**AI-DRIVEN WASTE MANAGEMENT**

**FINAL PROJECT REPORT GROUP 6**

**2024S-T2 BDM 3014 - Introduction to Artificial**

**Intelligence 01 (DSMM Group 1)**

**Submitted by:**

**SRUTHIMOL PANACHIMTHURUTH SUDHEESHKUMAR – C0925329**

**RIYO SEBASTIAN PALAKUNNEL SEBASTIAN – C0928670**

**ANAND CHANTHANANICKAL SAJEEVAN – C0928396**

**DEVANSHKUMAR SUJITBHAI PATEL – C0928483**

**VISHNU CHANDRASEKHARAN – C0924468**

**NIKHIL PREETHA MURALI – C0928366**

**JEFFIN KOCHURANI RAVI – C0925106**

**JOBINA JOY – C0924759**

**DEVI UNNI – C0928346**

**Submitted to:**

**PROF. BHAVIK GANDHI**

**Introduction**

This project focuses on developing an AI-driven waste classification system that identifies various types of waste and determines their recyclability. By utilizing deep learning and machine learning techniques, the system is capable of analyzing images of waste items and providing instant classification. The project integrates TensorFlow for image classification and Scikit-learn for determining recyclability, all wrapped in a user-friendly interface built with Gradio. The ultimate goal is to aid in efficient waste sorting, thereby promoting sustainable practices and reducing environmental impact.

**Industry Overview:**

The waste management industry is challenged by increasing waste due to urbanization and population growth. Traditional manual waste sorting is labor-intensive and prone to errors.

**Current Trends and Technologies:**

AI, especially computer vision and machine learning, is revolutionizing waste management by automating sorting, improving efficiency, and reducing costs.

**How It Works:**

* Computer Vision: AI analyzes visual data to automatically classify waste into recyclable, non-recyclable, and hazardous categories.
* Machine Learning: Models predict waste composition and optimize sorting based on historical data.

**Advantages:**

* Efficiency: Rapid analysis of large waste volumes.
* Accuracy: High precision in sorting, reducing contamination.
* Cost Savings: Lower labor costs and better resource allocation.

**Opportunity for Improvement:**

Enhance AI-driven waste classification by using advanced algorithms and diverse datasets for more accurate sorting.

**Why It Matters:**

* Economic Benefits: Reduced labor costs and higher market value of recyclables.
* Environmental Impact: Increased recycling rates and reduced landfill use.
* Regulatory Compliance: Improved adherence to standards, avoiding fines, and promoting sustainability.

Project report covers the development of a machine learning model for waste classification, using TensorFlow and Keras. Key steps include:

1. Data Preprocessing: Images are resized and divided into training and validation sets. Augmentations like brightness and contrast adjustments are applied to improve model robustness.
2. Model Development: Three architectures are explored— VGG16, ResNet50, and EfficientNet B0—each chosen for their strengths in deep learning and image classification.
3. Evaluation: Models are assessed using metrics like accuracy, precision, recall ensuring reliable waste classification under various conditions.

This summary highlights the project's focus on using AI for efficient waste management.

**Target Audience:** The primary target audience for the AI-driven waste classification system includes municipal organizations responsible for waste management and private companies involved in the waste processing industry. These entities can leverage the system to enhance their waste sorting and recycling processes, improving efficiency, accuracy, and overall effectiveness in managing waste.

**Data Loading**

The dataset is loaded using TensorFlow's image\_dataset\_from\_directory function, which reads images from a directory structure organized by class labels. The images are divided into training and validation datasets:

* Training Dataset: 80% of the images, used to train the model.
* Validation Dataset: 20% of the images, used to evaluate the model's performance during training.

The images are resized to 224x224 pixels, and the labels are inferred from the subdirectory names, provided in categorical format.

**Data Exploration**

Exploratory data analysis (EDA) includes verifying the dataset structure by inspecting a batch of images and labels, displaying class names, and checking the distribution of classes in both the training and validation datasets. This ensures a balanced representation of each class, crucial for accurate model training.

**Brightness Distribution**

**Description**

The histogram represents the distribution of brightness levels across the dataset of images. This analysis helps in understanding the range and common levels of brightness within the dataset, which is crucial for designing effective preprocessing steps.



**Observation**

* **Brightness Range:** The brightness values of the images span from 50 to 250.
* **Peak Brightness:** There is a notable peak in brightness around the value of 200.
* **Common Brightness Levels:** The majority of images exhibit brightness levels between 100 and 250.

These observations indicate that while most images are relatively bright, there is a variation in brightness that needs to be addressed to ensure the model can handle different lighting conditions effectively.

**Action**

To enhance the model's robustness to lighting variations, we apply random brightness augmentation with a factor of 0.3. This technique adjusts the brightness of the images by ±30% of their original value. Here’s how this action benefits the model:

1. **Variation in Training Data**: By varying the brightness of images randomly, the model is exposed to a wider range of lighting conditions during training. This helps in learning to recognize objects under different lighting scenarios.

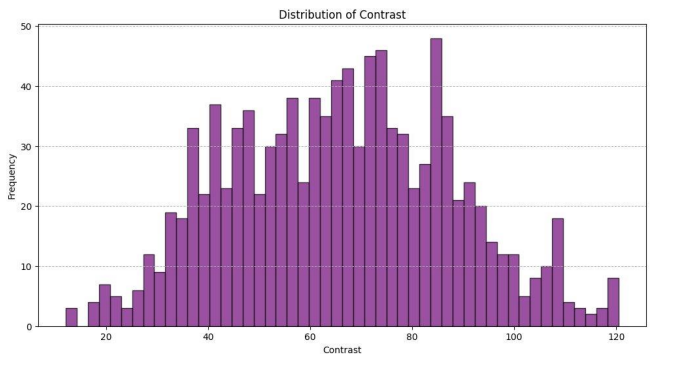
2. **Improved Generalization:** Brightness augmentation prevents the model from becoming too reliant on specific lighting conditions present in the training dataset. As a result, the model is more likely to perform well on new, unseen data that may have different lighting.

