Group #8
Image-Based Mushroom Edibility Classification with Deep Learning

Chiu-Chiu (JoJo) Lin*, Nicholas Shor†, Joanna Tam‡ and Xuewen (Daphne) Yang§
Halıcıoğlu Data Science Institute, University of California - San Diego
La Jolla, CA, 92093 USA
Email: *jol145@ucsd.edu, †nshor@ucsd.edu, ‡j7tam@ucsd.edu, §xuy001@ucsd.edu

***Abstract*—Automated mushroom identification is an important problem for food safety because many toxic and edible species share very similar visual characteristics, making misidentification dangerous for non-experts. In this study, we develop an image-based mushroom edibility classification system using deep learning and the Kaggle Mushroom Classification dataset containing more than 100,000 real-world images. During exploratory data analysis, several dataset challenges were identified, including duplicate images, severe class imbalance, and inconsistent image resolutions. To improve data quality, we constructed a preprocessing pipeline that performs metadata extraction, exact and near-duplicate detection using perceptual hashing, dataset standardization, and data augmentation. We evaluated multiple deep learning architectures, including convolutional neural networks (ResNet-18 and ResNet-50) and a transformer-based model (Swin Tiny Transformer). Experimental results show that the Swin Tiny model achieved the best overall performance, reaching approximately 92.26% accuracy and 94.02% recall for the toxic class, demonstrating strong ability to learn visual features relevant to mushroom morphology while prioritizing safety-critical detection of toxic mushrooms. These results highlight the potential of modern computer vision methods as assistive tools for mushroom identification and safety screening.**

***Index Terms*—Mushroom classification, deep learning, convolutional neural networks, computer vision, image classification, data augmentation.**

I. Introduction

Mushroom identification is a challenging and potentially dangerous task due to the large number of species and the visual similarity between edible and toxic varieties. Many poisonous mushrooms closely resemble edible species, making reliable identification difficult even for experienced foragers. Misidentification can lead to serious health consequences, including severe poisoning or death. As a result, developing reliable automated tools for mushroom recognition has become an important research problem in computer vision and food safety.

Traditional mushroom identification methods rely heavily on expert knowledge, morphological taxonomy, and manual comparison. While these approaches can be effective for trained specialists, they are difficult for non-experts to use and do not scale well to large image collections. Furthermore, real-world mushroom images often vary significantly in lighting conditions, camera angles, backgrounds, and image resolutions, which makes manual identification and rule-based systems less reliable.

Recent advances in deep learning have significantly improved performance in image recognition tasks. Convolutional neural networks (CNNs) are capable of learning hierarchical visual representations directly from image data, allowing them to capture complex patterns such as texture, shape, and structural relationships. More recently, transformer-based vision models have also demonstrated strong performance by modeling global spatial relationships within images using attention mechanisms.

In this work, we investigate the use of deep learning models for image-based mushroom edibility classification. Our study evaluates both convolutional neural network architectures and transformer-based models to determine how effectively they can learn visual features associated with mushroom morphology. The goal is to classify mushrooms into edibility categories using image data and to analyze the impact of different model architectures on classification performance.

Our work aims to contribute to the development of automated mushroom identification systems by exploring how modern deep learning approaches can capture complex visual patterns in mushroom images and improve classification reliability.

II. Related Work

Mushroom identification has traditionally been performed by hand, relying on expert mycologists who examine physical characteristics such as spore prints, gill structure, cap shape, and odor. While accurate, this approach is inaccessible to non-experts and does not scale. Automated image-based classification has emerged as a promising alternative, with deep learning methods demonstrating strong results across a range of mushroom datasets [1], [2].

CNN-based approaches. Convolutional Neural Networks have been the dominant approach for mushroom image classification. The ResNet family [3], which introduced skip connections to stabilize gradient flow in deep networks, has been widely adopted as the backbone of choice. Skip connections were a particularly clever innovation. They solved the vanishing gradient problem that previously made very deep networks impractical to train, enabling far richer feature hierarchies.

Prior work applying CNNs to mushroom classification has shown strong performance on species-level identification tasks [4]–[6]. Wang et al. [5] demonstrated effective classification of a single species using image processing, while Zhao et al. [6] applied machine vision to detect traits in shiitake caps. However, these studies focus on single species or controlled environments, limiting their generalization to wild mushroom classification across many species. Additionally, Yin et al. [1] reviewed computer vision and machine learning applications across the mushroom industry, finding that CNNs consistently outperform traditional feature engineering methods, but noting that most systems are evaluated on clean, lab-collected images that do not reflect real-world conditions.

Transformer-based approaches. More recently, Vision Transformers (ViT) [7] have emerged as a competitive alternative to CNNs. Rather than learning local features through convolution, ViT applies self-attention across image patches, allowing it to model global spatial relationships. This is particularly relevant for mushroom identification, where holistic features such as cap-to-stem proportions are as diagnostic as local texture. The Swin Transformer [8] improved on ViT by introducing hierarchical shifted window attention, making it computationally efficient while retaining the ability to capture both local and global features. The DINO framework [9] further demonstrated that self-supervised Vision Transformers produce rich visual representations without any labels, suggesting strong potential for domains where labeled mushroom data is scarce. The current state of the art in general image classification favors large-scale transformer models, though CNN-based approaches remain competitive for domain-specific tasks with limited data.

