On-Device Prediction for Chronic Kidney Disease

Alex Whelan
Computer Science and Engineering
Santa Clara University
Santa Clara, USA
awhelan@scu.edu

Soham Phadke
Computer Science and Engineering
Santa Clara University
Santa Clara, USA
smphadke@scu.edu

Alessandro Bellofiore
Biomedical Engineering
San José State University
San Jose, USA
alessandro.bellofiore@sjsu.edu

David C. Anastasiu
Computer Science and Engineering
Santa Clara University
Santa Clara, USA
danastasiu@scu.edu

***Abstract*—The number of people diagnosed with advanced stages of kidney disease has been rising every year. Although early diagnosis and treatment can slow, if not stop, the progression of the disease, many lower income individuals are unable to afford the high cost of frequent testing necessary to keep the disease progression at bay. To address this issue, we designed a kidney health monitoring system that allows for affordable and quick testing through the use of inexpensive test strips and a mobile application. Moreover, the application serves as a research framework for testing and improving detection models for the disease. In this paper, we describe the application we developed and several preliminary machine learning models we trained to classify the severity of the kidney disease as normal, intermediate risk, or kidney failure. We thoroughly evaluated the effectiveness of our models and found that our histogram of colors-based boosted tree method outperformed alternatives and exhibited good overall prediction performance (F1-score $> 90\%$).**

***Index Terms*—kidney health, test strip localization, experiment framework, machine learning, model improvement**

I. Introduction

More than 15% of US adults, or 37 million people, are estimated to have Chronic Kidney Disease (CKD) [1]. However, as many as 9 in 10 adults with CKD and 2 in 5 adults with severe CKD do not even know they have the disease [1] and may only find out when it is too late to slow, or stop, the progression of the disease. Part of the problem is that most tests for the disease involve laboratory testing of the patient's blood or urine, which is both expensive and not a common screening practice for most adults without other underlying symptoms. To alleviate this problem, with a goal of promoting global *good health and well-being*, we propose an inexpensive testing mechanism that can be used for initial screening of the disease.

Kidneys function as the filter of the human body, cleaning blood impurities and normal waste byproducts from day-to-day activities such as walking and breathing. When people develop CKD, their kidneys are no longer able to clean impurities from blood, which over time can lead to acute health problems such as high blood pressure, heart disease, stroke, and even death [1]. Early detection through frequent screening can lead people to take steps to protect their kidneys with the help of their health care providers.

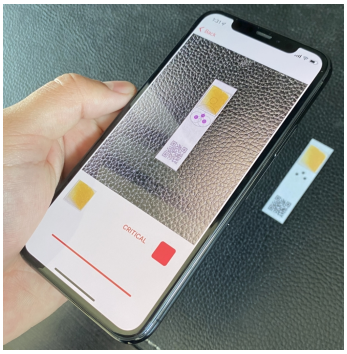

Fig. 1. CKD prediction using our application.

Two types of tests exist currently for CKD, based on either urine or blood. Urine tests check for protein buildup which may indicate kidney damage. Blood tests, check for the level of creatinine, which is a waste byproduct of normal muscle operations as they break down proteins. Our system decides the severity of CKD based on Jaffe's colorometric reaction between creatinine and picric acid in an alkaline solution [2] in a test strip. Similar to a blood glucose test, a subject would place a drop of blood on a test strip. After a few minutes, they would use the mobile phone application we developed, which is the subject of this article, to scan the test strip and find out the severity of their kidney disease. The application uses embedded computer vision and machine learning algorithms to identify the detection zone of the test strip, extract complex features from it, and classify the outcome. The machine learning models are able to associate small color variations in the test strip with CKD severity for a particular patient. The colors green, yellow, and red are used to indicate healthy kidneys, presence of kidney disease, and need for immediate medical attention, respectively. Fig. 1 shows an example test strip prediction using our application.

The application we developed further serves as a research

framework for testing and improving detection models for the disease. While in *user* mode, the application records different screenings for a single subject. In *research* mode, the application can be used to conduct multiple experiments, each using different criteria or prediction models, the results of which are synchronized to cloud storage for further analysis.

It is important to note from the start that this is a preliminary study meant to validate the design of our system. As such, the experiments we conducted did not involve human subjects and instead relied on creatinine solution in lieu of human or human-analog blood. Moreover, this article focuses on the application, computer vision algorithms, and preliminary machine learning models we developed for CKD prediction, while details of the test strip design and its chemical composition are being reported elsewhere. The main contributions of this paper are as follows:

- We describe a novel and inexpensive AI-based CKD prediction system which is essential in combating kidney disease and promoting *good health and well-being* in the global population.
- We present computer vision algorithms and machine learning models we developed to facilitate CKD prediction. Our models run directly on the device, in real time, eliminating the need for data transfer availability and making the application usable in remote areas.
- We describe a research framework that allows continuous development and improvement of prediction models using our application.
- We evaluate both the effectiveness and efficiency of our models and find that our preliminary models are quite effective, with F1-score results above 90%.

