Automated Counting of Influenza Virus Plaques Using a Two-Stage Deep Learning Pipeline

Julien Erbland Max Henrotin Francesca Fino

December 2025, EPFL Lausanne

1 Introduction

Manual plaque counting is a fundamental but time-consuming task in virology laboratories, including the Laboratory of Environmental Virology (LEV). It is prone to human variability, fatigue, and inconsistency, especially for dense wells, and becomes a bottleneck as experimental throughput increases. To address this issue, the LEV proposed the development of an automated pipeline for plaque detection and quantification across multiple influenza strains.

2 Data and Preprocessing

2.1 Cropping the wells from raw images

The dataset provided by the LEV consists of images captured under a controlled illumination setup, each containing a 12-well plate. We decided to train our model using the individual wells rather than full-plate images. Splitting each plate into separate wells removes irrelevant background and standardizes the input provided to the model.

To standardize the input images and isolate each well for analysis, we implemented a two-step automated cropping pipeline using *OpenCV*, a Python library for image processing and computer vision.

First, we detect and crop the 12-well plate from the raw image by combining several detectors: ring-template matching, HSV color segmentation, Hough-circle detection, and an edge-based fallback. The system selects the first detector that confidently identifies at least 10 wells. If no detector is sufficiently reliable, the user is asked to manually crop the plate. This situation occurs very rarely when the imaging setup is consistent.

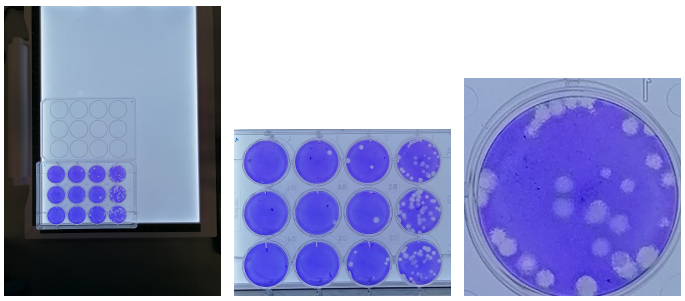
In the second stage, we apply a Hough-circle procedure to the cropped plate to locate exactly 12 wells, group them into rows and columns, and extract square crops centered on each well.

Figure 1 illustrates this pipeline.

2.2 Analysis of dataset distribution

The LEV lab also provided several Excel files containing tables with the plaque counts for each well. Using these files, we created a unified dataset that associates each well image with its corresponding plaque count, making it easier to use during model training and evaluation. Based on this curated dataset, we then performed data analysis to examine the distribution of the labels.

Despite the LEV providing all the images available, the dataset we worked with was limited in size and not uniformly distributed. It consisted of 792 images of individual wells. Approximately 22% of the wells showed plaque overgrowth, where individual plaques could not be distinguished. We excluded these wells from the dataset, following the same procedure applied by researchers when counting by hand. Among the remaining countable wells, 61% corresponded to sterile samples with no detectable plaques, while only 39% contained a measurable number of plaques. This imbalance, combined with the relatively small size of the dataset, made the raw data insufficient for training a robust model and therefore required the use of data augmentation.



(a) Raw image (b) Cropped plate (c) Cropped well (A1, top-right well)

Figure 1: Three-step cropping pipeline. From left to right: raw input image, detected and cropped 12-well plate, and final cropped individual well used as model input.

