

Research Article :

**Arabic Sign Language Recognition and Generating Arabic Speech Using Convolutional Neural Network and YOLO**

**Youssef hassan 1 , Fares samy 2 , Wijdan Mostafa 3 , Nada Emad 4 , Dr. Eman Ebrahim 5**

Department of Computer Science, Benha University , Egypt

**Abstract:**

Sign language is used by 70 million people around the world and there’s a lack in communication between deaf and dump community and our community. Sign language encompasses the movement of the arms and hands as a means of communication for people with hearing disabilities. An automated sign recognition system requires two main courses of action: the detection of particular features and the categorization of particular input data. A vision-based system by applying CNN for the recognition of Arabic hand sign-based letters and translating them into Arabic speech is proposed in this paper. The proposed system will automatically detect hand sign letters and speaks out the result with the Arabic language with a deep learning model. This system gives 97% accuracy to recognize the Arabic hand sign-based letters which assures it as a highly dependable system. After recognizing the Arabic hand sign-based letters, the outcome will be fed to the text into the speech engine which produces the audio of the Arabic language as an output.

**Introduction:**

Sign language is made up of four major manual components that comprise of hands’ figure configuration, hands’ movement, hands’ orientation, and hands’ location in relation to the body .There are mainly two procedures that an automated sign-recognition system has, vis-a-vis detecting the features and classifying input data.

Many approaches have been put forward for the classification and detection of sign languages for the improvement of the performance of the automated sign language system. while little attention is paid on the Arabic language. This may be because of the non availability of a generally accepted database for the Arabic sign language to researchers. So, researchers had to resort to develop datasets themselves which is a tedious task. Specially, there is no Arabic sign language reorganization system that uses comparatively new techniques such as Cognitive Computing, Convolutional Neural Network (CNN), IoT, and Cyberphysical system that are extensively used in many automated systems .

The cognitive process enables systems to think the same way a human brain thinks without any human operational assistance. The human brain inspires the cognitive ability On the other hand, deep learning is a subset of machine learning in artificial intelligence (AI) that has networks capable of learning unsupervised from data that is unstructured or unlabeled which is also known as deep neural learning or deep neural network [1–2]. In deep learning, CNN is a class of deep neural networks, most commonly applied in the field of computer vision. The vision-based approaches mainly focus on the captured image of gesture and get the primary feature to identify it. This method has been applied in many tasks including super resolution, image classification and semantic segmentation, multimedia systems, and emotion recognition [3–4]. One of the few well-known researchers who have applied CNN is K. Oyedotun and Khashman [5] who used CNN along with Stacked Denois ing Autoencoder (SDAE) for recognizing 24 hand gestures of the American Sign Language (ASL) gotten through a public database. On the other hand, the proposal to use Convolu tional Neural Network (CNN) for recognizing the Italian sign language was made by Pigou et al. [6]. Whereas Hu et al. had made a proposal for the architecture of hybrid CNN and RNN to capture the temporal properties perfectly for the electromyogram signal which solves the problem of gesture recognition [7]. An incredible CNN model that automatically recognizes the digits based on hand signs and speaks the particular result in Bangla language is explained in [8], which is followed in this work. In [9] as well, there is a proposal of using transfer learning on data collected from several users, while exploiting the use of deep-learning algorithm to learn discriminant characteristics found from large datasets.

YOLOv8 is the newest state-of-the-art YOLO model that can be used for object detection, image classification, and instance segmentation tasks. YOLOv8 was developed by Ultralytics, who also created the influential and industry-defining YOLOv5 model. YOLOv8 includes numerous architectural and developer experience changes and improvements over YOLOv5.YOLOv8 is under active development as of writing this post, as Ultralytics work on new features and respond to feedback from the community.

**Data Preprocessing :**

Data preprocessing is the first step toward building a working deep learning model. It is used to transform the raw data in a useful and efficient format. Figure 1 shows the flow diagram of data preprocessing.

