Bacteria Image Classification and Segmentation

Andres Rios, Akhil Punia, Carlo Provinciali, Zhejin Dong, Paridhi Singh, Xinyuan Cao
Data Science Institute, Columbia University

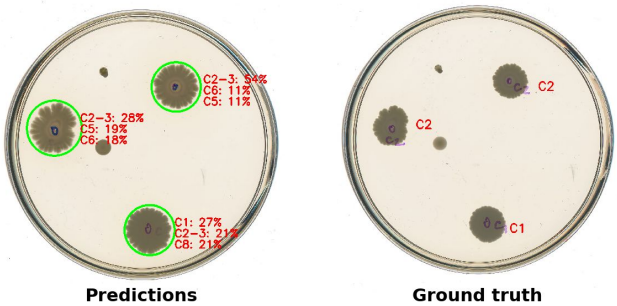


Figure 1: Image segmentation and classification on petri-dish with mixed bacteria species.

Abstract

Standard bacteria identification methods are time consuming, laborious and expensive. We show Convolutional Neural Network (CNN) models can be used to develop a rapid, automated process to accurately classify colonies of different bacteria strains, from scanned images of petri dishes containing mixed bacteria species (fig.1). Our method first divides the images into small patches, then we use an ensemble model consisting of a calibrated Random Forest (RF) that uses the scores from 4 Residual-CNN's [1], to output confidence scores to predict each patch either as negative (no bacteria colony) or belonging to a specific strain, achieving an accuracy of 93% when validated on petri dishes, unseen during training. Our method also allows to identify classify regions in the image where colonies are present, using the patch-level confidence scores to construct image-wide segmentation. As important contributions, we provide comprehensive evidence showing that growing bacteria colonies affect the surrounding growth media. In addition, unlike previous related works limited by manual labeling [9], we developed a semi-automated pipeline to accurately label positive and negative patches extracted from images of bacteria colonies cultured on solid media on a petri dish, useful for future work.

1. Introduction

Bacteria identification is a crucial problem in biology, with important applications in medicine, veterinary, biochemistry, food industry and farming. In medicine bacteria identification is critical for disease diagnosis, treatment of infection, and trace-back of bacteria outbreaks.

The gold-standard method for bacteria identification consists in culturing bacteria colonies on solid media in petri dishes (shallow cylindrical glass or plastic lidded dishes used in biology to culture microorganisms), followed by staining and microscopy. It follows stringent culturing procedures and safety protocols, requires dedicated equipment and chemical agents used for staining and the assistance of expert specialists for morphological identification. It is a time-consuming, laborious, and expensive process based on comparative analysis of the obtained samples with referential ones.

Therefore, the development of a widely used, rapid and automated process to identify and characterize bacterial species, would constitute a major innovation with application in several fields.

Some of the important features that can be recognized on the images are the shape of the bacteria cell and the shape and size of the colonies formed by the bacteria, with some species living solitary and some living in colonies colonies with characteristic structures and spatial arrangements [5]. However some species are morphologically diverse and their colonies may have multiple sizes and shapes depending on temperature, incubation time, and concentration of nutrients in the growth medium. In addition, even bacteria colonies of the same species can grow to vastly different shapes. Thus, recognizing of bacteria species under the

microscope is a challenging task even for experienced specialists and may require additional information of microbiological characteristics [5].

Different approaches using traditional machine learning and modern deep learning techniques, have been used in various studies for automated recognition and classification of bacteria species. Most of these studies use microscope-scale images.

The purpose of our work was to develop a machine learning model that using colony morphology, color, and other visual features, is able to classify bacteria species from real-scale (non-microscopic) images of mixed bacteria colonies cultured on solid media on a petri dish (figure 1).

As such, microscope level features like the shape of the bacteria cell, will not be available to our model. Nonetheless, we were still able to develop a method that produced high accuracy results from our experiments.

This paper is organized as follows. Section 2 reviews related work on using machine learning methods for classification of bacteria species and our contributions to the field. Section 3 discusses the dataset. Details of our method and results are presented in Section 4. Section 5 presents conclusions, limitations and proposes future work.

2. Related Work

Different approaches using machine learning methods, have been used in various studies for automated classification of bacteria species. Although many of the initial methods involved the use of handcrafted morphological features, we focused our review on those approaches that involve machine learning methods for either automated feature extraction or for classification from automatically extracted features.

Goodacre et al. [2] used the spectra obtained from bacterial suspensions, as input features to Multilayer perceptrons (MLP) to classify 5 bacteria species from urinary tract infections. The authors reported an accuracy of 80% using the Renishaw dispersive Raman spectroscope to obtain the bacteria spectra.

Fiannaca et al. [3], proposed a bacteria classification approach from 16S ribosomal RNA sequenced metagenomes, using the k-mers representation to map sequences as vectors, which were used as input features to train and compare two different deep learning architectures: a CNN and a deep belief network (DBN). The authors reported an accuracy of 91.3% on artificially simulated short-reads, from 1000 16S full-length sequences, using amplicon (AMP) technology for sequencing.

Huang et al. [4] propose a CNN architecture to be used as a first step towards identifying bacterial species. Their CNN models were primarily inspired by the AlexNet architecture. They also tested an unsupervised CNN, using an Autoencoder to automatically extract features from unlabeled images. The dataset they utilized consisted of microscope images of bacteria colonies belonging to 18 classes of the most common human pathogenic bacteria, which they collected from the clinical microbiology lab of Peking University First Hospital. The authors report a classification accuracy of 73%.

