Agriculture Pest Classification using Convolutional Networks

Akash Ajay Nair
Arizona State University
anair66@asu.edu

Kavya Chandrika Vempalli
Arizona State University
kvempall@asu.edu

Shreyans Mehta
Arizona State University
smehta76@asu.edu

Abstract

The Agriculture Pest Classification project endeavors to harness the power of machine learning techniques to address the persistent challenge of pest invasions in agricultural settings. Pests, including insects, fungi, and pathogens, pose a substantial threat to crop health, often resulting in significant yield losses and financial strain for farmers. To combat this issue, our project focuses on developing a state-of-the-art predictive model utilizing cutting-edge algorithms such as convolutional neural networks (CNNs). These CNNs are adept at extracting intricate spatial features from input images, enabling the model to discern subtle patterns and associations indicative of specific pest species. By leveraging sophisticated deep learning architectures, our model aims to achieve high accuracy in pest classification, thereby empowering farmers with timely and precise pest identification capabilities. This predictive capability not only facilitates early detection of pest outbreaks but also enables targeted interventions, such as pest-specific pesticide application or crop management strategies. Ultimately, the integration of advanced machine learning technologies into agricultural practices holds immense promise for optimizing crop yields while minimizing the detrimental effects of pest infestations.

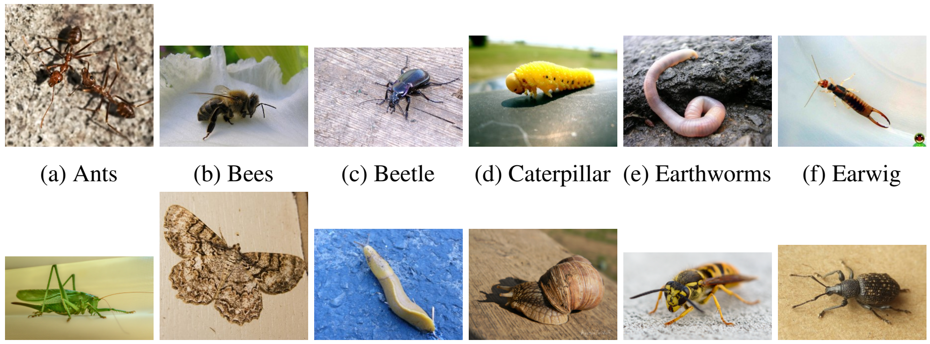
1 Execution Plan

The upcoming sections will provide a comprehensive overview of our execution strategy, including detailed discussions on the sources of data we plan to utilize, the specific tasks we aim to accomplish, potential challenges we anticipate encountering, and various other relevant aspects essential to the successful completion of our project.

1.1 Data Collection and Preprocessing

The primary data source for our project is Kaggle. We've searched for datasets relevant to our use case, focusing on those containing 12 classes of pests. Although we're not exclusively relying on Kaggle, we aim to delve into additional publicly accessible agricultural databases or potentially annotate our own data.

As a part of our data preprocessing procedures, we will ensure uniformity in image dimensions by resizing them to a consistent size, typically 224x224 pixels. This standardization facilitates smoother processing and analysis. Additionally, we will normalize pixel values to bring uniformity and stability to the dataset. Normalization aids in optimizing the performance of machine learning models by ensuring that each feature contributes equally to the learning process. Furthermore, we will employ various data augmentation techniques, including rotation and flipping, to augment the dataset and introduce variability. These techniques are crucial for enhancing the model's ability to generalize and perform effectively across different scenarios, ultimately improving its robustness and performance.



(a) Ants (b) Bees (c) Beetle (d) Caterpillar (e) Earthworms (f) Earwig

(g) Grasshop (h) Moth (i) Slug (j) Snail (k) Wasp (l) Weevil

Figure 1: Sample Images from Dataset

1.2 Training and Experiments

Our approach involves initially replicating the findings of existing research solutions. Regarding model selection and architecture, we aim to choose a suitable CNN architecture from options like ConvNext. ConvNext represents a new convolution-based architecture that not only outperforms Transformer-based models like Swin but also demonstrates scalability with the volume of data. In terms of training setup, we will partition the data into distinct training, validation, and test sets, while configuring essential hyperparameters such as learning rate and batch size. The training process will entail the training of the CNN model using the designated training data, with continuous monitoring of both loss and accuracy throughout the training phase. Additionally, we plan to conduct experiments involving the exploration of various values for the hyperparameters. We will also investigate the potential of transfer learning from relevant domains, such as plant disease detection, to further enhance model performance and adaptability.