**1.First Dataset For CNN Model**

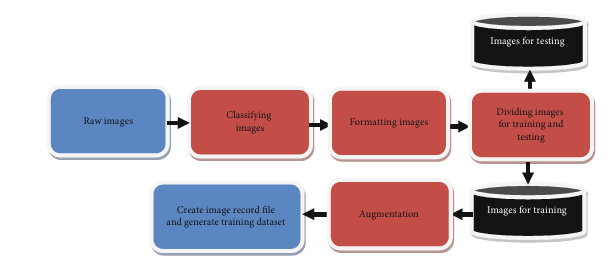
Raw Images. Hand sign images are called raw images that are captured using a camera for implementing the proposed system. The images are taken in the following environment:

* 1. From different angles
  2. By changing lighting conditions
  3. With good quality and in focus
  4. By changing object size and distance

The objective of creating raw images is to create the data set for training and testing.

Classifying Images. The proposed system classifies the images into 32 categories for 32 letters of the Arabic Alphabet. One subfolder is used for storing images of one category to implement the system. All subfolders which represent classes are kept together in one main folder named “dataset” in the proposed system.

Formatting Image. Usually, the hand sign images are unequal and having different background. So, it is required to delete the unnecessary element from the images for getting the hand part. The extracted images are resized to 128 × 128 pixels and converted to RGB.



**Figure 1:** Flow diagram Of data preprocessing.

**2. Second Dataset For YOLO Model**

The contribution is a large fully-labelled dataset for Arabic Sign Language (ArSL) which is made publically available and free for all researchers. The dataset which is named RGB Arabic Alphabet Sign Language (ArSL) dataset Image Dataset consists of 21868 images for the 31 Arabic sign language sign and alphabets collected from 40 participants in different age groups. Different dimensions and different variations were present in images which can be cleared using pre-processing techniques to remove noise

Preprocessing steps :

Auto-Orient: Applied

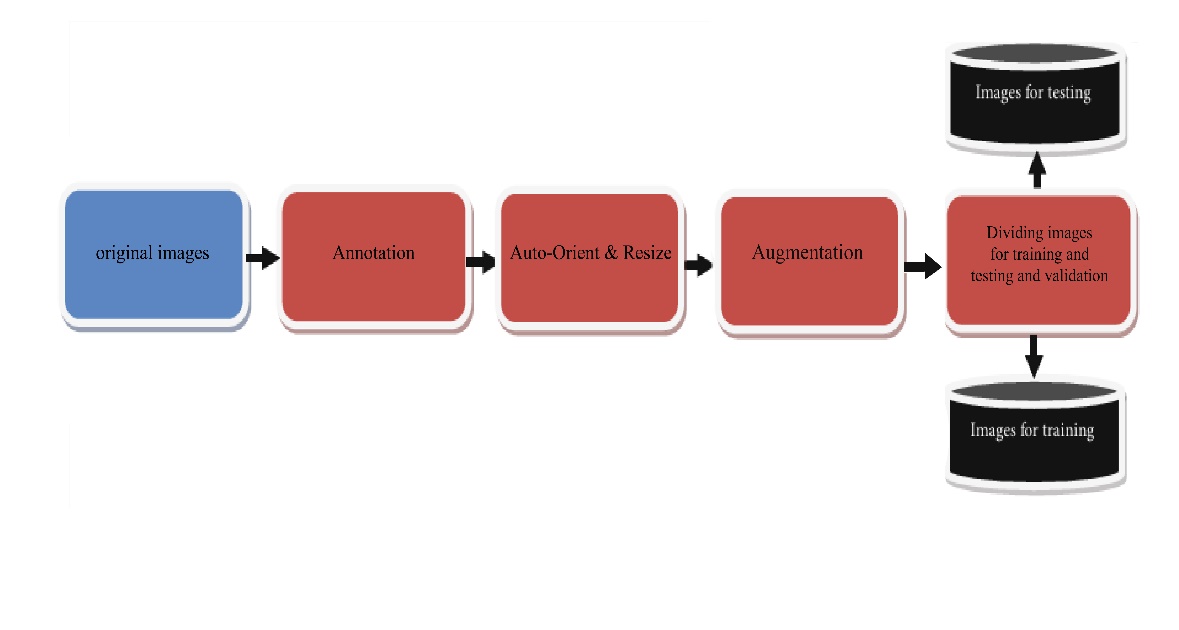
Resize: Stretch to 640x640

Augmentations :

