Species Recognition from Pollen Images

Nguyen Ha Anh
Hanoi University of Science and Technology
Hanoi, Vietnam
anh.nh224298@sis.hust.edu.vn

Vu Thanh Hang
Hanoi University of Science and Technology
Hanoi, Vietnam
hang.vt224310@sis.hust.edu.vn

Abstract

Pollen classification is important for fields like ecology, agriculture, and medicine, helping with tasks such as environmental monitoring, crop management, and allergy prediction. This project focuses on a pollen classifier using DenseNet121, a type of transfer learning, to automatically identify pollen grains from microscopic images based on their shape. While the results are promising, there's still potential for improvement.

1. Introduction

Pollen is a substance produced by flowering plants, trees, grasses, and weeds to fertilize other plants of the same species. There are various types of pollen, each with distinct morphological features that can be identified under a microscope. Accurate identification of pollen species is crucial for verifying honey quality, reconstructing past vegetation to understand historical climate changes, biodiversity, and human impacts, as well as serving as a forensic tool.

Traditionally, pollen identification is carried out manually with a microscope, which is very time-consuming. This inspection involves the recognition of differences in shapes, texture, and other visual features from the pollen exine that can, sometimes, be very subtle and lead to classification errors by novice palynologists.

With the importance and issues mentioned above, this project aims to develop a robust system for the automated recognition of pollen species through the application of advanced image processing and deep learning techniques. We will address the use of transfer learning techniques to leverage pre-trained models, enhancing the accuracy and efficiency of the system in recognizing pollen species.

1.1. Desired Outcomes

Input: The input consists of microscopic images of pollen grains. For this image-based classification, each input is a high-resolution image which contains one or several pollen grains in an image. The given dataset is provided by the lecturer.

Output: The model should output the predicted class label for each pollen grain image or feature set, identifying the specific species of the pollen grain.

1.2. Challenges

- **High variability in appearance and inter-class similarity:** Pollen grains often have similar features in the same species or genera, and the appearance can vary due to factors like the angle of view, lighting conditions, and the presence of contamination or noise.
- **Limited labeled data:** Large, high-quality labeled datasets for pollen classification may be scarce, which can make training deep learning models challenging.
- **Preprocessing and noise reduction:** Microscopic images can contain noise, contamination, or artifacts, requiring effective image processing and preprocessing techniques to enhance classification accuracy.

2. Related Work

Before the widespread use of machine learning, pollen classification was performed manually by expert botanists and palynologists. Pollen grains are typically examined under a microscope, and key features such as shape, size, surface texture (e.g., spines, grooves), and the number of pores (apertures) are used to distinguish among variations of a species. However, this technique is time-consuming, and it requires high expertise, prone to human error and struggles to handle large datasets since workloads are high [5] . The performance of the feature extraction stage elevated with the rise of SEM (Scanning Electronic Microscope). There is an experiment conducted SEM for feature extraction pollen of Prunus species [3] but still heavily rely on manual labour for the act of samples' preparation.

Further research using CST algorithm, known as color-based technique, for feature extractions [1] .They using 4 Machine Learning techniques has been implemented and fine-tuned for this field of research: 2 variations of SVM (SMO, C-SVC), J48 and KNN and given result that SVM-based classifier outperform the two others.

In recent years, deep learning techniques have shown great promise in automating both feature extraction and classification from raw image data. These models are trained on large annotated datasets and can learn highly discriminative features for classifying pollen grains. But pollen grain labeled data is scarce, transfer learning has become a new promising approach [2]. CNN models final's layer are fully connected layers with softmax activation were replaced by a flattened layer to transform data. Still, Transfer learning requires a sufficient amount of annotated data and computational resources. In this report, we would like to use the DenseNet architecture to classify pollens from 22 different classes.

3. Proposed Method

In this course's final project, with limited experience and time, we addressed the challenge of identical pollen structure and a small dataset by using transfer learning with DenseNet, a pre-trained Convolutional Neural Network (CNN), combined with Data Augmentation to improve model robustness.

Transfer Learning is a technique where knowledge learned from one task is reused to boost performance on a related task.