Gaps addressed by this work. Despite this progress, the existing literature has three notable limitations. First, most work frames mushroom classification as a binary problem, edible versus poisonous. This overlooks the important category of conditionally edible mushrooms, which are safe only under specific preparation conditions. Our work addresses a three-class problem that more closely reflects real-world foraging risk. Second, there are few direct head-to-head comparisons of CNN and Transformer architectures on the same mushroom dataset under identical conditions. Most papers evaluate a single architecture in isolation, making it difficult to draw conclusions about which family of models is better suited to this task. Third, and most critically, existing studies rarely track per-class recall for the toxic class specifically. A classifier with high overall accuracy may still fail dangerously if toxic mushrooms are frequently misclassified as edible. We explicitly monitor toxic recall throughout our experiments, treating it as the primary safety metric [10].

III. Dataset and Features

A. Dataset Description

We used the Mushroom Classification Dataset from Kaggle.com [11] in this project. The dataset contains real-world mushroom images organized by species and edibility. The images are classified into four classes: edible, poisonous, deadly, and conditionally edible. Initial exploratory data analysis revealed substantial duplicate content, severe class imbalance, and variability in image properties.

The raw dataset contains 102,711 images. Image resolutions and aspect ratios varied widely across images. Most images had low to medium pixel counts with a few high-resolution outliers, while aspect ratios were concentrated near 1. The exploratory data analysis also indicates that 15% of the images were duplicates and that the dataset demonstrated severe class imbalance as shown in Fig. 1, with the deadly class accounting for less than 1% of the total samples.

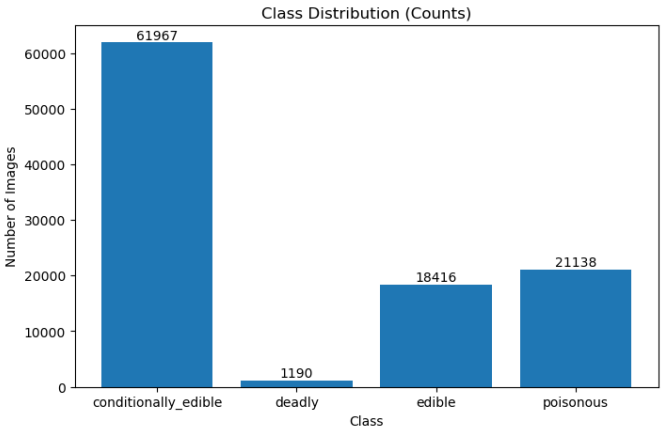

Fig. 1. Class distribution of the raw dataset.

To enhance dataset quality, we employed perceptual hashing (pHash) [12] technology to identify and remove duplicate and near-duplicate images. Images with identical hash values were considered exact duplicates, while near-duplicates were identified by calculating the Hamming distance between pHash values—images with a distance below the 5 threshold were deemed visually similar. Only the highest-resolution image was retained within each duplicate group.

To address class imbalance, we first merged the deadly and poisonous categories into a single toxic class. Subsequently, random undersampling was applied to balance the distribution across the three categories. The final balanced dataset for model training and evaluation comprises three categories: conditionally edible, edible, and toxic, each containing 16,058 images (see Fig. 2).

After balancing, stratified sampling was used to split the dataset into training, validation, and test sets with a ratio of 0.8:0.1:0.1 using scikit-learn's `train_test_split` function [13], resulting in 38,539 training samples, 4,818 validation samples, and 4,817 test samples. This approach ensured that the class distribution remained consistent across all subsets.

Prior to model training, we applied different data standardization and augmentation strategies to the training and evaluation sets to enhance model robustness. All three sets underwent resizing and normalization. However, the training set incorporated additional data augmentation techniques such as color dithering and random flipping to boost generalization capabilities and mitigate overfitting.

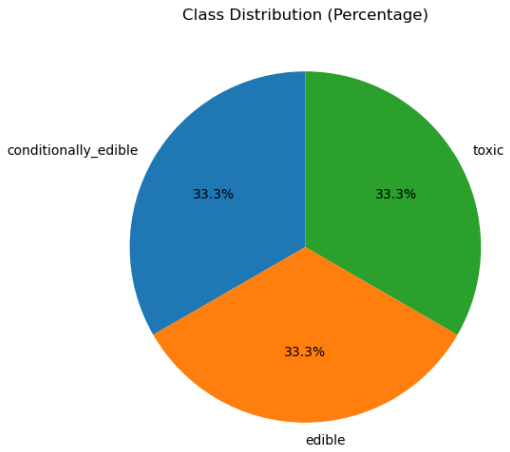

Fig. 2. Class distribution of the balanced dataset.

Representative examples from each edibility class of the balanced dataset are shown in Fig. 3.

Data preprocessing and analysis were performed using NumPy [14] and pandas [15].

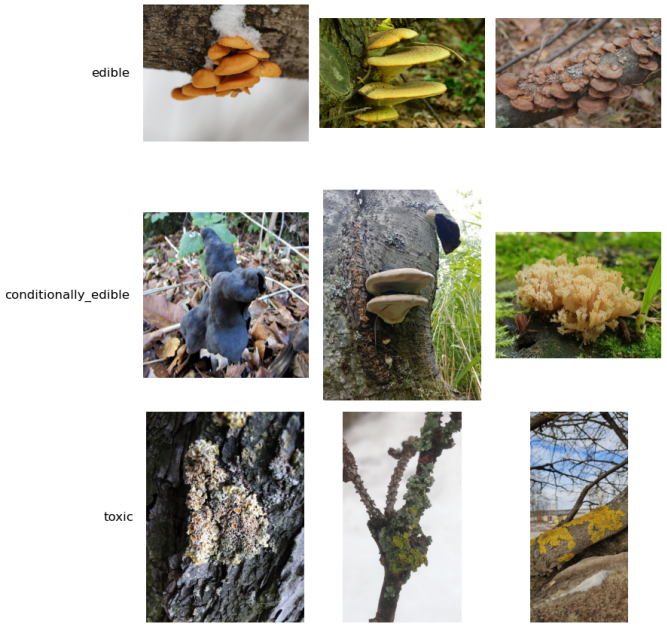

Fig. 3. Representative examples from the three edibility classes: edible (top), conditionally edible (middle), and toxic (bottom).

