# The AutoScope: An Automated Point-of-Care Urinalysis System

by

Sidney R. Primas

B.S.E. in Electrical/Computer Engineering and Biomedical Engineering
Duke University (2012)

Submitted to the Department of Electrical Engineering and Computer Science in partial fulfillment of the requirements for the degree of

Masters of Engineering in Electrical Engineering and Computer Science
at the
MASSACHUSETTS INSTITUTE OF TECHNOLOGY
June 2018



Signature redacted

Author _____

Sidney R. Primas
Department of Electrical Engineering and Computer Science
May 23, 2018

Signature redacted

Certified by _____

Charles G. Sodini
LeBel Professor of Electrical Engineering and Computer Science
Thesis Supervisor

Signature redacted

Accepted by _____

Leslie A. Kolodziejski
Professor of Electrical Engineering and Computer Science
Chair, Department Committee on Graduate Students

# The AutoScope: An Automated Point-of-Care Urinalysis System

By: Sidney R. Primas

Submitted to the Department of Electrical Engineering and Computer Science
On May 23, 2018, in partial fulfillment of the
requirements for the degree of
Masters of Engineering in Electrical Engineering and Computer Science

## Abstract

The goal of this project was to develop an automated, low-cost microscopic urinalysis system that could accurately detect red blood cells (RBCs), white blood cells (WBCs), and other particles in urine. The Autoscope is a proof-of-concept for this end-to-end automated microscopic urinalysis system. A similar system can eventually be applied to microscopic analysis of blood.

Urinalysis is one of the most common diagnostic techniques in medicine. Over 200 million urine tests are ordered each year in the US, costing between $800 to $2,000 million in direct costs. 46% of all urinalysis tests include microscopic analysis, which involves identifying and counting each particle found in the urine. Microscopic urinalysis is a costly and complex process done in medical laboratories. An inexpensive and automated cell-counting system would (1) increase access to microscopic urinalysis and (2) shorten the turn-around time for physicians to make diagnostic decisions by permitting the test to be done at the point-of-care.

The system we built – the AutoScope - has three parts: the image acquisition system, the segmentation system and the classification system. The image acquisition system used a reversed lens approach to achieve an end-to-end resolution of 5.86-8.29um for a total bill of materials cost of $57-$92. The automated particle segmentation and particle classification were each performed with different neural networks.

We calculated the accuracy, sensitivity, and specificity of the Autoscope system with respect to urine solutions composed of RBCs, WBCs, and microbeads. The specificity and sensitivity was determined by generating 209 digital urine specimens modeled after urine received in medical labs. The Autoscope had a sensitivity of 88% and 91% and a specificity of 89% and 97% for RBCs and WBCs, respectively. Next, we determined the Autoscope's accuracy by fabricating 8 synthetic urine samples with RBCs, WBCs and microbeads. The reference results were confirmed through a medical laboratory. The AutoScope's counts and the reference counts were linearly correlated to each other ($r^2$= 0.980) across all particles. The sensitivity, specificity, and R-squared values for the AutoScope are comparable (and mostly better) than the same metrics for the iQ-200, a $100,000-$150,000 state-of-the-art semi-automated urinalysis system.

Thesis Supervisor: Dr. Charles G. Sodini
Title: Clarence J. LeBel Professor

# Acknowledgements

I would first like to thank Charlie Sodini, my thesis advisor and mentor through out my time at MIT. He has not only guided me through my academic journey, but has also helped me think through numerous personal decisions. It has been a pleasure and honor working with him. Next, I want to thank Tom O'Dwyer, who provided weekly technical guidance through out this research project. Tom brought an industry perspective to this research, which included helping us focus on practical technical solutions. Thank you Joel Voldman and Dan Wu, who have helped me navigate the wet lab and biological components of my research with their deep intuition and experience. Also, thank you to Ryan O'Shea, who worked with me at the early stages of this project. And, I would like to give a huge thank you to Analog Devices Inc., who funded me and my project through out my time at MIT.

Thank you to my lab mates, Maggie Delano, Joohyun Seo, Grant Anderson and Sabino Pietrangelo. Thank you to all my friends and family for their support over the past years. I could not have done this without them, and I deeply appreciate their support. My parents (Karina and Robert Primas), my brother (Joshua Primas) and Sid Kandan have especially supported me at every step in my life journey. Lastly and most importantly, I would like to thank Lina A. Colucci for her advice and support. Thank you!

# Table of Contents

# 1 Introduction

Microscopic analysis of biological matter, especially blood and urine, remains one of the most in-demand diagnostic techniques today. However, access to microscopic analysis is hindered by 1) manual sample analysis and 2) the high-cost of the test equipment[1]. Largely due to these factors, microscopic analysis is almost exclusively performed in major hospitals and offsite clinical laboratories. A low-cost and automated microscopic system would allow microscopic tests to be performed at the point-of-care and at-home settings. Such a device would lead to multiple advantages, including allowing patients to monitor health conditions at home (e.g. for chronic conditions), reducing the frequency of physician visits, enabling physicians to make more rapid and accurate decisions in clinics, and increasing the likelihood of diagnosing a disease early.

The goal of this project is to develop a low-cost, portable microscope with sufficient image acquisition properties to perform automated cell detection and classification.
This microscopic system can eventually be applied to the cell analysis of both urine and blood. For practical reasons, we will focus on microscopic analysis of urine in this thesis.

## 1.1 Microscopic Urinalysis

Urinalysis is a fast, simple and inexpensive tool for evaluating a patient's health, including detecting diabetes, pregnancy-related disorders, liver disease, kidney stones, etc. Urinalysis consists of a visual, chemical and microscopic examination of the urine. Microscopic analysis of urine is especially useful for diagnosing urinary tract infections (UTIs) and kidney disease. In UTIs, microscopic analysis is used to conclusively find bacteria in urine. UTIs lead to 6.7 million physician office visits and 2.6 million ER visits in the United States, with a 30% chance that the infection will recur in 3 months[2]. A home-based test could reduce clinic visits, especially after initial diagnosis. For kidney disease, casts and blood cells in the urine are indicators of the type and stage of kidney disease. With 1 in 9 adults having chronic kidney disease, and over 20 million adults being at increased risk, this device could routinely track the progression and flares of a patient's chronic kidney disease[2].

When urinalysis is used for health screening, the European Urinalysis Guidelines suggest two steps: 1) visual inspection of urine and a chemical dipstick test, and 2) microscopic analysis

of the urine[3]. If the dipstick test indicates the presence of hemoglobin (indicating red blood cells), leukocyte-esterase-activity/nitrite (indicating presence of bacteria and white blood cells) or protein, a follow-up microscopic test is performed. Dipsticks are designed to reduce false-negatives with the trade-off of having significant false-positives. Thus, microscopy is needed to confirm and then quantify the presence of erythrocytes (red blood cells), leukocytes (white blood cells), and bacteria. In fact, before a physician makes clinical decisions based on the dipstick analysis, they first perform a microscopic analysis, which requires sending a urine sample to a laboratory. A point-of-care microscopic analysis tool will enable physicians to make these clinical decisions for the patient more immediately and decrease the turn-around time for making diagnostic decisions.

