SE/EE 444

لغتي إشارتي

Bayan Al Shikh Zien/ Layali Elkordy/Reema Alotaibi

Introduction

In today’s society, deaf and mute citizens usually face difficulties communicating with others via sign language, as our community lacks this knowledge. Although 360 million of the world’s population are deaf/mute, which is more than 5% of the global population. Sign language is not a universal language, and each language has its own gestures for the letters. As technology is evolving, different solutions have been designed to help conquer this problem in many languages, yet there are still no similar technologies for Arabic language.

The proposed solution is creating a model that can recognize hand gestures which represent different Arabic letters through a camera lens. The model will connect the captured letters to form a sentence or a word.

The main purpose of “لغتي إشارَتي” is to help people who lack the knowledge of sign language to understand and be able to communicate with deaf and mute people.

Background and Related Work

- ***Object Detection***

The main technique used to execute the project is Object detection. Our model was trained to specifically recognize and localize the object within an image using a contour and then classifying the gesture to its respective letter using image classification. Object detection is a computer vision technique that works to identify and locate objects within an image or video. Specifically, object detection draws bounding boxes around these detected objects, which allow us to locate where said objects are in a given scene. When humans look at images or video, we can recognize and locate objects of interest within a matter of moments. The goal of object detection is to replicate this intelligence using a computer.

Image Classification

Image Classification is a fundamental task that attempts to comprehend an entire image as a whole. The goal is to classify the image by assigning it to a specific label. Typically, Image Classification refers to images in which only one object appears and is analyzed.

Moreover, Image classification is straight forward, but the differences between object localization and object detection can be similar, especially when all three tasks may be just as equally referred to as object recognition.

Image classification involves assigning a class label to an image, whereas object localization involves drawing a contour box around one or more objects in an image. Object detection is more challenging and combines these two tasks and draws a bounding box around each object of interest in the image and assigns them a class label. Together, all these problems are referred to as object recognition.

Object Detection Algorithm

There are many object detection algorithms that help to detect and determine what the gesture is that each algorithm targets. One of the many object detection algorithms providing better accuracy with fast and responsive results is the You Only Look Once (YOLO) algorithm which is used to evaluate the structure and mechanism deduction of hand gesture recognition.

YOLO is a convolutional neural network algorithm, which is highly efficient and works tremendously well for real-time object detection. A neural network not only helps in feature extraction, but it can also help us understand the meaning of gesture and help to detect an object of interest. As the name suggests, the algorithm requires only a single forward propagation through a neural network to detect objects. This means that prediction in the entire image is done in a single algorithm run. The CNN is used to predict various class probabilities and bounding boxes simultaneously.

The YOLO algorithm consists of various variants. Some of the common ones include tiny YOLO and YOLO v3. YOLO algorithm works using the following three techniques:

- Residual blocks
- Bounding box regression
- Intersection Over Union (IOU)

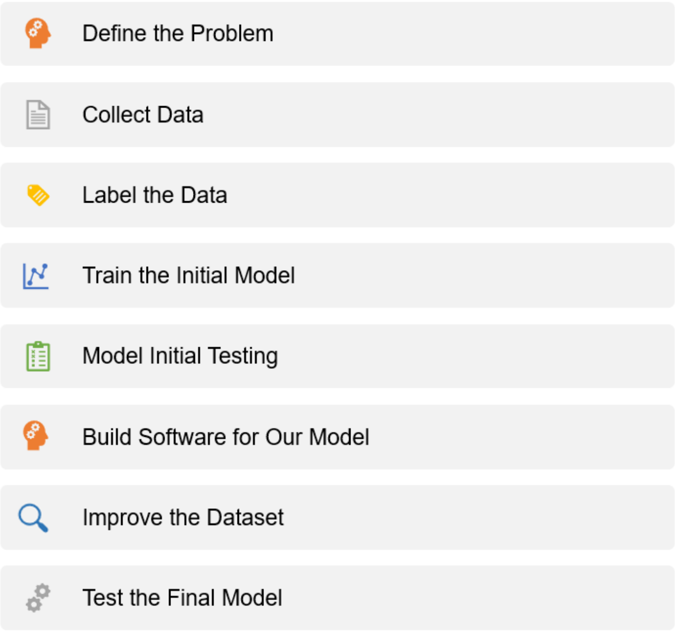
In Residual blocks, the image is divided into various grids. Each grid has a dimension of S x S. In the image, there are many grid cells of equal dimension. Every grid cell will detect objects that appear within them. For example, if an object center appears within a certain grid cell, then this cell will be responsible for detecting it.

