Group 3 Report: TrashNet

Jupiter Kang, Long Peter Huynh, Stephen Davanzo
DATS 6303
Prof. Amir Jafari

1. Introduction

Group 3 undertook the task of enhancing an existing image classification framework, TrashNet, to improve its effectiveness in garbage sorting and recognition. Our primary objective was to advance the baseline pre-trained models through transfer learning, integrate modern architectures, and develop an application that demonstrates accurate trash classification. This report provides a comprehensive overview of the TrashNet dataset, the theoretical background of the neural networks employed, our experimental methodology, results, and key conclusions..

2. Data Description

The TrashNet dataset, developed by Gary Thung and Mindy Yang at Stanford University, is a curated collection of waste images designed for machine learning applications in recycling and waste classification. It comprises approximately 2,527 RGB images, each with a resolution of 512×384 pixels, captured against white or cardboard backgrounds to reduce background noise. The dataset is categorized into six distinct classes of waste materials: cardboard (403 images), glass (501 images), metal (410 images), paper (594 images), plastic (482 images), and trash (137 images). The uneven distribution of images across classes—particularly the underrepresentation of the "trash" class—introduces a class imbalance challenge that must be addressed during model training.

In their 2016 white paper, Thung and Yang constructed a Convolutional Neural Network (CNN) from scratch rather than leveraging pre-trained models. However, with only ~2,500 images, their dataset was insufficient for effective CNN training. Their model, loosely inspired by AlexNet, was too shallow to generalize well on such a small dataset. Furthermore, the original study lacked advanced techniques such as robust data augmentation, hyperparameter tuning, and class balancing. Interestingly, the researchers achieved better performance with a Support Vector Machine (SVM), attaining 66% accuracy—a benchmark we aimed to surpass.

3. Algorithms and Networks

a. Baseline Code

Our baseline code, sourced from a Kaggle competition utilizing the TrashNet dataset, trained six pre-trained networks and identified the highest-performing model. However, it missed

opportunities for optimization, such as implementing robust data augmentation, fine-tuning base models by unfreezing layers, and scheduling learning rates.

Model Name	Year Released	Key Info
MobileNetV2	2018	Fast, lightweight, inverted residuals
ResNet101V2	2016	Deep (101 layers), pre-activation, good feature extraction
ResNet152V2	2016	Very deep (152 layers), great extractions, slow
MobileNet (V1)	2017	Lightweight, fast, depthwise convolutions
MobileNetV3Small	2019	Ultra-efficient, built for small compute power
MobileNetV3Large	2019	Balanced options for speed and accuracy

b. Network Selection

We evaluated six pre-trained networks, with a particular focus on modern architectures suited for small datasets. Among these, EfficientNetV2S emerged as a standout performer due to its efficiency and effectiveness. Unlike heavier models like ResNet152V2, which has 60 million parameters across 152 layers and tends to overfit on small datasets, EfficientNetV2S offers over 300 layers with only 24 million parameters. This efficiency stems from its use of Mobile Inverted Bottleneck Convolution (MBConv) and Fused-MBConv layers. MBConv layers employ an inverted structure (narrow → wide → narrow), using 1x1 convolutions to compress data, 3x3 depthwise convolutions to expand it, and another 1x1 convolution to reduce it again. Fused-MBConv layers, unique to EfficientNetV2S, streamline this process by merging the expansion and spatial convolution steps, enhancing computational efficiency.

Stage	Operator	Stride	#Channels	#Layers
0	Conv3x3	2	24	1
1	Fused-MBConv1, k3x3	1	24	2
2	Fused-MBConv4, k3x3	2	48	4
3	Fused-MBConv4, k3x3	2	64	4
4	MBConv4, k3x3, SE0.25	2	128	6
5	MBConv6, k3x3, SE0.25	1	160	9
6	MBConv6, k3x3, SE0.25	2	256	15
7	Conv1x1 & Pooling & FC	-	1280	1

EfficientNetV2S further benefits from advanced regularization techniques, including stochastic depth, label smoothing, and automated image augmentation, making it an ideal choice for TrashNet classification.

