Summer 2018

Machine Learning Methods for Kidney Disease Screening

Rathna Ramesh
San Jose State University

MACHINE LEARNING METHODS FOR KIDNEY DISEASE SCREENING

A Thesis

Presented to

The Faculty of the Department of Computer Engineering

San José State University

In Partial Fulfillment

of the Requirements for the Degree

Master of Science

by

Rathna Ramesh

August 2018

The Designated Thesis Committee Approves the Thesis Titled

MACHINE LEARNING METHODS FOR KIDNEY DISEASE SCREENING

by

Rathna Ramesh

APPROVED FOR THE DEPARTMENT OF COMPUTER ENGINEERING

SAN JOSÉ STATE UNIVERSITY

August 2018

ABSTRACT

MACHINE LEARNING METHODS FOR KIDNEY DISEASE SCREENING

by Rathna Ramesh

The number of people diagnosed with advanced stages of kidney disease has been rising every year. Early detection and constant monitoring are the only way to prevent severe kidney damage or kidney failure. Current test procedures require expensive consumables or several visits to the doctor, which results in many people foregoing regular testing. To address this problem, we propose a cost-effective teststrip-based testing system that can facilitate kidney health checks from the comfort of one's home by using mobile phones. The specially designed teststrip facilitates a colorimetric reaction between alkaline picric acid and creatinine in a blood sample that has been applied to the teststrip. Our system uses state-of-the-art deep learning localization models to capture quality images of the teststrip using a cell phone, and then processes them using computer vision and machine learning techniques to predict the concentration of creatinine in the sample based on the change in color. The predicted creatinine concentration is then used to classify the severity of the kidney disease as normal, intermediate risk, or kidney failure. We thoroughly evaluate the effectiveness of our models, both in the localization and classification tasks, and find that our histogram of color-based, hybrid nearest neighbor methods outperform alternatives and exhibit good overall prediction performance.

ACKNOWLEDGMENTS

I would like to thank my advisor, Dr. David C. Anastasiu, Assistant Professor in the department of Computer Engineering, San José State University. He was always eager to hear out my ideas and nudged me in the right direction when needed. Without his constant motivation and immense technical expertise, this thesis would not have been possible.

I would also like to thank Dr. Alessandro Bellofiore, Assistant Professor in the department of Biomedical, Chemical and Materials Engineering, and his Masters student Ragwa M. Elsayed, for partnering with us on this research and providing us with the teststrip image dataset used in the experiments. Moreover, I would like to thank Dr. Magdalini Eirinaki, Professor in the department of Computer Engineering, for her encouragement, insightful comments, and the hard questions she posed as part of my thesis committee.

Thanks to the College of Engineering, San José State University, for providing access to their High Performance Compute cluster to run my many experiments on. I would also like to acknowledge the help extended by my roommates, Neha Kamthe and Mrinal Bijjargi, in annotating the localization model dataset and proof reading the sections of the thesis related to teststrip design, respectively. Finally, I would like to thank the gods for giving me such wonderful parents who have been a wall of support throughout.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

1 Introduction

Between 8 and 10 percent of the world's adult population have some form of kidney damage, and every year millions die prematurely of complications related to chronic kidney diseases (CKD) [1]. Luckily, with early diagnosis and treatment, it is possible to slow or stop the progression of kidney disease. However, even the most basic kidney screening tests available, such as the blood urea nitrogen (BUN)/creatinine ratio test, cost $25 or more [2]. The high cost affects the frequency of testing among the lower income population, making early detection less likely. Patients get tested only after they start experiencing symptoms and, by then, it is already too late for treatment to be effective. To address these issues, in this thesis, we present a novel kidney health screening system that is affordable and easy-to-use. Our system uses state-of-the-art computer vision-based machine learning techniques on a mobile phone to analyze a picture of a teststrip and predict the status of kidney health.

Creatine is a compound formed during the protein metabolism. It is a source of energy for muscles. The by-product waste substance created when muscles break down creatine is called creatinine. The kidneys are tasked with filtering almost all of the creatinine from the blood and releasing it into urine. This makes blood creatinine levels a good indicator of kidney health. Most kidney health clinical testing use Jaffe's reaction, a colorometric reaction of creatinine and picric acid in an alkaline solution, to measure the levels of creatinine in blood and urine. When picric acid and creatinine react, they form a yellowish red compound. Max Jaffe observed that the color change that occurs is directly proportional to the concentration of creatinine. Our system also makes use of Jaffe's reaction to estimate the creatinine concentration. We use the predicted creatinine concentration to compute a generalized kidney health indicator, called the estimated glomerular filtration rate (eGFR). In our research, the modification of diet in renal disease (MDRD) equation, proposed by Levey et al. [3] is used to determine the eGFR value. The

MDRD equation accounts for factors such as age, gender and ethnicity, which results in a better estimate of effective renal function. The equation gives a value in the range of 0–150, where 60 and above indicate a healthy kidney, and patients with values between 0–15 require immediate dialysis or kidney transplant.

The current creatinine concentration measuring methods, such as absorbance spectrometry [4], require expensive devices and expertise in handling the instruments. Our research aims to lower the cost of kidney health checks, which would allow for more frequent testing and could potentially save lives. The testing procedure is designed to be simple and does not require expensive specialized equipment. We are developing a smart phone application, Kidney Health Monitor (KHM), and specially designed cost-effective teststrips, which would lead to a potential cost of less than $2 per test. Although in this work we have tested our system using only synthetic creatinine compounds and did not use human blood for carrying out experiments, the system design and teststrip construction will not change significantly in a human trial, keeping the cost relatively similar.

Our kidney health monitoring system employs several novel techniques for taking quality pictures of teststrips and analyzing them. The application uses deep learning-based object localization techniques to ensure the quality of the captured teststrip image that will be used in further analysis. The application only captures the image of the teststrip when the teststrip has been correctly positioned and oriented. Machine learning regression models are then used to predict the level of creatinine based on the color pixels captured in the image. Based on the eGFR value determined from the creatinine prediction, the application displays a color indicative of the user's kidney health. The colors green, yellow, and red are used to indicate healthy kidneys, presence of kidney disease, and need for immediate medical attention, respectively. The main contributions of this research are as follows:

- We designed a novel end-to-end system that uses computer vision techniques and machine learning to estimate kidney health from the color change in a teststrip.
- The system uses object detection and localization models to capture quality pictures, eliminating the need for external, more costly devices.
- A novel hybrid regression model is introduced that first estimates the range of creatinine concentration and then predicts the actual creatinine concentration value using specialized functions.

2 Literature Review

Blood creatinine concentrations continuously change throughout the day in a healthy adult. The test methods we are developing are designed to act as an early warning system, by letting the user know when creatinine levels in the blood are high. This would prompt the user to seek medical advice early on and avoid allowing the disease to progress unnoticed. To keep the cost as low as possible, we eliminated the need for a smartcard device that attaches to the mobile phone. Instead, our system uses teststrip detection and localization to guide the user in positioning and orienting the teststrip, which will ensure that quality pictures are taken that can result in accurate classification of disease progression. This section discusses current kidney disease screening technologies, state-of-the-art object detection and localization methods, and image processing and machine learning techniques that informed the choice of our regression models.

2.1 Kidney Disease Screening Technologies

Jaffe's reaction is still the most prevalent method to measure creatinine concentration in blood/urine sample in laboratories. Otto Folin, in his 1916 Lab Manual of Biological Chemistry [5], formalized using Jaffe's reaction in clinical and laboratory testing. To measure creatinine concentration, the Centers for Disease Control and Prevention [4] recommend charting absorbance of a precise volume of the sample infused into an alkaline picrate solution. Absorbance readings are taken at 520 nm between 19 and 25 seconds after sample injection. Over the years, laboratories world-wide have automated many analytical processes and developed systems to analyze the samples in bulk, which helped reduce the overall testing cost.

Despite being the most prevalent form of medical diagnostic testing, centralized testing is inconvenient for many patients. The testing and consultation process are often not provided as a combined service in one visit, making a second trip necessary to complete each assessment. In particular, those with a chronic disease require regular monitoring and laboratory testing. This has led to growth in self-monitoring point-of-care testing (PoCT), which brings medical diagnostic testing to the patients at the time and place of care. The healthcare sector has been making huge strides towards PoCT, partly as a result of economic pressures of high testing fees [2], and also to make healthcare less fragmented and more patient-centered [6]. However, the PoCT devices currently available on the market are expensive, making their utility limited. With the advent of artificial intelligence, PoCT researchers have turned to machine learning for building robust systems that can perform to laboratory standards for a fraction of the cost.

StatSensor [7], a point-of-care whole blood creatinine and eGFR testing device recently launched by Nova Biomedical, has gained wide popularity among medical practitioners. It incorporates multi-well technology adapted from their hospital glucose monitoring system [8]. Like most of the glucose meters, StatSensor uses an electrochemical reaction to measure creatinine concentration using the coulometric method [9]. Teststrips contain a capillary that absorbs a small amount of blood. The creatinine in the blood reacts with an enzyme electrode containing creatinine oxidase. When the assay on the electrode is reduced, it generates an electric current to re-oxidize the compound. The oxidized creatinine is directly proportional to the total charge passing through the electrode. Despite its mobility and ease of use, the high cost of the device has restricted its popularity with consumers.

Table 1 summarizes specifications of prominent PoCTs currently available in the market for measuring blood creatinine concentrations, as first presented by Shephard [10]. All of these devices require expensive single use consumables, in addition to custom concentration measuring hardware, making them very expensive for frequent domestic use.

Table 1
Non-analytical Specifications of PoCT Devices

Device	**iSTAT**	**ABL 800 Flex**	**Reflotron**	**StatSensor**
Price	$14,925.77 [11]	$8,500.00 [12]	NA	$4,080.66 [13]
Method	Enzymatic	Enzymatic	Enzymatic	Enzymatic
Principle	Amp biosensor	Amp biosensor	Dye reflectance	Amp biosensor
Consumable	Cartridge	Sensor cassette	Dry strip	Dry strip
Time for result	2 min	1 min	2 min	0.5 min
Range	0.20-20 mg/dL	0.11-22.62 mg/dL	0.51-10 mg/dL	0.30-11.95 mg/dL
Size	Hand held	Bench top	Bench top	Hand held
Weight	0.7 kg	33 kg	5 kg	0.4 kg

2.2 Object Detection and Localization Models

The object localization techniques have evolved, starting with gradient based localization [14], all the way to region proposal-based localization techniques. In these region proposal-based systems, the method looks for presence of an object in certain regions of the image. The regional convolutional neural networks (R-CNN) method by Girshick et al. [15], the first method to achieve mAP of 53.3% on the VOC 2012 dataset for object detection and localization, used the region recognition method proposed by Gu et al. [16] to extract around 2000 bottom-up region proposals. For each of the region proposals, features are computed using a large convolutional neural network, after which each region is classified using class-specific linear SVMs. The Fast R-CNN method, devised by Girshick [17] improved upon its predecessor by using selective search for region proposals. This allowed 9$\times$ faster training on large datasets and 213$\times$ faster testing than R-CNN. Hosang et al. further improved on the method by adding a region proposal network (RPN) to propose fewer higher quality regions in Faster R-CNN [18]. The RPN network ranks pre-defined region boxes and proposes the ones most likely to contain the object. This drastically reduces the number of regions proposed and thus reduces the time taken to classify and allows testing 5 frames per second on a GPU. A comprehensive literature survey and comparison between different state-of-the-art object proposal methods is provided by Hosang et. al. [19], [20].

For our system, the speed of prediction and ability to use the method in the memory- and computation-constrained space of mobile applications were the primary factors we considered when choosing the method for teststrip detection and localization. As the method needs to run on a smart phone, it needs to be light-weight, as well as accurate. A state-of-the-art fast method that meets these requirements is the Redmon et al. YOLO deep neural network model [21]. The model produces all of the detections in only one pass of the image (hence the name, ”You Only Look Once”). The input image is logically divided into $S \times S$ grids. Each grid cell predicts B bounding box coordinates and box confidence scores for the object whose center is likely to be within the grid. That is, if the model is trained to localize C different objects, then the model would generate $S \times S \times (B \times 5 + C)$ bounding box proposals and would provide a confidence level for the likelihood of each bounding box containing the sought-after object.

The single-shot detection (SSD) method by Lin et al. [22] is another viable option which also provides multiple box prediction in only one pass, similar to YOLO. It is more accurate than YOLO and is comparable to slower techniques that perform explicit region proposals and pooling (*e.g.*, Faster R-CNN). In SSD, the input image is split into default bounding boxes of various sizes with aspect ratios and feature map scale values, rather than grids. At prediction time, the network generates confidence scores for the presence of each object category in each default box and produces adjustments to the box, such as scaling, to better fit the object. However, preparing the data for training and decoding the prediction output is a complex task. The generated output bounding boxes need to be mapped to the default boxes in the region by applying size adjustment transformations predicted from the aspect ratio and scale parameters. This makes the adoption of the SSD neural network structure in other artificial intelligence frameworks difficult.

A number of improvements were introduced in YOLO version 2 [23], including pre-computing anchor aspect ratio priors using k-means clustering, adding a pass-through layer from an earlier layer at 26$\times$26 resolution to the fine-grained feature maps, and augmenting images during training. These changes make the network aware of multiple object scales and lead to improved overall effectiveness. In our work, we adopted YOLO version 2 due its simplicity, effectiveness, and wide usage in mobile-based applications. In the future, we intend to expand our research to combine concentration prediction and teststrip detection in one model, and YOLO's network architecture makes it possible to make such additions in the prediction outputs.

2.3 Image Processing

Image processing consists of techniques that are applied to images in order to enhance discriminative features for different tasks important in computer vision applications. Noise and unwanted information are generally removed from a processed image. Then, features are sought that can exacerbate similarities and differences between images. The colorspace the image will be represented in and feature extraction techniques are chosen to accommodate the specific application the images is being processed for. The colorspace defines how the colors in the pixels of an image are represented as numbers, generally in a vector or matrix form. The computer can then use the vectors in developing machine learning models, or functions, that can answer real world questions, such as the presence or absence of a cat in a picture. Our application domain requires detecting subtle differences in color as the creatinine in the blood reacts with the picric acid solution in the teststrip.

Many computer vision applications, such as the famous Viola-Jones real-time face detection [14], use grayscale images in their methods to reduce computation and memory cost. Grayscale images encode only one color, instead of three, and thus require less resources to process. Information such as gradient and focus, which are crucial for pattern recognition problems, can be easily extracted from grayscale images through simple differencing operations. However, our problem is centered around color changes between images, rather than patterns found within an image. Thus, grayscale images will not be suitable for our application.

Since the popularization of color triangles through photography experimentations by James Clerk Maxwell in the 1860s, colors have been represented as chromatic triplets for use in television and photographic technologies. The most widely used colorspace, RGB, is obtained by linearly combining the values of red (R), blue (B), and green (G) present in the sample. Both chrominance (color information) and luminance (intensity of incident light/exposure) values are combined in the RGB representation. Instead, the HSV color encoding [24] provides more distinction between the chrominance and luminance values. It has a cylindrical representation system where hue (H) is the angular component, encoding the color spectrum as angles between 0°and 360°, and saturation (S) denotes the intensity of the color radially increasing outwards. The value component (V) encodes brightness, ranging from completely dark (0) to intense coloring (255). In our research, we have extracted features from both the RBG and HSV colorspaces to train our machine learning models.