Rotation: Between -4° and +4°

Cutout: 3 boxes with 10% size each

Grayscale: Applied **.**



**Figure 2:** Flow diagram Of data preprocessing for second data.

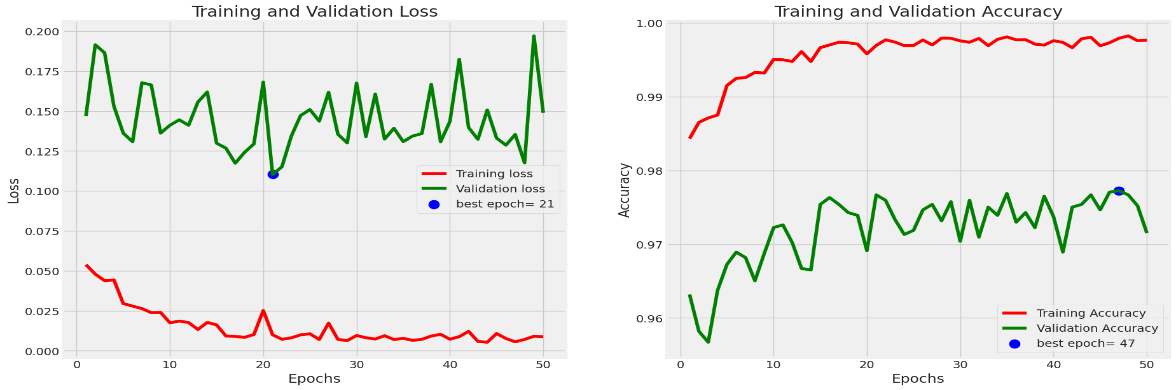
**5. Classes**

| **#** | **Letter name in English Script** | **Letter name in Arabic script** | **# of Images** | **#** | **Letter name in English Script** | **Letter name in Arabic script** | **# of images** |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | Alef | أَلِف)أ) | 2800 | 17 | Thal | ظَاء)ظ) | 136 |
| 2 | Beh | بَاء) ب) | 343 | 18 | Ayn | عَين)ع) | 251 |
| 3 | The | أتَاء) ت) | 397 | 19 | Ghain | غَين)غ) | 627 |
| 4 | Thā | ثَاء) ث) | 2168 | 20 | Feh | فَاء)ف) | 975 |
| 5 | Jeem | جِيمْ) ج) | 217 | 21 | Qāf | قَاف) ق) | 426 |
| 6 | Hā | حَاء) ح) | 398 | 22 | Kāf | كَاف)ك) | 186 |
| 7 | Khah | خَاء) خ) | 96 | 23 | Lām | لاَمْ)ل) | 513 |
| 8 | Dal | دَالْ) د) | 476 | 24 | Meem | مِيمْ)م) | 3060 |
| 9 | Thāl | ذَال) ذ) | 881 | 25 | Noon | نُون)ن) | 164 |
| 10 | Reh | رَاء) ر) | 266 | 26 | Hah | هَاء)ه) | 210 |
| 11 | Zain | زَاي) ز) | 478 | 27 | Wāw | وَاو)و) | 155 |
| 12 | Seen | سِينْ) س) | 464 | 28 | Teh\_Marbuta | ة)ة) | 156 |
| 13 | Sheen | شِينْ) ش) | 296 | 29 | Al | ال)ال) | 2077 |
| 14 | Sād | صَادْ)ص) | 698 | 30 | Laa | ﻻ)ﻻ) | 1598 |
| 15 | Dād | ضَاد)ض) | 88 | 31 | Yeh | يَاء) يَاء) | 160 |
| 16 | Tā | طَاء)ط) | 1105 |  |  |  |  |

**Results:**

1. **CNN**

Results from the Convolutional Neural Network (CNN) model for the Sign Language Recognition project were highly encouraging, achieving an accuracy of 97% on the test dataset. The model was trained on a dataset of 54,049 images with 32 classes representing different sign language gestures. To enhance the model's performance, data augmentation techniques were applied during training