A bounding box is an outline that highlights an object in an image. YOLO uses a single bounding box regression to predict the height, width, center, and class of objects. In the image above, represents the probability of an object appearing in the bounding box.

Intersection over union (IOU) is a phenomenon in object detection that describes how boxes overlap. YOLO uses IOU to provide an output box that surrounds the objects perfectly. Each grid cell is responsible for predicting the bounding boxes and their confidence scores. The IOU is equal to 1 if the predicted bounding box is the same as the real box. This mechanism eliminates bounding boxes that are not equal to the real box.

Methods

Throughout our journey in developing (لغتي إشارتي) project, we followed the following approach :



- Define the Problem
- Collect Data
- Label the Data
- Train the Initial Model
- Model Initial Testing
- Build Software for Our Model
- Improve the Dataset
- Test the Final Model

• *Define the problem:*

It's hard to communicate with deaf mute people without previous experience in Arabic sign language, so we need to find a solution that help to recognize and understand the gestures used to represent the Arabic letters.

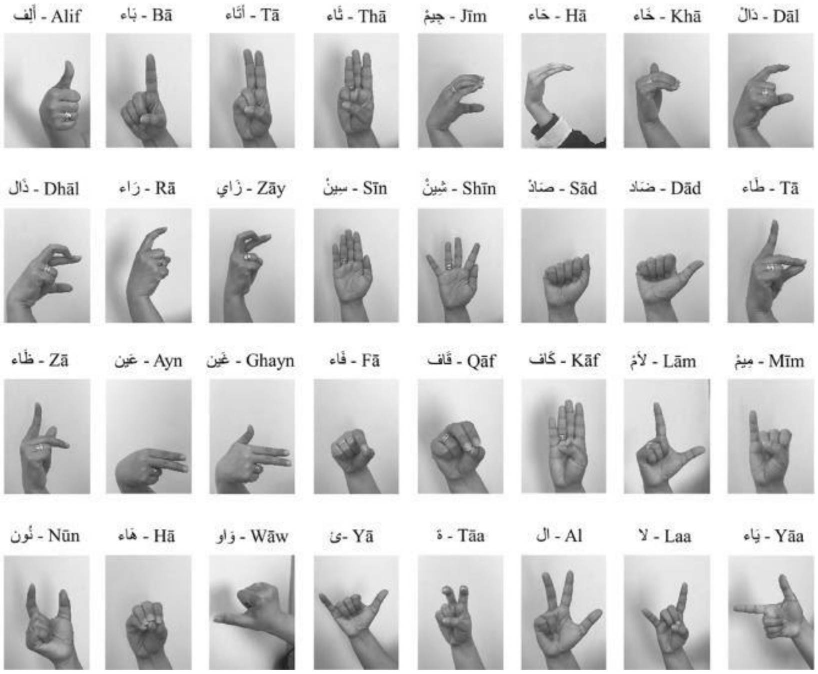
• *Collect Dataset*

Our dataset consisted of 20558 images of Arabic Sign letters. These images were further classified into 32 different sets. Each set held an average of 600 images. The images from the dataset are basically slices from a video captured of the hand gestures from different directions, while considering the lighting effect differences.

Our dataset considered multiple attributes such as different skin tones, background colors, different poses directions, hand accessories, and long-sleeved clothing.

