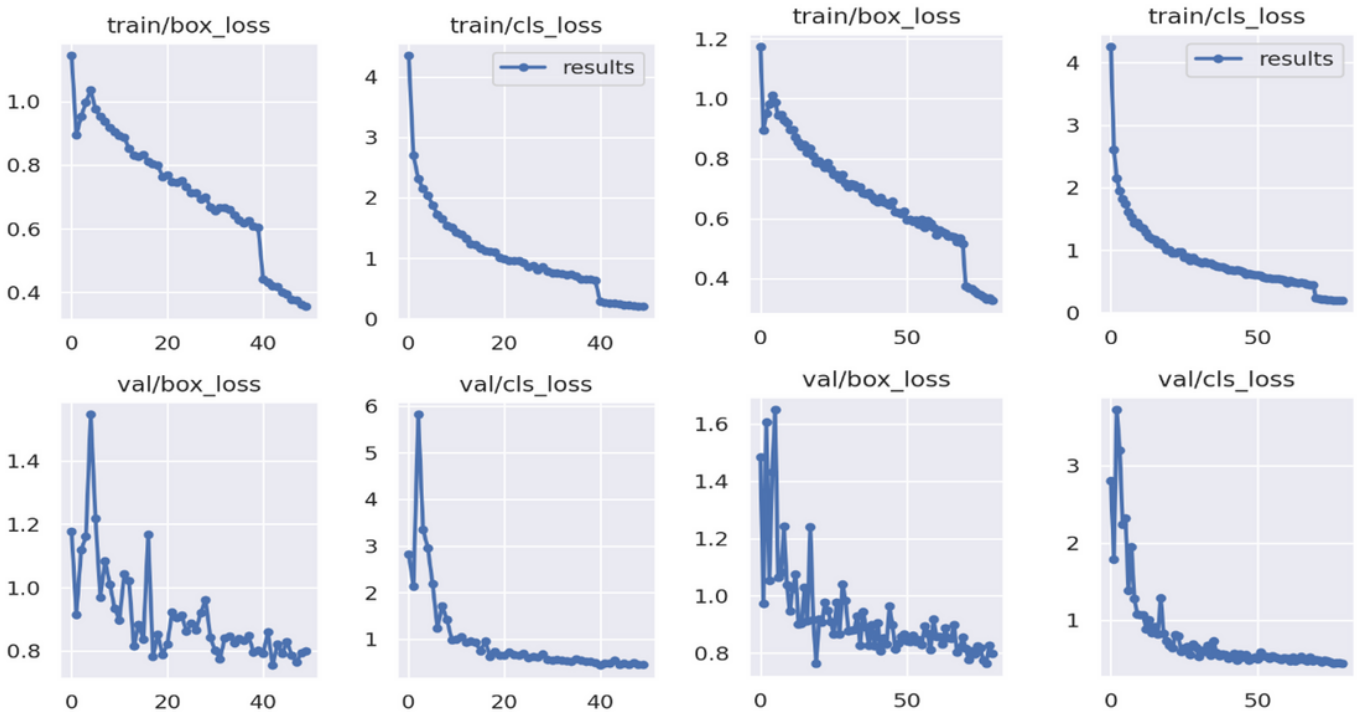

Figure 3

Loss reduction for YOLOv5 (left) and YOLOv8 (right)