The remainder of the paper is organized as follows. Section II discusses research related to our application. Section III describes how data were collected for training and evaluating the models described herein, the design of our application, and algorithms behind identifying the test strip detection zone and predicting the severity of kidney disease. Section IV describes our evaluation methodology and presents experiment results, and Section V proposes avenues for future work and concludes the paper.

II. Related Works

While first introduced by Max Jaffe in 1886, the directly proportional relationship between the concentration of creatinine and the color of its mixture with picric acid in an alkaline solution was first formalized in a clinical setting by Otto Folin in his 1916 "Lab Manual of Biological Chemistry [2]". The reaction is the basis for most clinical and point-of-care (PoC) testing systems for CKD, but the cost of most such systems is prohibitive for most people. For example, three such systems that rely on enzymatic reactions for the detection, namely StatSensor [3], ABL 800 Flex [4], and iSTAT [5], cost between $4,000 and $44,000, while test strips for the devices cost, on average, less than $1 per test strip. Our proposed system replaces the expensive sensing device with a *free* mobile phone application which, coupled with inexpensive test strips, can enable humanitarian CKD testing in all parts of the globe.

Machine learning plays an increasingly greater role in healthcare. It has been used to solve important medical problems, including Autism Spectrum prediction [6], cancer detection [7], and antibiofilm peptide identification [8], to name just a few. Additionally, the ubiquity of smart phones has attracted much interest in biomedical device research. SmartBioPhone [9], a European Union industrial collaborative project, aims to bring smart phone-based point-of-care testing using Lab-on-a-Chip (LOCs) devices to environment, food, cancer and drug monitoring applications. For example, Oncescu et al. [10] developed a smartphone-based cholesterol testing system that uses an attachment to ensure uniform and repeatable image acquisition. The attachment ensures the test strip detection zone is positioned right in front of the camera and the picture is taken in consistent lighting conditions. In contrast, our system automatically identifies the position of the detection zone without the aid of an attachment and the choice of extracted features ensures our model performs well in diverse lighting conditions.

The use of machine learning techniques in image processing is quite wide-spread. However, there are few works that attempt to associate small color variations with a prediction outcome, as we do in our models. A histogram of colors extracted from the RGB color space image representation has been successfully used to index images for image retrieval applications [11]. Color constancy has been shown to depend on statistics of outputs of linear [12] and non-linear filters [13], [14]. However, convolutional neural network-based methods have been proven to be much more effective than statistical methods in achieving color constancy [15]–[17], though the representation of colors in deep neural networks is still poorly understood [18].

One closely related system to the one we designed is the homemade-spectrometer by Debus et al. [19], which predicts continuous values for color change observed in reaction between urine and picric acid solution. Despite being a low cost system, the setup requires technical acumen and is time-consuming. An early experiment to classify temperature of thermal paints based on their color from a camera feed was conducted by Lalanne and Lempereur [20]. More recently, Paulraj et al. showed that a 5 hidden layer artificial neural network could predict the ripeness of bananas as ripe or not ripe with an accuracy of 96% [21].

III. Methods

A. Data Collection

The presence of kidney disease and its progression is usually determined using the estimated glomerular filtration rate (eGFR) scale, which determines kidney health based on creatinine concentration using the Modification of Diet in Renal Disease (MDRD) equation [22], defined as

$$eGFR = 175 \times S_{Cr}^{-1.154} \times \text{age}^{-0.203} \times 0.742 \text{ if female} \times 1.212 \text{ if African born}, \quad (1)$$

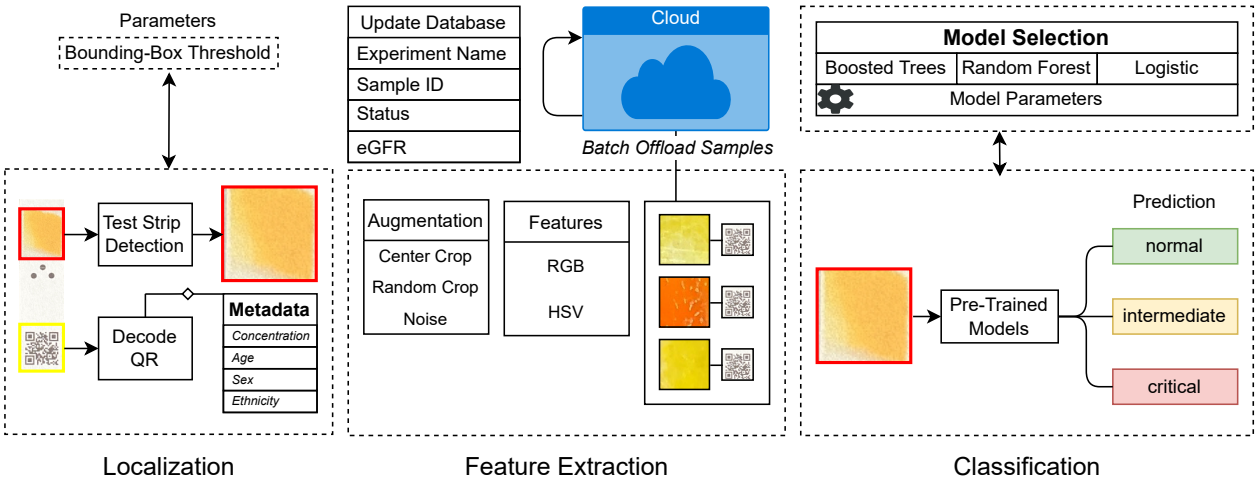

Fig. 2. Application workflow.