At a laboratory, trained technicians perform the microscopic analysis. First, 10-15mL of urine is centrifuged at 1500-3000 rpm for five minutes. The supernatant is then removed, the sediment re-suspended, and a drop of liquid transferred to a slide for manual examination[4]. Recently, laboratories have been using partially automated equipment (like the Iris iQ-200 and the Dirui FUS-200) to perform the microscopic analysis with either flow cytometry or image-based analysis[1,5]. These systems are expensive (the iQ-200 costs between $100,000 to $150,000), and still require trained staff to verify certain results.

## 1.2 Previous Work

### 1.2.1 Image Acquisition

The ubiquity of digital cameras (including mobile phone cameras), and the resulting decrease of hardware cost has recently led to multiple research efforts to develop a low-cost microscope device. The core trade-off in these efforts is between achieving the appropriate resolution for analysis while maintaining both a low cost and a large field of view. Digital microscopes are used to acquire these images, often combining some type of analog magnification (a simple lens, an objective, a ball lens, etc.) with an image sensor that digitizes the photons.

A digital microscope's resolution is defined by its optical resolution (the shortest distance between two entities that can be distinguished as separate) and digital resolution (the image sensor's spatial resolution). To increase optical resolution, spherical ball lenses (as sold by

Edmund Optics among other vendors) have offered a low-cost approach to achieve significant magnification. With a sufficiently small sphere diameter, over 350x magnification can be achieved at costs between $10-$26 (at low volume). The drawback of the ball lens approach is distortion at the edges of the field of view, further reducing the overall field of view[6]. In another approach, the Fletcher Group used a reversed mobile phone camera lens (with a cost as low as $6 per lens) to provide a resolution <5um[7,8]. In comparison to the optical resolution that depends on the properties of the lenses, the digital resolution depends on the pixel density of the CCD/CMOS sensor. Image-processing techniques can be used to increase digital resolution. Pixel-shifting is a technique where the image sensor is shifted by sub-pixel distances, leading to an oversampling of the spatial distribution. The oversampling is then used to obtain sub-pixel resolution. These techniques in combinations with high-quality image sensors are used to achieve sufficient spatial resolution for microscopy.

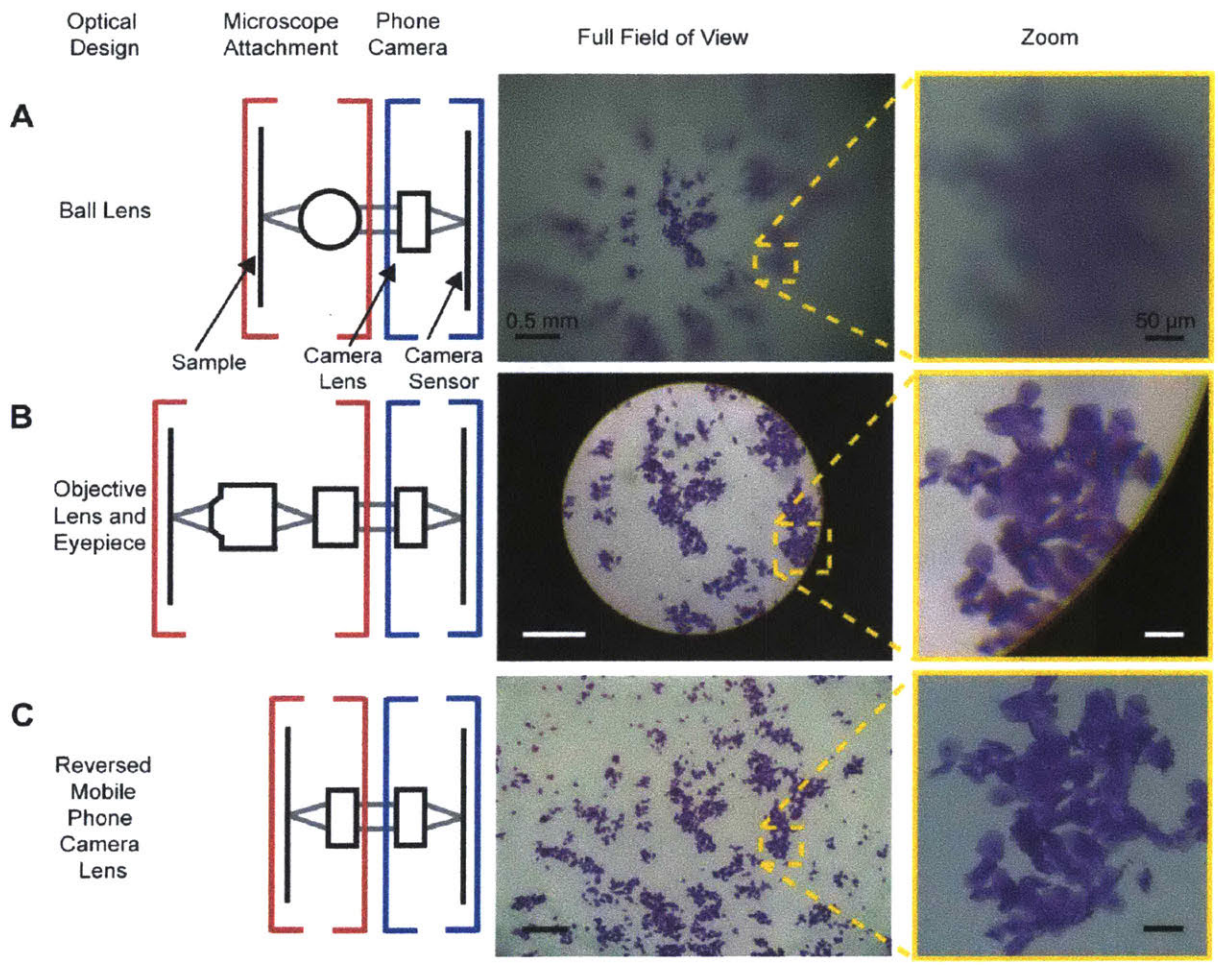



*Figure 1-1. Three techniques of microscopic magnification are demonstrated on epithelial cells, including (a) a ball lens, (b) an objective lens and (c) a reverse camera lens. The magnification and field of view is shown to the right. Note the blurring of the field of view when using the ball lens. Figure from Switz et al* [7].

