Comparative Study of Machine Learning Methods on Spectroscopy Images for Blood Glucose Estimation

Tahsin Kazi[1], Kiran Ponakaladinne[1], Maria Valero[1(✉)], Liang Zhao[1], Hossain Shahriar[1], and Katherine H. Ingram[2]

[1] Department of Information Technology, Kennesaw State University, Kennesaw, GA 30060, USA
mvalero2@kennesaw.edu

[2] Department of Exercise Science and Sport Management, Kennesaw State University, Kennesaw, GA 30060, USA

Abstract. Diabetes and metabolic diseases are considered a silent epidemic in the United States. Monitoring blood glucose, the lead indicator of these diseases, involves either a cumbersome process of extracting blood several times per day or implanting needles under the skin. However, new technologies have emerged for non-invasive blood glucose monitoring, including light absorption and spectroscopy methods. In this paper, we performed a comparative study of diverse Machine Learning (ML) methods on spectroscopy images to estimate blood glucose concentration. We used a database of fingertip images from 45 human subjects and trained several ML methods based on image tensors, color intensity, and statistical image information. We determined that for spectroscopy images, AdaBoost trained with KNeigbors is the best model to estimate blood glucose with a percentage of 90.78% of results in zone "A" (accurate) and 9.22% in zone "B" (clinically acceptable) according to the Clarke Error Grid metric.

Keywords: Non-invasive monitoring · spectroscopy · machine learning · blood glucose concentration

1 Introduction

The USA is facing an epidemic of diabetes [26], with more than 11.33% of the population affected by diabetes alone [7] and another 30% by metabolic syndrome [16]. The lead indicator of these diseases is the blood glucose (BG) concentration. Measuring blood glucose typically involves either painful blood extraction multiple times per day or implanting needles under the skin for continuous monitoring.

Supported by College of Computing and Software Engineering and Office of Research at Kennesaw State University.

A. Tsanas and A. Triantafyllidis (Eds.): PH 2022, LNICST 488, pp. 60–74, 2023.
https://doi.org/10.1007/978-3-031-34586-9_5

Non-invasive estimation of blood glucose levels has emerged as an exciting alternative for the monitoring and management of metabolic diseases [33]. Among those, technologies that use optical approaches are both practical and inexpensive [2,13,23,31]. Optical methods function by directing a light beam through human tissue, and the energy absorption, reflection, or scattering is used to estimate blood glucose concentration [20]. Optical methods are portable and can be easily applied to fingers and other extremities such as earlobes. Several optical methods for detecting blood glucose have been developed, though most of these have limitations that restrict their utility. They include: (1) fluorescence spectroscopy [19], which may result in harmful exposure to fluorophore [12]; (2) Raman spectroscopy [10] criticized for its lengthy spectral acquisition time and poor signal-to-noise ratio [32]; (3) photoacoustic spectroscopy [18], which introduces noise from its sensitivity to environmental factors [32]; (4) optical coherence tomography [11], which is overly sensitive to skin temperature; and (5) occlusion spectroscopy [4], known to result in signal drift [27]. An alternative optical method (6) near-infrared absorption spectroscopy, avoids these limitations and is both more practical and cost-efficient than those described above [2,13,15,21,23,31]. In addition, near-infrared absorption spectroscopy is fundamentally simple to use in the creation of a powerful sensor prototype. Just a laser light and camera are needed.

In our previous work [31], we designed and tested a non-invasive sensor prototype for estimating blood glucose based on near-infrared absorption spectroscopy. A device composed of laser light, a raspberry Pi, and a camera was used to collect fingertip images, as shown in Fig. 1. These images were used to estimate blood glucose by applying a Machine Learning (ML) model, specifically a Convolutional Neural Network (CNN). The model used light absorption data from the images to approximate a function for estimating blood glucose concentration. However, the initial accuracy only reached 72% with the limited dataset and that particular CNN model. Therefore, more studies were needed to determine a suitable ML method that provides higher accuracy for blood glucose estimation.

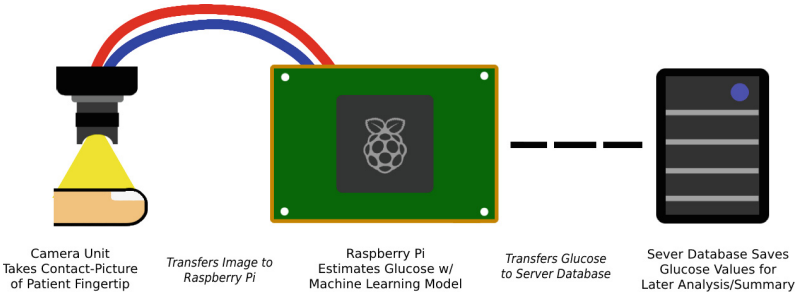

Fig. 1. Demonstration of the Blood Glucose Measuring System

This paper presents a comparative study of diverse ML methods trained with spectroscopy image data to identify the best model for estimating blood glucose. Metrics including mean absolute error (MAE), root mean square error (RMSE), and Clark error grids were used to determine accuracy. We further discuss the data preprocessing methods used to feed diverse ML models with the same dataset.