TABLE I
DISTRIBUTION OF TESTED CREATININE CONCENTRATIONS

Concentration range	Steps	No. of concentrations	No. of samples
0–3.9 mg/dL	0.1 mg/dL	40	160
4–7.5 mg/dL	0.5 mg/dL	8	32
8–19 mg/dL	1 mg/dL	12	48
20–60 mg/dL	10 mg/dL	5	20

TABLE II
DATASET STATISTICS

Dataset	Train	Test	Healthy	Int.	Critical
No Crop	780	260	47	104	109
Center Crop	780	260	52	100	108
Random Crop	3900	1300	242	523	535

where S_{Cr} is the creatinine concentration in the blood sample, measured in mg/dL. EGFR values between 0 and 15 are considered critical, between 15 and 60 are considered intermediate, while those above 60 indicate healthy kidneys. For a given patient, the only critical factor in the prediction is the amount of creatinine in their blood. In our experiments, we simulated creatinine extracted from blood by using a creatinine solution at different concentrations, which we applied to identical test strips. Given the inverse relationship of eGFR and creatinine concentration, we chose more samples from low creatinine concentration ranges, as shown in Table I.

We applied creatinine solution at each chosen concentration to 4 test strips and took pictures of the test strips, in a laboratory, at 4 different time intervals, namely 2, 12, 22, and 32 minutes after applying the solution. This is because, as time progresses, the saturation and intensity of the color on the test strip may also change. In this study, we only used pictures taken at 12 minutes after application, which we found most effective in our analysis. Finally, each experiment sample was associated with one or more patients drawn at random from a distribution with similar racial, gender, and age proportions as the U.S. population according to the 2010 census, which allowed identifying the ground-truth CKD category for samples based on the eGFR level computed using Equation 1.

The detection zone of the test strip is the area that has been imbibed with the appropriate chemistry to react to the creatinine solution. In order to train CKD classification models, which we describe in detail in Section III-D, we extracted image patches from the detection zone part of the experiment test strips under three augmentation conditions. For each experiment sample, we extracted a *no crop* sample that contained all pixels of the detection zone, a *center crop* sample containing a single 64×64-pixel patch centered at the center of the detection zone, and 5 *random crop* samples that were 64×64-pixel patches randomly positioned within the detection zone. Table II shows the statistics of our three augmentation datasets which we used in training our models. The first three columns show the dataset and numbers of training and test samples, respectively, and the last three columns show the number of test samples in each CKD category, namely healthy, intermediate, and critical. Training samples have similar distributions of labels in each category.

B. Application Design

Fig. 2 shows our application's workflow. In order to predict the severity of kidney disease, our application captures a frame using the phone's camera and must first localize the test strip within the image and the detection zone within the test strip. Then, features are extracted from the localized detection zone and fed into a pretrained classification model along with several additional features encoding the patient's age, gender, and race. Finally, the detection zone, along with the predicted CKD outcome and patient demographic data, are stored in a cloud database.

We designed our application to be used in two different modes of operation. In ***user mode***, the application uses the latest/best model to generate predictions and stores the user's readings over time. In ***research mode***, a scientist can run

one or more experiments using different meta-parameters (e.g., prediction model, augmentation technique, patch size, etc.), taking one or more sample readings for each experiment.

To simplify recording data for each sample and associating each sample with a patient, we encode patient and creatinine concentration information, along with other experiment parameters, in a QR code that is printed on the test strip. The scientist can then scan the QR code using the application, apply the solution with the prescribed creatinine concentration, wait the prescribed number of minutes, and then scan the test strip to complete the prediction. The application automatically uploads the detection zone image and prediction results to the cloud database. After all readings in an experiment are completed, results can be extracted from the cloud database and further analyzed or used to train additional prediction models.

1) Application Interface: We purposely designed our application to be easy to use, while at the same time secure. Fig. 3 shows the start and sign-up screens for our application. A user can sign-up for an account to use the application and the application will ensure the password they choose is a strong password. Moreover, if they choose to use the application in *user* mode, they are asked to provide the personal information necessary to perform CKD predictions for the user. These data are not necessary in *research* mode as they are encoded in the test strip QR codes.