B. Probable Predictive Tasks

There are several predictive tasks can be done using this dataset:

- **Multi-class edibility classification:** determine the edibility category of a mushroom.
- **Multi-class species classification:** determine the species of a mushroom.
- **Binary toxicity classification:** determine whether a mushroom is toxic or non-toxic.

We focus on multi-class edibility classification for several reasons. This task is more challenging than binary toxicity classification as it requires predicting multiple categories rather than two classes. In addition, in practice, some mushrooms are toxic when raw but become edible after proper preparation, so binary classification is not able to capture these distinctions. We did not select the multi-class species classification task because the raw dataset contains over 100 species with limited samples per species, which could lead to overfitting.

C. Feature Engineering

For our models based on ResNet and Vision Transformer architectures, no explicit feature extraction is required because the models automatically learn hierarchical feature representations directly from raw pixel data. We only resize the images to 224×224 pixels and represent them as three-channel RGB tensors of size $224 \times 224 \times 3$ as input to each model.

IV. Methods

In this study, we evaluate both convolutional and transformer-based vision models to predict mushroom edibility from images. Models were implemented using the PyTorch framework [16] with pretrained architectures from TorchVision [17].

A. ResNet Architectures

ResNet-18 and ResNet-50 are convolutional neural networks that can be utilized for image classification tasks. These networks have residual blocks that sum up the input and output of each layer. This enhances the learning of identity functions and gradient backpropagation. This is because, in deep networks, gradients vanish during backpropagation. This issue can be solved in residual networks. The residual function can be defined as:

$$y = F(x, \{W_i\}) + x \tag{1}$$

where x is the input to a residual block and $F(x, \{W_i\})$ is the residual function learned by the block.

The primary difference between ResNet-18 and ResNet-50 is that they have 18 and 50 layers, respectively. This makes ResNet-50 deeper and has more capacity to learn deeper features.

The network outputs a vector of logits $z \in \mathbb{R}^C$, which are converted to class probabilities using the softmax function:

$$p_c = \frac{e^{z_c}}{\sum_{j=1}^{C} e^{z_j}} \tag{2}$$

where p_c is the predicted probability for class c.

For both ResNet-18 and ResNet-50, baseline models are trained to maximize overall classification accuracy using the standard Categorical Cross-Entropy (CE) loss. This function treats all classes equally and penalizes the negative log-likelihood of the true class:

$$\mathcal{L}_{\text{CE}} = -\sum_{c=1}^{C} y_c \log(p_c) \tag{3}$$

where y_c is the true class label and p_c is the predicted probability for class c.

Both enhanced ResNet-18 and ResNet-50 models prioritize detection of the toxic class because false negatives (predicting toxic mushrooms as edible) are more dangerous than false positives. Although they share the same safety motivation, the two models use different methods to achieve this objective.

B. Enhanced Model — ResNet18

The toxic class is assigned a larger weight in the cross-entropy loss to penalize misclassification of toxic mushrooms as edible more heavily. The weighted cross-entropy loss is defined as:

$$\mathcal{L}_{\text{WCE}} = -\sum_{c=1}^{C} w_c y_c \log(p_c) \tag{4}$$

where w_c is the class weight for class c, y_c is the true label, and p_c is the predicted probability.

Model selection is based on a safety-oriented priority score:

$$S_{\text{priority}} = 0.55\, R_{\text{toxic}} + 0.30\, F1_{\text{macro}} + 0.15\, Acc_{\text{balanced}} \tag{5}$$

where R_{toxic} is recall for the toxic class, $F1_{\text{macro}}$ is the overall performance, and Acc_{balanced} is the average accuracy score across all classes. This score prioritizes toxic detection while maintaining overall classification performance.

C. Enhanced Model — ResNet50

The enhanced ResNet-50 model uses Weighted Focal Loss. This loss introduces two critical modifications: a modulating factor and class weighting. The formulation is given by:

$$\mathcal{L}_{\text{focal}} = -\sum_{c=1}^{C} w_c y_c (1 - p_c)^{\gamma} \log(p_c) \tag{6}$$

where:

- $\gamma = 2.0$ is the focusing parameter,
- $w_{\text{toxic}} = 8.0$ is the class weight for the toxic class,
- p_c is the predicted probability for class c.

D. Swin Tiny Transformer

The Swin Tiny Transformer [8] is a hierarchical vision transformer that processes images by dividing them into non-overlapping patches and applying self-attention within local windows. Unlike the original Vision Transformer (ViT) [7], which applies global self-attention across all patches at once, Swin restricts attention to a fixed local window of size 7×7 patches and shifts those windows between successive layers. This shifted window mechanism allows the model to capture both local fine-grained features and cross-window global context, while keeping computational complexity linear with respect to image size rather than quadratic.