### 1.2.2 Image Analysis

Today, significant portions of microscopic analysis are still performed by trained technicians manually manipulating samples, and counting biological components of interest[5]. Recently, hospitals/laboratories are transitioning to automated microscopic processing, but often still use technicians to complement and confirm their automated analysis. The core obstacles with segmentation and classification are poor particle resolution, poor particle contrast, the variability of particle shape and size, and background noise related to other particles[9]. The

detection and classification of particles usually includes a three-step process: 1) segmentation, or identifying the location of the particle in an image, 2) feature extraction, including any preprocessing of the images, and 3) a probabilistic determination of the type of particle based on the features[10]. Techniques recently used to probabilistically classify urine particles include: support vector machines[11], and fuzzy neural networks[12]. In 2010, a paper used a compilation of images across multiple illumination wavelengths to perform classification on urine sediment, including classification with Principle Component Analysis and Support Vector Regression. They achieved a classification accuracy of 92.6% across 6 classes of urinary particles[9].

## 1.3 Thesis Objectives

The overall goal of this project is to build a portable, low-cost microscopic device that can acquire sufficiently high-resolution images to count particles in urine. The cell counting and classification will be fully automated using image processing and machine learning techniques. Eventually, the device can be used for applications with both blood and urine. As a starting point, the device will focus on urine since urine is 1) easier to handle in a research setting, 2) easier to acquire and handle by patients, and 3) can be processed at larger sample volumes. Within urine, we plan to count and classify red blood cells and white blood cells, which are the most clinically relevant and widely used markers extracted from urinalysis. Microbeads with a similar diameter as white blood cells will also be included to make the classification problem more difficult.

Specifically, the project goals are to:

1. **Image Acquisition**: capture digital images of urine with sufficiently high magnification and resolution to allow for segmentation and classification.
2. **Cell Counting and Classification**: count and classify the cells in the resulting images using image processing and machine learning techniques. This will be achieved in a three-step process: image segmentation (where cells or clumps of cells are isolated), feature extraction and classification.

## 1.4 Thesis Organization

In the next several chapters, we will present the AutoScope, an end-to-end automated, portable microscopic urinalysis system. In Chapter 2, we will introduce the theory underpinning neural networks. Then, Chapters 3, 4 and 5 we will discuss the functional building blocks of the AutoScope. Specifically, in Chapter 3, we characterize the AutoScope's image acquisition sub-system, including the mechanical housing and the the AutoScop's unique optics. In Chapter 4, we demonstrate the accurate classification of low-resolution urine particles using a convolutional neural network. And, in Chapter 5, we present a segmentation algorithm that accurately identifies the location of urine particles. In Chapter 6, we combine these subsystems to perform an end-to-end validation of the AutoScope. We validate the AutoScope system by comparing its particle counts from synthetic urine to known reference counts. Finally, in Chapter 7, we will summarize our work and discuss future work.

# 2 Background for Neural Networks

In this chapter, we will introduce the foundational theory behind convolutional neural networks. First, we will describe the building blocks of neural networks, including neurons, types of layers and a block of connected layers. Second, we will introduce a specific type of neural network, the convolutional neural network (CNN). Third, we will review the infrastructure for training neural networks, including the loss function and stochastic gradient descent (SGD). The background on neural networks presented in this chapter is inspired by material from 'Convolutional Neural Networks for Visual Recognition' at Stanfrod (Li and Karpathy) and from 'Advances in Computer Vision' at MIT (Freeman and Torralba)[13,14].

## 2.1 Building Blocks of Neural Networks

### 2.1.1 The Neuron

The neuron is the basic functional building block of a neural network. Thus, to understand a neural network, we must first understand the architecture of a neuron. The computations that a single neuron performs can be mathematically represented as:

$$N_f(x_i) = act\left(\sum_i w_i x_i + b\right)$$

$x_i$ represents an input value, that can stem from other neurons or the data input. $w_i$ represents a learned weight applied to the neurons input. $b$ is the learned bias, providing an offset to the neuron that's independent of the neuron's input. And, $act()$ is the activation function, usually providing a non-linearity to each neuron.

### 2.1.2 The Non-Linear Activation Function

The non-linearity is a key feature of the neuron, enabling a neuronal network to significantly increase its representational power. Without a non-linear $f$, the neuronal network can be collapsed into a simple linear regression, only representing a multivariate linear function. With the a non-linear $f$, the neural network becomes a universal function approximator, having the representational power to approximate an arbitrarily complex function. Architectures use a

wide variety of non-linear functions, but the most popular is the Rectified Linear Unit (ReLU). The ReLU function is defined as:

$$act(a) = \max(0, a)$$

The ReLU non-linearity is simple to evaluate, and boasts significant improvements to a networks learning speed.

### 2.1.3 Fully Connected Layers

A neural network is composed of multiple layers and each layer is composed of multiple neurons. The simplest and most common layer is the fully-connected layer. A fully connected layer is represented by a set of neurons where each neuron is fully connected to the previous layer. Figure 2-1 has two fully-connected layers: the hidden layer that's fully connected to the input layer, and the output layer that's fully connected to the hidden layer. Each neuron performs the $N_f$ function on it's input data (with the hidden layer using a nonlinear activation and the output layer using using a linear activation). Design choices can be made about the number of fully connected layers as well as the number of layers in each neuron.

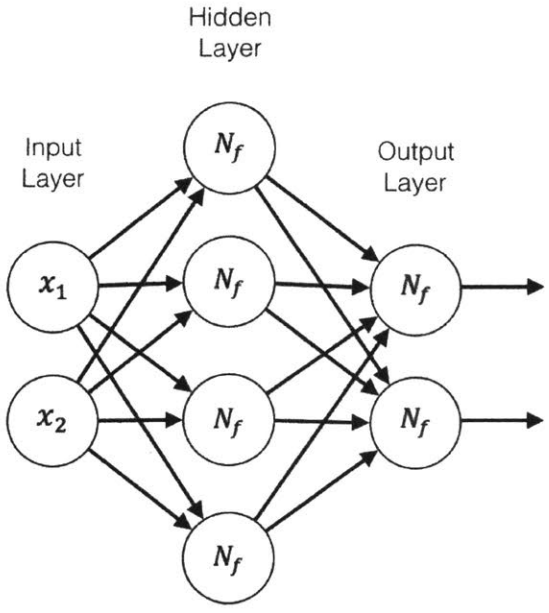



*Figure 2-1. A small neural network that takes two input values ($x_1$ and $x_2$) and passes these values through two fully connected layers.*

Since the dot product of matrices are highly computationally efficient, networks are setup to leverage this operation to increase performance. Specifically, instead of applying the $N_f$ operation of each neuron separately, the operation of the fully connected layer can be represented as a single set of matrix operations:

$$f(Wx + b)$$

where $W$ is the the weight matrix (with each row representing the weights of a single neuron), $x$ is the input vector and $f$ is the activation function applied to each element. In the fully-connected output layer, the weight matrix $W$ has a size of (number of output classes X number of neurons in the previous hidden layer). This produces an output vector with a numerical score for each output class, with the highest score mapping to the predicted class.

## 2.2 Convolutional Neural Networks

Convolutional neural networks (CNNs) are a subclass of deep neural networks that originated from architectures designed to process images. From an architecture perspective, CNNs can be split into two separate steps: the convolutional layers and the fully-connected layers. The convolutional layers use learned filters to extract features from the input images. The fully-connected layers collapses these extracted features into a per class score through a neural network of learned weights.