4. Experimental Setup

First, we slightly changed the test split from the baseline code to 20%. However, since our dataset is not big enough and we have to use all we have, the original baseline did a second ImageDataGenerator, in which it has its own train-val 75/25. We didn't figure out a way to solve this problem, leaving it as a potential data leakage problem, due to the

independent application of val_split by each generator on the full dataset, and could lead the possible test samples included in either train or validation set of the second generator, as well as leading overlap and miscounts

The baseline code employed a simplistic transfer learning approach: it loaded the TrashNet dataset using ImageDataGenerator with minimal augmentation, trained pre-trained models like ResNet and MobileNet for a fixed number of epochs, and used default optimizer settings. This approach limited its potential, risking undertraining or overfitting. Our enhancements focused on improving regularization, introducing additional hyperparameters, and refining the training strategy to boost model performance.

.

- **Two-Phase Training Strategy**: We adopted a two-phase training approach. In the first phase (initial training), we froze the pre-trained base model and trained only the newly added classifier head (a dense layer). In the second phase (fine-tuning), we unfreeze all layers except the top fine_tune_layers (e.g., 20 or 50), recompiled the model with a reduced learning rate (fine_tune_lr), and trained for additional fine_tune_epochs. This fine-tuning phase allowed the model to adapt the pre-trained weights to the TrashNet dataset, with a low learning rate preventing drastic disruptions to the pre-trained weights.
- **Hyperparameter Enhancements**: We introduced distinct hyperparameters for each training phase, including separate epochs and learning rates, a dropout rate, and an early stopping patience parameter. This granular control enabled precise adjustments across training stages, enhancing model performance compared to the baseline's rigid settings.
- **Regularization Techniques**:
    - **Dropout**: We added a dropout layer (with a 50% rate) after the dense layer in the classifier head. By randomly disabling 50% of neurons during training, dropout forced the model to learn more robust representations by distributing learning across all neurons, mitigating overfitting by preventing over-reliance on specific neurons.
    - **Early Stopping**: We implemented early stopping to monitor validation loss, halting training if no improvement was observed after a specified patience period. This not only prevented overfitting but also saved training time and restored the best-performing weights.
    - **Data Augmentation**: We enhanced data augmentation in the ImageDataGenerator by adding transformations such as shear_range and brightness_range. These random transformations increased dataset diversity, improving the model's robustness to variations and further reducing overfitting.
- **Class Weighting**: To address class imbalance in the TrashNet dataset (e.g., 594 images for "paper" vs. 137 for "trash"), we implemented class weighting in the loss function. Weights were calculated using the formula:

$$weight\ i\ = \frac{total\ samples}{number\ of\ classes \times samples\ in\ class\ i}$$

We adjusted these weights with an alpha factor of 0.2:

$$adjusted\ weight\ \ i\ = 1.0\ +\ \alpha\times(weight\ i\ -\ 1.0)$$

This adjustment ensured the model did not overly prioritize majority classes, improving performance on minority classes and reducing overfitting.

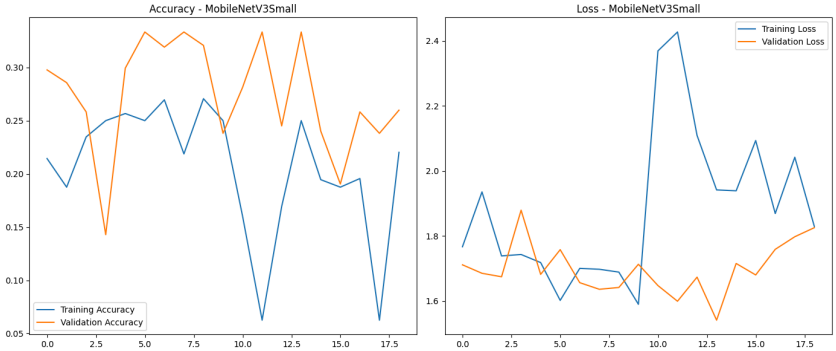
- **Optimizer Selection**: We experimented with optimizers like Adam and AdamW. While AdamW offers decoupled weight decay for potentially better regularization, its performance in practice was suboptimal for our dataset. Consequently, we relied on the Adam optimizer, focusing on other regularization techniques to enhance performance.