Zieliński et al. [5] used Fisher Vectors (FV, a classic texture representation) to classify genera and species of bacteria from microscopic images of bacteria colonies. The steps followed by their approach are: 1) obtain two types of 'image descriptors', the first being the Scale-Invariant Feature Transform (SIFT) [6] and the second type being the deep features extracted by truncating a CNN pre-trained on the ImageNet dataset [7]. 2) 'Image descriptors' were encoded into a single vector of features, using FV. 3) After extracting the FV representation, they used the vector as input features to be classified by Support Vector Machine (SVM) or Random Forest (RF) models. The authors reported an accuracy accuracy of 97.24% on a new dataset that they made publicly available for use by other researchers, containing 660 microscope images with 33 different genera and species of bacteria.

Talo [8] used a pre-trained ResNet-50 [1], replacing the fully connected (FC) layers with a new FC layer to be trained; to classify bacterial species from microscope images, using the same dataset developed by Zieliński et al. [5]. Two dropout layers were added to the pre-trained network to avoid overfitting. The author reports an accuracy of 99.2%.

Nie et al. [9] used CNNs to classify real-scale (non microscope-scale) images of bacteria colonies cultured on petri dishes. They first extracted small patches from the entire images. To distinguish *positive* patches (containing bacteria colonies) from *negative* patches (no bacteria colonies present), they used a Convolutional Deep Belief Network (CDBN) to obtain high level features that were fed as training data to an SVM trained to classify such *positive* and *negative* patches. For this task, the authors report a non-balanced accuracy of 97.14%, with precision of 89.9% and recall of 95.16%. Using *positive* patches they trained a CNN to predict the bacterial species in those patches. For this task, the authors report prediction accuracy ranges from 52.5% to 78.47%.

The work from Nie et al. [9] is, to the best of our knowledge (after exhaustive research), the only related work prior to ours that used real-scale images of bacterial colonies on petri dishes instead of microscope images. They labeled *positive* and *negative* patches manually, which limited their training data to a small subset of images. In addition, their choice of examples to be manually labeled, had a direct impact on the performance of their *positive* vs *negative* classification model. Hence they reported carefully choosing images to label so that each bacterial species and each growth phase was covered, but not necessarily in all contexts. In addition, they assumed that the solid growth media was not relevant in the task of species prediction, but recognizing that different strains may have different effects on the medium surrounding the colonies.

The contributions of our work are: 1) a semi-automated pipeline to accurately label *positive* and *negative* patches extracted from images of bacteria colonies cultured on solid media on a petri dish. This allowed us to automatically select the amount of *positive* pixels used as trigger for a patch to be considered *positive* or *negative* so, contrary to the work from Nie et al. [9], our model could consider the possible effect that different species might have on the surrounding media. 2) We provide comprehensive empirical evidence showing that bacteria colonies affect the surrounding media, which is an important contribution of our work. 3) A simpler one-step model to classify *positive* versus *negative* patches, which considerably outperforms the results presented by Nie et al. [9]. 4) We show that using existing CNN architectures developed for natural images, with subtle modifications, can outperform the results obtained by creating hand designed CNN architectures.

3. Data Preparation

The utilized data was collected by the Department of Biomedical Engineering at Columbia University (BME), who sponsored this work. The dataset was produced with 10 bacterial colonies obtained from soil samples, hereinafter called C1, C2, … C10. Each of the 10 colonies were cultured in 3 different petri dishes, *Serial*, *Control* and *Streak*, for a total of 30 petri dishes, using the following procedure:

1. Inoculum containing bacteria (from freezer) + 2mL of LB Broth (a nutritionally rich medium primarily used for the growth of bacteria) was prepared in 3 different Falcon tubes for each colony (30 tubes in total), which were then inoculated overnight at 37° Celsius.

2. Each of the 30 petri dishes were prepared with 15 ml of LB Agar (LB Broth containing Agar, whose properties allow the plating of bacterial cultures and generation of colonies), without air bubbles.

3. On ***Serial*** petri dishes, 9 different 2μL drops of inoculum dilutions at different concentrations were evenly spaced on one petri dish for each of the 10 colonies. The first drop had the same concentration of the original inoculum and drops 2 through 9 were diluted by sequentially adding 40μL of PBS (Phosphate Buffered Saline) each time.

4. On ***Control*** petri dishes, 1 single 2μL drop of the original inoculum was added to the center of the dish, using one petri dish for each of the 10 colonies.

5. On ***Streak*** petri dishes, streaking from top to bottom was performed on a different dish for each of the 10 colonies using 10μL of the original inoculum.

6. The petri dishes were left on incubator at 37° Celsius for the next 4 days. During such period, the petri dishes were scanned in the morning and in the -

late afternoon each day, for a total of 8 scans for each *Serial, Control* and *Streak* dish, which yielded 80 *Serial,* 80 *Streak* and 80 *Control* images. Scans

were taken in 48 bit color, using Epson scanner at resolutions ranging from 400 to 3200 dpi.