### 2.2.1 Convolutional Layers

Convolutional layers extract features by convolving learned filters across datasets to produce feature maps. The goal of the convolutional layers is to extract the most important features from an image, which can then be used by the fully-connected layer to classify the image. In the past, these features were extracted through manually designed filters based on a deeper understanding of the data. Within CNNs, the features are extracted through learned filters that converge on extracting the most important features through training. An important benefit of the convolutional layers is the dimensionality reduction of both the weights and input data. The dimensionality of the data is reduced through the pooling layer. And, the dimensionality of the weights is reduced by sharing weights across neurons (represented through learned filters per image instead of per pixel).

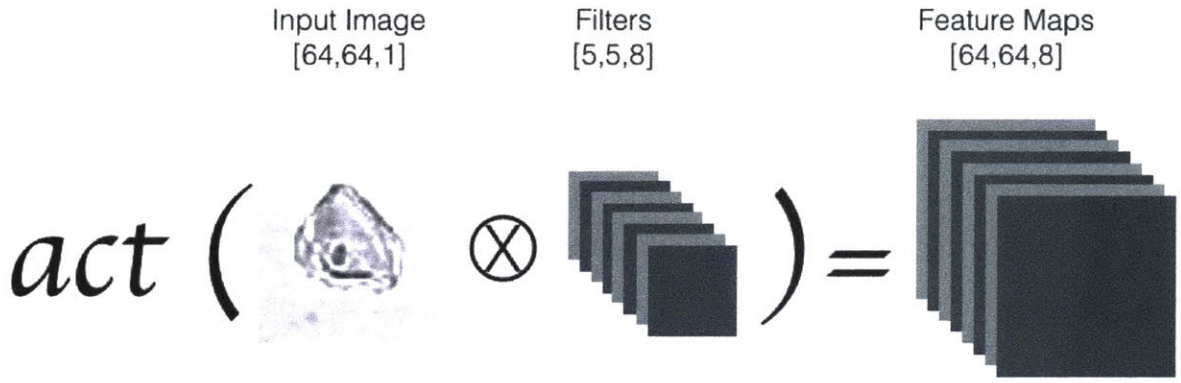



*Figure 2-2. A visual representation of a convolutional layer that generates 8 feature maps by convolving a single input image with 8 filters followed by a pixel-wise activation function.*

Convolutional layers are usually grouped into a functional block, with each block containing three types of functions: filtering (mathematically implemented as a cross-correlation), activation and max pooling. The simplest block can be represented by a single function, given by:

$$Output_{feature\ maps} = \{pool(act(w_1 * x\_image)), \ldots, pool(act(w_n * x\_image))\}$$

Each convolutional block has a set of filters, represented by a 3-dimensional matrix called $W_{filters} = \{w_1, \ldots, w_n\}$. In Figure 2-2, the 3D filter matrix $W_{filters}$ can be interpreted as having 8 filters of size 5 weights by 5 weights, where each of these weights is learned during training. The input image ($x\_image$) is convolved with each of these filters, producing a set of filtered outputs. The activation function, the ReLU function in this work, is then applied to each pixel producing a feature map that has the same width/height dimensions as the input image (as seen in Figure 2-2).

The final step in the convolutional block is the pooling layer, which is performs a non-linear downsampling of the feature maps. Pooling can be represented by a $(m * m)$ window that we slide across the feature map with a specific step size. The window takes the $(m * m)$ inputs and outputs a single value, thus reducing the dimensionality of the dataset. The most common window function is termed max pooling, which only passes forward the the maximum input value. In this work, the max pooling is applied across a 2x2 area with a stride of 2. Thus, for each pooling layer, we reduce the number of features by 75%. The pooling layer incentivizes the network the learn coarser features at the higher convolutional blocks. For example, the early

feature maps might represent edge detection feature while the later blocks might represent object features, like an entire face.

The convolutional layers of a network often contain multiple convolutional blocks (usually 2-5 blocks), allowing for coarser feature extraction and higher representational power of the network. The architecture of each block might vary as well. For instance, instead of using a single filter layer in each block, best practice suggests to use two back-to-back filtering layers (with 3x3 filters) followed by an activation and max pool layer [VGG reference]. The back-to-back 3x3 filters operate to increase the filtering layer's receptive field from 3x3 to 5x5 input pixels.

## 2.3 Training Infrastructure for Neural Networks

The CNN can be seen as a scoring function, translating the raw image pixels of an image into a set of class scores at the output. CNNs are highly complex scoring functions with millions of trainable parameters. However, they still can be simply expressed as:

$$f(x\_image_i, W) = s_i$$

where $x\_image_i$ is a vectorized input image, W is the set of trainable weights of the network, and $s_i$ is the set of class scores. The subscript $i$ represents the values for a specific image. A loss function is used to quantify the quality of these scores. In supervised learning, the loss function compares the scores to the ground truth label for each image ($y_i$,), and returns a loss ($L_i$). During training, the goal is to adjust the weights, W, of the network in order to minimize $L_i$, the output of the loss function across our entire training set. This is accomplished by iteratively updating the weights based on the gradient of the loss function.

### 2.3.1 Loss Function

For supervised CNNs, a loss function quantifies the quality of the network by comparing the network's scores to the ground truth labels. A variety of loss functions are commonly used within CNNs, including support vector machine (SVM) loss and least square errors (L2 loss). In this work, we obtain the overall loss ($L_i$) by combining the cross entropy data loss ($L_{entropy}$) with a regularization loss ($L_{regularization}$).

$$L_i = L_{entropy} + L_{regularization}$$

#### *2.3.1.1 Cross-Entropy Loss*

The cross-entropy loss function is defined as:

$$L_{entropy} = -\sum_{j} p_j \log(q_j)$$

where p is the ground truth probability distribution, q is the estimated probability distribution, and j is the index of each class. This loss formulation quantifies the difference between distribution p and q, and is derived from comparing encoding schemes within information theory. Since both p and q are probability distributions, the values $p_j$ and $q_j$ need to be between zero and one, and need to sum up to 1. $p$ is simply a vector of 0s with a 1 at the position of the ground truth label. For $q$, the softmax function transforms the raw class scores produced by the CNN into a probability distribution of predicted classes. The softmax function is defined as:

$$softmax(s)_j = \frac{\exp(s_j)}{\sum_k \exp(s_k)}$$

where s is the raw class score vector and j is the index of each class. The softmax function uses exponents in order to augment differences in the class scores and appropriately account for negative scores. By substituting in $p$ and $q$ in the cross-entropy loss function, we obtain a simplified cross-entropy loss:

$$L_{entropy} = -\log\left(\frac{\exp(s_{y_i})}{\sum_k \exp(s_k)}\right)$$

where $s_{y_i}$ represents the predicted raw score of the ground truth class. Another common interpretation of the of this final form of the cross-entropy loss function is as the negative log likelihood, since we are taking the negative log of the likelihood of the ground truth class (where the likelihood is given by the softmax of the raw scores of the correct class as indicated by the label).

