Deep Learning

SF MASK - Face recognition

NEFZI Wacim, PRUDHOMME Pierre

wacim.nefzi@ensiie.eu

pierre.prudhomme@ensiie.eu

École nationale supérieure d'informatique pour l'industrie et l'entreprise

Abstract

This paper explores the development of a face mask detection system leveraging transfer learning with pre-trained architectures such as MobileNetV2 and EfficientNet. The work highlights the challenges of optimizing for both accuracy and computational efficiency, particularly in real-time scenarios on embedded devices. Through experiments with hyperparameter tuning and data augmentation strategies, the study identifies key trade-offs and presents insights for practical implementation. Results show that MobileNetV2 outperforms in speed, while EfficientNet achieves superior accuracy.

1 Introduction

This project aims to design a **face mask recognition system** leveraging advanced deep learning techniques. This project is a pedagogical endeavor aimed at exploring the technical aspects of face mask detection using deep learning. It is not intended to infringe on individual privacy or freedoms, nor to promote state surveillance practices that could lead to overreach or misuse, as seen in certain authoritarian regimes.

The project is divided into two main phases:

- **Image-based Classification :** Classify static images into two categories: *masked faces* and *unmasked faces*. This phase involves optimizing model performance, particularly in handling challenges like varying lighting, angles, and image quality.
- **Real-time Detection:** Extend the solution to **real-time face mask detection on video streams**. This requires integrating computer vision tools, such as OpenCV, and optimizing for dynamic environments. And adapt it for embedded platforms such as the JETSON Nano (already owned), while optimizing resource constraints.

The primary goal is to utilize transfer learning with pre-trained models like *EfficientNet* and *MobileNetV2*, which are known for their balance between accuracy and computational efficiency.

2 Dataset

The SF-Mask dataset contains a total of 45,826 images of individuals. It is divided into two main categories: '*compliant*' and '*non-compliant*'. The '*compliant*' category consists of images of individuals wearing masks, while the '*non-compliant*' category includes images of individuals without masks.

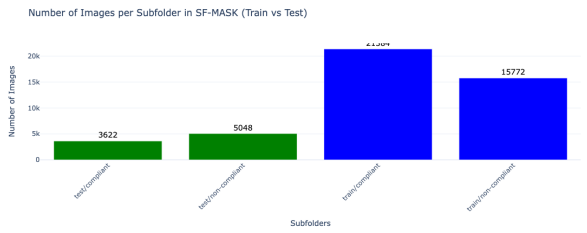


Figure 1: Number of images per folder for *SF-MASK*.

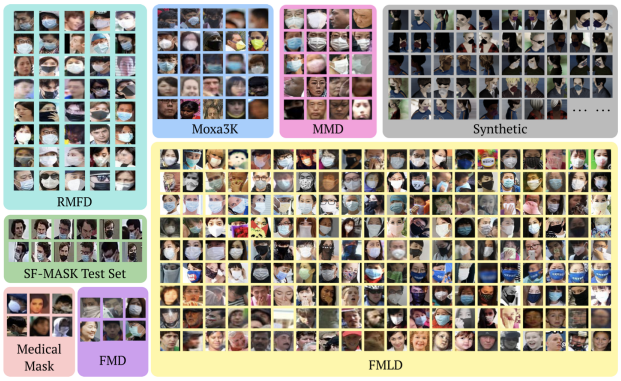


Figure 2: Composition of SF-MASK dataset using a mix of other datasets.

Figure 3: Comparison of visual data in SF-MASK dataset.

This dataset consists of **several sub-datasets**, including *RMFRD*, *FMLD*, *Moxa3K*, and others. Additionally, it includes synthesized faces created specifically for this purpose. We explored how other datasets are structured to better understand the approaches used in their development.

There will be a few challenges to address with this dataset. First, we need to reshape the images to work with them without compromising their quality. This will involve **a trade-off between maintaining image quality and ensuring uniform dimensions**. To address this, we can start by excluding images that are too small. Then, we can resize the remaining images to a consistent size. On this processed subset, we can apply augmentations such as brightness adjustments, rotations, and even noise addition.

The *RMFRD* dataset contains 400 subfolders of individuals with a few dozen images each time, and for each individual, we have a few test images. This dataset construction is very different from our dataset, and even though it's one of the most used for this type of problem, it would require too much computation time and resources to work because it deals with people one by one.
We also chose the *SF-Mask* dataset due to its low-resolution images. Specifically, this dataset consists of images ranging from 7x7 pixels to 64x64 pixels. To better understand the distribution, we created a graphic to visualize how many images belong to each pixel resolution category.