2 Background

In this section, we provide a background of the neural networks and linear statistical models applied to the spectroscopy image data obtained by our prototype [31], the CNN models VGG16 [30] and MobileNetV2 [28] were used to determine blood glucose concentration. On the other side, several linear models were employed, including Random Forest [9], Support Vector Machine [22], Bayesian Ridge [24], XGBoost [6], AdaBoost Ensemble [1], Histogram Gradient Boosting [5], Elastic Net [34], and KNeighbors. To apply these methods to the spectroscopy images, we used data transformation techniques to create new suitable databases for each method (Sect. 3). CNN models can only be used on tensor data because the algorithms are based on Linear Algebra that are suitable only for use with multi-dimensional matrices (tensors). Linear models are suitable for all scalar data and use a wide variety of statistical techniques to approximate the function of the data.

2.1 CNN Models

CNN models work by passing filters through images (represented as tensors) to extract features such as edges, shapes, and colors. These two-dimensional features are then flattened and mapped as scalar data which is then processed through normal neural network layers [3]. CNN models can use different types of filters through images of varying sizes, providing a wide range of applications. Since spectroscopy images were used, we applied CNN to determine blood glucose concentration using VGG16 and MobileNetV2.

VGG16 Neural Network: VGG-16 is a 16 layered deep CNN. A pretrained version of the network can be loaded which is trained on more than a million images from the ImageNet database [25]. The pretrained network can classify images into 1000 object categories. As a result, the network has learned rich feature representations for a wide range of images. However, the model was changed to output a single numeric value (blood glucose) instead of the 1000 categories it was trained on. The network has an image input size of 224-by-224, however the model's input size was changed to fit 160-by-120. VGG16 is well-suited for this project due to its ability to detect many different features and patterns as well as its performance when compared to other models [30]. An example of the VGG-16 architecture can be found in Fig. 2 (Taken from [29]).

Fig. 2. Demonstration of VGG-16 Model Architecture, from [29].

MobileNetV2: MobileNetV2 is a mobile architecture that enhances the state-of-the-art performance of mobile models across various model sizes, tasks, and benchmarks. In contrast to conventional residual models, which use expanded representations for the input, the MobileNetV2 architecture is based on an inverted residual structure, where the input and output of the residual block are thin bottleneck layers. Although the architecture of this model is more complex than most other CNN models, it performs well considering its computational power. Therefore, it was possible to train MobileNetV2 normally instead of using a pre-trained model version. MobileNetV2 was chosen for this study because of its low computational power usage, fast training times, and high-performance [28].

2.2 Linear Statistical Models

Linear models are a staple of machine learning and statistical modeling due to the countless algorithms available for function approximation, decision making, regression, classification, clustering, and prediction. Like CNNs, linear models were also chosen due to their wide range of applications and their superior performance. They are significantly faster and less computationally intensive than neural networks, but they can provide similar or better results in many instances. Many of the linear models used in this study applied bagging, boosting, or ensemble learning techniques, which allow for higher performance, lower error, and more optimized training. We propose a mix of models using these techniques to determine the most effective for estimating blood glucose.

Random Forest Regressor (RFR). Random Forest is a supervised learning algorithm built on Decision Trees and the Ensemble Learning Approach [35]. Decision Trees are tree-diagrams of statistical decisions that lead users to a specific outcome, result, or prediction. Random Forest uses an optimized approach to ensemble learning called bagging (bootstrap-aggregating), which works like this: the model creates multiple decision trees that train on random segments of the training data, these trees are then used in unison to predict unknown values. Random Forest was chosen for this study for its novel combination of Decision Trees and bagging, and its high performance in many domains [9].

Support Vector Regressor (SVR). Support Vector Regression [22] works on the principle of the Support Vector Machine (SVM) [17]. This model is based on

simple regression algorithms, to fit a line/curve/plane through the data to create an approximate function. In simple regression, the goal is to minimize the error rate while in SVR it is to fit the error inside a certain threshold. The flexibility of SVR allows to decide how much error is acceptable in the model, and it will find an appropriate line (or curve or plane) to fit the data accordingly. This technique was included for its ability to reduce overfitting and handle outliers in data. It is a well-performing and versatile model.

