Garbage Classification Based on MobileNet V3

MingJie Yin
1023040815
Nanjing University of Posts and Telecommunications
School of Computer Science
Nanjing, China

Abstract – The waste problem is increasingly becoming an urgent environmental challenge to be solved. This paper emphasizes the importance of waste classification as an effective resource management method to achieve sustainable development and environmental protection. This paper introduces the background, importance and problems of garbage classification in detail, and focuses on the motivation and goal of the course design of garbage classification system based on MobileNet V3. Through this design, this paper aims to realize efficient and accurate garbage sorting on mobile devices, so as to contribute to improving the environment.

Keywords – Garbage classification, MobileNet, deep learning

I. INTRODUCTION

With the continuous development of society and the acceleration of urbanization, the problem of garbage has become an urgent environmental challenge that needs to be solved. Garbage classification, as an effective resource management method, is crucial for achieving sustainable development and environmental protection[1]. This introduction will explore the background, importance, and existing issues of garbage classification, and introduce the motivation and goals of building a garbage classification system based on MobileNet V3 in this course design. Through this design, this article aims to achieve efficient and accurate garbage classification on mobile devices, contributing to improving the environment.

II. DATA PREPROCESSING

A. Preparation of Data

The dataset of this paper is called the TrashBig dataset, which mainly comes from the ImageNet dataset. ImageNet is a large-scale image database and one of the most widely used datasets in the field of computer vision. This dataset consists of a rich variety of images, covering over 1000 object categories. The most famous version of the ImageNet database is the version used in the ImageNet Large Scale Visual Recognition Challenge (ILSVRC). This dataset was carefully planned through independent collection and includes 12 different subcategories, each belonging to four main types of waste: recyclable waste, kitchen waste, hazardous waste, and other waste. The dataset includes 12 subcategories including bananapeel, battery, cardboard, clothes, drinks, glass, lightbulb, metal, paper, papercup, plastic, and vegetable. Each subcategory is carefully labeled to facilitate the training and evaluation of machine learning models.

B. Data processing

The preprocessing process of the TrashBig dataset aims to ensure the consistency of image data and improve the model's ability to extract meaningful features. Using the PyTorch library for processing, first use transformers Resize (256) adjusts all images to the same size to ensure that the model receives input with consistent spatial size during training. Subsequently, through transformers RandomCrop (224) introduces random cropping to increase the diversity of training data. Transformers RandomHorizontalFlip() introduces random horizontal flipping to better generalize the model to objects in different directions[2][3]. Next, transformers ToTensor() converts images into PyTorch tensors and normalizes them, which helps the model learn more easily. Finally, through transformers Normalize() standardizes images to ensure that the mean and variance of input data are close to predefined values, improving model stability. The selection of these steps is based on the best practices of image processing and deep learning model training, aimed at improving model performance and generalization ability.

III. RELATED WORK

After deciding to do the task of garbage classification, this paper investigated some network structures to do garbage classification. In the field of garbage classification, InceptionV2, also known as MobileNetV1, is highly praised for its effective use of 1x1 convolution and Inception modules in garbage classification tasks. Its lightweight structure ensures real-time classification on resource limited devices. The MnasNet series architecture is designed specifically for mobile and edge devices, and its different variants (MnasNet-A and MnasNet small) are suitable for different model sizes, providing flexible choices for garbage classification in different scenarios. The MobileNet series, especially MobileNetV2 and MobileNetV3, have achieved widespread success in garbage classification with deep separable convolutions and innovative architectures, with MobileNetV3 taking an important step towards improving accuracy and efficiency. NasNet adopts neural architecture search, providing a powerful tool for automatically discovering efficient network structures, and is widely used to optimize garbage classification network structures. ProxylessNAS simplifies the process of optimizing neural network structures through a proxy free neural architecture search method, making a significant contribution to the discovery of efficient garbage classification models. As a representative of the ResNet family, ResNet-50 plays a multifunctional role in garbage classification tasks with its residual learning framework, providing a robust foundation for feature extraction. ShuffleNetV2 further reduces computational costs through channel shuffling and group convolution, providing a feasible solution for real-time garbage classification on resource constrained devices[4]. Its efficient design is crucial for task processing. These works together build a diverse garbage classification network architecture pool, providing researchers with rich options for selecting appropriate models in different scenarios.