**Figure 3** : Training and validation Loss and Accuracy

**Testing Result :**

Accuracy: 0.9715078630897317

Precision: 0.9715078630897317

F1 Score: 0.9715078630897317

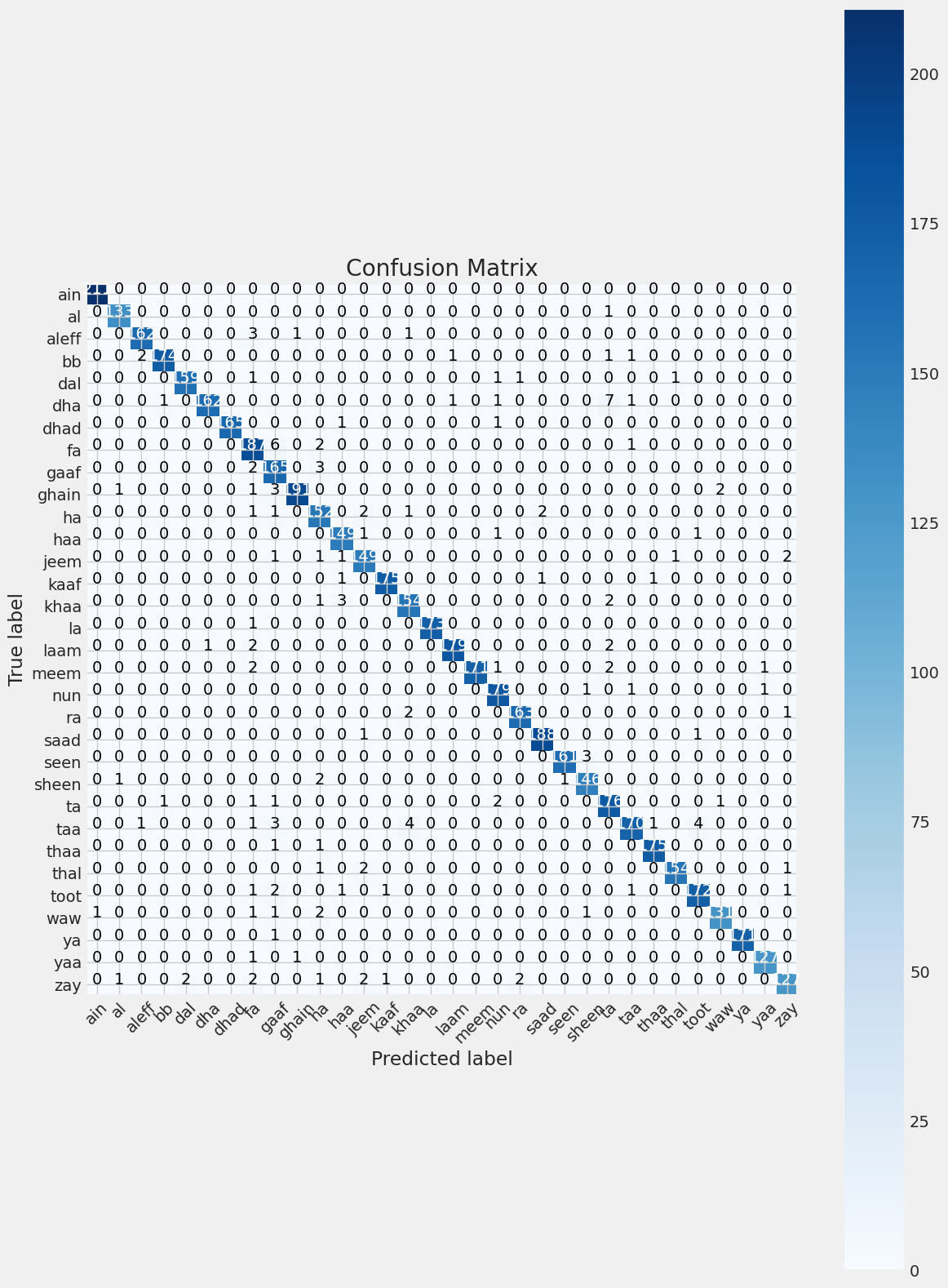
Sensitivity: 0.9715078630897317

Specificity: 1.0

|  |  |  |
| --- | --- | --- |
| **Figure 4** : Model Accuracy | **Figure 5** : Training & Validation precision | **Figure 6 :**  Plot training & validation recall |

**Confusion Matrix :**

The confusion matrix revealed that the model performed well across all 32 classes, with accuracy ranging from 95% to 97.15% for individual classes. The precision and recall scores were calculated for each class, demonstrating an average precision of 0.978 and an average recall of 0.97.