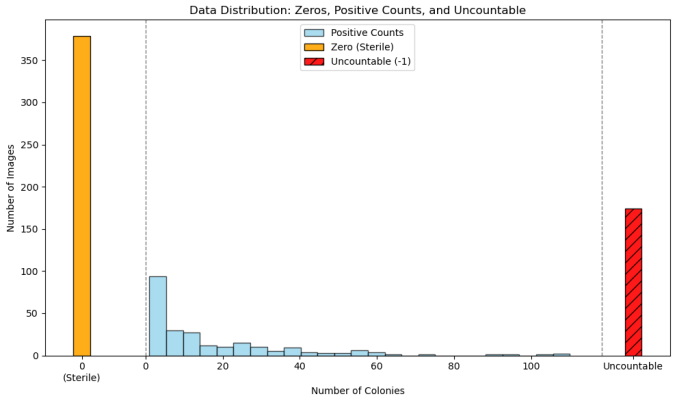


Figure 2: Distribution of plaque counts per well before data augmentation. The dataset is highly imbalanced, with a majority of sterile wells and a limited number of wells containing higher plaque counts.

2.3 Data augmentation

To increase the size of the dataset and reduce class imbalance, we generated new training samples from the existing data using the *Albumentations* library. For each original image, we produced up to four additional variants by randomly combining operations that preserve the appearance and count of circular plaques: horizontal flips, vertical flips, and 90-degree rotations. We performed augmentation in a stratified way over predefined plaque-count bins, so that additional samples were created preferentially to underrepresented ranges. Wells with zero plaques or uncountable plaques were excluded from augmentation, since their classes were already sufficiently represented.

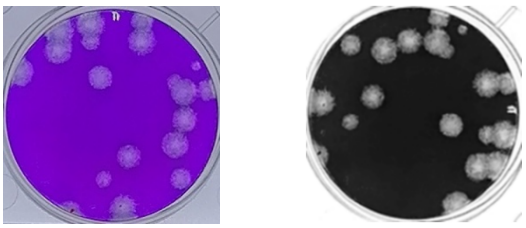
In addition, we designed dedicated image preprocessing and transformation pipelines using the `torchvision.transforms` framework. The goal of these transformations is to highlight the most relevant visual information in the plates. The key transformations include:

Grayscaling: To reduce color dependency and focus the model on colony shapes.

Contrast Augmentation: To emphasize the distinction between colonies and the background.

Gaussian Blur: To reduce high-frequency noise, allowing the model to focus on meaningful spatial patterns rather than pixel-level artifacts.

Furthermore for the training set, but not for the test set, we introduced controlled randomness to the intensity of some transformations. This improves data augmentation diversity and helps the model generalize to unseen images.



(a) Original (b) Transformed

Figure 3: Same well before and after processing

2.4 Final Dataset

After applying data augmentation, our dataset increased to 1,748 images and the proportion of sterile wells was reduced from 61% to 22%. This redistribution resulted in a more balanced dataset, as shown in Figure 4. Using this data in the training of our model, we improved its ability to generalize across varying plaque counts.

3 Methodology

The dataset was partitioned into training, validation, and test sets using a stratified 70/15/15 split. Data augmentation was applied to the training and validation sets to deal with class imbalance and limited data size.

The core of the tool is a two-stage Convolutional Neural Network (CNN) pipeline aimed at reliably identifying countable wells and accurately estimating their plaque counts. In the first stage of the pipeline, we trained a classification model using a pretrained EfficientNet-B0 backbone to discriminate between countable and uncountable wells. In the second stage, a regression model built upon a pretrained ResNet34 backbone was used to estimate plaque counts for wells identified as countable.

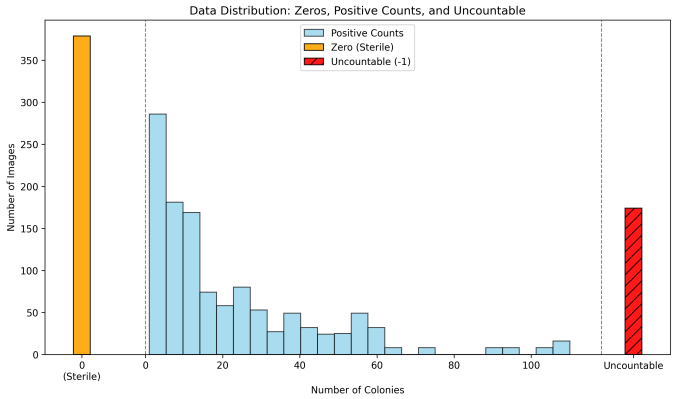


Figure 4: Distribution of plaque counts per well after data augmentation. Augmentation reduces class imbalance and increases representation across the full range of plaque counts.

