# Sign Language Detection

Artificial Intelligence, MCSC102
Department of Computer Science,
University of Delhi

**Harsh Yadav (20)**
**Purnima Kumar (41)**
**Yashi Sharma (58)**

# Overview



# Quick Recap

- We aim to develop a **computer-vision based AI system** that can accurately interpret and translate **sign language gestures.**
- **Input:** Images of hand gestures representing American Sign Language (ASL).
- **Output:** Accurate textual translations of English alphabet (A-Z) and numbers (0-9).

# Methodology

Having conducted a comprehensive literature review, we have identified certain methodologies that have demonstrated notable efficacy in the detection of Sign Language Gestures.

## Transfer Learning

What is transfer learning?

It is a process where the weights of an existing pretrained model that was created to perform a similar task are retrained on a new data in order to perform a new, but similar task.

## Convolutional Neural Networks

Since, we are dealing with a multi-class image classification problem Convolutional Neural Networks is a go-to architecture.

# CNN Architecture

## 01 Input

INPUT holds the raw pixel values of the image, in this case an image of width , height , and with three color R, G, B.

## 02 Convolutional Layers

The Convolution layers computes the output of the neurons that are connected to local regions in the input.

## 03 Pooling Operations

The Pooling layer performs a downsampling operation along with the spatial dimensions.

## 04 Activation Functions

ReLu activation function applies an elementwise activation function, i.e., f(x) = max(0, x).

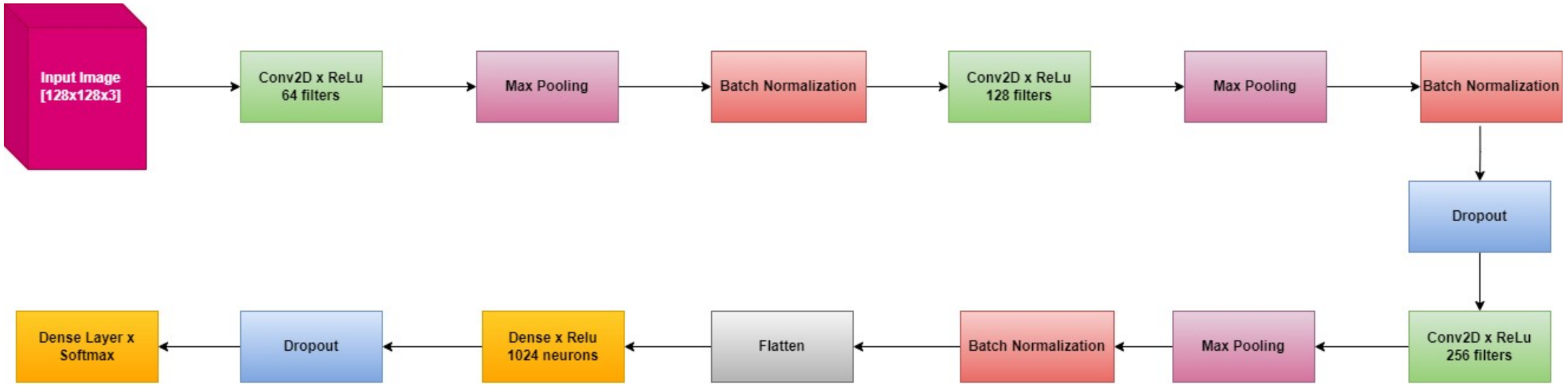
## 05 Fully Connected Layers

In the end, the fully connected layers computes the class scores.

