Face Mask Detection Using InceptionV3

Mohona Mazumdar 40129421
Github Link:
https://github.com/mohona6646/COMP478_Project

***Abstract*– Face masks have become a household essential recently, with disposable masks now available for free in many public places. Less than four years ago, face masks were considered a foreign item in our society. Today, in several countries, masks remain mandatory to step foot in some locations. Not wearing one meant giving up your access to leisurely places such as movie theaters and restaurants. Many people viewed this requirement as a violation of their freedom and chose not to wear masks. This retaliation was cause for problems because it risked the safety of the immunocompromised people in our society. This is why there is a need for a face mask detection and alert system that uses image-processing techniques, particularly, deep learning algorithms. In this paper, Inception V3, a conventional neural network is used to classify multiple images of unmasked and masked individuals, the two categories of the dataset. The evaluation metrics of the model are also presented to demonstrate the performance of the classifier.**

I. INTRODUCTION

Heavily shaken by the Covid-19 pandemic, the authorities were quick on their feet to put in place safety measures to decline the spread of the virus. Given the high transmissibility of the coronavirus via airborne viral particles, wearing masks in everyday life has been highly encouraged as a preventive measure to limit the spread of the virus among individuals. To ensure compliance with mask-wearing mandates, public places such as grocery stores, movie theaters, and restaurants have been hiring additional staff members to monitor customers' adherence to the face mask policy before allowing them to enter. However, with the number of clients coming in and out of the store, the supervision of masks becomes quickly challenging to the employee. Not only is the worker responsible for verifying that the client possesses a face mask, but they also need to ensure that the mask is being worn correctly at all times in accordance with the safety guidelines. As a consequence, many individuals enter stores without wearing a mask or with a poorly-fitted mask, increasing the risk of spreading the virus among other people.

This is where image processing comes into play. By training deep learning models to recognize patterns within thousands of images, the algorithm can accurately distinguish between masked and unmasked individuals, allowing for a more efficient regulation of safety measures surrounding face masks. The implementation of such a system brings many benefits, mainly reducing manual work from the government and employees and protecting the public's health more effectively. Even to this day, with the slow rate of coronavirus cases, wearing a face mask in certain situations is mandatory which is why the deep learning system will be of great help to our society. Technology has surpassed human ability in every aspect making the systems more reliable to perform repetitive work for longer periods of

time. With technology advancing faster than ever before, humans must learn to exploit its capabilities to improve the quality of life and stay ahead of the curve.

This project involves the implementation of a deep learning algorithm that will allow the computer to detect individuals who are wearing masks and those who are not. To do so, the deep learning network used for the system will be keras which is an open-source library that provides a python interface for artificial neural networks and acts as an interface for the Tensorflow library. This high-level network API is considered to be very user-friendly since it is built using python. Using Keras as the frontend for the project, Tensorflow, a popular open-source framework for building and training machine learning models is used as the backend to build the deep learning model. Regarding deep learning models, the neural network that is used to classify images through image segmentation is Inception V3, a convolutional neural network that is 48 layers deep.

To accurately differentiate between individuals wearing and not wearing masks, the neural network will require a vast dataset for training. The necessary dataset has been sourced from Kaggle and contains a large number of labeled images of both masked and unmasked individuals.

Rather than building a new neural network model from scratch, the project will utilize a pre-trained model for transfer learning. By doing so, we can take advantage of the knowledge already gained from the pre-trained model and transfer it to our new project, which is particularly useful when the dataset is relatively small or when training time is limited. This approach can save us a significant amount of time and computational resources, as we can skip the time-consuming process of training a model from the ground up. In this case, we will use a convolutional neural network, which is particularly well-suited for image recognition tasks. By leveraging the pre-trained features of the network, we can quickly adapt it to our specific use case and reduce the number of images required to train the network effectively. This approach has been widely adopted in various deep learning applications and has proven to be an effective way to achieve high accuracy while minimizing the amount of training required.

This paper will discuss the image processing steps involved in developing mask detection software, specifically to provide insights into how image processing techniques can be applied to real-world problems such as the COVID-19 pandemic.

II. RELATED WORK

In response to the pandemic's widespread impact on healthcare and daily life, many professionals have focused on developing ways to mitigate the spread of COVID-19. Image processing, specifically face mask detection, has emerged as a popular approach.

