Ore Image Classification Based on Improved CNN: Reproduction and Analysis

Jakub Spišák, Daniel Zemančík

Abstract

In this paper, we present a thorough replication and evaluation of a convolutional neural network (CNN)-based method originally proposed by Zhou et al. (2022) for automated ore image classification. The original methodology combined transfer learning (TL), data augmentation (DA), and Squeeze-and-Excitation Networks (SENet), achieving high classification accuracy with limited labeled data. To validate and further analyze these findings, we reimplemented the described CNN models—specifically AlexNet, VGG16, ResNet50, InceptionV3, and MobileNetV2—in PyTorch, and conducted extensive comparative experiments. Our replication experiments confirmed the key conclusions of the original study, particularly the effectiveness of TL and DA, and the significant performance benefits brought by the SENet attention module. Specifically, MobileNetV2, enhanced with SENet, exhibited the best overall performance. However, due to ambiguities in the original training details and the use of MobileNetV2 instead of MobileNetV1, our exact accuracy differed slightly, with our best model achieving around 93% accuracy compared to the original paper's reported 96.89%. We further explored backbone freezing strategies, revealing that freezing the CNN backbone for the initial epochs significantly improves SENet training stability and accuracy. These insights clarify critical methodological choices and validate the robustness of the original CNN-based approach for practical ore classification.

I. Introduction

The identification of ore types in mining is an important and time-consuming task traditionally done by experts based on visual properties (color, texture, luster). Automated image-based ore classification can greatly improve efficiency and consistency. However, training deep learning models for this task is challenging because obtaining large labeled datasets of ore images is difficult. Convolutional Neural Networks (CNNs) have achieved great success in image classification, but they typically require thousands of images per class to generalize well. With limited data, CNNs tend to overfit, resulting in poor performance.

Recent advances in transfer learning and data augmentation offer solutions to the small dataset problem. Transfer learning (TL) [1] involves fine-tuning a CNN pre-trained on a large generic dataset (such as ImageNet) for the target classification task. This leverages learned features from millions of images and can significantly boost performance on the new task with few samples. Data augmentation (DA) artificially expands the training dataset by applying transformations (e.g., flips, rotations, crops, color jittering) to create additional varied images, thereby improving the model's robustness. Zhou et al. [2] applied these techniques to the problem of ore classification. Furthermore, they introduced an attention mechanism, Squeeze-and-Excitation (SE) blocks [3], into the CNN architecture to adaptively recalibrate feature importance, aiming to further improve accuracy. Their approach is referred to as an ”improved CNN” for ore classification, which combines a CNN backbone with TL, DA, and SENet.

In the original study, the authors achieved state-of-the-art results on a small ore image dataset. This report provides a detailed summary of their methodology and findings. In addition, we describe our reproduction of the experiments using PyTorch, following the paper's approach. We present the results of this implementation, including training curves, confusion matrices, and performance metrics, and compare them with the reported results. Furthermore, we discuss the effectiveness of the approach and propose possible improvements or future work, such as exploring other attention modules (e.g., CBAM [4]), ensemble modeling, and refined augmentation.

II. Summary of Proposed Methodology and Results

A. Dataset and Problem

The ore classification task involves recognizing the type of ore mineral from images. Zhou et al. used a public dataset of ore images acquired from Kaggle [2]. The dataset contains 957 images of rocks, covering 7 different ore categories (the specific ore types were not explicitly listed in the paper, but include various mineral classes). This dataset is relatively small for training a deep CNN from scratch, which motivates the use of transfer learning and augmentation. The images were divided into training, validation, and test sets (60% for training and the rest for validation - 20% and testing - 20%). The goal was to classify each image into the correct ore type automatically.

The challenges addressed by the paper include the limited number of training samples and the potential overfitting of complex CNN models. The authors identified that conventional machine learning methods (e.g., KNN, decision trees) require manual feature extraction and do not scale well, while CNNs can automatically learn features but need sufficient data [2]. Thus, the core problem is how to effectively train a CNN for ore classification with less than a thousand images spread across seven classes.

B. CNN Architectures and Transfer Learning

To tackle the ore classification task, five well-established convolutional neural network (CNN) architectures were employed: AlexNet[5], VGG16[6], ResNet50[7], InceptionV3[8], and MobileNet[9]. These models represent a diversity of architectural styles and depths—from the relatively shallow AlexNet to the deep residual blocks of ResNet50 and the computationally efficient MobileNet family. All models were pre-trained on the ImageNet [10] dataset and used as feature extractors or were fine-tuned for our specific classification task. Leveraging transfer learning allowed us to reuse rich, hierarchical representations learned from millions of labeled natural images, making these architectures highly effective despite our limited dataset size.

In this study, three experimental regimes were used to train the CNNs: training from scratch, transfer learning (TL), and transfer learning combined with data augmentation (TL+DA).

Training from scratch: When models were trained with randomly initialized weights on the ore dataset, they exhibited poor generalization due to insufficient data. For example, MobileNet achieved only 44% test accuracy, and even deeper models failed to exceed 75%, suffering from overfitting and unstable convergence.

Transfer learning (TL): With TL, the convolutional layers were initialized using pre-trained ImageNet weights and fine-tuned on our ore dataset. This allowed the models to adapt high-level feature detectors to the mineral domain, resulting in significant improvements in performance. Fine-tuning was done using a low learning rate ($\alpha = 10^{-4}$) to prevent drastic changes to the pretrained filters. TL significantly improved model generalization and convergence speed. For instance, MobileNet improved from 44% to 82% accuracy using this method.

Transfer learning with data augmentation (TL+DA): To further improve generalization, the training dataset was expanded using image augmentation techniques such as center cropping, zooming, brightness shifts, and edge cropping. These transformations simulate real-world variability and help the model become invariant to non-semantic visual changes. While most architectures benefited from augmentation (e.g., ResNet50 and MobileNet saw increases in test accuracy from 84.9% to 87.5% and 94.8%, respectively), VGG16's performance slightly declined, potentially due to oversensitivity to local distortions introduced by aggressive augmentation.

Activation and regularization techniques: All models used the Rectified Linear Unit (ReLU) as the activation function, defined as:

$$\text{ReLU}(x) = \begin{cases} x, & \text{if } x \geq 0 \\ 0, & \text{otherwise} \end{cases} \tag{1}$$

ReLU introduces non-linearity and avoids the vanishing gradient problem while maintaining computational simplicity.
Dropout was employed between fully connected layers to prevent overfitting by randomly deactivating neurons during training. The forward pass with dropout is defined as:

$$r_j^{(l)} \sim \text{Bernoulli}(p), \quad \tilde{y}^{(l)} = r^{(l)} \cdot y^{(l)}, \quad z_i^{(l+1)} = W_i^{(l+1)} \tilde{y}^{(l)} + b_i^{(l+1)}, \quad y_i^{(l+1)} = f(z_i^{(l+1)}) \tag{2}$$

Here, $r^{(l)}$ is a binary mask sampled from a Bernoulli distribution with probability p, and $\cdot$ denotes element-wise multiplication. A dropout probability of 0.5 was chosen based on empirical tuning.
Normalization and standardization were applied during image preprocessing to improve numerical stability and ensure convergence. Each pixel value was normalized to the $[0, 1]$ range and then standardized using:

$$x_i = \frac{x_i - \mu}{\sigma} \tag{3}$$

where μ and σ are the dataset-wide mean and standard deviation for each color channel, respectively.
Softmax and loss function: The output layer employed the softmax activation to produce class probabilities:

$$p_i = \frac{e^{z_i}}{\sum_j e^{z_j}} \tag{4}$$

where z_i is the i-th logit (raw score) from the model output. The final model was trained using the cross-entropy loss, which measures the divergence between the predicted distribution q and the true label distribution p:

$$H(p \| q) = -\sum_i p(i) \log_2 q(i) \tag{5}$$

Cross-entropy is widely used in classification tasks because it penalizes confident but incorrect predictions heavily, driving the model to output accurate and confident results. The loss function is minimized using the Adam optimizer[11], which adaptively adjusts learning rates per parameter and accelerates convergence [2] [3].