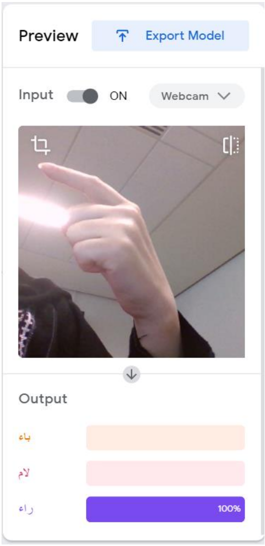
- ## *Label the Data*

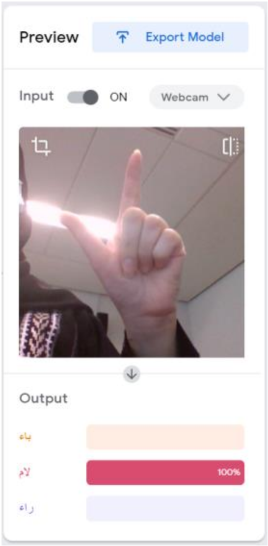
Each image set for each letter is contained under one folder. Every folder is labeled with the respective letter.

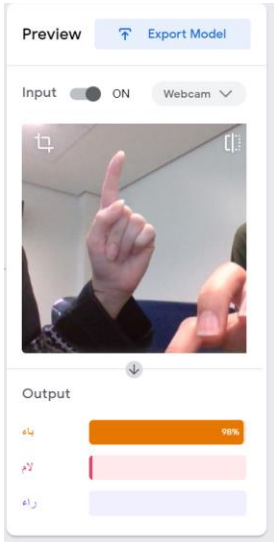


- ## *Train the Initial Model*

Google teachable machine was used to train the dataset using some of the letters as classes in order to prove the concept. In this module only three letters were used.







Build a Software for Our Model

We planned to build a software that uses our model to recognize the different sign gestures and connect the respective letters to form words or complete sentence. Our program mainly uses two libraries, which are TensorFlow and OpenCV

- **TensorFlow** is mainly an open-source library that help us to train our ML model via a python code.
- **OpenCV** is the library responsible for real-time computer vision. It helps us with real time sign language recognition.

Iteration 1: Implement Image Classification Concept

We started developing a program that implemented the concept of image classification. Our model detected a minor number from the 32 classes. In this iteration, the model classified most of the detected gestures incorrectly.

