Machine Learning Project 2 - Droplet Counter

Lovro Nuić, Vishal Pani
CS-433, Machine Learning, Fall 2021.
École Polytechnique Fédérale de Lausanne, Lausanne, Switzerland

***Abstract*—The results of several biological experiments are determined by counting the number of cells or particles in a given image. From cell concentration experiments to counting bacteria colonies in aqueous droplets in an emulsion determine: the need for determining the exact number of a particular kind cell or particle remains crucial. In this paper, we develop a pipeline that is able to detect aqueous droplets in an emulsion and classify whether it contains any bacteria or not. We use learning-based approaches for both our detector and classifier and compared them against traditional algorithms. Our final detector is able to get an IoU of 0.835 and our final classifier as able to get accuracy, and F1 score up to 96%. For both detection and classification our learning-based models outperforms the traditional methods and are also robust to the variety of input images.**

I. Introduction

In this report, we present our method to count the frequency of droplets (having a diameter of about 5-100 μm) containing bacteria in an emulsion. A section of a sample image is displayed in Fig. 1.

To achieve our objective we developed a two-stage pipeline. The first stage deals with the detection of circles in the image. The number of circles gives us the number of aqueous droplets in the image. The second stage classifies if a given droplet contains bacteria cells or not.

In Sec. II we elaborate on the different approaches we implemented for our circle detector. In Sec. III we describe our baseline model which uses traditional computer vision techniques to detect and classify the aqueous droplets. In this section, we also show the disadvantages of using such a pipeline. The necessity of using a learning-based approach to solve this task is presented in Sec. IV. In Sec. V we present qualitative and quantitative comparisons between the approaches we used for detection and classification. Finally in Sec VI, we conclude by underscoring the future work required to improve our solution.

II. Detection

The detection of circles in the image is a crucial step in our pipeline. To tackle this, we explore two different options. We begin by using (i) Hough Gradient Method, then move on to using a (ii) learning-based detector, YOLOv5 [1]. Both of these methods are described in the following sub-sections:

A. Hough Gradient Method

Circle detection can be done by following an approach similar to Hough Transformation for line detection [2]. Instead of a 2D accumulator, as used in line detection, a 3D accumulator with circles parameterized by the coordinates of the center and the radius (x_{cen}, y_{cen}, and r) can be used. This is however inefficient, so we use the gradient information of the edges to drive the process. In our implementation, we use OpenCV's [3] HoughCircle method to detect the circles.

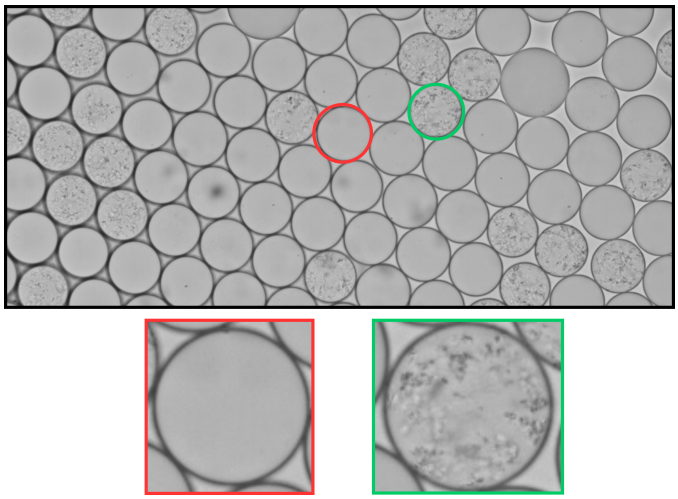

Fig. 1: **Sample Image, Top:** A section of a sample image. **Bottom Left:** A sample empty droplet, marked with a red circle in the top image. **Bottom Right:** A sample droplet with bacteria, marked with a green circle in the top image.

While this method works well for most of the images in our dataset, it is not able to detect some circles when the image contains a large variance in the radii of the droplets. The left image in Fig. 2 is an example of such a case. To overcome this we use YOLOv5, a learning-based detector, that is more robust to variance in radii and lighting condition of the input image.

