**Task 1 – Identifying a suitable algorithm and dataset preparation.**

**SE4.1 -Evaluate and research possible scenarios for the application of object detection and tracking algorithms.**

1. **A critical review of each:**

**Image Classification:**

Description: Image classification involves assigning a label or category to an entire image based on its content. This method is particularly useful when you have a dataset of images containing different types of waste items, and you want the system to classify each image into predefined categories (e.g., organic waste, recyclable materials, non-recyclable waste).

Application: You can use image classification to quickly identify the type of waste in a given image, making it an efficient and straightforward approach.

**Object Detection using Neural Networks:**

Description: Object detection involves not only categorizing the contents of an image but also locating and outlining the specific objects within the image. Convolutional Neural Networks (CNNs) are commonly used for object detection tasks.

Application: Object detection is useful when you want to identify and locate multiple waste items within an image. For instance, if you have an image of various waste items scattered on the ground, object detection can provide bounding boxes around each item along with its corresponding category.

**Instance Segmentation:**

Description: Instance segmentation goes a step further than object detection by not only identifying and locating objects but also precisely outlining the boundaries of each instance of the object. This is particularly helpful when multiple instances of the same object class exist in an image.

Application: In the context of waste identification, instance segmentation can be beneficial when you need a detailed understanding of the boundaries of each waste item in an image. This method can help in precisely delineating individual items, providing a more granular analysis.

Each of these methods has its own strengths and weaknesses, and the choice depends on the specific requirements of the waste identification task. Often, a combination of these techniques may be employed in a comprehensive waste identification system to achieve accurate and detailed results. Implementing these methods typically involves training neural networks on labelled datasets that include examples of the waste items you want to identify.

**Strengths and weaknesses of each classification method for identifying waste items:**

**Image Classification:**

Strengths:

Simplicity: Image classification is relatively simple and computationally efficient.

Easy Implementation: It is easier to implement compared to more complex techniques.

Interpretability: Results are easily interpretable, as each image is assigned a specific category label.

Weaknesses:

Limited Detail: It may lack detailed information about the specific location and boundaries of waste items within an image.

Single Label: Each image is assigned a single label, which might be limiting when there are multiple types of waste in one image.

**Object Detection using Neural Networks:**

Strengths:

Localization: Provides information about the location of waste items through bounding boxes.

Handling Multiple Instances: Capable of handling multiple instances of different waste items within a single image.

More Information: Offers more detailed information than simple image classification.

Weaknesses:

Complexity: Implementation and training can be more complex compared to image classification.

Computationally Intensive: May require more computational resources, especially with larger datasets and more complex network architectures.

**Instance Segmentation:**

Strengths:

Precise Boundaries: Provides precise boundaries for each instance of a waste item.

Fine Detail: Offers a more granular understanding of the image content compared to object detection.

Handling Overlapping Objects: Can handle cases where waste items overlap in the image.

Weaknesses:

Computational Intensity: More computationally intensive than both image classification and object detection.

Complexity: Implementation and training are more complex, requiring careful annotation of detailed boundaries.

Data Requirements: Typically requires larger and more detailed datasets for effective training.

Choosing the right method depends on the specific goals of waste identification, the level of detail required, and the available computational resources. In many applications, a combination of these methods might be employed to leverage their respective strengths and address their weaknesses for a more robust waste identification system.

1. **Your choice and justification of your preferred classification method.**

Image classification was selected for the process of classification as this offered the best compromise given that the computer used for the exercise lacked the computation power and GPU availability to handle computational complexity of image detection using neural nets or instant segmentation.

1. **A review of the Python implementation of your chosen method. (you may refer to tutorials for this task)**

Tensorflow open-source machine learning platform was used for the implementation of the model.

Activating the Scripts:

terminal: .\ipcv\Scripts\activate

Installing tensorflow & tensorboard library in python environment:

terminal: pip install tensorflow[and-cuda]

terminal: pip install tensorboard

**References:**

The tutorial for loading of data using Keras utility was followed in order to create the dataset: (<https://colab.research.google.com/github/tensorflow/docs/blob/master/site/en/tutorials/load_data/images.ipynb#scrollTo=2ZgIZeXaDUsF>).

