Safecheck AI – Using Convolutional Neural Networks to Improve Safety at Warehouses

Introduction

In industrial environments, especially warehouses, maintaining safety compliance is paramount to ensuring worker well-being and avoiding costly workplace accidents. Proper personal protective equipment (PPE), such as safety vests and helmets, play a crucial role in safeguarding employees from potential hazards. Despite strict regulations, human error often leads to lapses in wearing safety gear, which can lead to injuries or worse.

Safecheck AI is a computer vision solution that seeks to address this issue by automatically detecting whether workers entering a warehouse are wearing the necessary PPE. The system uses real-time video feeds and convolutional neural networks (CNNs) to identify individuals without the required safety gear and alert management to take corrective action.

This report documents the development, methodology, and results of the Safecheck AI project and highlights the next steps for its future development.

Problem Definition

The primary problem that Safecheck AI addresses is the failure of workers to wear safety gear such as helmets and vests. This issue not only puts the individual at risk of injury but also jeopardizes the safety of other employees. In the context of warehouses, where heavy machinery and hazardous materials are often present, even small lapses in safety can lead to serious consequences.

By providing an automated detection system, Safecheck AI aims to minimize the human oversight required to ensure compliance with safety protocols, thereby reducing the likelihood of accidents and improving overall safety standards in the workplace.

Audience

The intended users of Safecheck AI are:

- *Safety Officers*: Responsible for monitoring and enforcing safety compliance.
- *Warehouse Managers*: Overseeing daily operations and ensuring the health and safety *of their workforce.*
- *Compliance Officers*: Ensuring that the warehouse meets safety standards and regulatory requirements.

Potential customers of this technology include large-scale logistics companies, manufacturing plants, and distribution centers where worker safety is of paramount importance.

Data Sources

The datasets for Safecheck AI were sourced from Kaggle, an online platform providing open-source data:

- Employee Wearing Safety Gear
- Safety Helmet and Reflective Jacket
- Negative Dataset (No Safety Gear)

These open-source datasets provided a diverse collection of images showing workers with and without safety gear, which allowed for the development of an accurate detection model. Open-source data like this, is crucial for projects like Safecheck AI, allowing developers to access diverse, labeled datasets that support model training and validation without the need for costly data collection.

About the Data

The datasets used consist of over 12,000 images, including both positive samples (workers wearing safety gear) and negative samples (workers without gear). The images were labeled accordingly and were provided in varying conditions such as different lighting, backgrounds, and positions.

Each dataset included a mix of helmet and vest combinations, as well as images without any gear, making it ideal for training a binary classifier to detect whether safety gear was present or absent.

Method

The methodology behind Safecheck AI revolves around training a CNN to classify whether a worker in a given image is wearing the appropriate safety gear. The project involves:

- Preprocessing and data augmentation of the images.
- Building and training a CNN to classify images.
- Evaluating the performance of the model through accuracy and validation metrics.
- Implementing a proof-of-concept application that uses the model to detect safety gear in real-time.

Data Exploration

Before training the model, a detailed exploration of the dataset was conducted. Some of the key findings were:

- *Class Distribution*: The train data had a positive ratio of roughly 2:1, while the test data had an approximately equal distribution between images with and without safety gear..
- *Image Quality*: The images varied in quality, resolution, and background complexity. This variety was beneficial for training a robust model that can generalize well.
- *Object Focus*: The images mainly concentrate on the worker’s torso and head, as these areas are essential for identifying vests and helmets. However, the dataset also includes several full-body images and instances with multiple workers. This was done intentionally to capture a wide variety of scenarios and improve the model’s ability to generalize across different cases.

Modeling

The algorithm used is a custom Convolutional Neural Network (CNN) with five convolutional layers followed by activation layers, max pooling, batch normalization, and dropout for regularization. The network was trained using the binary cross-entropy loss function and Adam optimizer.

The CNN architecture was designed to learn features such as edges, shapes, and patterns that distinguish safety gear (e.g., the reflective vest and helmet) from the rest of the image.

The model architecture can be found in Appendix 1.