#### *2.3.1.2 Regularization Loss*

The cross-entropy data loss provides a metric to evaluate the fit of the model's weights based on training data. The goal is to minimize the loss. However, since the training data is only a subset of the total data, the minimization of the cross-entropy loss function does not provide a unique solution (it's an underdetermined problem). For the weights to better converge to a single solution during training, we add regularization loss to the loss function. The regularization loss is defined as:

$$L_{regularization} = \lambda \sum_{m} \sum_{n} W_{m,n}^{2}$$

where W is a matrix of all the CNNs weights and $\lambda$ is a weighting hyperparameter. Specifically, regularization loss sums the squares of the weights in the CNN, essentially incentivizing smaller weights that minimize the cross-entropy loss over the larger weights that might also minimize the cross-entropy loss. By incentivizing smaller weights during training, the model is encouraged to take into account more of the evidence (through smaller, more diffuse weights) instead of relying on a few select pieces of evidence (through larger, more concentrated weights). A general benefit of regularization is making the model more generalizable to data outside of the training set, instead of overfitting the model to the data within the training set.

### 2.3.2 Network Optimization

The loss function provides a methodology to evaluate the quality of a CNN. During training, the goal is to select weights that minimize this loss function across the image dataset. This loss function is a highly-dimensional space (often with millions of dimensions), with each trainable weight in the CNN network representing a unique dimension. Thus, finding the the optimal set of weights to minimize the loss function is highly complex, interdependent problem.

The strategy to select a set of weights is to iteratively update each weight in the network through a technique called Stochastic Gradient Descent. Broadly, SGD uses gradients of the loss function with respect to each trainable weight to iteratively update each weight in the network. Specifically, each parameter is updated using the following equation:

$$W_{j\,(new)} = W_j - \gamma * (\nabla_{W_j} L_i)$$

where $\gamma$ is the SGD learning rate hyperparameter and $\nabla_{W_j} L_i$ is the gradient of the loss function with respect to the parameter to be updated, $W_j$. Intuitively, each weight is updated in the direction that further minimizes the loss function, with a magnitude defined by the step size and the magnitude of the 'slope' of the loss function at it's current weight.

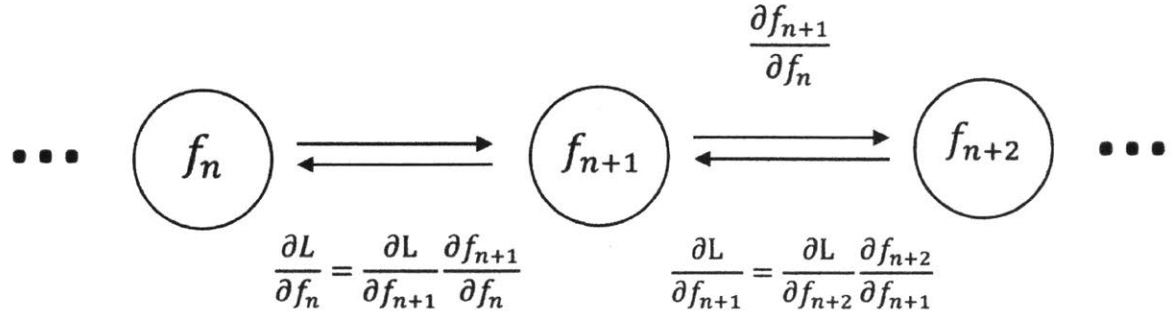



*Figure 2-3. A demonstration of how CNNs recursively calculate the gradient with respect to the sub-functions in the CNN.*

The key to implementing the SGD algorithm for training is efficiently calculating the $\nabla_{W_j} L_i$. This is achieved through a technique called backpropagation, which computes the analytical gradients of complex functions through the recursive application of the chain rule[15]. Figure 2-3 is a visual representation of how CNNs recursively calculate the gradient with respect to each function. CNNs can be represented as many separate functions ($f_n$, $f_{n+1}$, $f_{n+2}$) chained together into a complex network of functions. During forward propagation, the algorithm iteratively evaluates the partial derivative of each function with respect to it's input ($\frac{\partial f_{n+1}}{\partial f_n}$). Then, during backpropagation, the gradients of the loss function with respect each intermediate function is calculated ($\frac{\partial L}{\partial f_{n+1}}$). This process propagated backwards recursively to calculate the gradient of of the loss function to each local function, including the gradients of all the input weights. And thus, the gradients of all the input weights are efficiently calculated through a single forward pass and a single backwards pass of information through the network. By extending the scheme from Figure 2-3, arbitrarily complex networks can be designed, as long as each sub-function within the network is differentiable (and a few other restrictions).

# 3 Image Acquisition System

In this chapter, we will characterize the image acquisition system of the AutoScope. In order to accurately classify urine particles, the system needs to capture images with sufficiently high resolution. In order to make this a feasible point-of-care urinalysis system, the system also needs to be low-cost and portable. The trade-off between cost/portability and image resolution is the core tension in designing the AutoScope, and will drive most of the major design decisions. In this chapter, we will first review the mechanical system that enables image acquisition. Then, we will take a deep dive into the lens design, which is the most important component of the image acquisition system.

## 3.1 Mechanical System

The mechanical system includes three broad functional parts: the focusing system, the imaging system and the lighting system. These three functional components are housed in an inexpensive case made from 3D printed components and screws. The 3D printed components were designed in SolidWorks, and printed with a Form 2 3D printer. Screws were used both for structural integrity (as 3D material can warp) and to keep costs low (as 3D material is expensive). The AutoScope system measures 8.3x6.0x8.8cm (length x width x height).