Bayesian Ridge Regressor (BRR). Ridge Regression is a classical regularization technique widely used in Statistics and ML [24]. Bayesian regression allows a natural mechanism to survive insufficient or poorly distributed data by generalizing the data, which significantly reduces overfitting and handles outliers. In addition, this model outputs with a probability distribution, which means that it outputs multiple predicted values and chooses the most likely value. This method was used in this study because it performs well regardless of data quality.

XGBoost Regressor (XGB). XGBoost uses gradient boosting, an ensemble learning using boosting. It trains multiple decision trees to create an ensemble learner and it relies on the intuition that the best possible next model, combined with previous models, minimizes the overall prediction error. Through combining multiple models training, the model achieves high performance, even in cases where insufficient data and outliers exist. Extreme Gradient Boosting (XGBoost) is an efficient open-source implementation of this gradient boosting algorithm [6]. The two benefits of using XGBoost are training speed and model performance, which is why it is chosen for this study.

Histogram Gradient Boosting Regressor (HGB). Histogram-based gradient boosting is an algorithm that uses the same gradient boosting as XGBoost, but instead of outputting a single value for blood glucose, it employs binning. Binning is a technique that converts continuous values into categories, similar to those used in classification scenarios [5]. By converting regression values to classification values, it can dramatically increase training speed and reduce the amount of memory used. Due to this, it is a much faster and lighter alternative to the XGBoost algorithm, which is why it is chosen for this study.

AdaBoost Ensemble Regressor (ABR). An AdaBoost regressor is a meta-estimator that begins by fitting another model on the original dataset and then fits additional copies of that model on the same dataset, but where the weights of instances are adjusted according to the error of the current prediction [1]. It creates more versions of the same model to tackle different sections of the training data, reducing error overall. Due to the large number of varying estimators that AdaBoost creates, the model is much less prone to overfitting than other models. The model we chose to train AdaBoost with is KNeighbors, which is described below.

KNeighbors Regressor (KNN). K-Nearest Neighbors (KNN) classifies a data point based on its nearest neighbors in the graph [14]. This algorithm is a non-parametric supervised learning method used for classification and regression.

In regression cases, the model takes the output value from a specific number of its nearest neighbors in the data, averages those values, and outputs that average. This algorithm does not make assumptions, so it handles outliers and minimizes error much better than decision trees and linear regression in many cases. This model was chosen for this study due to its novel approach to ensemble learning, high training speed, and high performance.

Elastic Net Regressor (ENR). Elastic Net is a regularized regression model that combines l1 and l2 penalties, i.e., lasso and ridge regression [34]. By combining both penalties, this model dramatically reduces overfitting. However, this model also performs feature selection, removing unnecessary features from the data. It was selected for this study because of its novel use of penalties and feature selection.

3 Datasets

3.1 Dataset of Spectroscopy Images

For this study, we used the non-invasive blood glucose monitor prototype ("Glucocheck") presented in our previous work [31]. Images were chosen instead of other forms of spectroscopy measurement, such as light intensity and PPG signals, because image capture is more replicable, accessible, and faster than other methods of spectroscopy data collection. Spectroscopy images were collected from the fingers of 43 participants between 18–65 years old. Two sets of 15 images were collected per participant. The first set was collected in a low-glucose fasting state, while the second set was collected one hour following a meal. Blood glucose was determined via finger prick using a commercial glucometer (FORA 6 Connect BG50) per manufacturer instructions. A set of 4 images is presented in Fig. 3. The images were taken after the finger prick at seconds 8 (top left), 16 (top right), 24 (bottom left), and 32 (bottom right). All images were collected from fingertips in the same format. A 640×480 resolution was chosen to preserve small details without sacrificing computing time and resources. The standard RGB color format was used. After removing any unclear images, the final dataset consisted of 1128 samples, each with two features, the image, and the corresponding blood glucose value.

3.2 Data Collection Ethics

The study was approved by the Institutional Review Board at Kennesaw State University (IRB-FY22-318). All participants provided written consent before participating.

3.3 Modified Datasets for CNN and Linear Models

Data transformation techniques were applied to the original data to generate three datasets, as described below.

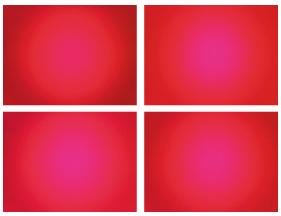

Fig. 3. Example of finger spectroscopy images collected from one individual.

Image Tensor Dataset. The "Tensor Dataset" was created in order to train the CNN models (VGG16 and MobileNetV2). Tensors are multi-dimensional matrices of numbers used in linear algebra; however, their application extends to images since images are multi-dimensional matrices of numbers as well. An image matrix consists of three dimensions: height, width, and color (red, green, and blue).

