The Berry Project

Andrew Downing
downina3@wwu.edu

Jane Elbegzaya
elbegzj@wwu.edu

Abstract

We worked with the Sitka Sound Science Center to support their effort to understand how climate change is shifting berry phenology in Southeast Alaska. Our project focused on salmonberry images from the 2023 harvest season, beginning with around 12,478 labeled photos of 4 locations and filtering them down to 10,141 usable training images. We built a multi-label classification system to automatically identify key phenological stages such as breaking leaf buds, open flowers, and fruit ripening. Most of our experiments used ResNet50 [4] and Vision Transformer (ViT) [2] models, with ConvNeXt Base [10] added later as a more modern architecture. Our results show that machine learning can reliably assist in automating phenology annotation, helping SSSC scale their monitoring efforts and better track climate-driven changes in local berry development.

1 Introduction

Berry plants matter in Southeast Alaska for food, wildlife, and local foraging. Recently, changes in weather and temperature have made the timing of berry growth less predictable. For example, leaf growth, open flowers, fruit ripening, and leaf fall may happen earlier or later than usual. Tracking these changes helps researchers understand how climate change is affecting local ecosystems.

The Sitka Sound Science Center (SSSC) runs a long-term monitoring project to study berry growth patterns. They collect images from multiple field sites using time-lapse cameras, along with other environmental data. Right now, a major challenge is that the images still need to be labeled by hand to mark which growth stages appear in each photo. This takes a lot of time, and it becomes harder as the dataset grows each season.

In this project, we worked with SSSC to build a machine learning system that can predict salmonberry growth stages from images. We started with about 12,478 labeled photos from four sites from the 2023 season and filtered them down to 10,141 usable training images. Since more than one stage can appear in one image, we treated the task as multi-label classification. We trained and compared several model types, including ResNet50 [4], Vision Transformer (ViT) [2], Mixture of Experts (MoE) [11], and later ConvNeXt Base [10]. Our goal is to reduce manual labeling work and help SSSC process new image data faster in future seasons.

2 Related Work

Plant phenology research increasingly relies upon computer vision and deep learning to automate laborious tasks such as manually annotating thousands of individual images. For species such as salmonberries, automated computer vision tools enables researchers to process large datasets more efficiently and consistently.

A challenge in many outdoor phenology studies is the variability in image quality. Trail camera images often suffer from weather interference, sun glare, background noise, and the potential to have the camera disturbed later during the data collection process. Correia et al. [8] demonstrated that machine learning models could automatically filter out low quality images based on color and texture features, ensuring that only high quality and usable images went into the training set.

Following image filtering, segmentation and detection models are frequently employed to isolate plant structures. Prior work has used a variety of preprocessing techniques like image correction, cropping, and pixelwise classification using CNN-based frameworks. [9, 6, 7, 1]. CNN architectures have been particularly influential, providing strong performance on semantic segmentation in phenological imaging, as summarized by Jian and Li [5].

Deep learning models have demonstrated strong accuracy in fruit and flower detection in phenotyping. Grimm et al. [3] achieved high performance using VGG-based CNNs for grapevine phenotyping, suggesting that similar model architectures may generalize to other berry species with developmental stages similar to grapevines. While their data setup was strongly controlled with segmentation of berry plants directly from any background, it's approach suggested that these computer vision models may have the capability of locating berries and flowers in a sea of noise with a degree of accuracy.

Recent advances in computer vision architectures have further influenced image based phenological analysis. Convolutional neural network backbones such as ResNet [4] remain a solid option because of their strong representational capacity and stability across diverse visual tasks. More recent CNN architectures, including ConvNeXt [10], modernize convolutional neural nets by incorporating architectural principles inspired by transformer models while preserving convolutional inductive biases.

Transformer based approaches such as Vision Transformers (ViT) [2] have demonstrated strong performance in image classification tasks, making them suitable for classifying phenological stages. Additionally, Mixture of Experts (MoE) models [11] enable specialization across multiple expert subnetworks, which may be advantageous when phenological stages in the data present large class imbalances.

3 Methods and Approach

All code utilized was done with Python 3.13 using PyTorch. Our approach centers on constructing a modular machine learning pipeline capable of identifying multiple salmonberry phenology stages within a single image. The complete system includes dataset preparation, image preprocessing and filtering, multi-label image classification, model architecture definition, and the choice of loss functions and evaluation metrics. All training and inference procedures are implemented through a unified pipeline that ensures consistency across architectures and data configurations.