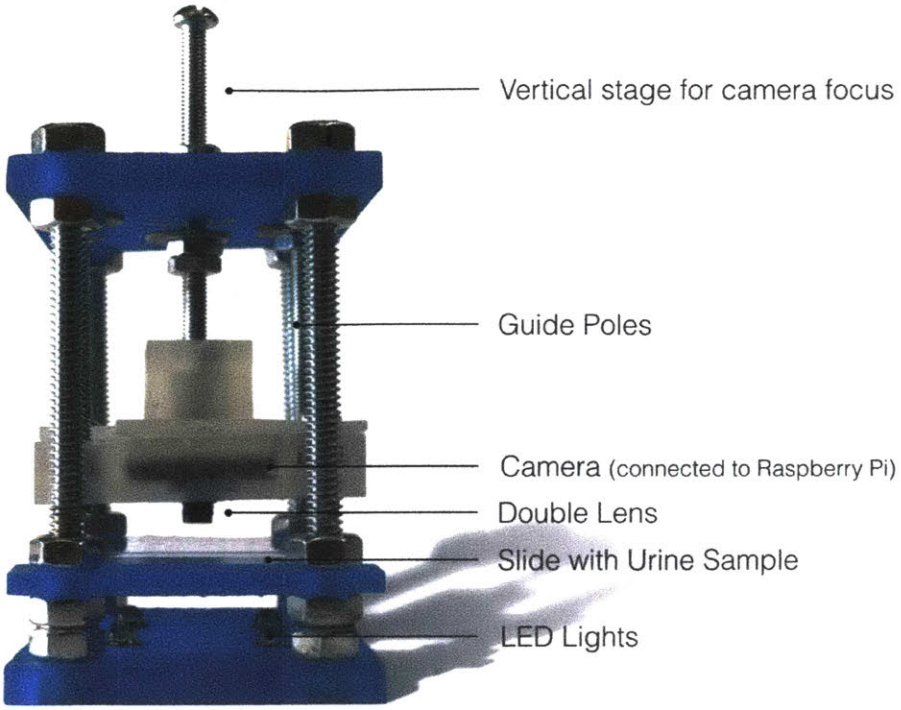



*Figure 3-1. A side-view of the AutoScope system.*

### 3.1.1 Focusing Subsystem

The focusing system includes a vertical stage that allows for fine z-height adjustment of the camera. To ensure that the camera does not rotate or tilt during z-height adjustments, we included two guide poles that go through the camera case. Theses guide poles are fixed to the top plane of the AutoScope, and thus stabilize the camera during z-height adjustment. The vertical stage has a range of ~1.5cm, allowing for a variety of object sizes to be interrogated.

### 3.1.2 Imaging Subsystem

The imaging system is composed of a Raspberry Pi (RPi), an image sensor and a reversed lens. The RPi controls the image sensor, sending commands to capture images and then locally stores these images. Depending on the application, the RPi is used to either perform image analysis to count the urine particles locally or to transfer the image to an Amazon Web Server (AWS) for analysis[16]. We used a Raspberry Pi 3 Model B, which costs $35 for a single unit. We use the 8 megapixel IMX219 CMOS imaging sensor from Sony[17]. The imaging sensor is mounted on a Raspberry Pi Camera Module V2, which includes a lens. The full RPi Camera Module V2 with image sensor, lens, PCB and cable costs between $7-$20 depending on if we

build the systems directly vs. go through the Raspberry Pi Foundation. The lensing system includes two compound lenses combined into a reversed lens configuration. In the next section, we will take a deep dive into the lensing system.

*Table 1. Summary of the material costs for the AutoScope system.*

| Component | Cost Estimates |
|---|---|
| Raspberry Pi 3 Model B | $35 |
| RPi Camera Module V2 | $7-$20 |
| Additional Compound Lens | $2-$6 |
| Structural Material | $5-$15 |
| High Power LEDs | $3-$4 |
| Power System | $5-$12 |
| Total | $57-$92 |

### 3.1.3 Lighting Subsystem

The final functional system is the lighting system. We designed the lighting system to approximate darkfield illumination. Darkfield illumination is a technique that allows for increased particle contrast, and is especially useful for unstained and transparent specimens (like human cells)[18,19]. Microscopes usually use brightfield illumination, where contrast is caused by the attenuation of light as it passes through a sample. This leads to an attenuated luminance for the particle as compared to the background. In darkfield illumination, the particle has a higher luminance than the background. Darkfield illumination works by having light rays at an angle sufficiently oblique to the field of view so that no direct light rays enter the objective lens. Thus, when there is no object in the field of view, the field of view (or background) is dark. Once a specimen is introduced into the field of view, the specimen reflects, refracts and diffracts light. The specimen essentially deflects light directly into the objective lens, causing particles to be bright (high luminance) and the background to be dark (low luminance). When the particle diffracts light, the zeroth order lobe still does not enter the objective lens. However, some of the side lobes created during diffraction will now enter the objective lens, causing an increase in the luminance at the particle location.

Darkfield illumination provides significant improvements in terms of particle contrast compared to brightfield illumination and oblique illumination. Furthermore, our implementation of darkfield illumination reduces the cost and complexity of the lighting system as compared to phase-contrast microscopy.

The AutoScope uses 4 high-powered LEDs (specifically, neural white light Luxeon K2 Lumiled LEDs)[20]. These LEDs are embedded symmetrically into the bottom plane of the AutoScope at angles that enable darkfield illumination. Due to the low-complexity of our lighting setup, we have other objects than the specimen also deflecting light into the objective lens. These objects include the structural screws of the system, the structural 3D printed cases, the LED cases/lenses and even the slide. Better controlling for these interfering objects would improve contrast, and possibly the AutoScope performance.

## 3.2 The Reversed Lens

The goal is to develop an image acquisition system that captures diagnostic-quality images at an affordable price. The lens design for an optical microscope is probably the most important design parameter since it drives both the resolution and the price of the system.

In the past decade, the economies-of-scale in the consumer electronics world have led to high-quality lenses at affordable prices. Specifically, the competitive market place for mobile phones has led to higher-quality, more compact, and lower cost lenses and image sensors. The result is a highly-complex compound lens designed to be optimally coupled to a specific image sensor has a cost below $6 per lens and size of <10mm in length. Due to the compound lens design, these lenses boast high numerical apertures (often between 0.17-0.23) with built-in design adjustments for chromatic and spherical aberrations[7]. All told, an incredible amount of optimization has gone into these lenses, and we propose to leverage these optimizations for our imaging system. Specifically, we use a mass-produced compound lens coupled with the Sony IMX219 image sensor as the foundation of our optical imaging system

The lens and image sensor function as the foundation of the optical system. However, these consumer-oriented lensing systems are designed to focus on objects that are meters away while a microscope needs to focus on objects that are millimeters away. To enable microscope applications, we utilize a secondary lens element that refocuses light rays from an infinity focus to a millimeter focus. Even though there are other techniques to alter the focal point of a lens

(like changing the distance from the lens to the image sensor), this secondary lens also functions to improve the overall properties of the optical system. Past research has demonstrated multiple options for this secondary lens element, including an objective lens coupled with an eyepiece, a ball lens and a reversed lens approach. The objective lens coupled with an eyepiece is the standard approach for optical microscopes, but leads to bulky, more expensive systems (an inexpensive objective lens costs between \$60-\$200)[21]. The ball lens approach uses a spherical ball of glass as a lens, and can achieve magnifications of over 1000x at very low costs[22,23]. However, the ball lens has a very small usable field-of-view due to high field curvatures and delivers highly-variable results depending on any impurities of the ball lens.