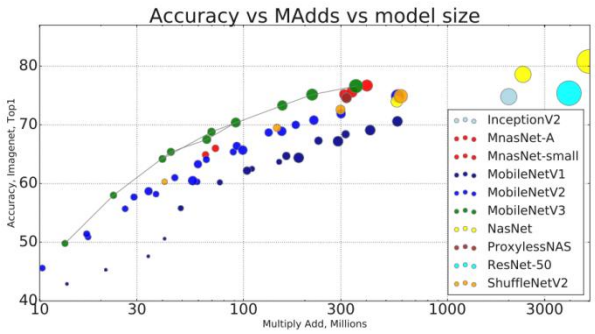

Fig 1 Accuracy,Madds and model size

This graph illustrates the trade-off between computational complexity (MAdds), model size, and Top-1 accuracy among multiple deep learning models (InceptionV2, MnasNet-A, MnasNet small, MobileNetV1, MobileNetV2, MobileNetV3, NasNet, ProxylesNAS, ResNet-50, ShuffleNetV2). Specifically, the figure shows the performance comparison of these models on different hardware or software frameworks, with a focus on the trade-off between MAdds (number of multiplication and accumulation operations) and Top-1 accuracy. This article introduces the method we adopted using MobileNetV3 large and small models to provide the next generation of high-precision and efficient neural network models to support computer vision on devices. The new network has driven the advancement of the most advanced technology and demonstrated how to combine automatic search with novel architectural advancements to build effective models.

A. Traditional Convolutional Networks

The mobile mode has been established on increasingly efficient building modules. MobileNetV1 introduces depthwise separable convolution as an effective alternative to traditional convolutional layers. Deep separable convolution effectively decomposes traditional convolutions by separating spatial filtering from feature generation mechanisms. Deep separable convolution is defined by two independent layers: lightweight deep convolution for spatial filtering and heavier 1x1 point convolution for feature generation. MobileNetV2 introduces linear bottlenecks and reverse residual structures to construct more effective layer structures by leveraging the low rank nature of the problem. This structure is defined by 1x1 extended convolution, followed by depth convolution and 1x1 projection layer. When and only when they have the same number of channels, the input and output are connected using residual connections. This structure maintains a compact representation in both input and output, while expanding internally to higher dimensional feature spaces to increase the expressiveness of nonlinear cross channel transformations. MnasNet introduces a lightweight attention module based on squeezing and stimulation into the bottleneck structure on the basis of MobileNetV2 architecture[7][8]. Please note that compared to the proposed ResNet based module, the extrusion and excitation modules are integrated in different locations. This module is placed after the deep filter in the extension to focus attention on the maximum representation.

B. Efficient Mobile Building Blocks

For MobileNetV3, we use the combination of these layers as building blocks to build the most efficient model. The layer has also been upgraded to modify swish nonlinearity. Squeeze, excitation, and sway nonlinearity all use S-shaped curves, which have low computational efficiency and are difficult to maintain accuracy in fixed-point algorithms. Therefore, we replace them with hard S-shaped
curves.

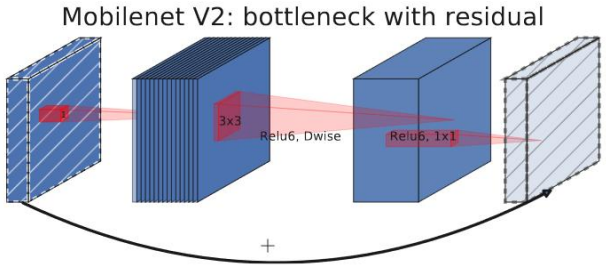

Fig 2 Mobilenet V2 blocks

MobileNetV2 layer (reverse residual and linear bottleneck). Each block is composed of narrow inputs and outputs (bottlenecks) that are not nonlinear, and then extended to a higher dimensional space and projected onto the output. The remaining connection bottleneck (rather than expansion).

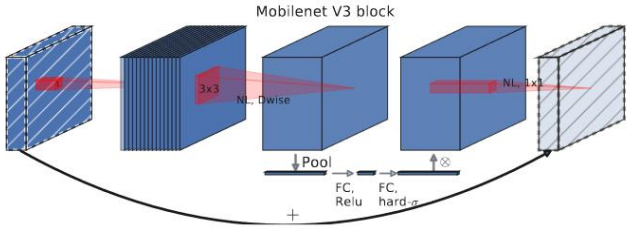

Fig 3 Mobilenet V3 blocks

MobileNetV2+Squeeze and Excitation. We apply compression and excitation in the residual layer. We use different nonlinearities based on different layers.

IV. GARBAGE CLASSIFICATION

After selecting the method and dataset, deep learning can begin to train the dataset.

