IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 7.39 ⁜ Vol. 11, Issue 5, May 2022

DOI: 10.17148/IJARCCE.2022.11554

Real-Time Hand Gesture and Sign Language Detection and Translation

Lavneesh Jaggi[1], Nitish Pasricha[2], Namita Goyal[3]

Maharaja Agrasen Institute of Technolgy, New Delhi

Abstract: Due to the huge advancements in computer vision research in recent years, many real-world tasks can be automated without human intervention using deep learning models. Hand gesture and sign language detection and translation is also one such area where using a person's webcam, the hand gestures can be automatically detected and translated accurately. In this paper, we have explained our research on developing a real-time hand gesture and sign language detection and translation web application. Details about our implementation and the tools used are given. Finally, a summary of the results, future scope and conclusion are explained.

1. INTRODUCTION

Hand gesture recognition is one of the hot topics of research in the past few years. It has a wide range of applications including but not limited to human-computer interaction and sign language translation. Real-time sign language translation will help many people to communicate not only with other people but also with computers. It will help in making deaf and dumb people more independent by reducing the need for personal translators.

This project aims to create a robust real-time hand gesture and sign language detection and translation web application that is easy to use for everyone and provides the output instantly with as high accuracy as possible. This project benefits society as a whole by helping the disabled community by providing them with a platform for easier communication.

We have divided the hand gesture detection and translation process into two main parts:

- detecting the locations of key points of all fingers
- using these key points to predict the gesture represented bythe hand

For the sign language letter detection and translation, we have used transfer learning on SSD MobileNet V2 model from tensorflow which we trained on our own dataset.

The paper has been divided into the following sections: Section 2 explains the theory of the models used in the project. Section 3 gives details about our implementation and the tools used to develop the application. Section 4 describes the results of the application. The future scope has been described in Section 5. Finally, section 6 provides the conclusion of the paper.

2. THEORY

2.1 Hand Gesture

To develop the hand gesture detection and translation algorithm, we decided to use the handpose model from TensorFlow and fingerpose classifier. These models work together to form a pipeline for hand gesture detection starting from detecting whether a palm is present, then locating the key points to differentiate between different fingers and also the thumb to finally predict the gesture using the key points.

2.1.1. Handpose model

Handpose model handles the first task of detecting a palm and locating the finger key points and gives their coordinates as output. This model uses two models for its pipeline:

- The palm detector model to detect the palm from the full-sized input image
- The landmark model that takes the cropped hand image from the palm detector model and predicts the landmark key points of allfingers

The palm detector model is used to improve the performance of the landmark model since landmarks are only detected when a hand is present in the image and the initial bounding box of the detected hand is continuously used to detect the

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 7.39 ⁜ Vol. 11, Issue 5, May 2022

DOI: 10.17148/IJARCCE.2022.11554

hand key points until the hand is removed. This also reduces the need to run the palm detector model on every frame. This model is known as the BlazePalm detector and is trained to detect a palm instead of an entire hand with fingers since detecting a palm is much simpler and easier for the model because of the absence of individual fingers. It has been trained on a real-world dataset. It uses an encoder-decoder and a focal loss which gives an average precision of 95.7%.

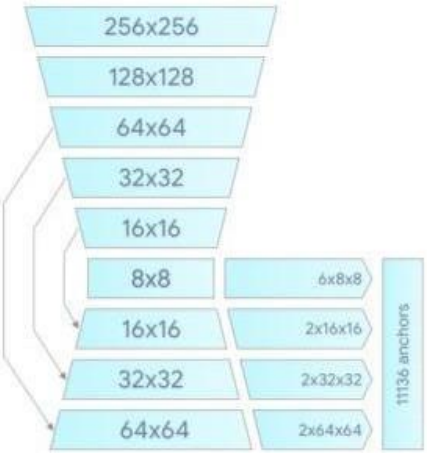

Figure 1: Palm detector model architecture

The landmark model gives the following outputs:

- 21 finger key points including the x-coordinate, y-coordinate and relative depth
- Probability of hand being present in the given image
- Classification of left-handed vs right-handed

The landmark model has been trained on real-world datasets as well as synthetically generated datasets. The synthetic dataset is used to train the model to predict the relative depth from the central landmark. This model is also trained to predict handedness using a binary classifier.