A histogram of colors extracted from the RGB colorspace image representation has been successfully used to index images for image retrieval applications [25]. However, color histograms obtained from RGB are susceptible to light intensity and exposure changes, limiting their application in our research. Color constancy approaches depend on statistics of outputs of linear [26] and non-linear filters [27], [28]. Convolutional neural network-based methods have been proven to be much more effective than statistical methods in achieving color constancy [29]–[31], though the representation of colors in deep neural networks is still poorly understood [32]. In our application, it is also important that the reason behind the output classification decision can be easily understood by the user, if they choose to know.

2.4 Classification and Regression Models

One closely related system that predicts continuous values for color change observed in the reaction between urine and picric acid solutions is the homemade spectrometer by Debus et al. [33]. The system uses a laptop screen and a mobile phone to capture the absorbance spectrum in a strictly light controlled setup. Images of the absorbance spectrum are used to find the creatinine concentration using a Matlab program. Despite being a low cost system, the setup requires technical acumen and the detection is time consuming. We did not find any machine learning system that predicts the amount of color change as continuous values. Almost all of the previous works use a classification model to solve their respective problems.

One of the early experiments to classify images based on color from a camera feed was conducted by Lalanne and Lempereur [34]. They used a supervised algorithm based on Bayesian statistics and a two layer neural network to classify the temperature of thermal paints based on their color. More recently, Paulraj et al. used a 5 hidden layered artificial neural network to predict the ripeness of a banana with an accuracy of 96% [35]. In their experiment, each of the red, blue and green color channels were converted into a three binned histogram, which was given as input to the neural network. Our project focuses on identifying different shades in a smaller color range than in the experiments conducted to decide the ripeness of a banana, and using histogram of RGB color channels image representations may not work as well in our application.

3 Chemical Experiment Design

This project is a collaboration between the department of Biomedical Engineering and the department of Computer Engineering at San José State University. We collaborated on designing the experiment by verifying that the color change is prominent enough for the image processing techniques to differentiate the minutest change in concentration values. We tested for different concentrations of picric acid, dimensions of the detection zone, light intensities, and distances from the camera to perform the experiments. Our primary responsibility, and the subject of this thesis, was to use the images captured throughout the chemical experiments to develop machine learning models that can predict the amount of creatinine in a blood sample. In contrast, our collaborators, Dr. Alessandro Bellofiore and his student Ragwa Elsayed, executed the chemical experiments and did not participate in processing images or learning prediction models. This section discusses the teststrip design and the data collection process.

3.1 Teststrip Design

The teststrip design is inspired by the simplicity and cost-effectiveness of over-the-counter pregnancy tests. The design is similar to that of lateral flow assays (LFA). LFAs are simple paper-based devices intended to detect the presence (or absence) of a target compound in a liquid sample without the need for specialized and costly equipment. Fig. 1 shows the typical components of an LFA [36].

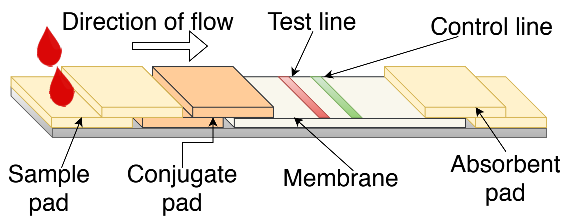

Fig. 1. Lateral flow assay teststrip design.

The sample pad is made up of an absorbent material, which ensures that the applied liquid sample spreads evenly and in controlled amounts onto the conjugate pad. The activated analyte from the conjugate pad migrates along the strip into the porous membrane called the detection zone. There, the analyte whose presence is to be detected in the sample reacts with the reagent in the membrane, producing a colorimetric response. The control line is an optional component and indicates proper liquid flow through the strip. The pad at the other end of the membrane absorbs any extra fluid that flows out of the membrane.

Since the teststrip acts as a reaction bed that facilitates Jaffe's reaction to quantify the color change, we do not need both the control and test lines in our experiment. The membrane in the detection zone is pretreated with picric acid. The reaction between picric acid (reagent) and creatinine (analyte) in the blood sample creates a color change in the detection zone. The concentration of creatinine in the blood sample is predicted from the degree and direction of the color change observed in the detection zone. Fig. 2 shows three example teststrips on which different creatinine concentrations were applied. Figures in this thesis are best viewed in color. The color difference between the first and the third teststrips, which have had 1.0 mg/dL and 20.0 mg/dL creatinine solutions applied, respectively, is evident even with the naked eye. However, the difference in color is not noticeable between the first and second teststrips, which differ in concentration by only 1.0 mg/dL. This indicates that the feature extraction and prediction models we develop must be able to capture discriminative features humans may not be able to detect. Fig. 3 shows a large number of the teststrips that were used in our experiments.

As this is a preliminary study meant to validate the design of our kidney disease prediction system, we used a synthetic creatinine solution to create our dataset, rather than human or human-analog blood. When the creatinine solution is applied to the sample pad, the solution flows through it into the detection area. There is no conjugate pad in this

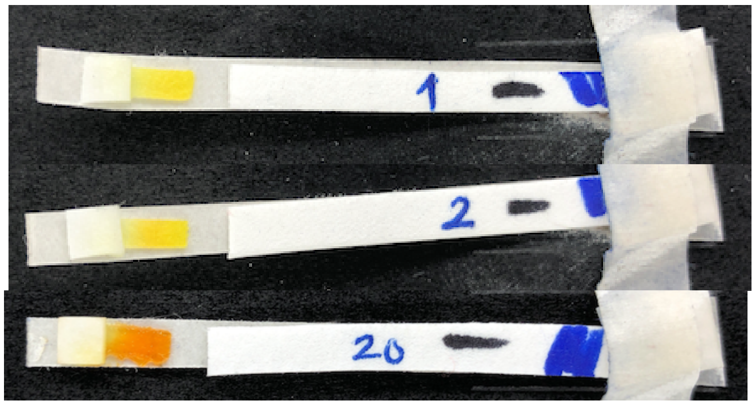

Fig. 2. Example teststrips with different creatinine concentrations.

Fig. 3. Testrips created and used in our experiment.

teststrip design as the creatinine solution does not need to be filtered from blood in our experiments. In the next stage of the project, a conjugate pad which separates the red blood cells from the sample will be added to make the teststrip ready for experiments with blood samples.

3.2 Data Collection

The presence of kidney disease and its progression can be determined using the eGFR scale, as shown in Fig. 4. In our study, we use the MDRD equation [3] to determine the eGFR value from the creatinine concentration, which is defined as

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

where S_{Cr} is the creatinine concentration in the blood sample, measured in mg/dL. Analyzing the formula, we can see that the eGFR value is inversely proportional to creatinine concentration to the power of 1.154. The changes in factor contributed by different creatinine concentration in the eGFR equation is shown in Fig. 5. We can see that drastic changes occur in the factor for small changes in creatinine concentrations between 0–2 mg/dL. Beyond 4 mg/dL, the factor is very close to zero. Thus, the precision of the creatinine concentration predicting regression model between 0–4 mg/dL is crucial to ensure good overall kidney health prediction.

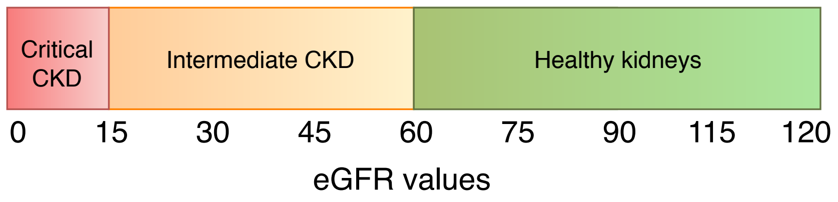

Fig. 4. Kidney disease progression given different eGFR values.

The eGFR values of 15, 30, 60 and 90 are considered critical, as they differentiate between different stages of kidney disease (Fig. 4). To determine the range of creatinine concentrations that give these critical boundary eGFR values, we plotted a graph between the creatinine concentration and the age of the patient for each of the critical eGFR values. To capture the variance introduced by gender and ethnicity, we plotted four functions in the graph, one for each combination of gender and ethnicity noted in the equation. Fig. 6 shows our generated plots for each of the eGFR boundaries. The plots indicate that, for the majority of age groups, creatinine concentrations between 0.4 mg/dL and 10 mg/dL lead to *critical* kidney health conditions. It is thus essential that we capture the color change

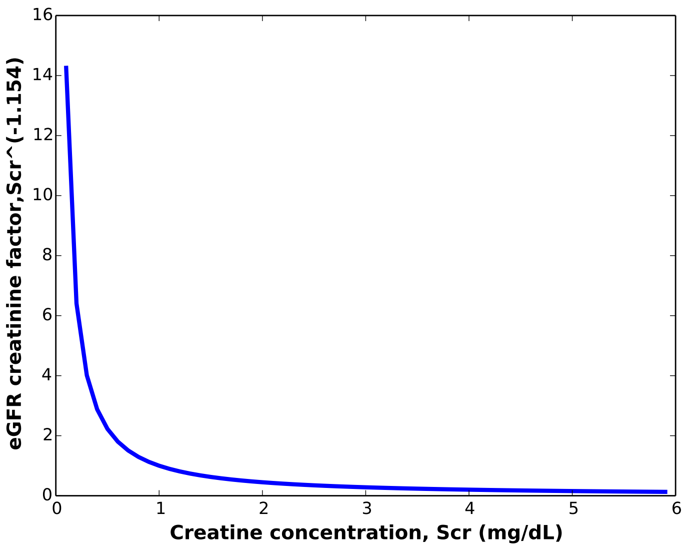

Fig. 5. Variation of the eGFR creatinine factor given different creatinine concentrations (Scr).

observed in creatinine concentrations in smaller steps in this range, to provide machine learning models with enough data to accurately predict the patient's kidney health.

Given our analysis results, we designed our experiment to collect data in steps of 0.1 mg/dL between 0 mg/dL to 3.9 mg/dL of creatinine concentration, so that we could capture the minutest differences in color. For higher creatinine concentrations, the differentiation between kidney disease severity codes involves wider concentration ranges, which we uniformly sample using wider steps. We tested concentrations between 4 mg/dL and 7.5 mg/dL in steps of 0.5mg/dL, between 8 mg/dL and 19 mg/dL in steps of 1 mg/dL, and between 20 mg/dL and 60 mg/dL in steps of 10 mg/dL. Moreover, we prepared 4 teststrips for each concentration and took pictures of each strip at 5 different stages of applying the creatinine solution, namely

- *stage 0*: before applying the creatinine solution,
- *stage 1*: 2 minutes after applying the creatinine solution,

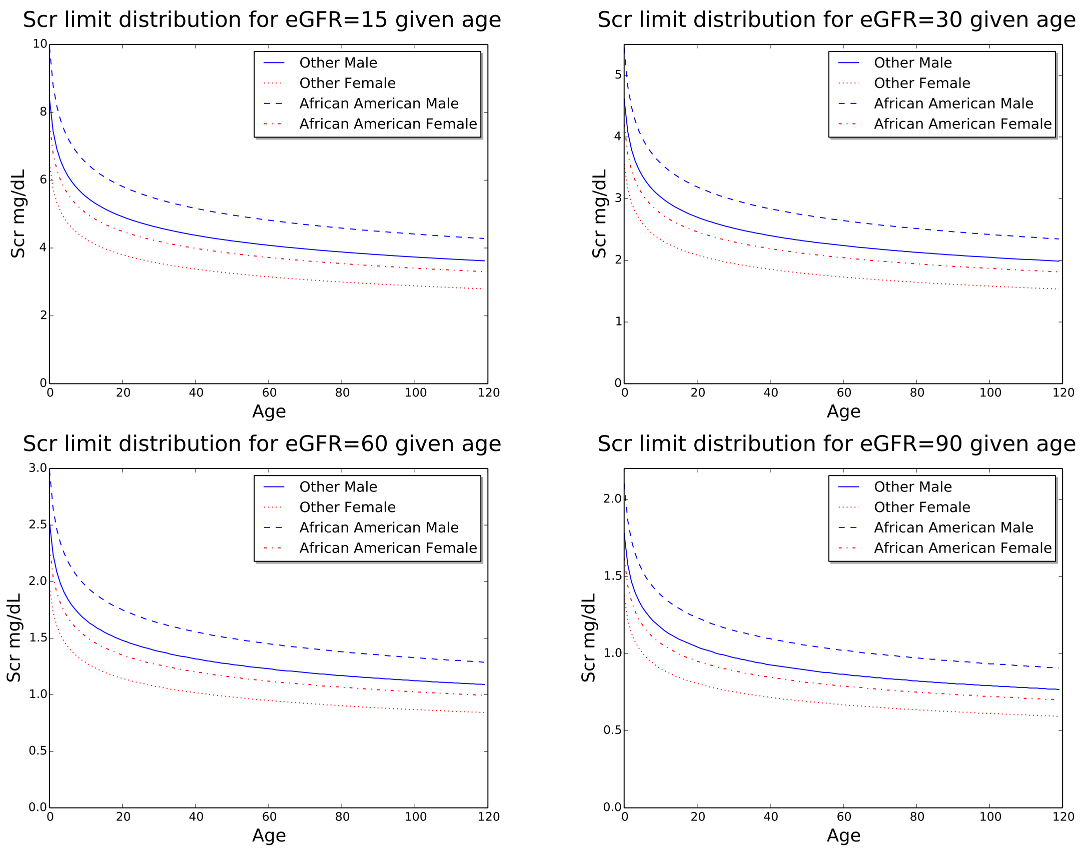

Fig. 6. Creatinine concentrations (Scr) vs. age at critical eGFR values.

- *stage 2*: 12 minutes after applying the creatinine solution,
- *stage 3*: 22 minutes after applying the creatinine solution, and
- *stage 4*: 32 minutes after applying the creatinine solution.

The stages were chosen with the knowledge that, as time progresses and the teststrip dries, the saturation and intensity of the color on the teststrip may also change. One of the goals in our experiment is to find a time segment after which the color changes are significant enough for all concentrations of creatinine so that machine learning algorithms can differentiate between them and learn from them. We created 4 teststrips to test the repeatability of the chemical experiment per concentration. Thus, a total of 260 teststrips were prepared for this experiment, as summarized by Table 2. For each stage of the experiment, except stage 0, we captured two pictures of each teststrip, to ensure that at least one high quality picture existed for each chemical experiment data point.

Table 2
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

In the current research, our goal is to validate the idea of using image processing and artificial intelligence on a smart phone to predict kidney health. In this preliminary study, we eliminated several sources of error that could potentially affect the effectiveness of our prediction models. As changes in ambient light can affect the color being perceived by cameras, we used a light box to avoid external light interference. Moreover, even though our project involves designing methods to aid users in taking quality pictures of the strips, we took all pictures from a fixed distance, using fixed focus and aperture settings on the camera. Additionally, while we developed models for localizing the teststrip in an image taken by the camera, we used manually annotated bounding boxes in training and testing our prediction models. Successful prediction in this setting will signal that our kidney health monitoring system may also work in general lighting conditions and using the localization model to extract the detection zone from the teststrip.

