**CAB (CUT AND BENT) BEND DETECTION USING COMPUTER**

**VISION AND DEEP LEARNING BASED IMAGE CLASSIFICATION**

**USING PYTHON**

**DOCUMENT REVISION LIST**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Revision No. | Revision Date | Revision Description | Action Taken | New Page | Release Notice Reference |
|  |  |  |  |  |  |

**1) About this Document**

The document describes the development of CAB (Cut and Bent) bend detection, and the deep learning model used for image classification.

**2) Intended Audience**

* Head Of Analytics and Insights
* Analytics Project Manager
* Solution Architect
* Data Scientist
* Data Engineer
* Data Visualization Engineer
* Business Owner/User

**Extracting rebar images from original BBS:**

**A screen shot of a computer code

Description automatically generated**

The code iterates over a list of shapes/paths, for each path the inner loop iterates over a list of items such as lines, rectangles, quadrilaterals and curves. It raises a ValueError if it encounters any item that is not recognized.

**Saving High Quality Images from Processed BBS:**

A screen shot of a computer code

Description automatically generated

Gets the total number of pages in the pdf doc and iterates over each page, the code utilizes the matrix method to output images based on the specified zoom factor (in this case 5) i.e. the saved image file will have 5 times the pixels for length and breadth for better clarity.

**Executive Summary**

**Problem Overview:**

The Cut and Bent (CAB) process is crucial for producing various components. However, manual inspection of bends for quality control is time-consuming, prone to human error, and can lead to production bottlenecks. Automating bend detection and measurement offers significant advantages in efficiency, accuracy, and cost reduction.  
  
This project aims to develop a robust and scalable solution for CAB bend detection and measurement using computer vision and deep learning techniques. By leveraging image processing algorithms and machine learning models, we can achieve real-time bend detection, classification, and measurement, enabling automated quality control and process optimization in the steel industry.

**Key Objectives:**

* Develop a system for automated bend detection in cut and bent rebar
* Utilize deep learning models for image classification of various Rebar Cases.

**Approach:**

* Extracting the drawings/rebar images from the processed BBS image file and storing the extracted images in a separate folder.
* Pre-Processing the extracted BBS image files by removing noise and converting them to binary images making it easier to process and to work with.
* To prepare the dataset respective class folders were made and images to corresponding classes were inserted.
* Creating the Deep-Learning Model by adding required Layers for feature extraction and training the model on the created labelled dataset.

**Challenges:**

* Lack of Appropriate Resources: The BBS images received were blurry and pixelated making it difficult to pre-process and to detect bends.
* Complexity of bend variations: CAB Rebars have diverse bend types, angles, and radii, making detection and measurement complex
* Labelling the data was tedious and Time consuming

**Benefits:**

* Increased Efficiency: Automation can drastically speed up the inspection process compared to manual methods, potentially increasing production throughput.
* Scalability: The Image Classification Model Can Very easily be Scaled by inserting other various rebar cases.

**Rectangular Cab Bend Calculation:**

Image Pre-Processing:

Imported required Libraries, converted image to binary format, removed noise and applied bilateral filtering for smoother edges.