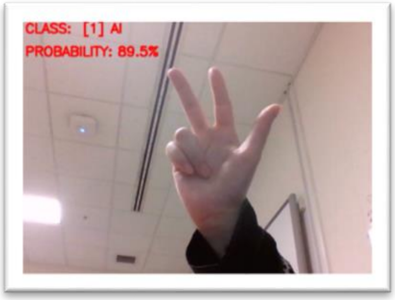


Figure 5

Correct

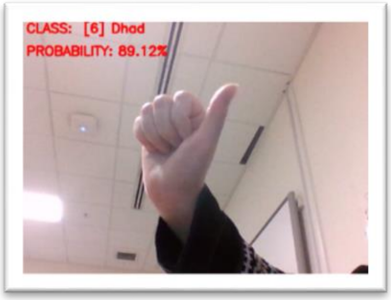


Figure 6

Correct

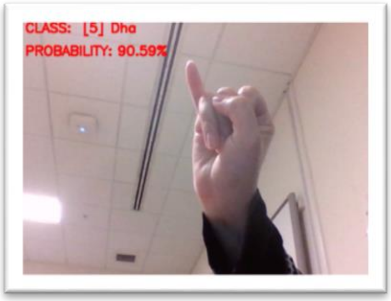


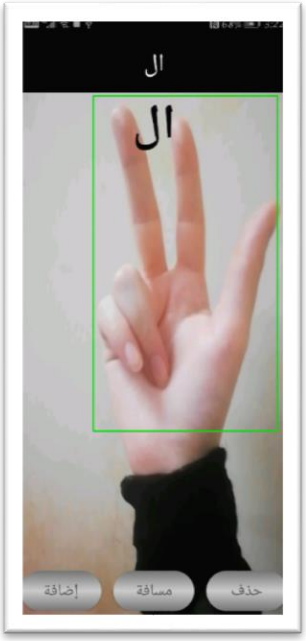
Figure 7

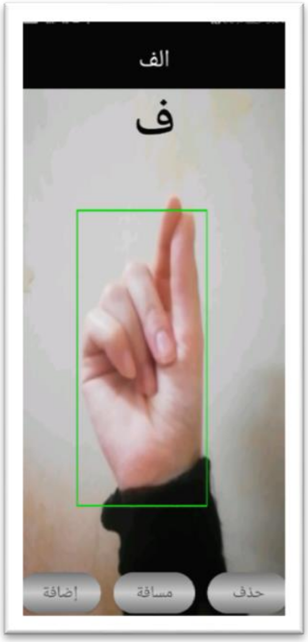
Incorrect

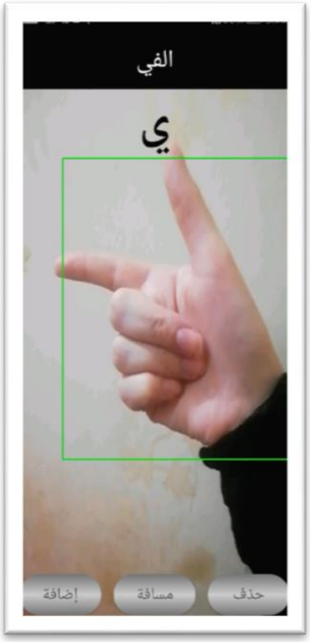
Correct Letter: م

Iteration 2: Implement Object Detection Concept

We developed an Android mobile application that implemented the concept of object detection. Our aim was to detect the hand position first, then classify the image within the contour to its assigned label. This technique improved the detection process as it separates the hand from the background noise. In this model, a number of 10 letters were used with a batch size of 32 and epochs of 40. Our model significantly improved, and most of the letters could be detected by the app.







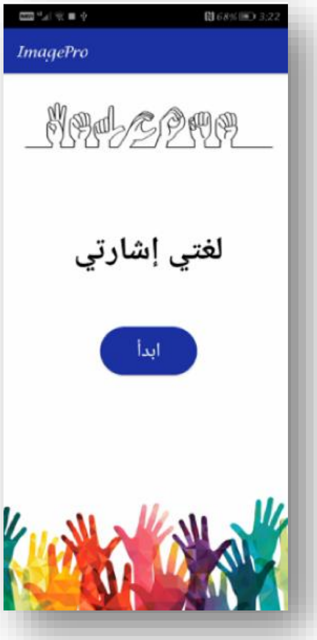
- ***Improve the Dataset***

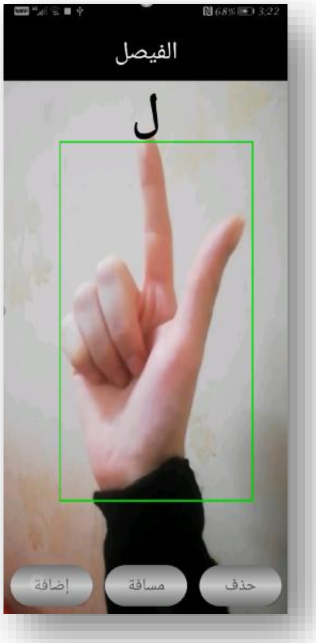
We did some elimination for the images that may confuse the machine and limit the hand gesture detection, such as the far hands images or the ones that show the whole forearm. Also, we eliminated the images that shows side hand poses cause a confusion for the model as hands from sides are not showing a clear gestures, which resulted in extra improvement in our model.

- ***Test the Final Model***

Android App Description:

An Android app that detects the different hand gestures and translates it into Arabic letters. The detected letter will be displayed on the screen immediately while doing the gestures. Also, the app has a text field at the top of the screen, with three different buttons at the bottom. The user can add the displayed letter to the text field by using the add button, so user can form words or even sentences after connecting the letters. There is other two buttons to add a space and delete the last written letter.





Sign Language Recognition Model (Classification Model):

We built and trained our model which consists of 32 classes using TensorFlow library in Python. In the first iteration, we started with batch-size = 96 and epochs = 30. The model reflected a bad accuracy, and just 2 classes are detected correctly. Then, we worked on improving our model by increasing the number of epochs and decreasing the batch size. Also, we checked the dataset and tried to eliminate the images that can confuse our model, especially the images that captures side poses of the hand. The model detection was improving gradually with the edits done. After testing over multiple iterations, we ended with batch size = 32 and epochs = 100.

