Assignment-13

Name: Shukla Kushang Akshay

Email ID: kushangashukla1@gmail.com

Mobile No.: +91 8511350065
Enrollment No.: 221130107024
College Name: 113-Sal College of Engineering
Branch: Computer Engineering
Semester- 6th
Faculty Code: T00111- Mikin Dagli Sir

- ## Project Report:
  - Real-Time Anomaly Detection on Tiles using Teachable Machine & Streamlit
  - Drive Link for the Project: https://drive.google.com/drive/folders/1Ok3zuRPL0Wg_GUVmtUH-lhUGaqFz8skZ?usp=sharing

- ## Introduction:

  - This assignment demonstrates the creation of an **anomaly detection system for industrial tile surfaces** using a deep learning model trained via **Teachable Machine**, and deployed in a real-time interface using **Streamlit**. The model classifies tiles into categories like normal or defective (crack, glue strip, gray stroke, oil, rough tiles) by leveraging a TensorFlow SavedModel.

- ## Objective:
  - Train a classification model using **Teachable Machine** on various tile surface images.
  - Export the trained model as **TensorFlow SavedModel** compatible with Keras 3.
  - Build an **interactive Streamlit app** to:
    - Allow users to **upload tile images** for prediction.
    - Provide **real-time tile inspection using webcam**.

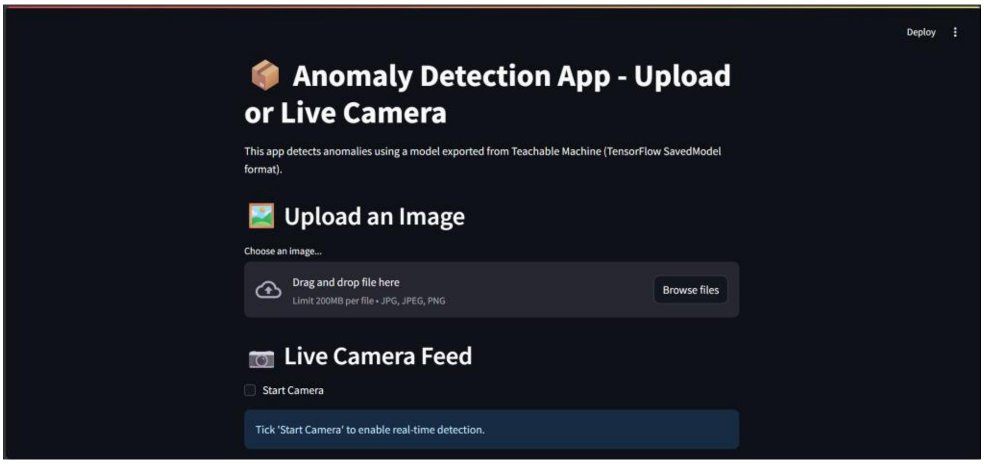
  - Achieve anomaly classification with confidence metrics shown on the image stream or uploaded image.

- ## Dataset Overview:
  - **Training Dataset (Teachable Machine):**

        - Class 1: Normal Tile
        - Class 2–6 (Anomalies):
            - Crack
            - Glue Strip
            - Gray Stroke
            - Oil Stain
            - Rough Surface
    - **Test Dataset:**
        - A mix of unseen examples from all classes used for performance evaluation in the deployed app.

- **Model Architecture & Export:**
    - **Model Training:**
        - Trained using Google Teachable Machine (Image Project).
        - Image resolution auto-handled at 224x224.
        - Data augmented internally.

    - **Model Export:**
        - Exported as TensorFlow SavedModel format.
        - Class names exported separately in labels.txt.
        - Structure:
            - /saved_model/
            - ├── saved_model.pb
            - └── variables/
            - ├── variables.index
            - └── variables.data-00000-of-00001

- **Deployment with Streamlit:**
    - **Technologies Used:**
        - Streamlit (Frontend + UI)
            - 

📦 Anomaly Detection App - Upload or Live Camera

This app detects anomalies using a model exported from Teachable Machine (TensorFlow SavedModel format).

🖼️ Upload an Image

Choose an image...