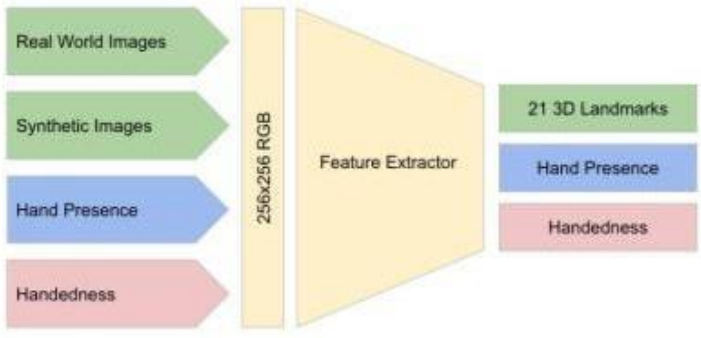

Figure 2: Landmark model architecture

2.1.2. Fingerpose classifier

The fingerpose classifier works like an extension to the handpose model to complete the pipeline of gesture detection. It uses the landmarks given by the handpose model to estimate the hand gesture represented.
It works in two steps:

- Estimating the curls and direction of each finger using mathematical and geometric functions
- Comparing the results of curls and direction to predefined gesture descriptions to predict the closest matching gesture

We have defined 8 gestures that this model can recognize and translate in real-time using different curls and directions of each finger and thumb.
The following values are used for the accepted finger curls:

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 7.39 ⁜ Vol. 11, Issue 5, May 2022

DOI: 10.17148/IJARCCE.2022.11554

- NoCurl
- HalfCurl
- FullCurl

The following values are used for the accepted finger directions:

- Vertical Up
- Vertical Down
- Horizontal Left
- Horizontal Right
- Diagonal Up Right
- Diagonal Up Left
- Diagonal Down Right
- Diagonal Down Left

2.2. Sign Language Letters

Tensorflow's SSD MobileNet V2 model was used to predict the English letters. It is pretrained on COCO 2017 dataset and has been created using the Tensorflow Object Detection API. This model is a Single Shot Detector model that learns to predict bounding box locations and classify these locations in one pass. Hence, SSD can be trained end-to-end. The SSD network consists of base architecture (MobileNet in this case) followed by several convolution layers.
SSD models are generally much faster than RPN (Regional Proposal Network) approaches because they only use one shot to detect the region as well as the object present in that region.

The model gives the following outputs:

- num_detections: a tf.int tensor with only one value, the number of detections [N].
- detection_boxes: a tf.float32 tensor of shape [N, 4] containing bounding box coordinates in the following order: [ymin, xmin, ymax,xmax].
- detection_classes: a tf.int tensor of shape [N] containing detection class index from the label file.
- detection_scores: a tf.float32 tensor of shape [N] containing detection scores.
- raw_detection_boxes: a tf.float32 tensor of shape [1, M, 4] containing decoded detection boxes without Non-Max suppression. M is the number of raw detections.
- raw_detection_scores: a tf.float32 tensor of shape [1, M, 90] and contains class score logits for raw detection boxes. M is the number of raw detections.
- detection_anchor_indices: a tf.float32 tensor of shape [N] and contains the anchor indices of the detections after NMS.
- detection_multiclass_scores: a tf.float32 tensor of shape [1, N, 91] and contains class score distribution (including background) for detection boxes in the image including background class.

We used detection_boxes, detection_classes and detection_scores to display the results to the user.