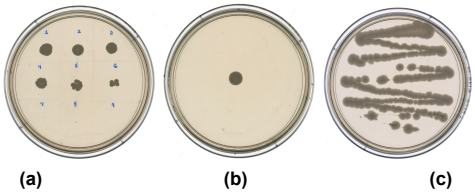


Figure 2: a) *Serial, b) Control and c) Streak,* petri dish.

Obtaining images at different growth phases is important because the morphological features of some bacterial colonies may change over time.

Figure 3: Colony C5 *Control* dish at 4 different growth phases.

Each of the 10 colonies was classified into its corresponding strain, using conventional methods, with 8 different bacteria strains identified (table 1).

Colony(ies)	Species
C1	Bacillus Megaterium
C2-C3	Bacillus Aryabhattai
C4-C7	Bacillus Paranthracis
C5	Lysinibacillus Boronitolerans
C6	Bacillus Mycoides
C8	Bacillus Wiedmannii - 1
C9	Bacillus Bingmayongensis
C10	Bacillus Wiedmannii - 2

Table 1: Identified species class for each original colony.

Colonies C8 and C10 were both classified as Bacillus Wiedmannii species, but it was determined that they corresponded to 2 different strains, with strong morphological differences so we treated them as 2 different classes for the purposes of our work.

4. Methodology

4.1. Exploration of Model Feasibility

We assumed different bacteria strains to be separable, on the basis that they either have differentiable visual features (static features) or differentiable growing patterns (dynamic features). To test the differentiability of static features, we compared the similarity of strain's feature vectors by embedding images.

4.1.1 CNN Autoencoder

Since the size of the original images is ~1500x1500 pixels, we used a CNN autoencoder to extract a compressed representation of the images. We also used the PCA to further reduce the dimensions to 2-d which can be easily visualized. We mixed *Control* images at all growing phases to train the autoencoder, whose outputs are expected to encode the general visual features of a particular bacteria strain. Figure 4 shows the encoder's output: on the left side is the compressed presentation of the right side image of colony C1.

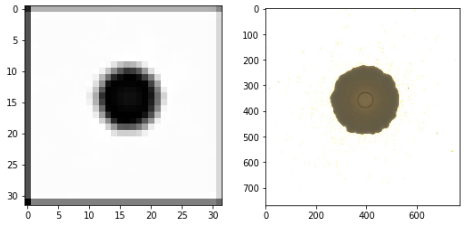


Figure 4: Image embedding(left) and original C1 image(right)

4.1.2 Distribution of the static embedding vectors

After applying PCA to reduce the embedding vectors to 2-d, we are able to visualize the distribution of each bacteria species. From figure 5, we can see that C10 and C9 have the most distinguishable visual features because they are furthest away from all the other colonies. However, C2-3 (green) and C6 (brown), C4-7 (red) and C5 (purple), distribute closely which indicates they share similar visual features. That might be an obstacle for the model trained on static features. In general, static features are able to disperse all 8 bacteria colonies which verifies the feasibility of a models based on static visual features.

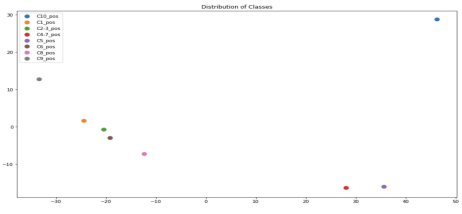


Figure 5: Distribution of the static embedding vectors

4.1.3 CNN-LSTM Autoencoder

Our second assumption is that different colonies may have diverging growing patterns such as C9 which stops expanding much earlier than the others. Such dynamic features can be exploited by the models that are built on the whole growing path rather than pure static pixel features.

LSTM models [13] are able to extract time series features from the entire growing process. We used image embedding vectors extracted from the CNN encoder, as input to an LSTM model. The outputs from the LSTM model are expected to encode the growing patterns of different bacteria strains. We used the LSTM model structure proposed by Srivastava et al. [12], which is shown in fig 6. The LSTM model consists of an encoder and decoder, and the output from the former ideally encodes the entire growthing patterns. The average Mean Absolute Error is around 0.22 and the average Mean Square Error (MSE) is around 0.063.

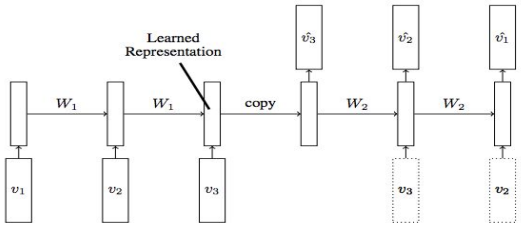


Figure 6: LSTM autoencoder structure [12]

We used *Serial* images as training data for the LSTM autoencoder. A specific CNN autoencoder is trained for a particular time interval which is expected to more concisely compress the images than a general CNN encoder. The outputs from 8 CNN autoencoders over the 8 growth phases would be used as input to the LSTM autoencoder where the final dynamic representation can be generated.

4.1.4 Distribution of the dynamic embedding vectors

We used PCA to reduce the dimensions of the dynamic representations to 2-d and then visualized the results. Unexpectedly, we found that C2-3 (green) and C6 (brown), C4-7 (red) and C5 (purple) are distant from each other, which could solve the obstacle mentioned in 4.1.2. Besides, we also found C1 and C2-3, C4-7 and C10 share similar growing patterns because they are physically close in the plot.