The objective of the paper “Single Camera Masked Face Identification” was to create an identification system that could recognize the identity of a person wearing a mask using a pre-stored database [1]. Two methodologies were used in their research, the first being YOLOv3 and the second being YOLO-face or RetinaFace.

For their first approach, YOLOv3 was trained using a custom dataset. This method was designed for the recognition of specific individuals. YOLOv3 is the third and improved version of YOLO which includes several advanced features such as logistic regression and multi-label class predictions.

For their second method, they used a two step approach where the faces were first localized using either YOLO-face or RetinaFace. Next, the VGGFace2 and ResNet-50 neural network architectures were used to calculate the facial vectors of the individuals.

Finally, their paper concluded that RetinaFace outperforms other algorithms in performance and preciseness, however the suggested future work to be focused on YOLOv4, which is the fourth version of the YOLO algorithm, since it offers a perfect balance of speed and accuracy.

III. PROPOSED METHODOLOGY

The detection of proper mask usage requires the model to be trained with a dataset, which in this case was obtained from Kaggle.The data is then preprocessed to clean it and divide it into training and testing datasets. The training dataset trains the model to create a fully trained model, while the testing dataset tests the model to create a tested model. Once the model is fully trained and tested, it can autonomously detect if a person is wearing a mask or not. To ensure the system is working accurately, a validation dataset is used to validate its performance.

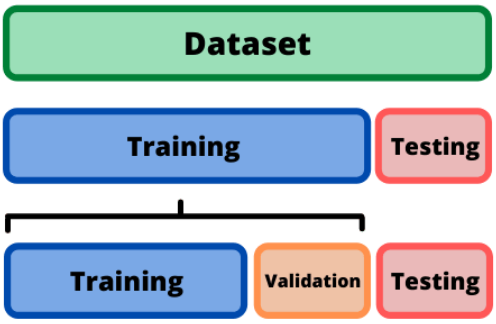

Figure 2. Training and Test split [2]

A. *Dataset*

The Kaggle dataset utilized for this project is composed of three primary directories - Train, Test, and Validation [3]. Each of these directories has two subdirectories, one containing images of people wearing masks and the other with images of individuals without masks. The Train directory consists of a total of 600 images, out of which 300 images depict individuals not wearing masks, while the remaining 300 images show people wearing masks. The test directory also has two files, “mask” and “non mask” where each of the subdirectories hold 50 images. Lastly, the validation folder comprises a total of 306 images with 153 images for both classes. Therefore, the dataset includes a total of 1006 images. The figure below displays a sample of the dataset showcasing people with and without a face mask.

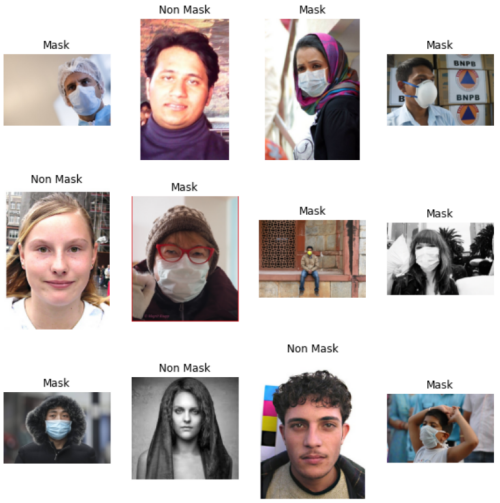

Figure 3. Sample of the dataset

B. *Processing*

To optimize the quality of the training set, several preprocessing and data augmentation techniques were utilized on the dataset.

Image augmentation techniques such as RandomFlip, RandomRotation, and

RandomZoomwere used to generate new training images from existing ones, improve the Inception V3 model's performance and reduce overfitting. This is done by applying random transformations to the original images, such as flipping, rotating, or zooming in and out, which creates a more diverse set of images for the model to train on.

Figure 4: Original image and its data augmented versions [4]

Moreover, the images are also resized and rescaled using the Sequential module from Keras.The resizing layer resizes the images to a fixed size of 224x224 pixels, which is the input size expected by the InceptionV3 model. The rescaling layer then rescales the pixel values from the range of 0-255 to the range of 0-1. This normalization step can help the model to converge faster and more reliably during training. By resizing and rescaling the images before they are fed into the model, the code ensures that all images are the same size and have pixel values in the same range.