3.1 Dataset and Label Preparation

Phenology annotations provided by the Sitka Sound Science Center consisted of time charts that indicated when each phenology stage occurred at each monitoring site. These timelines were converted into image-level CSV files through a separate preprocessing step. Each CSV entry represented one image and included a binary one-hot encoded vector marking the presence or absence of nine salmonberry stages. As multiple stages may be visible simultaneously (for example, increasing leaf size and open flowers), this representation naturally supports multi-label classification.

After image filtering (detailed below), the final dataset used by the training pipeline contained 10,141 high-quality labeled images. These CSVs were read by our data loading module, which paired each image with its corresponding multi-label target vector.

3.2 Image Preprocessing

Field images collected from time-lapse game cameras required considerable preprocessing due to variability in weather, automatic focus adjustment, overexposure from flash or sunlight, or from bugs within the camera hardware itself. To build a model robust to these environmental factors, we constructed a multistep preprocessing process that standardized each of the images and filtered them to ensure that the training process was as clean and consistent throughout.

Rotation and Cropping Due to a camera bug, some images were incorrectly rotated. We corrected their orientation, so all images were consistent. We also cropped out the black metadata bar added by the camera, since it does not contain useful visual information for identifying phenology stages.

Two-Stage Image Quality Filtering A portion of the raw field were not suitable for phenology annotation. Common issues included blurry, grainy, dark, or overexposed images due to lighting or weather conditions. To filter these systematically, we implemented a two stage process:

1. **Laplacian Image Filter** First, we pass the images through a Laplacian image filter, which operates by analyzing the RGB values of the individual pixels and those that surround them to see how quickly their values change. The quicker a color value changes,

the higher the score that the algorithm assigns, whereas slower changes around object edges is seen as a lower quality image. This in turn works well as an initial filter to condense the images into good, usable training images or into poor quality images. The Laplacian image filter does not fare well for nighttime images, which places most of their respective images into the poor quality image.

2. **ChatGPT API Filter** Secondly, we take the batch of poor quality images identified by the Laplacian filter as they need further analysis. We use ChatGPT's API to prompt it to determine whether the image was either daytime or nighttime, and if the image was high or low quality.

Together, these two filtering steps filtered out poor quality images and ensured that the remaining dataset contained clear and high quality training images.

Resolution Scaling The original camera resolution is space inefficient so we resized them to several different resolutions while maintaining the original aspect ratio. We ultimately chose to resize them down to 1024x576 for training purposes, as it provided a good balance between image clarity and space efficiency.

3.3 Streamlit Web App

For our trained models and to provide users with a clean user interface, we utilized a free cloud service provided by Streamlit to host our models and to allow users to engage with to analyze images and predictions. Our trained models are staged on HuggingFace and are downloaded when the app provisions a virtual machine when a new user visits the Streamlit app.

3.4 Model Architectures

Our system supports several machine learning architectures, each adapted for multi-label output by replacing the final classification layer with a sigmoid activation. The primary models used in this project were:

- **ResNet50**: a convolutional network with skip connections, effective for capturing detailed textures.
- **Vision Transformer (ViT)**: a transformer-based model that divides images into patches and uses self-attention to capture global relationships within the plant's structure.
- **ConvNeXt Base**: a more recent CNN architecture inspired by transformer design principles, added later in development to test its suitability for image analysis.
- **Mixture of Experts (MoE)**: a series of specialized models trained on individual models, effective in identifying stages affected by class imbalance.

Architectures are instantiated through a unified model-loading interface that ensures consistency across training runs.

3.5 Training Configuration Components

The core components of our training configuration include:

- the AdamW optimizer with cosine learning rate scheduling and warmup,
- a batch size of 42,
- fixed 120 epoch counts for comparability across runs,
- sigmoid-activated outputs for multi-label classification.

3.6 Loss Functions

We evaluated two loss functions due to the natural imbalance in phenology stages, where flowering and fruiting occur less frequently than leaf growth:

- **Binary Cross-Entropy (BCE)**: suitable for multi-label prediction but sensitive to imbalance.
- **Focal Loss**: balances underrepresented phenological stages to perform better during training.