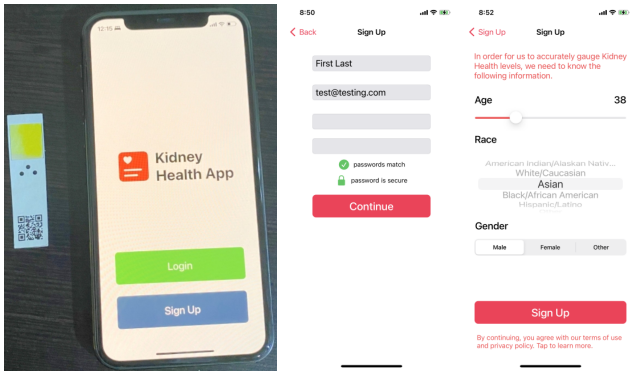

Fig. 3. Application login and sign-up screens.

In user mode, the application will simply show a summary panel of the user's readings, i.e., scans of test strips (Fig. 4 right). They can click on the + button to initiate a new reading. On the other hand, in research mode, the application will show the list of existing experiments (Fig. 4 left), and the user can click the + button to initiate a new experiment (Fig. 4 center). When starting a new experiment, the user must choose experiment parameters, such as which model will be used for predictions in the experiment. Selecting an existing experiment will show the list of current readings for that experiment and the user can then initiate a new reading using the + button in that screen, just as in the user mode.

The application was designed to simplify the prediction process. When initiating a new reading, the application uses the phone's camera to localize the test strip. Our localization algorithm, detailed in Section III-C, continually scans the

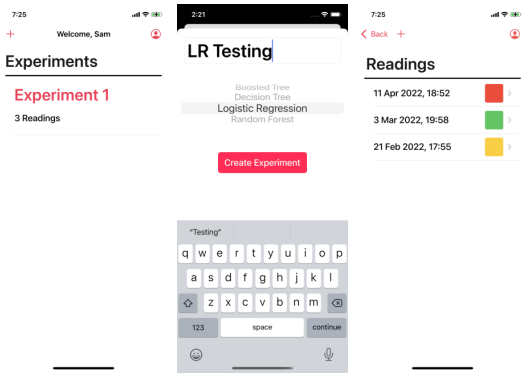

Fig. 4. Application experiment and readings screens.

frames provided by the camera until the test strip is identified and deemed to be in focus. Then, the image freezes and the user is shown the identified detection zone and asked to confirm whether prediction should proceed. Fig. 5 (left) shows this confirmation in action. While not strictly necessary, the image on the phone also overlays on the test strip artifacts that were used to identify the position of the detection zone. When the user clicks "Predict," the application engages the selected (or default/best) prediction algorithm to identify the severity of kidney disease. As soon as that prediction is complete, the detection zone image and results are saved locally and the result is shown to the user (Fig. 5 right). An asynchronous process then uploads new results to our cloud database once every 5 minutes, as long as a data connection is available.

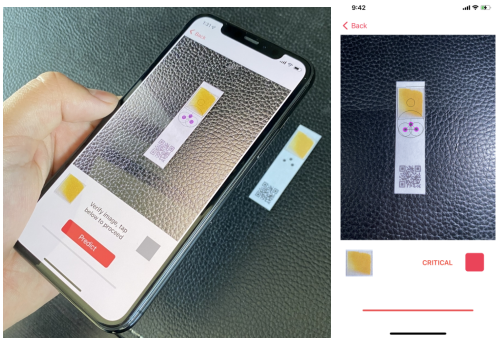

Fig. 5. Application CKD prediction screens.

C. Test Strip Localization

Fig. 6. Test strip

In order to allow the device camera to reliably and consistently identify the detection zone of the test strip, we added three black dots to the test strip that form an isosceles triangle, as seen in Fig. 6. This represents a unique identifier which serves as the starting point for the computer vision strip detection algorithm. The triangle is located right below the detection zone of the test strip. The length of the segments between the three dots and the angles formed by

the segments can be used to not only identify the presence of a test strip in the image, but also to localize the detection zone.

Algorithm 1 Test strip detection zone localization.

Input: I, θ ▷ Input image
Output: B ▷ Bounding box

1: $B \leftarrow \varnothing$
2: $C \leftarrow FindCircles(I)$
3: $D \leftarrow ||c_i - c_j||_2 \ \forall \ c_i, c_j \in C$ ▷ Distance matrix
4: **for all** $i, j, k \in \{1, 2, \ldots, |C|\}$ **do**
5: **if** $D[i, j] \sim D[j, k] \wedge \cos(j, i, k) \sim \theta$ **then**
6: **if** $Verify(i, j, k)$ **then** ▷ Box focus and color
7: $B \leftarrow B \cup Box(i, j, k)$
8: **end if**
9: **end if**
10: **end for**
11: **return** $\max_b B$ ▷ Non-maximum suppression

Algorithm 1 describes the procedure our application uses to identify where the test strip detection zone is located. The method first uses the Hough Circle Transform method [23], [24] from the OpenCV library to identify all circles in the image. Circles with large radius are removed and the centers of the remaining circles are added to C. After computing all pairwise distances between identified circle center points (line 3 in Algorithm 1), the method looks for roughly equidistant segments with a point in common, e.g., ij and jk, such that the cosine of the angle between them is roughly θ. If such an isosceles triangle is found, the method then computes the likely location of the bounding box, the center of which can be found along the height of the triangle at a distance of $3h$, where h is the triangle height from angle ijk. Given a potential detection zone location, the method verifies whether the retrieved pixels in the detection zone are in the correct color range by sampling several random pixels in the potential detection zone and checking that their red, green and blue components are within predefined thresholds. The method also ensures that the detection zone is in focus. If it is not, it instructs the device to focus at the perceived center of the detection zone and returns nothing, awaiting for another frame where the detection zone is in focus. Otherwise, the bounding box of the detection zone is added to set B (line 6 of Algorithm 1). Finally, given a non-empty set B, the algorithm returns the box b that maximizes a localization quality score via a non-maximum suppression algorithm.