Furthermore, the images are preprocessed using the image_dataset_from_directory function from TensorFlow's preprocessing module, which converts the images to tensor format and applies various preprocessing operations such as normalization, resizing, and batching.

C. *Dependencies*

The project utilizes Keras, a machine learning framework built on top of Tensorflow. Tensorflow, an open source platform developed by Google, is widely recognized as the most popular machine learning framework today and is used by many large companies. Keras is designed to be scalable, capable of running on large clusters of GPUs, and is also known for its industry strength. It allows for easy and fast prototyping of deep learning models by providing a wide range of pre-built layers, activation functions, loss functions, and optimizers.With its low-level flexibility, Keras is ideal for research implementations, but it also includes high-level convenience features to speed up the development process.

D. Deep Learning Model Architecture

A convolutional neural network (CNNs) utilizes learned features to apply 2D convolutional layers to input data. Typically, a CNN comprises two primary parts. The first part is a convolution/pooling mechanism, which dissects the image into features for further analysis. The second part is a fully connected layer, which takes the output of the previous stage to predict the most appropriate label for describing the image.

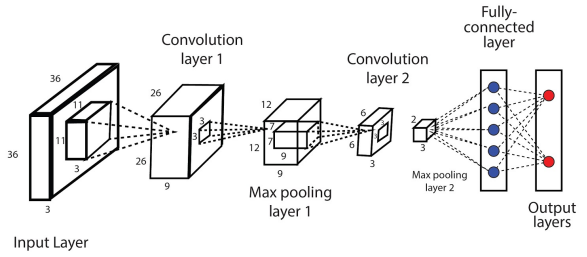

Figure 5. CNN Architecture [5]

In a Convolutional Neural Network (CNN), the two primary types of layers are the feature learning layers and classification layers.

The feature learning layer is responsible for transforming the input data (in this case, images) into a set of learned features that can be used by the network to make predictions. This layer performs a series of operations on the input data, such as convolution and pooling, in order to extract and learn different features present in the images.

On the other hand, the classification layer is responsible for using the learned features to classify the input images into different categories. This layer takes the output of the feature learning layer as input and maps it to a probability distribution over the different classes that the model is trained to recognize. Once the classification layer has made a prediction, the output can be interpreted as the probability that the input image belongs to a particular class.

In this project, the CNN architecture used is the Inception V3, which employs its convolutional layers to extract features from input images to classify them in the two classes: mask or non mask. One of the key features of Inception V3 is its use of an inception module, which is a combination of multiple convolutional layers with different filter sizes. This allows the network to extract features at multiple scales and resolutions to capture both fine and coarse details of an image.

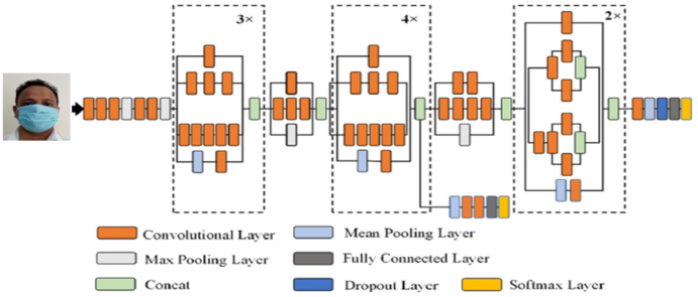

Figure 6. Inception V3 Architecture [6]

To be able to classify the images accurately, the Inception V3 model was constructed using the following functions.

- **Dropout**: The purpose of dropout is to prevent overfitting by randomly dropping out (setting to zero) a certain percentage of the neurons in the network during training.

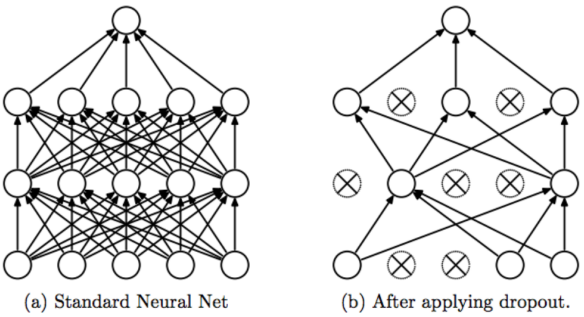

Figure 7. Network before and after Dropout[7]