The integration of transfer learning, data augmentation, dropout, and standardization helped address the core challenges of this task—overfitting and data scarcity. Among all tested CNN architectures, MobileNet (particularly in its second version with SENet) achieved the highest performance, confirming its strength in low-resource image classification tasks.

C. *Incorporating Squeeze-and-Excitation (SE) Attention*

After establishing MobileNet as the best base model when using TL and DA, Zhou et al. further improved the classification performance by integrating a Squeeze-and-Excitation Network (SENet) module [3] into the architecture. SENet is an attention mechanism that enhances important features by reweighting the feature channels adaptively and the architecture is further explained in Sec. IV-D.

In the improved model, the authors took the feature maps produced by the MobileNet CNN and fed them into an SE block, then into the final classifier. This hybrid MobileNet+SENet model is the "improved CNN" highlighted in the paper. Training this model (using the features from MobileNet and fine-tuning with the SE module) led to a new state-of-the-art result: the classification accuracy reached **96.89%** on the test set [2]. This was about 2% higher than using MobileNet with the standard softmax classifier (94%), demonstrating the efficacy of the SE attention in squeezing out additional performance.

Overall, the original research concluded that a CNN-based approach with transfer learning and data augmentation can effectively classify ores with high accuracy even with a limited dataset. The addition of SENet provided a further boost, making the solution robust and highly accurate for automated ore classification. The combination of techniques (pre-trained MobileNet + DA + SENet) was shown to outperform any single method, and the final accuracy of nearly 97% is a significant improvement over earlier attempts or conventional methods.

III. Dataset Description and Analysis

To better understand the characteristics and challenges of the dataset, we conducted an exploratory data analysis (EDA) before model training. The ore image dataset consists of seven classes representing different mineral types. While the dataset contains a total of 957 images, the distribution across classes is not uniform. For instance, Malachite is the most represented class with 235 samples, while Biotite has only 68 images. This moderate imbalance is visualized in Figure 1. Such discrepancies in class representation may lead to biased training if left unaddressed, as models might tend to favor the majority classes during optimization.

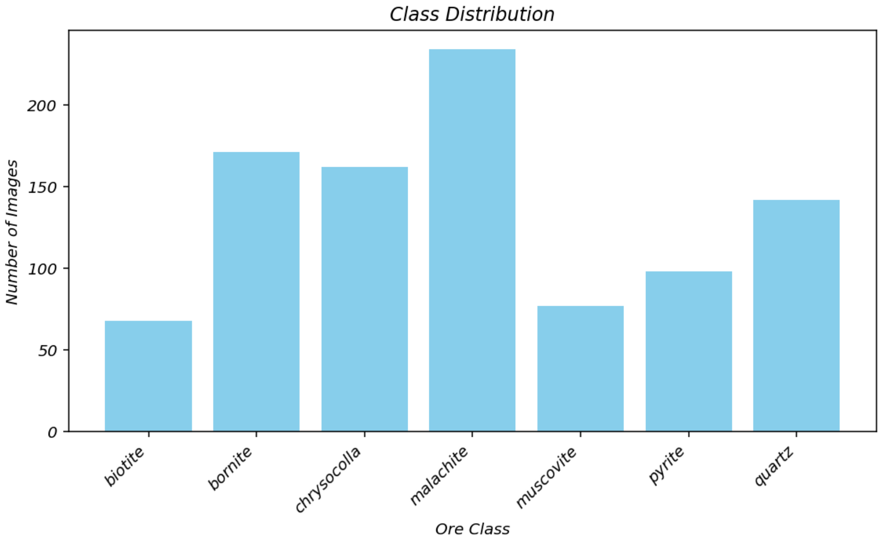


Fig. 1: Number of images per ore class. The dataset shows a moderate imbalance, with Malachite being overrepresented relative to Biotite.

A. *Exploratory Data Analysis*

Visual inspection of the dataset revealed high variability in image conditions. Figure 2 displays one randomly selected image from each class. It is evident that the dataset contains variations in lighting (from overexposed to underexposed conditions), non-uniform backgrounds (ranging from clean white to cluttered surfaces), and scale and orientation of the minerals. Some minerals are centered and neatly cropped, while others are rotated or positioned toward image edges. These factors highlight the need for strong data augmentation strategies, such as horizontal/vertical flipping, random rotations, affine scaling, and color jitter, to improve model generalization.

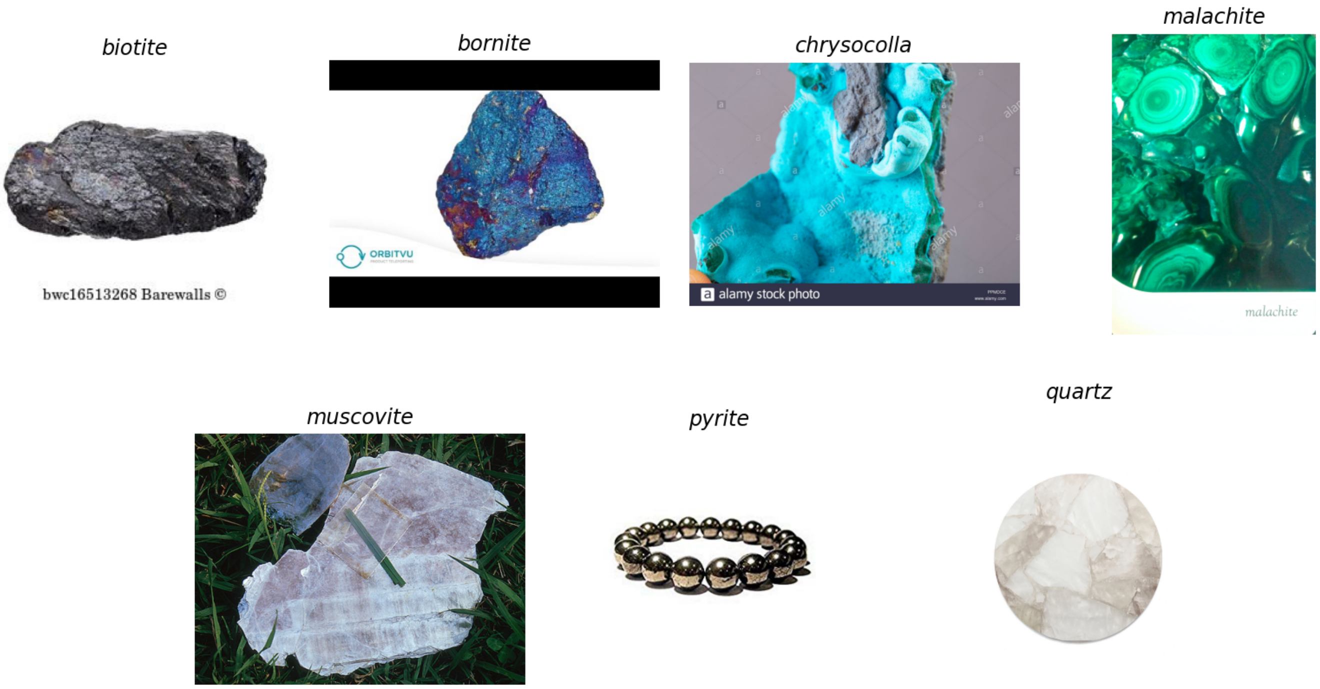


Fig. 2: Example images sampled from each class, highlighting variability in lighting, background, and mineral appearance.

