1003

REAL WASTE

Team- Falcon

- Mahit Alva
- Rani Dynna Parida
- Anushka Mukherjee

Guide’s Name

Nikhil Maurya

GOAL

1. **Enhanced Waste Classification:** The RealWaste dataset categorizes nine material types, enhancing waste sorting accuracy and improving recycling and waste management processes

2. **Sustainable waste management**: RealWaste enhances sustainable waste management by training machine learning model to automate waste identification and sorting, boosting recycling rates

3. **Real World Application:** . The dataset from Whyte's Gully Waste and Resource Recovery Centre enables realistic waste classification models, addressing limitations of traditional methods using artificial datasets.

4. **Machine Learning Advancements:** The RealWaste project utilizes Convolutional Neural Networks, achieving up to 89.19% classification accuracy for real-world waste samples, enhancing waste management automation.

5. **Addressing Current Limitations :** Leveraging machine learning with the RealWaste dataset enhances waste classification, reducing manual errors and improving scalability in waste management.

6 . **Research and Development Opportunities :** The dataset supports academic and industrial research, fostering innovation in waste management and AI applications for environmental sustainability.

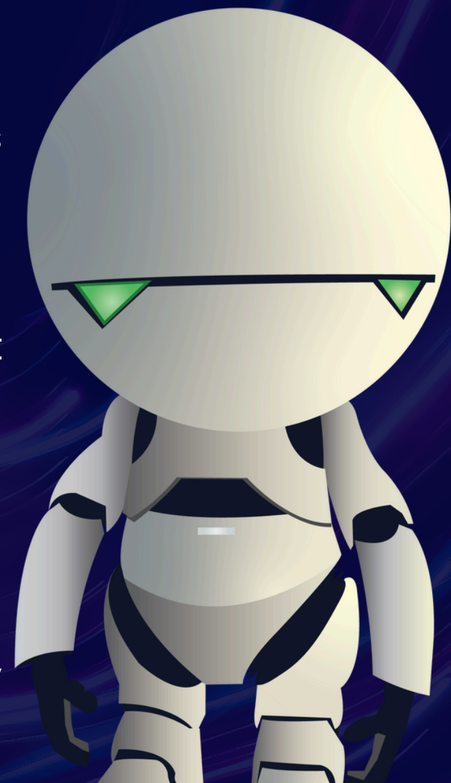
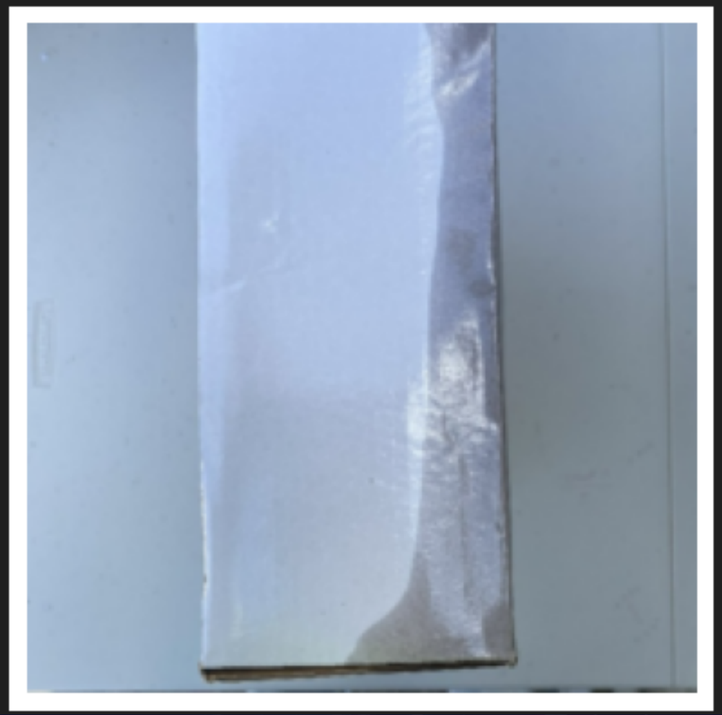


IMAGE PREPROCESSING

The preprocessing implemented in this setup involves a series of transformations to prepare images for model training, testing, and validation. Training images undergo essential preprocessing steps, including resizing to 224x224 pixels, center cropping, converting to tensors, and normalizing using standard mean and standard deviation values. This approach helps ensure that the model receives input data in a consistent format. For validation and test images, we apply the same resizing, cropping, and normalization transformations, maintaining consistency during evaluation without introducing excessive alterations. This strategy helps strike a balance between data diversity and model performance.