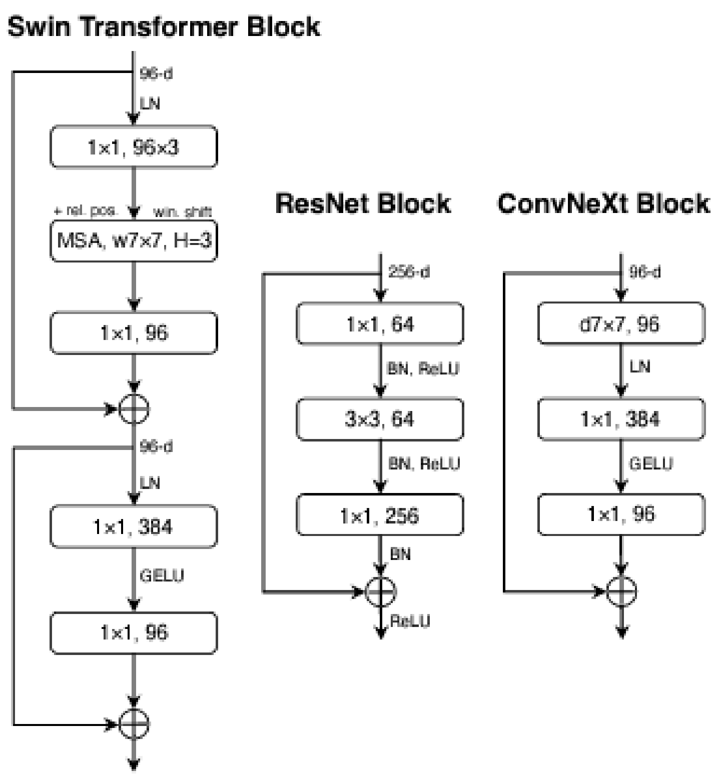


Figure 2: Block designs for a ResNet, a Swin Transformer, and a ConvNeXt.

1.3 Anticipated Challenges and Strategies for Mitigation

One significant challenge we foresee is the limited availability of labeled data for training, which may hinder the development of robust models. To address this, we plan to implement various mitigation strategies. Firstly, we will employ data augmentation techniques such as rotation, flipping, and zooming to artificially expand the dataset size, thereby enhancing model generalization. Additionally, we intend to leverage transfer learning by fine-tuning pre-trained models from related domains like ImageNet on our pest dataset. Moreover, we will actively engage domain experts in the labeling process to augment the dataset further.

Another challenge we anticipate is overfitting, particularly given the complexity of convolutional neural networks (CNNs) and the scarcity of data. To combat this issue, we will implement several mitigation approaches. This includes incorporating regularization techniques such as dropout layers, L2 weight regularization, and early stopping during training to prevent overfitting. Furthermore, we will utilize k-fold cross-validation to evaluate model performance and detect overfitting early on. We also plan to monitor validation loss closely and terminate training when validation loss begins to increase, preventing the model from overfitting to the training data excessively.

Moreover, we recognize the potential issue of imbalanced class distributions within our dataset, where certain pest classes may have significantly fewer samples than others. To mitigate this, we will adopt various strategies such as class weighting, where higher weights are assigned to underrepresented classes during training. Additionally, we will generate synthetic samples for minority classes through data augmentation techniques. Furthermore, we will explore oversampling and undersampling methods to balance the class distribution in the training set, ensuring each class receives adequate representation.

