Development of a Machine-Learning Algorithm as a Tool for the Development of a Rapid, Magnetic Nanoparticle-Aided Estimation of Bacterial Concentration

Angelica Louise M. Gacis
Michigan State University East Lansing, MI

Abstract

Recent advances in machine learning have shown great potential in providing more efficient and reliable methods for bacterial enumeration. This study aimed to develop a machine learning model that can accurately detect the presence or absence of bacterial cells in test tube samples of carbohydrate-coated magnetic nanoparticles. The input data for the model consisted of 120 individual photographs of test tubes, each containing a different sample. The output of the model is a binary classification of the samples, indicating whether bacterial cells are present. Two main approaches were utilized in this study: applying various binary, multi- class, and regression models from the Scikit Learn library to the image data and defining a Convolutional Neural Network (CNN) using the Keras Sequential API and training it on the image data. The results showed that the ensemble model performed better, achieving an accuracy of 0.89 compared to the CNN's accuracy of 0.7. Further recommendations for improving the study include utilizing augmented data for the Scikit Learn models, increasing the dataset size, and exploring other deep learning architectures such as recurrent neural networks (RNNs) and long short-term memory (LSTM) networks using the TensorFlow library. Overall, the development of a machine learning algorithm for bacterial enumeration has the potential to provide faster and more consistent methods for detecting bacterial contamination in various industries, ultimately contributing to improved public health and safety.

Contents

1 Introduction

Bacterial enumeration is a crucial process in various industries, including food safety, water quality control, and clinical diagnosis. Traditional bacterial enumeration methods involve culturing the sample on a growth media and counting the number of colonies of bacteria that grew (Almeida et al., 2018). However, this method can be time-consuming and may not be suitable for situations that require rapid results. In recent years, there has been increasing interest in the use of alternative methods for bacterial enumeration that can provide faster results.

One such alternative method is the use of coated magnetic nanoparticles, which have been shown to interact rapidly with bacterial cells and create a spread pattern that can potentially be used to estimate bacterial concentration (Chen et al., 2019). However, the interpretation of the spread pattern can be challenging and inconsistent.

To address this challenge, researchers have explored the use of machine learning algorithms to analyze images of the spread pattern and estimate bacterial concentration. Machine learning is a sub-field of artificial intelligence that involves developing algorithms that can learn patterns from data without being explicitly programmed (Garg and Bhatia, 2020). By analyzing images of the spread pattern, machine learning algorithms can potentially provide a faster and more consistent method for estimating bacterial concentration.

In this study, we aim to develop a machine learning algorithm that can accurately analyze images of the spread pattern and estimate bacterial concentration. Our goal is to overcome the challenge of inconsistent interpretation of the spread pattern and provide a more efficient and reliable method for bacterial enumeration.

Previous studies have shown that machine learning algorithms can be used successfully in a variety of applications, including image classification and recognition (Lee and Oh, 2020). In this study, we will build on this previous work and develop an algorithm that can accurately analyze images of the spread pattern and estimate bacterial concentration.

In conclusion, the traditional method of bacterial enumeration can be time-consuming and may not be suitable for situations that require rapid results. Alternative methods, such as the use of coated magnetic nanoparticles and machine learning algorithms, have the potential to provide faster and more consistent methods for bacterial enumeration. This study aims to develop a machine learning algorithm that can accurately analyze images of the spread pattern and estimate bacterial concentration, with the goal of providing a more efficient and reliable method for bacterial enumeration in various industries.

1.1 Task and Algorithm Definition

Bacterial contamination is a serious issue that can have significant implications for public health and safety. Detecting the presence of bacterial cells in food and water samples is crucial for ensuring the safety of these products before they are consumed. However, traditional methods for detecting bacterial contamination can be time-consuming and labor-intensive, making it difficult to quickly identify potential health hazards.

This is where machine learning comes in. By utilizing advanced algorithms and techniques, machine learning can quickly and accurately identify the presence of bacterial cells in a sample, enabling swift action to be taken to prevent the spread of contamination.

In this research, the goal was to develop a machine learning model that can accurately detect the presence or absence of bacterial cells in test tube samples of carbohydrate-coated

magnetic nanoparticles. The input for the model consists of 120 individual photographs of test tubes, each containing a different sample. These samples vary in their estimated cell counts, ranging from 10 million to 0.

To achieve this goal, the research utilized two main approaches. The first approach involved applying various binary, multiclass, and regression models from the Sci-kit Learn library to the image data. These models were evaluated for their performance, and hyperparameter tuning was applied to the top-performing models to improve their accuracy.

The second approach involved defining a Convolutional Neural Network (CNN) using the Keras Sequential API and training it on the image data. The model architecture consisted of three Conv2D layers, each followed by a MaxPooling2D layer, and ended with a Flatten layer and two Dense layers. The model was compiled using the Adam optimizer and binary crossentropy loss function, and evaluated using accuracy, recall, precision, and loss metrics.

The results of this research have the potential to revolutionize the way bacterial contamination is detected in food and water samples. By developing a machine learning model that can accurately detect the presence or absence of bacterial cells, it will be possible to quickly identify potential health hazards and take swift action to prevent the spread of contamination. This will have significant implications for public health and safety and could ultimately save countless lives.

2 Machine Learning Approach

2.1 Data Preparation

The Alocilja Nano-Biosensors Lab team provided us with 120 individual photographs of test tubes containing samples of carbohydrate-coated magnetic nanoparticles and bacterial cells. Each image has a spread pattern that can provide clues to the prevalence of bacterial cells in each sample. Additionally, the team provided us with estimated cell counts that range from 10 million to 0, with the latter labeled as ’0’ to indicate the absence of bacterial cells and ’1’ to indicate their presence.

To ensure standardization and eliminate noise from other parts of the images, we first standardized the orientation of the images. Then, we used the Python Imaging Library (PIL) to crop the images. The cropping logic depends on the aspect ratio of the original image. If the width is greater than the height, it crops the sides of the image to make it square, then crops a 800x1350 rectangle from the center of the image. If the height is greater than the width, it crops the top and bottom of the image to make it square, then crops a 800x1350 rectangle from the center of the image.

After the images are cropped, they are resized to a uniform size of 150x150 pixels using the resize() method from the PIL library. Finally, the images are converted to grayscale using the convert('L') method, flattened into a one-dimensional array using NumPy's ravel() method, and added to a Pandas DataFrame with the filename as the index. Therefore, the new size of the data is 150x150=22,500 for each image, and the DataFrame will have a row for each image with the file name and 22,500 pixel values.