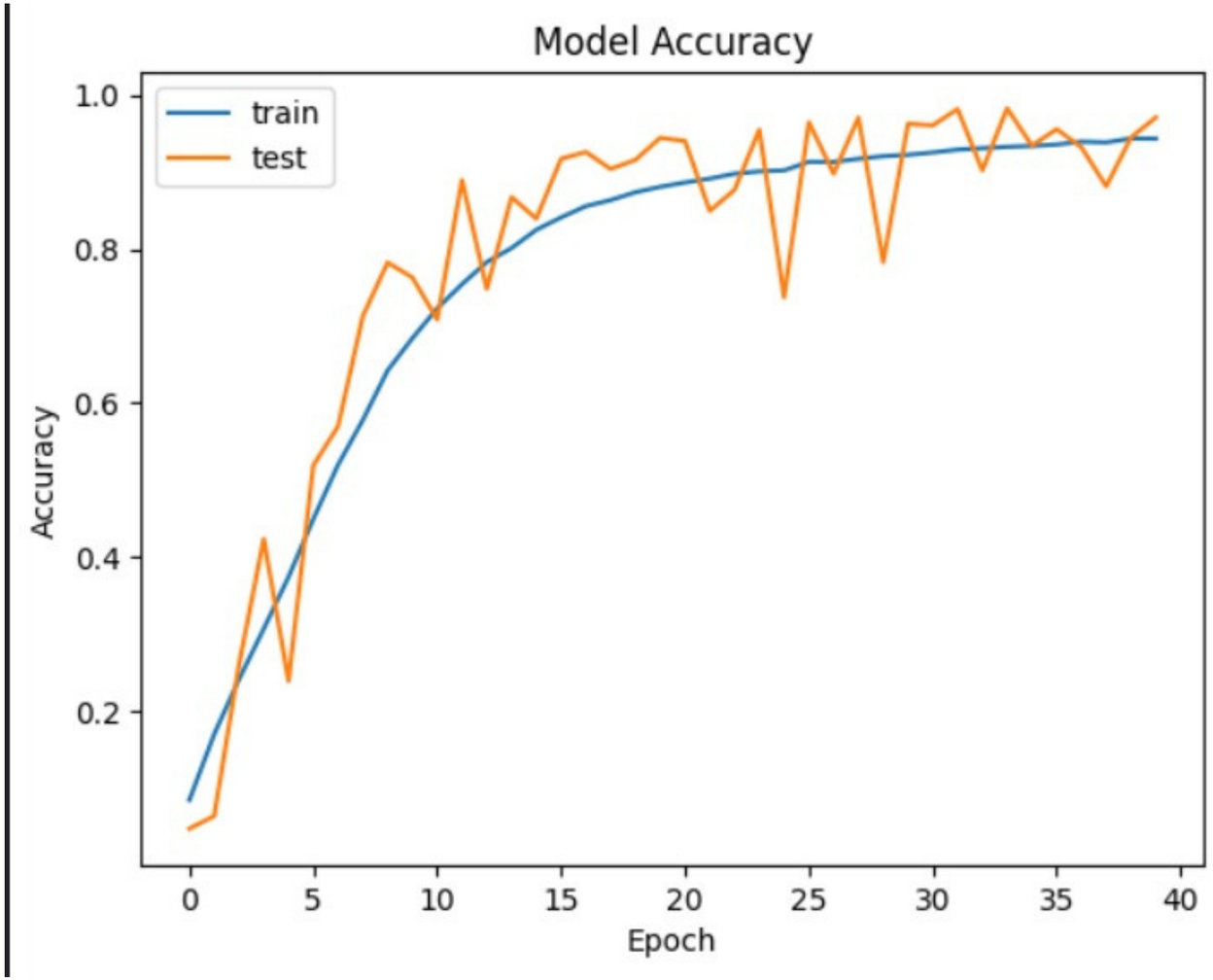
## 06 Output

We get a single neuron as an output which contains class scores for all the classes.