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 7.39 ⁜ Vol. 11, Issue 5, May 2022

DOI: 10.17148/IJARCCE.2022.11554

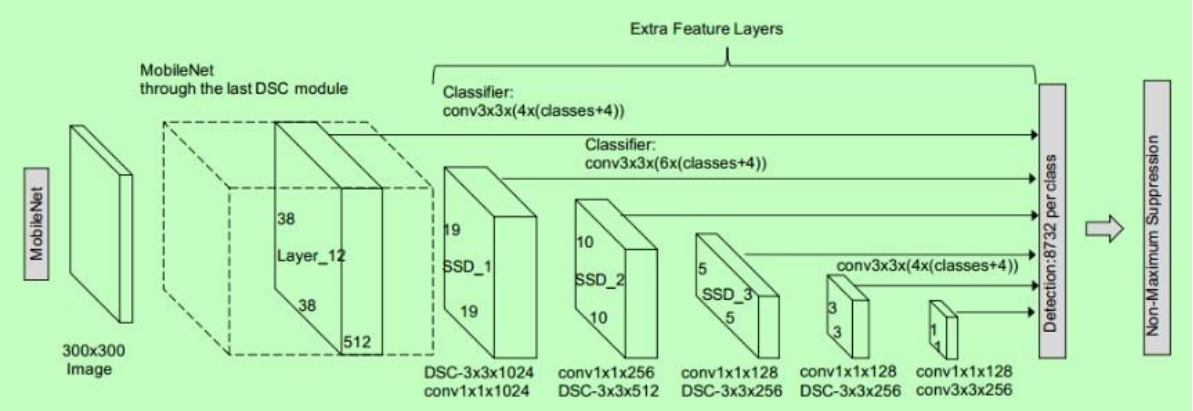

Figure 3: MobileNet Model Architecture

3. IMPLEMENTATION

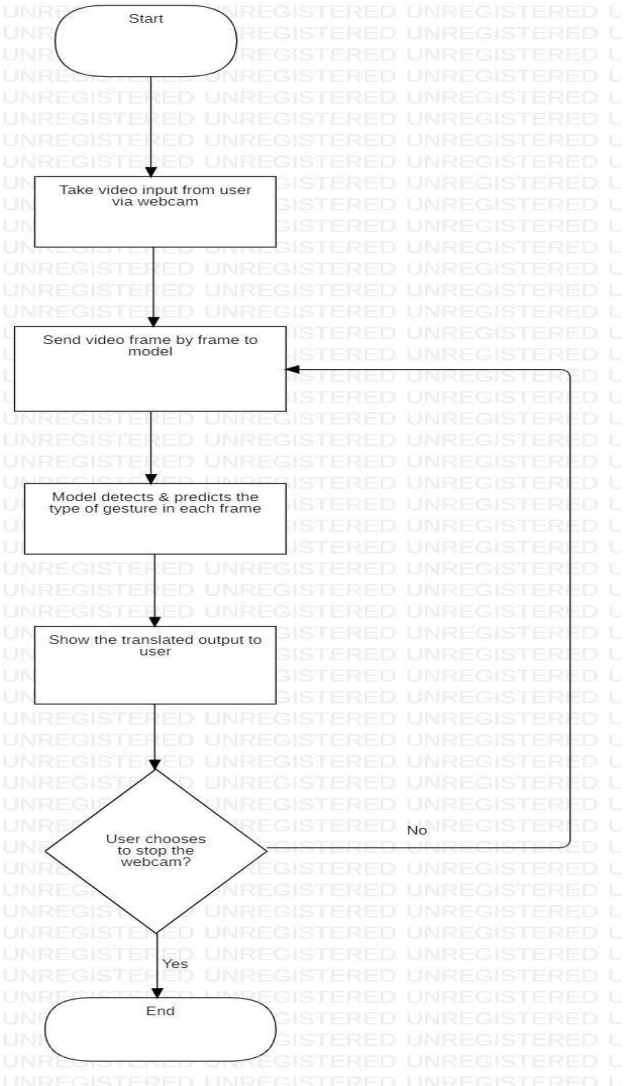

Figure 3: Flow chart diagram of the web application

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 7.39 ⁜ Vol. 11, Issue 5, May 2022

DOI: 10.17148/IJARCCE.2022.11554

The following javascript libraries were used for the development of this project:

React

React JavaScript framework was used to create the web application which is one of the most popular web frameworks currently. This particular framework was used because it is one of the best frameworks to handle real-time changes in the web page which is the biggest advantage for our website, as the predicted gesture output and visualizations are constantly being updated. React is an open-source library that is developed and maintained by Facebook. It was initially released on May 29, 2013.