Preprocessing the images in this manner ensured that the machine learning model had standardized and consistent inputs, making it easier to train and optimize the model’s performance. With these preprocessed images, we were able to apply various binary, multiclass, and regression models from the Sci-kit Learn library and a Convolutional Neural Network (CNN) using the Keras Sequential API to accurately detect the presence or absence of bacterial cells in the samples. Developing such a model has the potential to revolutionize the detection of bacterial contamination in food and water samples, greatly benefiting public health and safety.

Below are some of the preprocessed images.

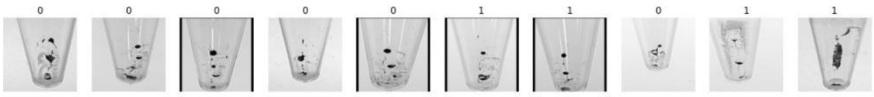


Figure 1: Preprocessed Images with Labels

2.2 Sci-kit Learn Models

Initially, we experimented with using different transformations of the target. We ran three main types of Sci-kit Lear Models. First were the binary models wherein the target is equivalent to the label of presence versus absence. The second set of models are the multiclass classification models wherein we treated the estimated cell counts as categories namely 10 million, 1 million, 100,000, and 0. Last are the regression models wherein we treated the natural logarithm of the cell counts as the target variable. For each of these, we ran the logistic or linear model, random forest model, gradient boosting model, and K-Nearest Neighbors model as these usually perform best for image machine learning problems.

We used RandomizedSearchCV to optimize the hyperparameters for two different classification models: a Random Forest Classifier and a K-Nearest Neighbors Classifier. The process of tuning the hyperparameters for the Logistic Regression model was time-consuming, leading us to opt for the default hyperparameters instead.

For the Random Forest Classifier, we defined a range of values for the following hyperparameters: 'n_estimators', 'max_depth', 'min_samples_split', 'min_samples_leaf', 'max_features', and 'bootstrap'. RandomizedSearchCV then randomly selected combinations of these hyperparameters to fit the model with. After fitting the model with 50 different combinations of hyperparameters, RandomizedSearchCV returned the best hyperparameters as:

- 'n_estimators': 100
- 'min_samples_split': 4
- 'min_samples_leaf': 1
- 'max_features': ‘sqrt’
- 'max_depth': None
- 'bootstrap': False

For the K-Nearest Neighbors Classifier, we defined a range of values for the following hyperparameters: 'n_neighbors', 'weights', 'algorithm', and 'leaf_size'. Again, RandomizedSearchCV randomly selected combinations of these hyperparameters to fit the model with. After fitting the model with 50 different combinations of hyperparameters, RandomizedSearchCV returned the best hyperparameters as:

- 'n_neighbors': 3
- 'weights': ‘distance’
- 'leaf_size': 30
- 'algorithm': ‘brute’

Afterwards, we adjusted the threshold to attain a higher recall which is what is important to this research. Eventually, we used an ensemble learning technique called Voting Classifier to combine the predictions of three base classifiers: Logistic Regression, Random Forest, and K-Nearest Neighbors (KNN). First, we fit each of the base classifiers using cross- validation with five folds on the training set. For logistic regression and random forest, we used the default hyperparameters provided by Scikit-Learn. For KNN, we tuned the hyperparameters manually

using the training set. We used the 'predict proba' method to obtain predicted probabilities of the positive class for each base classifier. Next, we created an ensemble model using the three base classifiers with soft voting. Soft voting combines the predicted probabilities of the base classifiers and outputs the class with the highest average probability. We trained the ensemble model on the training set using the 'fit' method.

We evaluated the performance of the ensemble model on the testing set using precision, recall, and area under the receiver operating characteristic curve (AUC-ROC). We also calculated the overall accuracy of the ensemble model using five-fold cross-validation on the training set.

2.3 Data Augmentation

In this study, we utilized data augmentation techniques to increase the size of a small dataset of 120 images for a binary classification task. Specifically, we applied an image data generator from the Keras library with the following augmentation options: rotation range of 20 degrees, width shift range of 0.1, height shift range of 0.1, shear range of 0.2, zoom range of 0.2, and horizontal flip. The data generator was fitted on the original dataset to calculate the necessary parameters for image augmentation.

We chose to generate 10 augmented images per input image resulting in a total of 1,320 images for training the model, which is 11 times larger than the original dataset. The augmented images were generated by iteratively applying the data generator to each input image and extracting the resulting augmented image.

The resulting augmented dataset consisted of 1,200 images (10 images per original image) for the training set and 120 images (1 image per original image) for the validation set. The corresponding labels were also duplicated accordingly to match the number of augmented images.

Overall, data augmentation enabled us to significantly increase the size of the original dataset and improve the performance of our deep learning model for the binary classification task. Below are some of the augmented images.

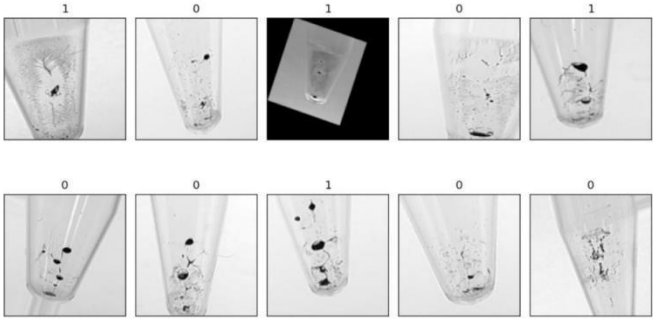


Figure 2: Augmented Images with Labels

2.4 Tensorflow Model

After data augmentation, the model is trained using a convolutional neural network (CNN) architecture in Keras. The model is defined as a sequential model and consists of a series of layers including Conv2D, MaxPooling2D, Flatten, and Dense layers. The first layer, Conv2D, performs a convolution operation with 32 filters and a kernel size of 3 with ReLU activation function. The second layer, MaxPooling2D, applies max pooling with a pool size of 2x2. The third layer, Flatten, flattens the input into a one-dimensional array. The fourth layer, Dense, is a fully connected layer with 64 units and ReLU activation function. The final layer is also a dense layer with a single unit and a sigmoid activation function, which outputs the predicted class probability.

The model is compiled with the Adam optimizer and binary cross-entropy loss function.