3.7 Evaluation Metrics

To provide a complete description of model performance in later sections, we define the metrics here:

- **Subset accuracy**: percentage of samples where all predicted labels match the ground truth.

- **Micro F1**: overall F1 score weighted by class frequency.
- **Macro F1**: average F1 score across classes, highlighting rare stages.
- **Hamming accuracy**: fraction of correctly predicted labels across all classes.

These metrics capture both per-class and per-image performance, which is essential for interpretation.

4 Experiments

Our experiments were designed to compare model performance across different training splits, loss functions, and architectures using the unified training pipeline described earlier. All experiments were run exclusively on images resized to 1024×576 resolution, as mentioned in the data preprocessing stage. This section describes the experimental setups without repeating methodological details already covered in the Approach section. Since the 95/5 splits use most of the data for training, the evaluation results mainly show how well the models perform on data that looks very similar to what they were trained on. These numbers should be viewed as optimistic performance estimates rather than true generalization.

4.1 Data Split Configurations

We conducted several experiments to evaluate both within-site and cross-site generalization:

Per-site 95/5 Split For each of the four salmonberry locations, we trained a model on 95% of that site's labeled images and validated on the remaining 5%. This setup measures how well a model adapts to the specific lighting conditions, camera angles, and plant appearances of a single site.

All-sites 95/5 Split All locations were merged into a single dataset, with 5% held out for validation. This exposes models to greater visual diversity during training and serves as our main "overall dataset" benchmark.

N-1 Cross-Site Split To test generalization across locations, we trained on three sites and validated on the held-out fourth site. This setup approximates how the system might behave on a new SSSC site with no labeled training data.

4.2 Model Variants Compared

Each of the model architectures implemented in our pipeline (ResNet50, ViT, ConvNeXt Base, SimpleCNN, and the Mixture of Experts (MoE) model) was trained separately under the same experimental settings. The goal was to compare model behavior rather than to tune architectures.

4.3 Training Setup

All training runs used the following consistent configuration:

- images resized to 1024×537 resolution,
- fixed number of training epochs,
- AdamW optimizer,
- cosine learning rate scheduling with warmup,
- batch size of 42,
- identical augmentation and preprocessing transforms,
- sigmoid outputs for multi-label prediction.

This uniform setup allows direct comparison between model types and data splits.

4.4 Loss Function Comparison

To evaluate the effect of class imbalance, each model was trained twice—once with Binary Cross-Entropy (BCE) and once with Focal Loss. This comparison helps determine whether emphasizing rarer classes improves recognition of stages such as open flowers and ripe fruits.

4.5 Threshold Tuning

After training, we optionally applied per-class threshold tuning by sweeping through candidate values using validation logits. This step allowed us to adjust decision boundaries for each phenology stage independently, improving multi-label performance without modifying the model.

4.6 No-Ground-Truth (Inference-Only) Experiment

In addition to experiments using labeled data, we also ran an inference only experiment on unlabeled images from all sites. This "no ground truth" setup used the best-performing trained model to generate predictions for new seasonal images. Although no quantitative metrics could be computed, this experiment allowed us to:

- check how well the model generalizes to unseen data,
- verify that predictions aligned with expected seasonal patterns,
- generate qualitative galleries for us to review.

This step mirrors a real deployment scenario where new images are used without annotations.

4.7 Experiment Outputs

For every experiment, our pipeline produced:

- model checkpoints,
- validation metrics (subset accuracy, micro/macro F1, Hamming accuracy),
- raw logits for threshold tuning,
- JSONL prediction files,
- optional gallery-style visualizations of predicted labels.

These outputs form the basis of the quantitative and qualitative analysis presented in the Results section.

5 Results

This section summarizes the performance of the models across the 95/5 all-sites split, 95/5 per-site training, and the N-1 generalization experiments.

5.1 Overall Model Comparison (All-Sites 95/5 Split)

Table 1 shows the performance of all architectures under the all-sites 95/5 split. ConvNeXt Base achieved the strongest results across nearly all metrics, followed by ResNet50 with focal loss. ViT performed moderately well, while BCE generally underperformed focal loss for all models. SimpleCNN served as a baseline.