DenseNet121 is a CNN architecture that enhances information and gradient flow between layers. Introduced by Gao Huang et al. [4], DenseNet is popular in image processing due to its efficient layer connections. DenseNet121, with 121 layers, was chosen for its high efficiency on small datasets and its ability to deliver high performance with limited resources.

Data Augmentation in image classification involves generating new training images by applying transformations like rotation, flipping, color adjustments, and noise injection. These techniques increase dataset diversity, improving the model's robustness and generalization, especially with limited data.

3.1. Data structure

Input images

DenseNet121 has some specific input requirements to ensure compatibility and efficiency: Input images must have a fixed size of 224×224 pixels, with 3 color channels (RGB). Pixel values in the image must be normalized to the range $[0, 1]$. Input images must be represented as tensors with the following shape: $(\text{batch\_size}, 224, 224, 3)$.

In order to meet these requirements, we first define a function to preprocess all the images in the dataset. After the preprocessing step, all images will have a shape of $(224, 224, 3)$. List $X = []$ was used to append processed images. Then, the list was converted into a numpy array: $X = \text{np.array}(X)$. After this step, X will have a shape of $(N, 224, 224, 3)$ with N being the number of images.

Input labels

Since this is a multi-class classification task, the labels also need to be one-hot encoded.

We use the list $Y = []$ to append all the class labels corresponding to each processed image. Then, the list was also converted into a numpy array: $Y = \text{np.array}(Y)$. After this step, Y will have a shape of $(N,)$, where N is the number of images, and each element in Y is a string.

To represent Y as a one-hot vector, the `LabelEncoder` was used to convert each string class label in Y to an integer: $Y_{\text{le}} = \text{le.fit\_transform}(Y)$. After that, the integer-encoded labels were converted into one-hot encoded vectors, where each vector has a length equal to M, which is the number of classes: $Y_{\text{cat}} = \text{to\_categorical}(Y_{\text{le}}, M)$. The final shape of Y_{cat} will be (N, M), with each row being a one-hot vector of length M.

The two arrays X and Y_{cat} were then split into training data, validation data, and test data with the ratio of $80/10/10$.

3.2. Model structure

The transfer learning technique uses DenseNet121 as the base model. DenseNet's key feature is that each layer connects to every other layer, which helps improve the flow of gradients and prevent the vanishing gradient problem. This is shown through its 4 Dense Blocks, where each block uses 3×3 convolution kernels, batch normalization, and ReLU. These blocks are connected, and between them, there are Transition Layers with 1×1 convolution and 2×2 average pooling to reduce the number of feature maps.

In our method, we use only the convolutional base for feature extraction. The model's weights were initialized with DenseNet121's pretrained weights and frozen during training. The output feature map was processed by GAP to form a 1D vector, which passed through fully connected layers. Dropout was applied to prevent overfitting. The final layer uses softmax to output a probability distribution for classification.

3.3. Algorithm

ImageDataGenerator was used for data augmentation, applying random transformations to improve robustness and reduce overfitting. The `data_generator` object handled the augmentation, with the `fit()` function used for feature-wise normalization.

Training Process: The model was compiled with the Adam optimizer, categorical cross-entropy loss, and evaluated using accuracy. It was trained with a batch size of 16 for up to 100 epochs, with EarlyStopping implemented to halt training if validation loss did not improve for 60 epochs.

Training data was fed through `data_generator.flow`, with performance validated on a separate set. Training metrics and loss values were displayed in the console (`verbose=1`) and stored in the `history` object for analysis.

4. Experiment

4.1. Dataset

Pollen_VNUA is a dataset created by Vietnam National University of Agriculture. This project will use only a subset of this dataset, containing 5322 images of 22 species.