An alternative to the proposed Computer Vision-based test strip localization approach would be to use deep learning object detection methods, such as YOLO [25], [26]. We tested such methods but found both their effectiveness and their efficiency sub-par compared to the proposed method. Our implementation took more than 0.5 seconds for the test strip localization alone on the phone we were testing with, making the approach prohibitive to real-time test strip localization and CKD prediction.

D. Kidney Health Prediction

Once the detection zone of the test strip has been localized, the pixels from that section of the image are processed to extract features suitable for solving the classification problem. In the remainder of this section, we discuss the types of features that we extracted and the machine learning models we used for classifying the test strips.

1) Feature Extraction: In order to obtain accurate predictions we must engineer meaningful features that have to account for possible variability in our data. Images are usually represented by a matrix or vector of the colors of each of the pixels in the image. However, the size of the detection zone bounding boxes returned by the localization model are non-uniform, meaning we must perform an additional post-processing step, possibly applying data-augmentation techniques, in order to extract the same number of features for each sample. Moreover, the color representation of each pixel plays a vital role in the ability of the features to capture the information needed to classify the severity of kidney disease. In our work, we tested two different feature extraction techniques, one based on the standard red-green-blue (RGB) color space, and the other based on the hue-saturation-value (HSV) color space.

a) RGB: In the RGB representation, we simply concatenate all red, green, and blue color values of each pixel of a 64×64-pixel region of the detection zone into a vector, which, after normalization, forms the input to our methods. Color values range between 0 and 255 and the input vector is normalized by dividing all of its components by 255. Along the pixel color features, we add 3 features for age, race, and gender, which are standardized individually across the samples. RGB feature vectors thus have $(64 \times 64 \times 3) + 3 = 12291$ dimensions.

b) HSV: Unlike the RGB color space, which captures the amount of red, green, and blue in a given color, the HSV color space models how colors appear under light. In other words, the specific color is captured by the H and S components of the space, and the V component simply indicates the amount of light, or vibrancy present in the color. This is beneficial in our case, as we wish to have the same representation of a color in different lighting conditions. Moreover, we observed that the color in the detection zone is not completely uniform, and non-detection zone pixels may be present around the edges of the extracted image patch. To rectify these problems, we represent the image as two concatenated histograms for the H and S color channels, respectively. For each color channel, we count the number of pixels that fall within equal-sized sections of a predefined color range for the channel. For example, assuming η bins and range $[H_s, H_e]$ for the H channel and σ bins and range $[S_s, S_e]$ for the S channel, we break the H range into η equal-sized bins and the S range into σ equal-sized bins, and

construct a count matrix,

$$C[i,j] = \sum_{x} \mathbb{1}x \qquad \textbf{s.t.} \tag{2}$$
$$x_H \in \left[i \times \frac{H_e - H_s}{\eta}, (i+1) \times \frac{H_e - H_s}{\eta} \right),$$
$$x_S \in \left[j \times \frac{S_e - S_s}{\sigma}, (j+1) \times \frac{S_e - S_s}{\sigma} \right),$$

where x is a pixel and x_H and x_S are its hue and saturation values, respectively. The HSV feature vector is constructed by concatenating the C matrix rows, after normalizing its values by the $L1$ norm, and appending the standardized age, race, and gender values for the patient assigned to the sample. Therefore, the HSV vector has $(\eta \times \sigma) + 3$ dimensions. The histogram approach provides several benefits over plain RGB features, including,

- Masked pixels — pixels that do not fall in the hue or saturation ranges are not considered (i.e., background),
- Orientation independent — the same sample produced from different camera orientations should return the same or very similar features which may facilitate in reducing error when using the application to run a batch of experiments,
- Detection zones do not have to be uniform therefore do not need to be cropped.

2) Machine Learning Models: The current version of our application was designed to work on Apple iOS devices. As a result, in order to ensure efficient real-time inference on the device, we chose to train classification models whose inference could be executed by the Apple ML library, which has been optimized for such devices. The models we included in our study include logistic regression, decision tree, random forest, and boosted trees. While decision tree and logistic regression learn a single classification function, random forest and boosted trees are ensemble methods that train multiple decision trees and use their collective wisdom to decide the predicted outcome.