To convert an image into a tensor, a three-dimensional matrix (tensor) is created with the resolution of the image and the color format. Since images used in this study were 640×480 using the RGB color format, the image tensor was 640 pixels by 480 pixels by three colors. Then each color value for each pixel was entered into each value in the tensor, obtaining a tensor of 921,600 values. The resulting image tensor dataset was maintained at $160 \times 120 \times 3$ pixels to decrease computational time and necessary resources, when compared to a 640×480 dataset. The final dataset included the tensors with their corresponding blood glucose value. A visual demonstration of the image-tensor conversion can be seen in Fig. 4.

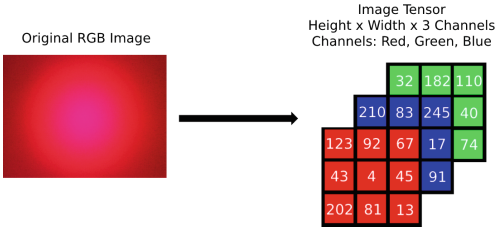

Fig. 4. Demonstration of Image Tensor Conversion (Color figure online)

CNN models are the only ones that can be trained with tensors because they use filtering techniques to analyze and process them. These filtering techniques are not available in other machine learning algorithms, which is why we used the two CNN models MobileNetV2 and VGG16.

Color Intensity Datasets. We have created four datasets based on extracting color intensity from the original images. For each possible value of red, green,

and blue (0–255), the number of pixels with that same value in an image can be counted and recorded in a histogram [2]. Through this process, a histogram with RGB values on the x-axis (256 possible values for red, green, and blue) and the number of pixels on the y-axis can be created, as shown by Fig. 5. This process of counting pixel-intensity values for each color was used to create three datasets: "Red Intensity", "Blue Intensity", and "Green Intensity". Each dataset consists of 257 features: 256 features for each possible value of that color and one feature for the blood glucose value of that image. Lastly, a final dataset, named "RGB-Intensity", was created by combining the intensity values for all three-color channels. The RGB-Intensity dataset consisted of 769 features: 256 values of red, 256 values of green, 256 values of blue, and one value for blood glucose [2].

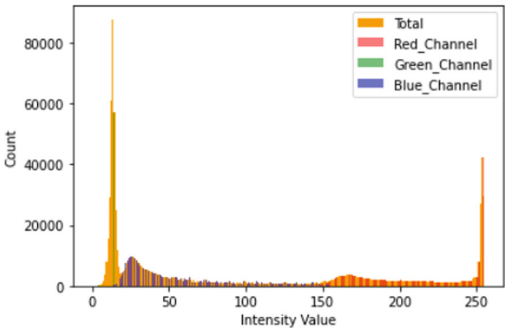

Fig. 5. Histogram of RGB Intensity Values in an Image. (Color figure online)

Image Measurement Datasets. The last five datasets were created by extracting measurement data from the images. To create the dataset, each image in the dataset was split into four channels: red, green, blue, and grayscale (the image with color removed). Then, for each color channel, the channel's pixel center of mass, minimum, maximum, mean, median, standard deviation, and variance were calculated. To calculate these values: the images are first converted into numerical tensors, then their tensors (3-dimensional matrix) are converted into an array for each channel, and then each channel array (1-dimensional list) is used for calculations such as mean, median, minimum, maximum, etc. A demonstration of this process can be seen in Fig. 6.

Values for each channel were compiled into the same dataset with the correct blood glucose value and repeated for every image. The resulting "Measurement Dataset", consisted of 29 features: seven measurements for each of the four channels and one feature for the blood glucose. After the creation of this dataset, four new datasets were created by merging the measurement features of each image with the intensity values of the same image created in the previously mentioned intensity datasets. This process resulted in four new datasets:

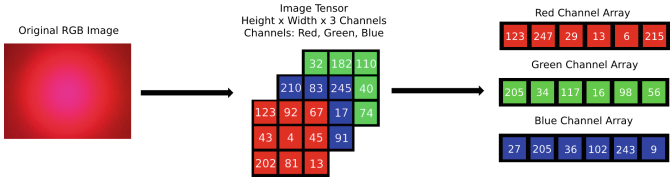

Fig. 6. Demonstration of Measurement Dataset Creation (Color figure online)

"Red-Measurement", "Green-Measurement", "Blue-Measurement", and "RGB-Measurement". The first three new datasets contained 285 features: 256 for the pixel intensities, 28 for the measurement features, and one for the blood glucose value. The last new dataset contained 797 features: 256 for each color channel, 28 measurement features, and one for the blood glucose value.