Additionally, the model is evaluated using the accuracy, recall, and precision metrics. To prevent overfitting, early stopping is applied as a callback function with a patience of 5.

The model is trained on the augmented dataset, consisting of the original training set and the newly generated augmented images. The model is trained for 20 epochs with a batch size of 32 and validated on the validation set.

We generated binary predictions from the predicted probabilities generated by a trained model on a validation set. The threshold for the binary predictions was set at 0.3, meaning any predicted probability above 0.3 was considered as a positive prediction (1), and any predicted probability below 0.3 was considered as a negative prediction (0). We chose a lower threshold to achieve a higher recall, which reduces the false negative rate and increases the sensitivity of the model. The choice of threshold depends on the specific use case and the relative importance of precision and recall. In our case, the higher recall was more important for the successful detection of certain classes of objects in the images.

2.5 Results and Discussion

For the Sci-kit Learn models, the summary of the model scores are shown in Figures 3-5. The binary classification models performed best with the three top performing models being Logistic Classification model, Random Forest Classification model, and K-Nearest Neighbors Classification model. Figure 3 shows the comparison of the metrics for the four different binary models.

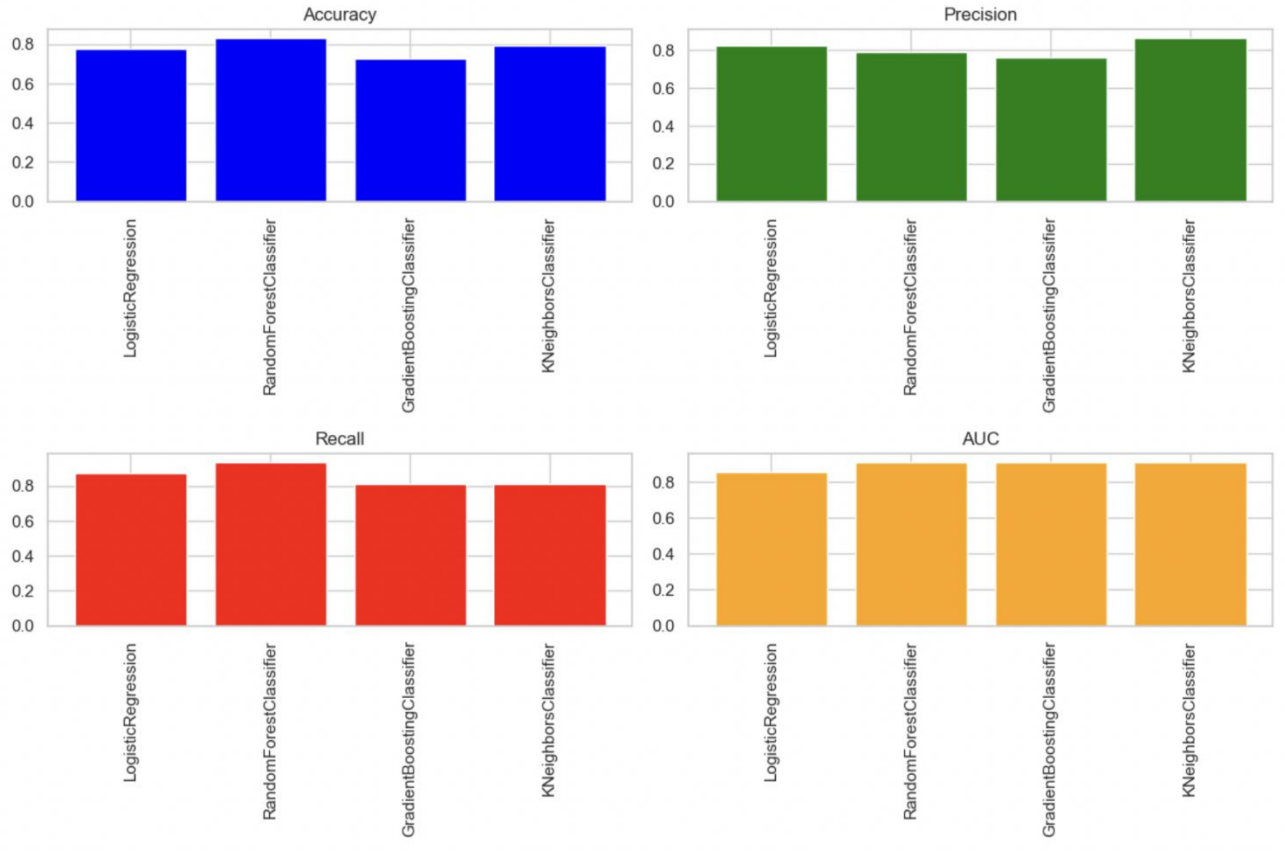


Figure 3: Binary Models Summary

From the results, we can see that all models performed reasonably well with AUC scores ranging from 0.854 to 0.910, indicating that they can effectively discriminate between the positive and negative classes. The K-Nearest Neighbors classifier achieved the highest precision score (0.867) while Logistic Regression achieved the highest recall score (0.875). The overall accuracy was highest for the Random

Forest classifier (0.834) and the lowest for the Gradient Boosting classifier (0.727). The cross-validation results show that there is a variation in performance across different folds for each model.