A screenshot of a computer program

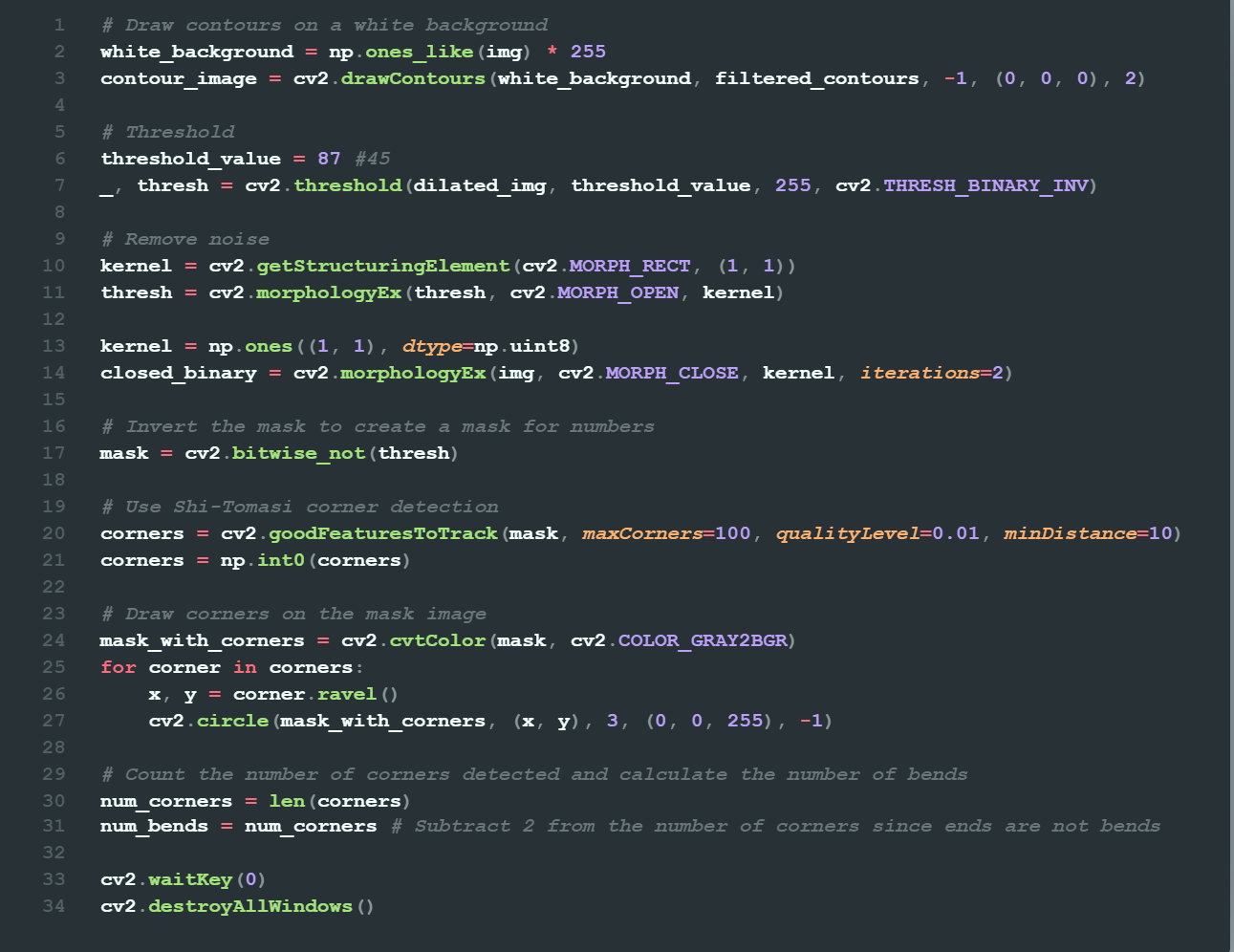
Description automatically generated

A black and white graph

Description automatically generated

The logic behind removing numbers is to find the perimeter of all the contours and setting a threshold so that all objects below the set value are removed i.e. numbers will have smaller perimeter compared to the rebar.

Utilized Shi-Tomasi Corner Detection for calculating number of bends and saved the results as jpg format image files.



A graph of numbers and numbers

Description automatically generated

**Straight Bar Bend Calculation:**

The Image had numbers/dimensions closely sticking to the rebar making it difficult to extract the rebar image. To tackle this issue Laplacian contouring is utilized for accurate detection of rebar.

Since the image has 2 open ends the code detects the ends as corners as well, so to tackle this issue a simple equation is used.

**A graph with a line

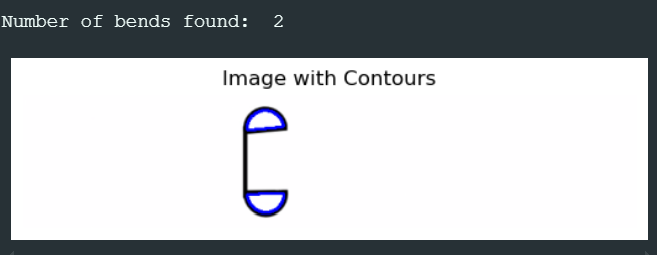
Description automatically generated with medium confidence A screenshot of a computer program

Description automatically generated**

**Curved Rebar Bend Calculation:**

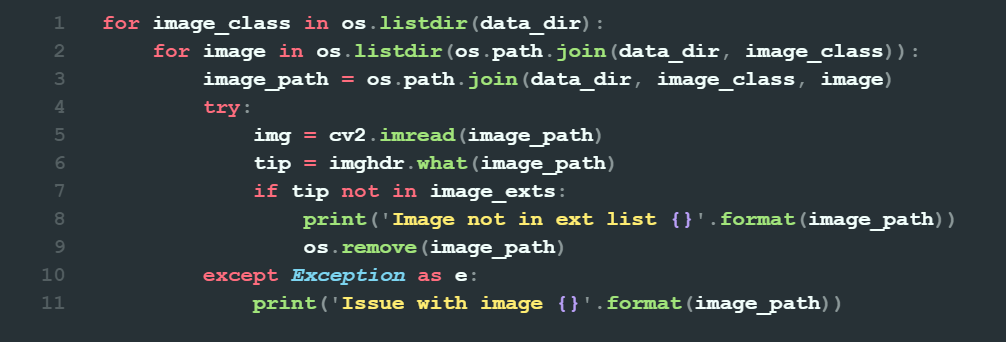
The approach behind bend calculation in the circular cases is to calculate the number of gaps formed, which was achieved by hierarchical contouring where 1st level are the outer counters and the 2nd level are the gaps.