4 Experiment

After creating the datasets, each model was trained, tuned, and tested to each dataset to compare results, with only two exceptions. Since they can only be trained on tensor data, VGG16 and MobileNetV2 were only trained on the Tensor Dataset. Furthermore, the other linear models can only be trained on scalar data, so they were trained on every dataset except for the Tensor Dataset. The CNN models were trained using image data generators, which come with the TensorFlow library for Python that was used for training models. Moreover, before the training process, the image data generators were used to scale down the pixel values from 0–255 to 0–1 to reduce error and GPU usage. Besides these changes during training, the testing of CNN models was the same as the other models. On another note, since the AdaBoost Ensemble Learning algorithm uses another algorithm as a base estimator, for each dataset, the AdaBoost model was trained with the model that had the highest accuracy for that dataset. A summary of the models trained with each dataset can be seen in Table 1.

Table 1. Models Trained on Each Dataset

Tensor Dataset	Intensity Dataset	Measurement Dataset	Intensity-Measurement Dataset
VGG16	Random Forest	Random Forest	Random Forest
MobileNetV2	Support Vector	Support Vector	Support Vector
	Bayesian Ridge	Bayesian Ridge	Bayesian Ridge
	XGBoost	XGBoost	XGBoost
	HGB	HGB	HGB
	AdaBoost	AdaBoost	AdaBoost
	KNeighbors	KNeighbors	KNeighbors
	Elastic Net	Elastic Net	Elastic Net

4.1 Training and Hyperparameter Tuning of Models

To train the models, all of the datasets were split into training/testing splits where the training data was used to fit the model, and the testing data was used to measure the model's performance. The training/testing split ratio was 75:25 to ensure sufficient data to train the models and ensure that they would not overfit. After creating training/testing splits, each model in the set was fitted to training data and then tested. However, to ensure that the models were compared effectively, each model's hyperparameters were tuned to each specific dataset to minimize error and overfitting. After the models were finished with training and tuning, they were tested, and the results were recorded in a table.

4.2 Testing Models

Three distinct metrics were considered for testing/tuning the models: MAE, RMSE, and the Clarke Error Grid. The MAE is the mean of all errors between the blood glucose values that a model predicts and the actual blood glucose value tied to an image. The RMSE is the root of the mean of each error squared. MAE is a more direct metric for calculating error as it is unbiased towards all errors and treated as an average. However, because it squares the errors, RMSE is biased against large prediction errors, making it weighted against outliers. RMSE is usually used in scenarios when an increase in error is disproportionate to the effect, for example, if the error increases from 5 to 10 and the effect is four times as bad. Since RMSE is always higher or equal to MAE, the difference between the two values is critical for evaluating outliers. If RMSE is significantly higher than MAE, then there are outliers in the predictions. For this reason, RMSE was used to tune the models to reduce overfitting but not recorded in the results or evaluation. Lastly, Clarke Error Grids were used to evaluate models since they have been widely used for several decades to evaluate the performance of blood glucose meters. Clarke Error Grids are scatterplots with predicted blood glucose values on the y-axis and actual blood glucose values on the x-axis. The

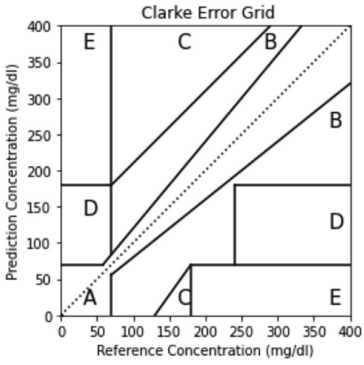

Fig. 7. Clarke Error Grid

grid is split into several zones, and each zone signifies a level of risk of a negative outcome due to the measurement error in blood glucose values which can be seen in Fig. 7.

There are 5 zones: A - Accurate, B - Clinically Acceptable, C - Overcorrection, D - Failure to Detect/Treat, and E - Erroneous Treatment [8]. These three metrics were all used when measuring the performance of the models during training and testing.

5 Evaluation

For comparing the performance of the models we used MAE and Clarke Error Grid (Zone A Percentage) metrics. The percentage of data points that fall into each zone of the clinical outcome can be determined by analyzing the grid. To get Zone A Percentage, the number of predictions in Zone A (Clinically Accurate) is recorded as a percentage of the total number of predictions made. After the models were trained and tuned, they were tested with the testing data, and the results were recorded in Table 2 and Table 3 respectively.