# Implementation Using CNNs

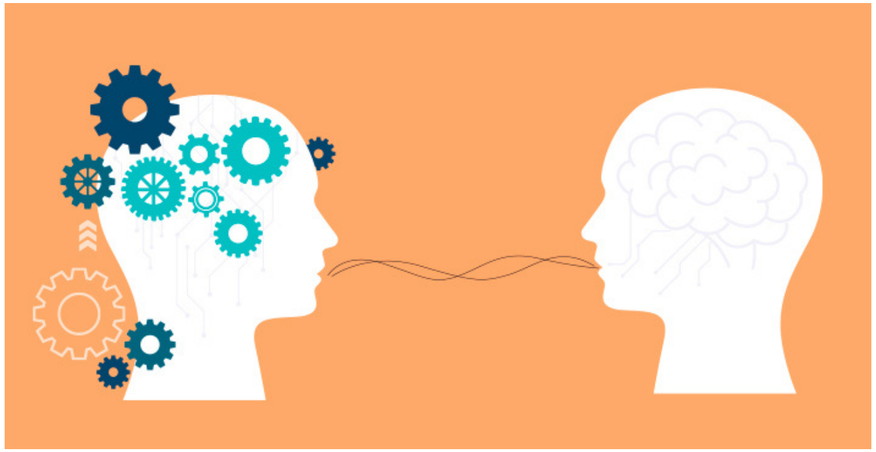


# Experimental Results



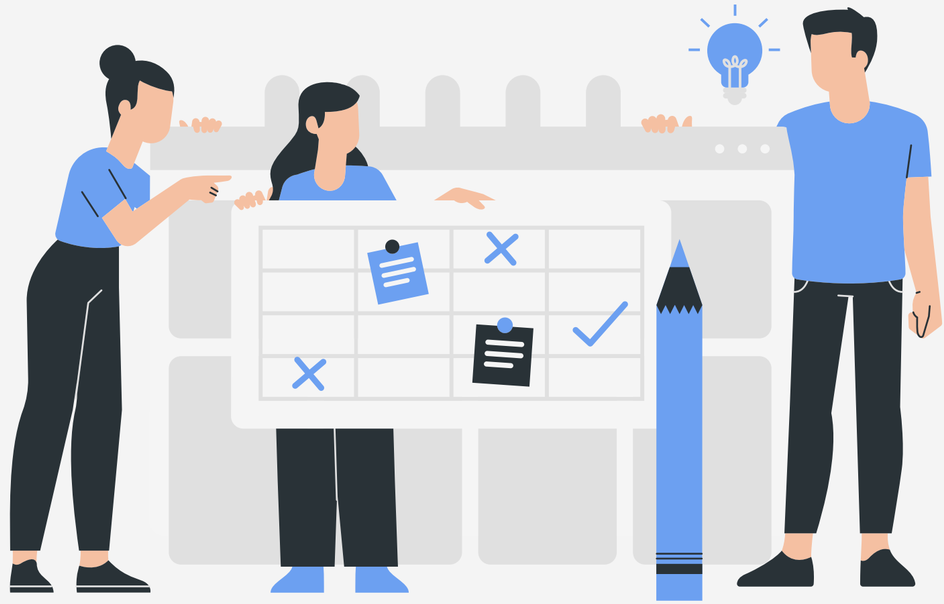
# Transfer Learning Approach

- Transfer Learning is a process where the **weights of an existing pretrained model** that was created to perform a similar task are **retrained on a new data** in order to perform a new, but similar task.



- Used Microsoft COCO dataset, which is a large-scale object detection dataset containing **330,000 images** consisting of **80 object categories**.