Before building the model, the data is split into 80% of training data and 20% test data for testing and evaluation purposes.

- As a first step, we did an initial testing for the model accuracy and check if it is acceptable to continue with. We implemented a small test that take 15 samples from the test data and make predictions using `model.predict()` method. The results are as follows (the classes represented as numbers from 0 to 31):

```
[[ 1.] [19.] [ 3.] [ 3.] [23.] [26.] [17.] [15.] [-0.] [24.] [19.] [14.]
[18.] [ 9.] [10.]]
```
→ predicted classes

```
[ 1. 19. 3. 3. 23. 26. 17. 15. 0. 24. 19. 14. 18. 9. 10.]
```
→ Actual class number

We can notice that we have initially a good model with high accuracy. All the 15 images samples from the test data are predicted correctly and returned the same class number.

- After testing some samples of our model, we want to test now performance of our classification model as whole. Confusion matrix is a summary of prediction results on a classification problem. The number of correct and incorrect predictions are summarized with count values and broken down by each class, so in case to the model confuse between two or more classes, it will be clarified within the matrix. Each letter validation dataset will be tested to give prediction that compared with the classes exist. The result is as follows:

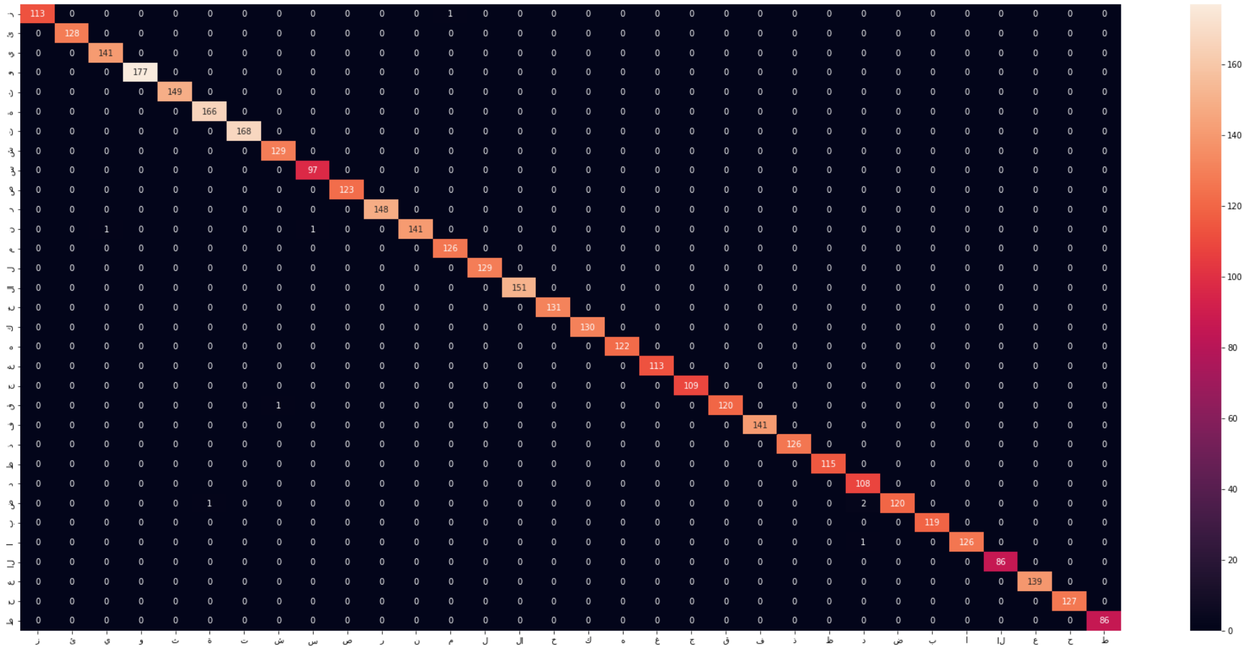


Figure 13

The matrix shows that almost all the images give correct prediction. Just few sample images confused with other letters, but we can consider it insignificant as it roughly forms no more that 2% of the whole validation dataset.