IV. Evaluation

A. Experimental Design

For each combination of our two feature types (RGB and HSV) and three augmentation datasets (no crop, center crop, and random crop), we trained four standard machine learning classification models (logistic regression, decision tree, random forest, and boosted trees). We first split each dataset into a 70% training, 5% validation, and 25% test set using a stratified sampling approach, i.e., each set had a similar percent of samples from each category (normal, intermediate, or severe CKD) as in the overall dataset. Then, we tuned meta-parameters for each model by maximizing the F1-score on the validation set and used the final best model of each type to predict samples in the test set. In all, we executed a grid-search to find the best model parameters, testing values for $\eta, \sigma \in [5, 10, \ldots, 50]$, and maximum tree depth of $[10, 15, \ldots, 100]$. For the HSV color space, we ignored H values above 40, which restricted acceptable colors to be counted in our histograms between light cream and deep reds. Ultimately, the best performing bin sizes where $\eta, \sigma = (40, 50)$ and $\eta, \sigma = (25, 45)$ for the not cropped and cropped datasets, respectively.

We measure the performance of each model by its F1-score, which is the harmonic mean between precision and recall, i.e.,

$$F1 = 2 \times \frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}} = \frac{TP}{TP + \frac{1}{2}(FP + FN)},$$

where TP, FP, and FN are the number of true-positive (correctly identified, i.e., predicted class is the same as the ground truth class), false-positive (identified incorrectly), and false-negative (not identified) samples, respectively.

However, in our experiment results, we also show the precision and recall scores, which are computed as,

$$\text{Precision} = \frac{TP}{TP + FP}, \text{ and}$$
$$\text{Recall} = \frac{TP}{TP + FN}.$$

All model training was executed on a MacBook Pro 11 system running a 4-core Intel Core i7 2.2 GHz CPU and equipped with 16 GB RAM. Models were trained using the Apple CoreML framework.

Our mobile phone application was implemented using Swift and Objective-C and makes use of the OpenCV library. Device testing was executed on an iPhone 11 Pro running iOS 15.4.1.

B. Experimental Results

In this section we detail the results of our extensive experiments measuring both the effectiveness and efficiency of our test strip localization and CKD prediction models. First, we study which feature extraction methods lead to better CKD predictions and how feature extraction parameters affect the overall classification performance of the model. Then, we investigate localization effectiveness and overall inference efficiency.

1) Model effectiveness: Table III shows the best model results for each augmentation dataset when extracting both RGB and HSV features from the samples. First of all, it was clear from our results that HSV features outperform RGB ones in all categories. The best RGB result is worse than the worst HSV result. This may be due to the fact that RGB features are inflexible to slight color variations in the detection zone, while the histogram constructed from the HSV color space is position invariant and can absorb small shifts in color.

Among the four classification algorithms we tested, the two ensemble methods constantly outperformed the simpler decision tree and linear regression models, irrespective of the augmentation dataset used. Of all models, boosted trees performed the best, achieving an impressive $\mathbf{90.38}$ F1-score using samples from the No Crop dataset.

Using HSV features, the No Crop samples performed best across the three types of samples extracted from the test strip detection zones. This may be due to the fact that they capture more information by considering all the pixels in the detection zone, rather than just those in a 64×64-pixel region.

TABLE III
MODEL EFFECTIVENESS

RGB Features

Augmentation/Dataset	**No Crop**			**Center Crop**			**Random Crop**		
Model	Precision	Recall	F1	Precision	Recall	F1	Precision	Recall	F1
Logistic Regression	83.32	83.08	83.17	75.52	75.77	75.31	80.99	81.08	81.00
Decision Tree	81.00	80.77	80.87	76.19	76.54	76.27	79.03	78.85	78.92
Random Forest	80.34	80.38	80.36	79.30	79.62	79.35	82.59	82.46	82.51
Boosted Trees	84.48	84.23	84.32	81.41	81.54	81.18	**85.11**	**85.00**	**85.04**

HSV Features

Augmentation/Dataset	**No Crop**			**Center Crop**			**Random Crop**		
Model	Precision	Recall	F1	Precision	Recall	F1	Precision	Recall	F1
Logistic Regression	83.74	83.85	83.85	78.50	78.55	78.46	82.23	82.15	82.18
Decision Tree	81.40	81.34	81.54	83.10	83.16	83.08	81.47	81.46	81.46
Random Forest	87.61	87.70	87.69	86.16	86.28	86.15	83.84	83.77	83.79
Boosted Trees	**90.34**	**90.37**	**90.38**	88.10	88.13	88.08	85.99	85.85	85.87