B. YOLOv5

YOLOv5 is a state-of-the-art one-stage detector that has one of the highest accuracy to speed ratios [1]. Due to this, we use it as our circle detector.

The YOLOv5 repository provides several pre-trained model. Since we require fast inference, we use a smaller version of YOLOv5, named YOLOv5s that has 7.2 million parameters and requires about 16.5 BFLOPs per instance of the model (when the input image is of size 640x640 pixels). This is small when compared to YOLOv5x which has 86.7 million parameters and requires about 205.7 BFLOPs for an input of same size.

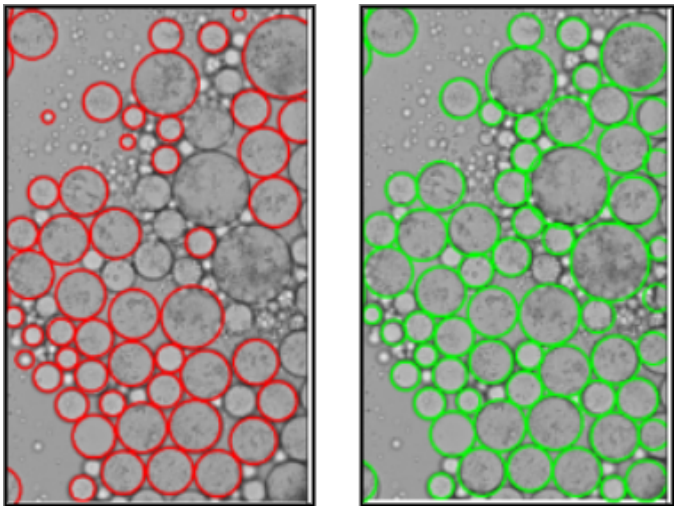

Fig. 2: **Comparison of Detections, Left:** Circle detection by Hough Gradient Method. **Right:** Circle detection by YOLOv5

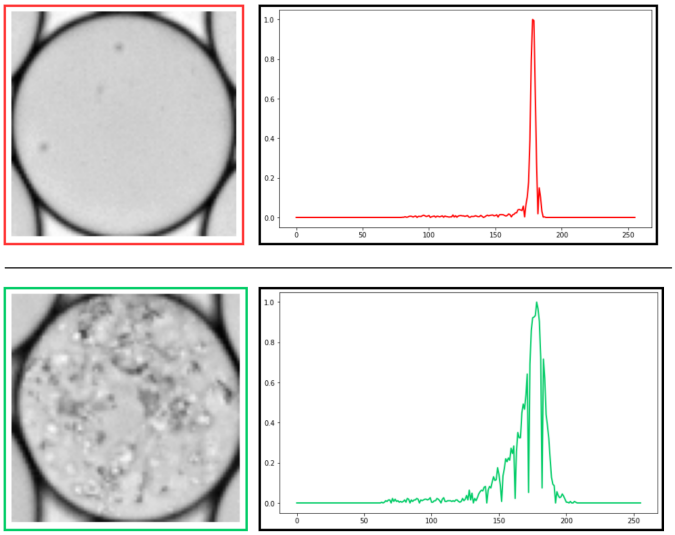

Fig. 3: **Image histogram of Droplets, Top:** Normalized histogram of empty droplet. **Bottom:** Normalized histogram of droplet with bacteria.

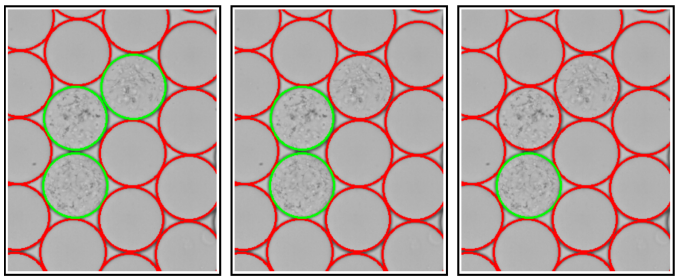

Fig. 4: **Effect of μ on classification by the baseline classifier:** Droplets in green are classified as having bacteria. As we increase the threshold by a small amount, we observe that misclassifications increase.