The light box we used in the experiment has a black exterior to keep external light from the sample and a silver interior reflects the light within the light box. An array of white chromatic LEDs attached to the roof are the only source of illumination within the light box. Controls are provided to adjust the intensity of the light within the light box. The light box has a small opening in its roof, big enough for a camera to capture images of objects placed inside it. The teststrips are placed at a distance of 10 cm from the roof of the light box, which ensures that the intensity of the light incident on the teststrips remains constant throughout the data collection. The teststrips were placed on a black background, which we found gave a better contrast, allowing easier detection and localization of the teststrips.

4 Methodology

To combine the image capture and prediction, we have designed an iOS application that

- detects the presence of the custom teststrip in the camera frame and localizes it,
- captures a high quality picture of the teststrip and localizes the detection zone,
- processes the color change in the detection zone and directly gives a color coded indication of kidney health.

In this section, we discuss the methods that we designed for teststrip localization, prediction of creatinine concentration, and eGFR-based classification of kidney health.

From a picture of teststrip, the patient's kidney health is predicted as healthy, intermediate risk, or critical. To determine the appropriate category, we use the MDRD equation noted in Equation 1 to compute the glomerular filtration rate from the predicted creatinine concentration. The eGFR value is influenced not only by the concentration of creatinine in blood but also by age, ethnicity, and the gender of the patient, which makes classifying kidney disease stages impossible from the teststrips directly. As an example, a 20 year old African born male whose blood creatinine concentration is 3.1 mg/dL has moderate renal function with an eGFR value of 31.289 $mL/min/1.73m^2$. However, a 70 years old non-African woman is at high risk of kidney failure for the same blood creatinine concentration, with an eGFR value of 14.854 $mL/min/1.73m^2$. Accurate creatinine concentration predictions are essential to correctly classify renal function. Thus, we used machine learning to build regression models that can accurately predict blood creatinine concentration. The predicted creatinine concentration value is then used in the MDRD equation to compute the predicted eGFR value for a specific patient and classify the intensity of the kidney disease into the three stages.

In this section, we discuss the design of Kidney Health Monitor (KHM), the mobile application we designed to capture quality teststrip images and predict kidney health.

First, we discuss the localization model we developed to enable KHM to capture quality pictures. Next, we discuss the various feature extraction techniques we developed to enable our machine learning models to make accurate predictions. We then discuss the various baseline machine learning models we used to predict the concentration of creatinine in the teststrip. Finally, we introduce our hybrid concentration-based binned regression models for predicting the creatinine concentration from teststrip images.

4.1 KHM Application Design

Kidney Health Monitor (KHM) is designed to predict the state of a patient's kidney health in real-time using on-device machine learning models. Until recently, Apple's Metal library [37] was the only means to run machine learning models on an iOS device. Metal provides application programming interfaces (APIs) to access the built-in GPUs in Apple devices. This requires writing complex code to control different aspects of an application, such as distributing and collecting data, work load balancing, communication between cores, check pointing, and so on. The alternative would be to design a distributed application. The data from the application can be sent to a remote system that can run the machine learning model and send back the results. This requires a robust system that can scale seamlessly to service a large amount of requests in parallel. This may increase the cost of the analysis for users, both due to data transmission costs and the service cost needed to cover the server maintenance.

The recently released Apple Core ML framework for machine learning [38], customized for on-device GPU, supports extensive deep learning models along with standard machine learning models, such as support vector machines and generalized linear model [39]. It is built on top of native low-level libraries, such as Metal and Accelerate, and is capable of optimal power and efficiency for Apple devices. We used the Core ML framework to run our localization and regression machine learning models. To enable the Core ML framework to run the model, the weights from other frameworks

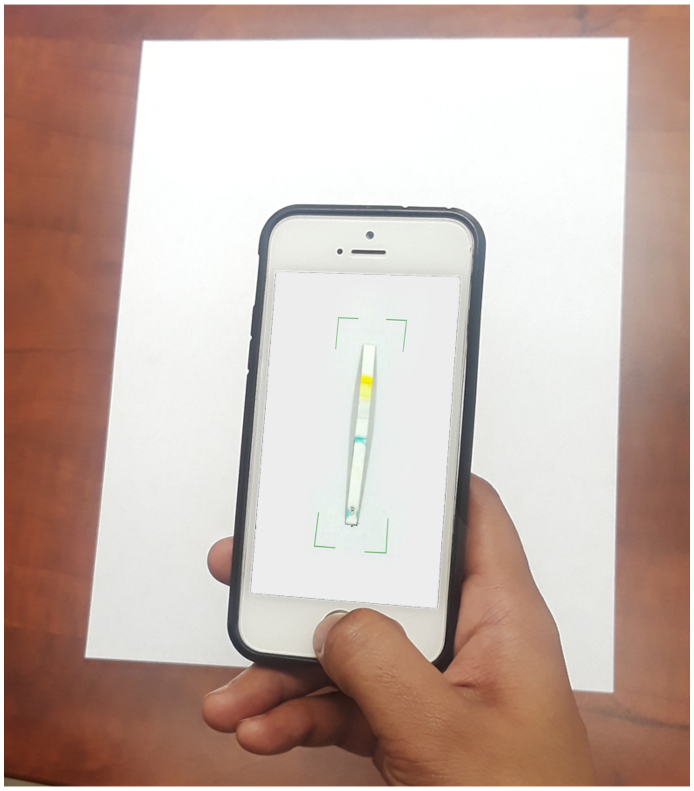

Fig. 7. Kidney Health Monitor application.

need to be converted to the *mlmodel* format expected by Core ML. The teststrip localization YOLO model was converted from the *weights* format used in *darknet* [40], which YOLO was designed to run on, to the keras-tensorflow supported *h5* format using YAD2K (Yet Another *darknet* 2 Keras tool) [41]. Then, the *h5* weights were converted to *mlmodel* format using Core ML tools [42].

We chose to build our app in the Swift 4 programming language. The various online resources made available by educational institutions, Apple, and the community have been very helpful in building the application [43], [44]. Applications written for iOS devices are required to follow strict guidelines on modularity, data and control distinctions, and data persistence. The readily available APIs, the type-written nature of the swift language, and the implicit data corruption checks, make it easy to develop iOS applications, despite the restrictions.

The application captures the camera feed at 15 frames per second. We resized these frames to 416×416 pixels, as required by the YOLO model that we trained to detect and localize the teststrip. The bounding box coordinates predicted by the localization model are used to check if the dimensions of the teststrip are as expected. If the ratio between the length and width of the teststrip do not fall under expected values, the user is prompted to re-orient the teststrip. If the length is longer, or shorter, than the expected range, the user is prompted to move the teststrip farther, or closer, accordingly. After the dimensions are verified, the application auto-focuses on the teststrip to get the best picture clarity and stores the image along with the user's name and the capture timestamp.

The localized image of the teststrip is further processed to identify the detection zone within the teststrip. Specifically, the detection zone is identified as the region containing a high concentration of yellow pixels, where yellow is defined by a range of hue and saturation values in HSV colorspace. From the detection zone, features are extracted and fed to machine learning models, which then predict the creatinine concentration in the sample and the associated eGFR value. The images captured, along with the predictions, are stored in the file system automatically. The user can choose to view or delete them from the history screen.

4.2 Teststrip Localization

We used the YOLO [23] deep neural network model to detect and localize the teststrips in our application. The network has 24 convolutional layers, followed by 2 fully connected layers. The output is optimized for the sum of squares error with a differential confidence scoring scheme for grid cells with and without objects. As each grid cell is responsible for predicting *B* bounding boxes, there could be a few box predictions without an object within the grid. The sum of squares error for such cells can push the overall confidence score of the grid cell to be very low. To remedy that, different multiplying factors, λ_{coord} and λ_{noobj} are used to calculate the error for predictions with and without an object, respectively, within each grid cell. In YOLO version 2, λ_{coord} is set to 5 and λ_{noobj} to 0.5. To make the model aware that error in prediction bounding boxes of a smaller object is more costly than that of a larger object, YOLO outputs the square of the width and height of objects internally. This enforces higher penalties for errors when predicting smaller objects, improving the quality of localization for both smaller and larger objects.

As discussed in Section 2.2, YOLO splits the input images into 7×7 grid cells. For each cell, 5 bounding box coordinates and associated confidence scores are predicted for an object whose center is likely to be within the grid. In the *darknet* framework, the bounding box coordinates of grids with confidence above a set threshold are returned as detections. Since we need to localize only one teststrip at a time, we chose the bounding box coordinates predicted by the grid with the highest confidence to localize the teststrip.

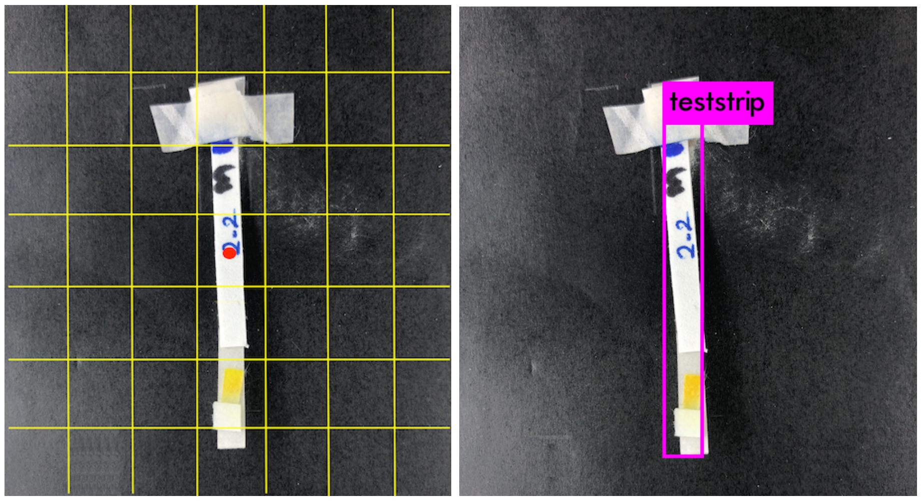

Fig. 8. Teststrip localization by the YOLO model in the *darknet* framework.

4.3 Feature Extraction

In our system, to predict the creatinine concentration from the images of teststrips, we use machine learning techniques applied to features extracted from the detection zone. To give our learning models the best chance at prediction, we manually annotated the detection zones for all teststrip images from the concentration prediction dataset. The BBox-Label-Tool [45] was used to identify the detection zone bounding box coordinates.

For each of the images in the dataset, the image is cropped to contain only the detection zone given by the appropriate bounding box coordinates using OpenCV library functions [46]. These cropped images are then used to extract features for the learning models. In the remainder of this section, we discuss the three types of features that we extract from the cropped images of the detection zones.

4.3.1 RGB Pixel Values

In the RGB representation, each pixel in an image is expressed as a vector of three integers, one for each of the red (R), green (G), and blue (B) color channels, with values in the range of $(0, 2^8)$. To extract RGB features, we first extract the detection zone images as 3-dimensional matrices of red, green and blue channels. However, the cropped detection zone images are each of different sizes. Further, since the detection zones were manually annotated, there is a good chance of annotating areas outside the color change, introducing noise in the pixels, as shown in the left image in Fig. 9.

We further crop the image to the center 64×64 pixels within the bounding box. This solves both the problem of irregular size and unintentional noise introduced in the feature vector. Fig. 9 (right) shows the centrally cropped image used to create our RGB feature vector. The RGB color values for each pixel in the cropped region are concatenated to create a single vector.

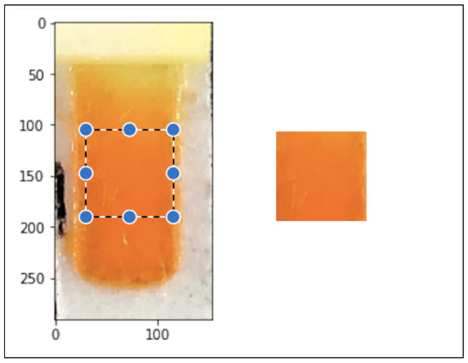

Fig. 9. Image of detection zone cropped using the bounding box annotations (left) and an image further cropped to the central 64×64 pixels (right).

4.3.2 Histogram of Colors

One drawback of using the RGB vector representation is that, if the reaction does not happen uniformly throughout the central part of the detection zone, different sections of the RGB vector may have deviations from the color representative of the creatinine concentration. To alleviate this problem, we devised histogram of colors (HOC) feature vectors, which represent the image as a distribution of colors they contain, rather than a sequence of colors contained in the image pixels. A histogram is a representation of distribution of elements within a set of values. An equal width histogram splits the range of the set into bins of equal sizes and counts the number of elements that fall into each bin. Normalizing these counts gives the distribution of the elements in the set.

For our data, we choose to calculate the color distribution based on the HSV color space values. In the HSV colorspace, the effect of change in lighting conditions on the color is captured by the value (V) channel, independent of the hue (H) and the saturation (S) channels. This gives images better color constancy than in the RGB representation, where the color and brightness information are held by all the three channels. In the HSV colorspace, the hue (H) represents the color spectrum information and saturation (S) encodes the intensity of coloring. The brightness, encoded by the value (V) component, is not significant for our application. Therefore, we created histograms to represent the distribution of only the hue and saturation values.

We apply the histogram on the cropped images of detection zones. Since the color change observed in Jaffe's reaction takes place in a particular range of yellow, pixels belonging to all the other colors are irrelevant to our analysis. Thus, we mask the hue and saturation values that are not encountered in our experiments, leaving an additional small buffer at each extremity. After experimenting with values in the range of [0, 45] for hue and [0, 255] for saturation, we chose the ranges [20, 35] and [100, 255] for masking the hue and saturation, respectively. Fig. 10 and 11 (left) show example images of detection zones before and after masking. From these masked images, histograms are computed.

To compute the histograms, we split the hue and saturation ranges into α and β equal-width buckets, respectively, creating a total of $\alpha \times \beta$ buckets. These buckets make up the feature space for representing our image. Each bucket will record the number of pixels (excluding the masked pixels) whose hue and saturation fall in the range represented by the bucket. We then compute the similarity between two images as the cosine similarity between their respective HOC vectors, which is defined as

$$Cos(\mathbf{x},\mathbf{y}) = \frac{\mathbf{x}^T\mathbf{y}}{||\mathbf{x}||_2\,||\mathbf{y}||_2},$$

where $\mathbf{x}^T\mathbf{y}$ is the dot-product of the two vectors and $||\mathbf{x}||_2$ is the L2-norm of the vector $\mathbf{x}$. Cosine similarity measures the angle between the two vectors and is invariant to vector scaling, since the angle does not change when changing the length of the vector. Therefore, as a way to improve computation efficiency, we normalize the feature vectors using the L2-norm, which scales the vectors to unit length and reduces the computation of cosine similarities to just finding the dot-product between the vectors. Fig. 10 and 11 (right) show example normalized HOC vectors extracted from the images of the teststrips depicted in the two figures.