**Figure 7 :** confusion matrix

1. **YOLO**

|  |  |  |  |
| --- | --- | --- | --- |
| **Class** | **P** | **R** | **mAP50** |
| Al | 0.658 | 0.967 | 0.917 |
| Alef | 0.848 | 1 | 0.991 |
| Beh | 0.972 | 1 | 0.995 |
| Dad | 0.977 | 1 | 0.995 |
| Dal | 0.973 | 1 | 0.995 |
| Feh | 0.933 | 1 | 0.995 |
| Ghain | 0.668 | 1 | 0.886 |
| Hah | 0.977 | 1 | 0.995 |
| Heh | 1 | 0.819 | 0.995 |
| Jeem | 1 | 0.824 | 0.995 |
| Kaf | 0.852 | 1 | 0.995 |
| Khah | 0.931 | 1 | 0.995 |
| Laa | 1 | 0.714 | 0.995 |
| Lam | 0.939 | 1 | 0.995 |
| Meem | 0.819 | 0.84 | 0.896 |
| Noon | 1 | 0.792 | 0.995 |
| Qaf | 0.791 | 1 | 0.995 |
| Reh | 0.976 | 1 | 0.995 |
| sad | 0.976 | 1 | 0.995 |
| seen | 0.788 | 1 | 0.995 |
| sheen | 0.768 | 1 | 0.995 |
| tah | 0.653 | 1 | 0.995 |
| teh | 1 | 1 | 0.995 |
| Teh\_Marbuta | 1 | 0 | 0.995 |
| Thal | 0.62 | 0 | 0.995 |
| Theh | 0.8 | 0 | 0.995 |
| Waw | 1 | 0 | 0.995 |
| Yah | 0.78 | 0 | 0.995 |
| Zah | 0.9 | 0 | 0.995 |
| Zain | 0.933 | 1 | 0.995 |
| Ain | 0.896 | 0.915 | 0.983 |

**Figure 8:** YOLO Model Train Results

**Column Definitions:**

Class: This is the category or type of object the model was trying to detect or classify.

Instances: Number of instances or examples of each class present in the dataset or during the evaluation.

P (Precision): Precision indicates the accuracy of the positive predictions. It is the ratio of correctly predicted positive observations to the total predicted positives. High precision relates to a low rate of false positives.

R (Recall): Recall (also known as sensitivity) is the ratio of correctly predicted positive observations to all actual positives. High recall relates to a low rate of false negatives.

mAP50 (mean Average Precision at IoU = 0.50): This metric evaluates the mean average precision of the model at an Intersection over Union (IoU) threshold of 0.50. It is a common metric in object detection that measures the model's accuracy in terms of detecting objects with a bounding box that overlaps at least 50% with the ground-truth bounding box.