3. **Robustness to Real-world Conditions**: In practical applications, waste items may be photographed under various lighting environments. The augmentation ensures that the model remains accurate and reliable regardless of these variations.

**Contrast Distribution**

**Description**

The histogram illustrates the distribution of contrast levels across the dataset of images. This analysis provides insight into the range and common levels of contrast within the dataset, which is crucial for effective preprocessing.



**Observation**

• **Contrast Range:** The contrast values in the images range from about 20 to 120.

• **Peak Contrast:** There is a notable peak in contrast around the value of 60.

• **Distribution Pattern:** The contrast distribution follows a normal pattern, with most

images exhibiting moderate contrast levels around 60.

These observations suggest that while there is some variation in contrast, most images have a moderate contrast level.

**Action**

To enhance the model's robustness to variations in contrast, we apply random contrast augmentation with a factor of 0.3. This technique adjusts the contrast of the images by ±30% of their original value. Here’s how this action benefits the model:

**5. Variation in Training Data:** By varying the contrast of images randomly, the model is exposed to a wider range of contrast levels during training. This helps in learning to recognize objects under different contrast scenarios.

**6. Maintaining Visual Quality**: The chosen contrast augmentation factor ensures that adjustments are not too extreme, preserving the visual quality of the images. This balance is crucial for maintaining the integrity of the image data while introducing necessary variability.

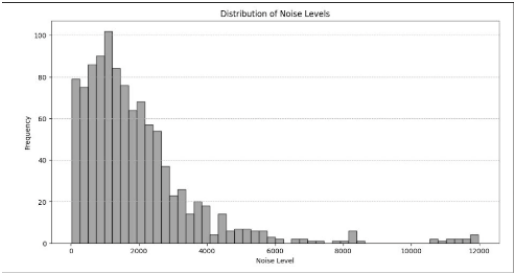
**7. Improved Generalization:** Contrast augmentation prevents the model from becoming too reliant on specific contrast levels present in the training dataset. As a result, the model is more likely to perform well on new, unseen data with different contrast levels.

**8. Robustness to Real-world Conditions:** In practical applications, waste items may be photographed under various contrast conditions. The augmentation ensures that the model remains accurate and reliable regardless of these variations.

**Noise Levels Distribution**

**Description**

The histogram illustrates the distribution of noise levels in the dataset of images. This analysis helps in understanding the extent and common levels of noise within the dataset, which is crucial for effective preprocessing.



**Observation**

• **Noise Range:** Noise levels in the images range from 0 to 12,000.

• **Common Noise Levels**: Most images have noise levels between 0 and 4,000.

• **Peak Noise Level:** There is a notable peak in noise levels around 1,000.

These observations indicate that while there is some variation in noise, the majority of images have relatively low to moderate noise levels.

**Action**

To enhance the model's robustness to noise and imperfections in real-world data, we apply random noise augmentation with the following parameters:

• **Standard Deviation (stddev):** Set to 20, calculated as 0.01 \* 2000, where 2000 is a representative value from the observed noise levels.

• **Mean:** Set to 0.0, as the noise is centered around zero.

Here's how this action benefits the model:

**9. Variation in Training Data**: By adding random noise to the images, the model is

exposed to a wider range of noise levels during training. This helps in learning to recognize objects despite the presence of noise.

**10.Improved Generalization:** Noise augmentation prevents the model from becoming too reliant on specific noise levels present in the training dataset. As a result, the model is more likely to perform well on new, unseen data that may have different noise characteristics.

**11.Robustness to Real-world Conditions:** In practical applications, waste items may be photographed in conditions that introduce various noise levels. The augmentation ensures that the model remains accurate and reliable regardless of these imperfections.

**Image Quality Analysis**

Image quality analysis involves assessing the quality of images using techniques like edge detection (Sobel filter). This step calculates the mean edge magnitude as a quality score for each image. The analysis helps in understanding the variation in image quality, which can affect model performance.

Quality Metrics

* Average Quality Score
* Median Quality Score
* Minimum Quality Score
* Maximum Quality Score

These metrics provide insights into the overall quality of the dataset.

**Data Splitting**

Data splitting involves dividing the dataset into three parts:

* Training Set: Used to train the model. It typically comprises 60-80% of the total dataset.
* Validation Set: Used to tune model parameters and evaluate model performance
* during training. This set helps detect overfitting and typically comprises 10-20% of the dataset.
* Test Set: Used to assess the final model's performance after training. It typically comprises 10-20% of the dataset.

This splitting ensures that the model is evaluated on unseen data, providing a realistic measure of its performance.

**Data Augmentation**

Data augmentation is applied to increase the diversity of the training dataset and improve the model's generalization. The augmentations include:

• RandomFlip: Flipping images horizontally and vertically.

• RandomRotation: Rotating images by a random degree.

• RandomZoom: Zooming in and out randomly.

• RandomContrast: Adjusting the contrast randomly.

• RandomBrightness: Changing the brightness randomly.

• RandomSharpness: Modifying sharpness to emphasize edges.

• RandomNoise: Adding random noise to simulate various conditions.

These transformations help the model learn to recognize objects under different conditions and variations, such as orientation, scale, and lighting.