# Methodology applied by us to detect words.



# Captured Images using OpenCV

## 5 classes (10 images each)
## Hello, Yes, No, ThankYou, ILoveYou

Hello

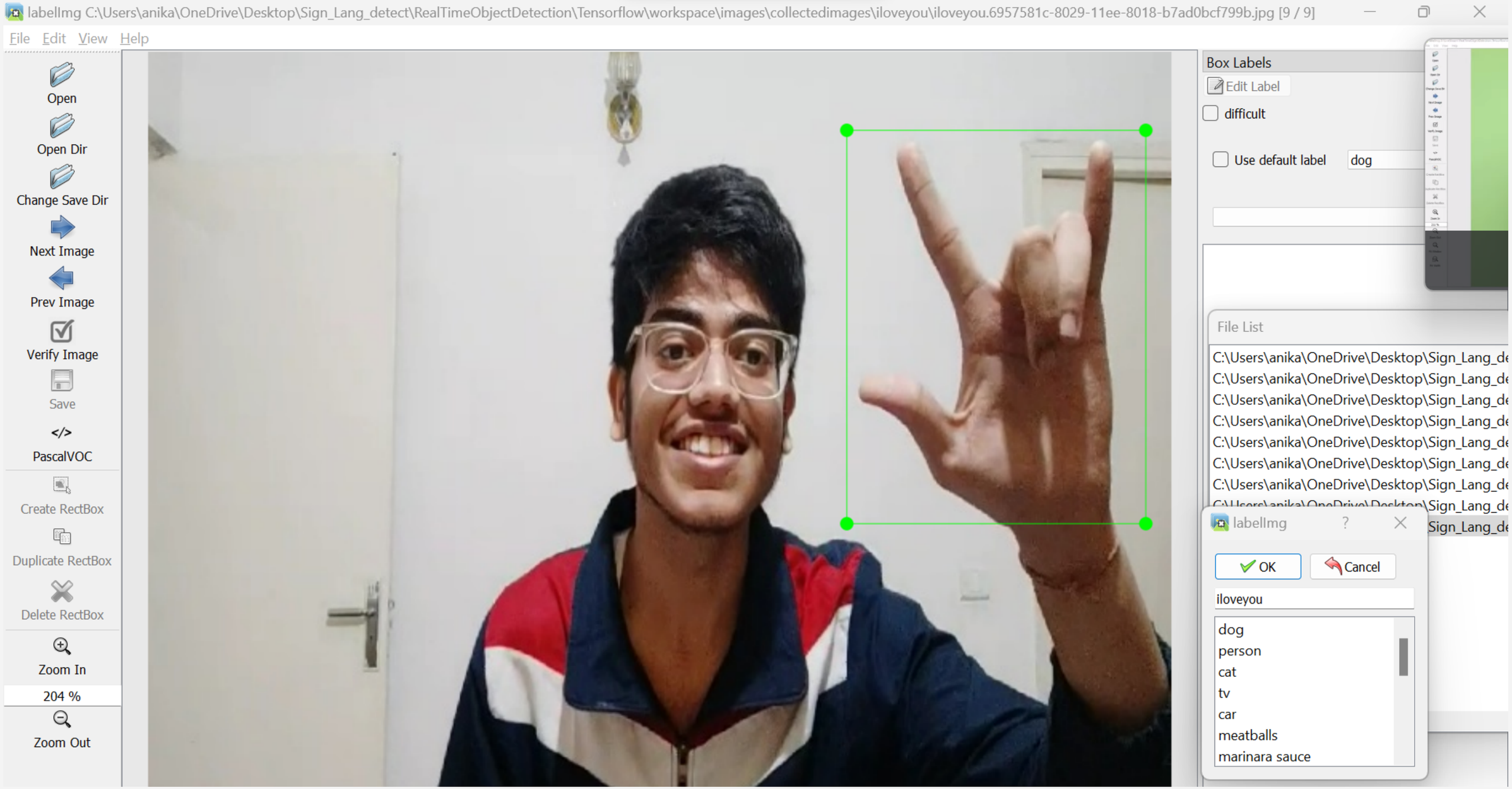
YES

# I Love You

No

Thanks

# Labelling Image for Object Detection



# Training the model.

Used **TensorFlow_2's SSD_mobilenet_v2** detection model, pre-trained on the COCO 2017 dataset.

Used this pre-trained model for '**Transfer learning**'.

Anaconda Prompt (anaconda

```
(base) C:\Users\anika\OneDrive\Desktop\Sign_Lang_detect\RealTimeObjectDetection>conda activate hh

(hh) C:\Users\anika\OneDrive\Desktop\Sign_Lang_detect\RealTimeObjectDetection>python Tensorflow/models/research/object_detection/model_main_tf2.py --model_dir=Tensorflow/workspace/models/my_ssd_mobnet --pipeline_config_path=Tensorflow/workspace/models/my_ssd_mobnet/pipeline.config --num_train_steps=10000
C:\Users\anika\anaconda3\envs\hh\lib\site-packages\tensorflow_addons\utils\tfa_eol_msg.py:23: UserWarning:

TensorFlow Addons (TFA) has ended development and introduction of new features.
TFA has entered a minimal maintenance and release mode until a planned end of life in May 2024.
Please modify downstream libraries to take dependencies from other repositories in our TensorFlow community (e.g. Keras, Keras-CV, and Keras-NLP).

For more information see: https://github.com/tensorflow/addons/issues/2807

  warnings.warn(
C:\Users\anika\anaconda3\envs\hh\lib\site-packages\tensorflow_addons\utils\ensure_tf_install.py:53: UserWarning: Tensorflow Addons supports using Python ops for all Tensorflow versions above or equal to 2.12.0 and strictly below 2.15.0 (nightly versions are not supported).
 The versions of TensorFlow you are currently using is 2.10.1 and is not supported.
Some things might work, some things might not.
If you were to encounter a bug, do not file an issue.
If you want to make sure you're using a tested and supported configuration, either change the TensorFlow version or the TensorFlow Addons's version.
You can find the compatibility matrix in TensorFlow Addon's readme:
https://github.com/tensorflow/addons
  warnings.warn(
2023-11-15 11:04:42.825036: I tensorflow/core/platform/cpu_feature_guard.cc:193] This TensorFlow binary is optimized with oneAPI Deep Neural Network Library (oneDNN) to use the following CPU instructions in performance-critical operations:  AVX AVX2
To enable them in other operations, rebuild TensorFlow with the appropriate compiler flags.
```