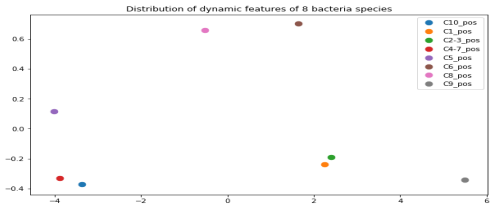


Figure 7: Distribution of the dynamic embedding vectors

4.1.5 Conclusion

In general, models trained on static visual features are expected to separate well most bacteria strains, except for some visually similar species such as C2-3 and C6, however their different growing patterns can be a good potential features to exploit in future work.

4.2. Image Processing

Since our goal was to develop a model that could locate and classify colonies from a petri-dish containing mixed bacteria species (fig. 1), our approach was to divide each training image into small patches and train a model to learn to classify each patch as either *negative* (no colony in patch) or the correct species for *positive* patches (patches containing colonies). This patch sampling technique also served the purpose of augmenting our available data, given the limited amount of available data.

4.2.1. Image Annotation

Instead of annotating patches manually - similar to Nie et al. [9] - or annotating *positive* and *negative* regions in the original images, we explored ways to automatically perform annotation, exploiting the fact that bacteria colony pixels are considerably darker than the background growth media pixels.

We had to first remove the petri-dish border from the images as such border includes dark regions that would otherwise be classified as *positive*. After trying different methods, we were able to identify the dish borders circles using the Hough Circle Transform function implemented in the OpenCV library [10]. In addition to helping us remove dish borders, identifying border circles allowed us to crop the images around the center of the petri dish, which was very useful to reduce the time required to generate the image patches (as some images included large white background areas around the

dishes) and it was additionally very useful to rotate the images around the petri dish center (used for data augmentation, as explained later), without losing any part of the image after performing rotations.

After removing dish borders, we designed an algorithm to automatically create masks identifying positive pixels. For this, we first segmented the grayscale version of the images using simple pixel value thresholding, which yielded binary images or masks with pixels above the threshold value, labeled as *positive* and pixels below the threshold, labeled as *negative*. Such masks contained noise so we labeled each connected *positive* region and filtered out small labeled regions to get clean masks with *positive* and *negative* pixels.

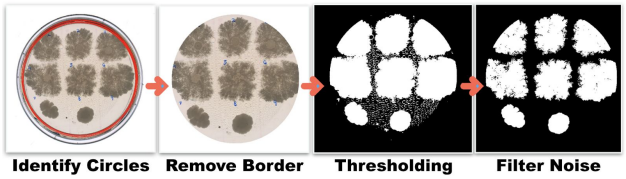


Figure 9: image annotation pipeline.

Using the annotation masks, we were able to generate images containing *positive* pixels only (fig. 10), from which we could sample patches for model training and compare results against model training using patches that include both *positive* pixels and surrounding growth media.

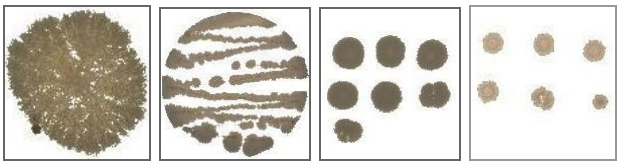


Figure 10: images with positive pixels only

4.2.2. Patch Sampling

Using the binary masks, we could accurately identify *positive* regions within the original image and consequently, easily identify *positive* and *negative* patches. For quick experimentation, we created a function to automatically generate training patches, where we could define several parameters including patch size, stride (step between patches), the minimum amount of positive pixels in a patch for it to be classified as a *positive* patch, rotations before extracting patches, and whether to generate *positive* patches that contain *positive* pixels only or generate *positive* patches that include both *positive* pixels and pixels corresponding to surrounding growth media.

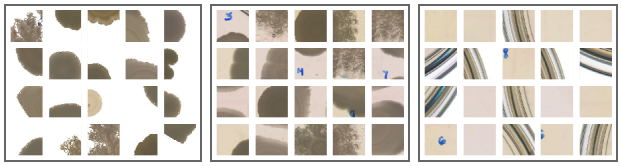


Figure 11: a) *positive* patches with *positive* pixels only, b) *positive* patches with surrounding media c) *negative* patches.

4.3. Baseline Models

Nie et al. [9] showed that deep CNN models significantly outperform other methods on bacterial colony image segmentation and classification, from real-scale (non microscope-scale) images of bacteria colonies cultured on petri dishes.

As baseline models, we adopted both hard coded features and CNN. For hard coded features, we utilized Hu moments[14] and Haralick textures[15], capturing shape and textures respectively. For simple NN, we used 18-layers Residual CNN (ResNet-18) proposed by He et al. [1].

We chose these models as they are, in our experience, robust enough to learn complex feature representations and at the same time, fast to train.

4.4. Training Set-up

After analyzing the data provided by the BME department, we noted that there was only one *Serial*, *Control* or *Streak* petri dish, for each of the original 10 colonies (thought for each petri dish, there were 8 different images corresponding to different growth phases). In order to set up a viable training and validation split without leaking information, we should use different petri dishes for training than for validation, which forced us to perform training using *Serial* dishes (where each petri dish cultured 9 different drops of the same original colony inoculum at different concentrations) and validate the model using *Control* petri dishes (where each petri dish cultured a single drop of the corresponding original colony inoculum). A limitation of this approach is that *Serial* dishes contain several growing colonies competing for resources, which might have an effect on the growing patterns as compared to *Control* dishes

containing a single colony, but we had no other option due to limited data available.