Hyperparameters

Key hyperparameters used during the training process:

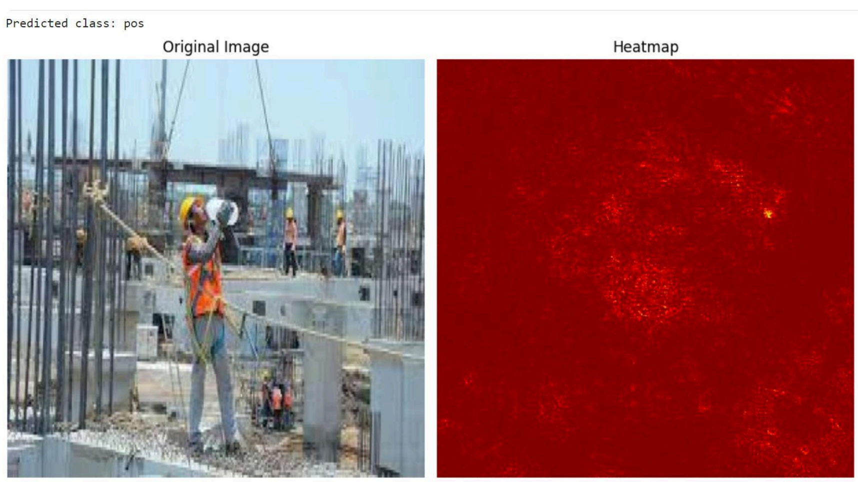
- *Learning Rate*: Set to 0.001 for optimized convergence.
- *Epochs*: The model was trained for 70 epochs to ensure adequate learning.
- *Batch Size*: A batch size of 32 was used to balance training speed and model performance.
- *Data Augmentation*: Techniques such as random rotation, brightness, and contrast adjustments were applied to enhance generalization.

Results

The model achieved a training accuracy of **93.79%**, with a validation accuracy of over **90%**. These metrics indicate that the model performs well in detecting the presence of safety gear under various conditions.

Heatmaps and Visualizations

We conducted further analysis using heatmaps to understand which parts of the image contributed most to the model's predictions. Below are some observations from the visualizations:



The model focuses heavily on the worker's vest, particularly the midsection, to determine if safety gear is being worn. This shows that the vest plays a significant role in classification.

In this image, the activation is mostly centered on the vest, with minor activation around the helmet. The plain background helps the model clearly distinguish the gear.

In this image, the vest remains the most influential factor in classification, but the more complex background in this image activates some pixels incorrectly. The edges of the painting in the background may have contributed to this.

Next Steps

Moving forward, Safecheck AI should focus on the following improvements:

- *Deployment*: The current proof-of-concept needs to be deployed in a real warehouse setting for live testing and feedback.
- *Re-Training*: The model can be re-trained using locally acquired images from the warehouse to better recognize environmental factors specific to the location.
- *False Positive Reduction*: Further refinement of the algorithm is needed to reduce the occurrence of false positives due to complex backgrounds.
- *Additional Features*: Future iterations may include multi-class classification to detect other forms of PPE beyond helmets and vests.

Appendix 1

Layer (type)	Output Shape
sequential (Sequential)	(None, 400, 400, 3)
conv2d_5 (Conv2D)	(None, 400, 400, 32)
activation_6 (Activation)	(None, 400, 400, 32)
batch_normalization_5 (BatchNormalization)	(None, 400, 400, 32)
max_pooling2d_5 (MaxPooling2D)	(None, 200, 200, 32)
dropout_6 (Dropout)	(None, 200, 200, 32)
conv2d_6 (Conv2D)	(None, 200, 200, 64)
activation_7 (Activation)	(None, 200, 200, 64)
batch_normalization_6 (BatchNormalization)	(None, 200, 200, 64)
max_pooling2d_6 (MaxPooling2D)	(None, 100, 100, 64)
dropout_7 (Dropout)	(None, 100, 100, 64)
conv2d_7 (Conv2D)	(None, 100, 100, 128)
activation_8 (Activation)	(None, 100, 100, 128)
batch_normalization_7 (BatchNormalization)	(None, 100, 100, 128)
max_pooling2d_7 (MaxPooling2D)	(None, 50, 50, 128)
dropout_8 (Dropout)	(None, 50, 50, 128)
conv2d_8 (Conv2D)	(None, 50, 50, 256)
activation_9 (Activation)	(None, 50, 50, 256)
batch_normalization_8 (BatchNormalization)	(None, 50, 50, 256)
max_pooling2d_8 (MaxPooling2D)	(None, 25, 25, 256)
dropout_9 (Dropout)	(None, 25, 25, 256)
conv2d_9 (Conv2D)	(None, 25, 25, 512)
activation_10 (Activation)	(None, 25, 25, 512)
batch_normalization_9 (BatchNormalization)	(None, 25, 25, 512)
max_pooling2d_9 (MaxPooling2D)	(None, 13, 13, 512)
dropout_10 (Dropout)	(None, 13, 13, 512)
flatten_1 (Flatten)	(None, 86528)
dense_2 (Dense)	(None, 2048)
activation_11 (Activation)	(None, 2048)
dropout_11 (Dropout)	(None, 2048)
dense_3 (Dense)	(None, 2)