Table 2. Model Testing Results from Tensor and Intensity Datasets - MAE and Zone A percentages from clarke error grid analysis

	Image Tensor (IT)	Red-Intensity (RI)	Green-Intensity (GI)	Blue-Intensity (BI)	RGB-Intensity (RGBI)
VGG16	16.58–87.59%	–	–	–	–
MobileNetV2	15.68–87.23%	–	–	–	–
Random Forest	–	13.17–86.17%	13.31–85.11%	14.04–86.17%	12.46–88.65%
Elastic Net	–	15.59–85.46%	16.23–82.27%	15.53–84.4%	14.42–84.4%
KNeighbors	–	9.88–90.78%	14.06–88.3%	14.35–85.46%	10.84–88.65%
Support Vector	–	14.43–89.36%	15.71–89.36%	14.3–89.36%	13.14–88.65%
Bayesian Ridge	–	15.43–85.11%	16.01–83.33%	15.34–83.33%	14.28–84.4%
XGBoost	–	12.93–87.94%	14.1–84.75%	13.97–84.75%	12.26–89.72%
HGB	–	13.12–86.88%	14.99–84.04%	14.37–83.69%	12.53–87.59%
AdaBoost	–	9.66–90.78%	13.31–87.94%	14.08–85.46%	10.95–88.65%

Table 3. Model Testing Results from Measurement Datasets - MAE and Zone A percentages from clarke error grid analysis

	Measurement (ME)	Red-Measurement (RM)	Green-Measurement (GM)	Blue-Measurement (BM)	RGB-Measurement (RGBM)
Random Forest	14.27–83.33%	12.85–87.23%	12.63–86.52%	13.91–85.82%	12.74–88.65%
Elastic Net	16.38–81.56%	15.68–84.04%	16.89–85.11%	15.55–81.56%	14.41–83.69%
KNeighbors	15.02–81.91%	9.55–90.78%	14.3–86.17%	15.81–84.4%	12.43–87.59%
Support Vector	16.13–89.72%	14.3–87.94%	15.02–89.01%	14.58–87.23%	13.28–87.94%
Bayesian Ridge	16.37–81.56%	15.52–84.04%	17.43–85.46%	15.52–82.62%	14.3–83.33%
XGBoost	14.51–83.69%	13.03–86.88%	12.86–88.3%	13.6–86.52%	12.89–87.59%
HGB	14.78–81.21%	13.71–85.11%	13.17–86.17%	13.7–85.11%	12.58–87.94%
AdaBoost	15.13–80.5%	9.4–90.78%	12.74–86.88%	13.41–87.59%	13.18–86.52%

6 Discussion

AdaBoost with KNeighbors trained on the Red-Measurement dataset provided the most accurate estimates of blood glucose among all of the dataset-models tested. This dataset-model combination had an MAE of 9.4 mg/dl, an RMSE of 16.72 mg/dl, and a Clarke Error Grid Zone A Percentage of 90.78% illustrated in Fig. 8.

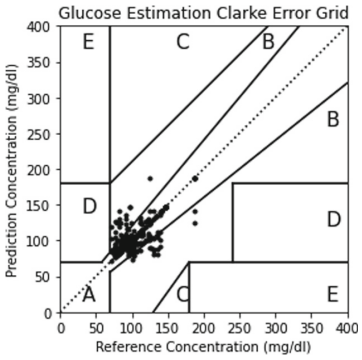

Fig. 8. Clarke Error Grid of AdaBoost model with KNeighbors Trained on Red-Intensity Dataset

From best to worst, the models ranked AdaBoost, KNeighbors, Random Forest, XGBoost, HGB, Support Vector, Bayesian Ridge, Elastic Net, MobileNetV2, and VGG16 as displayed in Fig. 9. From best to worst, the datasets ranked RGB Intensity, Red Measurement, Red Intensity, RGB Measurement, Green Measurement, Blue Intensity, Blue Measurement, Green Intensity, Measurement, and Image Tensor as shown in Fig. 10. The datasets containing Red and RGB data outperformed the other datasets by a large margin. However, combining measurement and intensity values did not seem to improve performance for the red dataset, but instead hindered it. Datasets with Blue and Green data appeared to perform equally, but their performance was inferior to the Red and the combined RGB overall. Furthermore, the Green data, but not the blue, seemed to perform better after combining intensity and measurement data. The intensity datasets performed better than the measurement datasets, and the dataset with only measurement values performed significantly worse. The image tensor dataset performed the worst of all datasets, while the CNN models performed the worst among the group of models. AdaBoost and KNeighbors performed the best with every dataset they were trained on, while XGBoost, Random Forest, and HGB generally outperformed the other models. These results suggest that the best data for blood glucose estimation by spectroscopy is color intensity data focused on either the red channel or all three channels. The results further suggest that the KNeighbors algorithm is well-suited for blood glucose estimation with scalar

data, and using AdaBoost as an ensemble learner can boost performance. Models that use boosting and bagging (XGBoost, AdaBoost, HGB, etc.) outperformed models that do not (Elastic Net, Bayesian Ridge, Support Vector). Furthermore, the penalties and feature selection in Elastic Net and the binning in Histogram-Based Gradient Boosting did not seem to increase performance compared to bagging and boosting. Finally, both the dataset and model results suggest that Convolutional Neural Networks and Tensor datasets perform worse than Linear Models, Ensemble Learners, and Scalar Data.