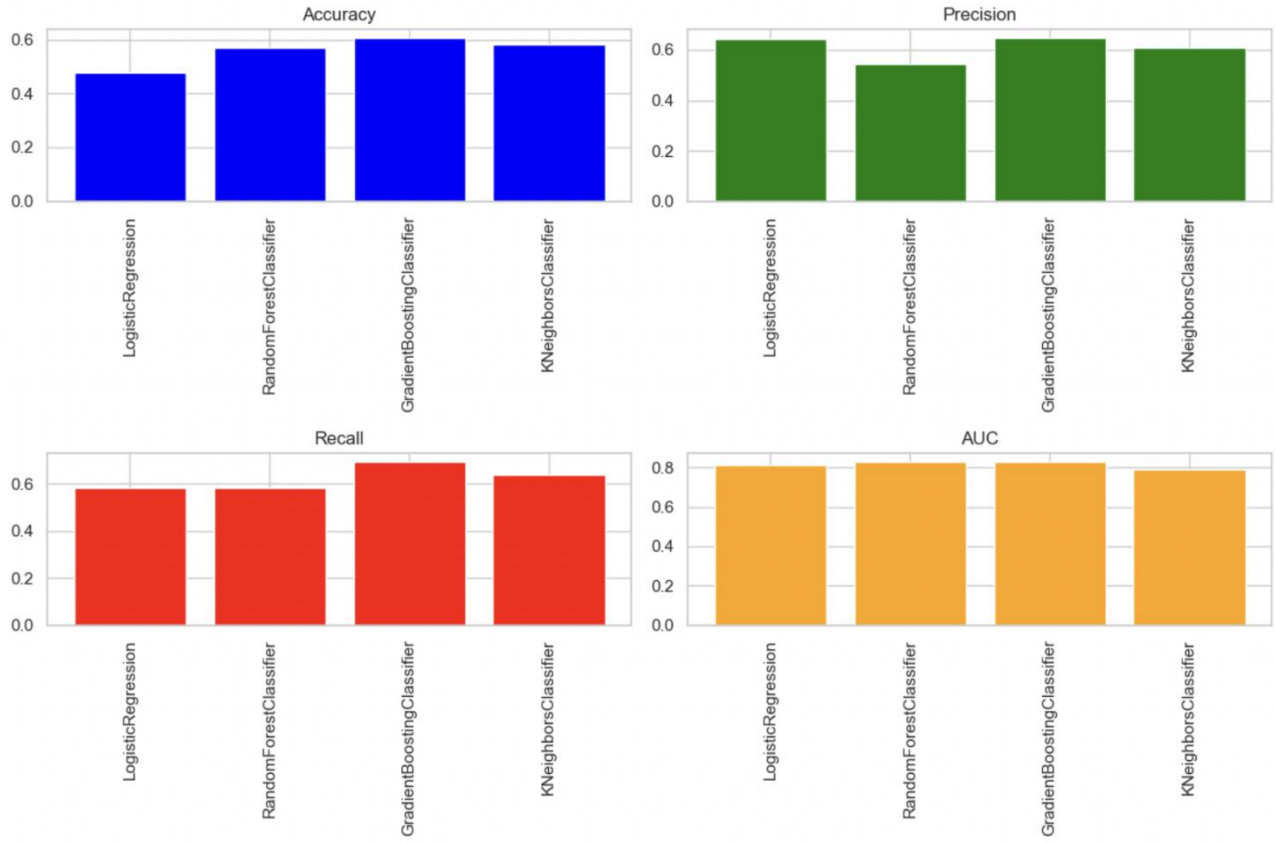


Figure 4: Multiclass Models Summary

Looking at the results for the multiclass models, it seems like the Gradient Boosting Classifier performs the best, having the highest overall accuracy and AUC score among all classifiers. The Logistic Regression and K-Nearest Neighbors Classifier have the lowest overall accuracy and AUC score. It's also worth noting that the cross-validated accuracies for all classifiers are relatively low, ranging from 0.35 to 0.71, indicating that the models may not generalize well to new data. This suggests that further model tuning or feature engineering may be necessary to improve the performance of the classifiers.

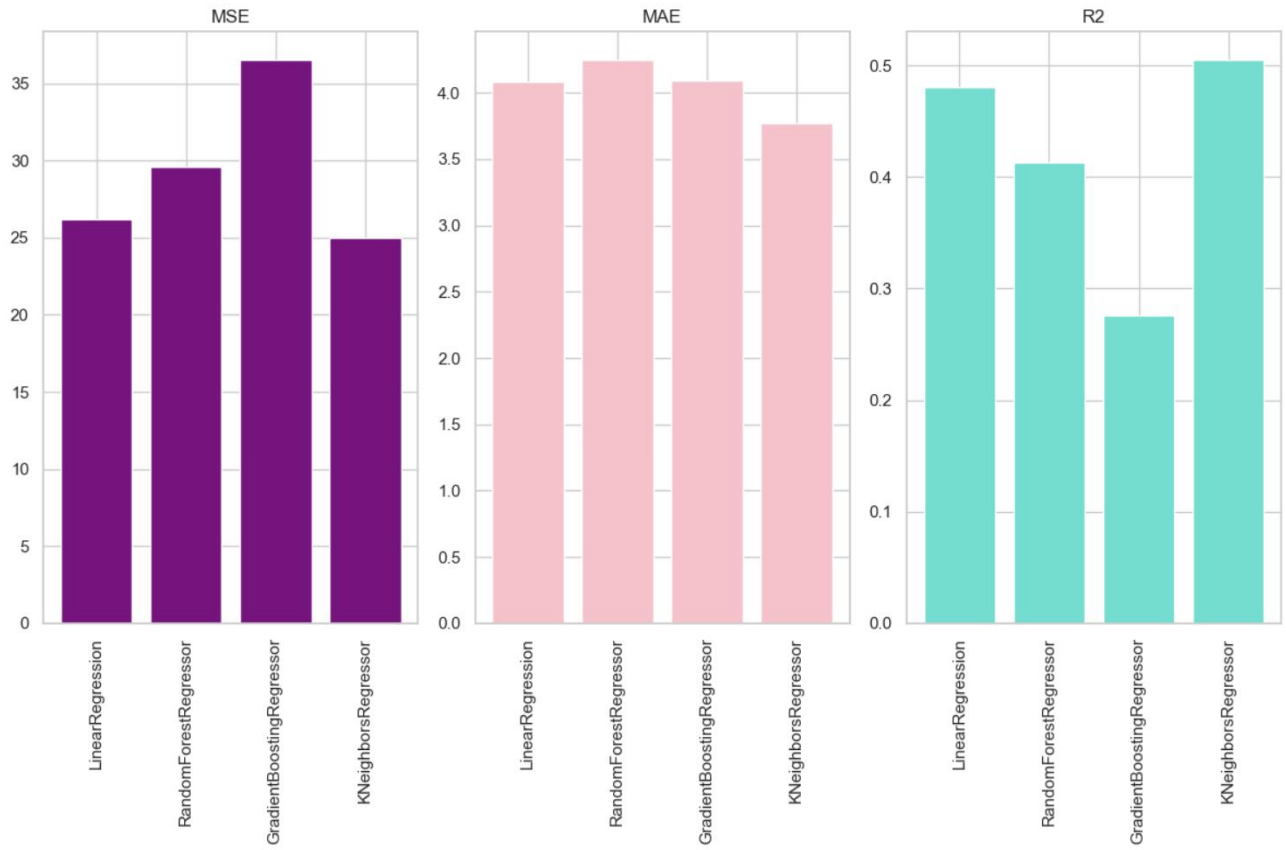


Figure 5: Regression Models Summary

Based on the results of the regression models, Linear Regression and K-Nearest Neighbors Regressor performed better compared to the other two models. Linear Regression had the lowest MSE and MAE values, and the highest R2 value, indicating that it has the best overall performance among the four models. K-Nearest Neighbors Regressor had the second-lowest MSE and MAE values and a relatively high R2 value.