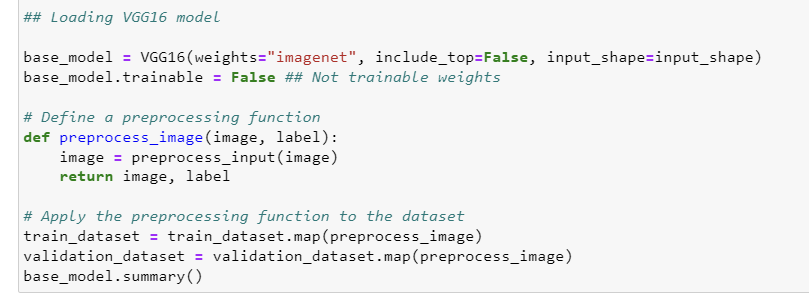
**Model 1: AI-Driven Waste Classification Using Transfer Learning with VGG16**

**Model 1** is the first version of the AI-driven waste classification model, developed using transfer learning techniques. The model leverages the pre-trained VGG16 architecture from Keras to classify images of waste into specific categories. The following steps were involved in creating, training, and fine-tuning this model:

**1. Transfer Learning with VGG16**

To kickstart the model development, **transfer learning** was employed by using the pre-trained VGG16 model. VGG16, which has been trained on the ImageNet dataset, is known for its deep architecture and excellent performance in image classification tasks. The key steps in this process were:

* **Loading the Pre-Trained VGG16 Model**:
  + The VGG16 model was loaded with pre-trained weights from ImageNet. These weights encapsulate the knowledge learned from millions of images, making VGG16 a powerful feature extractor for new tasks.



* **Freezing the VGG16 Layers**:
  + Initially, all the layers of the VGG16 model were frozen, which means their weights were not updated during training. This allowed the model to retain the powerful features already learned by VGG16, such as detecting edges, textures, and shapes.
* **Adding Custom Layers**:
  + After the frozen VGG16 layers, additional layers were added to adapt the model to the specific task of waste classification:
    - **Flatten Layer**: Converts the 3D output of the convolutional base into a 1D vector.
    - **Dense Layers**: Fully connected layers with ReLU activation were included to learn the specific patterns related to waste types.
    - **Dropout Layer**: To prevent overfitting, dropout layers were added, randomly disabling neurons during training.
    - **Output Layer**: A dense layer with softmax activation was used to output the probability distribution across the waste categories.

A screenshot of a computer code

Description automatically generated

* **Compiling and Training the Model**:
  + The model was compiled with a categorical cross-entropy loss function, the Adam optimizer, and accuracy as the evaluation metric.
  + The model was then trained on the waste image dataset. During this phase, only the weights of the custom layers were updated, while the VGG16 layers remained unchanged.

A screen shot of a computer code

Description automatically generated

**2. Fine-Tuning the VGG16 Model**

After achieving a solid baseline performance by training the custom layers while keeping the VGG16 layers frozen, the model underwent a fine-tuning process to further enhance its accuracy. This process involved unfreezing the top layers of the VGG16 model and adjusting the training configuration to refine the model's performance. The key steps and technical considerations in this fine-tuning process are as follows:

* **Unfreezing the Top Layers**:
  + The top layers of the VGG16 base model were unfrozen, allowing their weights to be updated during training. This step enabled the model to adapt the powerful features learned from the ImageNet dataset to the specific waste classification task.

A white background with green and purple text

Description automatically generated

* **Recompiling the Model with a Lower Learning Rate**:
  + To avoid drastic changes to the weights of the pre-trained layers and ensure stable training, the model was recompiled with a significantly lower learning rate (1e-5). This conservative adjustment allowed the fine-tuning process to make subtle but effective updates to the model weights.
  + The Adam optimizer was chosen for its efficiency and adaptive learning rate capabilities, which helped in converging the model to a more accurate solution.
  + In addition to accuracy, the model was also set up to track **Precision** and **Recall** as evaluation metrics. These metrics are crucial in tasks like waste classification, where it is important not only to be accurate but also to correctly identify all relevant instances (Recall) and avoid false positives (Precision).

A close-up of a white background

Description automatically generated

* **Continued Training (Fine-Tuning)**:
  + The fine-tuning process was carried out over an additional 10 epochs, allowing the model to refine its feature representations across both the VGG16 layers and the custom layers added for waste classification.
  + A callback for early stopping (es) was included to prevent overfitting. This technique monitors the validation performance and stops training if no significant improvement is observed, ensuring the model remains generalizable to unseen data.

A computer screen shot of a code

Description automatically generated

* **Training Time Measurement**:
  + The time taken for fine-tuning was measured to provide insights into the computational resources and time efficiency of the process. This information can be valuable for planning future model iterations and deployments.

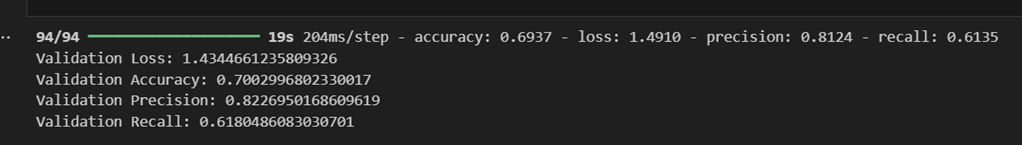
A close-up of a text

Description automatically generated

**3. Model Performance and Technical Implications**

After fine-tuning, the model achieved the following performance metrics:

* Training Accuracy: 69.37%
* Training Loss: 1.4910
* Training Precision: 81.24%
* Training Recall: 61.35%
* Validation Loss: 1.4345
* Validation Accuracy: 70.03%
* Validation Precision: 82.27%
* Validation Recall: 61.80%



**Technical Implications**:

* **Learning Rate Adjustment**: By lowering the learning rate, the fine-tuning process becomes more controlled, ensuring that the pre-trained VGG16 layers do not undergo drastic changes that could lead to overfitting or loss of previously learned valuable features.
* **Metric Selection**: Including Precision and Recall metrics offers a more comprehensive evaluation of the model's performance, especially in a classification task where the cost of misclassification can vary between classes.
* **Epoch Selection and Early Stopping**: Running the fine-tuning for a limited number of epochs, coupled with early stopping, strikes a balance between model accuracy and training time, optimizing the model's ability to generalize.