Lastly, we are mindful of the computational demands associated with training deep CNNs, which could pose a challenge given our hardware limitations. To address this, we plan to employ several mitigation tactics. This includes utilizing cloud services such as AWS or Google Cloud to access GPU/TPU resources, which can significantly accelerate training. Additionally, we will consider model pruning techniques to reduce unnecessary layers or overall model complexity, thereby decreasing computational requirements. Furthermore, we will adjust the batch size based on available memory to optimize training efficiency and resource utilization. These strategies collectively aim to address potential challenges and ensure the successful development of our pest classification model.

1.4 Workload Distribution

The table presented below outlines the meticulously planned allocation of tasks among team members. Our overarching objective is to foster an environment of equitable participation, ensuring that each member contributes substantively across diverse responsibilities essential for the success of our project. These responsibilities encompass the gathering of pertinent research papers, acquiring sample codes for reference and experimentation, and crafting the comprehensive project report. By methodically distributing tasks and responsibilities, we endeavor to optimize efficiency and promote collaboration, thereby facilitating the seamless execution of our project objectives.

Name	Responsibility
Kavya	Data collection and preprocessing
Akash	Model selection and training
Shreyans	Experimentation and fine-tuning

Table 1: Responsibilities Distribution

We have devised a structured plan to effectively realize our project objectives within a span of four weeks. This approach is aimed at ensuring that our efforts are organized and progress steadily towards achieving our goals. Each week of this timeline is meticulously allocated to specific tasks and activities, allowing us to maintain focus and manage our time efficiently. By adhering to this well-defined schedule, we aim to maximize productivity and optimize the outcome of our project. The detailed breakdown of tasks for each week is elaborated upon in the table presented below, providing clarity and guidance for our project execution.

Week	Tasks
Week 1	Acquire research papers and codebases, and replicate their findings.
Week 2	Execute individual tasks.
Week 3	Consolidate outcomes from all tasks and formulate the final results.
Week 4	Address any pending tasks, finalize implementation, and document the entire project.

Table 2: Project Timeline

2 Evaluation Plan

In our project, we are dedicated to employing various performance metrics to comprehensively assess the effectiveness of our model. These metrics serve as quantitative indicators, allowing us to evaluate different aspects of our model's performance. Among the metrics we intend to utilize are accuracy, which measures the overall correctness of our model's predictions; precision, which assesses the proportion of true positive predictions out of all positive predictions made by the model; recall, which measures the proportion of true positive predictions out of all actual positive instances in the dataset; F1 score, which provides a balance between precision and recall; and AUC-ROC curves, which illustrate the trade-off between true positive rate and false positive rate across different threshold values.

While we are actively considering the best approach for assessing individual member performance, we are yet to finalize the specific metrics or criteria for this purpose. However, it is essential for us to ensure fairness and transparency in evaluating the contributions of each team member. To achieve this, we are committed to documenting the individual tasks undertaken by each member and the impact of their contributions on the project's progress and outcomes. This documentation will provide a clear record of each member's involvement and achievements throughout the project lifecycle.

In addition to documenting individual contributions, we are contemplating the inclusion of a peer review section in each milestone submission. This peer review mechanism would provide an opportunity for team members to offer feedback and evaluations on each other's work. By incorporating peer reviews, we aim to foster a collaborative and constructive environment where team members can openly discuss and provide insights into the quality of work completed by their peers. This feedback loop will not only contribute to improving the overall quality of our project but also enhance the professional development and accountability of team members.

References

[1] A ConvNet for the 2020s, Zhuang Liu1,2* Hanzi Mao1 Chao-Yuan Wu1 Christoph Feichtenhofer1 Trevor Darrell2 Saining Xie1† 1Facebook AI Research (FAIR) 2UC Berkeley

[2] https://medium.com/atakanerdogan305/convnext-next-generation-of-convolutional-networks-325607a08c46

[3] https://keras.io/api/applications/convnext/

[4] https://www.kungfu.ai/blog-post/convnext-a-transformer-inspired-cnn-architecture

[5] https://towardsdatascience.com/implementing-convnext-in-pytorch-7e37a67abba6

[6] Swin Transformer: Hierarchical Vision Transformer using Shifted Windows , Ze Liu†* Yutong Lin†* Yue Cao* Han Hu*‡ Yixuan Wei† Zheng Zhang Stephen Lin Baining Guo Microsoft Research Asia