Drag and drop file here

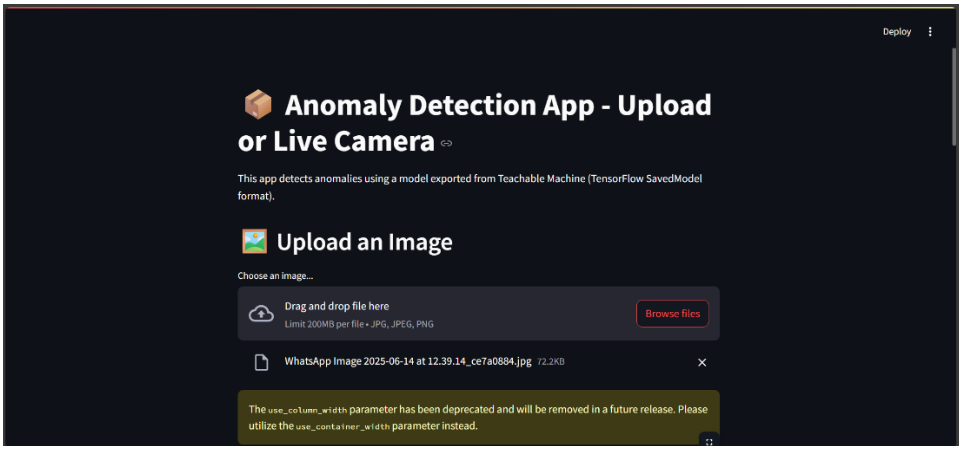
Limit 200MB per file • JPG, JPEG, PNG

📷 Live Camera Feed

□ Start Camera

Tick 'Start Camera' to enable real-time detection.

- TensorFlow TFSMLayer (Model loading)

➢ 

📦 Anomaly Detection App - Upload or Live Camera

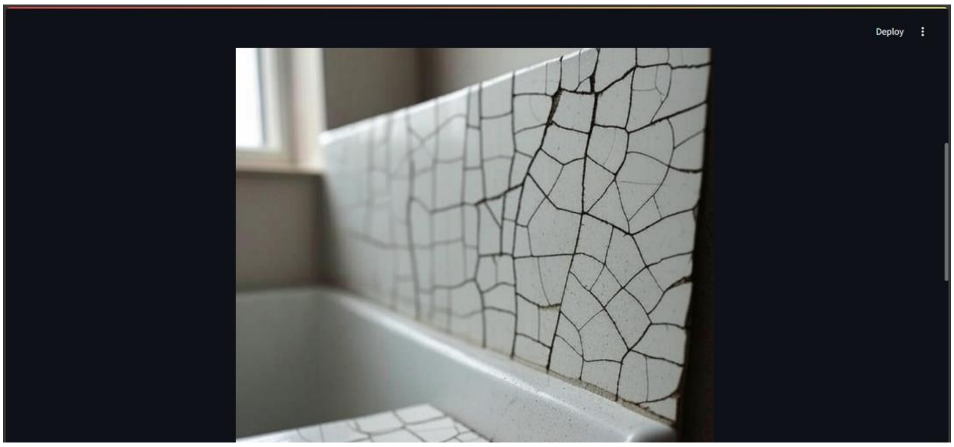
This app detects anomalies using a model exported from Teachable Machine (TensorFlow SavedModel format).

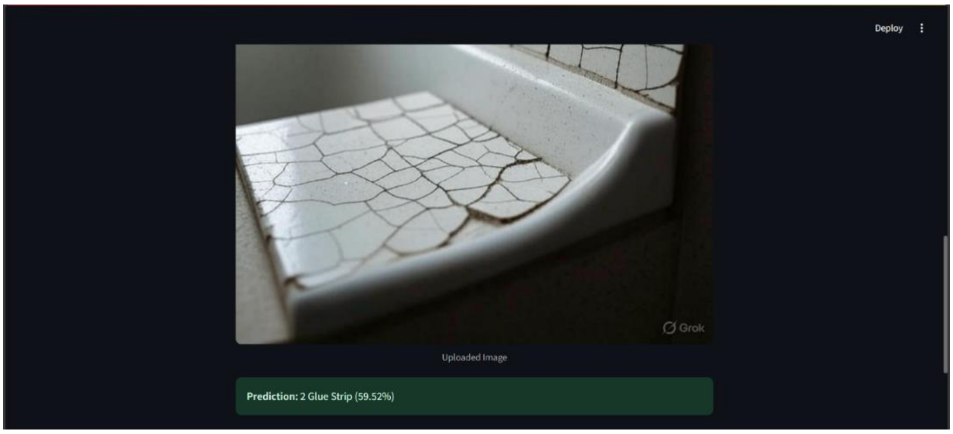
🖼️ Upload an Image

Choose an image...

WhatsApp Image 2025-06-14 at 12.39.14_ce7a0884.jpg 72.2KB ✕

The `use_column_width` parameter has been deprecated and will be removed in a future release. Please utilize the `use_container_width` parameter instead.

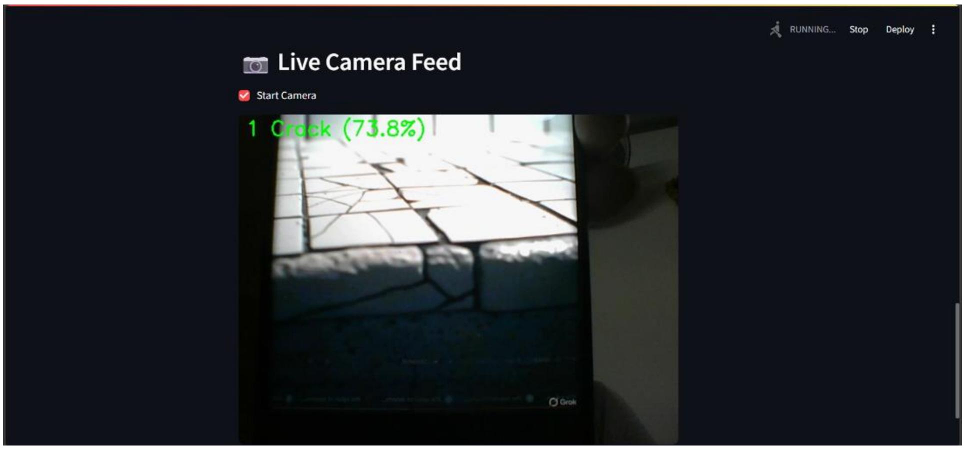
➢ 

➢ 

Uploaded Image

Prediction: 2 Glue Strip (59.52%)