Model	Subset Acc.	Micro F1	Macro F1	Hamming Acc.
ResNet50 (Focal)	**0.9345**	**0.9663**	0.9232	**0.9910**
ViT (Focal)	0.8560	0.9151	0.8241	0.9767
ViT (BCE)	0.7173	0.7864	0.7178	0.9529
ResNet50 (BCE)	0.8613	0.9065	0.7712	0.9804
ConvNeXt Base (Focal)	**0.9607**	**0.9803**	**0.9583**	**0.9948**
SimpleCNN (Focal)	0.7853	0.8889	0.7304	0.9700

Table 1: Performance of all models on the all-sites 95/5 split.

These results come from a 95/5 split where only a small portion of the data is held out for evaluation, and all images come from the same sites used during training. Because of this, the reported metrics reflect strong in-distribution performance and likely overestimate how well the models would perform on completely new sites. ConvNeXt Base achieved the best overall performance across all metrics, showing that it was the most reliable model for this task. ResNet50 with focal loss also performed well and remained stable across different phenological stages. Using focal loss consistently improved results compared to BCE, especially for less frequent stages such as open flowers and ripe fruits.

These confusion matrices (Figure 1) show how each model tends to mix up classes on the all-sites 95/5 split. Even though this is a multi-label task, the plots still help us see which stages are predicted the most consistently and which ones get confused, especially for similar-looking stages like increasing leaf size, open flowers, and colored leaves.

5.2 Per-Site 95/5 Split Results

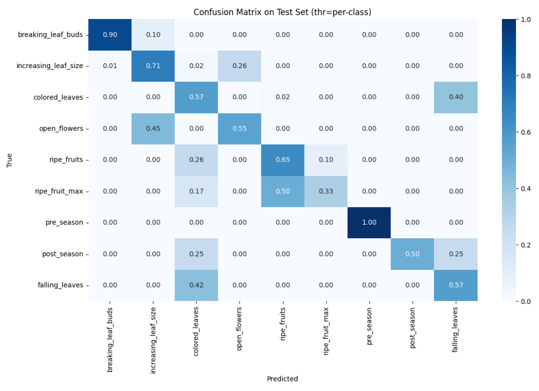
To evaluate within-site performance, we trained and validated a ResNet50 model with focal loss using a 95/5 split for each individual site. This setup measures how well the model adapts to site-specific conditions such as lighting, camera angle, and plant appearance.

Table 2 summarizes the results across the four monitored locations. While overall performance remains strong, metrics vary by site due to differences in data volume and class distribution.

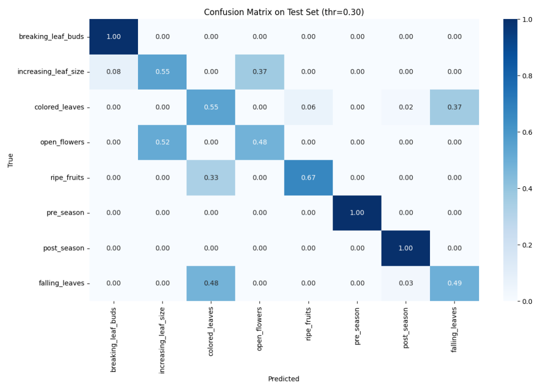
Site	Subset Acc.	Micro F1	Macro F1	Hamming Acc.
Charles	0.9259	0.9720	0.8222	0.9918
FOB	0.7778	0.9072	0.4638	0.9697
Katlian	0.8333	0.9524	0.7143	0.9860
Lance	0.8846	0.9600	0.7800	0.9904

Table 2: Per-site 95/5 validation results using ResNet50 with focal loss. Metrics vary by site due to differences in data volume and class diversity.

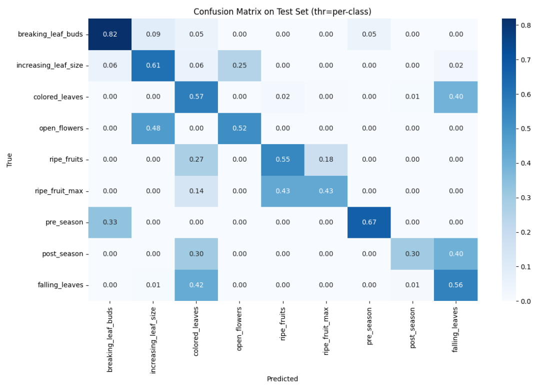
The per-site confusion matrices (Figure 2) show that results change from site to site. Some sites have cleaner diagonals, while others have more mix-ups. This likely comes from site differences like lighting, camera angle, and how often each class appears.