We couldn't use *Streak* dishes, as the visual growing patterns greatly differ from *Serial* and *Control* dishes and as mentioned previously, for *Streak* dishes, we only had a single petri dish for each of the original 10 colonies so we weren't able to set up a viable training-validation split using *Streak* dishes, without leaking information.

For *Serial* petri dishes, for each of the 10 original colonies, we had 8 images corresponding to 8 different growing phases, for a total of 80 training images available for training. After grouping images corresponding to the same bacteria strain, we got an imbalanced number of available training images per class, as detailed in table 2.

Class	**# of *Serial* images**
C1	8
C2-3	16
C4-7	16
C5	8
C6	8
C8	8
C9	8
C10	8

Table 2: Serial images for each class.

In addition, because some classes contained larger *positive* regions than others, performing patch sampling resulted in a different number of *positive* patches being generated per class, which accentuated the class imbalance. A similar situation occurred with patches sampled from *Control* petri dishes to be used to validate training results.

To overcome the class imbalance, we included class weights in the training loss function and used balanced accuracy when validating and comparing results. Other training parameters used are:

- Learning rate (lr): 1e-3, decay: 0.01
- L2 regularization: 0.1
- Optimizer: RMSprop, momentum: 0.9
- Trained 50 epochs
- Reduced lr rate by 0.85 on plateau of val loss, with the patience of 3 epochs.

Validation patches were sampled from original images, without any type of image preprocessing. Using *positive* patches, we trained our baseline NN model to learn to predict the correct bacteria species, obtaining a balanced validation accuracy of 81.7%.

We also tried extracting hard coded features [14][15], used as input to a RF but this method yielded a lower validation accuracy of 73.34%.

4.5. Experiments

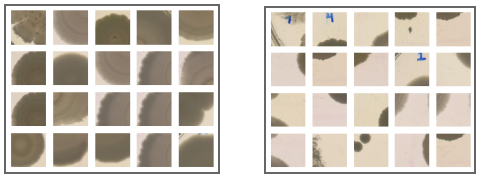
We carried out several experiments, aimed to improve the results obtained with our baseline NN model, on predicting the correct bacteria species from *positive* patches.

Because the patch size would influence the results (as an example, too small patches may not capture enough features), we experimented with different patch sizes (32x32, 64x64, 112x112, 128x128, 224x224, 256x256). Since the original ResNet-18 architecture is designed for input images of size 224x224 pixels, patches were resized to 224x224. Nevertheless, after modifying the network input size, we found that using patch size of the same size as the network input (without resizing), always produced the best results. Best results were obtained with network input size and patch size of 128x128.

Because we reduced the input size of the original architecture, we experimented with different kernel size for the first convolution layer and got the best results reducing its size from original 7x7 to 3x3.

The only other modification that we made to the original ResNet-18 architecture, was to use the new residual block proposed by He et al. in [11].

We also experimented with training our baseline model with *positive* patches containing almost entirely *positive* pixels (fig. 12.a) and gradually reducing the total amount of *positive* pixels, which produced gradually better results, starting to decrease only until we trained with patches containing almost no *positive* pixels but mostly growth media (fig. 12.b).



(a) (b)

Figure 12: *positive* patches with a) almost entirely *positive* pixels, b) a low amount of *positive* pixels.

The above was an indication that the surrounding growth media contained valuable information for the classification task (we used different petri-dishes to validate the training results). We then compared training using *positive* patches that contained *positive* pixels only (fig. 11.a) versus training with *positive* patches that include both *positive* pixels and surrounding growth media (fig. 11.b) and we got better results with the latter method, confirming that the surrounding growth media was relevant for the classification task and consequently, that different bacteria strains have different effects on the solid growth media surrounding the colonies, which is an important contribution of our work.

After training using the best set up from the above experiments, we improved the results of our baseline NN model from a balanced accuracy of 81.7% to 84.65% predicting the correct bacteria species from *positive* patches. This, without using any data augmentation, other than sampling overlapping patches.

4.6. Adopted Model

From the confusion matrix of the classification results, we realized that our model was having difficulties to distinguish class C1 from C2-3 and to distinguish class C4-7 from C5. As strategy, using the same architecture and configuration, we trained 2 other Residual CNNs, one to learn distinguish C1, from C2-3, from 'all other' classes; and other trained to learn distinguish C4-7, from C5, from 'all other' classes. In addition, we trained a similar CNN to distinguish from *positive* and *negative* patches..

As a result, we had 4 models, the first one producing 8 predicted probabilities (one for each bacteria strain), the second one producing 3 predicted probabilities (C1, C2-3, 'all other'), the third one also producing 3 predicted probabilities (C4-7, C5, 'all other') and the last one producing 2 predicted probabilities (*positive* and *negative*).

We produced an augmented training dataset (with image rotations and flips), consisting of 9 classes (8 bacterial strains + *negative* patches) and using the combined 16 probabilities from the above 4 CNN models, we cross validated traditional machine learning classifiers including a logistic linear classifier, a linear SVM, a kernel SVM and a Random Forest (RF); trained to learn to predict either *negative* or the correct bacterial species from positive *patches*. The Random Forest model yielded the best accuracy results. We then calibrated the selected Random Forest model so that the output for each class can be directly interpreted as a confidence level or probability of such class being present. We will refer to this RF model as the *combine model*.