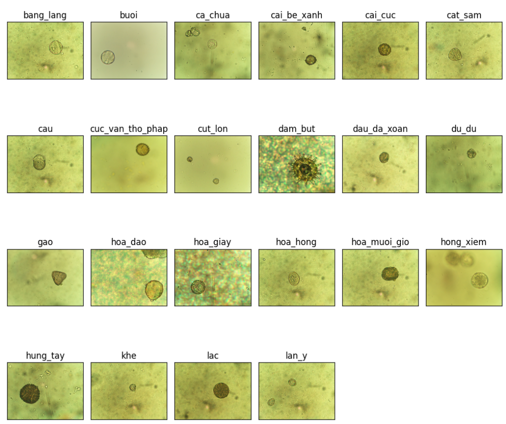

Figure 1. Example images of 22 species from the subset

After several steps of exploration, we created a plot showing the number of images for each species. We observed that the dataset is quite imbalanced, with the species *hoa giay* has the largest number of samples (593 images), while *cai be xanh* has the fewest (only 69 images). The remaining species have sample sizes ranging from 200 to 400 images.

4.2. Experiments and Results

We conducted three experiments to evaluate our model's performance under different conditions:

- **Input image size** 224×224**:** This experiment uses the standard input size for DenseNet121. The results will be used to evaluate the model's performance with data augmentation, analyze the relationship between the number of images and accuracy, and compare with existing works.

- **Input image size** 128×128**:** This experiment evaluates the model with smaller input sizes, intended for scenarios with limited resources, and will be compared against the standard input size.

- **Without data augmentation:** This experiment aims to determine the impact of data augmentation in solving problems with limited datasets.

Standard vs Reduced Input Size

For the standard input size (224×224), shown in Figure 2, the model achieved an accuracy of approximately 97% on the test set, indicating excellent performance. When using the reduced input size (128×128), shown in Figure 3, the model still maintained a strong performance with 93% accuracy on the test set, suggesting that the model can perform effectively even with reduced computational requirements.

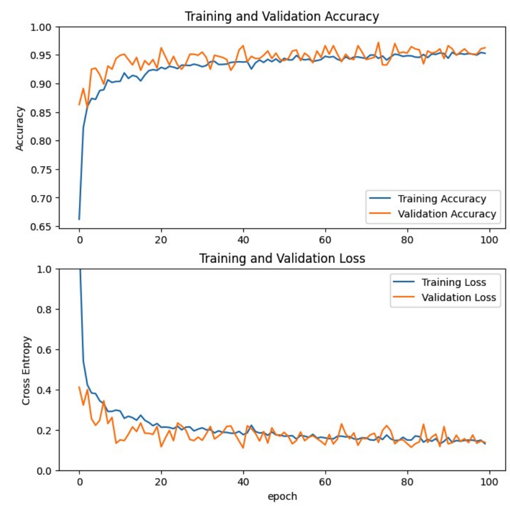

Figure 2. Training and Validation accuracy(above), Training and Validation loss(below) of input size 224×224

Figure 3. Training and Validation accuracy(above), Training and Validation loss(below) of input size 128×128

Both accuracy and loss graphs demonstrate steady improvement throughout training, with close alignment between training and validation metrics indicating good generalization. The progress decrease in loss values suggests effective feature learning by the model. Figure 4 presents a detailed comparison of accuracy across different pollen classes for both scenarios (input size(128×128) and (224×224)). Overall, the model input size of (224×224) outperformed the other configuration particularly on class 3, 5, 6, 15 and 18.

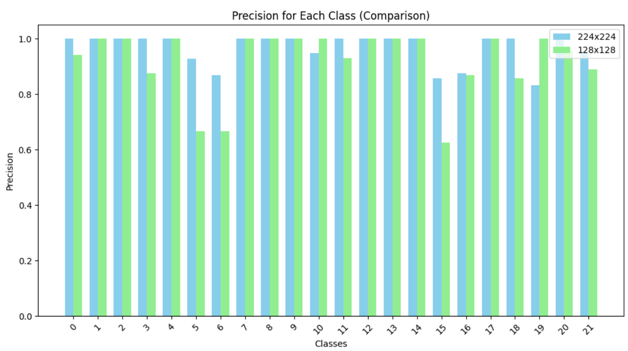

Figure 4. Comparison of accuracy for each pollen class in both case

Impact of Data Augmentation

The transfer learning approach with DenseNet121 yielded robust results in both augmented and non-augmented datasets. In Figure 5, the model performance on unseen data is slightly lower than model trained with augmentation, since validation accuracy is lower and validation loss is higher compared to Figure 2. Moreover, the experiment stated that, without data augmentation, there is a marginal decrease in the accuracy on the test set, from 97 % to approximate 96%. This result might indicate that data augmentation is being helpful in the learning process of model.