The architecture is hierarchical: patch features are merged progressively across four stages, producing feature maps at resolutions of $\frac{H}{4}$, $\frac{H}{8}$, $\frac{H}{16}$, and $\frac{H}{32}$. This is analogous to the feature pyramid in a CNN and makes Swin well-suited as a general-purpose backbone for dense prediction tasks as well as classification. Self-attention within each window is computed as:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^{\top}}{\sqrt{d}} + B\right) V \tag{7}$$

where Q, K, V are the query, key, and value matrices, d is the head dimension, and B is a learned relative position bias matrix. The position bias is a key difference from standard self-attention and allows the model to encode spatial structure within each window.

We initialized the model using ImageNet-pretrained weights via the `timm` library [18] and replaced the classification head with a linear layer mapping to three output classes. Three training configurations were evaluated. **Run 1** performed end-to-end fine-tuning from the first epoch using AdamW with a learning rate of 10^{-4} and weight decay of 0.01, optimizing standard cross-entropy loss for 20 epochs with cosine annealing. **Run 2** used a two-phase strategy: the backbone was frozen for 3 epochs while only the classification head was trained at 10^{-3}, followed by full fine-tuning with differential learning rates (10^{-4} for the head, 10^{-5} for the backbone), a linear warmup over 2 epochs, cosine annealing, and early stopping with patience of 7. **Run 3** repeated the Run 1 setup but added label smoothing ($\epsilon = 0.1$) to the cross-entropy loss, which softens the hard target distribution:

$$\mathcal{L}_{\text{LS}} = (1 - \epsilon)\, \mathcal{L}_{\text{CE}} + \frac{\epsilon}{C} \tag{8}$$

where $C = 3$ is the number of classes. Run 3 also applied a lowered decision threshold of 0.35 for the toxic class at inference time: if the softmax probability for toxic exceeded 0.35, the sample was classified as toxic regardless of the argmax prediction, prioritizing toxic recall for safety.

To interpret model decisions, Gradient-weighted Class Activation Mapping (Grad-CAM) [10] was applied to the final transformer block's normalization layer. Because Swin outputs a sequence of patch tokens rather than a spatial feature map, a reshape transform was applied to convert the 49-dimensional token sequence back to a 7×7 spatial grid before computing the CAM overlay.

V. Experiments/Results/Discussion

A. ResNet18

1) Hyperparameters: The baseline model used the standard cross-entropy loss function. It was optimized with AdamW [19] with a learning rate of 3×10^{-4} and a weight decay of 1×10^{-4}. The enhanced model used weighted cross-entropy with label smoothing ($\varepsilon = 0.05$) and assigned larger class weights to the toxic class. Training was performed with a mini-batch

size of 32, enabling efficient data loading while maintaining good generalization performance.

For optimization, the enhanced model uses layer-wise learning rates on AdamW: smaller learning rates (10^{-5}) for early layers and larger learning rates (10^{-3}) for later layers. When validation set performance plateaus, the ReduceLROnPlateau scheduler automatically decreases the learning rate.

The baseline model selected the best model based on the accuracy of the validation set in the training steps, while the enhanced model used the toxic-priority score in Eq. (5) as the criteria to select the best model since false negatives (predicting toxic mushrooms as edible) have the greatest risk. During training, the models will run with an early stopping patience of 5.

2) Evaluation Metrics: We evaluate performance using Overall Accuracy, Precision, Recall, and F1-score.

Accuracy is defined as:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{9}$$

Precision and Recall for the toxic class are:

$$\text{Precision} = \frac{TP}{TP + FP}, \quad \text{Recall} = \frac{TP}{TP + FN} \tag{10}$$

The F1-score balances Precision and Recall:

$$F1 = \frac{2PR}{P + R} \tag{11}$$

3) Results: Figures 4 and 5 show the confusion matrices of the baseline and enhanced ResNet-18 models on the test set, generated using matplotlib [20] and seaborn [21].

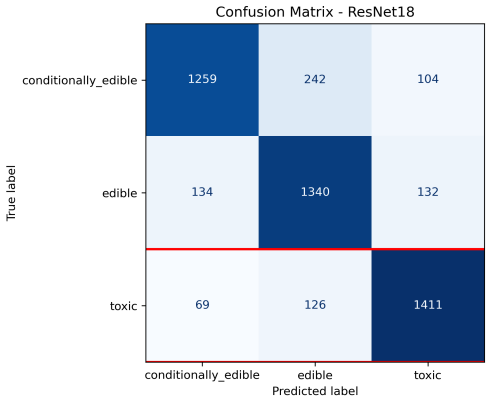

Fig. 4. Confusion matrix of the baseline ResNet-18 model on the test set.

Table I summarizes the performance for test-set, including overall metrics and toxic class related metrics.

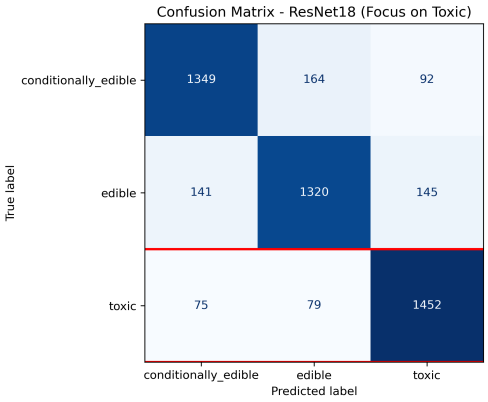

Fig. 5. Confusion matrix of the enhanced ResNet-18 model with toxic-focused training.