The complete training pipeline of the adopted model is illustrated in figure 13.

As a final step, in addition to sampling overlapping patches, we experimented using additional data augmentation techniques and obtained the best results when we augmented the number of sampled patches by rotating training images around the center of the petri dish, every 45º (0, 45, 90, 135…), before performing patch sampling.

Nie et al. [9] found that the rotation of images weakened the performance of their model, hinting that parts of the features captured by their method were dependent on a particular orientation.

Rotating training images around the petri dish center before patch sampling helped our model to become rotational invariant, without dependence from the orientation of bacterial colonies at prediction time. We tested the rotational invariance of our model by randomly rotating validation images and model performance didn't decrease.

Other data augmentation techniques (e.g. horizontal and vertical flips, blurring, etc.) didn't have a meaningful effect in the performance of our model.

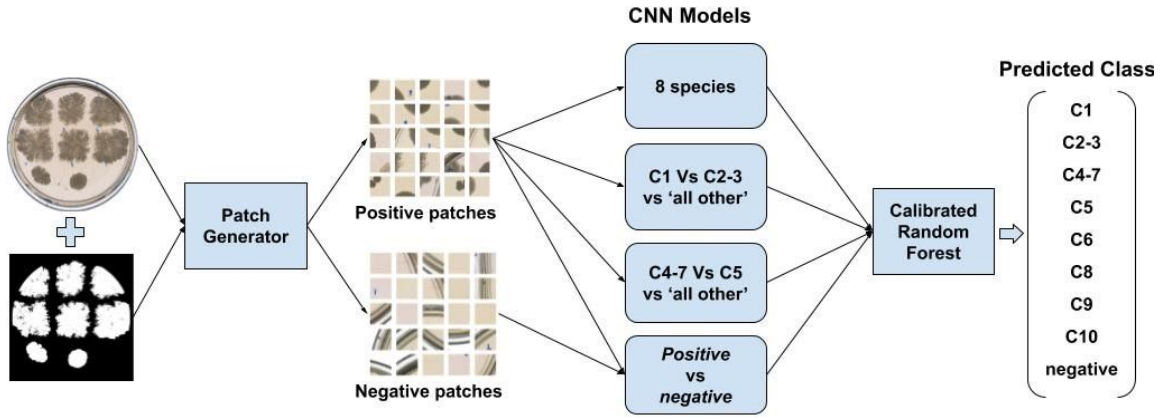


Figure 13: Training pipeline for the adopted model.

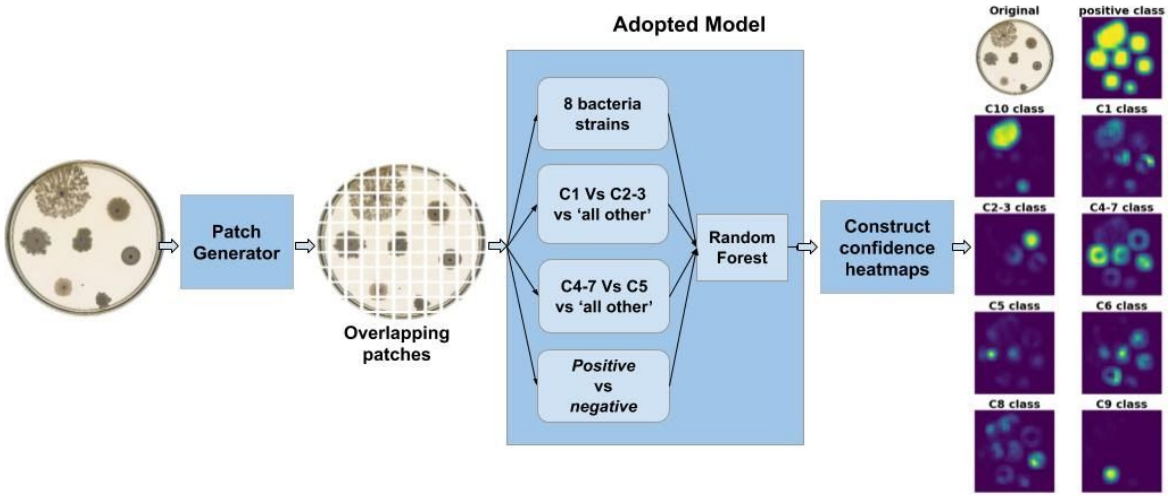


Figure 16: End-to-end prediction pipeline.

4.7. Prediction Pipeline

Figure 16 illustrates the complete prediction pipeline. Importantly, our method does not require any type of preprocessing at prediction time.

In the prediction pipeline, square patches of size 128 pixels are sampled on a rectangular grid with 8-pixel stride. In this way, each 8x8 pixel region within the original image is covered by a total of 16 (128/8) overlapping patches. For each of such overlapping patches, our adopted model outputs 9 class confidence scores (*negative* patch class + 8 bacteria strain classes). We then use simple voting, which is equivalent to average the 16 different 9-arrays of class confidence scores for each 8x8 pixel region, which we can use to construct image-wide confidence heatmaps for each of the 9 classes. Figure 14 shows an example of an image of a petri dish with colonies of different bacteria strains, followed by the confidence heatmaps for each of the 9 classes in our model.