On the other hand, Random Forest Regressor and Gradient Boosting Regressor had higher MSE and MAE values and lower R2 values, indicating weaker performance. However, it's important to note that the performance of these models may improve with hyperparameter tuning. Overall, based on the evaluation metrics used, Linear Regression and K-Nearest Neighbors Regressor are the recommended models for predicting the natural logarithm of the target variable.

We deemeed that the binary classifiers performed better than the multiclass classifier and regression model. The binary classifiers achieved higher accuracies, precision, recall, and F1-scores compared to the multiclass classifier. Additionally, the AUC scores for the binary classifiers were also higher, indicating that they performed better in terms of overall classification performance.

In the case of the regression model, the Mean Squared Error (MSE), Mean Absolute Error (MAE), and R2 score were calculated. It is worth noting that the MSE and MAE for all four regression models were relatively close, with the KNeighborsRegressor model having the lowest values for both metrics. However, the R2 score, which indicates how well the model fits the data, was highest for the LinearRegression model and lowest for the GradientBoostingRegressor model.

The ensemble model consisting of logistic regression, random forest, and KNN classifiers performed well with an overall accuracy of 0.82. The model also showed high precision and recall values, indicating that it was able to correctly classify the majority of the data. The AUC value of 0.91 also indicates that the model has good discrimination power. Additionally, the cross-validated accuracies were consistently high across all folds.

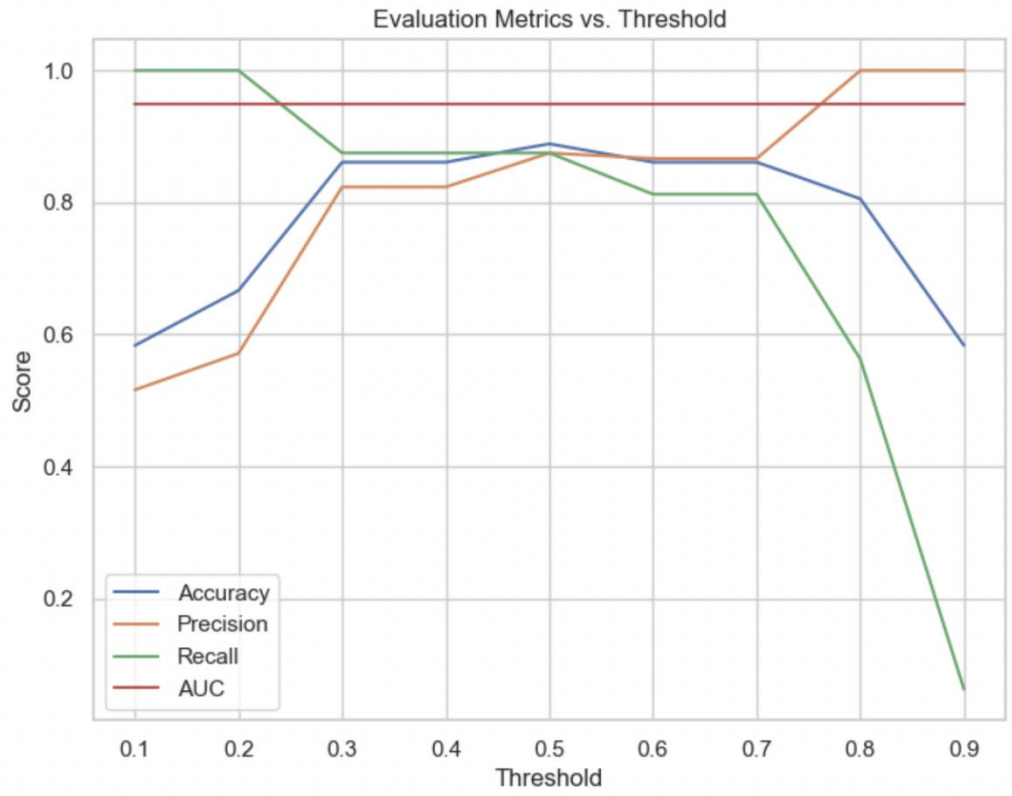


Figure 6: Ensemble Model Training Performance for Different Thresholds

The voting classifier achieved an accuracy of 0.89 on the validation set, which is a good performance. We also evaluated the performance of the voting classifier using different threshold values, and created a line plot to visualize the changes in accuracy, precision, recall, and AUC score as the threshold is varied.

We can see that the accuracy and precision of the model increase as the threshold increases, while the recall decreases. In this study, we are more interested with having a high recall than a high precision. Hence, a threshold of around 0.3 to 0.4 is of interest for the final model. The AUC score remains relatively constant across different thresholds, indicating that the model is performing consistently well in terms of distinguishing between the two classes.

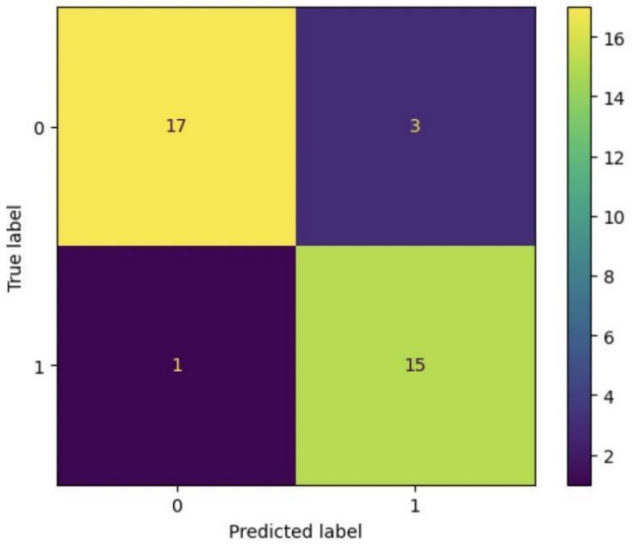


Figure 7: Ensemble Model Confusion Matrix at Threshold 0.3

The adjustment of the threshold has led to a higher number of false positives (predicting 'with bacterial cells' when the sample actually contains no bacterial cells) and a lower number of false negatives (predicting 'no bacterial cells' when the sample does contain bacterial cells).