TABLE I
TEST PERFORMANCE COMPARISON — RESNET18

Metric	Baseline	Enhanced
Overall Accuracy	0.8325	0.8555
Macro F1	0.8324	0.8552
Toxic Recall	0.8786	**0.9041**
Toxic Precision	0.8567	0.8597
Toxic F1	0.8675	0.8813
False Negatives (Toxic)	195	154

4) Discussion: The enhanced model outperformed the baseline across all reported metrics. Toxic recall increased from 87.9% to 90.4%, and false negatives for the toxic class decreased from 195 to 154. Overall accuracy and macro F1 also improved from about 83% to 85.5%. The enhanced model reduced the risk of misclassifying toxic mushrooms to edible or conditionally edible, while maintaining a moderate to strong overall performance. However, further improvements are still possible.

B. ResNet50

1) Experimental Setup and Hyperparameters: One **Baseline** and two **Robust** models were trained on the ResNet-50 V2 architecture. The Robust model's training was optimized for recall, where false negatives, i.e., misclassifying toxic mushrooms as edible, pose a greater risk than false positives. Model 2 warmup_scheduler is used to stabilize gradient updates at the initial stage of the training, followed by cosine annealing to refine convergence toward a robust solution. See table II for Comparison of Hyperparameter Configuration for ResNet-50 Models.

2) Comparative Results and Discussion: Table III summarizes the performance of the three models.

Figure 6 shows the confusion matrix of the Robust Model 3 on the test set.

The primary objective of the Robust models was to minimize false negatives in the Toxic class—mushrooms incor-

TABLE II
COMPARISON OF MODEL CONFIGURATIONS — RESNET50

Parameter	Baseline M1	Robust M2	Robust M3
Learning Rate	3×10^{-4}	3×10^{-4}	Warmup + Cosine Decay
Loss Function	Standard Cross Entropy	Weighted Cross Entropy [1.0, 1.0, 8.0]	Weighted Cross Entropy [1.0, 1.0, 8.0]
Early Stopping Metric	Validation Loss	Validation F2 Score	Validation F2 Score
Class Weight Focus	Balanced	Toxic class (8×)	Toxic class (8×)

TABLE III
TEST PERFORMANCE COMPARISON - RESNET50

Metric	Baseline M1	Robust M2	Robust M3
Overall Accuracy	0.8815	0.8902	0.8887
Macro F1 Score	0.8811	0.8901	0.8886
Toxic Precision	0.8757	0.9065	0.9118
Toxic Recall	0.9215	0.9172	**0.9234**
Toxic F1	0.8981	0.8960	0.9004

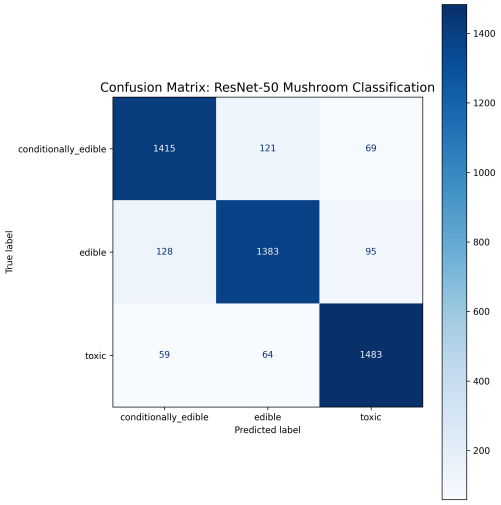

Fig. 6. Confusion matrix of Robust Model 3 on the test set.

rectly classified as edible. Using a class weight of 8.0 for toxic samples, both Robust models successfully reduced toxic misclassifications compared to the baseline.

Model 3 is the best model for this task because it maximizes safety by correctly identifying the most toxic samples (1,483 vs. Model 2's 1,473) while misclassifying fewer toxic mushrooms as edible (64 vs. Model 2's 76).

Although Model 2 achieves the highest overall accuracy (89.02% vs. 88.87%), Model 3's Toxic Recall (92.34% vs. 92.17%) and Toxic Precision (90.04% vs. 89.60%) yield the highest Toxic F1-score (91.18% vs. 90.65%), making it the most reliable choice for deployment where misclassifying toxic mushrooms as edible carries the greatest risk.

C. Swin Tiny Transformer

1) Experimental Setup and Hyperparameters: All three Swin Tiny runs used a batch size of 32, chosen to balance GPU memory on the DSMLP cluster with stable gradient estimation. Strong data augmentation was applied during training including random horizontal flips, rotation, color jittering, RandAugment, and random erasing ($p = 0.25$). Validation and test sets used only resizing and normalization. All runs used AdamW with weight decay 0.01, which applies decoupled L2 regularization to reduce overfitting. No cross-validation was performed; a fixed stratified 80/10/10 split was used throughout.

Run 1 fine-tuned end-to-end for 20 epochs at $lr = 1\times10^{-4}$ with cosine annealing, which gradually reduces the learning rate following a cosine curve to avoid overshooting minima in later epochs, and standard cross-entropy loss. **Run 2** used a two-phase strategy: head-only training for 3 epochs at 1×10^{-3}, then full fine-tuning with differential learning rates (1×10^{-4} for the head, 1×10^{-5} for the backbone) and early stopping (patience = 7). **Run 3** repeated Run 1 with label smoothing ($\epsilon = 0.1$):

$$\mathcal{L}_{\text{LS}} = (1 - \epsilon)\,\mathcal{L}_{\text{CE}} + \frac{\epsilon}{C} \tag{12}$$

where $C = 3$ is the number of classes. This discourages overconfident predictions by softening hard one-hot targets. At inference, a lowered decision threshold of 0.35 was applied to the toxic class. Any sample whose softmax toxic probability exceeded this value was classified as toxic regardless of the argmax.