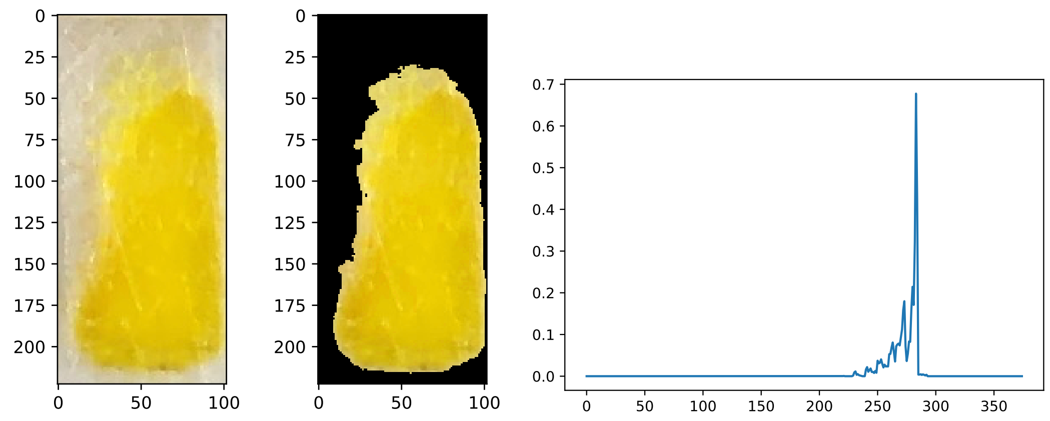
Fig. 10. Cropped detection zone (left), image after masking pixels (center), and the histogram of colors (right) of teststrip before application of creatinine solution.

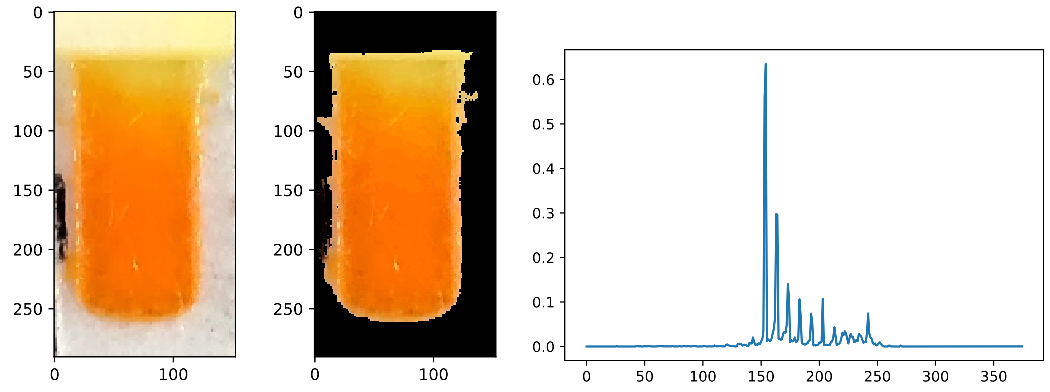
Fig. 11. Cropped detection zone (left), image after masking pixels (center), and histogram of colors (right) of teststrip 32 minutes after application of creatinine solution.

4.3.3 Histogram of Gradients

Similar to the HOC features, the histogram of gradients (HOG) features are created as L2-normalized histogram counts of features. However, the features being tracked are gradient orientations in localized portions of the image. To find these features, small portions of the image are multiplied by the kernel (-1,0,1) in both the horizontal and vertical directions. We use the HOG descriptor function from the OpenCV library, which uses sliding windows to extract these features. Fig. 12 shows an example of the sliding windows concept. Using sliding windows, gradients are extracted and counted from a section of the image, before moving on to the next section, which overlaps the first by a small margin. The extracted gradient feature counts from each section are concatenated into a single HOG feature vector, which is then L2-normalized.

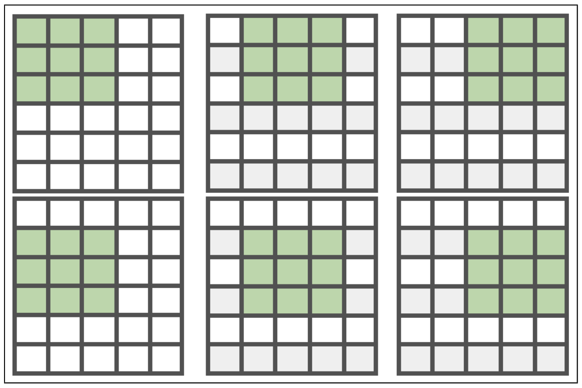

Fig. 12. Sliding windows with window size 3×3 and stride 1×1.

4.4 Baseline Prediction Models

For each of the feature extraction techniques used, we built regression models, one for each of the four time points we captured images at after adding the creatinine solution to the teststrip. In the remainder of this section, we provide a description of each of the baseline models that we used in our research. We did not apply neural network models, as the number of data points we had was limited and insufficient for learning the large number of weights in a neural network without over-fitting.

4.4.1 Linear Regression

The linear regression algorithm fits the data into a straight line equation $y = \mathbf{a}^T\mathbf{x} + b$, where $\mathbf{x}$ is the feature vector, and $\mathbf{a}$ and b are meta-parameters, such that most of the data points lie on (or close) to the line. The meta-parameters are learned iteratively, with a goal to reduce the cost of prediction given by a chosen cost function. We use the root mean squared error (RMSE) as the chosen optimization cost function for the linear regression model. Given n samples, the RMSE can be computed as

$$RMSE = \frac{1}{n}\left(\sum_{i=1}^{n}(y_i' - y_i)^2\right)^{1/2},$$

where y_i' is the predicted creatinine concentration and y_i is the actual creatinine concentration for the ith sample.

4.4.2 Support Vector Regression

The support vector regression (SVM) algorithm, which is based on the support vector machines (SVM) [47] method, uses the features in data to learn a hyperplane that all samples lie on or are close to. The model can predict the values of new query points by computing the hyperplane equation, which is a function on the query features. In kernelized SVM, the input data is mapped into a higher dimensional space using a kernel function, which allows the samples to be more easily aligned to the hyperplane. The parameters of the kernel are learned such that the data mapped into the new feature space can be linearly represented.

4.4.3 Nearest Neighbor Regression

The nearest neighbor regression (NNR) algorithm is a non-parametric method introduced by N. S. Altman [48]. Unlike the previously discussed methods, it does not make any assumptions about the distribution of data (*e.g.*, LR assumes that the data is distributed approximately along a straight line). Instead, at prediction time, the algorithm finds the closest k neighbors in the training set for each query sample and predicts the regression value as a function of the actual regression values of the identified neighbors. The NNR implementation in the *sklearn* library, which we used in our experiments, allows for three different ways to combine the regression values of neighbors. Uniform weighting assigns all the neighbors the same weight, computing the query regression value as the mean of all *k* neighbor regression values. Distance-based weighting uses the inverse of distance from query of each neighbor as coefficients for the neighbor regression values. This results in neighbors closer to the query influencing the predicted query regression value much more than other neighbors. The third option is to provide a custom weight distribution vector to the algorithm. Finally, the proximity between points can be measured using different functions, such as the L2 distance or cosine similarity. In the case of a similarity proximity function, similarity values, which range between 0 and 1, are transformed into distances by subtracting them from 1.

4.5 Hybrid Prediction Models

In Section 3.2, we established that creatinine concentrations between 0 and 4 mg/dL result in major changes in eGFR values (Fig. 5). Also, the critical eGFR values of 15, 30, and 60 fall in the creatinine concentration range of 0.4–3.5 mg/dL for all age, ethnicity and gender combinations (refer Fig. 6). We hypothesize that we can improve eGFR classification performance by splitting the regression prediction task among several concentration ranges. For each range, we build specialized prediction models that focus on a subset of the samples with creatinine concentrations in the given range. In our experiments, we built regression models for four different concentration bins, choosing the same ranges as the ones used during data collection. When a new query sample is received, it is first classified into one of the four concentration bins. Then the specific regression model for that bin is used to predict the creatinine concentration. Finally, the MDRD equation is used to determine the eGFR classification given the predicted creatinine concentration.

We experimented with three different classification algorithms, support vector classification, nearest neighbor classification, and logistic regression classification, to find the classifier-regression combination that gives the least error for each time point. The three regression algorithms from the baseline methods are used to train for the concentration bin-specific regression models.

4.5.1 Support Vector Classifier

The support vector classifier (SVC) uses the SVM algorithm to find a hyper plane which maximizes the distance between the closest points belonging to different classes. For a two class classification problem, the support vectors are made up of the positive and negative class objects closest to the separating boundary. A hyperplane that maximizes the distance between the support vectors is the optimal choice for a decision boundary that can distinctively separate the two classes. For a dataset with more than two classes, a one-versus-all approach is often used, where objects of one class are considered positive and objects of all other classes are considered negative. A different model is learned for predicting each class. At prediction time, the class of the class-specific model with the highest confidence is chosen.

4.5.2 Nearest Neighbor Classifier

The concept behind the nearest neighbor classification (NNC) algorithm is the same as that of the nearest neighbor regression detailed in Section 4.4.3. After the k neighbors are found, the final class is decided based on the votes of each of the neighbors. Generally, the class represented by the majority of the neighbors is predicted as the class of the query. This method of choosing the prediction class is called majority voting. Alternatively, the query class can be computed as the linear combination of the neighbor classes and a weight vector consisting of the similarities between the query and the neighbors.

4.5.3 Logistic Regression Classifier

Logistic regression (LR) is a classification algorithm that predicts, for each query sample, the probability that it belongs to each of the predefined classes. In most cases, the model learns the hyper-parameters of a linear equation, as in the case of a linear regression model. The regression values are then transformed using the logistic sigmoid function to a range of 0 to 1. Functions with a characteristic S-curve are termed as sigmoid functions. The sigmoid function used for logistic regression is given by,

$$S(x) = 1/(1+e^{-x}),$$

where x is the input regression value.

5 Experiment Design

5.1 Data

We have created and used two datasets in this project. The localization dataset consists of 1121 images of teststrips taken in different backgrounds and lighting conditions. The images are 416×416 pixels in dimension and are in JPEG format. The concentration prediction dataset contains images from the chemical experiments conducted to capture the magnitude of color change observed in Jaffe's reaction. For each of the 65 test creatinine concentrations we identified in Section 3.2, the experiment was carried out using four different teststrips. For each teststrip, an image was captured before applying the creatinine solution to the teststrip. Subsequently, two images were captured 2, 12, 22, and 32 minutes after the application of the solution. Table 3 summarizes the distribution of the collected creatinine teststrip samples. Images in the concentration prediction dataset are in PNG format, have a resolution of 3024×4032 pixels, and each takes up 15.5 MB of storage.

Table 3
Images Collected as Part of Our Experiment

Conc. bin	**Conc. gap**	**# Strips**	**# Time points per teststrip**	**# Duplicates per time point**	**Total**
0-4 mg/dL	0.1 mg/dL	41	4	2	369
4.5-8 mg/dL	0.5 mg/dL	8	4	2	72
9-20 mg/dL	1 mg/dL	12	4	2	108
30-60 mg/dL	10 mg/dL	4	4	2	36

All 2340 teststrip images from the concentration prediction dataset were converted into three feature matrices using the feature extraction methodologies described in Section 3.2. The number of features resulting from each of the three feature extraction methods is given in Table 4.

In our paper, we use the term sample to refer to an image with a unique teststrip number, creatinine concentration, and taken at a particular time point after applying the

Table 4
Types of Features Extracted from Teststrip Images

Feature type	**Number of features per image**
Histogram of colors	375
Histogram of gradients	325
RGB pixels	12288

creatinine solution. We partitioned our concentration prediction datasets into 5 groups for each time point, where the first four groups use data from each of the four teststrip sets and the fifth combines all of the data from all four teststrips. We trained all models using each group of teststrips. Fig. 13 shows the different combinations of experiments possible given our data grouping, feature, and model selection choices. Overall, we have trained 480 different models for different combinations of data groups (5), time points (4), and regression algorithms (12).

5.2 Teststrip Localization

For training and testing the YOLO model, we created a dataset of 1211 images of size 416×416 pixels. To ensure the robustness of the model, we took pictures of teststrips with different creatinine concentrations in various lighting and orientation conditions, while posing the samples on a variety of backgrounds. As suggested by Nils et al. in [49], all the images were annotated using the BBox-Label-Tool [45], a Python graphical user application that stores the bounding boxes drawn on the images as pixel coordinates in individual text files. YOLO learns the center coordinates, height, and width of a prediction object as ratios of dimensions of the image. As an example, an object of dimensions (10×40) pixels with center at (10, 10) in a (100×200) image will be localized within the bounding box coordinates (0.1, 0.05, 0.1, 0.2).

We trained and tested the model using the CUDA enabled *darknet* framework [40]. Instead of using random initial weights, we initialized the model using pre-trained ImageNet classification weights provided by the creators of *darknet* [50]. The data were

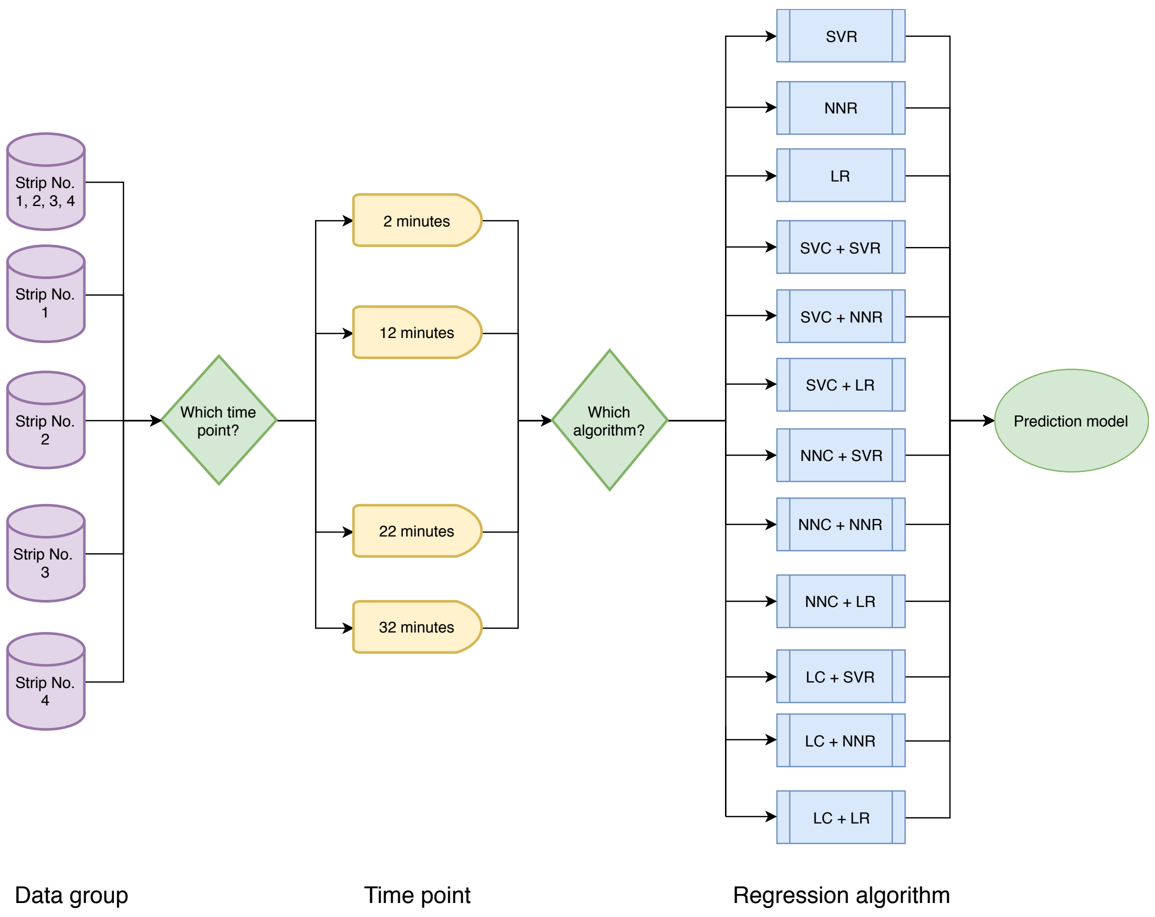

Fig. 13. Logical flow for model selection in our experiments.

split into training and test sets using the ratio 80:20. We further split the training set into train and validation, using the ratio 90:10, and used the validation dataset to tune parameters during the training. The *darknet* framework provides a configuration file to define the inputs and tune the parameters when training. The network was trained in batches of 64 images with a learning rate of $1E-5$. Model training reached an RMSE loss of 2.56 in 900 iterations.

