Multiclass Mushroom Classification

Wei-Cheng Hsu
Wirtschaftsinformatik M.Sc.
University of Applied Sciences Karlsruhe
Karlsruhe, Germany

Ilian Kohl
Informatik M.Sc.
University of Applied Sciences Karlsruhe
Karlsruhe, Germany

Michael Florath
Informatik M.Sc.
University of Applied Sciences Karlsruhe
Karlsruhe, Germany

***Abstract*—This study investigates the multiclass classification of the publicly available Kaggle mushroom dataset, focusing on 10 species found in Estonia. Transfer learning was applied to pre-trained models—AlexNet, ResNet, VGG16, DenseNet, and EfficientNet—fine-tuned with layer freezing strategies and evaluated using accuracy, per-class accuracy, precision-recall curves, and ROC curves. The effect of dropout layers on overfitting mitigation and generalization was analyzed in fine-tuned AlexNet and ResNet models. To enhance interpretability, Explainable AI techniques, including Concept Bottleneck Model and Integrated Gradients were employed. The findings offer insights into model selection, optimization, and interpretability for mushroom classification via transfer learning.**

Keywords—Computer Vision, Multiclass Classification, Transfer Learning, Explainable AI, Performance Optimization

I. Introduction

Mushroom classification plays an important role in agriculture and ecology, particularly in distinguishing edible species from toxic ones. Originally introduced in a Kaggle competition, this task involves the classification of 10 distinct mushroom species found in Estonia, as illustrated in "Fig. 1". By leveraging deep learning techniques in computer vision, this challenge presents an opportunity to investigate model performance and optimization in real-world multiclass classification problems.

II. Problem Statement

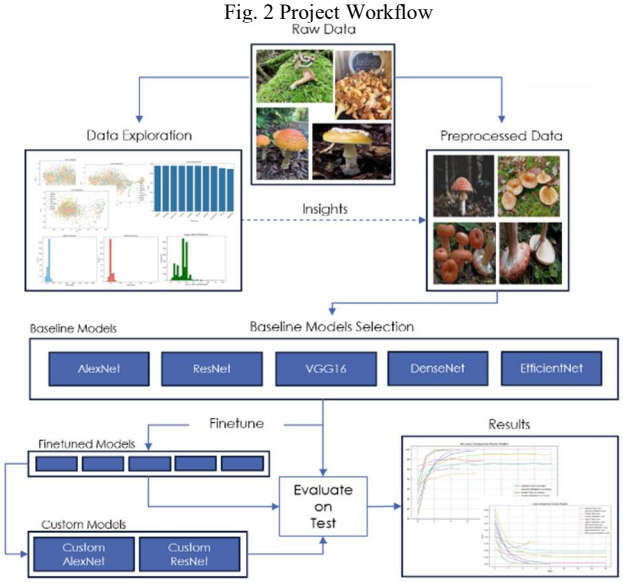
A. Problem Description

The goal of this study was to develop a model capable of classifying 10 different mushroom species using deep learning techniques. The species included are *Amanita, Boletus, Chantelle, Deterrimus, Rufus, Torminosus, Aurantiacum, Procera, Involutus, and Russula.*

The classification task is complicated by several challenges, such as the small dataset size, variability in image quality, and presence of distractor samples. To achieve high accuracy in distinguishing between these species, a robust classifier is required. Additionally, challenges like overfitting due to the small dataset size and ensuring the interpretability of the model's predictions must also be addressed.

Fig. 1 Mushroom Species of Kaggle Dataset

Fig. 2 Project Workflow



B. Dataset Challenges

The dataset used in this study is sourced from Kaggle and consists of approximately 3,000 images representing 10 mushroom species found in Estonia [1]. The dataset is relatively balanced, with approximately 300 images per class. The dataset includes two CSV files: *Train.csv*, containing 2,371 labeled images, and *Test.csv*, containing 600 unlabeled images, leaving 29 images unaccounted for.