2) Evaluation Metrics: Primary metrics are overall accuracy, per-class precision, recall, and F1-score. Toxic recall is treated as the most critical metric since a false negative, predicting toxic as edible, poses direct risk of harm. Overall accuracy alone is insufficient to evaluate safety performance, as a model can score well overall while still failing dangerously on the toxic class.

3) Results: Table IV summarizes test performance across all three configurations. Run 1 achieved the best overall accuracy at 92.32% with a toxic recall of 92.71%, demonstrating that simple end-to-end fine-tuning is a strong baseline. Run 2 underperformed at 90.68%, suggesting phased training did not benefit this dataset. Run 3 with label smoothing alone achieved 92.11%, and applying the 0.35 toxic threshold boosted toxic recall to **94.02%**, the best of any Swin configuration, at a modest cost to overall accuracy (92.26%).

Fig. 7 and Fig. 8 show the confusion matrices for Run 1 and Run 3 respectively. Both matrices show a strongly dominant diagonal, confirming reliable classification across all three classes. In Run 1, the toxic row shows 49 misclassifications as conditionally edible and 68 as edible, for 117 total toxic false negatives. Run 3 reduced this to 46 and 70 respectively (116 total), and with the 0.35 threshold applied these false

TABLE IV
TEST PERFORMANCE COMPARISON — SWIN TINY

Metric	Run 1	Run 2	Run 3	Run 3+T
Accuracy	0.9232	0.9068	0.9211	0.9226
Macro F1	0.9232	0.9067	0.9212	0.9226
Toxic Recall	0.9271	0.9234	0.9278	**0.9402**
Toxic Precision	0.9394	0.9143	0.9330	0.9258

negatives drop further. The largest off-diagonal errors in both runs occur between edible and conditionally edible rather than between toxic and non-toxic, which is the most encouraging pattern from a safety standpoint.

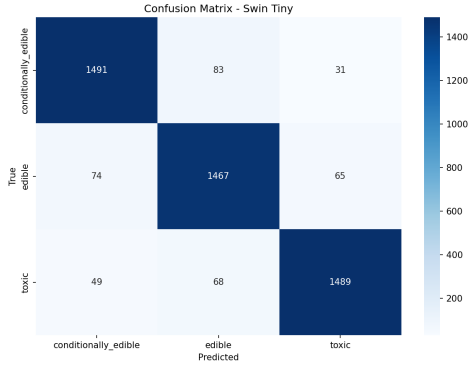

Fig. 7. Confusion matrix for Swin Tiny Run 1 (92.32% accuracy).

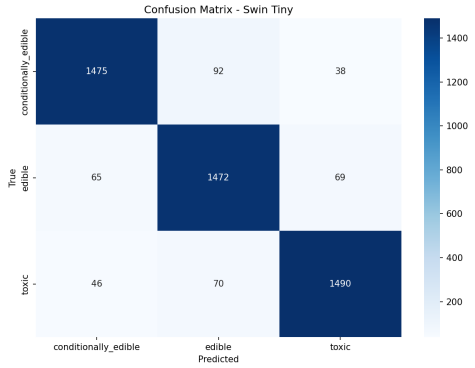

Fig. 8. Confusion matrix for Swin Tiny Run 3 (92.11% accuracy). Label smoothing slightly shifts errors but the overall pattern is similar to Run 1.

4) Grad-CAM Analysis: Fig. 9 shows Grad-CAM activations on correctly classified Run 1 samples. The model consistently focuses on the mushroom cap and gill area, with the strongest activation (red-orange regions) centered on the most distinctive parts of each mushroom. This shows the model is learning to recognize actual mushroom features rather than irrelevant background details like soil or leaves.

Fig. 10 shows misclassified samples from Run 2. Unlike Run 1, the activations here are spread out and often land on stems, background foliage, or the edges of the image instead of the cap. This helps explain why Run 2 performed worse.

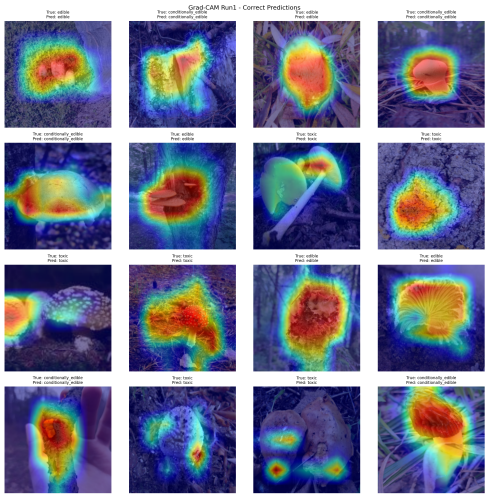

Fig. 9. Grad-CAM on correct predictions (Run 1). Activation focuses on cap and gill structures across all three classes.

The two-phase training approach resulted in the model paying attention to the wrong parts of the image.

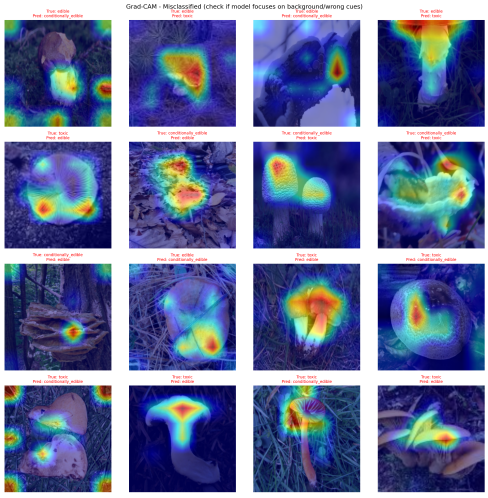

Fig. 10. Grad-CAM on misclassified samples (Run 2). Activation shifts to stems and background rather than the cap.