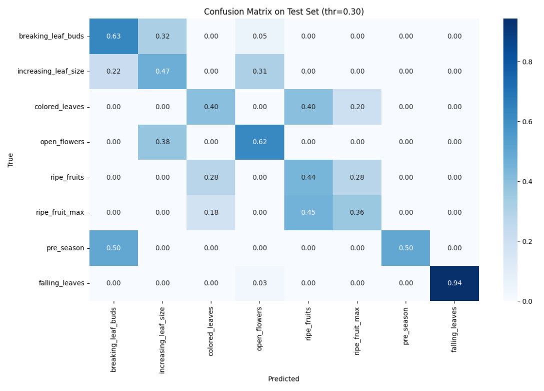
(a) ResNet50 (Focal)



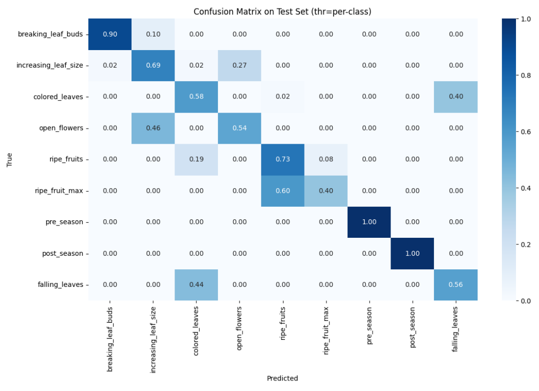
(a) Charles



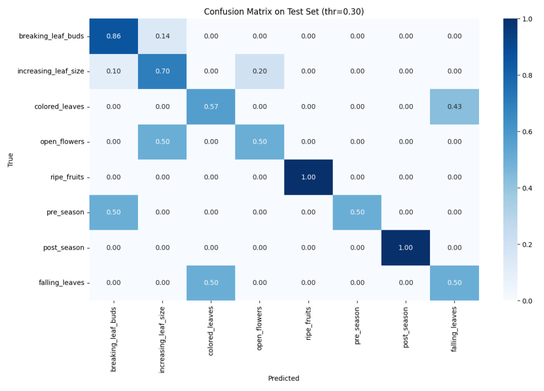
(b) Vision Transformer (Focal)



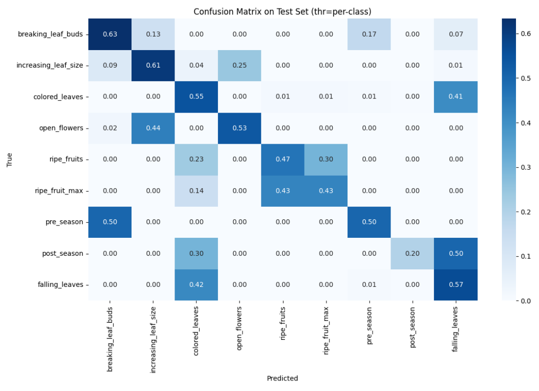
(b) FOB



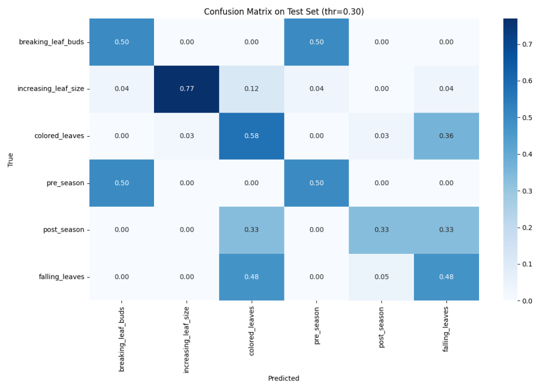
(c) ConvNeXt Base (Focal)



(c) Katlian



(d) SimpleCNN (Focal)



(d) Lance

Figure 1: All-sites 95/5 confusion matrices using per-class thresholds. These plots provide a qualitative view of per-class prediction behavior for each architecture in a multi-label setting.

Figure 2: Per-site 95/5 confusion matrices for ResNet50 with focal loss. As this is a multi-label task, these matrices are intended as qualitative diagnostics.

5.3 N-1 Cross-Site Generalization Results

For the N-1 experiments, models were trained on three sites using an 80/20 train–validation split and evaluated on a fourth held-out site with available ground-truth labels. Although labels exist for evaluation, the held-out site is never seen during training; this lets us test how well the model works on a new site that it never saw during training.