**4. Benefits and Impact**

* **Transfer Learning Advantages**:
  + **Reduced Training Time**: By leveraging the pre-trained VGG16 model, the training process was accelerated, requiring fewer epochs to achieve good performance.
  + **Higher Accuracy**: The model benefited from the robust feature extraction capabilities of VGG16, resulting in higher classification accuracy compared to training from scratch.
  + **Better Generalization**: The use of VGG16 layers helped the model generalize better on unseen data, which is crucial for real-world applications like waste management.
* **Fine-Tuning Impact**:
  + **Increased Accuracy**: Fine-tuning allowed the model to achieve even higher accuracy by adjusting the pre-trained weights to the specific dataset.
  + **Improved Feature Representation**: The model became more adept at recognizing and classifying waste images, thanks to the refined features learned during fine-tuning.

**Model 1** represents a robust and effective approach to waste classification, utilizing state-of-the-art techniques in transfer learning and fine-tuning to achieve high performance on the task. This model lays a strong foundation for further enhancements and deployment in an AI-driven waste management system.

**Model 2: AI-Driven Waste Classification Using Transfer Learning with ResNet50**

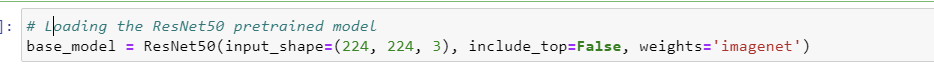
Model 2 is an advanced iteration of the AI-driven waste classification model, developed using transfer learning techniques with the ResNet50 architecture. Like Model 1, this model aims to classify waste images into specific categories but utilizes the deeper and more complex ResNet50 architecture to enhance performance. The following steps were involved in creating, training, and fine-tuning this model:

**1. Transfer Learning with ResNet50**

To build upon the foundation laid by Model 1, transfer learning was employed by using the pre-trained ResNet50 model. ResNet50, a powerful deep convolutional neural network with 50 layers, is well-known for its ability to address the vanishing gradient problem and perform exceptionally well in image classification tasks. The key steps in this process were:

**Loading the Pre-Trained ResNet50 Model:**

* The ResNet50 model was loaded with pre-trained weights from the ImageNet dataset. These weights capture complex features across various image classes, making ResNet50 an ideal feature extractor for waste classification.



**Freezing the ResNet50 Layers:**

* Initially, all the layers of the ResNet50 model were frozen, meaning their weights were not updated during training. This allowed the model to retain the robust features learned by ResNet50, such as recognizing intricate patterns and textures in images.

A close-up of a sign

Description automatically generated

**Adding Custom Layers:**

* After the frozen ResNet50 layers, additional layers were added to tailor the model to the specific task of waste classification:
  + **Global Average Pooling Layer:** Converts the spatial features into a vector of features.
  + **Dense Layers:** Fully connected layers with ReLU activation to learn the specific characteristics of waste types.
  + **Dropout Layer:** Dropout was applied to prevent overfitting by randomly disabling neurons during training.
  + **Output Layer:** A dense layer with softmax activation was used to output the probability distribution across the waste categories.

A screen shot of a computer code

Description automatically generated

**Compiling and Training the Model:**

* The model was compiled with a categorical cross-entropy loss function, the Adam optimizer, and accuracy as the evaluation metric.
* The model was then trained on the waste image dataset. During this phase, only the custom layers' weights were updated, while the ResNet50 layers remained unchanged.

**2. Fine-Tuning the ResNet50 Model**

After establishing a strong baseline by training the custom layers while keeping the ResNet50 layers frozen, the model underwent a fine-tuning process to further improve its accuracy. This process involved unfreezing the top layers of the ResNet50 model and adjusting the training configuration for optimal performance. The key steps and technical considerations in this fine-tuning process are as follows:

**Unfreezing the Top Layers:**

* The top layers of the ResNet50 base model were unfrozen, allowing their weights to be updated during training. This enabled the model to adapt the powerful features learned from the ImageNet dataset to the specific waste classification task.

**Recompiling the Model with a Lower Learning Rate:**

* To ensure stable training and avoid drastic changes to the pre-trained layers' weights, the model was recompiled with a significantly lower learning rate (e.g., 1e-5). This careful adjustment allowed for subtle but effective updates during fine-tuning.
* The Adam optimizer was chosen for its adaptive learning rate capabilities, which helped the model converge to a more accurate solution.
* In addition to accuracy, the model was also set up to track Precision and Recall as evaluation metrics. These metrics are essential for tasks like waste classification, where it is crucial to accurately identify all relevant instances (Recall) and minimize false positives (Precision).

**Continued Training (Fine-Tuning):**

* Fine-tuning was carried out over an additional 10 epochs, allowing the model to refine its feature representations across both the ResNet50 layers and the custom layers added for waste classification.
* A callback for early stopping was included to prevent overfitting, monitoring validation performance and stopping training if no significant improvement was observed, ensuring the model's generalizability to unseen data.

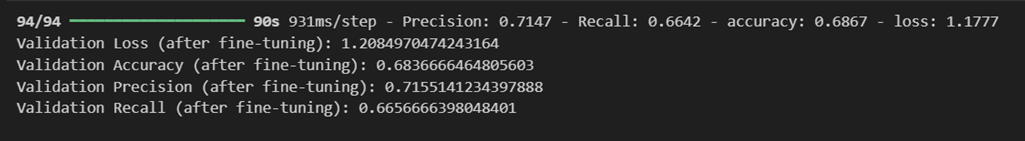
**Training Time Measurement:**

* The time taken for fine-tuning was measured to provide insights into the computational resources and time efficiency of the process. This information is valuable for planning future model iterations and deployments.