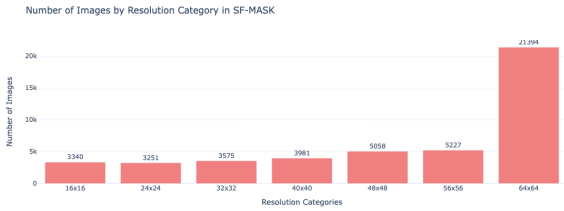


Figure 4: Distribution of image sizes in SF-MASK.

3 State of Art

Face mask detection, particularly using deep learning models, has gained attention due to its importance in public health surveillance. Recent advancements leverage datasets like SF-MASK, which contains over 20,000 labeled images to train robust face mask detection models.

Approaches

- **ANN and CNN:** Artificial Neural Networks (ANNs) and Convolutional Neural Networks (CNNs) are widely used for binary and multi-class classification tasks in face mask detection.
- **Transfer Learning:** Pre-trained models such as MobileNetV2, fine-tuned on datasets like SF-MASK, achieve high accuracy with reduced computational costs.

Caffe-MobileNetV2 for Photo and Video Recognition Recent studies have successfully applied Caffe-MobileNetV2 for both static images and real-time video, demonstrating its efficiency for edge devices without sacrificing detection performance. These models excel in both accuracy and speed, making them ideal for real-time applications in surveillance and mobile settings.

References

1. Howard, A. G., et al. *MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications.* arXiv preprint, 2017.
2. Tan, M., & Le, Q. *EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks.* ICML, 2019.
3. Loey, M., et al. *A hybrid deep transfer learning model for face mask detection.* Measurement, 2021.
4. Jiang, M., et al. *Real-time Face Mask Detection Using CNNs.* IEEE, 2020.

4 Approach & Methods

The goal of this project is to identify the most suitable model for real-time face mask detection on embedded systems like the JETSON Nano. We prioritize models that balance **efficiency** and **accuracy** while maintaining compatibility with resource-constrained environments. Our approach involves selecting and fine-tuning pre-trained architectures to adapt them to a binary classification task, specifically detecting whether a mask is present or not.

4.1 Model Selection

Two pre-trained architectures, **MobileNetV2** and **EfficientNet**, were chosen for their complementary strengths. Table 1 summarizes their key characteristics.

Aspect	**MobileNetV2**	**EfficientNet**
Designed for	Mobile/embedded	General tasks
Computational efficiency	High	Moderate
Suitability for JETSON Nano	Excellent	Acceptable
Key technique	Depthwise separable convolutions	Compound scaling

Table 1: Comparison of MobileNetV2 and EfficientNet.

4.2 Fine-Tuning Approach

Fine-tuning leverages the pre-trained weights of a model, which capture general features (e.g., edges, textures) from a large dataset like ImageNet, and adapts the higher layers for a specific task. This method is particularly effective for tasks with limited labeled data, reducing both computational cost and the risk of overfitting.

In our case, the final fully connected layers of both architectures were replaced to align with the binary classification task:

- **MobileNetV2:**

```
model.classifier[1] = nn.Linear(model.last_channel, num_classes)
```

- **EfficientNet:**

```
model.classifier[1] = nn.Linear(model.classifier[1].in_features, num_classes)
```

This approach retains the general feature extraction capabilities of the pre-trained model while allowing the final layers to specialize in distinguishing between the two target classes.

Why Fine-Tune?

Fine-tuning is particularly effective when the dataset is small or similar to the pre-training dataset. The earlier layers of the model extract generic features (e.g., edges and textures), while the later layers are more task-specific. By retraining only the final layers, we reduce computational cost and risk of overfitting.

4.3 Training Approach and Anticipated Challenges

To ensure robust model generalization, we adopt a comprehensive approach combining **cross-validation** and the use of a separate test set. Cross-validation helps optimize hyperparameters and **reduces the risk of overfitting** by training and validating the model on different data splits. A dedicated test set evaluates the model's performance under real-world conditions, ensuring its applicability beyond the training dataset. To enhance interpretability, visual tools such as learning curves and confusion matrices will be generated with detailed legends and referenced throughout the text.