Train Sites	**Test Site**	**Subset Acc.**	**Micro F1**	**Macro F1**	**Hamming Acc.**
FOB, Katlian, Lance	Charles	0.6111	0.8423	0.4012	0.9445
Charles, Katlian, Lance	FOB	0.5556	0.8117	0.3568	0.9312
Charles, FOB, Lance	Katlian	0.5833	0.8289	0.3894	0.9386
Charles, FOB, Katlian	Lance	0.6019	0.8354	0.4127	0.9421

Table 3: N-1 cross-site results using ResNet50 with focal loss. Models are trained on three sites and evaluated on a fourth site.

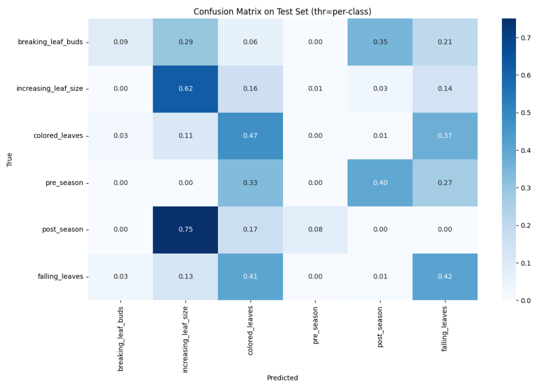


Figure 3: N-1 cross-site confusion matrix for Lance using ResNet50 with focal loss and per-class thresholds. The model was trained on the remaining three sites (80/20 split) and evaluated on Lance, which was excluded from training but retains ground-truth labels.

We only include one example confusion matrix (Figure 3) for the N-1 setting, instead of showing all four. In the N-1 experiments, the classes are not evenly present at every site because different sites reach stages at different times. So in some held-out sites, a few classes barely appear (or do not appear at all), which makes the confusion matrix look almost empty.

Since this is a multi-label task, these confusion matrices can be hard to interpret when classes are missing. Showing four nearly empty matrices would take space without adding much information. Instead, we show one representative example and use the table metrics to compare all N-1 runs.

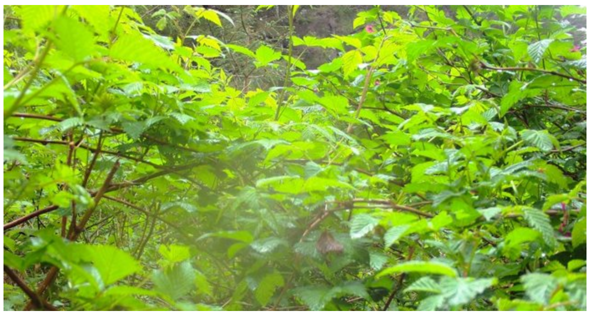


Figure 4: Challenging example: The model correctly predicts increasing leaf size but does not detect open flowers. In practice, increasing leaf size and open flowers often co-occur temporally and visually, making this overlap difficult to distinguish without explicit flower visibility.

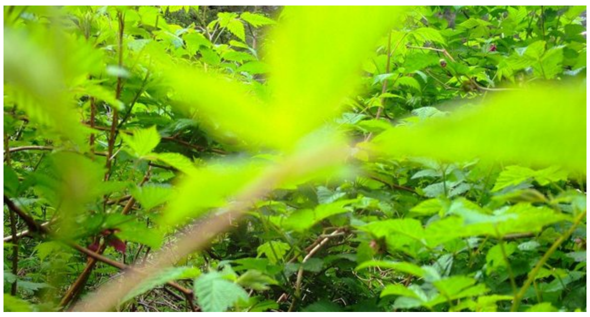


Figure 5: Challenging example: The image contains open flowers, but the model predicts ripe fruits. Closed or partially occluded flowers may visually resemble early drupelets, leading to confusion between these stages.

5.4 No-Ground-Truth Qualitative Evaluation

To evaluate model behavior in a realistic deployment setting, we conducted a no-ground-truth inference experiment using the ConvNeXt Base model on images from the Galankin site. This site was not included in any training or validation split, and no human annotations were available for quantitative evaluation. As a result, performance was assessed qualitatively by visually inspecting model predictions and comparing them to expected seasonal patterns. Figures below show representative examples illustrating both successful predictions and common failure modes.