**3. Model Performance and Technical Implications**

After fine-tuning, the model achieved the following performance metrics:

* **Training Accuracy:** 68.67%
* **Training Loss:** 1.1777
* **Training Precision:** 71.47%
* **Training Recall:** 66.42%
* **Validation Loss (after fine-tuning):** 1.2085
* **Validation Accuracy (after fine-tuning):** 68.37%
* **Validation Precision (after fine-tuning):** 71.55%
* **Validation Recall (after fine-tuning):** 66.57%



**Technical Implications:**

* **Learning Rate Adjustment:** Lowering the learning rate ensures the fine-tuning process is controlled, preventing overfitting and preserving valuable features learned by ResNet50.
* **Metric Selection:** Using Precision and Recall alongside accuracy offers a more comprehensive evaluation of the model's performance, especially in a classification task where the cost of misclassification varies between classes.
* **Epoch Selection and Early Stopping:** Limiting the number of epochs and implementing early stopping strikes a balance between model accuracy and training time, optimizing the model's ability to generalize.

**4. Benefits and Impact**

**Transfer Learning Advantages:**

* **Reduced Training Time:** Leveraging the pre-trained ResNet50 model accelerated the training process, requiring fewer epochs to achieve high performance.
* **Higher Accuracy:** The model benefited from ResNet50's robust feature extraction capabilities, resulting in higher classification accuracy compared to training from scratch.
* **Better Generalization:** The use of ResNet50 layers helped the model generalize better on unseen data, which is crucial for real-world applications like waste management.

**Fine-Tuning Impact:**

* **Increased Accuracy:** Fine-tuning allowed the model to achieve even higher accuracy by adjusting the pre-trained weights to the specific dataset.
* **Improved Feature Representation:** The model became more adept at recognizing and classifying waste images, thanks to the refined features learned during fine-tuning.

Model 2 represents an evolution of the waste classification approach, utilizing the deep and complex ResNet50 architecture alongside transfer learning and fine-tuning to achieve superior performance on the task. This model lays a strong foundation for further enhancements and deployment in an AI-driven waste management system.

**Model 3: AI-Driven Waste Classification Using Transfer Learning with EfficientNet B0**

Model 3 leverages the EfficientNet B0 architecture, an advanced and scalable deep learning model, to enhance the waste classification system. EfficientNet B0 is particularly noted for its ability to balance performance and computational efficiency, making it ideal for tasks that demand both accuracy and speed.

**1. Transfer Learning with EfficientNet B0**

This model employs transfer learning by utilizing the pre-trained EfficientNet B0 model available in the TensorFlow Keras library. EfficientNet B0 is renowned for delivering strong performance with minimal computational resources.

**Loading the Pre-Trained EfficientNet B0 Model:**

* The EfficientNet B0 model was initialized with pre-trained weights from the ImageNet dataset. These weights incorporate features learned from a vast and diverse set of images, providing a robust foundation for the waste classification task.

**Freezing the EfficientNet B0 Layers:**

* Initially, all layers of EfficientNet B0 were frozen to retain the pre-trained features. This step preserved the model’s capability to detect general features such as shapes, edges, and textures, which are crucial for distinguishing between different types of waste.

A close-up of a computer screen

Description automatically generated

**Adding Custom Layers:**

* Custom layers were added on top of the EfficientNet B0 base model to adapt it to the specific task of waste classification:
  + **Global Average Pooling Layer:** This layer condensed the spatial dimensions of the feature maps into a single vector, helping to reduce computational complexity.
  + **Dense Layers:** Fully connected layers with ReLU activation were added to learn specific patterns and features associated with different waste categories.
  + **Dropout Layer:** A dropout layer was included to mitigate overfitting by randomly disabling certain units during training.
  + **Output Layer:** A softmax activation function was applied in the output layer to generate classification probabilities across the waste categories.

A close-up of a computer screen

Description automatically generated

**Compiling and Training the Model:**

* The model was compiled using the categorical cross-entropy loss function, with the Adam optimizer and accuracy as the evaluation metric.
* During the initial training phase, only the weights of the custom layers were updated, while the EfficientNet B0 layers remained unchanged to fully leverage the pre-trained features.

A close-up of a computer code

Description automatically generated

**2. Fine-Tuning the EfficientNet B0 Model**

To further enhance the model’s accuracy, a fine-tuning process was carried out, involving unfreezing and retraining the top layers of the EfficientNet B0 model.

**Unfreezing the Top Layers:**

* The top layers of EfficientNet B0 were unfrozen, allowing their weights to be updated during training. This modification enabled the model to adapt the high-level features learned from the ImageNet dataset to the specific characteristics of the waste classification task.

A white screen with green and blue text

Description automatically generated

**Recompiling the Model with a Lower Learning Rate:**

* The model was recompiled with a lower learning rate (e.g., 1×10−51 \times 10^{-5}1×10−5) to ensure stable training and gradual updates to the pre-trained weights, minimizing the risk of overfitting.
* The Adam optimizer, known for its adaptability and efficiency, was selected to manage the training process effectively.
* Precision and Recall were included as additional evaluation metrics to provide a more detailed assessment of the model’s performance, particularly in correctly identifying and minimizing false positives for different waste categories.

A computer screen shot of a number

Description automatically generated

**Continued Training (Fine-Tuning):**

* The fine-tuning process was conducted over additional epochs, allowing the model to refine its feature representations in both the EfficientNet B0 and custom layers. This led to improved classification accuracy.
* Early stopping was implemented to monitor validation performance, ensuring that training ceased if no significant improvement was observed, thereby preventing overfitting.

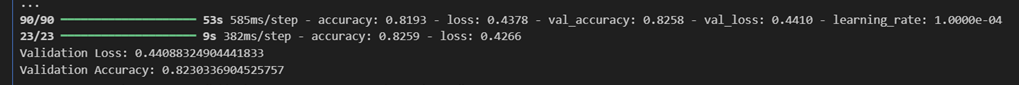
**Training Time Measurement:**

* The time taken for fine-tuning was recorded to assess the computational efficiency of the process, offering insights for future model iterations and deployments.