We tested our model on both the train and test datasets, which had 938 and 183 samples, respectively. As our tests were performed on only positive images, i.e., the teststrip was always present in the image, we chose F1 score as our prediction performance measure, as it is well suited for imbalanced data.

In order to measure localization performance, we compute the intersection over union (IOU) score between the ground truth and predicted bounding boxes. The IOU is computed as the ratio of the area of overlap between the predicted and the actual bounding boxes and the union area of the two boxes. In our experiment, the predictions with an IOU score greater than or equal to 0.5 are classified as true positives (T_p). All other predictions are classified as false positives (F_p). Since we trained the model to detect presence of only one class of objects (teststrip), and we did not test the system with images that do not contain the teststrip, there are no false negatives (F_n). We computed precision, recall and F1 score for the predictions using the formulae,

$$Precision(P) = T_p/(T_p + F_p),$$

$$Recall(R) = T_p/(T_p + F_n),$$

$$F1score(F1) = 2 * P * R/(P + R).$$

5.3 Execution Environment

All experiments were executed on a server with two 12-core 2.5 GHz Intel Xeon E52680 v3 (Haswell) processors and 384 GB RAM. While all algorithms employed in our analysis were implemented as serial methods in their respective Python and OpenCV packages, we used naïve parallelization techniques to execute up to 24 experiments at the same time. This allowed efficient execution of all experiments and multiple iterations over the machine learning experimental design.

5.4 Machine Learning Model Validation

We applied 10-fold cross-validation for all prediction models. This technique allows all the samples to be used for both training and validation, and ensures that each sample is validated only once. The technique helps verify whether a model will be generalizable, even in the absence of a large number of test samples. During 10-fold cross-validation, the dataset is randomly shuffled and split into 10 equal groups. For each iteration, the i^{th} group of samples is considered as the test set and the other nine groups are combined to make up the training set. The RMSE of all the predictions is used as the performance measure of the algorithm applied to the dataset.

For each model, we tuned meta-parameters using the GridSearchCV function from the *sklearn* [51] library. It performs an exhaustive search of the parameter space, finding the set of parameters that best fit the training samples for all our classification and regression models. To tune our support vector regression (SVR) models, we optimize the choice of kernel functions among the sigmoid, radial basis and polynomial kernels, and the prediction penalty, C, testing values from the set $\{2^{-4}, 2^{-2}, 2^{0}, 2^{2}, 2^{4}\}$. In addition, for our support vector classification (SVC) models, we tuned the kernel coefficient, γ, testing values from the set $\{1E{-}4, 1E{-}2, 1E{+}0, 1E{+}2, 1E{+}4\}$. For the nearest neighbor models, we chose the best number of neighbors closest to the query, k, in the search space of $\{1, 2, 3, 4, 5\}$ neighbors. For the logistic regression classifier models, we tested different optimization algorithm, such as *newton-cg*, *liblinear*, and *lbfgs*, and tuned the inverse regularization factor, C, testing values in $\{2^{-4}, 2^{-2}, 2^{0}, 2^{2}, 2^{4}\}$.

To understand the impact of the errors in our prediction models, we computed eGFR values for each of the predicted and actual creatinine concentration values and classified them as critical, intermediate and healthy. The accuracy of this classification shows the effectiveness of our mobile application in guiding our patients. To compute the eGFR values using the MDRD equation, we need to know the gender, ethnicity and age of patients. For this experiment, we assigned gender, ethnicity and age values for each sample in the dataset, distributed based on the US census reports for gender, population and ethnicity distributions from the year 2016 [52], [53]. Table 5 summarizes the population distributions and associated number of samples that were assigned to each population segment in our dataset.

Table 5
Distribution of Age, Gender and Ethnicity in the U.S. Population and the Number of Teststrips Associated with Each Population Segment

	Male, African		**Female, African**		**Male, non-African**		**Female, non-African**	
Age group	**% pop.**	**# strips**	**% pop.**	**# strips**	**% pop.**	**# strips**	**% pop.**	**# strips**
0-15	1.37	18	1.31	17	8.93	116	8.56	111
16-30	1.40	18	1.34	17	9.12	119	8.74	114
31-45	1.28	17	1.28	17	8.33	108	8.36	108
46-60	1.30	17	1.36	18	8.49	110	8.84	115
61-75	0.89	12	1.00	13	5.79	75	6.52	85
76-90	0.29	4	0.40	5	1.87	24	2.62	34
0-15	0.02	0	0.06	1	0.16	2	0.37	5

6 Results and Discussions

In this section, we present and comment upon the results of the experiments. First we discuss the results of our localization model and the viability of using this model in KHM to position and orient the teststrips. Next, we discuss the results of the experiments conducted on various baseline and regression models to find the best performing feature extraction technique, time point, and regression model for our dataset. Finally, we discuss the reproducibility of the chemical experiments conducted to validate the robustness of the kidney health monitoring system we are proposing.

6.1 Localization Effectiveness

Table 6 summarizes the experimental results obtained for our YOLO model trained to detect and localize the teststrips. The model has achieved a high F1 score of 0.972 on both the test and train sets.

Table 6
Teststrip Localization Results Using YOLO Model

	Test set	**Train set**
No. of samples	183	938
True positives	173	887
False positives	10	51
Precision	0.94535	0.94562
Recall	1.0	1.0
F1 score	0.97191	0.97205

We classified predictions with an intersection over union (IOU) of 0.5 or above as true positives and the rest as false positives in our experiment. Fig. 14 shows the variation of the IOU scores for the sample distribution. We can observe that more than 85% of the samples have an IOU score above 0.85, making the model consistent in its prediction. However, there is a steep decline in the IOU scores in the tail end of the distribution. This could be due to underrepresentation of several backgrounds in the training set. Overall, the performance of the localization model is satisfactory. This model is expected to perform well in detection and localization of teststrips in various background and lighting conditions. Using this model, our application will be able to guide the user in placing the teststrip at an optimal distance and orientation that will lead to quality image captures.

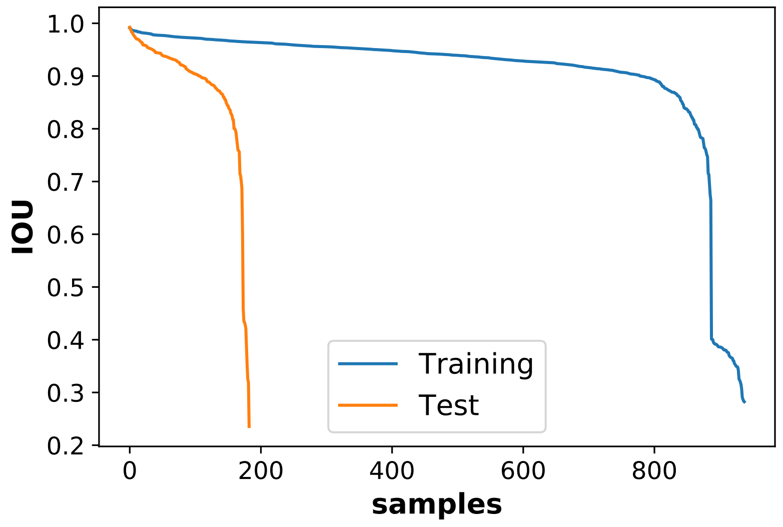

Fig. 14. Intersection of Union (IOU) of predicted and actual bounding boxes for test and train samples.

When using our localization model on an iPod Touch device, it exhibits an efficiency of one frame every 7 seconds, which is much slower than the minimum frames per second required for the human eye to perceive motion (12–15 frames per second). In the future, we plan to optimize model execution on mobile devices by using network sparsification and integer-based modeling techniques. We expect that these optimizations can achieve a localization efficiency of up to 45 frames per second on consumer GPUs and 10–15 frames per second on Apple devices with built-in GPUs.

6.2 Regression and Classification Experiments

Experiments were conducted to identify the best feature extraction method, image capture time point, and prediction model combination that consistently give high performance. We executed experiments using the different training data groups discussed in Section 5.1. First, we identified the feature extraction technique that best captures color changes in the detection zone. Then, for those type of features, we analyzed the performance of different machine learning models at different time points following the application of the creatinine solution to the teststrip. Appendix A summarizes the results of the creatinine concentration prediction and eGFR range classification experiments on which we base the conclusions drawn in this section.

6.2.1 Best Performing Features

To gauge whether the features extracted have been effective in representing the creatinine concentrations, we performed t-tests for the hypothesis that the predicted results are similar to those of randomly chosen floating point values in the same range as our test sample predictions. Here, we present significance values for models built using each feature type, based on the best performing regression algorithm for each feature type. The hypothesis is considered invalid for a p-value less than $1E{-}4$. From Table 7, we can see that all three models are significant, as the p-values are well below the chosen threshold. Thus, we conclude that our feature extraction techniques are statistically significant.

Table 7
T-test Results for Best Models of Different Feature Types

	t-statistic	**p-value**
HOC	7.4399	$1.7773E{-}11$
HOG	7.4824	$1.4271E{-}11$
RBG	8.1241	$4.9891E{-}13$

To find the best performing feature type, we compared the average eGFR accuracy across all the time points, regression models, and data groups for each feature type. We observed that the HOC features obtain the highest average accuracy, 74.04%. Table 8 shows the average eGFR accuracies for all three feature types.

HOG features primarily hold information about spatial distribution of color intensities. The poor performance of HOG features in the experiments prove that spatial spread of color intensities is not a significant factor in predicting the concentration of creatinine.

Table 8
Average eGFR Accuracies for Different Feature Types

Feature Type	Average eGFR Accuracy
HOC	74.04%
HOG	39.02%
RBG	60.04%

Table 9
Average eGFR Accuracies and Standard Deviation for Different Teststrip Groups

Feature Type	Average eGFR Accuracy	Standard deviation
All	75.85%	14.5115
Strip 1	74.82%	13.1612
Strip 2	70.97%	12.5995
Strip 3	72.63%	14.4157
Strip 4	75.95%	13.8433

The prediction RMSE and eGFR accuracy of models using RGB features are better than those of models using HOG features, but are not on par with those using HOC features. This could be because all the information available in the RGB channels has been made available to the models without filtering out unnecessary features. RBG channels encode both lighting and color information linearly, making it difficult to distinguish the color changes introduced by brightness and saturation changes. In the remainder of our analysis, we consider only the results of using HOC features as input to the different models.

We repeated each experiment four times, creating four teststrips for each concentration. To test the repeatability of experiments, we fit the regression models to each of these four teststrip groups. We noted some irregularities in the performance of the models across the groups, which is further discussed in Section 6.3. Table 9 captures the average eGFR accuracies and the standard deviations of the predictions.

Upon close inspection, we found that there were visible differences in the color observed in the detection zone of some teststrips even before application of the creatinine solution. Fig. 15 shows the color of the reagent, picric acid, before the creatinine solution was introduced. We see that there is a visible difference in coloring of the detection zone, where both the teststrips are expected to display the same coloring for the same concentration of picric acid solution. This could be a possible reason for the difference in prediction accuracies between the four teststrip groups. We decided to use the results from teststrip group 4 for further analysis, as it has the best average accuracy and comparatively less standard deviation (Table 9).

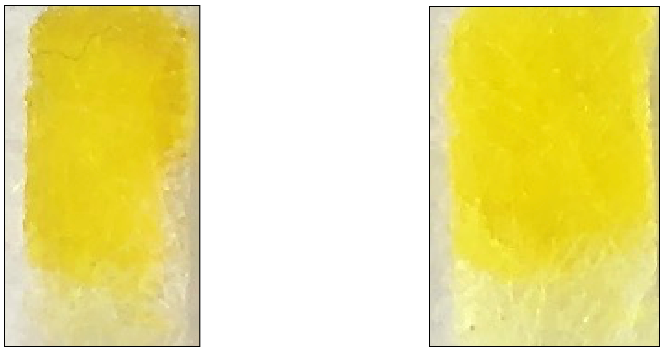

Fig. 15. Difference in detection zone color in two teststrips before adding the creatinine solution.

6.2.2 Best Time Point Identification

To find the time point at which the models perform the best, we compared the eGFR accuracy and prediction RMSE for all the regression models fit to teststrip 4 samples. Fig. 16 and 17 show the eGFR accuracy and RMSE values for each of the four time points for the different regression algorithms applied. Eight of the twelve models perform their best in eGFR classification at 12 minutes. Thus, the patient has to wait for 12 minutes before the system can accurately predict their blood creatinine levels and kidney health. This is a considerably higher wait time than usually needed when using PoCTs available in the market, which take a maximum of 2 minutes to predict the creatinine concentration (Table 1). In a future study, we will further investigate the optimal wait time, which may be between 2 and 12 minutes.

To further understand the effect of time on HOC features, we analyzed images taken of a teststrip with creatinine concentration of 1.5 mg/dL at the four time intervals. Fig. 18

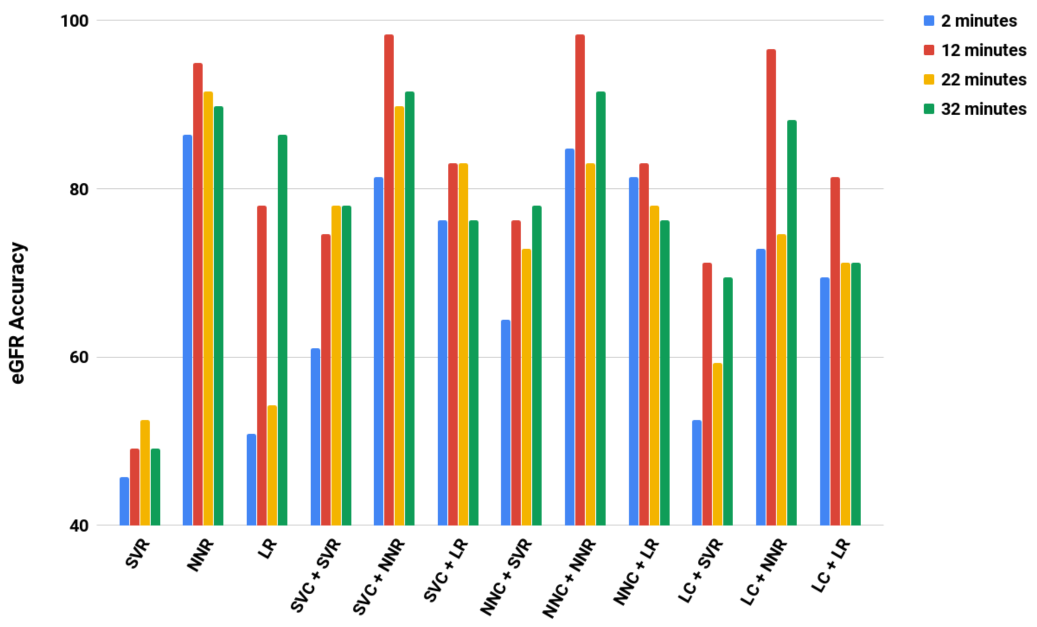

Fig. 16. Performance of eGFR classification models at four time points.