The dataset contains 67 images out of which we use 39 images for training and 29 images for validation. The dataset does not contain any coordinate annotations of the circles, so we use the output from the Hough Gradient Method to annotate our images and use it as the target while training the YOLOv5s model. We use 640x640 pixel images while training. Due to the few data samples, the training procedure also makes use of augmentations like flipping, color-space, and Mosaic augmentation [4]. Finally, we train the model for 200 epochs and save the best weights.

Since the output of the model are the top-left and bottom-right corners of a rectangle, we generate the center of the circle from the mean of the coordinates and diameter as the mean of the length of the sides of the rectangle.

In Fig. 2, we observe that the YOLOv5 model is successfully able to pick up the circles missed by Hough Gradient Method.

III. Histogram Classifier

We observe that droplets containing bacteria have a significant noise-like pattern, whereas, empty droplet are uniform. Performing histogram analysis on an empty droplet yields a histogram which has a narrow peak (Top plot of Fig. 3). Contrary to this, a droplet with bacteria yields a histogram that is more spread out due to the noise-like pattern (Bottom plot of Fig. 3).

Using the above observation, we design a baseline classifier that does not require a learning-based model. First, we extract the droplets using the Hough Gradient Method (as mentioned in Sec. II-A). Then for each droplet, we mask the background by using the radius of the circle and perform histogram analysis on it using OpenCV. The resultant image histograms are not normalized, so we perform min-max normalization on each of them. We then discern the spread of the normalized histogram by finding the area under the curve.

An empty droplet, having a sharp spike, will have a smaller area under the curve as compared to a droplet containing the bacteria. So to classify the droplets, we use a manual threshold μ, above which a droplet is determined to have bacteria.

While the proposed algorithm is fast and does not require any training data, it has a severe limitation. The decision boundary is highly sensitive to the choice of μ, as seen in Fig. 4. Also, the classifier is not robust to the lighting condition of the image. Darker images tend to have higher misclassifications.

IV. Learning-Based Classification

A. Labeling significant images

In order to make the process of labeling images as efficient as possible, a selection of images was made, which was manually labelled. The purpose is to classify similar images into clusters and manually label several random cluster representatives. The spatial dimension of the images is exceedingly

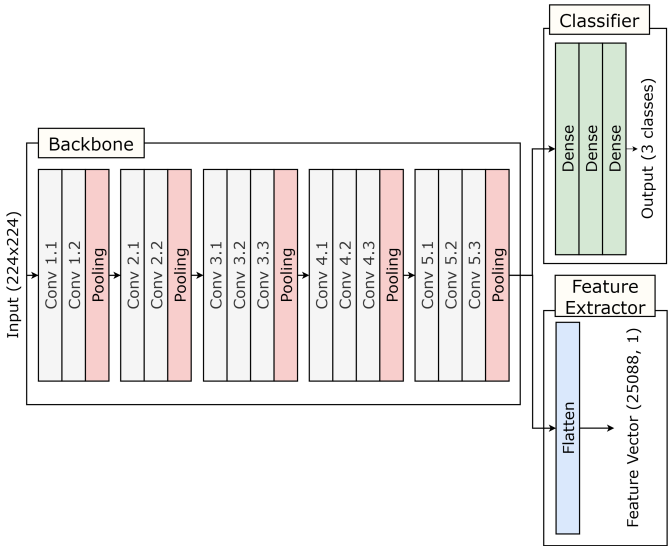

Fig. 5: **Feature Extractor and classifier:** VGG16, VGG19, and ResNet50 are used as backbones. The layers of the backbone are frozen. For the feature extractor, the output from the last layer of the backbone is flattened, it is later passed into PCA and KMeans Clustering. The layers of the classifier part are trainable. It can classify a given input droplet as empty, full, or not a droplet.

large so it is crucial to summarize the information contained in the images in order to correctly compare the images and decrease the effects of the curse of dimensionality.

1) Feature extraction: The VGG16 model with pre-trained weights for the ImageNet dataset was used for feature extraction. The VGG16 architecture consists of a succession of convolutional layers followed by several fully linked layers. The architecture can be divided between a backbone segment that finishes with the last max pooling layer and a residual that constitutes the classifier element of the design. It is consequently natural to take the outputs from the initial half of the architecture as the significant features of that image. By doing so, we have cleverly reduced the dimensionality of each image to 25088 features which utilize the information that VGG16 obtained from vast ImageNet dataset.