LOOKING AT SOME TRAINING IMAGES

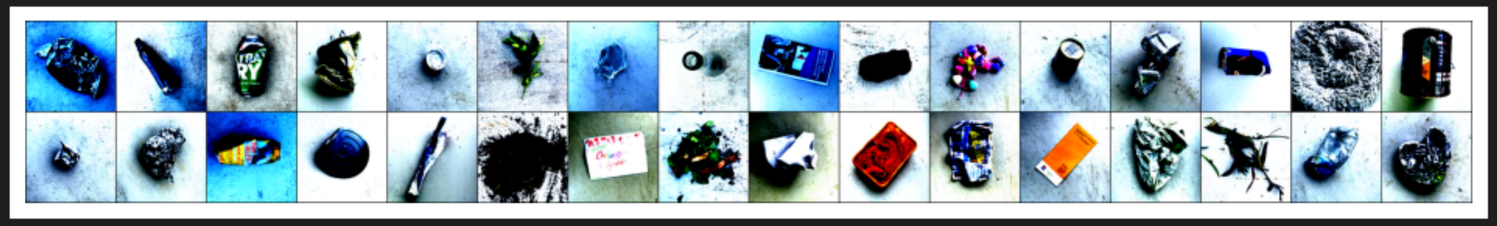
Label: Cardboard (0)



Label: Textile Trash (7)

Label: Plastic (6)

```
eda.show_batch(train_dl)
```



+ Code + Markdown

```
eda.show_batch(train_dl)
```

We can conclude that the process of augmenting the training dataset has been successful as the normal RGB images which are present in the dataset has many variations.

MODEL TRAINING

To train the model effectively, we utilized the pre-trained EfficientNet-B0 model from torchvision. This model is well-liked in PyTorch because it balances accuracy and speed nicely. It can adjust its depth, width, and resolution in an efficient way, which means it delivers strong performance without requiring a lot of computing power. This makes it a smart option for getting good results without heavy resource use.

```python
class Model(ImageClassificationModel):
    def __init__(self, out_size):
        super().__init__()

        self.efficientnet = models.efficientnet_b0(weights='DEFAULT')

        num_features = self.efficientnet.classifier[-1].in_features

        self.efficientnet.classifier[-1] = nn.Linear(num_features, out_size)

    def forward(self, x):
        x = self.efficientnet(x)
        return x
```