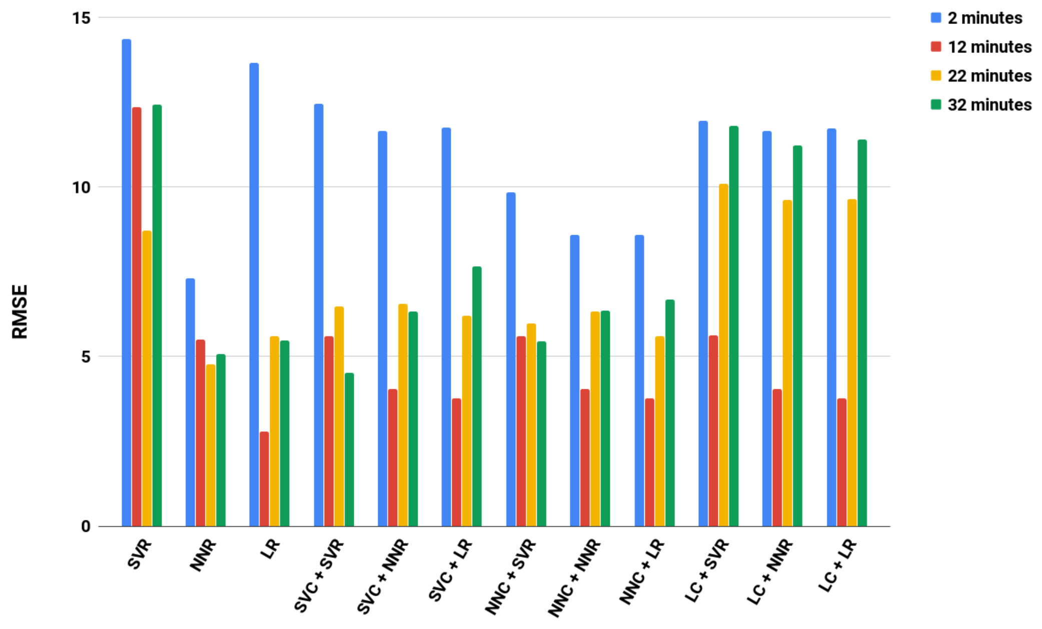

Fig. 17. Performance of regression models at four time points.

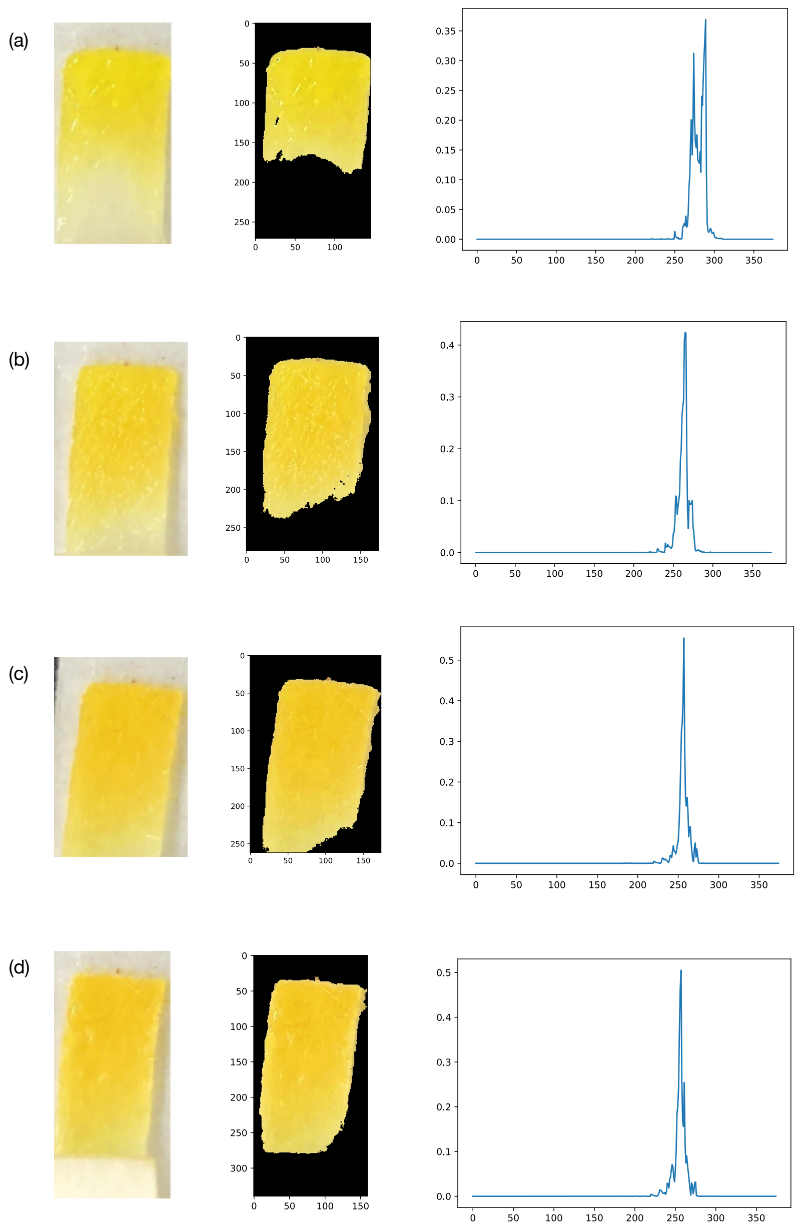

Fig. 18. Image of detection zone (left), after masking pixels (center), and the histogram of colors (right) of a teststrip at (a) 2 minutes, (b) 12 minutes, (c) 22 minutes, and (d) 32 minutes.

shows the histogram of a teststrip taken at the four time points. It is evident that, at 2 minutes, the color distribution is wider, 95% of pixels being spread between buckets 260 to 295. However, at 12 minutes, the majority of the pixels fall in the smaller bucket range of 255 to 270. The distribution narrows further as time increases to 22 minutes and 32 minutes, with buckets 255 to 260 holding 95% of the pixels. This slow change in color could be attributed to the strength of the chemicals used or the drying of the picric acid solution. We need to experiment more with different concentrations of picric acid solutions and teststrip drying pre-processing time.

6.2.3 Best Performing Regression Model

To determine the best performing prediction model for our data, we analyzed the performance of the baseline and hybrid models at different time points for models trained using HOC features. Fig. 16 shows that the support vector classification hybrid model combined with nearest neighbor regression (SVC + NNR) and nearest neighbor classification hybrid model combined with nearest neighbor regression (NNC + NNR) give the best eGFR accuracy of 98.30%. T-tests performed on both the SVC + NNR and NNC + NNR model results returned p-values of $1.78E-11$ each, which are well below the $1E-4$ threshold for rejecting the null hypothesis.

Results show that the baseline linear regression model achieves the lowest RMSE of 2.78. However, the eGFR accuracy for that model is 74.57%. On the other hand, both the SVC + NNR and NNC + NNR models obtained a higher RMSE of 4.04, but much better eGFR accuracies of 98.30%. This shows that the approach adopted in the hybrid model, that is, estimating the range of creatinine concentration first and then predicting the creatinine concentration value using specialized models performs better in the task of eGFR classification. While the classification performance is better in the hybrid models, the actual regression value prediction is, in general, slightly worse. This could be because the number of training samples in some of the concentration bins is small (*e.g.*, concentration bin 30 - 60 mg/dL has only four training samples), leading to more regression prediction errors. This affects the creatinine concentration value predictions and the RMSE negatively, but it does not affect the eGFR accuracy. Given more data, we anticipate the hybrid model would have lower RMSE than the baseline model.

6.3 Repeatability of Chemical Experiments

To test the repeatability of the chemical experiments, we computed the similarity of pairs of teststrip images, using the cosine similarity proximity function and HOC features, in two scenarios. First, we used pairs of images captured from the *same teststrip*, which we expect will display very high similarity in general. Second, we computed similarities of *different teststrips* that had the same creatinine solution applied to them. Given the same amount of picric acid in the teststrip and the same concentration in the creatinine solution, we expect that the teststrips would exhibit the same level of colorometric change. Fig. 19 plots the distribution of similarity values obtained in both cases.

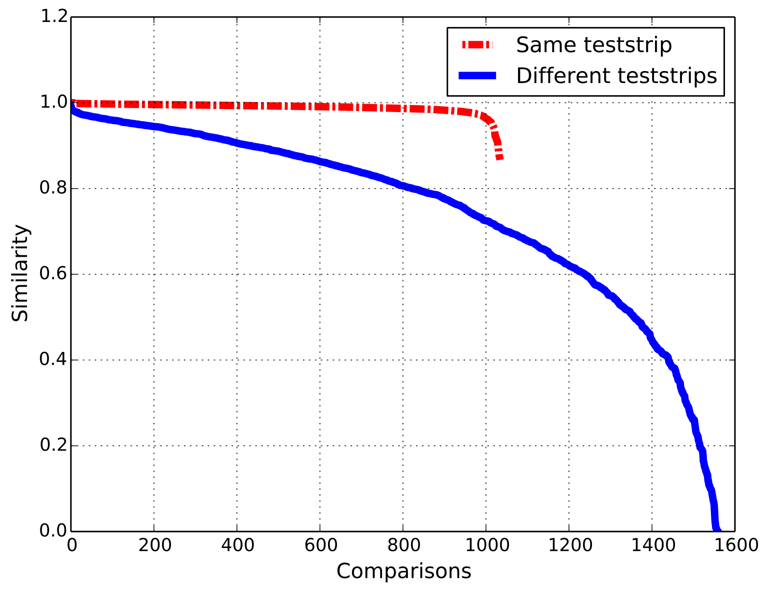

Fig. 19. Pairwise cosine similarity of teststrips.

It is evident that, while images of the same teststrip do indeed have very high similarities, as expected, different teststrips have much lower similarities in general.

Almost 50% of the teststrips with the same creatinine concentration have a similarity below 0.8, and almost 20% have a similarity below 0.5. The results imply that the chemical experiments are not robust and do not always produce consistent results for the same creatinine concentration. Low similarities between different teststrips with the same creatinine concentrations could be due to variations in delay of starting the creatinine experiments after application of picric acid solution to the detection zone. Another cause for the divergent similarities could be the existence of impurities in the teststrip materials or the chemical substances used in the chemical experiment. Finally, it is possible that the amount of picric acid applied to each teststrip was not consistent, leading to variability in the subsequent chemical reactions. As an example that motivates this hypothesis, Fig. 15 shows the visible difference in color between the pictures of two teststrips taken before applying the creatinine solution. The difference in color, which can be clearly seen with the naked eye, can only be attributed to differences in the experiment setup, not the amount of creatinine applied to the teststrip.

7 Future Work

While the models we trained on teststrip samples created with synthetic creatinine have been able to give an accuracy of up to 98.30% for eGFR range classification, further work is needed before the technology could be used to help humans monitor kidney function. A necessary next step in the research involves confirming that our methods would work using blood samples. The tests can first be done using human-analog porcine blood samples mixed with creatinine solution, before moving on to human trials.

In the current experiment, we eliminated certain potential sources of error in order to first validate whether computer vision techniques could be used to classify kidney health stages. Our images were captured using consistent light warmth and brightness levels to avoid changes in color due to ambient light. The histogram of color features extracted in this work partially addresses this issue. However, the robustness of these features to changes in light conditions needs to be further investigated.

In the current work, we did not experiment with deep neural network techniques for feature extraction and regression models. This was due to the limited number of samples that we had available, which could overfit deep learning models and lead to poor generalization. Given additional chemical experiment results, multi-task deep neural networks that localize and predict the concentration at the same time would be another interesting area to explore.

8 Conclusions

The prevalence of chronic didney disease is estimated to increase from 13.2% in 2014 to 14.4% in 2020 and 16.7% in 2030 in adults of 30 years of age or older [54]. The best way to slow this trend is early detection and constant monitoring of the disease. Our research proposes and validates a mobile phone-based system for monitoring kidney health from our homes.

We proposed a mobile phone-based system that helps users capture quality images of teststrips using a deep neural network model, and several feature extraction and kidney health stage prediction methods that rely on computer vision and machine learning techniques. We conducted experiments in a light-controlled environment, using synthetic creatinine solution on custom teststrips specially designed for our study. Our models classify the severity of the kidney disease as normal, intermediate risk, or kidney failure based on the prediction of the concentration of creatinine in the sample. The histogram of colors features, constructed using range-constrained hue and saturation values for pixels in the image, have achieved RMSE of 4.05 and eGFR range classification accuracy of 98.30% on a randomized population assignment that simulates the U.S. population distribution.

Overall, our experiments confirm that mobile phone-based kidney health monitoring systems are feasible. However, more work is needed to develop robust models that can be used by the general public in a variety of settings.

Literature Cited

[1] T. N. Kidney and U. D. I. C. (NKUDIC), "Kidney disease statistics for the united states," U.S. Department of Health and Human Services; National Institute of Diabetes and Digestive and Kidney Diseases, 3 Kidney Information Way, Bethesda, MD 20892, Tech. Rep. 12-3895, 6 2012.

[2] F. L. T. Online. Bun/creatinine ratio in online lab tests stores. Accessed 2018-03-30. [Online]. Available: https://www.findlabtest.com/lab-test/kidney-function-test/bun-creatinine-ratio-quest-296

[3] 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. [Online]. Available: http://clinchem.aaccjnls.org/content/53/4/766

[4] C. L. S. L.L.C, "Laboratory procedure manual; creatinine refrigerated serum," Center For Disease Control and Prevention (CDC), Tech. Rep. NHANES 2011-2012, 2012.

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

[6] A. S. John and C. P. Price, "Existing and emerging technologies for point-of-care testing," *The Clinical Biochemist Reviews*, vol. 35-3, pp. 201–213, 8 2014.

[7] N. Biomedical. Statsensor, point-of-care whole blood creatinine and egfr testing. Accessed 2018-04-21. [Online]. Available: http://www.novabio.us/statstrip-creatinine/

[8] ——. Statstrip, glucose monitoring system. Accessed 2018-04-21. [Online]. Available: http://www.novabio.us/statstrip-glu/

[9] J. J. Lingane, "Coulometric analysis," *Journal of the American Chemical Society*, vol. 67, no. 11, pp. 1916–1922, 1945. [Online]. Available: https://doi.org/10.1021/ja01227a013

[10] M. D. Shephard, "Point-of-care testing and creatinine measurement," *The Clinical Biochemist Reviews*, vol. 32-2, pp. 109–114, 5 2011. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3100277/

[11] A. W. Company. Istat handheld blood analyzer abbott 04j4850. Accessed 2018-06-11. [Online]. Available: https://www.aaawholesalecompany.com/abb-04j48-50-ea.html

[12] I. Block Scientific. Radiometer abl800 flex blood gas analyzer. Accessed 2018-06-11. [Online]. Available: https://www.blockscientificstore.com/ABL800-FLEX-p/abl800-flex.htm

[13] labexchange. Nova biomedical statsensor. Accessed 2018-06-11. [Online]. Available: https://shop.labexchange.com/en/nova-biomedical-statsensor.html

[14] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," vol. 1, pp. I–511–I–518 vol.1, 2001.