Fig. 11 shows Grad-CAM on toxic mushroom samples from Run 3. When the model correctly identifies a toxic mushroom (top row), it focuses tightly on distinctive surface features like spotted caps and unusual coloring. When it makes a mistake and predicts edible instead (bottom row), the activation spreads to the gill edges or outer cap boundary. This suggests the model struggles with toxic mushrooms that lack clear visual markers, causing it to rely on less reliable features that look similar to edible species.

5) Discussion: Run 1 performed the best overall, showing that straightforward fine-tuning from pretrained ImageNet weights is the most effective approach for this task. The two-phase training in Run 2 added complexity without any benefit, and the Grad-CAM visualizations make it clear why

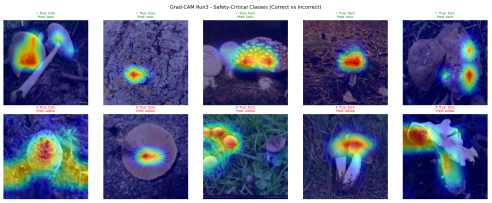

Fig. 11. Grad-CAM on toxic samples (Run 3): correctly classified (top) vs. misclassified as edible (bottom).

— the model ended up focusing on the wrong parts of the image. Label smoothing in Run 3 had little effect on its own, but combining it with a lower toxic decision threshold of 0.35 produced the best toxic recall of any configuration at 94.02%. This threshold approach is simple to apply at inference time and requires no retraining, making it a practical tool for improving safety in deployment. Across all three runs, most mistakes happened between the edible and conditionally edible classes, which look very similar to each other. Errors between toxic and non-toxic were much less common, which is encouraging from a safety standpoint. Overall, the Swin Tiny model proved to be a strong and reliable architecture for this classification task, achieving over 92% accuracy while maintaining high toxic recall.

VI. Conclusion

In conclusion, the threshold-tuned Swin Tiny Transformer is the strongest training model, achieving 92.26% overall accuracy while boosting toxic recall to 94.02%, and by lowering the decision threshold to 0.35.

The simple fine-tuning approach of the Swin Transformer outperforms the more complex training methods used in the RESNET50 Robust models. It demonstrates that the dataset is sufficiently large to leverage pretrained weights effectively without careful handling, and meaningful gains in safety-critical performance justify the modest accuracy trade-off.

From a safety standpoint, a toxic recall should be the primary metric for this classifier — a model that misclassifies a toxic mushroom as edible is far more dangerous than one that misclassifies an edible mushroom as toxic. On that basis, the threshold-tuned Swin model is the strongest candidate for deployment.

VII. Reply to Review

A. ***Comment 1:*** *On the slide with the feature hierarchy and the table with channel as a column, it wasn't clear to me what the channel is or why it's increasing. Maybe you can expand on that!*

The "channel" column refers to the depth of the feature maps (the number of filters) at a given layer in the network. The channel count increases as the spatial resolution (height and width) decreases. This is a standard architectural design (as in ResNet or VGG) that allows the model to extract a richer, more diverse set of high-level semantic features as the data becomes more compressed. Increasing the number of channels ensures the model has sufficient "representative capacity" to capture complex patterns that simpler, earlier layers might miss.

B. ***Comment 2:*** *There was an explanation of fine-tuning in the code overview, but it would have been nice to hear about it with some more details in the presentation. Could doing a grid search for hyperparameters potentially make sense?*

Fine-tuning involved initializing with ImageNet weights then updating all layers with a low learning rate (3e-4). Grid search absolutely makes sense—we could systematically explore learning rates (1e-3 to 1e-5), weight decay values, and class weight ratios to find optimal combinations. This would be more rigorous than our current fixed approach.

C. ***Comment 3:*** *Instead of using early stopping, it may be useful to compare it to other methods like pruning.*

Early stopping prevents overfitting by halting training when validation performance plateaus. Pruning reduces model size after training by removing less important weights/neurons. They serve different purposes—early stopping affects training duration, and pruning affects model architecture. Both could be combined: early stopping during training, then pruning for deployment efficiency.

VIII. Contributions

A. Chiu-Chiu

- Wrote the Abstract and Introduction, outlining the motivation for the research, the definition of problems, and the general objectives of the study.
- Conducted EDA and visualization, including dataset quality checks, class distribution analysis, image dimension distribution, and aspect-ratio analysis, and later implemented exact and near-duplicate detection to identify and remove duplicate images.
- Designed data standardization and data augmentation strategies to improve model training robustness.
- Implemented the initial ResNet-50 model and conducted preliminary experiments to evaluate its feasibility for mushroom edibility classification.
- Reviewed and revised sections of the full report and actively participated in team discussions.

B. Joanna

- Completed ResNet-50 in Methods, Experiments, Results, Discussion and Conclusion sections.
- Generated a cleaned dataset and exported the processed data into a structured CSV file for downstream model training and experiments.
- Implemented and evaluated the ResNet-50 model, including fine-tuning, training configuration and experimental evaluation.
- Actively participated in team discussions.

C. Nicholas

- Completed the Related Work, References and Swin Transformer in the Methods, Experiments, Results, Discussion and Conclusion sections of the report.
- Summarizing prior research on mushroom classification using convolutional neural networks and transformer-based vision models.
- Implemented, trained and conducted analysis for the Swin Tiny Transformer model, including multiple fine-tuning strategies and hyperparameter configurations.
- Generated confusion matrices and Grad-CAM visualizations to interpret model predictions and analyze classification behavior.
- Actively participated in team discussions.