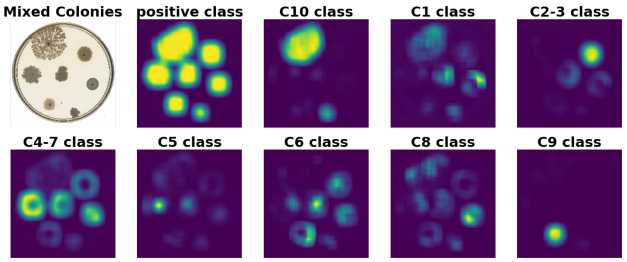


Figure 14: Image-wide probability heatmaps for each class.

For the *negative* patch class, for better appreciation, we visualize the complement, this is, the confidence heatmap for positive patches.

We used 8-pixel stride due to computing and memory limitations (we worked on free online

GPU servers), so every pixel contained within each 8x8 pixel region in the original image, will share the same class probability values. Smaller strides will produce more precise and smoother boundaries.

4.8. Using Confidence Heatmaps

Confidence heatmaps can be used in multiple ways. As an example, we can choose any region of any size within the image and get the predicted probabilities for each class. Figure 15 shows an example visualizing the top 3 predicted probabilities for several regions of different sizes.

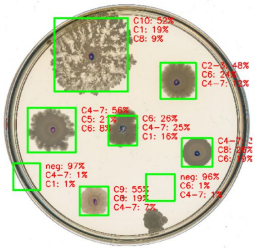


Figure 15: Top End-to-end prediction pipeline.

Instead of manually choosing the image regions to predict, we can use the confidence heatmap for the *positive* vs *negative* patch prediction, define a confidence or probability threshold to yield a binary mask with regions of the heatmap above and below the probability threshold being labeled as *positive* or *negative* (fig. 17).

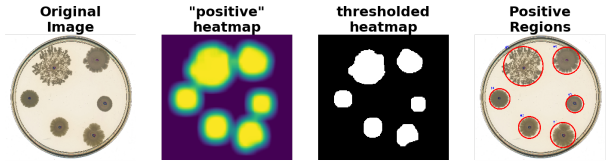


Figure 17: Labeling positive regions.

After automatically identifying and labeling *positive* regions, we can then use the confidence heatmaps to get the predicted probabilities for each *positive* class. Figure 1 shows an example visualizing the top 3 predicted probabilities for regions automatically labeled as positive.

Another way to use the confidence heatmaps is to color-code each pixel in the image by the highest class probability as in fig. 18. For better visualization, we can colour all except the *negative* patch class (no colony present).

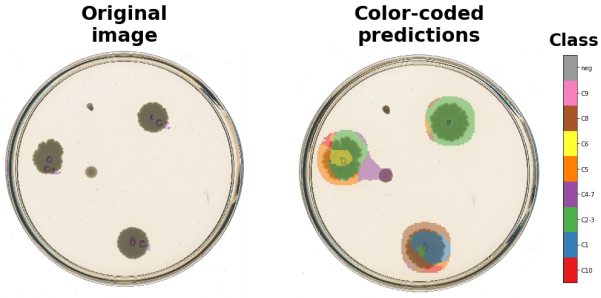


Figure 18: Color-coded image by highest class probability.

It is expected that for some pixels within a colony, a wrong class may show up with the highest confidence, but overall those wrong classes should get out-voted by the class that is consistently predicted across all the pixels within the colony so that the class with the most votes is predicted

4.9. Results

With the final adopted model, we were able to have a balanced accuracy of 93% predicting the correct bacteria strain from positive patches on the validation set. This is much better than our baseline CNN (84.65%) and hard coded features method (73.34%). For the detailed performance analysis, the confusion matrix is presented in fig. 19.

		Predicted class									
		C1	C2-3	C4-7	C5	C6	C8	C9	C10	Total / support	Recall
Actual class	C1	353	7	0	0	0	4	0	0	364	0.97
	C2-3	33	330	0	0	0	1	0	0	364	0.91
	C4-7	0	0	233	130	0	1	0	0	364	0.64
	C5	0	0	20	344	0	0	0	0	364	0.95
	C6	0	0	0	0	364	0	0	0	364	1.00
	C8	3	0	0	1	0	360	0	0	364	0.99
	C9	0	0	0	0	0	0	364	0	364	1.00
	C10	1	0	0	0	0	0	0	363	364	1.00
	Precision	0.91	0.98	0.92	0.72	1.00	0.98	1.00	1.00		

Figure 19: Confusion matrix for model validation

From the confusion matrix in fig. 19, we can observe that the adopted model performs well on most classes, except trying to distinguish C4-7 from C5, which may be due to their very similar morphological characteristics, indistinguishable to the human eye at the resolution level of our data.

As previously discussed, in order to set up a viable training and validation split without leaking information, we should use different petri dishes

for training than for validation, which forced us to perform training using Serial petri dishes and validate the model using Control petri dishes. This resulted in no data available for testing (unless we leaking information).

After discussing this with the BME, additional data was collected but such data was sampled from a different population and we found strong morphological differences, probably corresponding to different bacteria strains. To confirm this, a Whole Genome Sequencing (WGS) could have been performed but the cost would be beyond the scope and resources for our work.