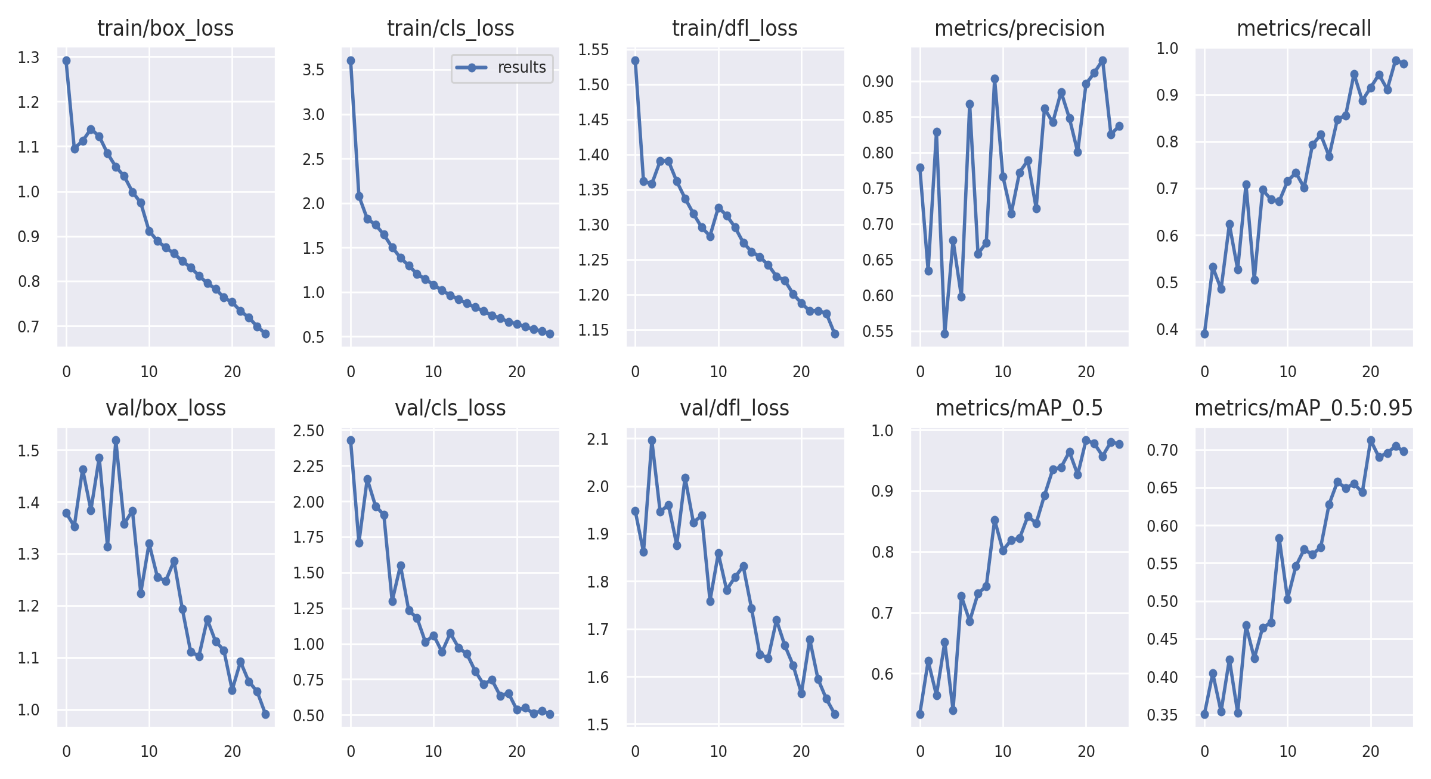
**Analysis of Results:**

Ain, Beh, Dad, Dal, Feh, Hah: These classes show very high precision and recall, indicating that the model performs excellently in detecting these classes with very few false positives or false negatives. Their mAP50 values are close to 1, showing almost perfect accuracy in bounding box predictions.

**Variable Precision and Recall:** Al: Despite a high recall of 0.967, its precision is only 0.658, indicating some false positives are being predicted as 'Al'.

Ghain: Shows lower precision (0.668) but perfect recall (1), suggesting the model captures all 'Ghain' instances but with some false positives.

Teh\_Marbuta, Thal, Theh, Waw: These classes have precision, recall, or both set to 0 despite having a high mAP50. This suggests possible issues in data labeling or errors in the calculation/reporting. Specifically, an mAP50 of 0.995 with a recall of 0 at the same time is inconsistent under normal circumstances.

**Metrics: **

**Figure 9 :** Yolo Model Metrics

train/box\_loss and val/box\_loss: These plots show the loss related to bounding box predictions during training and validation phases, respectively. The loss is decreasing over time, which indicates that the model is getting better at predicting the correct location of the bounding boxes around the objects.

train/cls\_loss and val/cls\_loss: These represent the classification loss during training and validation. A decrease in classification loss suggests that the model is becoming more accurate in assigning the correct class labels to the objects within the bounding boxes.

train/dfl\_loss and val/dfl\_loss: DFL could stand for a specific type of loss, potentially related to a distribution-focal loss which might be part of the model's architecture. Again, a decreasing trend indicates improvement in whichever aspect this loss is measuring.

metrics/precision: This is a measure of the number of correct positive predictions divided by the total number of positive predictions. The fluctuation is normal, but the overall trend is upward, suggesting an improvement in precision over time.

metrics/recall: Recall measures the number of correct positive predictions divided by the total

number of actual positives. The steady increase in this plot suggests that the model is getting better at detecting all the relevant objects.