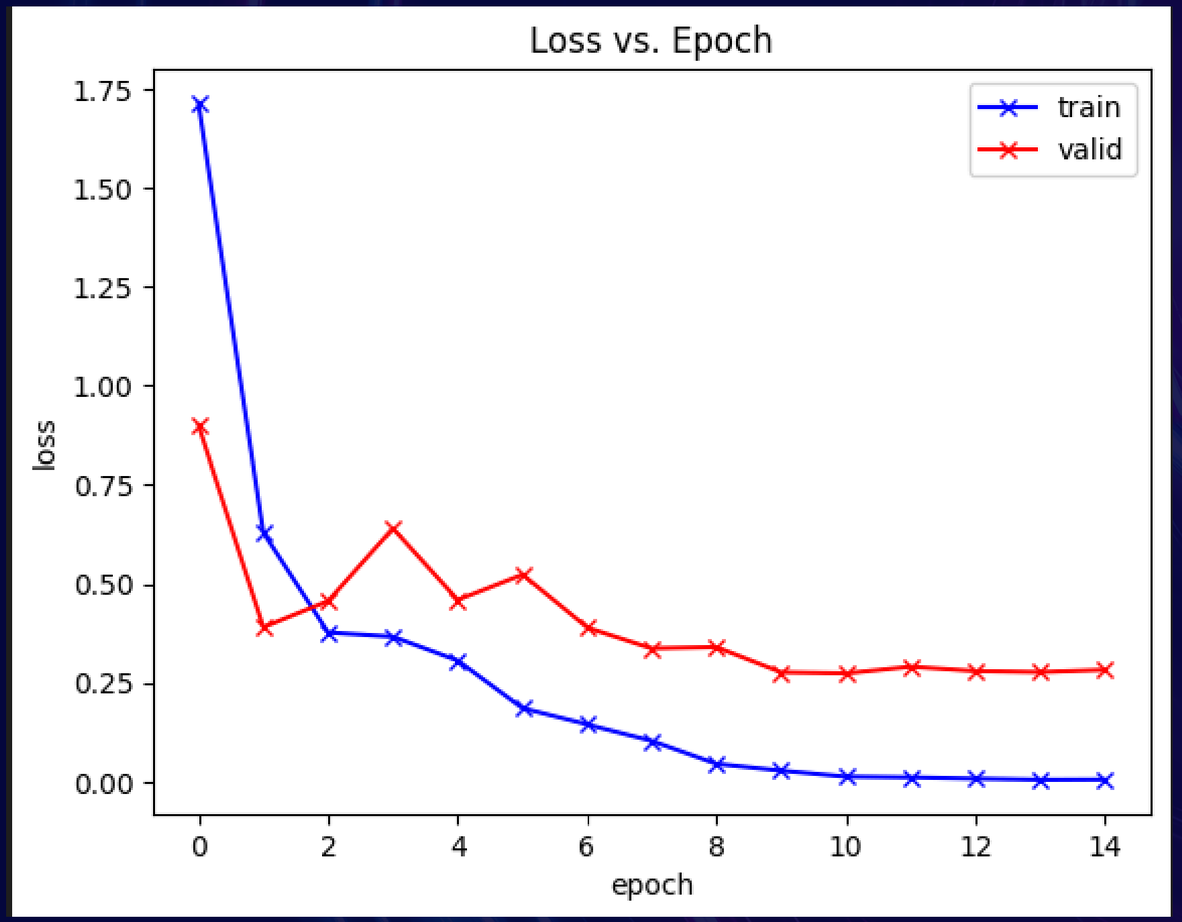
TRAINING PROGRESS

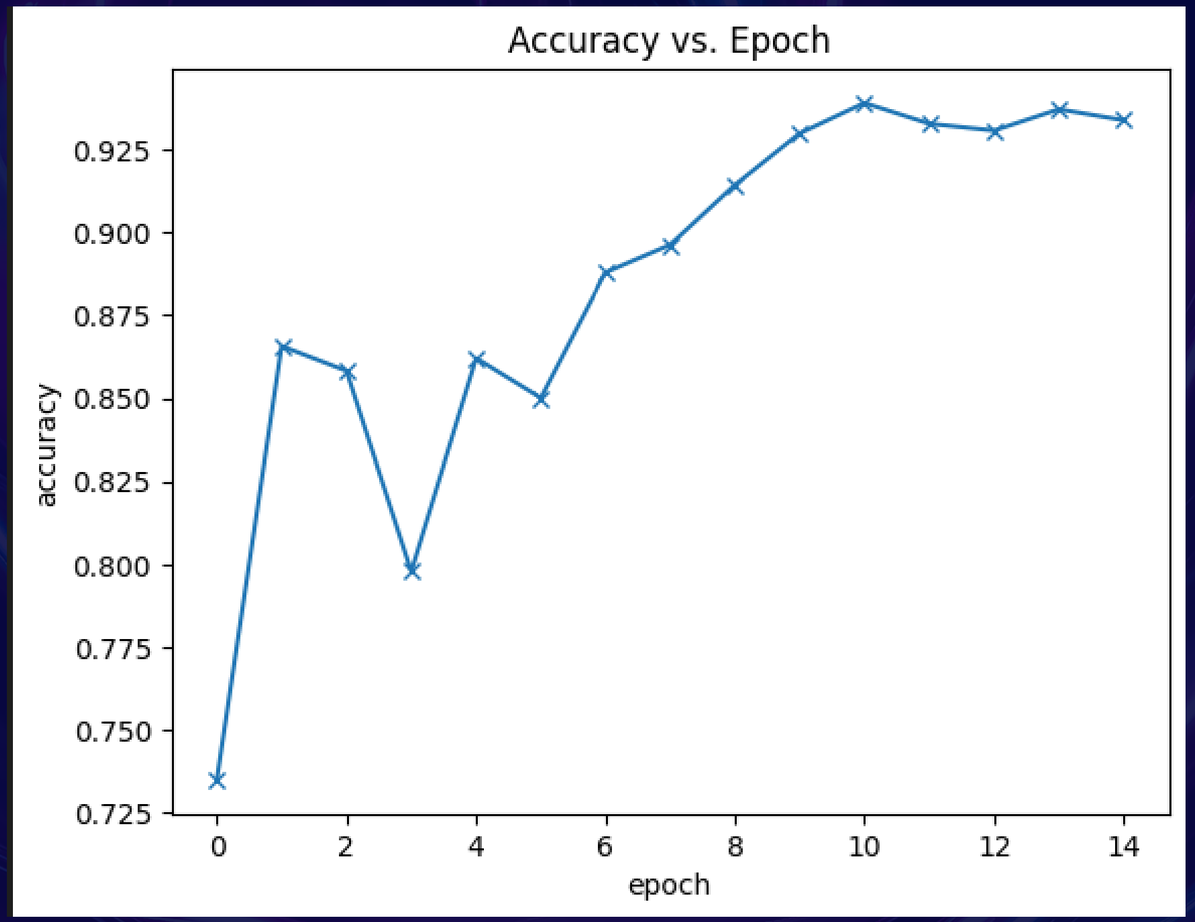
```python
import time
start=time.time()

history=model.fit(epochs=epochs,max_lr=max_lr,train_loader=train_dl,val_loader=test_dl,
                  weight_decay=weight_decay,grad_clip=0.1,
                  opt_func=optim.AdamW)

print(f'Training time: {(time.time()-start)/60} minutes')
```

```
Epoch [0], train_loss: 1.7141, train_acc: 0.4762, val_loss: 0.9000, val_acc: 0.7349
Epoch [1], train_loss: 0.6298, train_acc: 0.7896, val_loss: 0.3905, val_acc: 0.8656
Epoch [2], train_loss: 0.3772, train_acc: 0.8758, val_loss: 0.4566, val_acc: 0.8582
Epoch [3], train_loss: 0.3665, train_acc: 0.8829, val_loss: 0.6403, val_acc: 0.7978
Epoch [4], train_loss: 0.3062, train_acc: 0.8981, val_loss: 0.4595, val_acc: 0.8620
Epoch [5], train_loss: 0.1858, train_acc: 0.9408, val_loss: 0.5233, val_acc: 0.8500
Epoch [6], train_loss: 0.1448, train_acc: 0.9537, val_loss: 0.3897, val_acc: 0.8879
Epoch [7], train_loss: 0.1032, train_acc: 0.9664, val_loss: 0.3370, val_acc: 0.8962
Epoch [8], train_loss: 0.0450, train_acc: 0.9874, val_loss: 0.3403, val_acc: 0.9142
Epoch [9], train_loss: 0.0279, train_acc: 0.9916, val_loss: 0.2765, val_acc: 0.9299
Epoch [10], train_loss: 0.0127, train_acc: 0.9966, val_loss: 0.2739, val_acc: 0.9391
Epoch [11], train_loss: 0.0109, train_acc: 0.9968, val_loss: 0.2907, val_acc: 0.9328
Epoch [12], train_loss: 0.0077, train_acc: 0.9984, val_loss: 0.2796, val_acc: 0.9307
Epoch [13], train_loss: 0.0048, train_acc: 0.9989, val_loss: 0.2770, val_acc: 0.9372
Epoch [14], train_loss: 0.0053, train_acc: 0.9982, val_loss: 0.2828, val_acc: 0.9339
Training time: 4.24060392777125 minutes
```