D. Xuewen

- Consolidated image datasets organized in a directory hierarchy into a unified CSV file with metadata for each image (e.g., paths, dimensions, and resolutions).
- Trained and conducted the analysis of ResNet-18 baseline model and improved model.
- Completed the Dataset and Features and ResNet-18 sections of the report.
- Coordinated the creation of figures and tables and ensured proper referencing throughout the report.
- Reviewed and revised sections of the full report and actively participated in team discussions.

IX. Reflections

If we were to undertake this project again, we would adopt a different approach. We would focus more intently on the model design currently under development. Regarding experimental design, the Swin Transformer model demonstrated the best performance among all models tested. This indicates that novel Transformer models are better suited for this task than Convolutional Neural Network (CNN) models. Therefore, we would prioritize optimizing the Transformer model to strive for further performance improvements.

We would also rethink our model training methodology. Certain approaches would not be attempted early on, as some have proven to be of limited practical value. Concurrently, we would redouble efforts to ensure all mushroom species are accurately identified. Misclassifying poisonous mushrooms as edible carries extremely dangerous consequences. Techniques like class weights and focal loss could be employed to maximize edible mushroom detection while minimizing false negatives, thereby improving overall recognition rates.

Data set quality will also be optimized. We will ensure the data set is complete and error-free, documenting the quantitative distribution of each mushroom type. Supplementary image materials may even be added to enhance completeness. Finally, we will delve into the model's decision-making mechanisms, using techniques like Grad-CAM to observe the visual focus points during model predictions. These efforts will aid system optimization. By adopting more streamlined training methods and higher-quality data sets, we will ultimately build an efficiently operating system.

References

[1] H. Yin, W. Yi, and D. Hu, "Computer vision and machine learning applied in the mushroom industry: A critical review," *Computers and Electronics in Agriculture*, vol. 198, p. 107015, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0168169922003325

[2] M. S. Jacob, A. Xu, K. Qian, Z. Qi, X. Li, and B. Zhang, "Artificial intelligence in edible mushroom cultivation, breeding, and classification: A comprehensive review," *Journal of Fungi*, vol. 11, no. 11, 2025. [Online]. Available: https://www.mdpi.com/2309-608X/11/11/758

[3] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," 2015. [Online]. Available: https://arxiv.org/abs/1512.03385

[4] EPFL. (2019) Mushroom recognition using image processing. Accessed: 2026-02-01. [Online]. Available: https://www.epfl.ch/labs/mmspg/wp-content/uploads/2019/01/Semesterproject_mushroomrecognition.pdf

[5] F. Wang, J. Zheng, L. Wang, W. Feng, and L. Niu, "Classification method research of fresh agaricus bisporus based on image processing," *Computer and Computing Technologies in Agriculture*, pp. 333–340, 2017.

[6] J. Zhao, W. Zheng, Y. Wei, Q. Zhao, J. Dong, X. Zhang, and M. Wang, "Machine vision-based detection of key traits in shiitake mushroom caps," *Frontiers in Plant Science*, vol. 16, p. 1495305, 2025.

[7] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, "An image is worth 16x16 words: Transformers for image recognition at scale," 2021. [Online]. Available: https://arxiv.org/abs/2010.11929

[8] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo, "Swin transformer: Hierarchical vision transformer using shifted windows," 2021. [Online]. Available: https://arxiv.org/abs/2103.14030

[9] M. Caron, H. Touvron, I. Misra, H. Jégou, J. Mairal, P. Bojanowski, and A. Joulin, "Emerging properties in self-supervised vision transformers," 2021. [Online]. Available: https://arxiv.org/abs/2104.14294

[10] A. Zakeri, M. Fawakherji, J. Kang, B. Koirala, V. Balan, W. Zhu, D. Benhaddou, and F. A. Merchant, "M18k: A multi-purpose real-world dataset for mushroom detection, 3d pose estimation, and growth monitoring," *Computers*, vol. 14, no. 5, 2025. [Online]. Available: https://www.mdpi.com/2073-431X/14/5/199

[11] Z. Abdin, "Mushroom classification dataset," Kaggle, 2023, accessed: 2026-02-01. [Online]. Available: https://www.kaggle.com/datasets/zedsden/mushroom-classification-dataset

[12] J. Buchner, "Imagehash," https://pypi.org/project/ImageHash/, 2024, accessed: 2026-03-12.

[13] F. Pedregosa and et al., "Scikit-learn: Machine learning in python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[14] C. R. Harris and et al., "Array programming with numpy," *Nature*, vol. 585, pp. 357–362, 2020.

[15] W. McKinney, "pandas: a foundational python library for data analysis and statistics," 2011.

[16] A. Paszke, S. Gross, F. Massa, and et al., "Pytorch: An imperative style, high-performance deep learning library," *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[17] PyTorch Contributors, "Torchvision: Pytorch's computer vision library," https://pytorch.org/vision, 2025.

[18] R. Wightman, "Pytorch image models," https://github.com/rwightman/pytorch-image-models, 2024.

[19] PyTorch Contributors, "torch.optim.adamw," https://docs.pytorch.org/docs/stable/generated/torch.optim.AdamW.html, accessed: 2026-03-12.

[20] J. D. Hunter, "Matplotlib: A 2d graphics environment," *Computing in Science & Engineering*, vol. 9, no. 3, pp. 90–95, 2007.

[21] M. Waskom, "Seaborn: Statistical data visualization," https://seaborn.pydata.org, 2024.