The dataset presents several challenges:

- **Distractor Samples**: Some images include irrelevant elements such as drawings, human hands, text in focus, or mushrooms processed in glass containers.
- **Image Size Variability:** The images vary in size, requiring resizing for consistency.
- **Small Dataset Size:** With only around 3,000 images, the dataset is relatively small, posing challenges for generalization and increasing the risk of overfitting.

III. · Approach

The approach to solving this classification problem, as shown in "Fig. 2", began with data exploration to gather key insights in order to make informed decisions in the following data preprocessing steps. Pretrained models were then leveraged by applying transfer learning to create a strong initial classifier, using two different layer freezing strategies. To further enhance model performance, the fine-tuned AlexNet and ResNet models were selected for optimization, by applying regularization techniques to counter overfitting and improve generalization. The final evaluation of the models was conducted using key performance metrics, including accuracy, precision, recall, and per-class accuracy,

with results visualized through ROC and precision-recall curves and confusion matrix. Explainable AI (XAI) methods were explored to improve interpretability, including the implementation of a Concept Bottleneck Model (CBM) and the analysis of feature extraction using Integrated Gradients.

A. Data Exploration

The data exploration phase aimed to gain insights into the dataset, identify patterns, and define relevant concepts for the training of the CBM. First, the class distribution of the 10 mushroom species in train.csv was analyzed. The distribution in "Fig. 3" revealed that the first six classes contained 240 samples each, while *Amanita* and *Aurantiacum* had 238 samples, and *Rufus* and *Deterrimus* had the fewest samples, with 227 and 222, respectively.

To further understand the dataset, images from each class were examined. Visualizations of these images helped inform the definition of concepts for the CBM. For instance, the *Chantelle* class was predominantly yellow and often appeared in images with multiple mushrooms, as this species typically grows in groups. The *Amanita* class exhibited red mushroom caps with clearly defined warts and a characteristic ring. The *Procera* class also displayed images with visible scales and a ring, though the cap shape varied with the mushroom's growth stages, ranging from bulbous to convex as the mushroom matured. These observations led to the definition of concepts: *cap_color, cap_shape, cap_texture, ring_present*

Next, the image size distribution was analyzed to establish a reference for preprocessing, as shown in "Table I.".

TABLE I. IMAGE SIZE DISTRIBUTION

Statistic	***Width***	***Height***
Mean	251.11	204.44
Median	259.00	194.00

The UMAP dimensionality reduction did not reveal 10 distinct mushroom classes, as expected, due to the high-dimensional nature of the image data [6]**.** However, UMAP demonstrated superior performance compared to t-SNE and PCA by highlighting two main groupings in "Fig. 4". Further analysis of the smaller grouping revealed that the images predominantly featured white backgrounds and drawings. It remains unclear whether this separation is primarily driven by underlying features of these drawings, the white backgrounds, or a combination of both, as some real mushroom images were also cropped against white backgrounds, while not all drawings have a white background. Additionally, per-class clustering revealed two distinct clusters for some classes like *Amanita*, as illustrated in "Fig. 5". Upon inspection, the smaller clusters within these classes were also found to contain images with white backgrounds, highlighting a recurring pattern in the data.

B. Data Preprocessing

Although the image size distribution showed variability, resizing the images to 224x224 was chosen to align with standard input sizes used by pretrained models such as AlexNet, VGG16 and ResNet. This resizing choice provides a balance between preserving important image details and optimizing computational efficiency. Furthermore, it ensures consistency across all images while being close to the median dimensions, making it a practical choice for model training.