Tensorflow

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. TensorFlow was developed by the Google Brain team for internal Google use in research and production. The initial version was released under the Apache License 2.0 in 2015. Google released the updated version of TensorFlow, named TensorFlow 2.0, in September 2019.

LabelImg

LabelImg is a graphical image annotation tool. It is written in Python and uses Qt for its graphical interface. Annotations are saved as XML files in PASCAL VOC format, the format used by ImageNet. Besides, it also supports YOLO and CreateML formats.

TensorFlow Js

The TensorFlow library is the backbone of our models as the models are developed in TensorFlow and this library is needed to use and run the models. It is a machine learning library for JavaScript which can be used to develop and run models directly in the browser. It can also be used to convert python TensorFlow models to JavaScript-based models and retrain existing models on custom datasets.

React-webcam

The react-webcam library was used to get the user's video footage from the webcam so that it can be used for hand gesture detection. This library allows us to create a webcam component that can also be used to record the video footage.

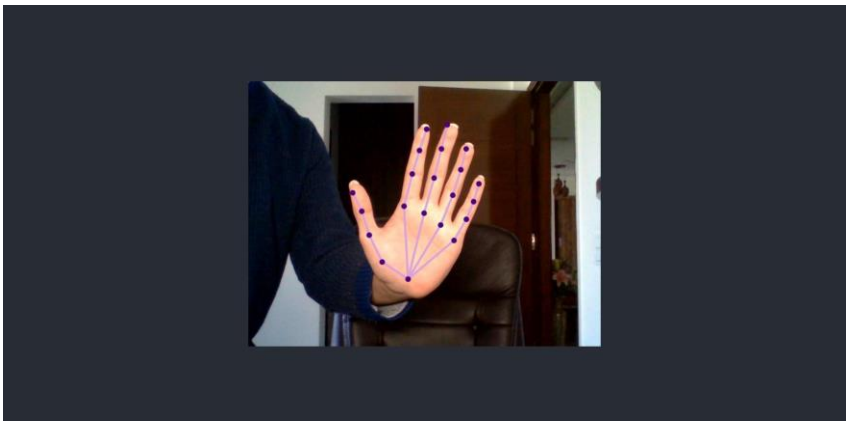

Figure 4: Visualization of the hand created by drawHand module

3.1. Gesture Detection

The following steps were followed for the implementation of the project:

- A module was created to fetch the video feed from the user's webcam in React
- Acanvas was created to display the webcam's feed to the user in a window where the results will be displayed
- The drawHand module was created to display the hand landmarks and join them to the palm to ensure the correct landmarks are detected and for better visualization for the user. This module used the hand landmarks predicted by the handpose model and looped in all the fingers and joined each pair of landmarks using lines. Canvas drawing functions were used to create the lines and arcs. See Figure 4 for the output.

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 7.39 ⁜ Vol. 11, Issue 5, May 2022

DOI: 10.17148/IJARCCE.2022.11554

- Eleven gestures have been defined for recognition and translation of the gesture. Curls and directions of each finger have been defined for accurate detection. Each gesture was defined in a separate JavaScript file
- The detect module was created which was used to perform the following tasks:

1. Detect if the webcam video is currently being processed or not
2. Set the attributes of the video and canvas such as height and width according to the input video
3. Call the handpose model by passing it the video using the estimateHands method
4. If a hand was detected by the handpose model, the gestures were initialized for the fingerpose classifier using the GestureEstimator method
5. Call the fingerpose classifier by passing the handlandmarks given by the handpose model
6. When the model detects a gesture it assigns a confidence score to multiple gestures, we loop through each confidence score of the gesture from the gesture array and identify the maximum confidence score to display the corresponding translation
7. Call the drawHand module to visualize the hand

- runModel module was created to load the handpose model and then call the detect module every 100ms to continuously detect the gestures in real-time
- Finally, a module to display the result of the predicted gesture was created which displayed the emoji image corresponding to the detected gesture