Regarding resolution, the original images varied from approximately 200×200 to 400×400 pixels, with aspect ratios deviating from a perfect square in several instances. Figure 3 shows a scatter plot of image width versus height, which confirms this heterogeneity. To standardize model input, we resized all images to fixed dimensions of 224×224 pixels (or 299×299 for InceptionV3), maintaining consistency during training and inference.

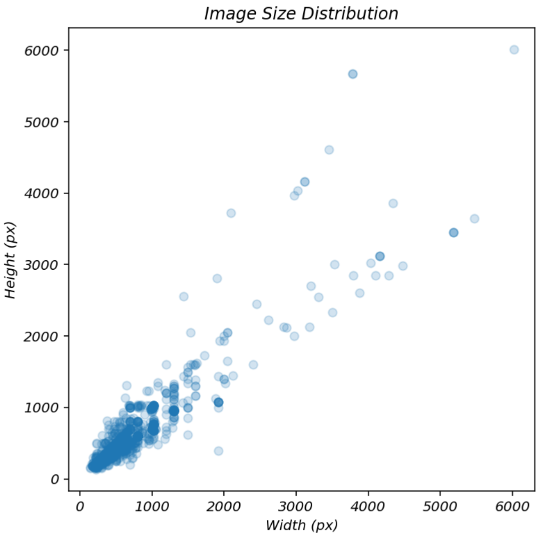


Fig. 3: Scatter plot of image dimensions. Most images cluster between 200–400 pixels, but aspect ratios vary.

In addition to spatial characteristics, we examined the pixel-level statistics of the dataset. The mean and standard deviation across RGB channels were calculated to be approximately $\mu = [0.48, 0.46, 0.41]$ and $\sigma = [0.23, 0.22, 0.22]$, respectively. These values were used for normalization, ensuring that input images had zero-centered and unit-variance features, which aids convergence during training. To explore the color distribution in more detail, we generated aggregated histogram of pixel intensity for malachite (Figure 4). These plot indicate that the class displays strong peak in the green channel, reflecting the natural color of the mineral and suggesting that color is a discriminative feature in this classification task.

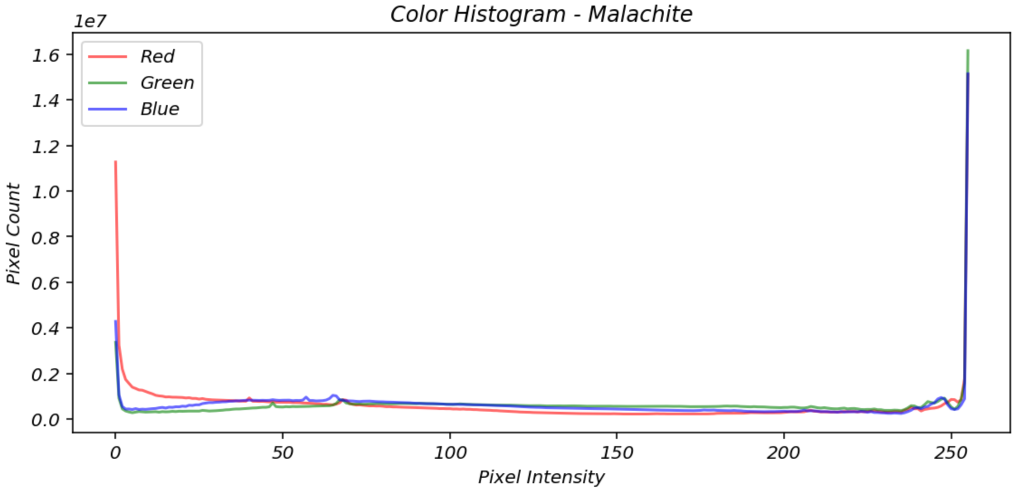


Fig. 4: Aggregated RGB channel histograms across malachite images. Distinct peak in the green channel reflect the color features of the specific mineral.

Lastly, we performed a check for outliers and data integrity. Some corrupted or unreadable images were detected. However, these were promptly excluded to ensure overall dataset uniformity.

Overall, the EDA confirmed several key considerations: the presence of class imbalance, significant image variability, and the importance of color and texture as discriminative features. These insights informed both our augmentation strategy and model selection in the subsequent experiments.

IV. Experimental and Reimplementation Methods

To verify and explore the findings of Zhou et al. [2], we implemented the described approach in Python using PyTorch. The same Kaggle ore image dataset (https://www.kaggle.com/datasets/asiedubrempong/minerals-identification-dataset) of 7 classes and 957 images was used in our experiments. We organized the data into training, validation, and test splits similar to common practice. All experiments were run on a workstation with an NVIDIA GPU, enabling efficient fine-tuning of the models. For reproducibility the random seed was set to 231. Implementation of the entire experiment is available online in our GitHub repository (https://github.com/jakub-spisak/CNN---article-replication.git) [12].

A. Initial Setup and Pipeline

We chose the similar set of CNN architectures for comparison: AlexNet, VGG16, ResNet50, InceptionV3, and MobileNetV2. Notably we decided to switch from MobileNet to MobileNetV2 due the unavailability of the MobileNet model in Pytorch. In our implementation, we utilized PyTorch's pretrained models for all five networks (notably, PyTorch provides ImageNet-pretrained weights for AlexNet as well, so unlike the original TensorFlow-based experiment, we were able to include AlexNet with TL). Each model's final fully-connected layer was replaced to output 7 classes, and we fine-tuned *all* layers of the networks . Training was done using the Adam optimizer with a learning rate of 1×10^{-4} and a dropout probability of 0.5 in the classifier, mirroring the hyperparameters in the original paper [2]. We firstly trained for 20 epochs per model along side cross-entropy loss and in the latter part of the experiment we switched to 50 epochs per model both using mini-batch gradient descent with a batch size of 32 (all theoretical descriptions are in Sec. II). The overall pipeline for ore image classification in our implementation follows a structured sequence of data preparation, model construction, training, and evaluation similar to the original article. The main stages of the process are outlined below:

1) **Dataset Splitting:** Using stratified sampling to preserve class balance, the ore dataset was split into 60 % training, 20 % validation, and 20 % testing subsets. This prevented data leakage and maintained evaluation fairness.
2) **Image Pre-processing:** Each image was resized to match the input resolution required by the CNN architectures: 224×224 pixels for most models and 299×299 for InceptionV3. Pixel values were normalized to the $[0, 1]$ range and then standardized using the dataset-wide mean and standard deviation for each RGB channel: where x_i is the pixel value, μ is the channel-wise mean, and σ is the standard deviation. The class labels were one-hot encoded to match the softmax classifier output (for exactl values see Sec. III).
3) **Model Construction:** Each CNN architecture was instantiated either randomly or with pretrained ImageNet weights. The final classification layer was replaced to accommodate seven output classes. In experiments involving scratch training, all model parameters were randomly initialized.

4) **Training and Evaluation:** During each epoch, the training loss and accuracy were computed, and performance was validated on the separate validation set. The model with the best validation accuracy was retained for final testing.

This pipeline reflects best practices for training deep neural networks on small-to-medium image datasets and was carefully tuned to replicate the methodological framework proposed in the original article.