Key evaluation metrics include **accuracy**, which measures the proportion of correctly classified samples, and the **F1 Score**, which balances precision and recall. These metrics are especially critical in managing potential issues such as class imbalance, where the **recall** metric ensures minority classes are appropriately represented.

Anticipating challenges, we address **class imbalance** through data augmentation techniques and weighted loss functions to ensure fair representation of all classes during training. **Overfitting** is mitigated using regularization techniques such as **early stopping**, and careful **tuning of model parameters**. For **hardware limitations**, resource-efficient models like MobileNetV2 are selected to ensure compatibility with devices such as the JETSON Nano. By integrating these strategies, our approach is designed to deliver robust and efficient models suited for real-world deployment.

5 Data Preprocessing

Effective data preprocessing is a cornerstone of deep learning workflows, especially for complex tasks such as face mask detection. In this project, the preprocessing pipeline was designed to address challenges posed by varying lighting conditions, facial orientations, and diverse image resolutions. The objective was to create a robust dataset that enhances the model's generalization capabilities while maintaining computational efficiency.

Dataset Proportion Selection

Given the computational constraints and the exploratory nature of this project, we utilized only **10% of the total dataset** during training and validation. This proportion was chosen to reduce training time while ensuring the subset **remains representative of the dataset's diversity**. This strategy allowed for rapid experimentation and iterative refinement of the models.

Data Augmentation Strategies

To improve the model's robustness against real-world variations, extensive data augmentation was applied to the training set. The augmentation techniques were carefully chosen to simulate diverse environmental conditions and enhance feature learning. Specifically, the following transformations were employed:

- **Random Horizontal Flip:** Simulates variations in face orientation. 50% of images will be flipped horizontally.
  `transforms.RandomHorizontalFlip(p=0.5)`
- **Random Rotation:** Accounts for differences in camera angles.
  We opted for a light rotation. `transforms.RandomRotation(degrees=10)` .
- **Color Jitter:** Adjusts brightness, contrast, saturation, and hue to mimic varying lighting conditions. We opted for slight variations in colour.
  `transforms.ColorJitter(brightness=0.2, contrast=0.2, saturation=0.2)`
- **Random Perspective Distortion:** Introduces viewpoint variations through slight perspective changes. We opted for a moderate distortion.
  `transforms.RandomPerspective(distortion_scale=0.1, p=0.3)`
- **Random Affine Transformations:** Captures real-world image scaling and translation variability.

These augmentations were applied exclusively to the training dataset, preserving the integrity of the validation and test sets.

5.1 Image Resizing and Normalization

All images were resized to a fixed resolution of **224x224 pixels**, adhering to the input requirements of pre-trained models such as MobileNetV2 and EfficientNet. Resizing employed bilinear interpolation, balancing computational efficiency and image quality. Subsequently, pixel values were normalized using mean and standard deviation values of $[0.5, 0.5, 0.5]$. This normalization step centered the data and scaled it, accelerating convergence during training.

Figure 5: Effects of each change on the original image

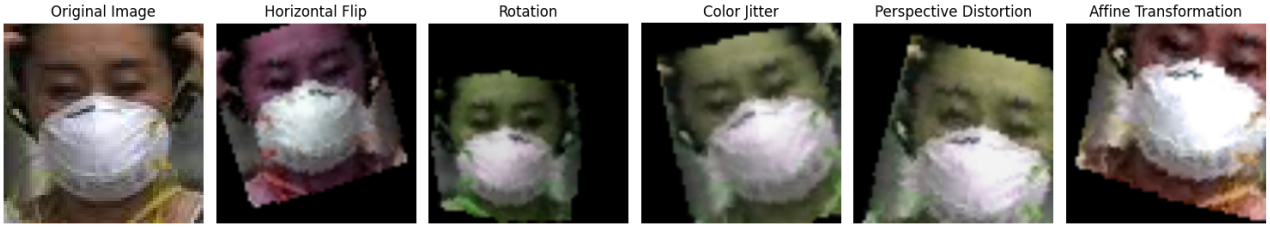


Figure 6: Combined effects of each change on the original image

Padding and Resizing for Uniformity

Images with extreme aspect ratios or small resolutions were adjusted using a combination of padding and resizing techniques. Padding was applied to maintain the original aspect ratios, followed by resizing to the target dimensions. This preprocessing step mitigates potential distortions and ensures uniformity across the dataset.