Overall, these examples suggest that ConvNeXt can still work on a new unseen site, especially for clear stages like ripe fruits and late-season leaf changes. Most mistakes happen when stages overlap or are hard to see in the image, not because the model is completely failing.

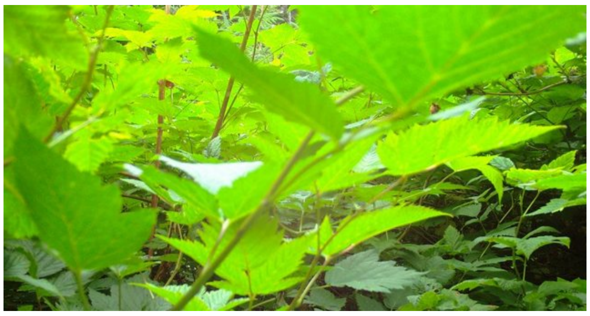


Figure 6: Successful example: The model correctly identifies ripe fruits. Berries are visually distinctive and consistently detected across sites.

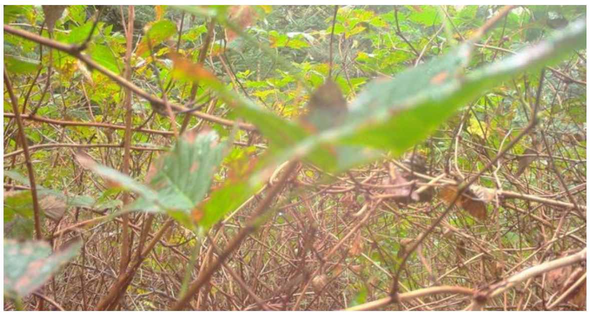


Figure 7: Successful example: The model predicts colored leaves and falling leaves. These late-season stages are strongly represented in the training data and exhibit clear visual cues.

5.5 Results Summary

Although the 95/5 split experiments show high performance across several models, these results should be interpreted with caution. Because only 5% of the data is used for evaluation and comes from the same sites as the training data, the metrics do not fully represent how the models would perform in new or unseen environments.

It does reflect that performance varied across the per-site 95/5 experiments, since lighting, camera placement, and class balance differed at each location. As expected, results were worse in the N-1 cross-site experiments, since here the model was being evaluated on a site it had not seen during training. This reflects the hard generalization problem to new locations.

The no-ground-truth experiment on the unseen Galankin site showed that the model is still capable of producing reasonable predictions, especially for visually clear stages like ripe fruits and late-season leaf changes. Most errors occurred between stages that naturally overlapped in time or appearance. Overall, the system can perform adequately on unseen sites but still leaves much room for improvement.

6 Project Reflection

Our project met the planned objectives of training and deploying a few machine learning models for the purpose of analyzing and predicting phenological stages for a given salmonberry image. We utilized services that allowed us to provide a clean, easy, and concise user interface to interact with when analyzing photos. However, we did not deliver a solution that allowed our clients a fully automated pipeline to annotate, analyze, and predict all images within a specified dataset for a harvest year as technical and practical limitations prevented a complete, reliable, and dependable tool from being developed. Andrew worked on most parts of the data preprocessing steps, specifically the cropping, filtering, resizing, and cleaning of all images within the dataset. Andrew also worked on developing the user interface with Streamlit, integrating the trained models and data preprocessing steps into the web application. Jane handled most of the model training and modified the pipeline to integrate within the current project workflow. Jane also worked on converting the manually annotated data, fixing data issues regarding the original annotation process, into usable training .CSV files. Both Andrew and Jane contributed equally to the presentation, the final report, along with any and all project deliverables. Throughout the course of the project, there were several major challenges faced as a group. Firstly, the data we received was arguably uncleaned and convoluted state. It took a few months to converge on a solution for data transfer, and we were faced with several challenges relating to the data itself. With twelve thousand images roughly totaling around 100gb in size, we needed to find a suitable solution for training on the WWU CS machines without exceeding local storage account limits. We resolved this issue by first downloading the images onto the temp drives where local storage restrictions were not in place to then filter and resize on a location-by-location basis. Then, we resized the images as stated in the data preprocessing stage to vastly reduce the file sizes of the total dataset. We noticed issues in the initial batch of labeled data requiring additional manual analysis to ensure that the proper labels were assigned with the conversion script. We both faced challenges regarding terminology and techniques used within the space of machine learning, including the differences of multi-class and multi-label classification and the approaches of quantitative examples of visualizing the successes and pitfalls of training. Through engagement with the faculty advisor for this project, we were able to learn and solidify our understanding of specific concepts relating to the training and evaluation aspect for machine learning and deep learning and overcome some of the challenges presented by our initial unfamiliarity with the field.