B. Data Augmentation

One challenge in the reimplementation was the lack of detailed description of the data augmentation in the paper. To emulate their augmentation strategy, we applied a series of image transformations to the training set. Specifically, each training image was transformed using operations such as:

- **Center crop**: This focuses the model on the image center while still enforcing invariance to small translations.
- **Edge crop**: This simulates off-center framing and encourages robustness to objects appearing near image borders.
- **Zoom**: A RandomResizedCrop directly produces output by randomly scaling the image by a factor in $[0.8, 1.2]$ and cropping.
- **Color Jitter**: Random adjustments to brightness and contrast (for example, $\pm 20\%$ change) to mimic different lighting conditions.

These augmentations were applied in various combinations to generate additional training examples. In practice, we expanded the training set by a factor of 5 (each original image yielding 4 augmented versions). We note that the exact augmentation parameters were not provided, so our augmentation may not exactly match theirs.

C. MobileNetV2 Architecture and Comparison with MobileNetV1

In this work, we employed the MobileNetV2 architecture, a refined successor to the original MobileNet model by Howard et al. [9] [13]. Both versions are optimized for efficiency on mobile and edge devices through the use of depthwise separable convolutions; however, MobileNetV2 introduces several architectural innovations that significantly improve performance and memory efficiency while maintaining a compact footprint.

The original MobileNet (V1) architecture is built primarily on depthwise separable convolutional layers, which factorize a standard convolution into a depthwise (channel-wise) convolution followed by a 1×1 pointwise convolution. This reduces the number of parameters and computational cost dramatically, as it avoids the full spatial-channel complexity of standard convolutions. Despite this, V1 applies non-linearity after every convolutional layer, which can harm information flow in deeper architectures.

MobileNetV2 enhances this design with two key concepts: the **inverted residual block** and the **linear bottleneck**. In contrast to traditional residual connections (which operate between high-dimensional layers), MobileNetV2 applies residual shortcuts between the low-dimensional *bottlenecks* rather than the expanded features (see Figure 5). Each inverted residual block begins with a 1×1 pointwise convolution that expands the input channels by a factor t, followed by a 3×3 depthwise convolution with ReLU6 activation, and finally another 1×1 pointwise convolution that linearly projects the features back to the bottleneck dimension.

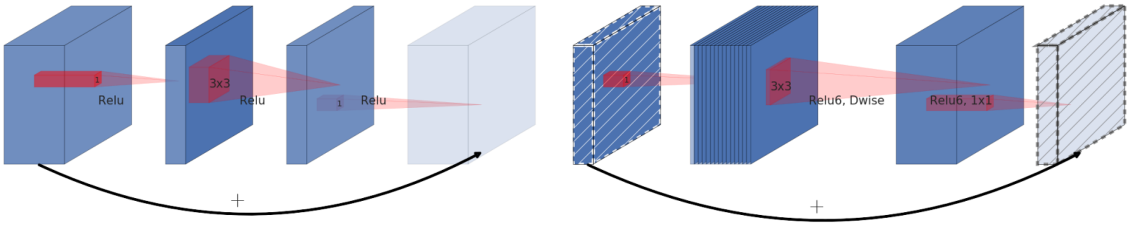


Fig. 5: Comparison of a classical residual block (left) with an inverted residual block (right). In MobileNetV2, skip connections link low-dimensional bottlenecks instead of the expanded layers.

This structure is motivated by the hypothesis that neural activation manifolds lie in low-dimensional subspaces and that applying non-linearities to narrow layers can lead to information loss. The use of a linear projection at the output of each block (without non-linearity) is thus critical in preserving representational power. Figure 5 illustrates the difference between the classical residual block (as used in ResNet) and the inverted residual block characteristic of MobileNetV2.

A detailed breakdown of the inverted residual block structure is provided in Table I, showing the sequence of operations and resulting tensor shapes. The complete MobileNetV2 architecture is shown in Table II, where each row describes one or more stacked bottleneck layers, the expansion factor t, output channels c, number of repetitions n, and stride s.

TABLE I: Bottleneck residual block operations, showing how a tensor of shape $h \times w \times k$ is expanded, transformed via depthwise convolutions, and projected to k' channels.

Input	Operator	Output
$h \times w \times k$	1x1 conv2d, ReLU6	$h \times w \times (tk)$
$h \times w \times tk$	3x3 dwise, $s = s$, ReLU6	$\frac{h}{s} \times \frac{w}{s} \times (tk)$
$\frac{h}{s} \times \frac{w}{s} \times tk$	linear 1x1 conv2d	$\frac{h}{s} \times \frac{w}{s} \times k'$

TABLE II: The full MobileNetV2 architecture. Each row describes a block or layer with input size, expansion factor t, output channels c, number of times repeated n, and stride s.

Input	Operator	t	c	n	s
$224^2 \times 3$	conv2d	-	32	1	2
$112^2 \times 32$	bottleneck	1	16	1	1
$112^2 \times 16$	bottleneck	6	24	2	2
$56^2 \times 24$	bottleneck	6	32	3	2
$28^2 \times 32$	bottleneck	6	64	4	2
$14^2 \times 64$	bottleneck	6	96	3	1
$14^2 \times 96$	bottleneck	6	160	3	2
$7^2 \times 160$	bottleneck	6	320	1	1
$7^2 \times 320$	conv2d 1x1	-	1280	1	1
$7^2 \times 1280$	avgpool 7x7	-	-	-	-
$1 \times 1 \times 1280$	conv2d 1x1	-	k	-	-

D. SENet Architecture

To further improve classification performance and encourage the model to focus on informative feature channels, we incorporated a Squeeze-and-Excitation (SE) module [3] after the final convolutional layer of the MobileNetV2 backbone. The SE block is an attention mechanism designed to recalibrate channel-wise feature responses by explicitly modeling interdependencies between the channels.

As illustrated in Figure 6, the SE module operates in three stages: *Squeeze*, *Excitation*, and *Recalibration*.

- **Squeeze:** Given the output feature map $\mathbf{U} \in R^{C \times H \times W}$ from the convolutional backbone, a global average pooling operation is applied across the spatial dimensions to generate a channel-wise descriptor:

$$z_c = \frac{1}{H \times W} \sum_{i=1}^{H} \sum_{j=1}^{W} U_c(i, j), \quad \forall c \in \{1, \ldots, C\} \tag{6}$$

  This results in a vector $\mathbf{z} \in R^C$ representing the global contextual information per channel.

- **Excitation:** The descriptor $\mathbf{z}$ is passed through a bottleneck-style two-layer fully connected network to learn non-mutually-exclusive channel-wise dependencies:

$$\mathbf{s} = \sigma\left(W_2 \cdot \delta(W_1 \cdot \mathbf{z})\right) \tag{7}$$

  where $W_1 \in R^{\frac{C}{r} \times C}$, $W_2 \in R^{C \times \frac{C}{r}}$, $\delta(\cdot)$ is the ReLU activation function, and $\sigma(\cdot)$ denotes the sigmoid function. The reduction ratio r is a hyperparameter (commonly $r = 8$) that reduces the intermediate dimensionality to limit model complexity.

- **Recalibration:** The learned activations $\mathbf{s} \in R^C$ are used to rescale the original feature maps via channel-wise multiplication:

$$\tilde{U}_c = s_c \cdot U_c, \quad \forall c \in \{1, \ldots, C\} \tag{8}$$

  where s_c is the excitation weight for channel c. This step emphasizes important features while suppressing less useful ones.

The recalibrated feature map $\tilde{\mathbf{U}}$ is then forwarded to the global pooling and fully connected classification head. The full MobileNetV2+SENet model architecture is visualized in Figure 6, where the SE module is seamlessly integrated at the end of the MobileNet backbone.

This attention mechanism introduces negligible computational overhead but yields significant gains in accuracy, particularly on small and imbalanced datasets such as ours. The use of SE blocks allows the network to adaptively emphasize discriminative mineral features, such as color intensity, texture patterns, and luster, which vary across ore types.