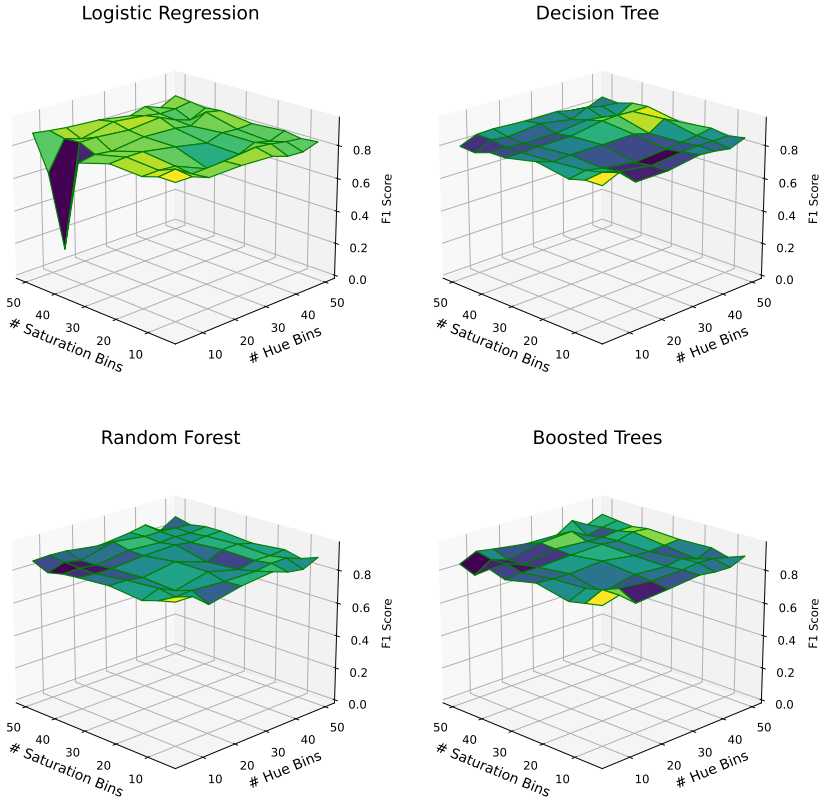

Fig. 7. Parameter tuning for the HSV feature representation.

In order to understand how the histogram parameters η and σ affect the performance of our models, we tested our models with HSV features and a variety of values for $\eta, \sigma \in [10, 50]$, drawn at random, and present the results in Fig. 7. Our results indicate that, in most cases, there is little variance in the F1 score for different η and σ values, indicating that our HSV histogram features are robust.

2) Localization effectiveness: We evaluated localization effectiveness by performing 100 consecutive predictions with a variety of test strips. We used the detection zone display at the bottom of the screen (see Fig. 5 right) to verify that the detection zone was correctly localized. We marked a failed localization if it had at least 20% non-detection zone pixels. During our 100 tests, we encountered 4 failures, indicating a localization error rate of 0.04.

3) Inference efficiency: Our localization algorithm is extremely efficient, taking up only several milliseconds, and is only delayed by 1-2 seconds if the camera needs to focus. As a result, we focused our inference efficiency analysis on the prediction task, and present results in Table IV. As expected, the

TABLE IV
INFERENCE EFFICIENCY

Model	Frames/second
Random Forest	171
Logistic Regression	120
Decision Tree	95
Boosted Trees	48

simple algorithms, such as decision tree and logistic regression, are faster than boosted trees, which is an ensemble method. Interestingly, random forest has the fastest inference, despite the fact that it is also an ensemble method. Overall, all methods we tested, including our best performing model, boosted trees, executed in only a fraction of a second and could thus be used for real-time CKD prediction.

V. FUTURE WORK AND CONCLUSION

In this work, we proposed an inexpensive yet efficient and effective method for chronic kidney disease testing that involves scanning a test strip with a mobile phone. The test strip changes colors when creatinine is applied to it, and the application uses the color change to predict the severity of the disease. We proposed computer vision algorithms for localizing the test strip detection zone once the phone camera is activated and trained machine learning models that can effectively classify the severity of the disease, achieving an F1-score greater than 90%.

In the future, we would like to construct even more sophisticated and robust CKD prediction models using deep learning and other hybrid approaches. With the advent of Apple's CoreML Unified Conversion API [27], which now supports popular Deep Learning frameworks such as TensorFlow, PyTorch, and Keras, transitioning to a deep learning model only seems natural. Additionally, while this preliminary work showed that machine learning and AI models can successfully be used to translate from the color of a test trip to CKD severity in a controlled environment, in our future work we will develop test strips and models that are effective for creatinine in human analogue and human blood.

In terms of application development, we would like to be able to dynamically push new and improved models from the cloud to device, without an application update, which would further improve the experimentation workflow.

ACKNOWLEDGEMENTS

The authors would like to thank Ragwa M. El Sayed and Rathna Ramesh for their work executing creatinine solution experiments, the results of which were used in this work.

REFERENCES

[1] Centers for Disease Control and Prevention, "Chronic kidney disease in the united states, 2021," tech. rep., US Department of Health and Human Services, Centers for Disease Control and Prevention, Atlanta, GA, 2021.

[2] F. Otto and H. Wu, "A system of blood analysis," *Journal of Biological Chemistry*, vol. 38-1, pp. 81–110, 1919.

[3] Nova Biomedical, "Statsensor and statsensor xpress creatinine and egfr meters." https://www.novabiomedical.com/statstrip-creatinine/. accessed 2022-04-27.

[4] Fisher Scientific, "Radiometer america radiometer abl800 flex blood gas analyzer series - abl835." https://www.fishersci.com/shop/products/abl835/NC2015964. accessed 2022-04-27.