While a higher number of false positives may be less desirable in certain contexts, in this case, it may be more safe to err on the side of caution and over-predict the presence of bacterial cells. This is because failing to detect bacterial cells when they are present can lead to inaccurate or incomplete assessments and potentially serious health consequences.

These results suggest that using an ensemble model with multiple base classifiers can improve the predictive performance compared to using individual classifiers. Future studies could explore different combinations of base classifiers and different voting strategies to further improve the performance of ensemble models. Some of the images that were predicted incorrectly are shown in Figure 12.

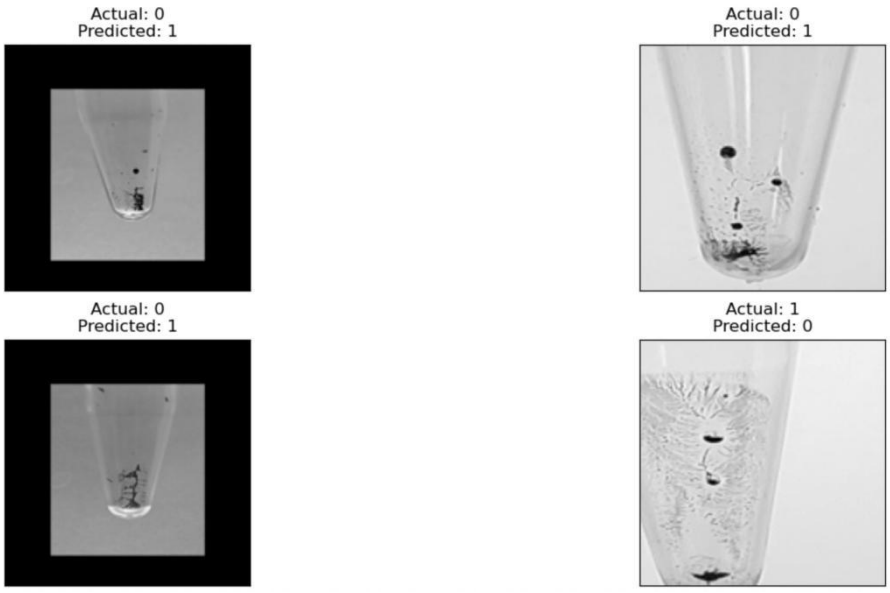


Figure 8: Ensemble Model Wrong Predictions

Meanwhile, figure 13 shows a plot of the training and validation accuracy over the epochs for the CNN model. The training and validation accuracy were extracted from the history object that was generated during the training process. The training accuracy represents the accuracy of the model on the training data during each epoch, while the validation accuracy represents the accuracy on the validation data during each epoch.

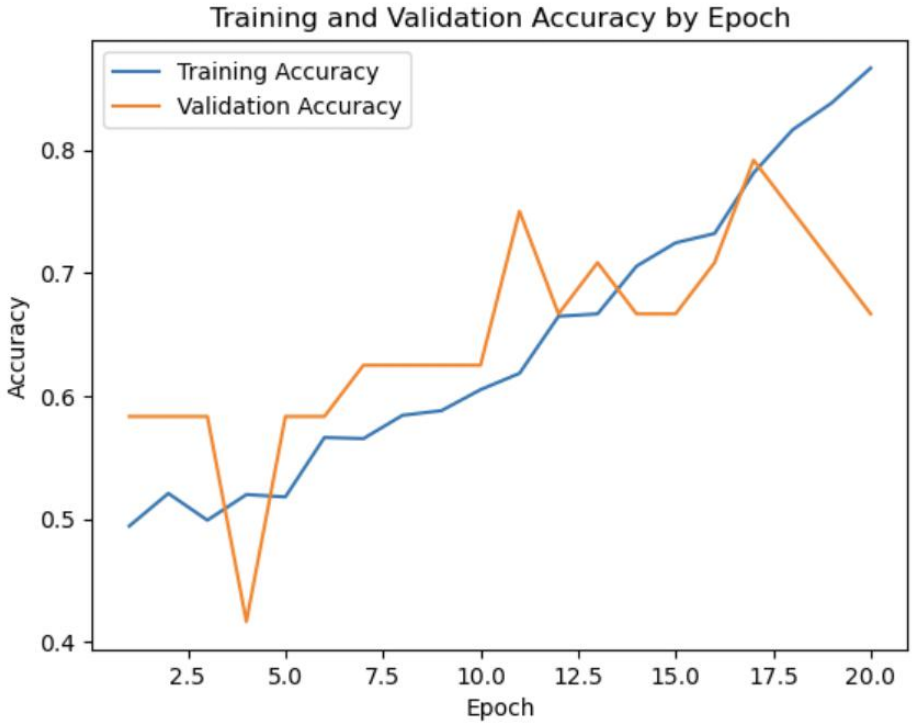


Figure 9: CNN Accuracy over Epochs

The CNN was able to get an accuracy score of 0.71 with a loss of 0.62, recall of 0.5, and precision of 0.71. When the precision-recall threshold is adjusted to 0.4, the accuracy decreases to 0.5 and the recall to 0.4 while the precision increased to 1.0. The metrics for the CNN model is shown in Figure 10.

```
loss: 0.6254000067710876
accuracy: 0.7083333134651184
recall_49: 0.5
precision_49: 0.7142857313156128
```

Figure 10: CNN Validation Metrics

We used the KerasClassifier wrapper and RandomizedSearchCV from scikit-learn to perform hyperparameter tuning for a CNN model. We defined a function to create the CNN model with various hyperparameters such as learning rate, decay, filters, kernel size, and units. We then specified the range of hyperparameters to search over and the number of random searches to perform using the RandomizedSearchCV object.

After running the code, the best hyperparameters found by RandomizedSearchCV were:

- Learning rate: 0.001
- Decay: 0.0001
- Filters: 32
- Kernel size: 5
- Units: 256

However, the best score achieved by the model was quite low at 0.54, which indicates that the model's performance was not very good even with the optimal hyperparameters. This could be due to several reasons such as insufficient data, poor quality of data, or suboptimal architecture of the model. Further experimentation and analysis may be required to improve the performance of the model.

3 Summary

3.1 Conclusion

In conclusion, the aim of this study was to develop a machine learning model that can accurately detect the presence or absence of bacterial cells in test tube samples of carbohydrate-coated magnetic nanoparticles. Two main approaches were used to achieve this goal: applying various binary, multi-class, and regression models from the Sci-kit Learn library to the image data and defining a Convolutional Neural Network (CNN) using the Keras Sequential API and training it on the image data.