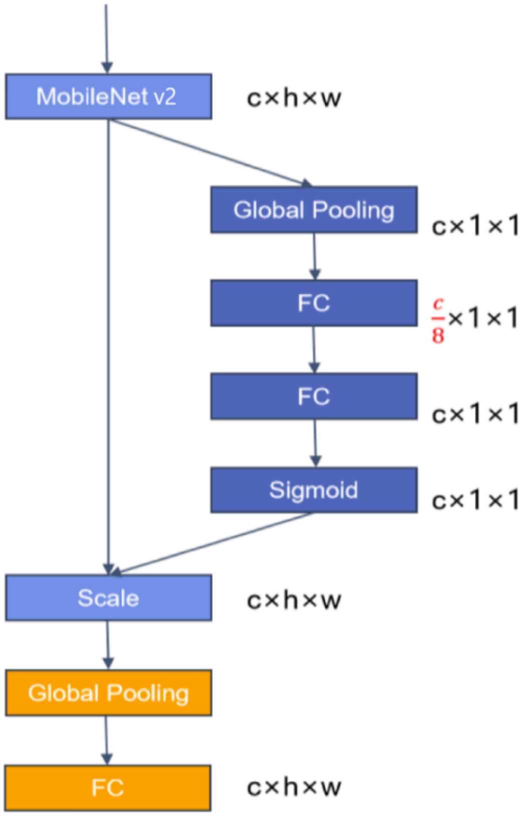


Fig. 6: SENet module structure added after MobileNet. The excitation block consists of two fully connected layers and a sigmoid activation that scales channel-wise features.

V. Experiments

To evaluate the performance of various CNN architectures on the ore classification task, we conducted a series of controlled experiments. The goal was to compare the effectiveness of different training strategies—ranging from training from scratch to transfer learning and data augmentation—and to assess the influence of architectural modifications, particularly the use of the SENet attention mechanism.

A. Baseline Comparison: Training from Scratch

Our first experiment involved training five popular CNN architectures—AlexNet, VGG16, ResNet50, InceptionV3, and MobileNetV2—on the ore dataset with all weights randomly initialized (`pretrained=False`). This setting, referred to as *training from scratch*, represents a lower bound in performance due to the limited amount of data and the absence of prior knowledge from large-scale datasets like ImageNet.

B. Transfer Learning

Next, we enabled transfer learning by initializing all models with ImageNet-pretrained weights (`pretrained=True`). In this setting, the network's backbone (convolutional layers) inherited generic feature extractors, which were then fine-tuned on the ore dataset. All layers were thus made trainable.

C. Transfer Learning with Data Augmentation

To further enhance generalization, we introduced data augmentation during training. The augmented dataset was created using a the techniques from IV-B. Afterwards, we evaluate all of the CNNs and take the one with highest test accuracy as the subject for further experimentation.

D. Enhanced MobileNet with SENet

The mode with strongest performance, was selected as the backbone for further enhancement using a Squeeze-and-Excitation (SE) module. This addition recalibrates channel-wise feature responses via global pooling, bottleneck fully connected layers, and sigmoid gating. The resulting attention weights are used to rescale the original features before classification (see Section IV-D).

E. *Ambiguities and differences*

While our implementation followed the general design described in the original article [2], several ambiguities in the training procedure required careful consideration. The authors mention fine-tuning the network, but do not specify whether any layers were frozen during training—particularly in the SE-enhanced models. In standard practice, SENet is typically trained with the backbone frozen initially (or entirely), especially when integrated post hoc into pretrained architectures.

To investigate this, we conducted two versions of the second part of the experiment:

- **Full fine-tuning:** All layers, including the MobileNetV2 backbone and the SE block, were trainable.
- **Partial fine-tuning:** Only the SE module and final classification head were trained. The backbone was frozen using:

```
for p in backbone.features.parameters():
    p.requires_grad = False
```

Models were evaluated on the held-out test set using test accuracy. Performance was logged and visualized through training curves and confusion matrices. We also plotted the training and validation losses as further performance visualisation.

VI. Results

In this section, we present the experimental outcomes from our comprehensive evaluation of five prominent CNN architectures (AlexNet, VGG16, ResNet50, InceptionV3, and MobileNetV2) under three distinct training scenarios: training from scratch, transfer learning (TL), and transfer learning combined with data augmentation (TL+DA). Additionally, we analyze the performance of the enhanced MobileNetV2 model integrated with the SENet attention module, highlighting specific insights from backbone freezing experiments.

A. *Overall Performance Across Architectures*

Table III summarizes the test accuracies for each architecture and training scenario, comparing the original paper's reported values [2] to our own reproduced results. While there are slight discrepancies between our reproduced results and the original article, the overall performance hierarchy remains consistent. Our results corroborate the conclusion from [2] that deeper models and lightweight, efficient architectures like MobileNetV2 substantially benefit from transfer learning and data augmentation, achieving significantly improved performance compared to training from scratch.

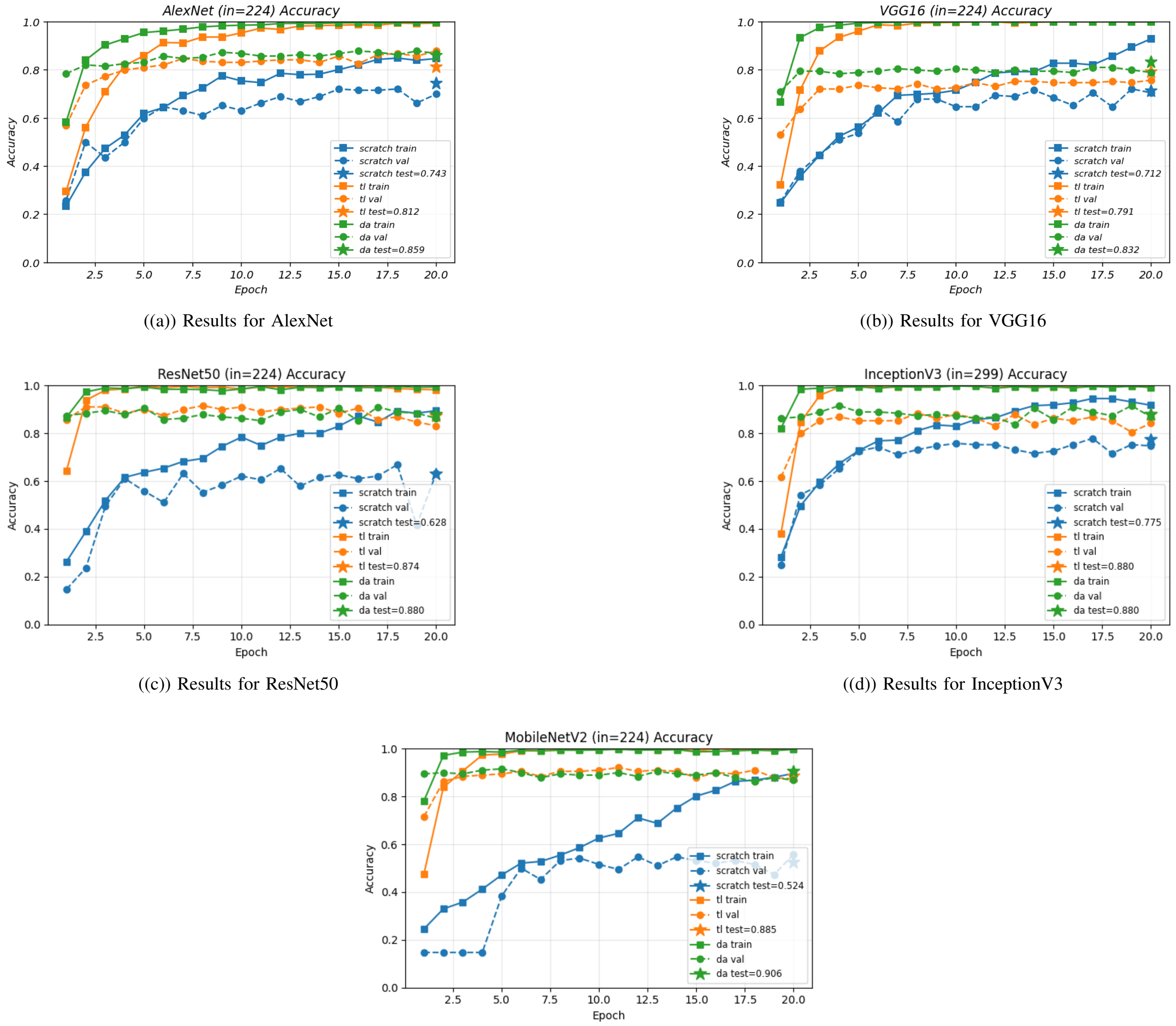
TABLE III: Test accuracy comparison across training regimes: original results from [2] and our reproduction.