Anaconda Prompt (anaconda

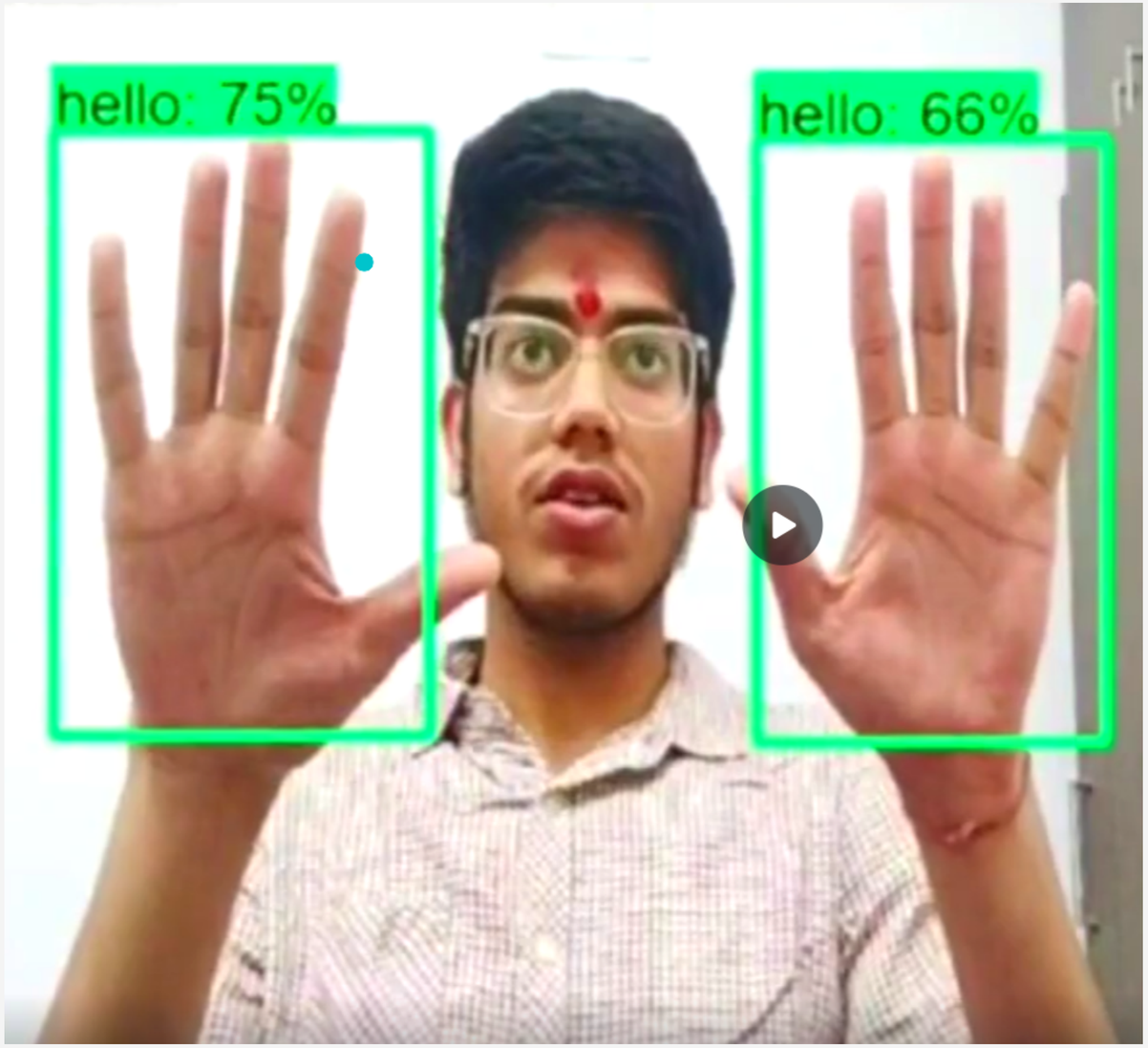
```
You can find the compatibility matrix in TensorFlow Addon's readme:
https://github.com/tensorflow/addons
  warnings.warn(
2023-11-15 11:04:42.825036: I tensorflow/core/platform/cpu_feature_guard.cc:193] This TensorFlow binary is optimized
 with oneAPI Deep Neural Network Library (oneDNN) to use the following CPU instructions in performance-critical oper
ations:  AVX AVX2
To enable them in other operations, rebuild TensorFlow with the appropriate compiler flags.
2023-11-15 11:04:44.263178: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1616] Created device /job:localhost/r
eplica:0/task:0/device:GPU:0 with 1340 MB memory:  -> device: 0, name: NVIDIA GeForce MX250, pci bus id: 0000:01:00.
0, compute capability: 6.1
INFO:tensorflow:Using MirroredStrategy with devices ('/job:localhost/replica:0/task:0/device:GPU:0',)
I1115 11:04:44.409381 15804 mirrored_strategy.py:374] Using MirroredStrategy with devices ('/job:localhost/replica:0
/task:0/device:GPU:0',)
INFO:tensorflow:Maybe overwriting train_steps: 10000
I1115 11:04:44.446255 15804 config_util.py:552] Maybe overwriting train_steps: 10000
INFO:tensorflow:Maybe overwriting use_bfloat16: False
I1115 11:04:44.446255 15804 config_util.py:552] Maybe overwriting use_bfloat16: False
WARNING:tensorflow:From C:\Users\anika\anaconda3\envs\hh\lib\site-packages\object_detection\model_lib_v2.py:563: Str
ategyBase.experimental_distribute_datasets_from_function (from tensorflow.python.distribute.distribute_lib) is depre
cated and will be removed in a future version.
Instructions for updating:
rename to distribute_datasets_from_function
W1115 11:04:44.483953 15804 deprecation.py:350] From C:\Users\anika\anaconda3\envs\hh\lib\site-packages\object_detec
tion\model_lib_v2.py:563: StrategyBase.experimental_distribute_datasets_from_function (from tensorflow.python.distri
bute.distribute_lib) is deprecated and will be removed in a future version.
Instructions for updating:
rename to distribute_datasets_from_function
INFO:tensorflow:Reading unweighted datasets: ['Tensorflow/workspace/annotations/train.record']
I1115 11:04:44.499662 15804 dataset_builder.py:162] Reading unweighted datasets: ['Tensorflow/workspace/annotations/
train.record']
```

```
memory trying to allocate 983.51MiB with freed_by_count=0. The caller indicates that this is not a failure, but this
 may mean that there could be performance gains if more memory were available.
2023-11-15 11:06:39.619296: W tensorflow/core/common_runtime/bfc_allocator.cc:290] Allocator (GPU_0_bfc) ran out of
memory trying to allocate 827.50MiB with freed_by_count=0. The caller indicates that this is not a failure, but this
 may mean that there could be performance gains if more memory were available.
INFO:tensorflow:Step 100 per-step time 0.845s
I1115 11:07:09.494543 15804 model_lib_v2.py:705] Step 100 per-step time 0.845s
INFO:tensorflow:{'Loss/classification_loss': 0.3461144,
 'Loss/localization_loss': 0.32647595,
 'Loss/regularization_loss': 0.15391405,
 'Loss/total_loss': 0.8265044,
 'learning_rate': 0.0319994}
I1115 11:07:09.529289 15804 model_lib_v2.py:708] {'Loss/classification_loss': 0.3461144,
 'Loss/localization_loss': 0.32647595,
 'Loss/regularization_loss': 0.15391405,
 'Loss/total_loss': 0.8265044,
 'learning_rate': 0.0319994}
INFO:tensorflow:Step 200 per-step time 0.386s
I1115 11:07:48.121766 15804 model_lib_v2.py:705] Step 200 per-step time 0.386s
INFO:tensorflow:{'Loss/classification_loss': 0.2539425,
 'Loss/localization_loss': 0.20083773,
 'Loss/regularization_loss': 0.15383898,
 'Loss/total_loss': 0.6086192,
 'learning_rate': 0.0373328}
I1115 11:07:48.123761 15804 model_lib_v2.py:708] {'Loss/classification_loss': 0.2539425,
 'Loss/localization_loss': 0.20083773,
 'Loss/regularization_loss': 0.15383898,
 'Loss/total_loss': 0.6086192,
 'learning_rate': 0.0373328}
INFO:tensorflow:Step 300 per-step time 0.407s
```