[5] AAA Wholesale Company, "Istat handheld blood analyzer abbott 04j4850." https://www.aaawholesalecompany.com/abb-04j48-50-ea.html. accessed 2022-04-27.

[6] M. Kapoor and D. C. Anastasiu, "A data-driven approach for detecting autism spectrum disorders," in *Data Science – Analytics and Applications*, iDSC 2019, (Wiesbaden), Springer Fachmedien Wiesbaden, 2019.

[7] T. Saba, "Recent advancement in cancer detection using machine learning: Systematic survey of decades, comparisons and challenges," *Journal of Infection and Public Health*, vol. 13, no. 9, pp. 1274–1289, 2020.

[8] B. Bose, T. Downey, A. K. Ramasubramanian, and D. C. Anastasiu, "Identification of distinct characteristics of antibiofilm peptides and prospection of diverse sources for efficacious sequences," *Frontiers in Microbiology*, vol. 12, 2022.

[9] J. M. Ruano-Lopez, M. Agirregabiria, G. Olabarria, D. Verdoy, D. D. Bang, M. Bu, A. Wolff, A. Voigt, J. A. Dziuban, R. Walczak, and J. Berganzo, "The smartbiophone[trade mark sign], a point of care vision under development through two european projects: Optolabcard and labonfoil," *Lab Chip*, vol. 9, pp. 1495–1499, 2009.

[10] V. Oncescu, M. Mancuso, and D. Erickson, "Cholesterol testing on a smartphone," *Lab Chip.*, vol. 14, pp. 759–763, 2 2014.

[11] M. Swain and D. Ballard, "Indexing via color histograms," in *Proc 3 Int Conf Comput Vision*, pp. 390–393, Publ by IEEE, 12 1990.

[12] B. A. Olshausen and D. J. Field, "Emergence of simple-cell receptive field properties by learning a sparse code for natural images," *Nature*, vol. 38I, p. 607 EP, 1996.

[13] P. O Hoyer and A. Hyvärinen, "A multi-layer sparse coding network learns contour coding from natural images," *Vision research*, vol. 42, pp. 1593–605, 07 2002.

[14] J. van de Weijer, T. Gevers, and A. Gijsenij, "Edge-based color constancy," *IEEE Transactions on Image Processing*, vol. 16, pp. 2207–2214, Sept 2007.

[15] J. T. Barron, "Convolutional color constancy," *CoRR*, vol. abs/1507.00410, 2015.

[16] S. Bianco, C. Cusano, and R. Schettini, "Single and multiple illuminant estimation using convolutional neural networks," *CoRR*, vol. abs/1508.00998, 2015.

[17] Z. Lou, T. Gevers, N. Hu, and M. Lucassen, "Color constancy by deep learning," *British Machine Vision Conference 2015*, pp. 76.1–76.12, 2015.

[18] M. Engilberge, E. Collins, and S. Süsstrunk, "Color representation in deep neural networks," in *2017 IEEE International Conference on Image Processing (ICIP)*, pp. 2786–2790, Sept 2017.

[19] B. Debus, D. Kirsanov, I. Yaroshenko, A. Sidorova, A. Piven, and A. Legin, "Two low-cost digital camera-based platforms for quantitative creatinine analysis in urine," *Analytica Chimica Acta*, vol. 895, pp. 71–79, 2015.

[20] T. Lalanne and C. Lempereur, "Color recognition with a camera: a supervised algorithm for classification," in *1998 IEEE Southwest Symposium on Image Analysis and Interpretation (Cat. No.98EX165)*, pp. 198–204, April 1998.

[21] M. P. Paulraj, R. H. C., R. P. Krishnan, and S. S. M. Radzi, "Color recognition algorithm using a neural network model in determining the ripeness of a banana," in *Proceedings of the International Conference on Man-Machine Systems (ICoMMS)*, 2009.

[22] A. S. Levey, J. Coresh, T. Greene, J. Marsh, L. A. Stevens, J. W. Kusek, and F. Van Lente, "Expressing the modification of diet in renal disease study equation for estimating glomerular filtration rate with standardized serum creatinine values," *Clinical Chemistry*, vol. 53, no. 4, pp. 766–772, 2007.

[23] H. Yuen, J. Princen, J. Illingworth, and J. Kittler, "Comparative study of hough transform methods for circle finding," *Image and Vision Computing*, vol. 8, no. 1, pp. 71–77, 1990.

[24] OpenCV, "Hough circle transform." https://docs.opencv.org/3.4/d4/d70/tutorial_hough_circle.html. accessed 2022-04-27.

[25] J. Redmon, S. K. Divvala, R. B. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," *CoRR*, vol. abs/1506.02640, 2015.

[26] J. Redmon and A. Farhadi, "YOLO9000: better, faster, stronger," *CoRR*, vol. abs/1612.08242, 2016.

[27] Apple, "Unified conversion api." https://coremltools.readme.io/docs/unified-conversion-api. accessed 2022-04-30.