Batching and Data Loading

A batch size of **32** was chosen to balance GPU memory constraints and training speed. Data loaders were configured to shuffle the training set, ensuring diverse batches during training epochs. For validation and test sets, shuffling was disabled to maintain sample order, preserving reproducibility during evaluation.

Impact on Real-Time Detection

The preprocessing decisions are anticipated to significantly influence the performance of real-time mask detection systems:

- **Augmentation Effects:** By exposing the model to a wide range of augmented samples, its ability to generalize across varying lighting, orientations, and distortions is enhanced.
- **Uniform Resizing and Normalization:** These steps standardize the inputs, ensuring compatibility with pre-trained models and reducing computational overhead during inference.

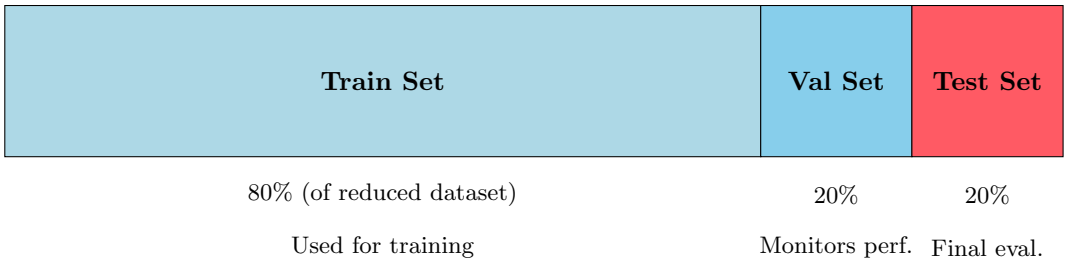
Anticipated Challenges

- **Augmentation Noise:** While augmentations improve robustness, excessive transformations may introduce unrealistic noise, potentially affecting model performance.
- **Inference Latency:** The preprocessing pipeline's resizing and normalization steps, although optimized for training, may slightly delay real-time video stream processing.

These preprocessing strategies align with the overarching goal of building an accurate and efficient face mask detection system for real-world applications.

5.2 Train-Validation-Test Split

The dataset was divided into three subsets:



The Test set was kept entirely separate to ensure unbiased evaluation of the final model. The stratified split ensures a balanced representation of both the *compliant* (masked) and *non-compliant* (unmasked) classes in all subsets.

5.3 Hyperparameter Tuning

Hyperparameter optimization targeted the learning rate (lr) and batch size (bs), with a grid search over:

- **Learning Rates:** {1e-2, 1e-3, 1e-4, 1e-5}.
- **Batch Sizes:** {16, 32, 64}.

Cross-validation with $k = 5$ folds ensured robustness.
Optimal values were identified as lr = 1e-4 and bs = 32.

6 Results and Analysis

6.1 Models Performance

Table 3 summarizes the performance metrics across train, validation, and test datasets.

Model	Dataset	Accuracy	F1 Score	Time
MobileNetV2	Train	96.1 ±2.9	96.1 ±2.9	**1755 s**
	Validation	94.2 ±1.2	94.2 ±1.2	
	Test	92.5	-	**0.2874** ±0.1511 **s**
EfficientNet	Train	**96.2** ±3.9	**96.2** ±3.9	2093 s
	Validation	**94.9** ±1.6	**94.9** ±1.6	
	Test	**94**	-	0.4639 ±0.1334 s

Table 2: Performance metrics for MobileNetV2 and EfficientNet.