Anaconda Prompt (anaconda

```
 'Loss/localization_loss': 0.010172847,
 'Loss/regularization_loss': 0.09720672,
 'Loss/total_loss': 0.13367103,
 'learning_rate': 0.07380057}
INFO:tensorflow:Step 9900 per-step time 0.514s
I1115 12:29:41.151496 15804 model_lib_v2.py:705] Step 9900 per-step time 0.514s
INFO:tensorflow:{'Loss/classification_loss': 0.05131547,
 'Loss/localization_loss': 0.009596345,
 'Loss/regularization_loss': 0.09673182,
 'Loss/total_loss': 0.15764363,
 'learning_rate': 0.073662736}
I1115 12:29:41.154494 15804 model_lib_v2.py:708] {'Loss/classification_loss': 0.05131547,
 'Loss/localization_loss': 0.009596345,
 'Loss/regularization_loss': 0.09673182,
 'Loss/total_loss': 0.15764363,
 'learning_rate': 0.073662736}
INFO:tensorflow:Step 10000 per-step time 0.488s
I1115 12:30:29.914326 15804 model_lib_v2.py:705] Step 10000 per-step time 0.488s
INFO:tensorflow:{'Loss/classification_loss': 0.07445922,
 'Loss/localization_loss': 0.018851975,
 'Loss/regularization_loss': 0.09630479,
 'Loss/total_loss': 0.18961598,
 'learning_rate': 0.07352352}
I1115 12:30:29.917315 15804 model_lib_v2.py:708] {'Loss/classification_loss': 0.07445922,
 'Loss/localization_loss': 0.018851975,
 'Loss/regularization_loss': 0.09630479,
 'Loss/total_loss': 0.18961598,
 'learning_rate': 0.07352352}
```