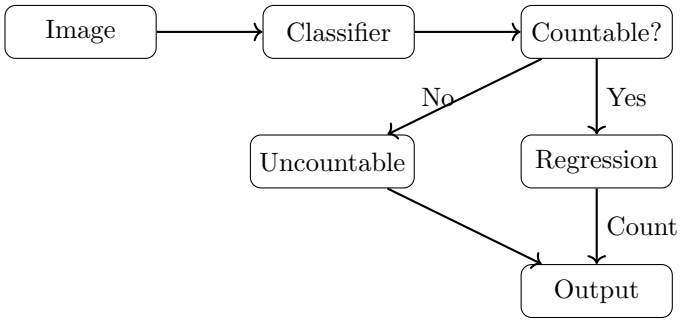


Figure 5: Two-stage plaque counting pipeline. Wells are first classified as countable or uncountable; regression is applied only to countable wells.

3.1 Model for classifying wells

The first stage of the pipeline determines if a well can be counted or not. This task is formulated as a binary classification problem, where each well image is labeled as either *countable* or *uncountable*. Wells are considered uncountable when plaques are too dense with too much overlap to be able to distinguish colonies.

We use an *EfficientNet-B0* architecture [1] for this task designed to provide strong classification performance with a limited number of parameters. It reduces the risk of overfitting with our small dataset by not having too much parameters.

The network is initialized with ImageNet pretrained weights using the PyTorch implementation [3]. The original classification layer is replaced with a custom binary classification head composed of fully connected layers, a ReLU activation, and dropout. The model outputs two logits corresponding to the *countable* and *uncountable* classes and is trained using a cross-entropy loss.

This classification stage acts as a filtering step, ensuring that only wells for which plaque counting is visually

meaningful are passed to the regression model.

3.2 Model for counting wells

For wells classified as countable, plaque counts are estimated using a regression model. It is based on a ResNet-34 architecture [2]. ResNet models use residual connections that allow each block to learn a correction to its input rather than a full transformation. Formally, the output of a block is given by

$$y = x + F(x).$$

where $F(x)$ is the residual function learned by the convolutional layers, which models how the features should be adjusted rather than recomputed and allows to catch more small visual differences.

The network is initialized with ImageNet pretrained weights using the PyTorch implementation [4]. The final classification layer is removed, and a custom regression head composed of fully connected layers, a ReLU activation, and dropout is added.

The model is trained using a Smooth L1 loss, defined as

$$\mathcal{L}_{\text{SL1}}(y, \hat{y}) = \begin{cases} \frac{1}{2\beta}(y - \hat{y})^2, & \text{if } |y - \hat{y}| < \beta, \\ |y - \hat{y}| - \frac{1}{2}\beta, & \text{otherwise,} \end{cases}$$

which strongly penalizes small errors while limiting the influence of outliers caused by noisy or inconsistent manual counts.

Optimization is carried out using the AdamW optimizer, which combines adaptive gradient updates with decoupled weight decay and has been shown to yield stable convergence in deep networks while providing effective regularization. A StepLR learning-rate scheduler with a step size of 10 is employed, reducing the learning rate after every 10 epochs.

We used model checkpointing based on validation performance. By computing the loss on a held-out validation set, the model weights corresponding to the epoch with the lowest validation loss were saved.

4 Models performances

All reported performance metrics are computed on a held-out test set that was not seen during training.

4.1 Classification performance analysis

The performance of the countability classifier was evaluated using a confusion matrix and standard classification metrics for classifiers.

Table 1: Performance metrics and error analysis of the countability classifier.