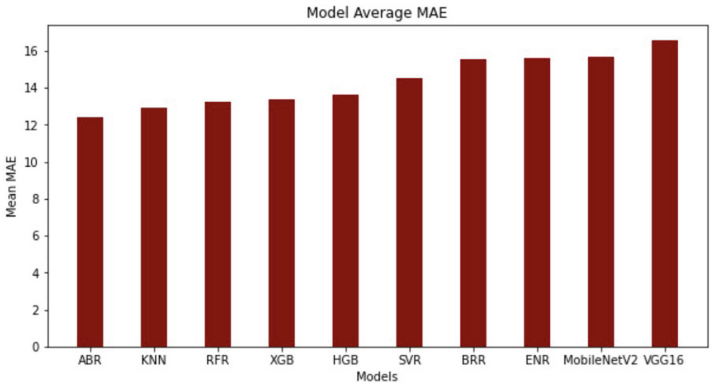

Fig. 9. Model Average MAE

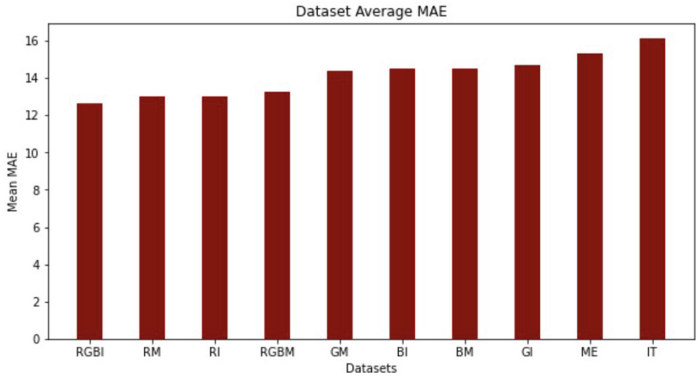

Fig. 10. Dataset Average MAE

7 Conclusion

From training, tuning, and testing ten machine learning models on ten different datasets, we have determined that the best model for estimating blood glucose

through spectroscopy images is AdaBoost trained with KNeighbors. Furthermore, the best image data to train the model is color intensity data collected from the red channel. Our highest performing dataset and model recorded a final Mean Absolute Error of 9.4, a Root Mean Squared Error of 16.72, and a Clark Error Grid Zone A Percentage of 90.78%. We also showed that intensity data outperformed measurement and tensor data, while the red and RGB channels outperformed all other color channels. Furthermore, models that utilize bagging and boosting outperformed those which did not, while linear models outperformed CNN models, regardless of their support for bagging or boosting.

References

1. Sklearn.ensemble.adaboostregressor. https://scikit-learn.org
2. Alarcón-Paredes, A., Francisco-García, V., Guzmán-Guzmán, I.P., Cantillo-Negrete, J., Cuevas-Valencia, R.E., Alonso-Silverio, G.A.: An IoT-based non-invasive glucose level monitoring system using Raspberry Pi. Appl. Sci. **9**(15), 3046 (2019). https://www.mdpi.com/2076-3417/9/15/3046/htm
3. Albawi, S., Mohammed, T.A., Al-Zawi, S.: Understanding of a convolutional neural network. In: 2017 International Conference on Engineering and Technology (ICET), pp. 1–6. IEEE (2017)
4. Amir, O., et al.: Continuous noninvasive glucose monitoring technology based on "occlusion spectroscopy" (2007)
5. Brownlee, J.: Histogram-based gradient boosting ensembles in Python (2021). https://machinelearningmastery.com/
6. Brownlee, J.: XGBoost for regression (2021). https://machinelearningmastery.com/xgboost-for-regression/
7. Centers for Disease Control and Prevention (CDC): National Diabetes Statistics Report website (2018). https://www.cdc.gov/diabetes/data/statistics-report/index.html. Accessed 2022
8. Clarke, W.L., Cox, D., Gonder-Frederick, L.A., Carter, W., Pohl, S.L.: Evaluating clinical accuracy of systems for self-monitoring of blood glucose (1987). https://doi.org/10.2337/diacare.10.5.622
9. Donges, N.: Random forest algorithm: a complete guide. https://builtin.com/data-science/random-forest-algorithm
10. Enejder, A.M., et al.: Raman spectroscopy for noninvasive glucose measurements. J. Biomed. Opt. **10**(3), 031114 (2005)
11. Haxha, S., Jhoja, J.: Optical based noninvasive glucose monitoring sensor prototype. IEEE Photonics J. **8**(6), 1–11 (2016)
12. Hull, E.L., et al.: Noninvasive skin fluorescence spectroscopy for detection of abnormal glucose tolerance. J. Clin. Transl. Endocrinol. **1**(3), 92–99 (2014)
13. Kasahara, R., Kino, S., Soyama, S., Matsuura, Y.: Noninvasive glucose monitoring using mid-infrared absorption spectroscopy based on a few wavenumbers. Biomed. Opt. Express **9**(1), 289–302 (2018)
14. Kramer, O.: K-nearest neighbors. In: Kramer, O. (ed.) Dimensionality Reduction with Unsupervised Nearest Neighbors, pp. 13–23. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-38652-7_2
15. Maruo, K., et al.: Noninvasive blood glucose assay using a newly developed near-infrared system. IEEE J. Sel. Top. Quantum Electron. **9**(2), 322–330 (2003)