- Accuracy is another value we wanted to check, and it’s the number of correct predictions divided by the total number of predictions. The accuracy score we got is:
  `0.9980544747081712`

All the previous tests are conducted on images from the test dataset, the next step is to predict the correct letter in real-time using OpenCV camera in Android.

Realtime Detection

Letter	Observation	Letter	Observation
ا	Detected well just in case the hand on side pose	ظ	Detected only on the bottom of the screen
ب	Medium accuracy, signal value is not stable	ع	Detected with good accuracy
ت	Detected with good accuracy, but need to keep the hand strait	غ	Not detected, Non-stable signal value
ث	Detected with good accuracy	ف	Detected with good accuracy, but only in very specific pose
ج	Detected with good accuracy	ق	Not detected, conflicts with (ف)
ح	Detected with good accuracy	ك	Detected with good accuracy
خ	Detected with good accuracy	ل	Detected with good accuracy
د	Detected with good accuracy	م	Detected with medium to low accuracy
ذ	Detected with good accuracy	ن	Not detected, Non-stable signal value
ر	Detected with good accuracy	ه	Detected with medium accuracy

ز	Hard to detect → the gesture is hard to perform ☹	و	Detected with medium accuracy
س	Detected with good accuracy, sometimes it conflicts with (ش)	ئ	Detected with medium to low accuracy
ش	Detected with medium accuracy	ة	Not detected, Non-stable signal value
ص	Not detected, Non-stable signal value	ال	Detected with good accuracy
ض	Not detected, Non-stable signal value	لا	Detected with medium accuracy
ط	Detected with good accuracy, but need to keep the hand strait	ي	Detected with good accuracy

From the previous table we can understand that almost 50% of the letters have good detection accuracy, around 25% has some conflicting problems, and the last 25% are not detected at all. Having 32 classes for similar object (hand in our model) typically cause conflicting problems. However, we need to investigate the weaknesses of our model and how we can improve it in the future. Based on our observation for the Realtime detection app, we noticed that in some cases the detection is affected by positioning the fingers in very specific poses, such as moving one of the fingers about 0.5cm from its suggested place, which can reduce the accuracy significantly, which mean the model is not flexible for some letters. Also, the background noises play a role in reducing the accuracy while detecting the hand gestures, so letters with good accuracy will have medium accuracy with the background noises, and medium ones to low. Another problem observed in the model is that it serves more right-handed people, which means left-handed people will face problems using the app.

Based on the previous observations, we can suggest the following as a future work for the project:

- Consider adding more images for each gesture with the different possible fingers positions.
- Consider adding images with different possible background noises such as laptops, notebooks, colorful carpets, and curtains.
- Consider adding images for gestures that suit left-handed people.

Conclusion

In this project, we have built an android app that implements an Arabic sign language recognition model using tools learned in Artificial Intelligence. With this app, people can educate themselves about Arabic sign language and most importantly Arab deaf and mute individuals can communicate easily with those who do not acquire this language. We as developers also, we learnt more about image processing and the techniques and tools that can be user to develop an object detection mobile application.

References

1. Brownlee, J. (2020, August 15). *What is a Confusion Matrix in Machine Learning*. Machine Learning Mastery. https://machinelearningmastery.com/confusion-matrix-machine-learning/
2. Brownlee, J. (2021, January 26). *A Gentle Introduction to Object Recognition With Deep Learning*. Machine Learning Mastery. https://machinelearningmastery.com/object-recognition-with-deep-learning/
3. Huilgol, P. (2019, August 24). *Accuracy vs. F1-Score - Analytics Vidhya*. Medium. https://medium.com/analytics-vidhya/accuracy-vs-f1-score-6258237beca2

4. *Image Classification Explained: An Introduction*. (2020). H. https://www.v7labs.com/blog/image-classification-guide
5. M. (2020, May 30). *YOLO — You Only Look Once - Towards Data Science*. Medium. https://towardsdatascience.com/yolo-you-only-look-once-3dbdbb608ec4
6. *Papers with Code - Object Detection*. (2019). Machine Learning Mastery. https://paperswithcode.com/task/object-detection
7. *What Is Object Detection?* (2018). MATLAB & Simulink. https://www.mathworks.com/discovery/object-detection.html