A. Training steps

Firstly, define the various configuration parameters required during the training and inference process. It includes directory path, dataset information, and hyperparameters such as epochs, batch size, learning rate, etc. Through these parameters, this article can easily adjust the training settings and other important parameters of the model. Then, by importing libraries such as PyTorch, Torchvision, OpenCV, etc., the program obtained the basic tools needed to build deep learning models. In addition, a custom MobileNetV3 model (MobileNetV3-Small) and a custom dataset class (TrashSet) specifically designed for processing garbage classification datasets were introduced. In order to flexibly select computing devices, two functions were defined in the program[5]. Get_ Default_ Device will return the corresponding device based on the availability of the GPU, while MyDevice allows users to specify the use of the GPU or CPU. The model training function (train_with_valid) is

the core of the entire script. It takes an instance of the TrashSet class as a parameter, initializes the model, loads pre trained weights, and trains and verifies through multiple epochs. During the training process, we use cross entropy loss function, Adam optimizer, and learning rate scheduler. The key information during the training process is recorded through Tensorboard's SummaryWriter, including loss, accuracy, and accuracy of categories and groups. At the end of each epoch, save the checkpoints of the current model. By using SummaryWriter in torch. utils. sensorboard, we can visually track metrics during training and validation processes. The training status of the model is saved at the end of each epoch for reloading when needed. In this way, we can easily interrupt and continue training during the training process. For each epoch, the script iterates the training batch, calculates the loss, and updates the model parameters. Meanwhile, evaluate the performance of the model on the validation set, calculate validation loss and accuracy. By using a step-by-step scheduler, we can dynamically adjust the learning rate during the training process to improve the convergence and performance of the model. After the entire training and validation, close the SummaryWriter on the Tensorboard, and the entire training process will come to an end. Through the above components and functions, we can establish a flexible and scalable garbage classification model training framework.

B. Training results

TensorBoard is a powerful visualization module that can be used for deep learning network model training to view model structure and training effects (prediction results, network model structure diagram, accuracy, loss curve, learning rate, weight distribution, etc.). It can help you better understand network models, design TensorBoard to call relevant code, and the above results can be saved. It is a good helper for integrating data and organizing models.
The following is a display of the results on the training set.

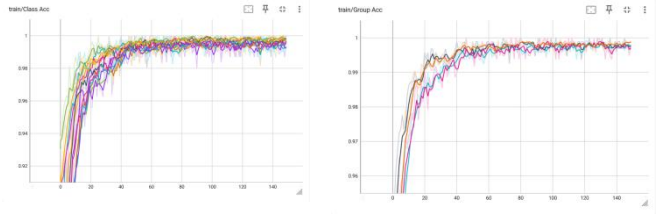

Fig 4 class acc and group acc in train

The 'train/Class Acc' curve shows the change in accuracy of the model for each category in each training epoch. By observing this curve, we can understand the learning progress of the model on different garbage categories. The train/Group Acc curve reflects the performance of the model at the group level of garbage classification tasks. Each group represents a relevant category of waste, such as recyclable waste, kitchen waste, hazardous waste, etc.

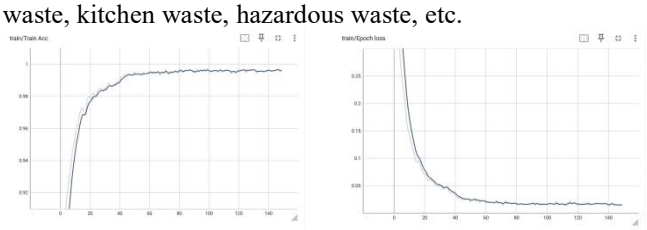

Fig 5 train acc and epoch loss in train

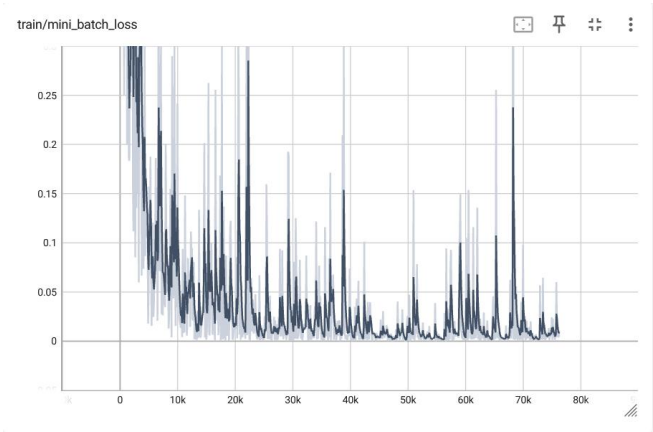

Fig 6 mini_batch_loss in train

The train/Train Acc curve shows the overall accuracy change of the model during each training epoch. This curve represents the dynamic evolution of the accuracy of the model on the training set. A gradually rising and stabilizing train/Train Acc curve indicates that the model gradually improves its fit to the training data throughout the entire training process.