Fig. 3 Class distribution

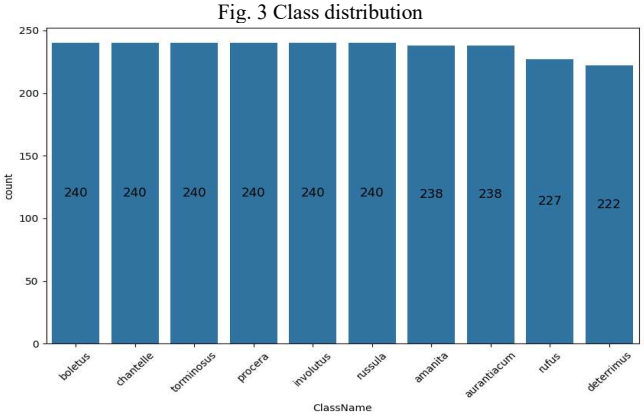


Fig. 4 Dimensionality Reduction using UMAP

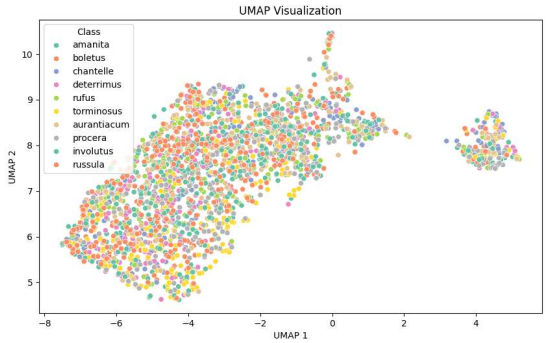
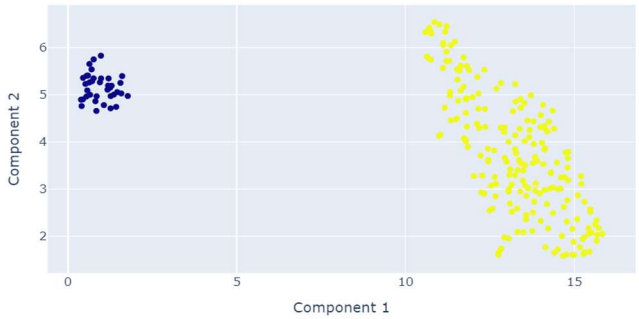


Fig. 5 UMAP + K-Means Clustering for class Amanita



To increase image variability and improve generalization, random horizontal flipping and rotations (up to 20 degrees) were applied, helping to reduce overfitting, especially with small datasets. Color jittering was used to adjust brightness and contrast, further diversifying the dataset. Finally, images were normalized using the ImageNet dataset's mean and standard deviation values, to ensure consistency with the pretrained weights for transfer learning.

For the CBM, data augmentation involved assigning generic concept labels to all images in the training set (train.csv) based on their class. Rather than manually annotating individual concepts for each image, we assumed concept attributes were consistent within each class to simplify the annotation process. For example, all *Amanita* mushrooms were labeled with the concept *red* for *cap_color*, even though some *Amanita* mushrooms have a brown cap. While this approach introduced noise due to the assumption of consistency across all images within a class, it was a practical and efficient way to enable training of the CBM, allowing us to focus on high-level concepts that are most representative of each class.

These concepts and their corresponding values are:

- ***cap_color:*** *red, brown, yellow, white*
- ***cap_shape:*** *convex, flat, bulbous, inverted*
- ***cap_texture****: smooth, scaly, warty*
- ***ring_present****: yes, no*

Since only *train.csv* is labeled, it was used to create new splits for training, validation, and testing. Initially, the 2371 labeled images were split into 80% for training and 20% for testing. This resulted in 1896 images for the training set and 474 images for the testing set. The training set (1896 images) was then further split, with 80% allocated for training (1517 images) and 20% reserved for validation (379 images).

C. Selection of Baseline Models

The models pretrained on ImageNet—AlexNet, ResNet, VGG16, DenseNet, and EfficientNet—were selected as baselines due to their architectural strengths and suitability for the mushroom classification task [5]:

- **AlexNet:** Simple architecture for flexible exploration and fast training.
- **ResNet:** Residual connections to address the vanishing gradient problem and enable deep networks.
- **VGG16**: Straightforward design with robust transfer learning, ideal for small datasets.
- **DenseNet**: Feature reuse through dense connections to reduce overfitting with limited data.
- **EfficientNet**: Optimized depth, width, and resolution for balanced performance and efficiency.