16. Moore, J.X., Chaudhary, N., Akinyemiju, T.: Peer reviewed: metabolic syndrome prevalence by race/ethnicity and sex in the United States, National Health and Nutrition Examination Survey, 1988–2012. Preventing Chronic Dis. **14** (2017)
17. Noble, W.S.: What is a support vector machine? Nat. Biotechnol. **24**(12), 1565–1567 (2006)
18. Pai, P.P., Sanki, P.K., Sahoo, S.K., De, A., Bhattacharya, S., Banerjee, S.: Cloud computing-based non-invasive glucose monitoring for diabetic care. IEEE Trans. Circuits Syst. I Regul. Pap. **65**(2), 663–676 (2017)
19. Pickup, J.C., Khan, F., Zhi, Z.L., Coulter, J., Birch, D.J.: Fluorescence intensity- and lifetime-based glucose sensing using glucose/galactose-binding protein. J. Diab. Sci. Technol. **7**(1), 62–71 (2013)
20. Pitzer, K.R., et al.: Detection of hypoglycemia with the GlucoWatch biographer. Clin. Diabetol. **2**(4), 307–314 (2001)
21. Rachim, V.P., Chung, W.Y.: Wearable-band type visible-near infrared optical biosensor for non-invasive blood glucose monitoring. Sens. Actuators B Chem. **286**, 173–180 (2019)
22. Raj, A.: Unlocking the true power of support vector regression (2020)
23. Robinson, M.R., et al.: Noninvasive glucose monitoring in diabetic patients: a preliminary evaluation. Clin. Chem. **38**(9), 1618–1622 (1992)
24. Rothman, A.: The Bayesian paradigm & ridge regression (2020). https://towardsdatascience.com
25. Russakovsky, O., et al.: ImageNet large scale visual recognition challenge. Int. J. Comput. Vis. (IJCV) **115**(3), 211–252 (2015). https://doi.org/10.1007/s11263-015-0816-y
26. Saklayen, M.G.: The global epidemic of the metabolic syndrome. Curr. Hypertens. Rep. **20**(2), 1–8 (2018)
27. Sakr, M.A., Serry, M.: Non-enzymatic graphene-based biosensors for continous glucose monitoring. In: 2015 IEEE SENSORS, pp. 1–4. IEEE (2015)
28. Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., Chen, L.C.: MobileNetv2: inverted residuals and linear bottlenecks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 4510–4520 (2018)
29. Shi, B., et al.: Learning better deep features for the prediction of occult invasive disease in ductal carcinoma in situ through transfer learning, p. 98 (2018). https://doi.org/10.1117/12.2293594
30. Tammina, S.: Transfer learning using VGG-16 with deep convolutional neural network for classifying images. Int. J. Sci. Res. Publ. (IJSRP) **9**(10), 143–150 (2019)
31. Valero, M., et al.: Development of a non-invasive blood glucose monitoring system prototype: pilot study. J. Med. Internet Res. JMIR Formative Res. (forthcoming/in press)
32. Vashist, S.K.: Non-invasive glucose monitoring technology in diabetes management: a review. Anal. Chim. Acta **750**, 16–27 (2012)
33. Vegesna, A., Tran, M., Angelaccio, M., Arcona, S.: Remote patient monitoring via non-invasive digital technologies: a systematic review. Telemed. e-Health **23**(1), 3–17 (2017)
34. Verma, Y.: Hands-on tutorial on elasticnet regression (2021). https://analyticsindiamag.com/hands-on-tutorial-on-elasticnet-regression/
35. Dong, X., Yu, Z., Cao, W., Shi, Y., Ma, Q.: A survey on ensemble learning. Fron. Comput. Sci. **14**, 241–258 (2020). https://doi.org/10.1007/s11704-019-8208-z