5. Results

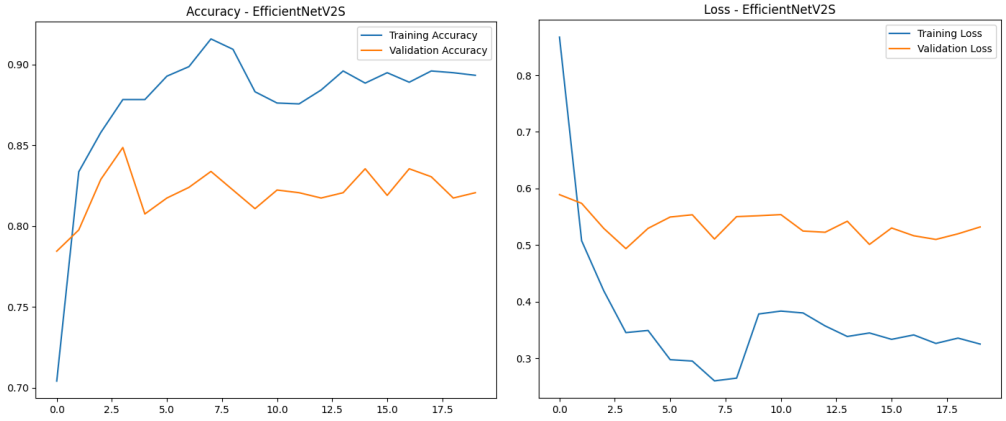
Our iterative improvements significantly enhanced the performance of the TrashNet classification models, surpassing both the baseline code and the original 2016 study. A detailed comparison of model performance is presented in the table below, highlighting the advancements achieved relative to the baseline code.

Model	Our Result	Baseline Result	Difference
EfficientNetV2	86.25%	/	/
ResNet152V2	80.62%	78.93%	1.69%
ResNet101V2	80.21%	77.14%	3.07%
MobileNetV2	77.29%	75.35%	1.94%
MobileNet	75.21%	75.15%	0.06%
MobileNetV3Large	44.79%	44.53%	0.26%
MobileNetV3Small	26.04%	35.59%	-9.55%

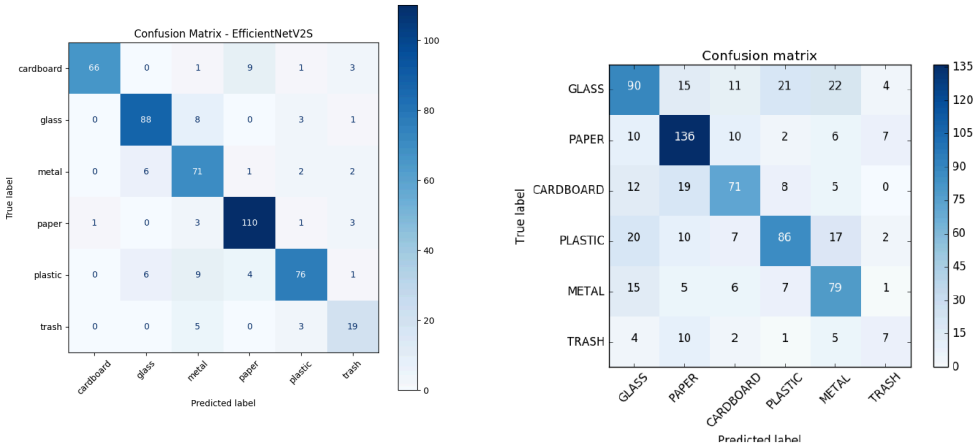
Our highest-performing model, EfficientNetV2S, achieved an accuracy of ~86%, a 9.25% improvement over the baseline's best model. Most legacy models also showed steady improvements, except for MobileNetV3Small, which exhibited unstable learning due to the new hyperparameters and tuning strategy



The learning curves for EfficientNetV2S indicated a stable model with slight overfitting, a challenge attributed to the small dataset size (~2,500 images). Despite this, the ~86% accuracy aligns with industry standards and demonstrates practical utility. Compared to the original researchers' confusion matrix, which showed poor class differentiation, our approach—leveraging transfer learning, modern architectures, and effective tuning—produced a balanced confusion matrix, highlighting significant improvements in classification performance.