The image illustrates the model's training journey over 15 epochs, presenting both training and validation metrics. At the start, the training loss is relatively high, but it consistently decreases, signaling that the model is effectively learning from the data. Concurrently, the validation accuracy shows a steady increase, suggesting that the model is enhancing its ability to make accurate predictions on unseen data. This balance between improving training and validation metrics is crucial, as it indicates that the model is learning patterns rather than simply memorizing the training set. Overall, the training process lasted approximately 4.2 minutes, indicating an efficient and smooth learning experience

EVALUATING METRICS





PERFORMANCE ON UNSEEN DATA

```
accuracy,f1,precision,recall=eval.evaluate(test_dl)
print(f'Test dataset metrics: ')
print(f'Accuracy: {accuracy*100}%')
print(f'F1 Score: {f1*100}%')
print(f'Precision Score: {precision*100}%')
print(f'Recall Score: {recall*100}%')
```

```
Test dataset metrics:
Accuracy: 93.37539432176656%
F1 Score: 93.33573256024304%
Precision Score: 93.33042083204379%
Recall Score: 93.5807298088428%
```

The image shows the evaluation metrics for a machine learning model that has been tested on a dataset. With an accuracy of about 93.38%, the model is performing well, correctly classifying the majority of test samples. The F1 score, which provides a balanced assessment of both precision and recall, is around 93.34%. This indicates the model's capability to manage both false positives and false negatives effectively.

Additionally, the precision and recall scores are both above 93%, highlighting that the model not only makes accurate predictions but is also consistent and reliable in its performance.

PREDICTING ON UNSEEN DATA

	Index	Actual	Predicted
888	888	Paper	Paper
652	652	Miscellaneous Trash	Miscellaneous Trash
515	515	Plastic	Plastic
321	321	Glass	Glass
687	687	Metal	Metal
295	295	Textile Trash	Textile Trash
728	728	Miscellaneous Trash	Miscellaneous Trash
706	706	Vegetation	Vegetation
824	824	Vegetation	Vegetation
940	814	Glass	Glass

	Index	Actual	Predicted
120	120	Cardboard	Cardboard
900	774	Food Organics	Food Organics
720	720	Cardboard	Cardboard
602	602	Metal	Metal
854	854	Food Organics	Food Organics
12	12	Vegetation	Vegetation
942	816	Plastic	Plastic
945	819	Glass	Glass
873	873	Plastic	Plastic
452	452	Vegetation	Vegetation

As we can see, the model is able the predict the correct class of waste excellently on about 20 random images from the unseen test dataset,indicating that the model is performing really well.


THANK YOU