3.2. Sign Language Letter Detection

- Image dataset was collected for American Sign Language of English letters. We captured 15 images of every letter making the whole dataset of 390 images. The images were captured using openCV
- The collected images were cropped and labelled using LabelImg so that the model is able to detect the region of the hand and also predict the sign being represented
- 12 images of each letter were used to train the SSD MobileNet V2 model using the Tensorflow Object Detection API. The model was trained for 50000 iterations
- After training, the model was saved and converted to Tensorflow JS model to use it in the web application
- The converted model was deployed on cloud storage using IBM cloud
- The model was loaded in the web application and a function was created to run the model and draw the bounding box of the hand and display the class predicted by the model

4. RESULTS

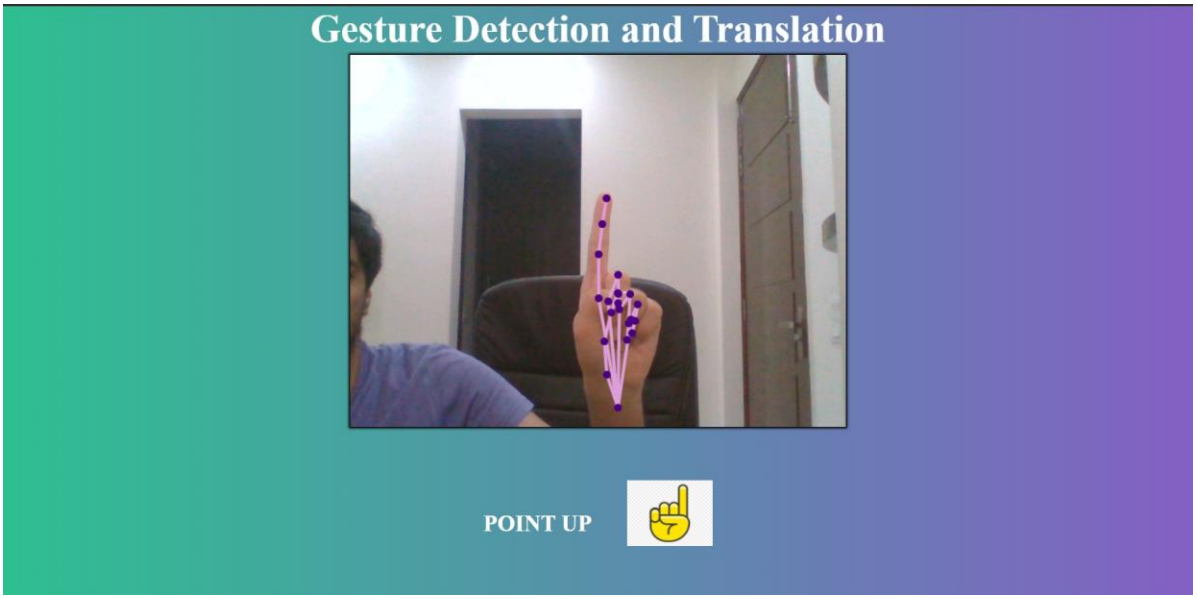

Figure 5: Working of hand gesture detection

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 7.39 ⁜ Vol. 11, Issue 5, May 2022

DOI: 10.17148/IJARCCE.2022.11554

The hand gesture detection can detect the following gestures:

- Victory
- Thumbs up
- Thumbs down
- I love you
- Point left
- Point right
- Point up
- Point down
- Okay
- Stop
- Yes

The model has an average accuracy of 96.2%. The application starts to detect the gestures immediately in real-time showing the output as an image along with a text. The output is displayed every 100ms continuously as long as a hand is present and a suitable gesture is detected. Apart from the gesture output, the hand is also visualized using landmarks and lines.

The sign language letter detection can detect all the English alphabets from A to Z. The accuracy of the model is 87.5% on the testing dataset. This model displays the score of the class being predicted and output is drawn as a bounding box along with the predicted class and score.