The confusion matrix for EfficientNetV2S, shown in the following Figure 1, highlights the model's robust performance on the TrashNet dataset, achieving an overall accuracy of ~86% with strong diagonal values: cardboard (66/81, 81.5%), glass (88/100, 88.0%), metal (71/88, 80.7%), paper (110/122, 90.2%), plastic (76/89, 85.4%), and trash (19/27, 70.4%), though the smaller trash class struggles due to class imbalance (137 images vs. 594 for paper), mitigated by class weighting (alpha=0.2); misclassifications reveal patterns such as cardboard confused with paper (9 instances), glass with plastic (6), and metal with glass (6), reflecting shared visual traits like texture or reflectivity, while the balanced performance across classes—compared to the original 2016 study's poor differentiation—demonstrates the efficacy of transfer learning, modern architectures, and tuning



6. Conclusions

EfficientNetV2S emerged as the top performer, owing to its modern architecture that balances depth (over 300 layers) with a modest parameter count (24 million). This efficiency, combined with advanced regularization techniques, enabled robust generalization on the small TrashNet dataset. While slight overfitting persists due to the dataset's limited size, our results demonstrate that proper transfer learning, hyperparameter tuning, and regularization can substantially improve classification performance.

7. Problems and Future Directions

The small size of the TrashNet dataset presents a unique challenge, as deep learning typically benefits from larger datasets. This limitation increases the risk of overfitting, as the model may memorize the data rather than generalize from it. To address this, we propose the following future directions:

- **Synthetic Data Generation**: Employ Generative Adversarial Networks (GANs) to generate synthetic images, addressing class imbalance and increasing dataset diversity. Synthetic data could also enable a cleaner train/test/validation split, which is currently constrained by the dataset's size.
- **Multi-Item Classification**: Enhance the model to handle images with multiple items, as industrial sorting scenarios often involve multiple pieces of waste in a single frame. Integrating object detection to identify and classify recyclables within such images would improve practical applicability.

References

Thung, G., & Yang, M. (2016). *Classification of trash for recyclability status*. Stanford University. Retrieved from

https://cs229.stanford.edu/proj2016/report/ThungYang-ClassificationOfTrashForRecyclabilityStatus-report.pdf

He, J., Zhao, Y., Zaslavskiy, M., Wang, X., Lin, Z., Jin, Q., & Wang, W. Y. (2021). *TransFG: A transformer architecture for fine-grained recognition*. arXiv preprint arXiv:2104.00298. https://arxiv.org/abs/2104.00298

AdamW — PyTorch 2.7 documentation. (n.d.). https://pytorch.org/docs/stable/generated/torch.optim.AdamW.html

Yassin, A. (2024, November 13). Adam vs. AdamW: Understanding Weight Decay and Its Impact on Model Performance. *Medium*. https://yassin01.medium.com/adam-vs-adamw-understanding-weight-decay-and-its-impact-on-model-performance-b7414f0af8a1

Tripathi, S., Augustin, A. I., Sukumaran, R., & Kim, E. (2022). MBConv and fused-MBConv blocks extensively utilized by EfficientNetV2 CNNs. In *HematoNet: Expert Level Classification of Bone Marrow Cytology Morphology in Hematological Malignancy with Deep Learning* [Figure 3]. ResearchGate. https://www.researchgate.net/figure/MBConv-and-fused-MBConv-blocks-extensively-utilized-by-EfficientNetV2-CNNs-Fused-MBConv_fig3_360226441