Model	**Experiment**	**Test Accuracy**	
		Original	Our results
AlexNet	Scratch	0.7047	0.743
	TL	X	0.812
	TL+DA	X	0.859
VGG16	Scratch	0.7461	0.712
	TL	0.8601	0.791
	TL+DA	0.8238	0.832
ResNet50	Scratch	0.6580	0.696
	TL	0.8497	0.874
	TL+DA	0.8756	0.880
InceptionV3	Scratch	.07513	0.775
	TL	0.7824	0.880
	TL+DA	0.8497	0.880
MobileNet	Scratch	0.4404	0.524
	TL	0.8238	0.885
	TL+DA	**0.9482**	**0.906**

Specifically, MobileNetV2, which initially performed poorly when trained from scratch (0.524 accuracy), exhibited remarkable improvements under transfer learning (0.885 accuracy) and even more pronounced enhancements when coupled with data augmentation (0.906 accuracy). Such findings reinforce the critical importance of leveraging pre-trained features and robust data augmentation methods, particularly when dealing with limited datasets.

B. *Training and Validation Dynamics*

Figure 7 and Figure 8 depict the training and validation accuracy and loss curves for each CNN across different experimental setups. These plots provide deeper insights into the training dynamics. For instance, models trained from scratch exhibited slow convergence and higher volatility in validation metrics, indicative of overfitting and ineffective learning. In contrast, introducing



((a)) Results for AlexNet

((b)) Results for VGG16

((c)) Results for ResNet50

((d)) Results for InceptionV3

((e)) Results for MobileNet

Fig. 7: (Combined training, validation and test accuracy for: (a)AlexNet. (b)VGG16. (c)ResNet50. (d)InceptionV3. (e)MobileNet.

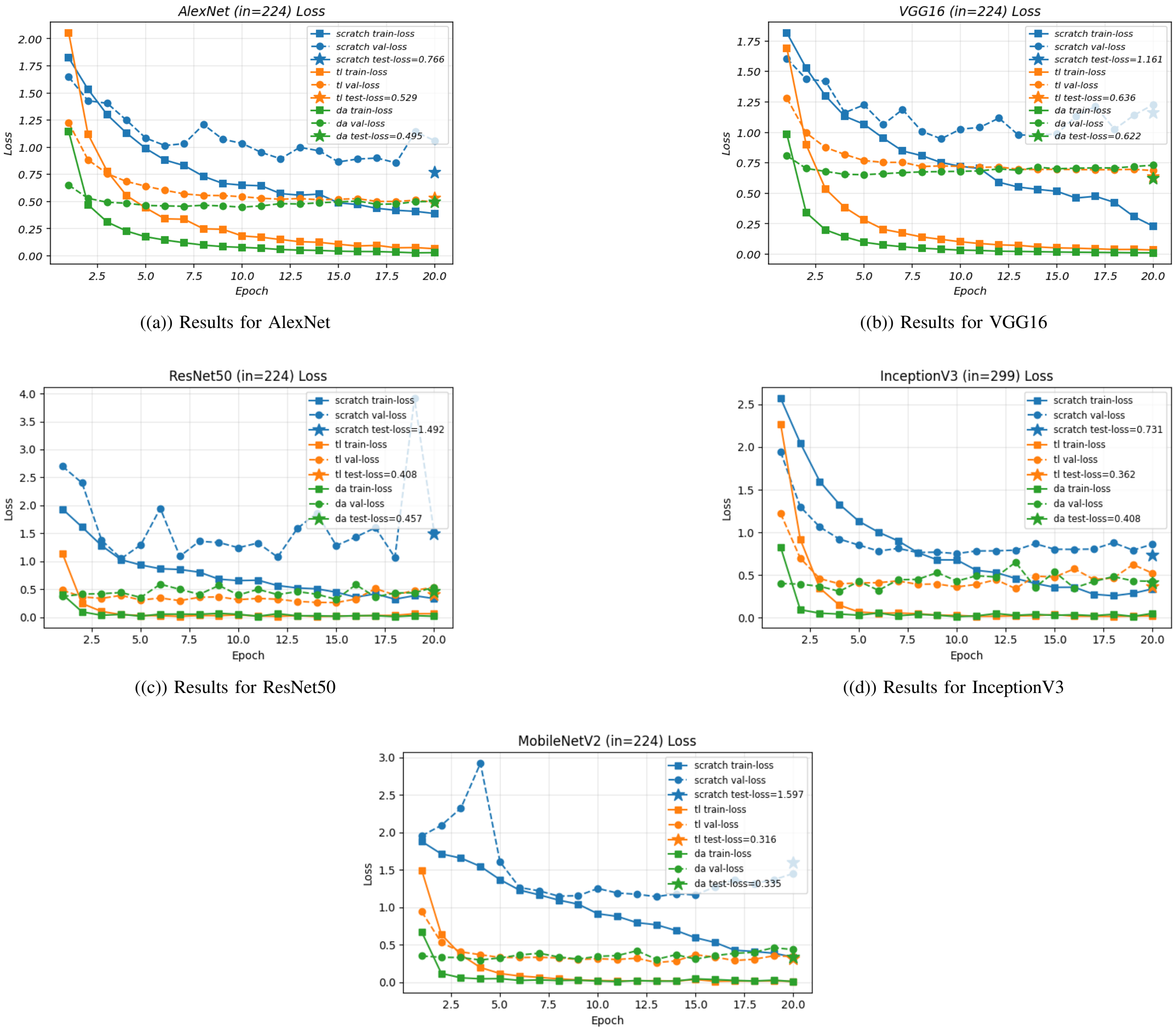
transfer learning markedly improved convergence speed, stability, and final accuracy for most models. Here we also concluded that the reduction from original 50 epochs to 20 epochs did not decrease the overall performance.

The addition of data augmentation further stabilized the training process, evident from the smoother validation accuracy and consistently lower validation losses across epochs. Among all architectures, MobileNetV2 demonstrated the most significant performance leap, quickly achieving higher accuracy and lower loss compared to other architectures under equivalent conditions.

C. *Enhanced MobileNetV2 with SENet Attention*

Motivated by MobileNetV2's superior performance, we extended it by integrating a SENet attention module, creating the `MobileNet + TL + DA + SENet` architecture. Results for this model configuration are detailed in Table IV, clearly indicating the performance benefit of incorporating channel-wise attention mechanisms.