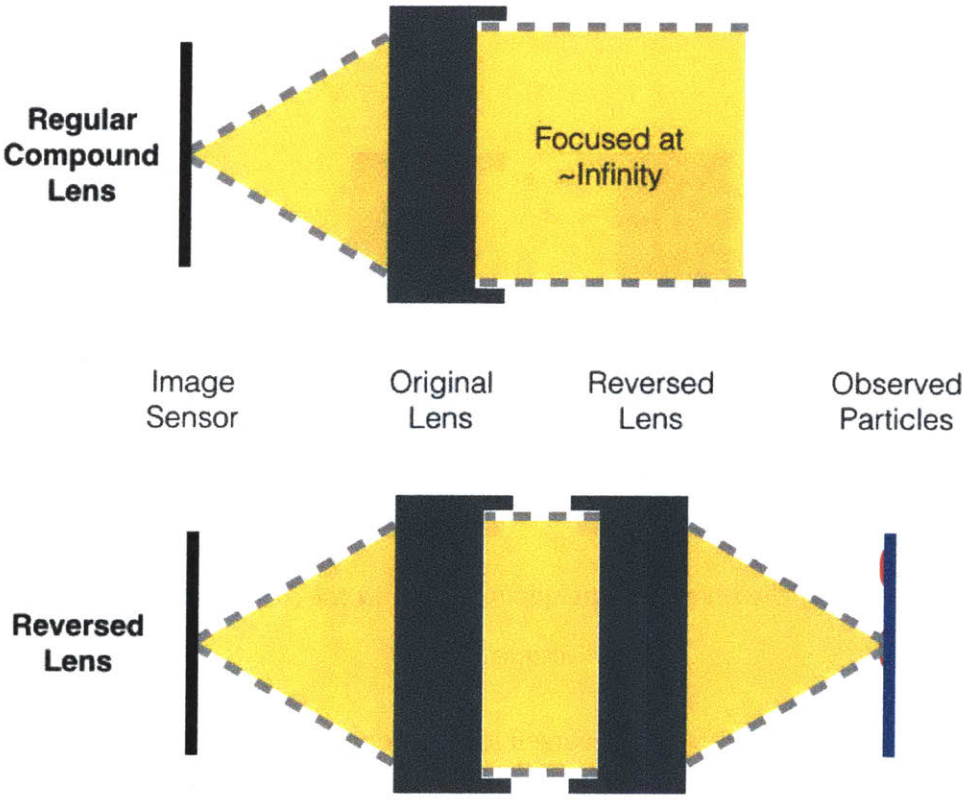



*Figure 3-2. In the reversed lens approach, we couple two identical compound lenses by attaching the front of one lens with the front of the other lens, aligning the lens pupils.*

In this work, we implemented the reversed lens approach pioneered by Switz et al[7]. In the reversed lens approach, we couple the original lens with the same lens but with it's direction reversed. The reversed lens improves the optical properties of the lensing system, ultimately

enabling a higher resolution. The benefits of this approach are similar to the benefits of selecting the base lens in the first place: an optimized numerical aperture with minimized aberrations in a small form factor at a low cost. In addition, since the same lens design is used twice, the lenses are matched.

### 3.2.1 Characterizing the Optics

The goal is to derive the properties of the entire lensing system using a combination of the properties of the individual lenses and the system design. In this section, we will reference a variety of online application notes provided by Nikon, Zeiss, Edmund Optics and the National High Magnetic Field Laboratory[18,19,24,25].

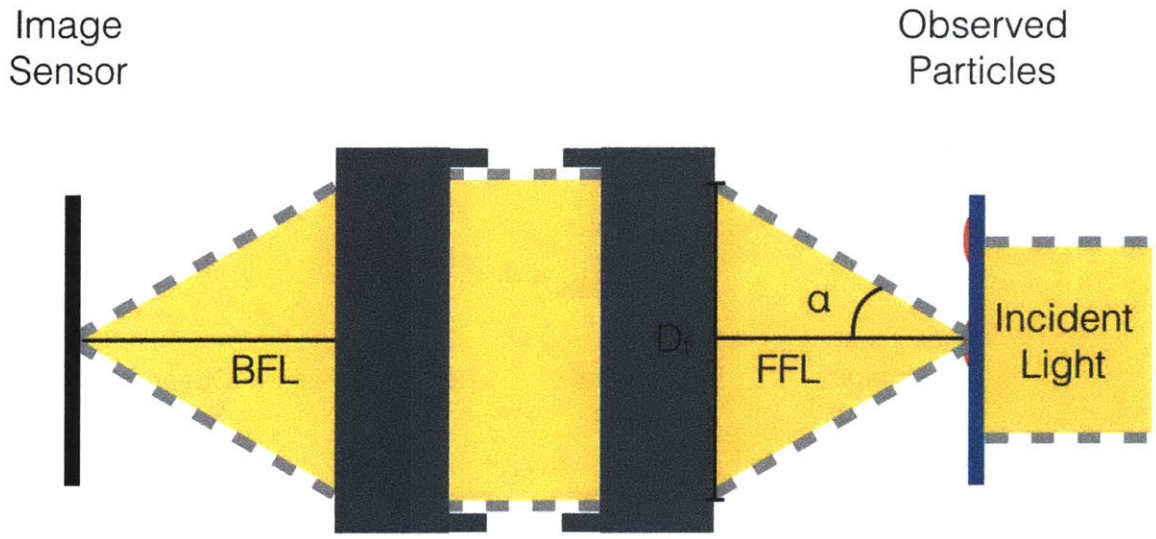



*Figure 3-3. Diagram of the reversed lens system showing the incident propagating toward the image sensor.*

*Table 2. Optical characteristics of the reversed lens system.*

| Lens Properties | Value |
|---|---|
| Back Focal Length (BFL) | 3.04mm |
| Front Focal Length (FFL) | 3.04mm |
| Primary Magnification (PMAG) | 1 |
| Sensor Size | 3.68mm x 2.76mm |
| Field of View | 3.68mm x 2.76mm |

| | |
|---|---|
| Focal Length of System ($\boldsymbol{f}$) | 1.52mm |
| Effective Aperture (Pupil Diameter) | 1.52mm |
| f-Number (Focal Ratio) | 1.0 |
| Object-Side Numerical Aperture ($\boldsymbol{NA_o}$) | 0.241 |
| System Numerical Aperture ($\boldsymbol{NA}$) | 0.482 |

*3.2.1.1 Primary Magnification*

Primary magnification (PMAG) is the ratio between the sensor size and the field of view. Specifically, the PMAG can be calculated with the following equation:

$$PMAG = \frac{BFL}{FFL}$$

where BFL (back focal length) is the the distance from the back focal point (usually the image sensor) to the back vertex of the lens and FFL (front focal length) is the distance from the front focal point (usually the object) to the front vertex of the lens. When the camera is focused on objects far away from the lens, the PMAG converges to 0. For the reversed lens system, the PMAG is calculated to be 1.

*3.2.1.2 Field of View*

Field of view is the area of the in-focus object that is projected onto the image sensor. For urinalysis, a large field of view is important because it allows us to interrogate a higher volume of urine for each image, which provides a more accurate indication of the concentration of particles in the urine sample. By increasing the PMAG of the lensing system, the field of view is decreased. Field of view can be calculated using PMAG:

$$field\ of\ view = \frac{sensor\ size}{PMAG}$$

Thus, the field of view is equal to the sensor size for the reversed lens system: 3.68mm x 2.76mm.