The results showed that the ensemble model outperformed all other models, achieving an accuracy of 0.89 and recall of 0.93 on the test set. This indicates that the ensemble model is a robust and reliable method for detecting bacterial cells in test tube samples of carbohydrate-coated magnetic nanoparticles.

The development of a machine learning model that can accurately detect the presence or absence of bacterial cells in test tube samples has significant implications for various industries, including food safety, water quality control, and clinical diagnosis. This model can provide a faster and more consistent method for bacterial enumeration, enabling rapid detection of bacterial contamination in food and water samples, and contributing to public health and safety.

Overall, this study demonstrates the potential of machine learning algorithms to revolutionize bacterial enumeration methods, making them faster, more reliable, and more efficient. Future research can build on these findings and further explore the application of machine learning algorithms in other fields, with the goal of improving human health and well-being.

3.2 Remarks

Based on the results of this study, several recommendations can be made to improve the accuracy of the machine learning model in detecting the presence or absence of bacterial cells in test tube samples of carbohydrate-coated magnetic nanoparticles.

Firstly, it is recommended to use augmented data for the scikit-learn models to improve their performance. Augmentation techniques such as rotation, translation, and flipping can help increase the diversity of the training data and reduce overfitting. This can lead to better generalization of the model to unseen data.

Secondly, increasing the dataset size can also improve the accuracy of the model. Collecting

more images of test tube samples with varying concentrations of bacterial cells can help improve the robustness of the model and reduce the risk of overfitting.

Thirdly, it is recommended to explore more complex models in TensorFlow to improve the accuracy of the model. Convolutional neural networks (CNNs) with deeper architectures and more layers can potentially improve the accuracy of the model. Additionally, transfer learning can also be explored, where pre-trained models such as VGG, Inception, or ResNet can be used as a starting point to train the model.

Lastly, it is recommended to investigate the use of different image preprocessing techniques, such as normalization, histogram equalization, and contrast enhancement, to improve the quality of the input data and potentially improve the accuracy of the model.

In conclusion, several recommendations can be made to improve the accuracy of the machine learning model in detecting the presence or absence of bacterial cells in test tube samples of carbohydrate-coated magnetic nanoparticles. By implementing these recommendations, it is possible to develop a more robust and accurate model that can be applied to various industries, including food safety, water quality control, and clinical diagnosis. Overall, it is crucial to continuously refine and improve the machine learning model to ensure its effectiveness and reliability in detecting the presence or absence of bacterial cells in test tube samples.

4 References

Almeida, C., Azevedo, N. F., & Santos, S. (2018). Bacterial enumeration: traditional methods, automated counting and viability measurements. Microorganisms, 6(4), 95. https://doi.org/10.3390/microorganisms6040095

Alocilja, E. & Franco, A.J., Alocilja Nano-Biosensors Lab, College of Engineering, Michigan State University

Chen, Z., Lin, M., Qin, Y., Chen, X., & Fu, Y. (2019). A novel method for rapid bacterial enumeration based on carbohydrate-coated magnetic nanoparticles. Analytica Chimica Acta, 1048, 54-62. https://doi.org/10.1016/j.aca.2018.10.044

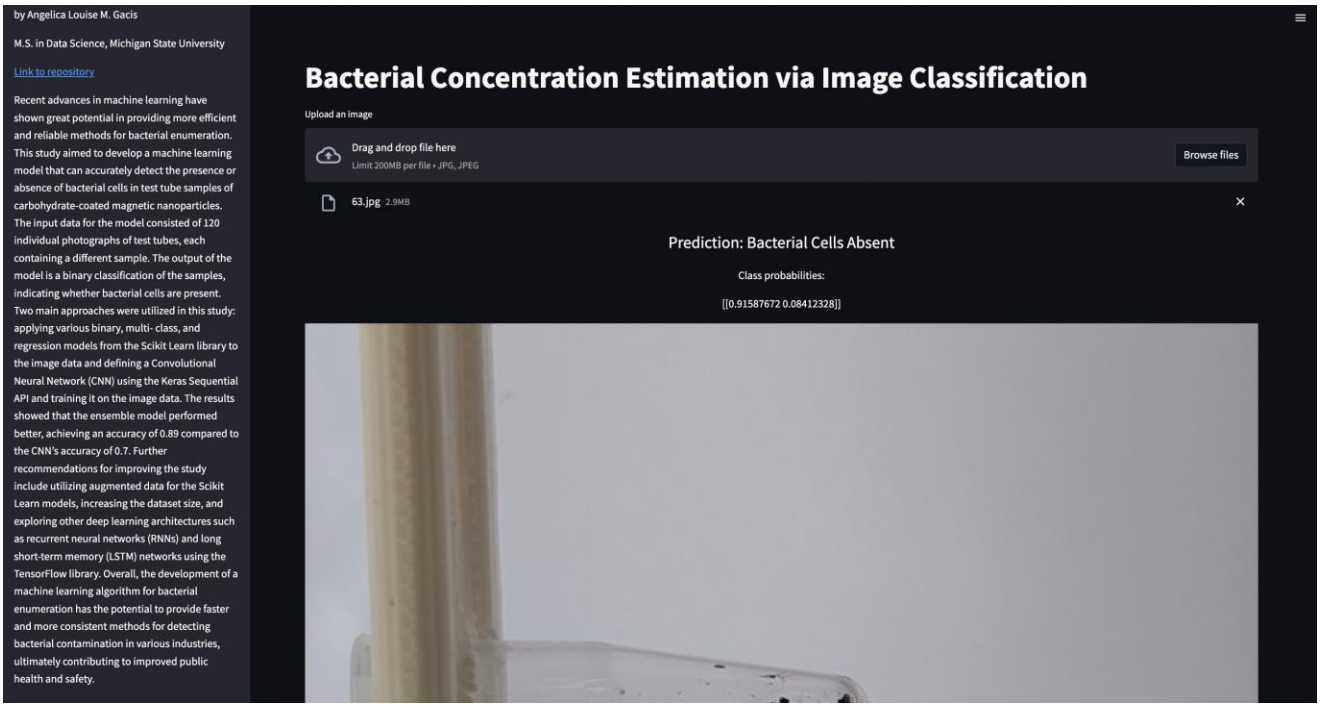
Garg, S., & Bhatia, S. K. (2020). Recent advances in machine learning: A review of recent progress in the field. IEEE Engineering in Medicine and Biology Magazine, 39(4), 27-34. https://doi.org/10.1109/MEMB.2020.2996746