- **ReLU Activation**: Rectified Linear Units (ReLU), an activation function that introduces non-linearity into the model and allows it to learn more complex features.
- **Softmax**: Function that converts the output of the network into a probability distribution over the possible classes

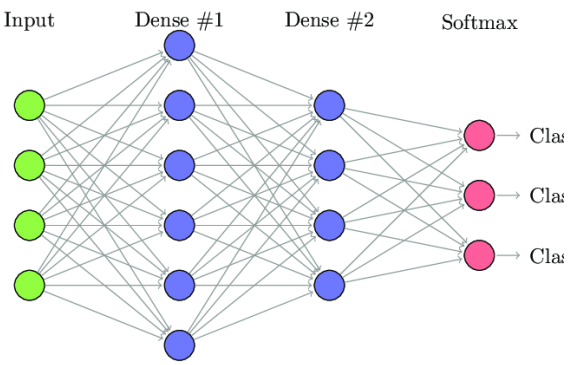

Figure 8. Softmax Layer [8]

- **Max Pooling**: This function reduces the spatial size of the feature maps by selecting the maximum value from each local region. This reduces the computational complexity of the model and helps prevent overfitting.

IV. EXPERIMENT AND RESULTS

1. **System Environment:**

The model was trained locally on a system with the following specifications:

- **CPU**: Intel(R) Core(TM) i5-8250U CPU @ 1.60GHz 1.80 GHz
- **RAM**: 8.00 GB

2. **Hyperparamters**:

As for the hyperparameters used to determine how the image processing algorithm will operate, a few of them are mentioned below:

- **Dropout**: 0.5 - the fraction of the input units to drop during training
- **Dense**: 64 - the number of neurons in the hidden layer
- **Dense**: 2 - the number of output classes
- **Learning rate** (LR) = 0.001 - the learning rate used by the Adam optimizer to update the weights during training
- **Loss**="categorical_crossentropy" - the loss function used for training the model
- **BATCH_SIZE** = 32 - the number of samples per gradient update
- **IMAGE_SIZE** = (224, 224) - the size of the image after resizing
- **SEED** = 0 - This sets the random seed for TensorFlow and NumPy to ensure reproducibility of results.
- **VAL_SPLIT** = 0.5**:** This is the fraction of the data used for validation during training.

3. **Evaluation Criteria**:

Evaluation is a critical step in building image processing models to ensure that the model is performing well on unseen data. There are several evaluation metrics used for measuring the performance of image processing models, including accuracy, precision, recall, and F1 score.

Evaluation metrics such as accuracy, precision, recall, and F1 score use the counts of true positives/negatives and false positives/negatives to measure the performance of image processing models.

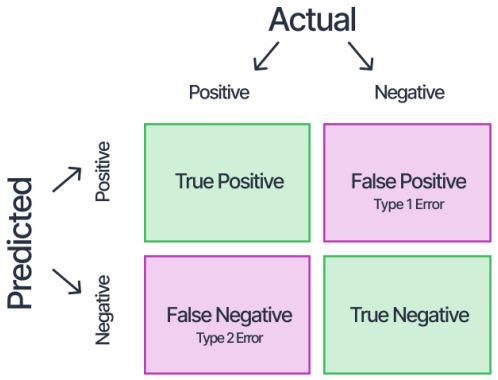

Figure 9. Confusion Matrix [9]

- **Accuracy**

Accuracy measures the proportion of correct predictions for the given test data. This can be computed by dividing the number of accurate predictions by the total number of predictions made.

The equation for accuracy is shown below:

$$\text{Accuracy} = \frac{\text{True Positives} + \text{True Negatives}}{\text{All Samples}}$$

- **Precision**

Precision is a measure of how accurate the positive predictions are and is defined as the ratio of true positives to the sum of true positives and false positives.

As for the precision, it is defined through the mathematical equation provided below:

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}}$$

- **Recall**

Recall, also known as the true positive rate of the model, is a performance metric that calculates the fraction of true positive examples that were correctly identified by the model out of all the examples that truly belong to a certain class. [10]

Mathematically, recall is defined as follows:

$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}}$$

- **F1-Score**

The F1 score is a single metric that combines both precision and recall by taking their harmonic mean. It is calculated as the harmonic mean of precision and recall, and ranges from 0 to 1, with a higher score indicating better performance. The higher the F1 score, the better the performance of the classification model.

$$\text{F1 Score} = \frac{2 \times (\text{Precision} \times \text{Recall})}{\text{Precision} + \text{Recall}}$$

4. **Results:**

When the training loss and accuracy graph is plotted, the x-axis represents the number of epochs (training iterations) and the y-axis represents the loss (or error) and accuracy metrics.