However, during experimentation, we encountered ambiguity regarding optimal training strategies for the SENet enhancement. Specifically, the original paper [2] lacked explicit detail about backbone fine-tuning or freezing protocols during training. Hence, we conducted parallel experiments comparing frozen and unfrozen backbone training. Notably, we observed a significant advantage when freezing the MobileNetV2 backbone for the first 10 epochs. Freezing facilitated stable early-stage training for



((a)) Results for AlexNet

((b)) Results for VGG16

((c)) Results for ResNet50

((d)) Results for InceptionV3

((e)) Results for MobileNet

Fig. 8: Error analysis for all experimental setups for: (a)AlexNet. (b)VGG16. (c)ResNet50. (d)InceptionV3. (e)MobileNet.

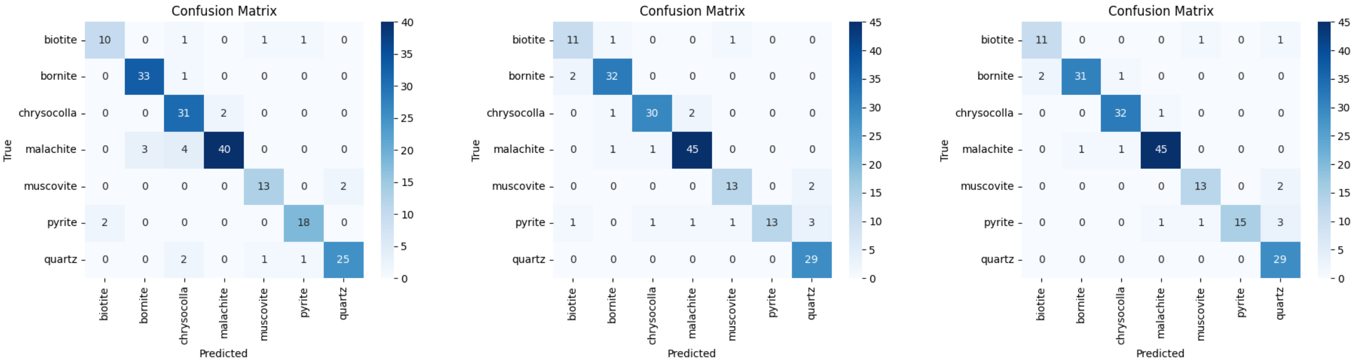
TABLE IV: Test accuracy for the preferred model (MobileNet) under different training settings.

Experiment	Original	Our Results
MobileNet + TL	0.8238	0.9058
MobileNet + TL + DA	0.9482	0.9162
MobileNet + TL + DA + SENet	0.9689	0.9319 (frozen) & 0.9005 (unfrozen)

the SENet module, ultimately achieving a higher test accuracy (0.9319) compared to end-to-end unfrozen training (0.9005). This clearly highlights the necessity of careful backbone freezing when incorporating SENet into pretrained architectures, a critical detail that was potentially omitted in the original study.

D. Confusion Matrix Analysis

To qualitatively evaluate the model's predictive capability, we analyzed confusion matrices from test set predictions. Figure 9 shows the confusion matrices for MobileNetV2 models trained under different scenarios (TL, TL+DA, and TL+DA+SENet). All matrices generally demonstrate strong diagonal dominance, indicating accurate predictions across most classes. However, subtle misclassifications persist, particularly for visually similar minerals such as malachite, pyrite and quartz. The addition of data augmentation noticeably reduces these confusions, and the integration of SENet further enhances precision in these challenging distinctions, highlighting the SENet module's effectiveness in recalibrating channel-wise features crucial for distinguishing similar mineral classes.



((a)) MobileNetV2 + TL

((b)) MobileNetV2 + TL + DA

((c)) MobileNetV2 + TL + DA + SENet

Fig. 9: Confusion matrices comparing three variants of MobileNetV2 training: (a) with TL only, (b) with TL and DA (DA), and (c) with TL, DA, and SE module.

In summary, our extensive experiments confirm the effectiveness of transfer learning, data augmentation, and SENet attention mechanisms for the ore image classification task. MobileNetV2, enhanced with SENet and trained using a strategic backbone freezing regime, clearly emerged as the top-performing model, striking an optimal balance between computational efficiency and predictive accuracy. Despite minor numerical differences, our findings strongly validate and extend the original study's results, providing additional practical insights into training attention-enhanced CNNs effectively.

VII. Discussion

The combination of transfer learning, data augmentation, and SENet attention proves to be a highly effective approach for classifying ore images with a limited dataset. Our analysis of the original work and the reproduced experiments yields several insights:

Firstly, **transfer learning is indispensable for this task**. Training CNNs from scratch on only a few hundred images simply does not provide enough data for the models to learn generalizable features, as evidenced by the very low accuracies in the scratch training experiment [2]. By leveraging pre-trained models, we start from a rich feature representation that only needs slight adaptation to the ore domain. This resulted in an enormous jump in performance (e.g., around 38% absolute accuracy increase for MobileNet). The experiments underscore that even for tasks in specialized domains like mineral images, the generic features learned from large datasets (edges, textures, shapes, etc.) are transferable and beneficial. This aligns with broad observations in deep learning literature that transfer learning can effectively compensate for data scarcity.

Secondly, **data augmentation plays a critical role in squeezing out extra performance and combating overfitting**. In our reproduction, we noticed that without aggressive augmentation, models like VGG16 would overfit almost immediately (memorizing the training images). Augmentation introduced enough variance to the training process to improve validation accuracy and make the most of the limited data. Zhou *et al.* credited augmentation as a key factor in reaching over 90% accuracy with MobileNet [2]. The precise choice of augmentations can matter: for instance, because ore images might have arbitrary orientation, including vertical flips (which are not typically used for natural images like animals or objects) was reasonable here. A takeaway is that domain knowledge can guide augmentation (e.g., knowing that color might be an important feature for certain minerals means one should be careful with color jitter so as not to distort the inherent color too much).

Thirdly, the introduction of the **SENet attention mechanism** provided a noticeable performance boost. SENet helps the model focus on the most relevant features. In the context of ore images, certain color channels or texture patterns might be particularly indicative of a specific ore (for example, the green hue of malachite versus the metallic luster of galena). By using SE blocks, the model can learn to amplify the neurons detecting those features when appropriate. The result was an improved accuracy with MobileNet, which is significant for a 7-class classification problem. This demonstrates the effectiveness of combining an existing CNN architecture with a channel-attention module for fine-grained image recognition tasks. The idea is that the SE block adds very few parameters and computational overhead but yields a non-linear feature recalibration that a plain CNN might not learn on its own, especially with limited data.

A. Differences and Similarities with Original Results

Despite our results following the same general trajectory as the original paper, we observed minor quantitative differences. Several factors may account for these discrepancies:

1) The original authors used the MobileNetV1 architecture, whereas our implementation utilized the improved MobileNetV2 variant. MobileNetV2 incorporates inverted residual blocks and linear bottlenecks, potentially leading to different patterns of convergence and slightly altered performance characteristics.
2) Differences in exact implementation details, including randomness in data splitting, augmentation parameters, or optimizer configurations, may contribute to variations in final performance.
3) The original paper did not explicitly disclose details regarding backbone freezing or fine-tuning strategies, especially in relation to the SENet module. As demonstrated by our experiments, these training nuances significantly influence the final accuracy.
4) Computational environment differences, including hardware specifications and numerical precision, could have further contributed to minor performance variations.