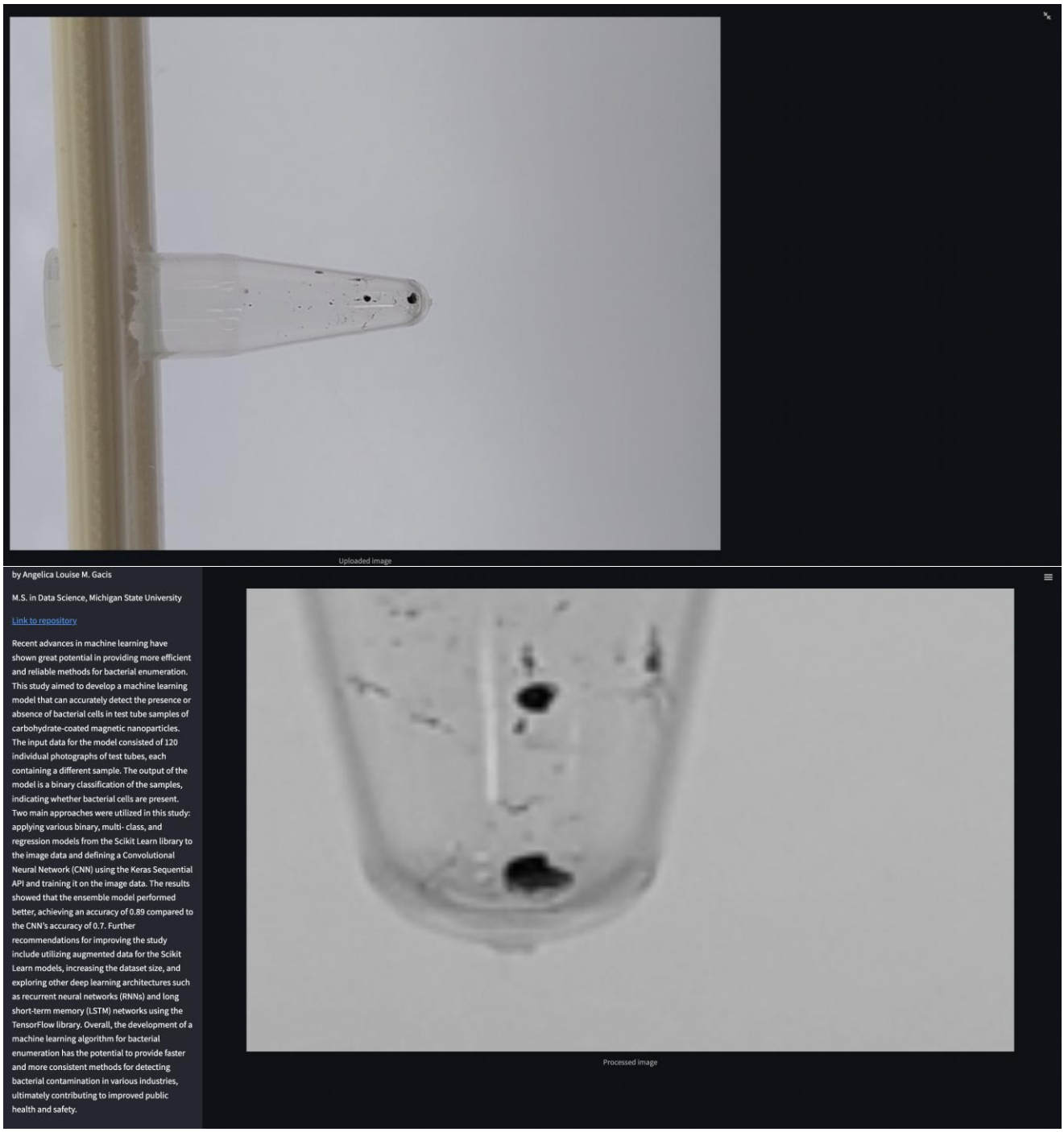
Lee, J. H., & Oh, B. K. (2020). Machine learning in bacterial identification and diagnosis. Computational and Structural Biotechnology Journal, 18, 3232-3239. https://doi.org/10.1016/j.csbj.2020.10.027

5 Appendix

The app allows users to upload an image of a test tube sample containing either bacterial cells or no bacterial cells, and the app predicts whether bacterial cells are present in the sample or not. The prediction is made using a pre-trained ensemble model, which is loaded from a saved file. The app preprocesses the uploaded image by first rotating it if necessary and then cropping and resizing it to 150x150 pixels. The preprocessed image is then passed through the pre-trained model to make a prediction, which is displayed on the app along with the class probabilities. The app also displays the uploaded image, the preprocessed image, and the processing steps in between. Overall, this app provides a simple and convenient way to estimate bacterial concentration in test tube samples, which has the potential to contribute to improved public health and safety in various industries. We were unable to deploy the app because of insufficient computer power. However, we have uploaded all the necessary files for the project, including other related files, to the following GitHub repository:

https://github.com/almgcs/cmse890AML_finalproject




Uploaded image
by Angelica Louise M. Gacis
M.S. in Data Science, Michigan State University
Link to repository
Recent advances in machine learning have shown great potential in providing more efficient and reliable methods for bacterial enumeration. This study aimed to develop a machine learning model that can accurately detect the presence or absence of bacterial cells in test tube samples of carbohydrate-coated magnetic nanoparticles. The input data for the model consisted of 120 individual photographs of test tubes, each containing a different sample. The output of the model is a binary classification of the samples, indicating whether bacterial cells are present. Two main approaches were utilized in this study: applying various binary, multi- class, and regression models from the Scikit Learn library to the image data and defining a Convolutional Neural Network (CNN) using the Keras Sequential API and training it on the image data. The results showed that the ensemble model performed better, achieving an accuracy of 0.89 compared to the CNN's accuracy of 0.7. Further recommendations for improving the study include utilizing augmented data for the Scikit Learn models, increasing the dataset size, and exploring other deep learning architectures such as recurrent neural networks (RNNs) and long short-term memory (LSTM) networks using the TensorFlow library. Overall, the development of a machine learning algorithm for bacterial enumeration has the potential to provide faster and more consistent methods for detecting bacterial contamination in various industries, ultimately contributing to improved public health and safety.
Processed image