****

 A screenshot of a computer

Description automatically generated

**Image Classification Model:**



* The above snippet iterates over the dataset directory and all the subdirectories representing the image classes, it also checks if the files available are image files of the specified format else that specific files is removed.

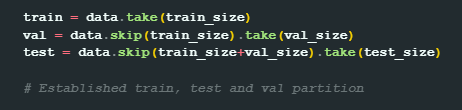
A screen shot of a computer program

Description automatically generated

* Used a Utility function that automatically instructs to determine the labels for images, creates batches of 8 images each, resizes and takes an input of images of (256 x 256), one hot coding the labels is important for properly labelling the classes to the images.

A group of colorful text

Description automatically generated



* Sets a defined size for training, validation and test dataset.

A screen shot of a computer program

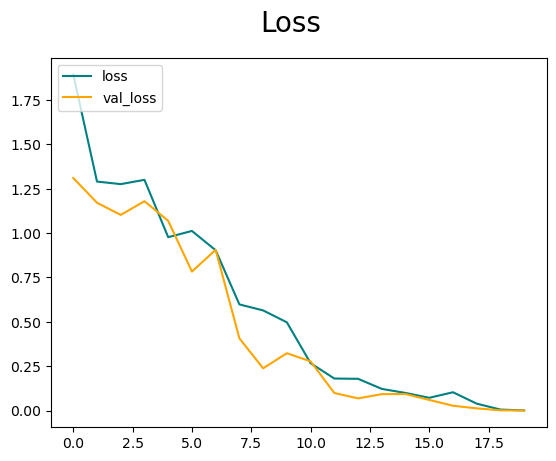
Description automatically generated

* Defined a CNN layer architecture for feature extraction, the architecture consists of convolutional filters to detect different features of the image along with MaxPooling layers for down sampling to avoid overfitting and improves efficiency.
* Flatten layer is used to convert the Convolutional layers to 1D vector before connecting to the fully connecting layers using the Dense Layer

**Train the Model:**



**Visualizing model performance during training process:**

 A graph with blue and orange lines

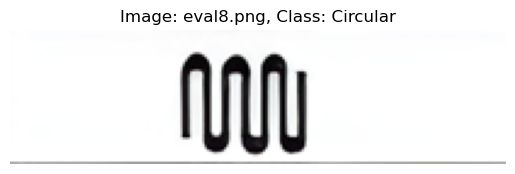
Description automatically generated

A screenshot of a computer program

Description automatically generated

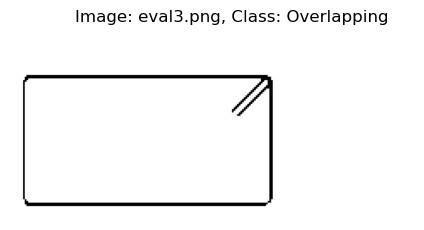
This code block iterates through a set of images from a specified directory, preprocesses them to match the format required by the trained TensorFlow model, and then uses the model to predict the class label for each image. Finally, it displays each image along with the model's predicted class.

**Results by the model for image from each class:**

A black line with text

Description automatically generated A black line with a red and white text

Description automatically generated with medium confidence

 A black line drawing of a rectangular object

Description automatically generated

**Challenges:**

* Restricted access to the original BBS file led to carrying out work with low quality pixelated images due to which bend detection was inaccurate in some cases.
* Lack of rebar cases to train the deep learning model on may lead to inaccurate classifications if a new rebar case is introduced that the model has not been trained on.

**References:**

* <https://youtu.be/oXlwWbU8l2o?si=Q14PmO7MbyVmNXFQ>
* <https://youtu.be/IA3WxTTPXqQ?si=Cf66Y1p5Ssv6vTEv>
* <https://youtu.be/jztwpsIzEGc?si=n4aOaUmzbRmKD4R3>
* <https://youtu.be/zfiSAzpy9NM?si=M1CcUSFLGO84N4qX>