Nonetheless, the broad alignment between our reproduced results and those reported in the original study strongly validates the effectiveness of transfer learning, data augmentation, and SENet attention mechanisms in this classification context. Crucially, our findings offer practical insights—particularly the observed advantage of MobileNetV2 over MobileNetV1, and the effectiveness of backbone freezing during SENet training—thus extending and clarifying the conclusions presented by Zhou et al. [2].

Despite the success, there are some **limitations and considerations**. The approach heavily relies on the availability of pre-trained models and the assumption that those learned features are relevant to the ore images. If the ore images were very different from typical photographs (for example, if they were microscopic images or had very unusual spectral characteristics), the transfer learning approach might be less effective. Additionally, the authors only experimented with a specific SE configuration added to MobileNet. It is possible that other architectures could benefit from attention mechanisms as well. In our implementation, we focused on MobileNet for SENet integration (following [2]), but one could imagine adding SE blocks to ResNet50 or InceptionV3 and perhaps improving their performance. The reason the authors chose MobileNet is likely because it was the best baseline and a natural choice to optimize further. Another consideration is the size of the SE enhancement: the paper used a single SE block after MobileNet's global features. The original SENet paper [3] typically inserts SE blocks in every module of a ResNet, for example. A more extensive integration of SE (or other attention modules) throughout the network might yield further gains, though it would need more data to train effectively.

The relatively lower performance of the larger CNNs (VGG, ResNet, Inception) points to an interesting observation: **bigger is not always better, especially for small datasets**. MobileNetV2, being a lightweight model, had an easier time fine-tuning on the small dataset without overfitting too severely. This suggests that for problems with limited data, choosing a simpler model can sometimes outperform a complex model that has higher capacity than the data can effectively utilize. This is an important practical point for practitioners working on similar tasks (e.g., classification of images in niche domains with small datasets): starting with a smaller pre-trained network or using techniques to reduce model capacity (like regularization or knowledge distillation) might yield better results than blindly using the deepest network available.

B. Potential Improvements and Future Work

While the results are already impressive, there are several avenues for further improvement or investigation:

- **Alternate Attention Mechanisms:** The success of SENet opens the door to trying other attention modules. For example, the Convolutional Block Attention Module (CBAM) proposed by Woo *et al.* [4] extends the idea of SENet by incorporating *spatial* attention in addition to channel attention. CBAM sequentially applies channel attention (like SENet) and spatial attention (focusing on important regions in the feature map). Using CBAM or similar mechanisms (e.g., ECANet, SKNet, or attention from transformer-based models) in place of or in addition to SENet could potentially capture complementary information and further improve classification performance. This would be especially useful if certain ore types are distinguished by specific textures or shapes in particular regions of the image (spatial attention could highlight those regions).
- **Ensemble of Models:** Ensembling is a well-known strategy to boost accuracy by combining multiple models' predictions. An ensemble of the top-performing models (for example, MobileNetV2+SENet together with another CNN like ResNet50) could yield higher accuracy than any single model. Each model may capture different aspects of the data or have different error patterns, so an ensemble can average out individual mistakes. In the context of ore classification, one could train several models (possibly with different random initializations or augmentation settings) and average their outputs to improve reliability. This might be particularly helpful in borderline cases where one model might misclassify an image but another gets it right. The downside is the increased computational cost, but for an offline classification system, an ensemble could be feasible.
- **Optimizing Data Augmentation:** As noted, augmentation is crucial for this task. Further optimizing the augmentation pipeline could yield better results, especially for the models that underperformed. This could involve more systematically searching for augmentation types and intensities that maximize validation performance. Techniques like AutoAugment (which learns the optimal augmentation policy) could be explored. Additionally, ensuring that augmentation does not inadvertently distort features that are key to distinguishing classes is important. For instance, if color is a primary

discriminator between certain ore types, one should use color jitter carefully. Another idea is to use synthetic data generation: possibly using generative adversarial networks (GANs) to create additional realistic ore images to supplement the training data.

- **Architecture Updates:** Since the original study in 2022, there have been new developments in CNN architectures. One could experiment with more recent architectures (for example, EfficientNet or DenseNet, or transformer-based vision models like Vision Transformers or ConvNeXt) on this ore dataset. EfficientNet, in particular, is known for optimizing the accuracy vs. complexity trade-off and might perform well with appropriate scaling. Some of these architectures also have variants with built-in attention or have shown strong performance with fewer training images. It would be interesting to see if any of these modern models could surpass the MobileNet+SENet combination.
- **Cross-Domain Transfer and Fine-tuning Strategies:** Another extension is to explore different transfer learning strategies. The paper fine-tuned all layers with a single learning rate. Perhaps freezing some early layers (which capture very generic features) and only fine-tuning the higher layers could yield similar performance with less risk of overfitting. Alternatively, using a differential learning rate (smaller for the base layers, higher for the new layers) can sometimes stabilize training. One might also attempt to use a model pre-trained on a domain closer to mineral images (if available, e.g., a network pre-trained on geology or remote sensing images) to see if that offers an advantage over ImageNet pre-training.

In terms of the application, achieving 94-96 % accuracy is already quite good, but for real-world deployment in mining operations, consistency and reliability are key. It may be worthwhile to incorporate a confidence measure or anomaly detection for cases where the model is uncertain or when an image does not resemble anything in the training set (for example, a completely new type of ore or rock). Ensuring the model is robust to varying imaging conditions (lighting, camera type, background) is another practical consideration—further data collection covering more variability could help with this.

VIII. Conclusion

In this replication study, we critically evaluated the CNN-based approach proposed by Zhou et al. (2022) [2] for classifying ore images, which combined transfer learning, data augmentation, and the SENet attention mechanism. Our extensive experiments demonstrated that these strategies effectively mitigate the challenges associated with limited labeled datasets and overfitting. Among the CNN architectures tested, MobileNetV2 emerged as the best performer, highlighting its advantageous balance between computational efficiency and predictive capability. The integration of the SENet module further boosted performance by adaptively recalibrating channel-wise feature importance, although careful management of backbone freezing proved essential to optimize its benefits. While minor numerical discrepancies between our results and the original paper were observed—likely due to differences in augmentation specifics, MobileNet architecture (V2 versus V1), and training protocol ambiguities—our findings substantively validated and extended the original conclusions. Future research directions include exploring additional attention mechanisms, ensemble modeling, more detailed parameter tuning, and the potential application of metaheuristic optimization methods to improve initial parameterization. Overall, our results affirm that the original hybrid approach remains highly robust and effective for automated ore image classification tasks.

References

[1] Pan, S. J., & Yang, Q. (2010). Q.: A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(10), 1345–1359.

[2] W. Zhou, H. Wang, and Z. Wan, "Ore image classification based on improved CNN," *Computers & Electrical Engineering*, vol. 99, Art. no. 107819, 2022.

[3] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 7132–7141.

[4] S. Woo, J. Park, J.-Y. Lee, and I. S. Kweon, "CBAM: Convolutional block attention module," in *Proc. European Conf. Computer Vision (ECCV)*, 2018, pp. 3–19.

[5] Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2017). ImageNet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6), 84–90.

[6] Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.

[7] He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 770–778).

[8] Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., & Wojna, Z. (2016). Rethinking the Inception Architecture for Computer Vision. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 2818–2826).

[9] A. G. Howard *et al.*, "MobileNets: Efficient convolutional neural networks for mobile vision applications," *arXiv preprint* arXiv:1704.04861, 2017.

[10] Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., & Fei-Fei, L. (2009). ImageNet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition* (pp. 248–255).

[11] Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

[12] J. Spišák, D. Zemančík, "*CNN—article-replication*," GitHub repository, https://github.com/jakub-spisak/CNN---article-replication.git.

[13] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "MobileNetV2: Inverted Residuals and Linear Bottlenecks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4510–4520, 2018.