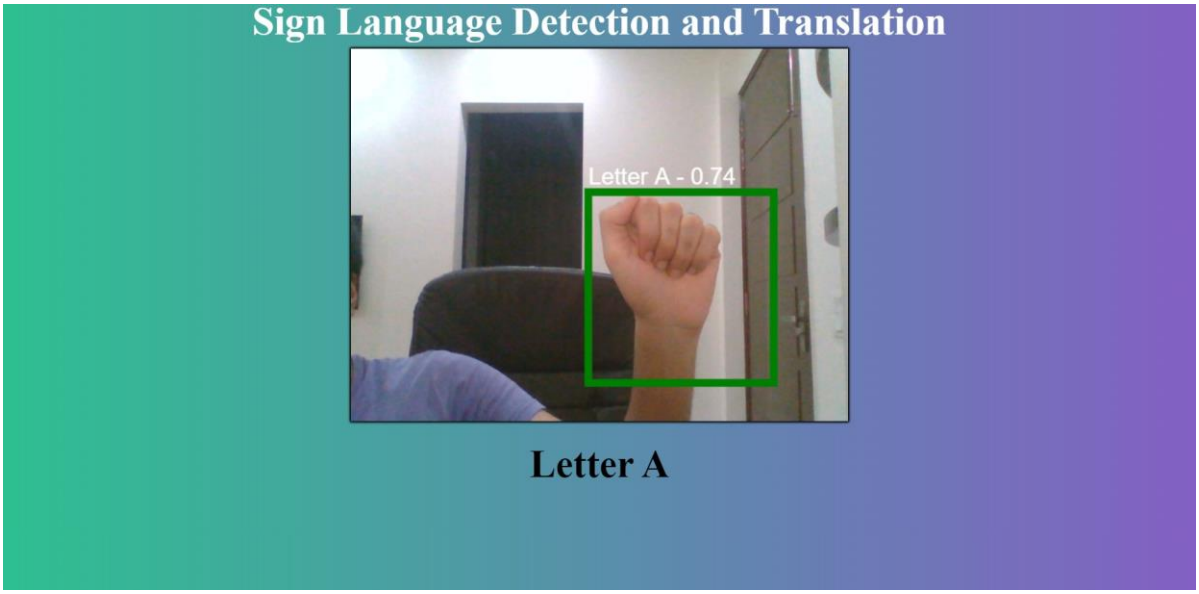

Figure 6: Working of Sign Language Letter Detection

5. FUTURE SCOPE

The project has a lot of scope for future enhancements to make it more useful, user-friendly as well as extend its functionalities. The following points describe some suitable extensions:

- Hand gesture detection pipeline can be used to translate sign language in real-time where individual letters can be combined to form a sentence in English and the whole output can bedisplayed to the user
- This pipeline can also be used to make a video calling application for deaf and dumb people so that they can communicate with others in real-time directly
- An application can be created to use different hand gesture inputs to control software like a web browser, video game or a productivity app without the use of a mouse and keyboard

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 7.39 ⁜ Vol. 11, Issue 5, May 2022

DOI: 10.17148/IJARCCE.2022.11554

6. CONCLUSION

We successfully built the project to detect hand gestures and sign language letters in real-time in a web application. We discussed the theory that we learned during the development of this project and also gained practical knowledge of the tools including but not limited to the React framework, TensorFlow, handpose model and the fingerpose classifier. We also discussed our approach to the implementation of this project using these technologies. We also described the results of the developed application along with its usage. Finally, we provided the opportunities for the future scope of this project and its wide range of applications which are not only limited to personal use cases but also to commercial purposes for software companies to develop such applications.

ACKNOWLEDGEMENT

We would like to thank our institute, Maharaja Agrasen Institute of Technology, for providing us with this opportunity to undertake this research project. We would also like to express our sincere gratitude to our mentor, **Mrs Namita Goyal,** for her constant support and guidance throughout the process.

REFERENCES

[1] Fan Zhang, Valentin Bazarevsky, Andrey Vakunov, Andrei Tkachenka, George Sung, Chuo-Ling Chang and Matthias Grundmann. MediaPipe Hands: On-device Real-time Hand Tracking, arXiv:2006.10214
[2] Handpose model https://www.npmjs.com/package/@tensorflow-models/handpose
[3] Fingerpose model https://github.com/andypotato/fingerpose
[4] React (JavaScript library) – Wikipedia https://en.wikipedia.org/wiki/React_(JavaScript_library)
[5] TensorFlow.js https://www.tensorflow.org/js
[6] React-webcam https://www.npmjs.com/package/react-webcam
[7] Tensorflow https://tfhub.dev/tensorflow/ssd_mobilenet_v2/2