Metric	**Value**
Accuracy	0.986
Precision	1.000
Recall	0.984
F1-score	0.992
True Positives (TP)	380
False Positives (FP)	0
False Negatives (FN)	6
True Negatives (TN)	44
Mean plaque count of FN wells	25.33

The precision of the classifier reaches 1.000, indicating that no uncountable well was incorrectly classified as countable. This is important because a false positive would propagate uncountable samples to the regression model and lead to terrible plaque count predictions. On the other hand, having some FN has less effect since, as we can see with the mean count of these plaques, it misclassified the wells that were still truly countable but at the limit of being uncountable.

Overall, the classifier prefers rejecting ambiguous wells rather than risking incorrect regression.

4.2 Comparison of regression models

For the counter, we tested three different regression architectures. All models were trained and evaluated using the same dataset splits, preprocessing steps, and evaluation metrics to allow a fair comparison.

First, because of its adaptability and high performance on our classifier, we adapted an *EfficientNet-B0* model by replacing its classification layer with a regression head that outputs a single value. However, we observed a strong bias where the model could never predict the low plaque counts.

Second, we tried a *ResNet-34* architecture because of its residual connections, it can learn small visual differences between wells. This is important for plaque counting, where small changes in the image can lead to different plaque counts.

Finally, we implemented a custom *ColonyCNN* model inspired by the architecture proposed in [5]. This model is specifically designed for biological images and uses a simpler convolutional structure focused on detecting circular patterns. However, this model was unable to predict high colony counts.

We compared the results of the 3 models and we decided to only keep Resnet34 since it was the best performing one by far as shown on table 2.

Table 2: Performance comparison of the three tested regression architectures on the test set.

Model	MAE ↓	R^2 ↑	Accuracy$_{(10\%+1)}$ ↑
EfficientNet-B0	18.16	0.019	6.8%
ResNet-34	**1.37**	**0.983**	**86.8%**
ColonyCNN	11.09	0.152	23.4%

4.3 Counter Model: K-fold cross-validation

To optimize the ResNet-34 regression model, we performed 5-fold cross-validation using Smooth L1 loss as the metric. We conducted a grid search over three key hyperparameters: learning rate, weight decay, and the loss parameter β. All runs utilized the AdamW optimizer, a StepLR scheduler, and a batch size of 32.

Table 3 summarizes the mean and standard deviation of the best validation loss for all eight configurations. The lowest mean loss was achieved with a learning rate of 1×10^{-4}, zero weight decay, and $\beta = 1.0$. Consequently, these values were selected for the final model training. Exploration of the hyperparameter possibilities (such as alternative optimizers, batch sizes, or scheduler types) was limited by computational constraints even when training on NVIDIA T4 GPU provided by the Google Colab platform.

Table 3: 5-fold cross-validation results for Resnet34.

LR	WD	β	Mean loss	Std. loss
$1e^{-4}$	0	0.5	1.7108	0.0968
$\mathbf{1e^{-4}}$	**0**	**1.0**	**1.5065**	**0.1539**
$1e^{-4}$	$1e^{-4}$	0.5	1.6333	0.0876
$1e^{-4}$	$1e^{-4}$	1.0	1.5945	0.2194
$5e^{-4}$	0	0.5	1.8347	0.0535
$5e^{-4}$	0	1.0	1.6952	0.1744
$5e^{-4}$	$1e^{-4}$	0.5	1.8438	0.1093
$5e^{-4}$	$1e^{-4}$	1.0	1.6172	0.1083

4.4 Results for the regression model

Table 4 reports the regression performance across different influenza strains over the test set. With these results we could conclude with confidence that the primary factors influencing model accuracy are image quality and the visual clarity of colony development, as demonstrated by the PR8v images, which exhibited variability in both respects.

Table 4: Per-plate regression performance using the 10%+1 tolerance criterion.