Typically, as the number of epochs increases, the model's accuracy improves, while the loss decreases.The training loss curve shows how well the model is fitting the training data, and it should ideally decrease steadily over time, indicating that the model is learning and fitting the data better.

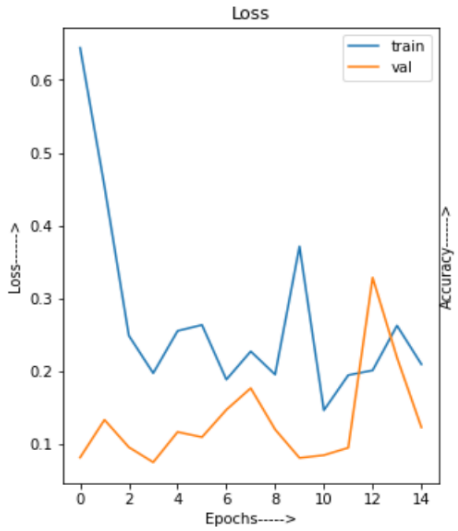

Figure 7. Training loss over epochs

In this case, it is clear that the training loss decreases as the number of iterations increases. At the very start of the image processing (epoch 0), the model has a training loss of over 0.6. However, at the second epoch, the training loss decreases significantly, approximately 0.24, indicating that the model is effectively learning and fitting the data better.

On the other hand, the training accuracy curve shows how well the model is predicting the training data. Ideally, the accuracy should increase steadily over time, indicating that the model is improving its ability to predict the correct output.

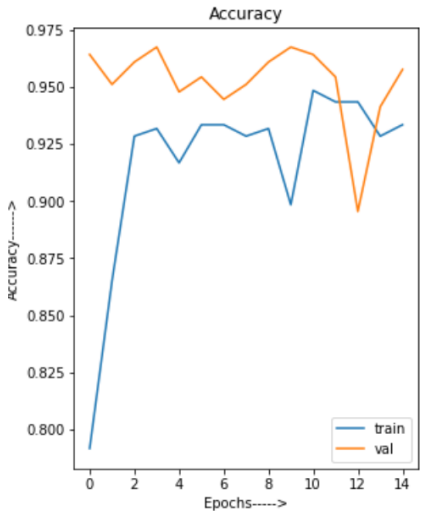

Figure 8. Model accuracy over epochs

For this project, it can be seen that the accuracy of the model increases over epochs suggesting that the model is getting better at predicting the right label for each image. In fact, the accuracy at epoch 0 for the training dataset is reported to be below 0.8. However, starting from the second epoch, the accuracy of the model grows significantly, reaching accuracy values of 0.95 or 95%.

Specifically, the accuracy of our model turns out to be 97.72%, signifying a high level of performance and reliability. Nevertheless, relying solely on accuracy is insufficient since other evaluation metrics provide more insight into the model's performance. Thus, the model's precision, recall, and f1-score are summarized in the following classification report and confusion matrix:

	precision	recall	f1-score	support
Mask	0.96	1.00	0.98	50
Non Mask	1.00	0.96	0.98	50
accuracy			0.98	100
macro avg	0.98	0.98	0.98	100
weighted avg	0.98	0.98	0.98	100