2) Principal component analysis (PCA): The principal component analysis (PCA) is a method that aims to extract, from a dataset with correlated features, the key orthogonal contributors (principal components) which explain most of the variance. In this research, once the features are extracted using VGG16, they are input into PCA which led to a reduction to about 300 discriminated features, therefore helping clustering methods in the next steps.

3) KMeans clustering: The process of finding similar images in this pipeline ends with the clustering using the KMeans method. For the starting positions, 100 points were selected which gradually converged to the center of the cluster by the iterative procedure of the Lloyd algorithm.

From each image cluster, arbitrarily 20 droplet images were selected and then manually labelled, resulting in a dataset that we will use for learning based models.

B. Models

In recent years, deep learning algorithms have been shown to be a very good tool for classifying photos. The invention of convolutional neural networks has defined a major breakthrough in computer vision [5]. Furthermore, the rise of processing power has helped to enable research into the implementation of these more and more complex technologies. Today, there are various neural network architectures that serve to successfully classify images [6], [7], [8]. For all these reasons the focus of this paper is on that part of machine learning.

Although it is possible that training from the beginning of the neural network would successfully solve the problem of droplet classification. In this paper, the focus is on transfer learning and the fine tuning of already existing models that have proven successful in similar tasks. With such an approach, the model has already learned the basic knowledge of image recognition on a large number of images. Therefore, it turned out that even with a small number of images in a new environment, the model can learn to classify them very quickly.

For the initial models, VGG16 and ResNet were selected as one of the standard architectures used in previous years. As we saw before that VGG16 can be divided into two parts (backbone and classifier) the same goes for ResNet. If we change the classification part of the mentioned trained models and train it on new data while freezing the layers of the backbone, we essentially train for the high-level features of the current data distribution. This method is called transfer learning. If in addition to the previous procedure, we also use a fine tuning approach where we train some of the layers from the backbone too. The parts of the the deep learning networks are represented in Fig. 5.

Both methods have proven to be extremely effective and we will therefore apply them to create two models for each architecture.

V. Results

As the paper is divided into two parts, in this chapter we will present the results of each one individually.

A. Droplet circle detection

To test the circle detection models, 5 images were manually labeled. The images were chosen to represent the population of images from the dataset, and these images were not used in any part of the training.

Qualitatively, one can see that in Fig. 6, YOLOv5 detected higher amounts of droplets as compared to Hough Gradient Method. A quantitative estimate of the each method was calculated by finding the Intersection over Union (IoU) of their respective outputs. The results are presented in Tab. V-A. The IoU metric for Images A and B as diplayed in Fig. 6 are shown in the first two columns and average IoU on all the

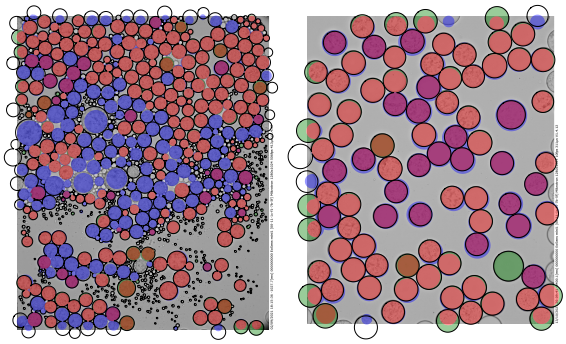

Fig. 6: **Circle Detection, Left:** Image A. **Right:** Image B. The blue circles denote the droplets marked by YOLOv5, the green circles denote the droplets marked by Hough Gradient Method, and the red/maroon circles denote the droplets marked by both the methods. Best viewed in colour and when zoomed.

5 images is shown in the last column. YOLOv5 outperforms Hough Gradient Method in all the cases.