D. Transfer Learning and Training

Each baseline model was initialized with pretrained IMAGENET1K_V1 weights, leveraging prior knowledge learned from large-scale natural image datasets. The final classification layer of each model was replaced with a fully connected layer containing 10 output units to align with the mushroom classification task. To preserve the learned feature representations from ImageNet, all layers were frozen except for the newly added classification head, which was fine-tuned on the mushroom dataset. Subsequently, the entire model, including the feature extraction layers, were unfrozen and fine-tuned to enable the learning of task-specific features at all levels of abstraction. This step aimed to leverage both generic and task-specific representations to optimize the model's performance for mushroom classification [3].

For training, Cross-Entropy Loss was used as the loss function, which is commonly employed for multi-class classification tasks for N classes.

$$Cross - Entropy\ Loss = -\sum_{i=1}^{N} (y_i \log(p_i)) \qquad (1)$$

A batch size of 32 was used and the models were optimized using the AdamW optimizer with a learning rate of 0.0001. The StepLR scheduler with step=1 was utilized to adjust the learning rate by a factor of 0.5 after every epoch, to promote convergence as the training progressed. Momentum was set to 0.9 to accelerate gradients in the relevant direction, and weight decay of 1e-5 was applied to prevent overfitting by penalizing large weights.

Additionally, early stopping was implemented with a patience of 3, halting the training if the validation loss did not improve for three consecutive epochs, thereby mitigating overfitting and conserving computational resources.

E. Optimization

To improve the performance and reduce overfitting of the finetuned models on the relatively small mushroom dataset, two approaches were applied:

- For **ResNet**, layers 2-4 were unfrozen for fine-tuning higher-level features, while the first layer remained frozen. The classifier head was replaced with a 512-unit FC layer, followed by batch normalization, dropout, and a 10-output classification layer.
- For **AlexNet**, early convolutional layers (up to layer 10) were frozen, with later layers fine-tuned. A new classifier head with a 1024-unit FC layer, batch normalization, and dropout was applied.

F. Explainability Methods

To increase interpretability two methods were used:

1) *Integrated Gradients*

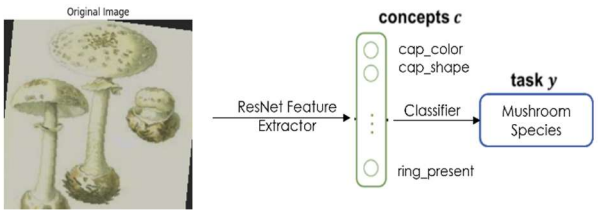
Integrated Gradients is a local, model-agnostic attribution method that assigns importance scores to each feature of the input image, by interpolating between a baseline (zero image) and the input image across multiple steps, calculating the gradients of the output with respect to these interpolated images, and then averaging the gradients. This results in an attribution map that highlights the most influential regions of the image, offering a visual explanation of the model's decision-making. The generated heatmaps show which features are most important for predicting the given class, providing insights into the model's behavior at a local level [2].

2) *Concept Bottleneck Model*

The CBM consists of two main components, as shown in "Fig.6": a feature extractor mapping inputs to the intermediate concepts defined during preprocessing and a classifier using the learned concepts for the final class prediction. Linking the model's decisions to humanly understandable concepts enables intervention. By modifying concept predictions, one can see how changes in intermediate concepts influence the classification [4].

- **Input to Concepts:** The image is passed through the ResNet-50 feature extractor, with the final FC layer replaced to predict four concepts: cap color, cap shape, cap texture, and ring presence.
- **Concepts to Classification:** The predicted concepts are fed into a classifier with two FC layers: the first has 128 units and ReLU activation, and the second outputs the final 10-class prediction.

Fig. 6 Concept Bottleneck Model Architecture [4]



Both methods were chosen for their complementary strengths: Integrated Gradients to visualize influential image regions, and CBM to understand how changes in intermediate concepts impact the final prediction.

IV. · Evaluation

The fine-tuned models were first evaluated with all layers frozen except for the final classifier head, followed by an evaluation where both the classifier head and the model layers were fine-tuned. In the final stage, the fine-tuned AlexNet and ResNet models with applied regularization methods were evaluated.