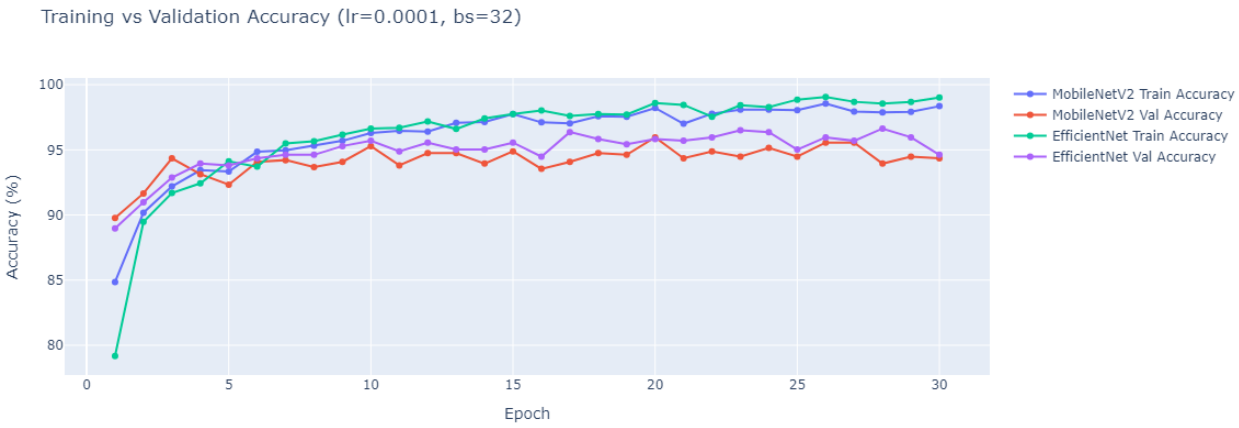


Figure 7: Training and validation accuracy/loss for both models.

7 Performance Analysis and Discussion

Table 3: Performance comparison of MobileNetV2 and EfficientNet-B0 on the test set.

Metric	MobileNetV2 (%)	EfficientNet-B0 (%)
Overall Accuracy	**93**	**92**
Compliant Class		
Precision	96	94
Recall	87	87
F1-Score	91	90
Non-Compliant Class		
Precision	91	91
Recall	97	96
F1-Score	94	94

The classification results for **MobileNetV2** and **EfficientNet-B0** are summarized in Table 3. Both models demonstrate excellent performance, with MobileNetV2 slightly outperforming EfficientNet-B0 in overall accuracy and F1-Score metrics for the *compliant* class. These results indicate that MobileNetV2 is better optimized

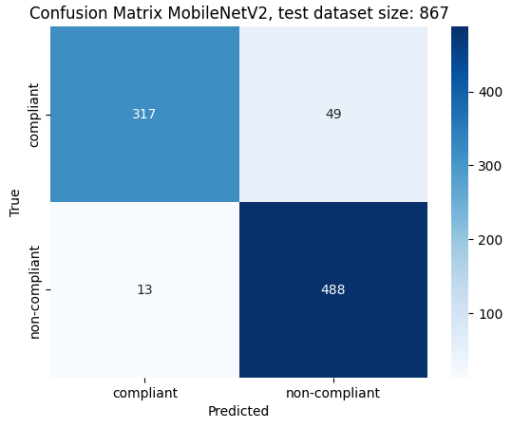


Figure 8: Confusion Matrix for Model 1 (MobileNetV2).

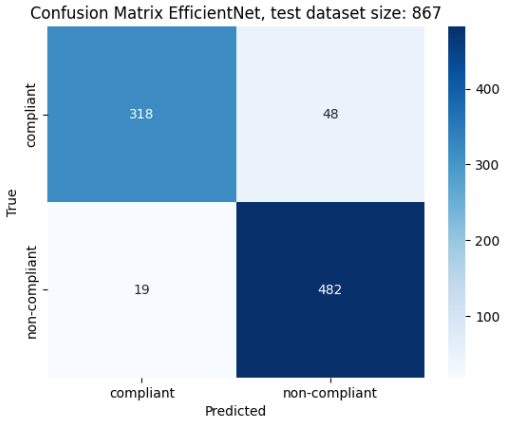


Figure 9: Confusion Matrix for Model 2 (EfficientNet).

Figure 10: Comparison of Confusion Matrices.

The Receiver Operating Characteristic (ROC) curves for **MobileNetV2** and **EfficientNet-B0** demonstrate exceptional classification performance, with both models achieving an **Area Under the Curve (AUC)** score of **0.98**. This indicates a 98% probability of accurately distinguishing between *compliant* and *non-compliant* mask categories.

The curves closely approach the top-left corner, reflecting **high sensitivity** and **low false positive rates**. The overlapping nature of the curves implies comparable performance between the architectures. While **MobileNetV2** excels in efficiency for deployment on resource-constrained devices, **EfficientNet-B0** demonstrates robust generalization due to its compound scaling. Further validation is recommended to ensure **real-world applicability**.