We however were able to test the accuracy of the model on the segmentation task, this is, distinguishing *positive* from *negative* regions in the image and obtained an accuracy of 99%.

5. Conclusions and Future Work

We show that it is possible to use CNN models to develop a rapid, automated process to classify colonies of different bacteria species on a petri-dish, with high accuracy. Our method first divided the images into small patches then we use an ensemble model consisting of a calibrated RF that uses the scores from 4 CNN's to output confidence scores for *negative* patch prediction and for each of the 8 different bacteria strains in our data, achieving an accuracy of 93% when validated on patches generated from petri dishes different to the ones used for training. Our method also allows to identify classify regions in the original image where colonies are present, using the patch-level confidence scores to construct image-wide segmentation and classification.

As important contributions, we provide comprehensive empirical evidence showing that growing bacteria colonies affect the surrounding growth media and we developed a semi-automated pipeline to accurately label *positive* and *negative* patches extracted from images of bacteria colonies cultured on solid media on a petri dish, which comes handy for future work, as previous related works we limited by manually labeling, like the work of Nie et al. [9].

Some limitations for our work were the limited amount of available data. As an example, in order to set up a viable training and validation split without leaking information, for training we used *Serial* petri dishes which contain several growing colonies competing for resources, which might have an effect on the growing patterns as compared to *Control* petri dishes which contain a single colony and were used to validate our model. We couldn't test this effect, due to the limited data available. Similarly, no although our method was validated on petri dishes not seen during training ubt used for model selection, no additional petri dishes were available for final testing, as discussed in the *Results* section of this paper.

Another important limitation to our work is the limited resolution of images. The most important features used by experts to classify bacteria species are the shape of the bacteria cell and the shape and size of the colonies formed by the bacteria, with some bacteria living solitary and some others living in colonies with characteristic structures and spatial arrangements observable on microscope images. In addition, as explained in 4.1.2, from available images, some colonies are not separable by static visual features (pixel), which may be ascribed to the low image resolution. Images of higher resolution may further improve model prediction performance. For future work, we expect a combination of microscope images plus real scale images scanned from petri dishes, to yield very high performance.

Our adopted model exploits static features. However, as we discussed in 4.1.2 and 4.1.4, dynamic features could help differentiate the colonies further such as C4-7 and C5. The reason we did not model over growing patterns is out of the consideration that the test datasets are static, which means we only have the images at 8 different growing phases, while using growing patterns would require adequate amount of data for time series analysis. Nevertheless, modeling the whole growing path is feasible, and we will define this project more like a video prediction problem rather than image prediction.

6. References

[1] He, K., Zhang, X., Ren, S., Sun, J. Deep residual learning for image recognition, 2016. In: CVPR.

[2] Goodacre, R., Burton, R., Kaderbhai, N., Woodward, A.M., Kell, D.B. and Rooney, P.J. Rapid identification of urinary tract infection bacteria using hyperspectral whole-organism fingerprinting and artificial neural networks, 1998. In: Microbiology, 144(5), pp.1157-1170.

[3] Fiannaca, A., La Paglia, L., La Rosa, M., Renda, G., Rizzo, R., Gaglio, S. and Urso, A. Deep learning models for bacteria taxonomic classification of metagenomic data, 2018. In: BMC bioinformatics, 19(7), p.198.

[4] Huang L, Wu T. Novel neural network application for bacterial colony classification, 2018. In: Theor Biol Med Model. 15:22.

[5] Zieliński B, Plichta A, Misztal K, Spurek P, Brzychczy-Włoch M, Ochońska D. Deep learning approach to bacterial colony classification, 2017. In: PLoS ONE 12(9): e0184554.

[6] Lowe DG. Object recognition from local scale-invariant features, 1999. In: Computer vision. The proceedings of the seventh IEEE international conference on. vol. 2. Ieee; 1999. p. 1150–1157.

[7] Russakovsky O, Deng J, Su H, Krause J, Satheesh S, Ma S, et al. Imagenet large scale visual recognition challenge, 2015. In: International Journal of Computer Vision; 115(3):211–252.

[8] Talo, M. An Automated Deep Learning Approach for Bacterial Image Classification, 2019. In: International Conference on Advanced Technologies, Computer Engineering and Science (ICATCES 2019).

[9] Nie, D., Shank, E.A. and Jojic, V. A deep framework for bacterial image segmentation and classification, 2015. In: Proceedings of the 6th ACM Conference on Bioinformatics, Computational Biology and Health Informatics (pp. 306-314). ACM.

[10] Bradski, G. The OpenCV Library, 2000. In: Dr. Dobb's Journal of Software Tools.

[11] K. He, X. Zhang, S. Ren, J. Sun. Identity Mappings in Deep Residual Networks, 2016. In: ECCV.

[12] Srivastava, N., Mansimov, E., & Salakhudinov, R. Unsupervised learning of video representations using lstms, 2015. In: International conference on machine learning (pp. 843-852).

[13] Hochreiter, S. & Schmidhuber, Jü. 'Long short-term memory', 1997. In: Neural computation 9 (8), 1735--1780.

[14] Hu, Ming-Kuei. "Visual pattern recognition by moment invariants." IRE transactions on information theory 8.2 (1962): 179-187.

[15] An introduction to haralick texture :http://wiki.awf.forst.uni-goettingen.de/wiki/index.php/Haralick_Texture