A. Fine-tuned Models with layer freeze

AlexNet and VGG16 achieved the highest training accuracy, above 90%, indicating effective use of pre-trained weights. However, both models showed significant overfitting, with validation accuracies of 73% (AlexNet) and 68% (VGG16), highlighting a large gap between training and validation performance. ResNet exhibited a smaller gap between training and validation accuracy, suggesting better generalization, but with around 50% a lower overall accuracy compared to AlexNet and VGG16. DenseNet and EfficientNet showed similar trends, with small discrepancies between training and validation accuracy, but both performed poorly with accuracies around 33%, likely due to the complexity of their architectures relative to the small dataset, as can be seen in “Fig. 7”.

B. Fine-tuned Models without layer freeze

After fine-tuning without layer freezing, ResNet achieved 99% training accuracy and 88% validation accuracy after 10 epochs, showing signs of overfitting, as seen in the gap between training and validation performance in the accuracy plot "Fig. 8". Similarly, AlexNet showed overfitting with 90% training accuracy and 73% validation accuracy after 12 epochs, while VGG16 had 97% training accuracy and 80% validation accuracy after 5 epochs. DenseNet and EfficientNet demonstrated a similar trend, with DenseNet reaching 98% training accuracy and 83% validation accuracy after 9 epochs, and EfficientNet achieving 84% training accuracy and 76% validation accuracy after 12 epochs.

The loss plot "Fig. 9" confirms these findings. While ResNet had a relatively small gap in training and validation loss, it still displayed overfitting due to the increase in validation loss. AlexNet and VGG16 had more pronounced overfitting, with AlexNet showing a training loss of 0.3 and a validation loss of 0.75, and VGG16 showing a training loss of 0.1 and a validation loss of 0.65. DenseNet showed less overfitting, with training and validation losses of 0.25 and 0.55, respectively, while EfficientNet demonstrated the most significant loss, with a training loss of 0.88 and a validation loss of 1.00. The final test accuracies can be derived from “Table II.”

TABLE II. Comparison of Fine-tuned Models without Layer Freeze on Test Accuracy

	AlexNet	***ResNet***	***VGG16***	***DenseNet***	***Efficient Net***
Test Accuracy	71.83%	89.01%	78.87%	85.35%	80.28%

Fig. 7 Accuracy comparison of fine-tuned models with layer freeze

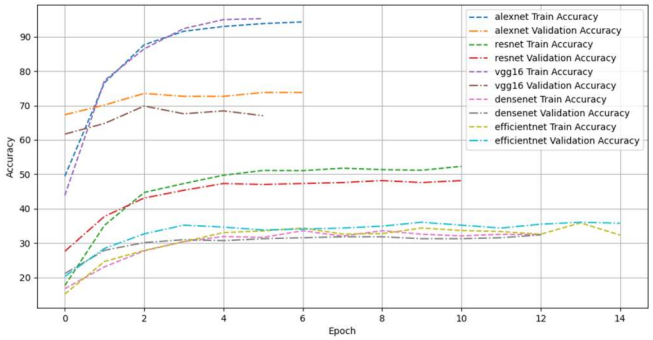


Fig. 8. Accuracy comparison of fine-tuned models without layer freeze

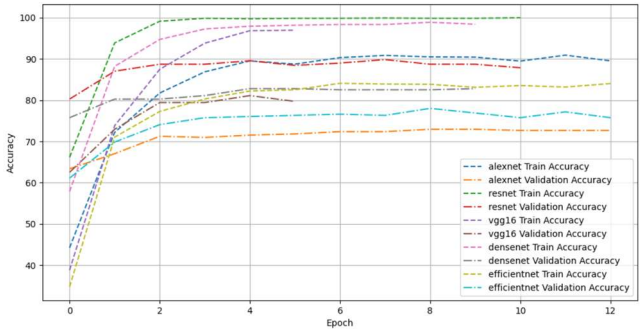
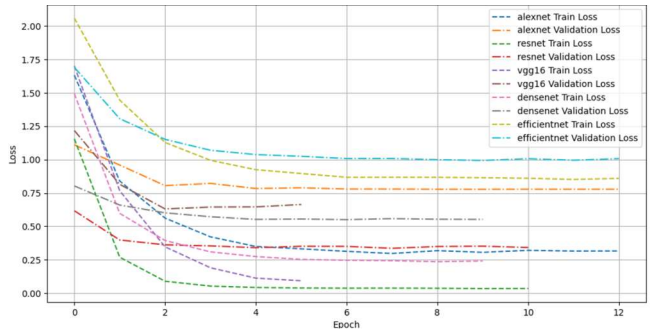