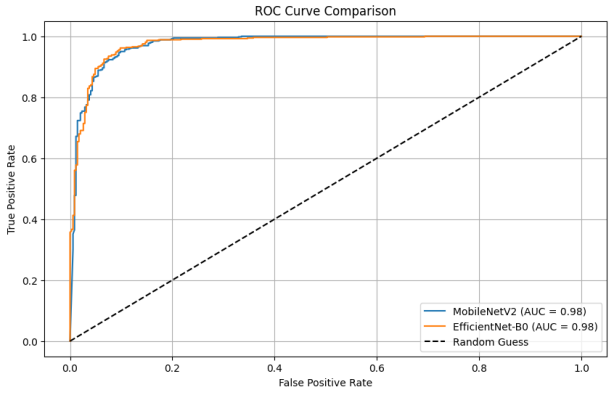


Figure 11: ROC curves for MobileNetV2 and EfficientNet-B0 with an AUC of 0.98.

for handling variations in the dataset, while EfficientNet-B0 remains competitive and robust for real-world applications.

7.1 Results on Test Set

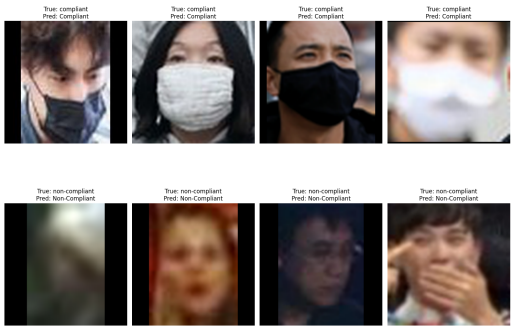


Figure 12: EfficientNet on Test Images

8 Impact of data augmentation

9 Discussion

Observations

Class Performance: Both models show higher performance on the *non-compliant class* (unmasked faces) compared to the *compliant class* (masked faces). This may be due to variations in mask types and occlusions caused by improper mask usage.

Model Comparison: MobileNetV2 achieves slightly better overall accuracy and F1-scores. Its higher precision for compliant faces indicates fewer false positives, making it a better choice for strict mask detection.

Improvements and Future Work

- **Data Quality:** - Ensuring clean and balanced datasets is critical. Low-quality or mislabeled images negatively impact model performance (garbage in, garbage out). - Synthetic data augmentation could improve diversity, especially for the compliant class.
- **Advanced Augmentation:** Techniques such as random erasing, CutMix, or MixUp can improve robustness, particularly for underrepresented scenarios.
- **Model Optimization:** - Investigate MobileNetV3 for potential accuracy gains. - Explore EfficientNet-B1 or B2 for better accuracy while balancing computational costs.

By addressing these improvements, the models can achieve better generalization and robustness, ensuring reliable performance in real-world applications.

10 Concrete Application

In this project, we aim to tackle real-world challenges related to mask detection by applying advanced machine learning techniques. The system is designed to process images and videos to identify whether individuals are wearing masks or not, providing actionable insights for public health and safety measures.

10.1 Street Mask Detection

The street mask detection module is designed to process static images taken in crowded public areas, such as streets or transportation hubs. Leveraging modern face detection techniques, the system extracts facial regions from images and classifies each face based on mask compliance. For this purpose, a pre-trained EfficientNet model has been fine-tuned to distinguish between "Mask" and "No Mask" labels with high accuracy.

The detection process begins with face localization, achieved using advanced algorithms such as MTCNN or Haar Cascades, depending on the application context. With a proper model for face detection we could improve the number of face detected. Each detected face is preprocessed to match the input dimensions of the mask detection model. This involves resizing the image to 224×224, normalizing pixel values, and expanding the batch dimensions. Once classified, results are overlaid onto the original image, providing a visual representation of mask compliance in the scene.

This application finds its utility in monitoring compliance in crowded areas, aiding enforcement agencies or public health officials in ensuring adherence to mask-wearing regulations.

10.2 Video Application

The video mask detection module extends the capabilities of the system to real-time scenarios. Using a live camera feed, the system continuously detects faces in video frames and classifies each detected face for mask compliance. This module is optimized for performance, ensuring minimal latency while maintaining high accuracy.

Figure 13: Model Application on a street during Covid.

The video processing pipeline involves several steps. Each frame is captured and resized to a standard resolution for consistent processing. Faces are detected in each frame, and the corresponding regions are passed through the mask detection model. The system overlays bounding boxes and classification labels directly onto the video feed, providing immediate feedback.

In addition, to optimize computational resources, predictions are made at regular intervals, balancing real-time processing with model inference time. This makes the solution scalable for deployment in environments such as public transport stations, malls, or workplaces.

11 Conclusion and Future Work

This study successfully developed a face mask detection pipeline optimized for accuracy and efficiency. Future work includes deployment on the Jetson Nano, scalability testing, and extending detection to additional facial attributes.