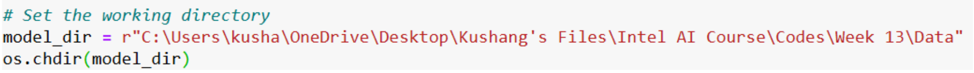
- OpenCV (Live camera feed)

➢ 

- Pillow (Image preprocessing)

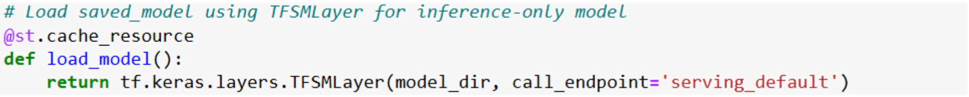
○ **Code Highlights:**

- Set the working directory using:

➢ 

```
# Set the working directory
model_dir = r"C:\Users\kusha\OneDrive\Desktop\Kushang's Files\Intel AI Course\Codes\Week 13\Data"
os.chdir(model_dir)
```

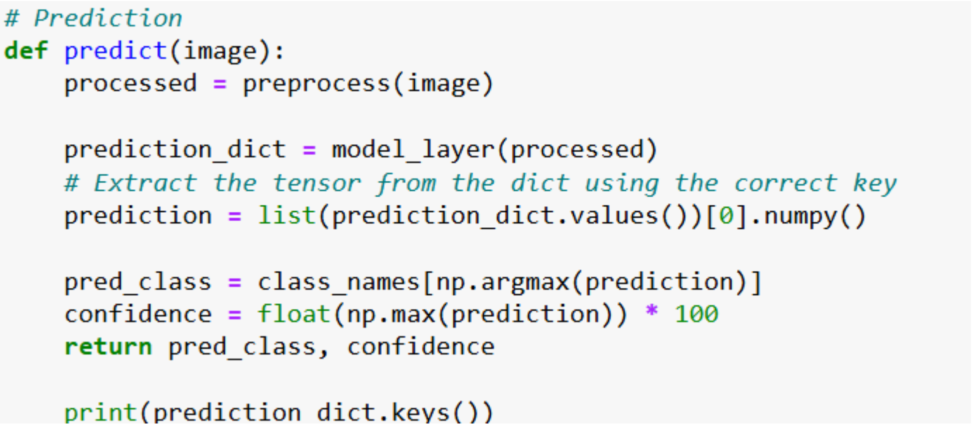
- Load model using:

➢ 

```
# Load saved_model using TFSMLayer for inference-only model
@st.cache_resource
def load_model():
    return tf.keras.layers.TFSMLayer(model_dir, call_endpoint='serving_default')
```

- Normalize and resize images to (224, 224).

- Prediction Function:

➢ 

```
# Prediction
def predict(image):
    processed = preprocess(image)

    prediction_dict = model_layer(processed)
    # Extract the tensor from the dict using the correct key
    prediction = list(prediction_dict.values())[0].numpy()

    pred_class = class_names[np.argmax(prediction)]
    confidence = float(np.max(prediction)) * 100
    return pred_class, confidence

    print(prediction_dict.keys())
```

- **Features of the Streamlit App:**

Feature	Description
Image Upload	Allows user to upload .jpg/.jpeg/.png image and run anomaly detection.
Real-Time Webcam Stream	Live detection on tiles using webcam feed. Predictions are displayed on video frames.
Confidence Metrics	Class prediction along with confidence percentage.
Model Caching	Cached model loading and label processing using @st.cache_resource.

- **Challenges Encountered:**

Issue	Solution
Keras 3 dropped support for load_model() on SavedModel	Replaced with TFSMLayer()
Path-related issues due to OneDrive sync	Used os.chdir() and absolute path handling
Webcam frame conversion errors	Used Pillow for correct RGB conversion before inference
Tensor output as dictionary	Extracted tensor using list(prediction_dict.values())[0]

- **Conclusion:**
    - **This assignment successfully achieves:**
        - Training of a custom tile anomaly detection model via Teachable Machine.
        - Smooth deployment of the model using Streamlit with webcam and upload capabilities.
        - Effective handling of real-time inference using OpenCV and TensorFlow’s TFSMLayer.

    - **This system is a prototype for industrial visual inspection tools used in manufacturing units for automated quality control.**