Fig. 9. Loss comparison of fine-tuned models without layer freeze



C. Improved fine-tuned AlexNet & ResNet

The fine-tuned AlexNet model achieved a test accuracy of 67.61% with a corresponding test loss of 1.0438, indicating worse classification performance compared to the fine-tuned without regularization. Training concluded at epoch 13, as early stopping was triggered due to no further improvement in validation performance. The final validation accuracy reached 65.92%, while the training accuracy stood at 68.58%, which shows the reduction of overfitting at cost of accuracy.

The improved ResNet model, also optimized with a dropout rate of 0.65, achieved a significantly higher test accuracy of 86.48%. Early stopping was triggered at epoch 17, with the training accuracy reaching 84.95% and validation accuracy at 78.59%, demonstrating better generalization compared to AlexNet. The accuracy is 3% worse than the fine-tuned ResNet without regularization, although a major improvement in overfitting reduction can be seen, as the Enhanced ResNet has a difference of roughly 6% between training and validation accuracy, compared to the prior with 10%.

The ROC curve in “Fig. 10” shows high AUC values across most classes, indicating strong discriminatory performance for most species. Classes 0, 2, 7, and 1 exhibit near-perfect AUC scores, while others, like class 8, show

slightly lower performance. However, due to the fairly balanced dataset, the ROC curve might not be the most meaningful evaluation metric, as it can overestimate performance by not reflecting class misclassifications as well. The Precision-Recall (PR) curve in "Fig.11", however, gives more insight into model performance, especially for classes with lower AUC scores, like class 8. The AP scores from the PR curve highlight more subtle variations in precision and recall for each class, offering a better reflection of the model's effectiveness, particularly for classes that are harder to distinguish.

Fig. 10 ROC-Curve of Enhanced ResNet on test set

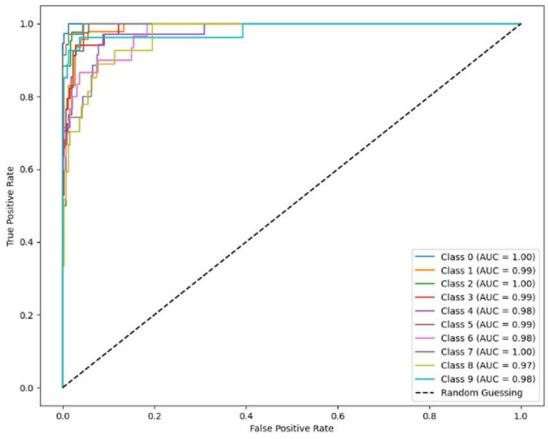
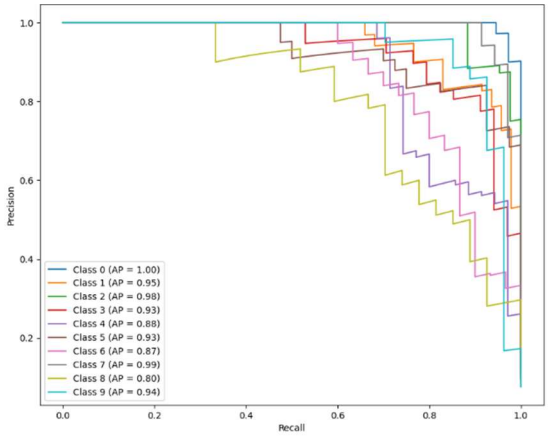