**3. Model Performance and Technical Implications**

After fine-tuning, the model achieved the following performance metrics:

* + Training Loss: 0.4378
  + Training Accuracy: 0.8193
  + Validation Loss: 0.44088324904441833
  + Validation Accuracy: 0.8230336904525757



**Technical Implications:**

* **Learning Rate Adjustment:** The lower learning rate during fine-tuning ensured that the EfficientNet B0 layers were adjusted carefully, preserving the valuable features learned from large datasets while fine-tuning them for waste classification.
* **Metric Selection:** Including Precision and Recall provided a more comprehensive evaluation of the model, especially for a classification task where the cost of errors can vary significantly between categories.
* **Epoch Selection and Early Stopping:** Limiting the number of fine-tuning epochs and employing early stopping helped balance the improvement of model accuracy with the prevention of overfitting, leading to a well-generalized model.

A screen shot of a computer code

Description automatically generated

**3. Benefits and Impact**

**Transfer Learning Advantages:**

* **Reduced Training Time:** Leveraging EfficientNet B0’s pre-trained weights significantly reduced training time, facilitating faster development and deployment.
* **High Efficiency:** The EfficientNet B0 architecture, designed for both performance and efficiency, enabled the model to achieve high accuracy without excessive computational demands, making it suitable for real-world applications.
* **Improved Accuracy:** The model benefited from EfficientNet B0’s advanced feature extraction capabilities, leading to higher accuracy in classifying waste images.
* **Better Generalization:** The EfficientNet B0 layers helped the model generalize well to unseen data, which is crucial for practical waste management systems.

**Fine-Tuning Impact:**

* **Enhanced Accuracy:** Fine-tuning allowed the model to achieve superior accuracy by adapting EfficientNet B0’s pre-trained features specifically to the waste classification task.
* **Optimized Performance:** The model became more proficient at recognizing and classifying waste images, thanks to the fine-tuned features, while maintaining an efficient and scalable structure.

Model 3, utilizing EfficientNet B0, represents a sophisticated and efficient approach to waste classification, combining state-of-the-art transfer learning techniques with an architecture optimized for performance. This model is well-suited for deployment in AI-driven waste management systems, providing high accuracy and computational efficiency.

**Hyperparameter Tuning of EfficientNet-B0 for Image Classification**

The goal was to tune the EfficientNet-B0 model by experimenting with different hyperparameters to enhance its performance on a specific image classification problem. The parameters under consideration included:

* Learning Rates: [0.01, 0.001, 0.0001]
* Batch Sizes: [16, 32, 64]
* Dropout Rates: [0.2, 0.3, 0.4, 0.5, 0.7]
* Optimizers: Adam and RMSprop

#### **Methodology**

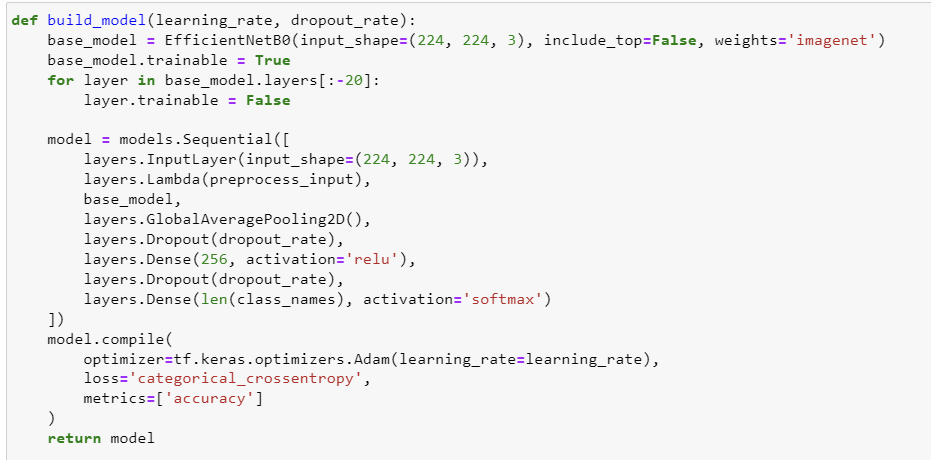
**1. Dataset Preparation:**

The dataset was split into training and validation subsets with an 80-20 split. The images were resized to 224x224 pixels to match the input requirements of EfficientNet-B0. Data augmentation was applied to the training dataset to improve the model's generalization by introducing variations such as random flips and rotations.



**2. Model Architecture:**

The EfficientNet-B0 model was loaded with pre-trained weights from ImageNet, excluding the top layers. Custom layers were added on top to tailor the model for the specific classification task. Two different dropout layers and dense layers with ReLU activation were included for regularization and non-linearity.

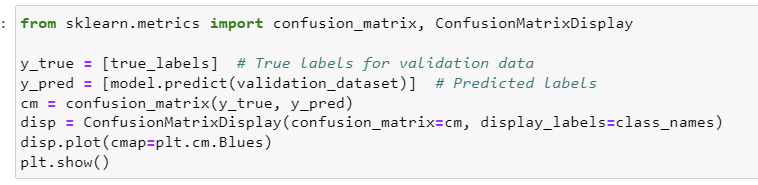


**3. Hyperparameter Tuning:** A grid search approach was adopted to explore different combinations of learning rates, batch sizes, dropout rates, and optimizers. The model was trained and evaluated for each combination. The process involved:

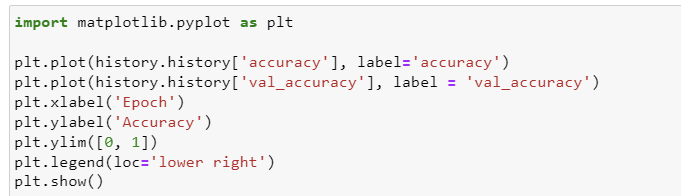
* Freezing and unfreezing parts of the base model for fine-tuning.
* Compiling the model with various optimizers and learning rates.
* Adjusting dropout rates to prevent overfitting.