[15] R. B. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," *CoRR*, vol. abs/1311.2524, 2013. [Online]. Available: http://arxiv.org/abs/1311.2524

[16] C. Gu, J. Lim, P. Arbelaez, and J. Malik, "Recognition using regions," vol. 0, pp. 1030–1037, 06 2009.

[17] R. Girshick, "Fast r-cnn," pp. 1440–1448, Dec 2015.

[18] S. Ren, K. He, R. B. Girshick, and J. Sun, "Faster R-CNN: towards real-time object detection with region proposal networks," *CoRR*, vol. abs/1506.01497, 2015. [Online]. Available: http://arxiv.org/abs/1506.01497

[19] J. H. Hosang, R. Benenson, and B. Schiele, "How good are detection proposals, really?" *CoRR*, vol. abs/1406.6962, 2014. [Online]. Available: http://arxiv.org/abs/1406.6962

[20] J. H. Hosang, R. Benenson, P. Dollár, and B. Schiele, "What makes for effective detection proposals?" *CoRR*, vol. abs/1502.05082, 2015. [Online]. Available: http://arxiv.org/abs/1502.05082

[21] 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. [Online]. Available: http://arxiv.org/abs/1506.02640

[22] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. E. Reed, C. Fu, and A. C. Berg, "SSD: single shot multibox detector," *CoRR*, vol. abs/1512.02325, 2015. [Online]. Available: http://arxiv.org/abs/1512.02325

[23] J. Redmon and A. Farhadi, "YOLO9000: better, faster, stronger," *CoRR*, vol. abs/1612.08242, 2016. [Online]. Available: http://arxiv.org/abs/1612.08242

[24] G. VALENSI, "System of television in colors," Patent US2 375 966 (A), 05, 1945.

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

[26] B. A. Olshausen and D. J. Field, *Nature*, vol. 38I, p. 607 EP, 1996. [Online]. Available: http://dx.doi.org/10.1038/381607a0

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

[28] J. van de Weijer, T. Gevers, and A. Gijsenij, "Edge-based color constancy," *IEEE Transactions on Image Processing*, vol. 16, no. 9, pp. 2207–2214, Sept 2007.

[29] J. T. Barron, "Convolutional color constancy," *CoRR*, vol. abs/1507.00410, 2015. [Online]. Available: http://arxiv.org/abs/1507.00410

[30] S. Bianco, C. Cusano, and R. Schettini, "Single and multiple illuminant estimation using convolutional neural networks," *CoRR*, vol. abs/1508.00998, 2015. [Online]. Available: http://arxiv.org/abs/1508.00998

[31] 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.

[32] M. Engilberge, E. Collins, and S. Ssstrunk, "Color representation in deep neural networks," 2017.

[33] 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. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0003267015011174

[34] 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)*, April 1998, pp. 198–204.

[35] 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.

[36] K. M. Koczula and A. Gallotta, "Lateral flow assays," *Essays in Biochemistry*, vol. 60-1, pp. 111–120, 6 2016.

[37] A. Inc. Metal 2, accelerating graphics and much more. Accessed 2017-07-11. [Online]. Available: https://developer.apple.com/metal/

[38] ——. Coreml - machine learning framework for ios, macos, tvos, watchos. Accessed 2017-10-2. [Online]. Available: https://developer.apple.com/documentation/coreml

[39] "World wide developers conference, apple," San Jose, California, 6 2017. [Online]. Available: https://developer.apple.com/videos/wwdc2017/

[40] J. Redmon. (2013–2016) Darknet: Open source neural networks in c. Accessed 2017-09-11. [Online]. Available: http://pjreddie.com/darknet/

[41] A. Zelener. (2017, 2) Yad2k: Yet another darknet 2 keras. https://github.com/allanzelener/YAD2K. Accessed 2017-09-11.

[42] A. Inc. (2017, 8) Coreml tools. https://github.com/apple/coremltools. Accessed 2018-02-13.

[43] S. University. Developing ios 10 apps with swift. Accessed 2017-07-11. [Online]. Available: https: //itunes.apple.com/us/course/developing-ios-10-apps-with-swift/id1198467120

[44] A. Inc. Start developing ios apps (swift). Accessed 2017-07-11. [Online]. Available: https://developer.apple.com/library/content/referencelibrary/\GettingStarted/ DevelopiOSAppsSwift/

[45] S. Qiu. (2014, 6) Bbox-label-tool. https://github.com/puzzledqs/BBox-Label-Tool. Accessed 2017-09-11.

[46] G. Bradski, “The OpenCV Library,” *Dr. Dobb’s Journal of Software Tools*, 2000.

[47] B. E. Boser, I. M. Guyon, and V. N. Vapnik, “A training algorithm for optimal margin classifiers,” in *Proceedings of the Fifth Annual Workshop on Computational Learning Theory*, ser. COLT ’92. New York, NY, USA: ACM, 1992, pp. 144–152. [Online]. Available: http://doi.acm.org/10.1145/130385.130401

[48] N. S. Altman, “An introduction to kernel and nearest-neighbor nonparametric regression,” *The American Statistician*, vol. 46, no. 3, pp. 175–185, 1992. [Online]. Available: https://www.tandfonline.com/doi/abs/10.1080/00031305.1992.10475879

[49] N. Tijtgat. (2017, 5) How to train yolov2 to detect custom objects. https://timebutt.github.io/static/how-to-train-yolov2-to-detect-custom-objects/. Accessed 2017-09-11.

[50] J. Redmon. (2013–2016) Imagenet pre-trained models. https://pjreddie.com/darknet/imagenet/.

[51] scikit learn.org. sklearn.model_selection.gridsearchcv. Accessed 2018-05-11. [Online]. Available: http://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV

[52] U. S. C. Bureau. (2016, 6) Annual estimates of the resident population by single year of age and sex for the united states: April 1, 2010 to July 1, 2016. Accessed 2018-04-11. [Online]. Available: https://www.census.gov/data/datasets/2016/demo/popest/nation-detail.html

[53] ——. (2016, 6) Acs demographic and housing estimates. Accessed 2018-04-11. [Online]. Available: https://factfinder.census.gov/faces/tableservices/jsf/pages/productview.xhtml?src=CF

[54] T. J. Hoerger, S. A. Simpson, B. O. Yarnoff, M. Pavkov, N. Rios Burrows, S. Saydah, D. E. Williams, and X. Zhuo, “The future burden of ckd in the united states: A simulation model for the cdc ckd initiative,” *American Journal of Kidney Diseases*, vol. 65(3), 11 2014.

Appendix A

Baseline and Hybrid Model Prediction Results

In this appendix, we present prediction results from each of our baseline and hybrid models and for each tested feature type, for both the classification and regression tasks.

A.1 HOC Results

Table 10
Baseline eGFR Accuracy Using HOC Features

Strip	Time	SVR	NNR	LR
all	2	57.0866	90.9449	54.3307
1	2	50.8475	76.2712	62.7119
2	2	44.0678	79.6610	55.9322
3	2	40.6780	72.8814	42.3729
4	2	45.7627	86.4407	50.8475
all	12	66.9291	90.9449	35.8268
1	12	47.4576	88.1356	72.8814
2	12	50.8475	88.1356	57.6271
3	12	47.4576	79.6610	52.5424
4	12	49.1525	94.9153	77.9661
all	22	62.2047	92.1260	54.7244
1	22	55.9322	81.3559	44.0678
2	22	54.2373	77.9661	49.1525
3	22	52.5424	84.7458	52.5424
4	22	52.5424	91.5254	54.2373
all	32	**67.3228**	90.9449	44.0945
1	32	52.5424	93.2203	61.0169
2	32	47.4576	93.2203	71.1864
3	32	49.1525	**96.6102**	72.8814
4	32	49.1525	89.8305	**86.4407**

Table 11
Baseline Regression RMSE Using HOC Features

Strip	Time	SVR	NNR	LR
all	2	13.1615	5.8486	>100
1	2	13.8943	7.4015	12.4347
2	2	14.2259	8.3555	10.4507
3	2	13.9656	7.4649	12.4208
4	2	14.3553	7.3045	13.6604
all	12	11.4706	2.4535	>100
1	12	11.8072	5.4156	4.5515
2	12	12.1980	4.7204	6.4438
3	12	11.8950	4.4214	4.2341
4	12	12.3671	5.5044	2.7876
all	22	9.3536	**2.4385**	>100
1	22	9.8538	5.0899	6.3018
2	22	**8.7003**	4.1025	5.6368
3	22	9.6625	4.4202	5.5128
4	22	8.7187	4.7848	5.6042
all	32	11.5707	3.4556	>100
1	32	11.8623	4.7485	6.4007
2	32	12.2381	5.7777	3.2636
3	32	11.9201	3.7668	**2.1868**
4	32	12.4375	5.0630	5.4708

Table 12
Hybrid eGFR Accuracy Using HOC Features

Strip	**Time**	**SVC+ SVR**	**SVC+ NNR**	**SVC+ LR**	**NNC+ SVR**	**NNC+ NNR**	**NNC+ LR**	**LC+ SVR**	**LC+ NNC**	**LC+ LR**
all	**2**	76.4479	88.8031	72.5869	75.6757	88.4170	72.5869	66.4093	83.3977	67.5676
1	**2**	62.7119	84.7458	71.1864	64.4068	86.4407	74.5763	57.6271	79.6610	66.1017
2	**2**	55.9322	71.1864	67.7966	57.6271	72.8814	67.7966	50.8475	67.7966	66.1017
3	**2**	52.5424	77.9661	66.1017	61.0169	81.3559	69.4915	52.5424	76.2712	67.7966
4	**2**	61.0169	81.3559	76.2712	64.4068	84.7458	81.3559	52.5424	72.8814	69.4915
all	**12**	86.1004	89.5753	63.7066	86.4865	89.1892	63.7066	85.7143	89.1892	62.1622
1	**12**	71.1864	91.5254	79.6610	69.4915	89.8305	81.3559	64.4068	84.7458	77.9661
2	**12**	69.4915	84.7458	83.0508	69.4915	86.4407	83.0508	64.4068	83.0508	77.9661
3	**12**	71.1864	79.6610	67.7966	69.4915	79.6610	67.7966	64.4068	76.2712	62.7119
4	**12**	74.5763	**98.3051**	83.0508	76.2712	**98.3051**	83.0508	71.1864	**96.6102**	81.3559
all	**22**	78.7645	91.5058	60.6178	78.7645	91.5058	61.3900	72.5869	86.4865	59.4595
1	**22**	71.1864	81.3559	84.7458	69.4915	79.6610	83.0508	57.6271	72.8814	74.5763
2	**22**	72.8814	86.4407	84.7458	69.4915	83.0508	81.3559	52.5424	67.7966	67.7966
3	**22**	77.9661	88.1356	84.7458	79.6610	89.8305	86.4407	64.4068	77.9661	72.8814
4	**22**	77.9661	89.8305	83.0508	72.8814	83.0508	77.9661	59.3220	74.5763	71.1864
all	**32**	**87.6448**	92.2780	67.1815	**88.8031**	92.2780	68.3398	**88.4170**	92.6641	67.1815
1	**32**	74.5763	93.2203	**93.2203**	74.5763	93.2203	**93.2203**	71.1864	89.8305	**89.8305**
2	**32**	71.1864	86.4407	72.8814	71.1864	89.8305	76.2712	67.7966	84.7458	71.1864
3	**32**	79.6610	94.9153	88.1356	81.3559	94.9153	89.8305	71.1864	89.8305	86.4407
4	**32**	77.9661	91.5254	76.2712	77.9661	91.5254	76.2712	69.4915	88.1356	71.1864

Table 13
Hybrid regression RMSE Using HOC features

Strip	**Time**	**SVC+ SVR**	**SVC+ NNR**	**SVC+ LR**	**NNC+ SVR**	**NNC+ NNR**	**NNC+ LR**	**LC+ SVR**	**LC+ NNC**	**LC+ LR**
all	**2**	6.3138	6.2177	87.2391	7.5043	7.4597	>100	8.6676	8.2170	>100
1	**2**	11.6947	11.1682	11.3136	11.4722	10.7639	10.4069	11.7237	11.1916	11.3578
2	**2**	12.5560	12.1704	12.1750	10.7634	10.1041	10.4526	11.9219	11.4316	11.6706
3	**2**	12.9296	12.2490	12.8142	7.5079	7.5539	8.1438	12.2782	11.7841	11.9912
4	**2**	12.4631	11.6458	11.7508	9.8408	8.5994	8.5852	11.9565	11.6471	11.7274
all	**12**	3.8395	**2.6319**	55.4155	3.8291	**2.6500**	46.1830	4.0122	**2.8055**	56.3161
1	**12**	4.9815	4.3750	4.7086	4.9922	4.3766	4.6875	8.8056	8.2281	8.9404
2	**12**	5.4167	5.9659	8.4094	5.6955	5.6162	8.1034	7.1017	7.1077	7.9951
3	**12**	3.9225	4.1410	4.8833	4.9599	4.2876	4.8743	8.7127	8.4605	9.0182
4	**12**	5.6096	4.0442	3.7601	5.6090	4.0471	3.7594	5.6361	4.0533	**3.7676**
all	**22**	**3.0142**	2.9948	>100	**2.7905**	2.8932	>100	5.1071	4.6870	>100
1	**22**	4.5031	4.7003	5.0424	5.5528	5.6619	5.8803	10.1574	9.4541	9.9533
2	**22**	4.7630	4.2806	**3.7316**	4.9731	4.4374	3.8974	8.9808	8.5276	9.1968
3	**22**	4.2862	3.7404	3.8296	3.9867	3.6062	3.6710	9.7162	9.3745	9.3679
4	**22**	6.4741	6.5613	6.2100	5.9871	6.3250	5.6095	10.0857	9.6157	9.6374
all	**32**	4.2686	3.0447	5.1313	4.4477	3.3105	5.1447	**3.8282**	2.9791	5.0045
1	**32**	6.6967	5.8110	5.8439	5.9618	5.6849	5.3879	9.7657	9.0184	9.4601
2	**32**	7.1546	5.6176	5.8667	5.6596	4.4605	3.2344	11.1022	9.9185	10.4837
3	**32**	4.9198	4.0607	3.9402	5.0787	3.1076	**3.1901**	9.6939	9.3067	9.2967
4	**32**	4.5142	6.3177	7.6672	5.4472	6.3448	6.6922	11.7922	11.2299	11.3988