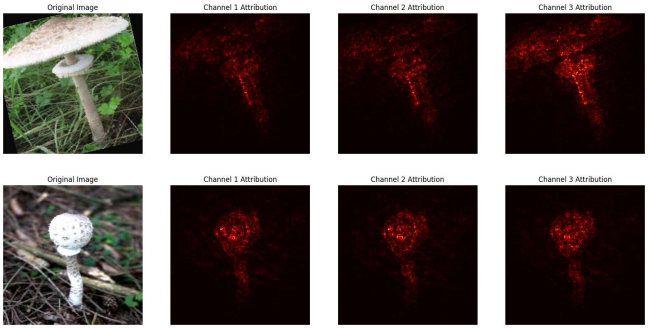
Fig. 11 Precison-Recall Curve of Enhanced ResNet on test set



D. Integrated Gradients

To understand the decision-making process of the models, we applied Integrated Gradients, a method that visualizes feature importance by attributing model predictions to input features. For this evaluation, we focused on the enhanced ResNet model and selected random images from the test set for which Integrated Gradients were applied to calculate pixel-wise attributions. These attributions were visualized as heatmaps in "Fig. 12", highlighting the most influential regions in the image. The heatmaps clearly showed that the enhanced ResNet model focused on key mushroom features, such as the cap, stem, and ring structures, with the attribution often corresponding to the silhouette of the mushroom. For example, in Amanita mushrooms, the model highlighted the characteristic ring and warts on the cap, confirming its reliance on biologically relevant features. The visuals offered insights into the model's decision-making process, confirming that predictions were driven by key morphological features of the mushroom, such as the cap and stem, rather than irrelevant factors like background noise or texture.

Fig. 12 Integrated Gradients on model Enhanced ResNet of *Procera* in adult stage with flat cap and present ring (top) and young stage with bulbous cap (bottom)



E. Concept Bottleneck Model

During the concept learning phase, the model achieved a training concept accuracy of 82.78%, with a corresponding validation accuracy of 77.18%, indicating a slight generalization gap after 15 epochs. The classification stage resulted in a training classification accuracy of 83.38%, while validation accuracy dropped to 71.27%. On the test set, concept accuracy dropped to 56.06%, while classification accuracy reached 72.96%, suggesting challenges in generalization and concept reliability, that might stem from the assumption of consistent concept values. Analysis of individual predictions revealed that incorrect concept estimations, such as cap shape, significantly influenced final classification outcomes. For example, a sample from the *Rufus* class was misclassified as *Amanita*, with incorrect concept predictions despite high confidence (89.71%).

CONCLUSION

As the results suggest that while removing layer freezing yields higher accuracy, it may not generalize well to unseen data due to the high variance in training performance. Given that all data originated from the same dataset—despite proper train-validation-test splits—the representativeness of the evaluation remains a concern. This highlights the need for further validation on diverse and independent datasets to better assess the model's robustness and real-world applicability. A key challenge is the availability of well-labeled mushroom datasets that can facilitate reliable test evaluations. Moreover, developing datasets with correctly labeled concepts is important for training CBMs without relying on predefined assumptions, allowing for a more data-driven approach to concept selection.

REFERENCES

[1] Desiree Himuskin, Dzvinka Yarish, dzvinn, Hannes Kuslap, and Rain Eich. Mushroom multiclass classification. https://kaggle.com/competitions/mushroom-multiclass-classification, 2024. Kaggle.

[2] M. Sundararajan, A. Taly, and Q. Yan, "Axiomatic Attribution for Deep Networks," arXiv, 2017. [Online]. Available: https://arxiv.org/abs/1703.01365.

[3] Prof. Dr. Jannik Strötgen, "2024-WS_AI_VL-04a-Transfer-Learning", HKA Vorlesung 2024-WS AI

[4] Prof. Dr. Jannik Strötgen, "Explainable AI: Concept-based Explanations II", HKA Vorlesung 2024-WS XAI

[5] The AI Summer Team, "The most popular CNN architectures explained," The AI Summer, 2023. [Online]. Available: https://theaisummer.com/cnn-architectures/. [Accessed: 24-Jan-2025].

[6] L. McInnes, J. Healy, N. Saul, and L. Großberger, "UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction," [Online]. Available: https://umap-learn.readthedocs.io/en/latest/basic_usage.html