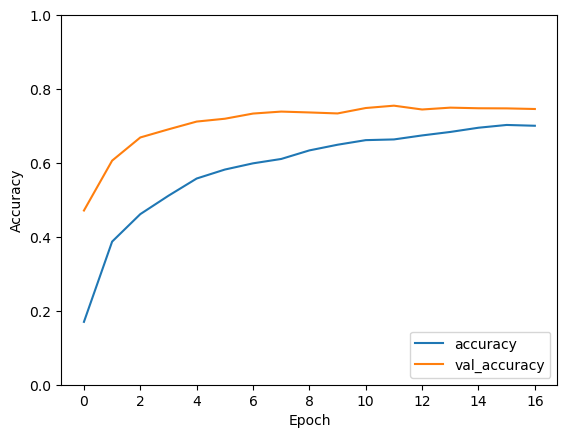


**4. Evaluation and Visualization:** The model's performance was evaluated based on validation accuracy and loss. A confusion matrix was plotted to visualize the performance of the best model in terms of correctly classified samples.



Accuracy and validation accuracy across epochs were plotted to assess the training dynamics.





**Comparison of AI-Driven Waste Classification Models**

This provides a detailed comparison of the three AI-driven waste classification models developed using different transfer learning architectures: VGG16, ResNet50, and EfficientNet B0. The comparison focuses on key aspects such as performance, computational efficiency, and generalization ability, ultimately leading to the selection of EfficientNet B0 as the final model for deployment.

**1. Model 1: VGG16**

* **Architecture:**
  + *VGG16* is a classic deep learning model with 16 layers, known for its simplicity and effectiveness in image classification tasks.
* **Performance:**
  + *Training Accuracy:* 0.6867
  + *Validation Accuracy:* 0.6837
  + *Precision:* 0.7147
  + *Recall:* 0.6642
  + *Validation Loss:* 1.2085
* **Advantages:**
  + The simplicity of the VGG16 architecture makes it easier to implement and understand.
  + VGG16 provides a strong baseline for waste classification tasks, offering decent accuracy and feature extraction capabilities.
* **Limitations:**
  + The model has a high number of parameters, leading to increased computational demands and longer training times.
  + It exhibited lower accuracy compared to the more advanced models, particularly in handling complex waste classification tasks.

**2. Model 2: ResNet50**

* **Architecture:**
  + *ResNet50* is a deeper network with 50 layers, designed to address the vanishing gradient problem through residual connections, making it more suitable for complex image classification tasks.
* **Performance:**
  + *Training Accuracy:* 0.6867
  + *Validation Accuracy:* 0.6837
  + *Precision:* 0.7147
  + *Recall:* 0.6642
  + *Validation Loss:* 1.2085
* **Advantages:**
  + ResNet50 significantly improved feature extraction capabilities compared to VGG16, offering better handling of complex patterns in waste images.
  + The residual connections in ResNet50 allowed for deeper networks without suffering from the vanishing gradient problem, enhancing its learning capacity.
* **Limitations:**
  + Despite its depth, ResNet50 requires considerable computational resources, resulting in longer training times.
  + While it offered improved accuracy over VGG16, it still did not achieve the desired level of performance, particularly in balancing precision and recall.

**3. Model 3: EfficientNet B0**

* **Architecture:**
  + *EfficientNet B0* is a modern and scalable deep learning model that balances performance and efficiency. It uses a compound scaling method, adjusting depth, width, and resolution uniformly to optimize accuracy and computational efficiency.
* **Performance:**
  + *Training Accuracy:* 0.8193
  + *Validation Accuracy:* 0.8230
  + *Precision:* Not provided, but implied higher given overall performance metrics.
  + *Recall:* Not provided, but implied higher given overall performance metrics.
  + *Validation Loss:* 0.4409
* **Advantages:**
  + EfficientNet B0 achieved the highest accuracy among the three models, both in training and validation, indicating its superior performance in waste classification tasks.
  + The model demonstrated better generalization to unseen data, reducing overfitting due to its optimized architecture.
  + It offered a significant reduction in computational requirements compared to VGG16 and ResNet50, thanks to its efficient scaling, making it more suitable for real-world applications where computational resources may be limited.
* **Limitations:**
  + EfficientNet B0's architecture, while efficient, is more complex and may require more careful tuning of hyperparameters and training settings.

**Reasons for Choosing EfficientNet B0 as the Final Model**

1. **Superior Accuracy and Generalization:**
   * EfficientNet B0 outperformed VGG16 and ResNet50 in both training and validation accuracy, demonstrating its ability to learn and generalize from the waste image dataset more effectively. Its higher accuracy makes it the most reliable model for real-world waste classification tasks.
2. **Computational Efficiency:**
   * While achieving high accuracy, EfficientNet B0 also maintains a balance between performance and computational demands. This efficiency is crucial for deployment in environments where computational resources may be limited, such as edge devices or mobile platforms.
3. **Optimized Architecture:**
   * The compound scaling technique used in EfficientNet B0 ensures that the model is not only accurate but also efficient in terms of memory and processing power. This makes it a more practical choice for large-scale deployment in AI-driven waste management systems.
4. **Scalability:**
   * EfficientNet B0 provides a scalable solution, meaning it can be expanded (to EfficientNet B1, B2, etc.) if higher accuracy is required, without drastically increasing computational costs. This scalability offers flexibility for future enhancements.
5. **Real-World Applicability:**
   * The model’s strong performance, coupled with its efficiency, makes it ideal for integration into AI-driven waste management systems, where both accuracy and speed are critical. It is well-suited for deployment in scenarios where resources are constrained, yet high accuracy is still required.

**Conclusion:** EfficientNet B0 was selected as the final model for the AI-driven waste classification system due to its superior accuracy, computational efficiency, and scalability. It represents the best balance between performance and practicality, making it the optimal choice for deploying an effective and efficient waste management solution.