4.3. Discussion

With the above experimental results, it is evident that using the standard input size along with data augmentation yields the best performance. This case will be utilized to further evaluate the impact of imbalanced datasets on classification performance.

Recall Analysis

We use recall metric to measure how well the model identifies all the true instances of a specific class (Figure 6). Most classes exhibit a high recall (close to 1.0), suggesting that the model performs well in identifying instances of these classes. However, certain classes, such as `cai be xanh` and `dau da xoan`, have noticeably lower recall. In the case of `cai be xanh`, the lower recall could be

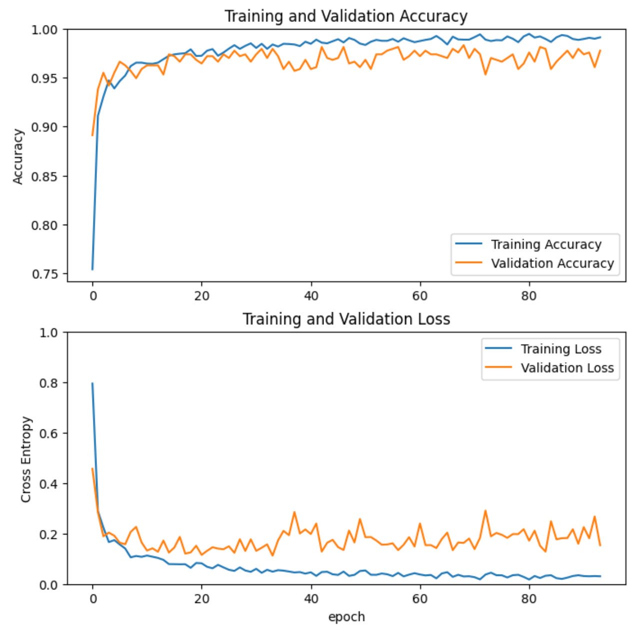

Figure 5. Training and Validation Accuracy/Loss for model without Data-augmentation

attributed to the lack of sufficient image data. In contrast, for `dau da xoan`, the lower recall may be caused by the class sharing visual features with other classes, leading to misclassification.

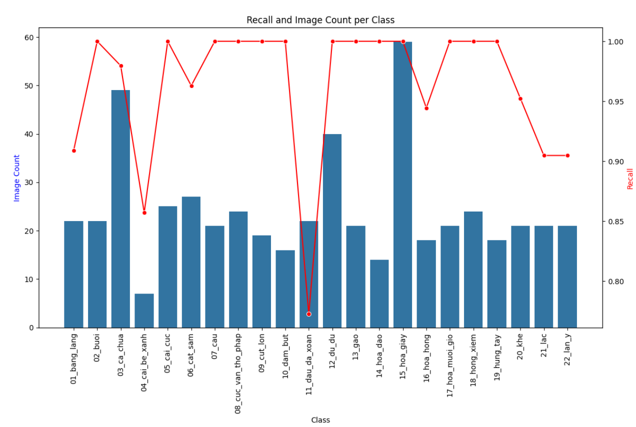

Figure 6. Correlation between the number of images and the respective recall for each class.

Misclassification Analysis

To further analyze this problem, we use misclassification matrix heatmap to detect the errors made by the model (Figure 7). The misclassification analysis shows two main patterns. The class `dau da xoan` is the most problematic, often being misclassified as `hoa giay` (twice) and also confused with `cau` and other classes. On the other hand, `cau` tends to receive misclassifications from five other classes (`cat sam`, `dau da xoan`, `khe`, `lac`, and `lan y`), but it isn't misclassified itself. This suggests that `cau` might have more generic features that the model uses when uncertain, while `dau da xoan` is harder to classify due to its variable appearance or complex features.