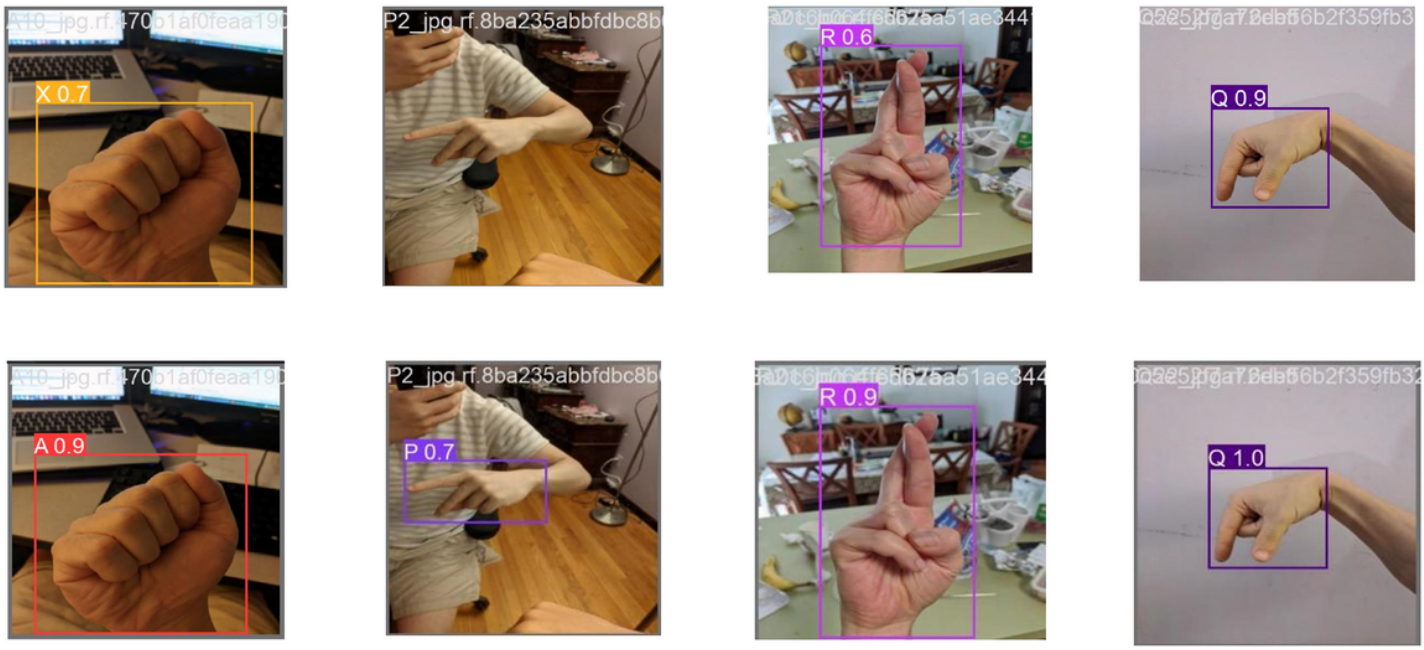

Figure 4

a: Validation result (YOLOv5)

b: Validation result (YOLOv8)

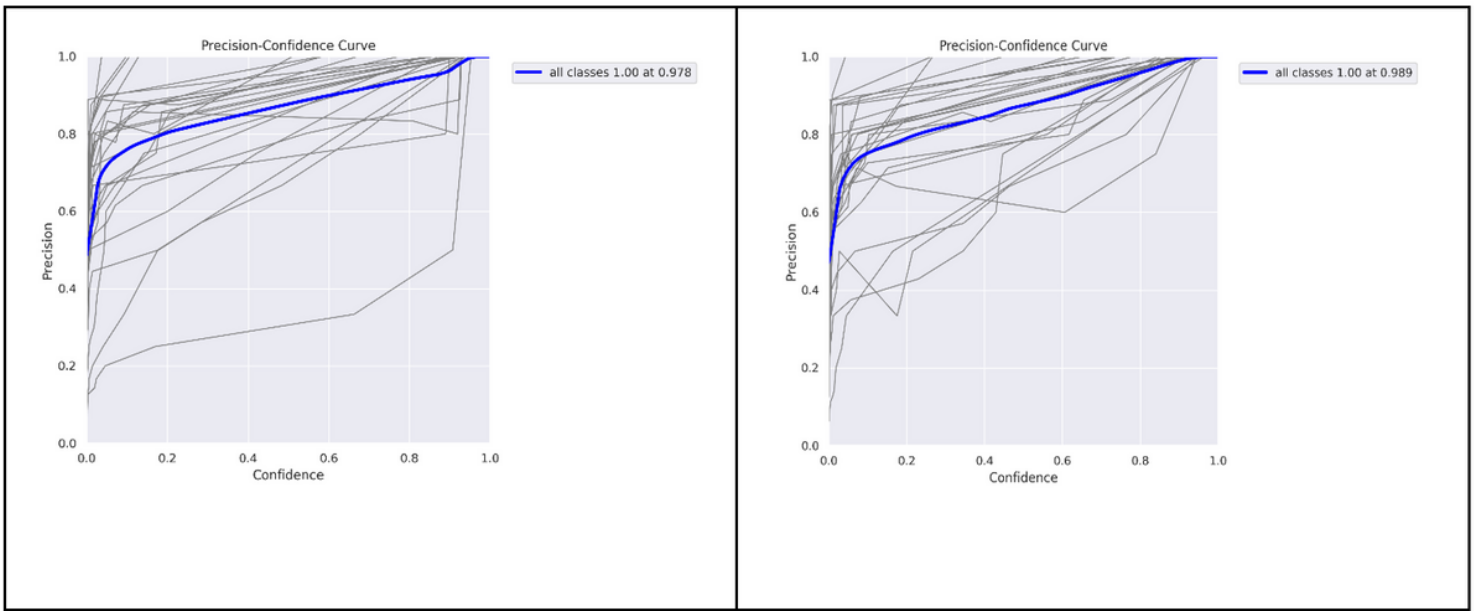

Figure 5

a: Precision curve (YOLOv5)

b: Precision Curve (YOLOv8)

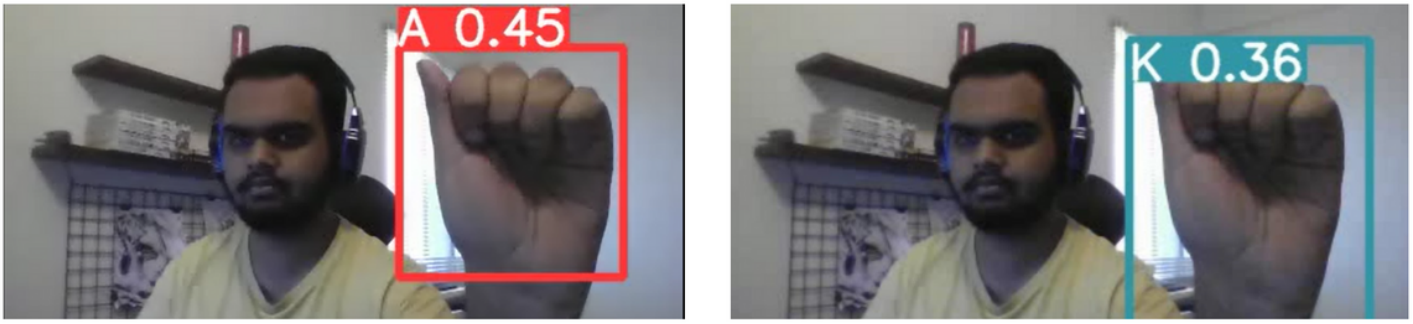

Figure 6

a: Correct prediction (YOLOv5)

b: Incorrect prediction (YOLOv8)