7 Conclusion and Future Work

Across our experiments, modern convolutional architectures outperformed transformer-based models in both overall accuracy and robustness to class imbalance, but in practice, both model architectures had their own strengths. These results demonstrate that architectural choices play a crucial role when working with outdoor time lapsed images.

Currently, our final product consists of a Streamlit website application built to interact with the trained models while implementing our data preprocessing steps. Through this interface, users can upload upwards of 50 images at a time, choose the preprocessing steps they'd like to apply, and observe how each chosen model predicts.

Evaluation across per-site, all-site, and cross-site experiments shows us that our current proposed system of training and evaluation reliably identifies visually prominent features for identifying phenological stages.

Overall, our work demonstrates that deep learning can be used as an effective tool for supporting large scale phenological research monitoring. By utilizing data preprocessing, varied model architectures, and a user interface to interact with, our Berry Project provides a practical foundation to support the Sitka Sound Science Center, along with the United States Forest Service in assisting with their climate research by reducing the manual annotation effort and enabling an automated analysis tool for phenological analysis of berry plants.

Regarding future work, several extensions could substantially improve the proposed system. In this, training was limited to the 2023 harvest year with a restricted number of site images. That year contained considerable variability and class imbalance, particularly across locations where some plants did not produce fruit. Expanding training to include additional harvest years and sites would increase the data diversity and variability, and is expected to improve the model generalization and robustness. Specifically, including sites with varying background conditions could help the model better distinguish phenological features from background noise. With the app, it would be nice to have it run locally so it isn't reliant on cloud infrastructure, an external service partner, and on internet data transfer speeds. With newer, more advanced models, it could become more autonomous, but in its current state, it requires additional human evaluation to ensure its prediction is accurate.

References

[1] S. Bargoti and J. P. Underwood. 2017. Image segmentation for fruit detection and yield estimation in apple orchards. *Journal of Field Robotics*, 34, 6, 1039–1060. doi:10.1002/rob.21699.

[2] Alexey Dosovitskiy et al. 2021. An image is worth 16x16 words: transformers for image recognition at scale. *International Conference on Learning Representations (ICLR)*.

[3] Johannes Grimm, Katja Herzog, Florian Rist, Andreas Kicherer, and Volker Steinhage. 2018. An adaptive approach for automated grapevine phenotyping using vgg-based convolutional neural networks. In *Proceedings of the International Conference on Computer Vision Systems*.

[4] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 770–778.

[5] Y. Jiang and C. Li. 2020. Convolutional neural networks for image-based high-throughput plant phenotyping: a review. *Plant Phenomics*, 2020, 1–22. doi:10.34133/2020/4152816.

[6] N. Katal, M. Rzanny, P. Mäder, and J. Wäldchen. 2022. Deep learning in plant phenological research: a systematic literature review. *Frontiers in Plant Science*, 13. doi:10.3389/fpls.2022.805738.

[7] U. Lee, S. Chang, G. A. Putra, H. Kim, and D. H. Kim. 2018. An automated, high-throughput plant phenotyping system using machine learning-based plant segmentation and image analysis. *PLOS ONE*, 13, 4, e0196615. doi:10.1371/journal.pone.0196615.

[8] Z. Li, R. Guo, M. Li, Y. Chen, and G. Li. 2020. A review of computer vision technologies for plant phenotyping. *Computers and Electronics in Agriculture*, 176, 105672. doi:10.1016/j.compag.2020.105672.

[9] Z. Li, R. Guo, M. Li, Y. Chen, and G. Li. 2020. A review of computer vision technologies for plant phenotyping. *Computers and Electronics in Agriculture*, 176, 105672. doi:10.1016/j.compag.2020.105672.

[10] Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. 2022. A convnet for the 2020s. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 11976–11986.

[11] Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, and Geoffrey Hinton. 2017. Outrageously large neural networks: the sparsely-gated mixture-of-experts layer. In *International Conference on Learning Representations (ICLR)*.