A.2 RGB Results

Table 14
Baseline eGFR Accuracy Using RGB Features

Strip	Time	SVR	NNR	LR
all	2	43.6293	69.1120	57.9151
1	2	**44.0678**	69.4915	55.9322
2	2	37.2881	77.9661	67.7966
3	2	**44.0678**	76.2712	55.9322
4	2	**44.0678**	71.1864	69.4915
all	12	40.5405	81.4672	55.5985
1	12	30.5085	77.9661	64.4068
2	12	37.2881	77.9661	62.7119
3	12	42.3729	77.9661	54.2373
4	12	37.2881	69.4915	62.7119
all	22	41.6988	81.4672	60.2317
1	22	37.2881	67.7966	64.4068
2	22	37.2881	**88.1356**	**74.5763**
3	22	35.5932	77.9661	62.7119
4	22	27.1186	81.3559	67.7966
all	32	40.5405	74.1313	50.9653
1	32	38.9831	77.9661	62.7119
2	32	40.6780	77.9661	57.6271
3	32	42.3729	72.8814	52.5424
4	32	**44.0678**	67.7966	67.7966

Table 15
Baseline Regression RMSE Using RGB Features

Strip	Time	SVR	NNR	LR
all	2	12.0285	**6.9273**	**7.4511**
1	2	12.3981	9.2034	9.1631
2	2	12.3446	9.1470	8.5776
3	2	12.4007	8.4141	10.5765
4	2	12.3838	9.7093	8.4687
all	12	**12.0284**	7.9747	8.3858
1	12	12.3981	9.8526	9.9319
2	12	12.3446	8.7157	9.7178
3	12	12.4007	9.3011	10.0161
4	12	12.3837	10.8724	9.6014
all	22	**12.0284**	8.3854	8.2785
1	22	12.3981	10.5345	9.1838
2	22	12.3446	10.9211	9.6532
3	22	12.4007	8.5961	8.8606
4	22	12.3837	10.4509	10.4343
all	32	**12.0284**	7.9345	8.1697
1	32	12.3981	10.4188	9.3923
2	32	12.3446	8.6192	9.1033
3	32	12.4007	7.6539	8.9335
4	32	12.3837	10.4224	10.0262

Table 16
Hybrid eGFR Accuracy Using RGB Features

Strip	Time	SVC+ SVR	SVC+ NNR	SVC+ LR	NNC+ SVR	NNC+ NNR	NNC+ LR	LC+ SVR	LC+ NNC	LC+ LR
all	2	64.9606	71.2598	72.8346	63.3858	68.8976	71.2598	41.7323	54.3307	60.6299
1	2	54.2373	59.3220	54.2373	57.6271	64.4068	59.3220	44.0678	50.8475	49.1525
2	2	59.3220	71.1864	72.8814	62.7119	72.8814	74.5763	42.3729	54.2373	55.9322
3	2	57.6271	59.3220	66.1017	66.1017	64.4068	72.8814	**45.7627**	49.1525	57.6271
4	2	62.7119	71.1864	**81.3559**	59.3220	64.4068	76.2712	38.9831	57.6271	**67.7966**
all	12	63.7795	**79.9213**	79.1339	61.8110	77.5591	77.1654	38.1890	**61.0236**	65.3543
1	12	52.5424	66.1017	64.4068	52.5424	66.1017	64.4068	28.8136	49.1525	45.7627
2	12	40.6780	62.7119	59.3220	44.0678	64.4068	59.3220	27.1186	50.8475	49.1525
3	12	52.5424	61.0169	62.7119	64.4068	74.5763	76.2712	33.8983	47.4576	52.5424
4	12	57.6271	66.1017	76.2712	59.3220	67.7966	**77.9661**	32.2034	45.7627	59.3220
all	22	66.9291	74.8031	79.1339	64.1732	74.0157	77.5591	39.3701	53.5433	59.4488
1	22	59.3220	64.4068	67.7966	59.3220	59.3220	64.4068	33.8983	38.9831	44.0678
2	22	57.6271	79.6610	69.4915	55.9322	**77.9661**	67.7966	32.2034	55.9322	45.7627
3	22	61.0169	59.3220	67.7966	61.0169	59.3220	67.7966	32.2034	37.2881	47.4576
4	22	62.7119	71.1864	76.2712	57.6271	67.7966	72.8814	28.8136	40.6780	47.4576
all	32	68.5039	75.1969	77.9528	66.5354	73.2283	75.9843	39.7638	53.9370	54.3307
1	32	**72.8814**	76.2712	77.9661	**72.8814**	76.2712	**77.9661**	42.3729	52.5424	54.2373
2	32	67.7966	72.8814	69.4915	66.1017	71.1864	67.7966	40.6780	54.2373	45.7627
3	32	66.1017	71.1864	72.8814	64.4068	71.1864	71.1864	42.3729	61.0169	57.6271
4	32	64.4068	67.7966	71.1864	64.4068	66.1017	71.1864	38.9831	52.5424	50.8475

Table 17
Hybrid Regression RMSE Using RGB Features

Strip	**Time**	**SVC+ SVR**	**SVC+ NNR**	**SVC+ LR**	**NNC+ SVR**	**NNC+ NNR**	**NNC+ LR**	**LC+ SVR**	**LC+ NNC**	**LC+ LR**
all	**2**	**8.8023**	**7.9113**	**8.1289**	9.1360	8.5098	8.6116	**12.5706**	12.0951	11.9170
1	**2**	10.3158	9.8864	9.7288	9.7796	8.9527	9.0818	12.8504	12.4217	12.4263
2	**2**	10.3019	10.2982	9.9282	9.4088	9.3076	8.9808	12.8010	12.4302	12.1947
3	**2**	12.0001	11.6569	11.6341	10.1528	10.2187	9.8516	12.8669	12.5698	12.3082
4	**2**	10.4241	9.9690	9.9406	10.9322	10.4899	10.5929	12.8473	12.2972	12.2880
all	**12**	9.0176	8.9742	8.7090	10.1062	9.6544	9.6933	12.5708	12.0193	**11.9164**
1	**12**	11.5392	11.1527	11.2374	11.2441	10.8338	10.9561	12.8504	12.4007	12.3714
2	**12**	12.3132	11.9502	11.9622	12.4836	12.9847	12.2407	12.8010	12.3576	12.3181
3	**12**	11.7542	11.4160	11.5352	9.5995	8.8766	9.2398	12.8669	12.4652	12.4356
4	**12**	11.1924	11.0731	11.0252	11.7282	11.5290	11.4750	12.8474	12.4091	12.2209
all	**22**	10.1613	10.0777	9.9379	10.1608	10.0451	9.9866	12.5707	12.0400	11.9940
1	**22**	10.5015	10.3261	10.1904	11.1377	11.2270	11.0318	12.8504	12.3734	12.3895
2	**22**	9.7444	9.8826	9.5927	12.9284	13.7450	12.8274	12.8010	12.2615	12.2670
3	**22**	10.2179	10.1773	10.1205	10.7202	10.5019	10.5328	12.8669	12.3365	12.3547
4	**22**	10.2528	10.1161	10.0851	10.3638	10.1352	10.1761	12.8474	12.5448	12.3417
all	**32**	9.0295	8.8945	8.7610	**8.7895**	**8.3754**	**8.0130**	**12.5706**	**11.9546**	12.1545
1	**32**	10.8048	10.2737	10.4706	11.2570	11.3484	10.9768	12.8504	12.3422	12.4521
2	**32**	9.1569	8.7134	9.0959	12.4061	11.5872	12.1727	12.8010	12.1716	12.3538
3	**32**	10.3839	10.2387	10.3192	9.7898	9.1023	9.5343	12.8669	12.3053	12.4497
4	**32**	10.3531	10.1723	10.1171	11.5039	11.3401	11.3592	12.8473	12.1305	12.4120

A.3 HOG Results

Table 18
Baseline eGFR Accuracy Using HOG Features

Strip	**Time**	**SVR**	**NNR**	**LR**
all	**2**	47.1042	48.6486	37.0656
1	**2**	49.1525	44.0678	40.6780
2	**2**	37.2881	40.6780	32.2034
3	**2**	**50.8475**	**55.9322**	40.6780
4	**2**	42.3729	42.3729	38.9831
all	**12**	46.7181	46.7181	33.2046
1	**12**	30.5085	52.5424	50.8475
2	**12**	37.2881	38.9831	44.0678
3	**12**	44.0678	44.0678	45.7627
4	**12**	35.5932	40.6780	33.8983
all	**22**	44.0154	42.4710	35.9073
1	**22**	37.2881	45.7627	42.3729
2	**22**	33.8983	37.2881	45.7627
3	**22**	37.2881	49.1525	**59.3220**
4	**22**	30.5085	50.8475	38.9831
all	**32**	45.9459	41.6988	37.8378
1	**32**	37.2881	38.9831	50.8475
2	**32**	47.4576	44.0678	32.2034
3	**32**	45.7627	42.3729	32.2034
4	**32**	42.3729	42.3729	45.7627

Table 19
Baseline Regression RMSE Using HOG Features

Strip	**Time**	**SVR**	**NNR**	**LR**
all	**2**	11.9252	11.0860	19.2557
1	**2**	12.3897	10.8365	11.8741
2	**2**	12.3448	13.2733	13.7704
3	**2**	12.3441	13.3836	12.7369
4	**2**	12.3455	12.6822	14.1027
all	**12**	**11.8931**	12.4661	18.4790
1	**12**	12.3318	11.1266	13.0855
2	**12**	12.3246	13.7684	13.8858
3	**12**	12.3553	12.4726	12.8478
4	**12**	12.3351	13.1251	15.1282
all	**22**	11.9872	**10.6728**	16.6677
1	**22**	12.3821	11.9671	14.7951
2	**22**	12.3279	13.1383	14.1922
3	**22**	12.3399	11.0031	**8.5845**
4	**22**	12.3751	11.9600	13.2331
all	**32**	11.9871	12.8449	16.5790
1	**32**	12.3802	14.3174	12.2513
2	**32**	12.3209	12.7902	12.3170
3	**32**	12.3874	12.3387	14.5297
4	**32**	12.3741	12.3036	12.1588

Table 20
Hybrid eGFR Accuracy Using HOG Features

Strip	**Time**	**SVC+ SVR**	**SVC+ NNR**	**SVC+ LR**	**NNC+ SVR**	**NNC+ NNR**	**NNC+ LR**	**LC+ SVR**	**LC+ NNC**	**LC+ LR**
all	**2**	**48.4252**	**50.0000**	**50.0000**	47.6378	47.6378	48.4252	**46.8504**	47.2441	**47.2441**
1	**2**	44.0678	37.2881	44.0678	**57.6271**	**50.8475**	**55.9322**	45.7627	38.9831	45.7627
2	**2**	44.0678	33.8983	33.8983	45.7627	35.5932	37.2881	40.6780	30.5085	32.2034
3	**2**	44.0678	47.4576	44.0678	44.0678	49.1525	40.6780	44.0678	**47.4576**	44.0678
4	**2**	38.9831	44.0678	44.0678	32.2034	37.2881	38.9831	38.9831	44.0678	44.0678
all	**12**	38.5827	42.1260	33.8583	44.0945	46.8504	38.5827	43.3071	45.2756	37.0079
1	**12**	35.5932	38.9831	44.0678	33.8983	37.2881	40.6780	32.2034	35.5932	42.3729
2	**12**	28.8136	30.5085	25.4237	28.8136	32.2034	25.4237	30.5085	32.2034	27.1186
3	**12**	35.5932	38.9831	32.2034	35.5932	37.2881	33.8983	37.2881	40.6780	33.8983
4	**12**	33.8983	30.5085	28.8136	35.5932	33.8983	30.5085	35.5932	32.2034	30.5085
all	**22**	39.3701	39.3701	34.2520	40.1575	39.3701	35.4331	40.5512	40.1575	35.8268
1	**22**	33.8983	27.1186	40.6780	32.2034	28.8136	38.9831	33.8983	27.1186	40.6780
2	**22**	32.2034	25.4237	30.5085	30.5085	25.4237	32.2034	32.2034	25.4237	30.5085
3	**22**	30.5085	30.5085	23.7288	30.5085	30.5085	28.8136	30.5085	30.5085	23.7288
4	**22**	25.4237	28.8136	38.9831	25.4237	27.1186	35.5932	25.4237	28.8136	38.9831
all	**32**	39.3701	37.0079	36.6142	43.3071	41.3386	38.9764	39.7638	37.4016	36.2205
1	**32**	40.6780	40.6780	42.3729	42.3729	42.3729	42.3729	42.3729	42.3729	44.0678
2	**32**	40.6780	33.8983	33.8983	47.4576	42.3729	38.9831	40.6780	33.8983	33.8983
3	**32**	42.3729	42.3729	38.9831	45.7627	45.7627	44.0678	42.3729	42.3729	38.9831
4	**32**	37.2881	38.9831	42.3729	47.4576	47.4576	50.8475	38.9831	40.6780	42.3729

Table 21
Hybrid Regression RMSE Using HOG Features

Strip	**Time**	**SVC+ SVR**	**SVC+ NNR**	**SVC+ LR**	**NNC+ SVR**	**NNC+ NNR**	**NNC+ LR**	**LC+ SVR**	**LC+ NNC**	**LC+ LR**
all	**2**	**11.2917**	**11.1785**	**11.1895**	12.1786	11.9821	11.8843	**11.5424**	**11.4240**	**11.3829**
1	**2**	12.8481	12.7977	12.9626	**10.3855**	**10.0281**	**10.1788**	11.9631	11.6987	11.8974
2	**2**	12.8296	12.8325	12.8936	13.6958	13.1858	13.6855	12.9640	12.9214	12.9786
3	**2**	12.8640	12.4856	12.5645	12.8664	13.2698	12.8502	12.8640	12.4856	12.5645
4	**2**	12.8467	12.7271	12.6893	13.1791	13.2342	13.2744	12.8467	12.7271	12.6893
all	**12**	12.5097	12.3646	12.4136	12.9400	13.1032	12.9365	12.3277	12.3344	12.2385
1	**12**	13.5463	14.1514	13.6531	12.2196	12.0553	12.1247	12.5066	12.2924	12.3481
2	**12**	12.7947	12.6105	12.4919	14.7059	14.7361	14.5189	12.8230	12.6447	12.5347
3	**12**	12.8597	12.6407	12.5653	14.3422	13.9776	13.8143	12.8720	12.6335	12.5662
4	**12**	12.8514	12.7380	12.8605	13.0550	12.9666	13.0584	12.6963	12.3756	12.6556
all	**22**	12.5426	12.4571	12.5343	12.6650	12.6197	12.6185	12.4347	12.3525	12.4461
1	**22**	12.8445	12.7915	13.0277	12.2124	12.2386	12.3188	12.8445	12.7915	13.0277
2	**22**	12.7895	12.7197	12.6290	12.8791	12.8562	12.7575	12.7895	12.7197	12.6290
3	**22**	12.8614	12.7187	12.7783	13.3926	13.4966	13.2333	12.8614	12.7187	12.7783
4	**22**	12.8521	12.8389	12.9002	12.1879	12.1216	12.0527	12.8521	12.8389	12.9002
all	**32**	12.5771	12.4302	12.5087	12.9516	12.8909	12.9503	12.5238	12.3754	12.4631
1	**32**	12.9198	12.7827	12.8047	13.1691	13.0033	12.9842	12.8505	12.6969	12.6369
2	**32**	12.7947	12.7813	12.9575	11.7611	12.0477	11.9780	12.7947	12.7813	12.9575
3	**32**	12.8541	12.7771	12.8261	12.2356	12.1505	12.1831	12.8541	12.7771	12.8261
4	**32**	12.8470	12.7425	12.7084	12.8632	12.9512	13.3858	12.5774	12.5480	12.5845