Figure 9. Classification Report

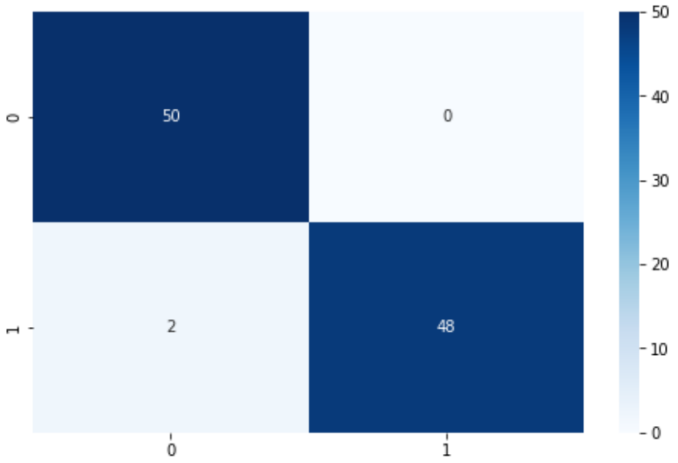

Figure 10. Confusion Matrix

The test dataset for this project consisted of 50 images per class, namely mask and non-mask. Based on the confusion matrix presented here, it can be inferred that the model performed well in correctly predicting the labels for each class, as a majority of the data points fall under the true mask and true non-mask categories. The model demonstrated remarkable performance as it accurately predicted the label of all individuals wearing masks and misclassified only two individuals who were unmasked in the given test dataset.

V. CONCLUSION & FUTURE WORK

In summary, the use of Inception V3 convolutional neural network with the Kaggle dataset and Keras has yielded an accuracy rate of 97.72%, demonstrating an effective and reliable solution with excellent performance.

The results of this project are valuable since the high accuracy rate allows us to consider implementing the system on indoor cameras to ensure proper mask-wearing, promote health, and reduce the burden on the healthcare system during the pandemic. It is clear that this system can raise awareness and encourage mask-wearing in public spaces.

The success of this project serves as evidence of the potential of machine learning techniques in tackling real-world problems, emphasizing the significance of ongoing research and development in this field.

VI. REFERENCES

[1] V. Aswal, O. Tupe, S. Shaikh and N. N. Charniya, "Single Camera Masked Face Identification," 2020 19th IEEE International Conference on Machine Learning and Applications (ICMLA), Miami, FL, USA, 2020, pp. 57-60, doi: 10.1109/ICMLA51294.2020.00018.

[2] S.D.S. Team, Ed., "How to Train and Test Data Like a Pro" *SDS Club* [Online]. Available: https://sdsclub.com/how-to-train-and-test-data-like-a-pro/?fbclid=IwAR1_LYaRJUDETdqe4wLv6mw55YGf5SRyfOfMMgnsoBai0hXyXn6MgUXCbWE [Accessed: 27-Apr-2023].

[3] P. Mitra, "Covid Face Mask Detection Dataset," *Kaggle*, 15-Jul-2020. [Online]. Available: https://www.kaggle.com/datasets/prithwirajmitra/covid-face-mask-detection-dataset [Accessed: 22-Apr-2023].

[4] A. Singh, "What is Image Data Augmentation: Data Science and Machine Learning," *Kaggle*, 2019. [Online]. Available: https://www.kaggle.com/getting-started/190280 [Accessed: 26-Apr-2023]

[5] R. Dwivedi, "5 common architectures in Convolution Neural Networks (CNN)," *Analytics Steps*, 05-Jul-2021. [Online]. Available: https://www.analyticssteps.com/blogs/common-architectures-convolution-neural-networks [Accessed: 29-Apr-2023].

[6] R. K. Bania, "Ensemble of Deep Transfer Learning Models for real-time automatic detection of face mask," *SpringerLink*, 01-Feb-2023. [Online]. Available: https://link.springer.com/article/10.1007/s11042-023-14408-y [Accessed: 30-Apr-2023].

[7] Pragati, "How to deactivate dropout layers while evaluation and prediction mode in Keras?," *For Machine Learning*, 13-Dec-2021. [Online]. Available: https://androidkt.com/deactivate-dropout-layers-while-evaluation-prediction-mode-keras/ [Accessed: 29-Apr-2023].

[8] I. Kouretas and V. Paliouras, "[PDF] hardware implementation of a Softmax-like function for deep learning: Semantic scholar," *Technologies*, 28-Aug-2020. [Online]. Available: https://www.semanticscholar.org/paper/Hardware-Implementation-of-a-Softmax-Like-Function-Kouretas-Paliouras/933397743ef4617dc035dd12080a8dbd640dc99f. [Accessed: 30-Apr-2023].

[9] R. Kundu, "Confusion Matrix: How To Use It & Interpret Results [Examples]," *www.v7labs.com*, Sep. 13, 2022. https://www.v7labs.com/blog/confusion-matrix-guide [Accessed Apr. 28, 2023].

[10]"What is recall," Iguazio, 13-Dec-2022. [Online]. Available: https://www.iguazio.com/glossary/recall/ [Accessed: 30-Apr-2023].