Plate	N	MAE	R^2	Bias	$\%_{10\%+1}$	MedAE	%0
IBV	33	1.33	0.998	-0.30	93.9	1.0	9.1
Panama	64	0.39	0.994	-0.08	100.0	0.0	32.8
Brisbane	71	1.01	0.994	0.45	95.8	1.0	12.7
PR8	86	0.79	0.992	0.47	94.2	0.0	50.0
PR8v	66	1.97	0.831	-0.27	80.3	1.0	3.0

Overall, we achieved a R^2 score of 0.983 with Mean Absolute Error of 1.37 and an accuracy of 86.8% within a margin of 10% + 1 colony. Those results are illustrated by the prediction distribution in Figure 6.

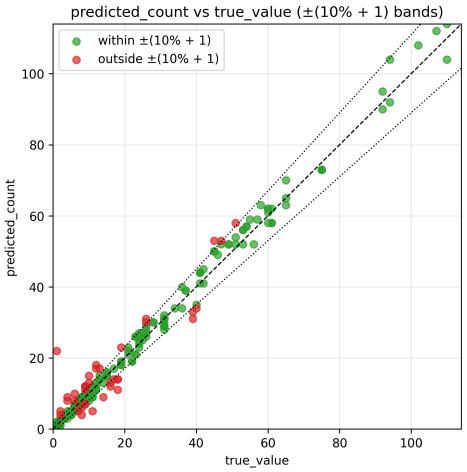


Figure 6: Predicted plaque counts for countable wells on the augmented data. Predictions within the tolerance criterion are shown in green.

5 Conclusion

The results are highly satisfactory given the challenges of the task. Biological assays are subject to natural variability, imaging noise, human bias and some inconsistence in the data. Despite these, the designed pipeline achieves consistent estimations across multiple influenza strains.

Robustness was externally validated using 120 independent images of a new plaque type with unseen strain growth to simulate real-world situation. The classifier made only four misclassifications, maintaining high performance in a new visual context. Regression accuracy is slightly lower, which is expected due to morphological differences from the five strains used during training.

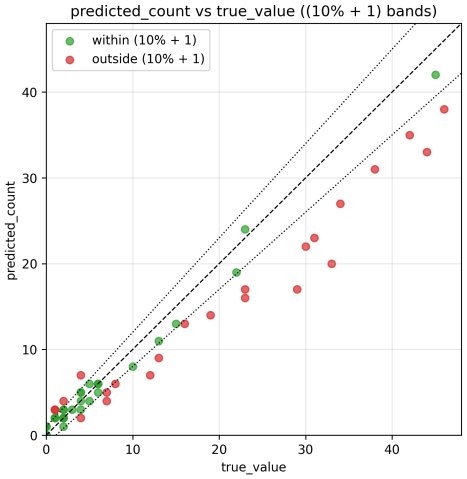


Figure 7: Predicted versus ground truth plaque counts for a new plaque type not seen during training.

These results suggest that the proposed pipeline is robust to variations in plaque and experimental setup, making it a tool for real laboratory conditions. Future work could focus on extending the training dataset, as well as further refining the regression model to improve performance on highly dense or visually complex wells.

References

[1] M. Tan and Q. Le, *EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks*, https://arxiv.org/abs/1905.11946

[2] K. He, X. Zhang, S. Ren, and J. Sun, *Deep Residual Learning for Image Recognition*, https://arxiv.org/abs/1512.03385

[3] PyTorch Contributors, *EfficientNet : PyTorch Documentation*, https://docs.pytorch.org/vision/main/models/efficientnet.html,

[4] PyTorch Contributors, *ResNet : PyTorch Documentation*, https://pytorch.org/hub/pytorch_vision_resnet/,

[5] Alessandro Ferrari, Stefano Lombardi, and Alberto Signoroni. *Bacterial colony counting with Convolutional Neural Networks in Digital Microbiology Imaging*. Pattern Recognition, 61:629–640, 2017. https://api.semanticscholar.org/CorpusID:35885342