Images for mixed recyclables and glass waste categories used for training purposes were obtained from Kaggle.com: (<https://www.kaggle.com/datasets/mostafaabla/garbage-classification>)

Images for e-waste categories were obtained from universe.roboflow.com: (<https://universe.roboflow.com/new-workspace-f7og7/e-waste-mx8fq/dataset/1/download>)

**Categories selected for training:**

1. E-waste
   1. Mice
   2. Printers
2. Mixed Recyclables
   1. Cardboard
   2. Plastic
3. Glass
   1. Oil Bottles
   2. Wine Glasses

**Total number of images used: 2513 images**

**Training Set:**

Found 2513 files belonging to 6 classes.

Using 2011 files for training.

**Validation Set:**

Found 2513 files belonging to 6 classes.

Using 502 files for validation.

**Number of Instances:**

Training Set: 2011 images

Validation Set: 502 images

**Class Names:**

['e\_waste\_Mouse', 'e\_waste\_Printer', 'glass\_oilbottles', 'glass\_winebottles', 'mixed\_recyclables\_cardboard', 'mixed\_recyclables\_plastic']

**Description:**

The dataset consists of images representing different classes of waste, specifically focusing on e-waste (mice and printers), mixed recyclables (cardboard and plastic), and glass (oil bottles and wine bottles). The training set comprises 2011 images, while the validation set consists of 502 images. The class names provide a clear distinction between the different types of waste items in the dataset. This dataset can be used for training and evaluating an image classification model to automatically categorise waste items into their respective classes.

**Reference:**

<https://colab.research.google.com/github/tensorflow/docs/blob/master/site/en/tutorials/images/classification.ipynb>

**Task 2 – Augment your dataset with geometrical transformations.**

1. Research and write a short review of geometric transformations which can be used to distort a waste item to produce a **twisting** or **crushing effect**

Geometric transformations are indispensable tools in the realm of computer graphics, offering a means to manipulate visual elements within images. When it comes to distorting waste items to simulate a twisting or crushing effect, two prominent classes of geometric transformations come into play: affine transformations and perspective transformations.

**Affine Transformations:**

Overview:

Affine transformations are a set of linear transformations that include translation, rotation, scaling, and shearing. These operations maintain parallelism and preserve ratios of distances between points.

Application to Twisting Effect:

Rotation: To create a twisting effect, rotation is a key affine transformation. By rotating the waste item, it introduces a spiral or helical distortion, capturing the dynamic nature of twisting.

Scaling: Adjusting the scale along specific axes can elongate or compress parts of the waste item, enhancing the perception of a twist.

Shearing: Shearing introduces a slanting effect, allowing the waste item to deform in a controlled manner, contributing to the overall twisting appearance.

Application to Crushing Effect:

Scaling: Scaling can be utilized to simulate compression by adjusting the dimensions of the waste item, replicating the deformation associated with crushing.

Shearing: Similar to the twisting effect, shearing can also play a role in introducing asymmetrical compressions, adding realism to the crushing effect.

**Perspective Transformations:**

Overview:

Perspective transformations are non-linear transformations that alter an object's appearance based on its position relative to the viewer's perspective. These transformations are crucial for simulating depth and three-dimensional effects.

Application to Twisting Effect:

Perspective transformations can be applied to mimic the way a twisted waste item appears from different viewing angles, adjusting the size and orientation of various parts based on their perceived depth.

Application to Crushing Effect:

By incorporating perspective transformations, the waste item can be distorted to create the illusion of crushing. As parts of the item move away from the viewer, they can be proportionally reduced in size, contributing to the perception of compression.

**Review:**

Geometric transformations, specifically affine and perspective transformations, helps the user to infuse waste items with dynamic and realistic visual effects. Affine transformations, with their linear nature, provide a more controlled approach to twisting and crushing effects through rotations, scalings, and shearings. On the other hand, perspective transformations add a layer of realism by simulating the impact of viewing angles on the appearance of the waste item, enhancing the overall depth and three-dimensionality.

The combination of these transformations enables artists, animators, and designers to craft visually compelling scenes, whether for creative expression or educational purposes. As technology advances, the integration of geometric transformations continues to play a pivotal role in enhancing the visual narrative and storytelling potential within the digital realm.