# Results

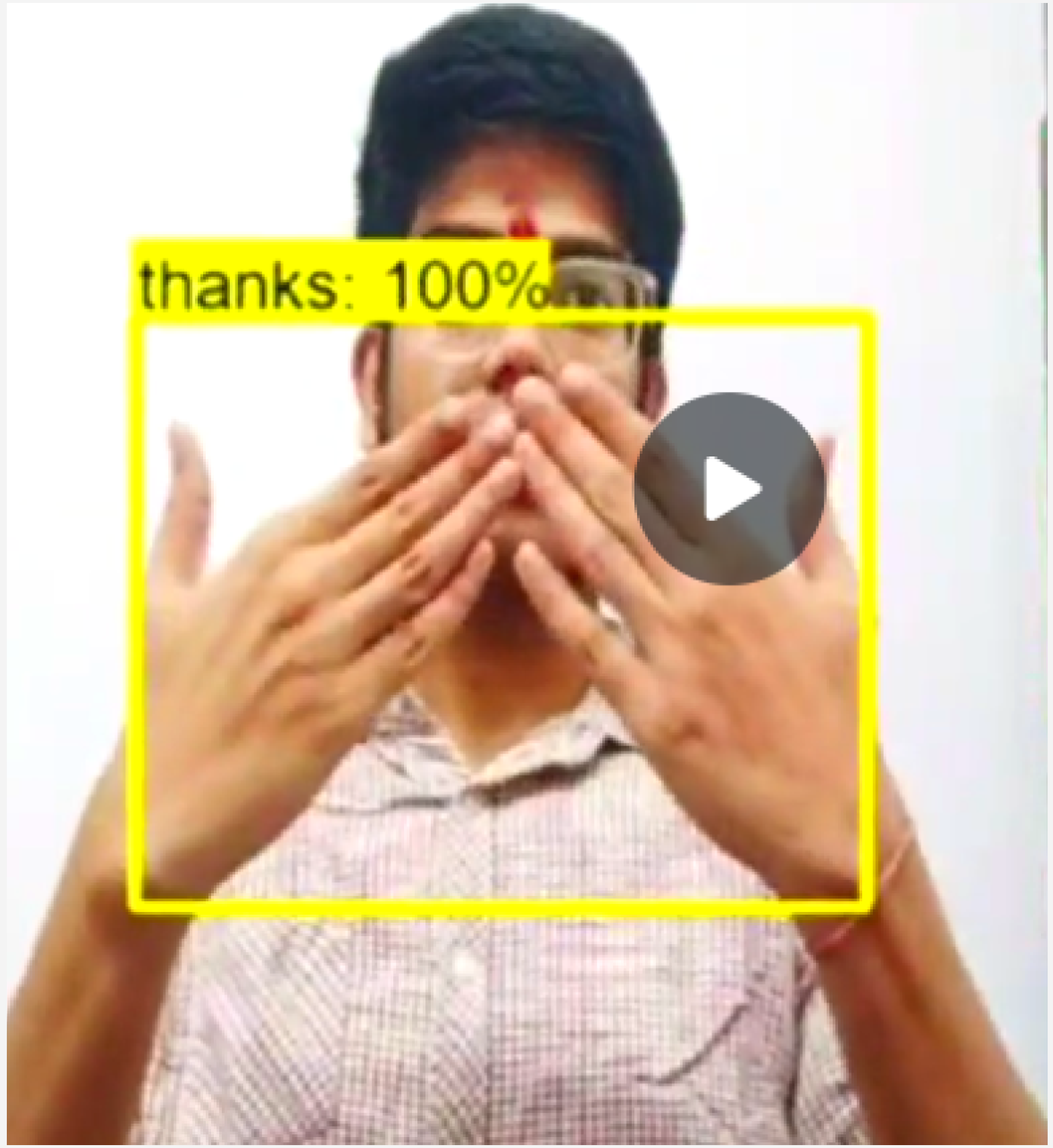



# Hello


# YES


# I Love You


# No



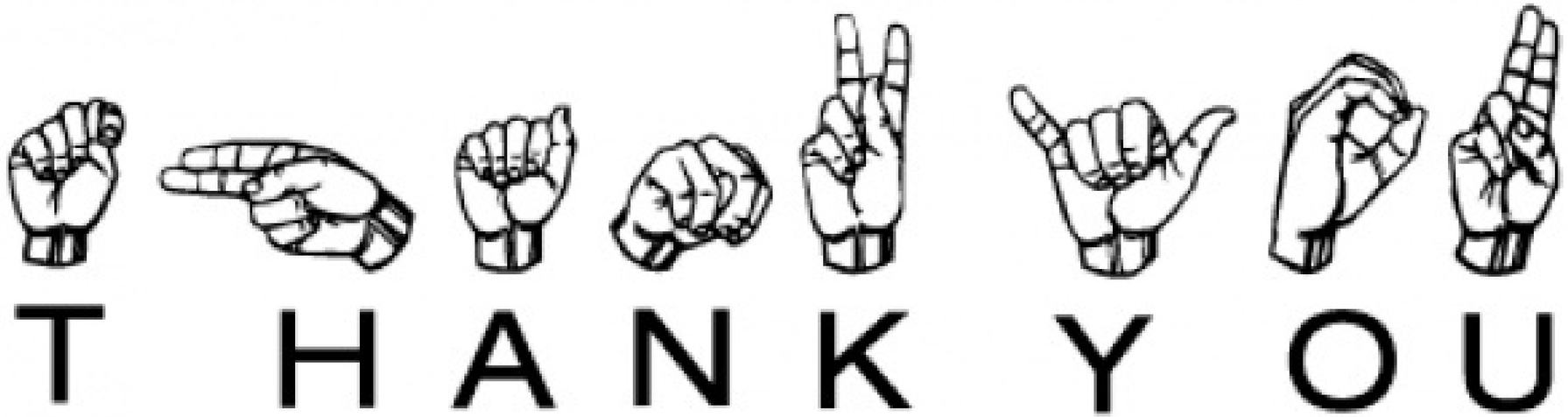

# Thanks

# Future Plans.

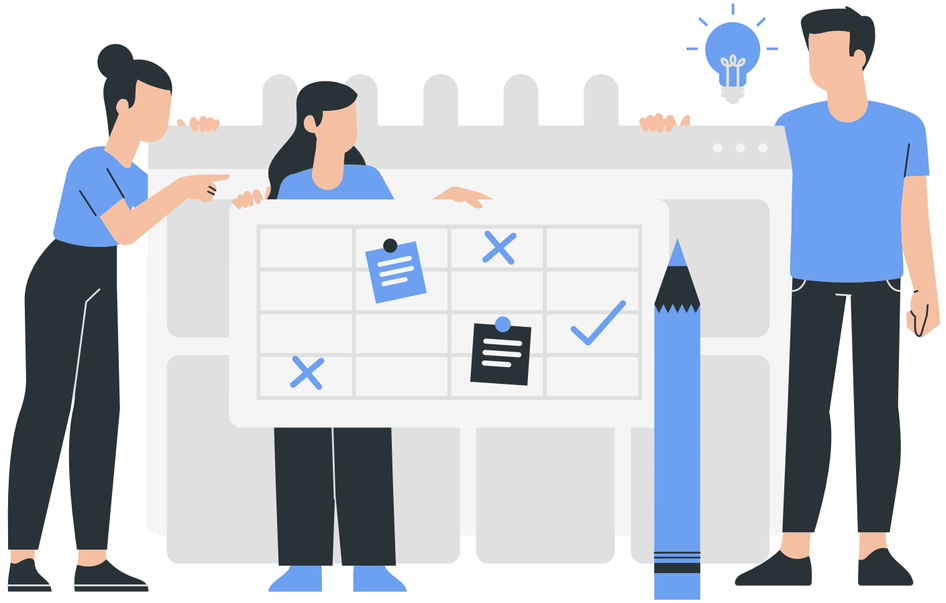
We plan to combine the two approaches demonstarted in the presentation to develop an end-to-end deployed AI system which can perform Sign Language Detection in real-time.

# References


- American sign language (ASL) recognition based on Hough Transform and Neural Networks. Qutaishat Munib, 2007.
- MacMaster, Gordon, "Sign Language Translation Using Machine Learning and Computer Vision" (2020).UVM Honors College Senior Theses. 480.
- https://www.youtube.com/watch?v=V0Pk_dPU2IY&t=4228s
- https://youtube.com/playlist=PLZoTAELRMXVNvTfHyJxPRcQkpV8ubBwHo&si=Nrrug5F3zotxGSkZ
- https://www.kaggle.com/datasets/debashishsau/aslamerican-sign-language-aplhabet-dataset


Yashi Sharma (58)



Harsh Yadav (20)

Purnima Kumar (41)