IoU	Image A	Image B	Overall
Hough Gradient Method	0.512	0.624	0.782
YOLOv5	**0.796**	**0.688**	**0.835**

B. Droplet classification

In order to compare the different models proposed for classification, we will use the dataset obtained by the procedure explained in the subsection IV-A. The dataset consists of 2000 images that are arranged in sets for training (50%), validation (20%) and testing (30%). Each image was scaled to a size of 224x224 and parts of it were removed outside the detected circle.

First, results were obtained for the baseline method for which the threshold parameter was learned from the data using the sweep line technique. A threshold was selected that maximizes the f1 score on the dataset, which proved to be extremely good. F1 score was used because the dataset is unbalanced. Also for the same reason, we will predominantly emphasize weighted values in further results.

		Predicted	
		Full droplet	Empty droplet
Actual	Full Droplet	81	45
	Empty Droplet	20	521

VI. Discussion & Future Work

For interactive inference and further annotation of images a GUI was made using PyQT. The GUI also displays the number of empty and bacterial droplets. The output results can also be saved as a .csv file for further analysis, such as determining the diameter and volume of the circles. More information about its usage is provided in the project repository [link].

YOLOv5, as a circle detector, is able to detect more of the droplets. However, it is not able to produce perfect detections at the edges of the images. Also, the detected circles are not completely aligned with the droplets. We think that this is caused because currently the training procedure of the YOLOv5 model does not consider the fact that the output should be a circle, hence, there is not restriction on the anchor boxes that it uses for training. Therefore, the future work can involve imposing a constraint that the anchor boxes used by the model are squares. This, we think, will lead to a more aligned output.

TABLE I: Performance metrics of the learning based classification models

Model Transfer	F1-score (full)	F1-score (empty)	F1-score (nad)	Accuracy
VGG16	0.88	0.97	0.93	0.95
VGG19	0.90%	0.98	0.96	0.96
ResNet50	0.77	0.93	0.80	0.89
Model Fine-tuning (block5)	**F1-score (empty)**	**F1-score (full)**	**F1-score (other)**	**Accuracy**
VGG16	0.87	0.97	0.94	0.95
VGG19	83.1%	*ds*	0.782	0.23

VII. Conclusion

In this project, we developed a pipeline that is able to detect aqueous droplets in an emulsion and classify whether it contains any bacteria or not. We used learning-based approaches for both our detector and classifier and compared them against traditional algorithms. Further, we developed an intuitive GUI that, we believe, will reduce the time for biologists to count the number of bacterial cells in a given image.

References

[1] G. Jocher, A. Stoken, J. Borovec, NanoCode012, ChristopherSTAN, L. Changyu, Laughing, tkianai, A. Hogan, lorenzomammana, yxNONG, AlexWang1900, L. Diaconu, Marc, wanghaoyang0106, ml5ah, Doug, F. Ingham, Frederik, Guilhen, Hatovix, J. Poznanski, J. Fang, L. Y. , changyu98, M. Wang, N. Gupta, O. Akhtar, PetrDvoracek, and P. Rai, "ultralytics/yolov5: v3.1 - Bug Fixes and Performance Improvements," Oct. 2020. [Online]. Available: https://doi.org/10.5281/zenodo.4154370

[2] R. O. Duda and P. E. Hart, "Use of the hough transformation to detect lines and curves in pictures," *Commun. ACM*, vol. 15, no. 1, p. 11–15, jan 1972. [Online]. Available: https://doi.org/10.1145/361237.361242

[3] G. Bradski, "The OpenCV Library," *Dr. Dobb's Journal of Software Tools*, 2000.

[4] A. Bochkovskiy, C. Wang, and H. M. Liao, "Yolov4: Optimal speed and accuracy of object detection," *CoRR*, vol. abs/2004.10934, 2020. [Online]. Available: https://arxiv.org/abs/2004.10934

[5] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 1*, ser. NIPS'12. Red Hook, NY, USA: Curran Associates Inc., 2012, p. 1097–1105.

[6] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv 1409.1556*, 09 2014.

[7] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *CoRR*, vol. abs/1512.03385, 2015. [Online]. Available: http://arxiv.org/abs/1512.03385

[8] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. E. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," *CoRR*, vol. abs/1409.4842, 2014. [Online]. Available: http://arxiv.org/abs/1409.4842