metrics/mAP\_0.5: mAP stands for mean Average Precision, and the '0.5' refers to the Intersection over Union (IoU) threshold. This metric is commonly used in object detection to evaluate model performance. The trend here is positive, showing that the model is improving in terms of precision at a 0.5 IoU threshold.

metrics/mAP\_0.5:0.95: This is the mean Average Precision calculated over different IoU thresholds from 0.5 to 0.95. This is a more stringent metric, as it requires the model to be accurate at various levels of overlap between the predicted and ground truth bounding boxes. The upward trend here is a very good sign, indicating that the model is performing better over time across a range of IoU thresholds.

**Work elements , techniques and Tools:**

**YOLO:**is the newest state-of-the-art YOLO model that can be used for object detection, image classification, and instance segmentation tasks.

**TensorFlow:** is a free and open-source software library for machine learning and artificial intelligence. It can be used across a range of tasks but has a particular focus on training and inference of deep neural networks.

**PyTorch:** Is a machine learning library based on the Torch library used for applications such as computer vision and natural language processing originally developed by Meta AI and now part of the Linux Foundation umbrella It is recognized as one of the two most popular machine learning libraries alongside TensorFlow, offering free and open-source software released under the modified BSD license. Although the Python interface is more polished and the primary focus of development, PyTorch also has a C++ interface

**References:**

[1] X. Chen, L. Zhang, T. Liu, and M. M. Kamruzzaman, “Research on deep learning in the field of mechanical equip mentfault diagnosis image quality,” Journal of Visual Communication and Image Representation, vol. 62, pp. 402–409, 2019.

[2] G. Chen, L. Wang, and M. M. Kamruzzaman, “Spectral classification of ecological spatial polarization SAR image based on target decomposition algorithm and machine learning,” Neural Computing and Applications, vol. 32, no. 10, pp. 5449 5460, 2020.

[3] B. Kayalibay, G. Jensen, and P. van der Smagt, “CNN-based segmentation of medical imaging data,” 2017, http://arxiv .org/abs/1701.03056.

[4] M. S. Hossain, G. Muhammad, W. Abdul, B. Song, and B. B. Gupta, “Cloud-assisted secure video transmission and sharing framework for smart cities,” Future Generation Computer Sys tems, vol. 83, pp. 596–606, 2018

[5] O. K. Oyedotun and A. Khashman, “Deep learning in vision based static hand gesture recognition,” Neural Computing and Applications, vol. 28, no. 12, pp. 3941–3951, 2017.

[6] L. Pigou, S. Dieleman, P.-J. Kindermans, and B. Schrauwen, “Sign language recognition using convolutional neural net works,” in European Conference on Computer Vision, pp. 572 578, 2015.

[7] Y. Hu, Y. Wong, W. Wei, Y. Du, M. Kankanhalli, and W. Geng, “A novel attention-based hybrid CNN-RNN archi tecture for sEMG-based gesture recognition,” PLoS One, vol. 13, no. 10, article e0206049, 2018.

[8] S. Ahmed, M. Islam, J. Hassan et al., “Hand sign to Bangla speech: a deep learning in vision based system for recognizing hand sign digits and generating Bangla speech,” 2019, http:// arxiv.org/abs/1901.05613.

[9] U. Cote-Allard, C. L. Fall, A. Drouin et al., “Deep learning for electro myographic hand gesture signal classification using transfer learning,” IEEE Transactions on Neural Systems and Rehabilitation Engineering, vol. 27, no. 4, pp. 760–771, 2019.

[10] A fully-labelled dataset of Arabic Sign Language (ArSL) images is developed for research related to sign language recognition launched in Prince Mohammad Bin Fahd University, Al Khobar, Saudi Arabia The dataset is made available publicly at https:// data.mendeley.com/datasets/y7pckrw6z2/1.

[11] RGB Arabic Alphabet Sign Language (AASL) dataset Image Dataset , RGB dataset contain 21868 label images [RGB Arabic Alphabet Sign Language (AASL) dataset - v5 2023-03-13 10:56am (roboflow.com)](https://universe.roboflow.com/rgb-arsl/rgb-arabic-alphabet-sign-language-aasl-dataset/dataset/5) s