Creating a twisting or crushing effect in computer graphics or geometric transformations involves manipulating the shape or appearance of an object. Some geometric transformations that can be used to achieve these effects are:

**Reference:** (<https://docs.opencv.org/3.4/da/d6e/tutorial_py_geometric_transformations.html>)

Rotation:

Twisting: Apply a rotation transformation around the object's center or a specific pivot point. Repeated rotations over time can create a twisting effect.

Crushing: You can rotate the object along its axes to deform its shape, simulating a crushing effect.

Shearing:

Twisting: Apply shearing transformations along different axes. Combining shearing along multiple axes can give a more complex twisting appearance.

Crushing: Shearing along one or more axes can compress or stretch the object in specific directions, simulating a crushing effect.

Scaling:

Twisting: Non-uniform scaling along different axes can introduce a twisting effect. For example, scaling more along one axis than another.

Crushing: Uniform or non-uniform scaling can simulate compression or expansion, creating a crushing effect.

Deformation:

Twisting: Apply a nonlinear deformation function to the object's vertices, distorting them in a way that produces a twisting appearance.

Crushing: Similar to twisting, use a deformation function that compresses or expands parts of the object to create a crushing effect.

Combination of Transformations:

Twisting or Crushing: Combine multiple transformations in a sequence to achieve more complex effects. For instance, you could rotate and scale the object simultaneously to create a twisting and stretching effect.

**Task 4 – Feature Finding algorithms.**

An 80/20 train and validation split was used.

**Training results:**

A screenshot of a graph

Description automatically generated

**Task 5 – Create a working AI waste separation assistant identifier**

Tensorboard was used in order to evaluate the model while training it. In order to make sure that the model was converging and not overfitting ‘**layers.Dropout’** was used.

**Training results with** *‘****layers.Dropout’*:**

A screenshot of a graph

Description automatically generated

A screenshot of a computer

Description automatically generated**Tensorboard Dashboard:**

**Proof of logs:**

A screenshot of a computer

Description automatically generated

**3a. Sample outputs showing image with prediction bounding box/segmentation and confidence %**

A screenshot of a computer screen

Description automatically generated

A white text with blue text

Description automatically generated

**3b. Report on model evaluation metrics:**

The model was trained on a dataset consisting of 80% training and 20% validation split, utilizing data augmentation techniques. After ten epochs, the model achieved an overall accuracy of 81% on the validation set. Precision values varied across classes, with the highest precision observed for glass oil bottles (82%) and mixed recyclables cardboard (92%). However, recall values varied, indicating that the model struggled to correctly identify e-waste Printer instances (50%). The F1-score, which balances precision and recall, ranged from 0.52 to 0.90 across classes. Overall, the model demonstrated good performance, but improvements are needed, especially in distinguishing between different e-waste categories. Future work may involve fine-tuning the model, exploring additional data augmentation strategies, and considering alternative architectures for enhanced performance.

A screenshot of a graph

Description automatically generated

A screenshot of a computer screen

Description automatically generated

**4.**

**a. Develop an algorithm which makes use of the results of the detection algorithm to report the waste type and bin the item should be thrown in.**

**b. For a set of images or a video feed demonstrate how your Waste Separation proof of concept is functioning by outputting sample output images/video.**

A collage of different objects

Description automatically generated

A white text with blue text

Description automatically generated

**c. Write a short report detailing the limitations of your proof-of-concept and suggestions for improvement.**

While the proof-of-concept successfully trained a waste classification model, there are notable limitations that should be acknowledged. First, the dataset size is relatively small, which may limit the model's ability to generalise well to diverse waste scenarios. Additionally, the dataset imbalance among different waste categories could lead to biased model predictions. The current architecture lacks complexity, and further experimentation with more advanced CNN architectures, such as deeper networks or transfer learning from pre-trained models, may enhance performance. The model's sensitivity to variations in object positioning and background clutter is another limitation, suggesting the need for additional robustness in handling real-world scenarios. Moreover, the evaluation metrics revealed challenges in accurately classifying certain waste categories, indicating potential areas for improvement in the model's discriminative abilities. To address these limitations, future work can focus on expanding and diversifying the dataset, optimising the model architecture, and incorporating advanced techniques to handle environmental variations and improve overall robustness.