The train/Epoch loss curve records the loss value at the end of each training epoch. The decrease in this curve indicates that the model gradually reduces the difference between the predicted and true labels on the training data, indicating that the decrease in loss values reflects the effectiveness of the model's learning. A stable low loss value usually indicates that the model has achieved good fitting results.
Train/mini_ Batch_ The loss curve records the loss value at the end of each small batch. The loss value for each small batch is calculated by the model when updating based on a small portion of training data, so this curve can display the learning dynamics of the model on different subsets of training data.

The following is a display of the results on the validation set.

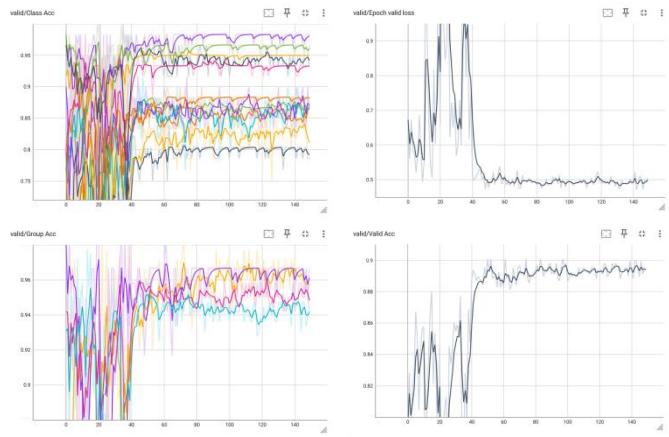

Fig 7 a display of the results on the validation set.

In the validation process of deep learning garbage classification models, we focused on a series of indicators, including valid/Class Acc (category accuracy of the

validation set), valid/Apoch valid loss (loss per validation epoch), valid/Group Acc (group accuracy of the validation set), and valid/Valid Acc (overall validation accuracy).

The valid/Class Acc curve shows the accuracy of the model for each garbage category in each validation epoch. By observing this curve, we can gain a deeper understanding of the performance of the model on different garbage categories. A steadily rising curve indicates that the model's classification of various categories is gradually improving, while fluctuations or declines may indicate learning difficulties or the need for more attention in some categories.

The valid/Epoch valid loss curve records the loss value at the end of each validation epoch. The decrease in this curve indicates that the model gradually reduces the difference between the predicted and true labels on the validation data, indicating that the decrease in loss values reflects the effectiveness of the model's learning. A stable low loss value usually indicates that the model has good generalization ability on the validation set.

The valid/Group Acc curve reflects the performance of the model at the group level of garbage classification tasks. Each group represents a relevant category of waste, such as recyclable waste, kitchen waste, hazardous waste, etc. By observing the evolution of this curve, we can understand the overall performance of the model on the overall garbage classification task. The improved valid/Group Acc representation model performs better on higher-level classification tasks.

The valid/Valid Acc curve provides an intuitive perspective for overall validation accuracy. This curve reflects the performance of the model on the entire validation set and is a comprehensive indicator of accuracy for all categories and groups.

RECOGNITION

Firstly, I would like to express my gratitude to Teacher Zou Zhiqiang for teaching us a lot of machine learning knowledge in the big data analysis course. Secondly, I would like to express my gratitude to my fellow teachers and brothers for providing me with a lot of help in my experiment. Your contribution has had a profound impact on this research.

REFERENCES

[1] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Man e. R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Vi Egas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng TensorFlow: Large scale machine learning on heterogeneous systems, 2015 Software available from tensorflow. org

[2] R. Avenash and P. Vishawanth Semantic segmentation of satellite images using a modified CNN with hard swish activation function In VISIGRAPP, 2019

[3] B. Baker, O. Gupta, N. Naik, and R. Raskar Designing neural network architectures using reinforcement learning CoRR, abs/1611.02167, 2016

[4] H. Cai, L. Zhu, and S. Han Proxylessnas: Direct neural architecture search on target tasks and hardware CoRR, abs/1812.00332, 2018

[5] L. - C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille Semantic image segmentation with deep convolutional nets and fully connected crfs In ICLR, 2015

[6] L. - C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille Deeplab: Semantic image segmentation with deep convolutional nets, around convolution, and fully connected crfs TPAMI, 2017

[7] L. - C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam Encoder decoder with around separable convolution for semantic image segmentation In ECCV, 2018

[8] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele The cityscapes dataset for semantic urban scene understanding In CVPR, 2016