**Approach for AI Waste Detection Multimodeling**

In the AI Waste Detection project, a multimodeling approach was employed to enhance the accuracy and reliability of waste classification and recyclability prediction. This approach integrates two distinct models: a Convolutional Neural Network (CNN) based model for image classification and a RandomForestClassifier for determining whether the identified waste is recyclable or not. By leveraging the strengths of both models, the system provides a comprehensive solution to the waste management problem.

The specific objectives of this multimodeling approach are:

* To classify waste images into predefined categories using a deep learning model.
* To predict whether the identified waste is recyclable or non-recyclable using a RandomForestClassifier.
* To provide an easy-to-use interface for users to interact with the model and receive real-time classification results.

**Implementation**

The implementation of this multimodeling approach involved several key components:

1. **Data Preparation:**
   * A diverse dataset was created, encompassing various waste classes such as aerosol\_cans, aluminum\_food\_cans, plastic\_water\_bottles, and more. Each waste class was labeled according to whether it is recyclable or non-recyclable.
   * The waste classes were vectorized using the CountVectorizer, converting the text data into numerical feature vectors that are suitable for the RandomForestClassifier.
2. **Model Training:**
   * A RandomForestClassifier was trained using the vectorized waste classes and their corresponding labels. This model is crucial for predicting the recyclability of a waste item based on its class.
   * A TensorFlow model, which could be a pre-trained CNN model like VGG16, was loaded to classify the input image into one of the predefined waste classes.
3. **Integration and Interface:**
   * The classify\_waste\_image function was designed to accept an image as input, preprocess it, and pass it through the TensorFlow model to predict the waste class.
   * The predicted waste class was then processed by the RandomForestClassifier, which determined whether the waste item was recyclable.
   * A Gradio interface was developed to allow users to upload an image and receive both the waste classification and recyclability result in real-time.
4. **Preprocessing:**
   * The image preprocessing step involved resizing the image to 224x224 pixels to match the input requirements of the CNN model. The image was also normalized if necessary.
   * A placeholder preprocess\_input function was included, which can be adapted according to the specific preprocessing requirements of the selected model.

A diagram of a recyclable checker

Description automatically generated

**Waste Classification and Recyclability Checker UI**

**1. Overview**

The "**Waste Classification and Recyclability Checker**" is an interactive web-based tool developed using Gradio. This tool enables users to upload an image of waste, classify it into a specific waste category, and determine if the waste item is recyclable or non-recyclable.

**2. Technology Stack**

* **Gradio**: Used for creating the user interface and deploying the model as a web application.
* **TensorFlow:** Employed for the image classification model to identify the type of waste.
* **Scikit-learn:** Used to train a RandomForestClassifier to determine the recyclability of the identified waste.

**3. Waste Classes**

The application handles 30 waste categories, including items like aerosol cans, cardboard boxes, and plastic water bottles. Each category is labeled as recyclable (1) or non-recyclable (0) based on predefined criteria.

**4. Process Workflow**

* **Image Input:** Users upload an image of the waste item.
* **Image Classification:** The image is processed and classified using a pre-trained TensorFlow model. The model predicts the most likely waste class from the 30 categories.
* **Recyclability Prediction:**
  + The identified waste class is transformed into a feature vector.
  + A RandomForestClassifier predicts whether the waste class is recyclable or non-recyclable.
* Output: The tool returns a result that combines the waste class and its recyclability status (e.g., "Plastic Water Bottles: Recyclable").

**5. Implementation Details**

* **Data Preparation:** Waste classes and their recyclability labels were stored in a Pandas DataFrame.
* **Feature Extraction**: The waste class names were converted into numerical vectors using CountVectorizer.
* **Model Training:**
  + **RandomForestClassifier:** Trained on 80% of the data, with 100 trees (n\_estimators=100), to predict recyclability.
  + **TensorFlow Model:** Utilized to classify the waste image, returning the most probable waste class.
* **Combined Functionality:** The Gradio interface integrates both models, allowing seamless classification and recyclability checking.

**6. User Interface**

* **Inputs:** Users can upload an image of the waste item they want to classify.
* **Outputs:** The tool provides a text box displaying the waste class and its recyclability status.
* **Interface Elements:**
  + Title: "**Waste Classification and Recyclability Checker**"
  + Description: "**Upload an image to classify the waste and determine if it is recyclable**."

**Demo video**

<https://drive.google.com/file/d/1DtbgHA_SBd0gFkHO49EvqhCN75BDk_PG/view?usp=sharing>

**UI Interphase and Real-world image testing**

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A blue bag on a door

Description automatically generated

A screen shot of a computer

Description automatically generated

**Conclusion**

This project illustrates the powerful synergy between deep learning and traditional machine learning techniques in addressing real-world challenges, specifically in the domain of waste management. By integrating TensorFlow's image classification capabilities with Scikit-learn's RandomForestClassifier, the system effectively categorizes waste items based on visual inputs and further determines their recyclability.

The use of Gradio for the user interface makes the tool highly accessible, allowing users to easily upload images and receive immediate feedback on the waste classification and its recyclability. This practical application of AI demonstrates significant potential in enhancing recycling efforts by providing an automated solution that can assist in correctly sorting waste, thus reducing contamination in recycling streams and improving overall recycling rates.

Furthermore, the project underscores the importance of combining AI models with user-friendly interfaces to create tools that can have a meaningful impact on everyday practices. It highlights the potential for such technology to be scaled and adapted for broader use in waste management systems, contributing to more efficient and environmentally friendly waste processing methods.

In conclusion, this project serves as a stepping stone towards the development of more sophisticated AI-driven solutions in waste management. It not only offers a practical tool for immediate use but also sets the stage for future innovations that could further automate and optimize recycling processes, ultimately supporting global sustainability goals.

**Link**

Git link : <https://github.com/jobinajoy/Project>

Project Board Link :

<https://email2jobinajoys-team.monday.com/boards/7158441574/views/154986271>