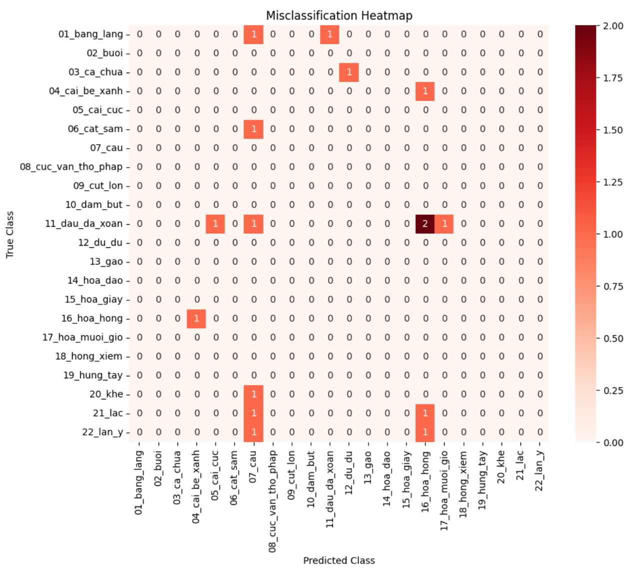

Figure 7. Misclassification

Comparative Analysis

To ensure objectivity, we evaluated our model against the models used in [2]. We selected top three models from the paper based on their high performance. The comparison results are presented in Table 1. Comparisons are made regarding batch size of 16.

Table 1. Comparison of Model Accuracy

Model	Accuracy (%)
ResNet50 [2]	94.257
InceptionV3 [2]	93.439
DenseNet201 [2]	97.217
Our Model	96.804

The comparison highlights the performance of various models, including our proposed model. While DenseNet201 achieves slightly higher accuracy (97.217%), our model follows closely with 96.804%, demonstrating competitive performance, outperforming ResNet50 and InceptionV3. These results suggest that our model can achieve state-of-the-art performance while maintaining efficiency, making it a strong alternative to more complex architectures like DenseNet201. Given that we also used DenseNet in our approach, its high performance further confirms its suitability for the pollen classification task.

5. Conclusion

5.1. Achievement

This project developed a deep learning system to classify pollen species from high-resolution microscopic images. By using transfer learning with DenseNet121, the model achieved a high accuracy , confirmed DenseNet's effectiveness for pollen recognition. However, recall and misclassification analyses highlighted the need for better data balancing and more detailed preprocessing to improve classification accuracy and handle similar classes more effectively.

5.2. Future Work

Future work should focus on improving preprocessing by better isolating pollen grains from backgrounds and handling noisy images. Refining segmentation techniques and experimenting with alternative architectures or hyperparameter tuning will enhance model performance. Additionally, addressing class imbalance through oversampling or synthetic data generation could further improve generalization and overall model robustness.

References

[1] Gonçalves AB, Souza JS, Silva GGd, Cereda MP, Pott A, Naka MH, and Pistori H. Feature extraction and machine learning for the classification of brazilian savannah pollen grains. *PLOS ONE*, 2016. doi:https://doi.org/10.1371/journal.pone.0157044. 1

[2] Benjamin Garga, Hamadjam Abboubakar, Rodrigue Saoungoumi Sourpele, David Libouga Li Gwet, and Laurent Bitjoka. Pollen grain classification using some convolutional neural network architectures. *J. Imaging*, 2024. doi:https://doi.org/10.3390/jimaging10070158. 2, 5

[3] Anna Geraci, Vincenza Polizzano, Pasquale Marino, and Rosario Schicchi. Investigation on the pollen morphology of traditional cultivars of prunus species in sicily. *Acta Societatis Botanicorum Poloniae*, 2012. doi:https://doi.org/10.5586/asbp.2012.017. 1

[4] Gao Huang, Zhuang Liu, and Laurens van der Maaten. Densely connected convolutional networks. *arXiv preprint arXiv:1608.06993*, 2018. 2

[5] France I., A.W.G. Duller, Duller G.A.T., and Lamb H.F. A new approach to automated pollen analysis. *Quaternary Science Reviews*, 2000. doi:https://doi.org/10.1016/s0277-3791(99)00021-9. 1