*3.2.1.3 Focal Length*

The focal length is a measure of a lenses ability to bend light. Specifically, its the distance between the lens and the point at which the lens bends parallel rays of light into focus

(usually the point of intersection on the optical axis). Using a thin lens approximation for the reversed lens system, the focal length $f$ is:

$$f = \frac{1}{BFL} + \frac{1}{FFL}$$

With a focal length of 1.52mm, the reversed lens system has half the focal length of the system's individual lenses. This makes intuitive sense since we are combining the optical power of two lenses with the same optical power, thus doubling the system's optical power or halving the system's focal length.

#### *3.2.1.4 Numerical Aperture*

The numerical aperture ($NA$) is a measure of the ability of a lens to capture light. The object-side numerical aperture, $NA_o$, is especially important in calculating the diffraction limited resolution of a lens system. The $NA_o$is determined by the maximal angle $\alpha$ in Figure 3-3 where the lens can effectively accept light, and refocus it on the image sensor. Mathematically, $NA_o$ is represented as:

$$NA_o = nsin(\alpha)$$

where n is the index of refraction (1.00 for air), and $\alpha$ is the maximal half-angle of the cone of light that the lens system can capture, and refocus on the image sensor. Specifically, $\alpha$ can be calculated by:

$$\alpha = \arctan\left(\frac{D_f}{2 * FFL}\right)$$

where $D_f$ is the effective entrance (front) pupil diameter of the lens. For perfect reconstruction, the lens needs to capture the 180°-degree cone created by diffraction, which can theoretically be accomplished with a FFL much smaller than $D_f$. The $NA_o$ was calculated to be 0.241. To compare NAs across different types of lenses irrespective of the selected focal planes, we calculate the system's NA to be 0.482, which is similar to an objective lens that provides 20x magnification.

### 3.2.2 Characterizing the Resolution

The resolution is a critical design parameter since it defines the smallest feature that can still be distinguished. Multiple factors determine the resolution of a system, including lens

aberrations, focus, lighting, contrast, etc. The microscope system is composed of an optical resolution and a digital resolution. Whichever of these two resolutions is lower limits the resolution of the entire microscope system. Table 3 summarizes the theoretical optical and digital resolutions achieved with the AutoScope imaging system. We show the process of deriving these resolutions across the next two sections.

*Table 3. A comparison of optical and digital estimates across different wavelengths.*

| Color (Wavelength) | Optical Resolution Estimate (w/ Rayleigh Criterion) | Digital Resolution Estimate (w/ Bayer Filter) |
|---|---|---|
| Blue (460nm) | 2.33um | 4.56um |
| Green (515nm) | 2.61um | 3.22um |
| Red (600nm) | 3.04um | 4.56um |

### *3.2.2.1 Diffraction-Limited Resolution*

In an ideal system, there is a theoretical limit to the optical resolution defined as the diffraction-limited resolution. Diffraction is the bending of a light wave around an object that has a size in the same range as the wavelength of the light. In Figure 3-4, an incident light wave is diffracted by an object with a width of $w$. The inverted cone represents the diffraction pattern, which includes a primary maximum ($0^{th}$ order lobe) in the same direction as the incident light beam, and secondary maxima that fan around the primary maxima. The $0^{th}$ order lobe has the maximum intensity while the side lobes having increasingly smaller luminous intensities. The diffraction pattern fans out in 180-degrees around the $0^{th}$ order lobe, with another maximum occurring at every $\theta$ degrees. The angle of diffraction, $\theta$, is determined by:

$$\sin(\theta) = \frac{\lambda}{w}$$

where $\lambda$ is the wavelength of the incident light and $w$ is the width of the object as shown in Figure 3-4.

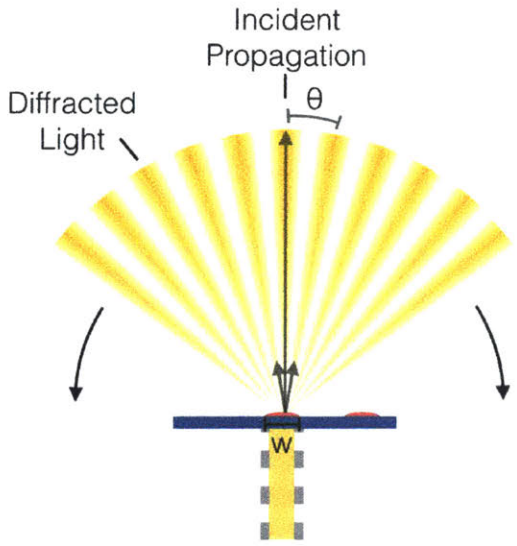



*Figure 3-4. A diffraction pattern created by an incident wavefront passing around an object with width w. The diffraction pattern fans out to create a spherical wavefront with each lobe separated by θ degrees.*

The diffraction limited resolution is determined by the amount of diffracted lobes that can be captured by the lens and refocused on the image sensor plane. As an object gets smaller ($w$ decreases), the diffraction pattern widens ($\theta$ increases). As $\theta$ increases, less of the diffracted lobes will be effectively refocused by the objective lens onto image sensor. With incoherent illumination, the theoretical resolution is given by:

$$diffraction - limited\ resolution = \frac{\lambda}{NA_o}$$

where $\lambda$ is the wavelength of the incident light.

Although the diffraction limited resolution is the best possible optical resolution, the Rayleigh Criterion is used to determine a more practical resolution that takes visual perception into account. Specifically, the Rayleigh Criterion assumes a 20% contrast is needed between the two features being differentiated. The Rayleigh Criterion resolution is given by:

$$Rayleigh\ criterion\ resolution = \frac{1.22 * \lambda}{NA_o}$$

where $\lambda$ is the wavelength of the incident light. The 1.22 is included to take into account the 20% contrast in between the maxima of the airy disks. It's important to note that there is no factor of 2 in the denominator since our system is not illuminated through an optical condenser (like most microscopes have). The Rayleigh criterion resolution limits for the optical RGB resolution of our system are given in Table 3.

#### *3.2.2.2 Digital Resolution*

The digital sampling of images can also limit the end-to-end resolution of a system. The distance between adjacent pixels on the image sensor represent the sampling interval. The maximal spatial resolution is determined by the Nyquist-Shannon sampling theorem that states the sampling interval must be less than or equal to half the size of the smallest measured feature. If this criterion is not met, then spatial aliasing will occur. Since our system has a PMAG of 1, the spatial resolution of the image sensor directly maps to the spatial resolution of the field of view.

In this work, the Sony IMX219 image sensor is used for sampling. The pixel size is 1.12um x 1.12um. The sensor uses a Bayer filter to produce a RGB (red, blue green) image. With a Bayer filter, each pixel has an optical filter in front of it that pass through a certain spectrum of light, usually either red, green or blue regions. The spectral response of the RGB filters in IMX219 is provided in Figure 3-5. During the post-processing, a demosaicing algorithm interpolates the data from the Bayer pattern so that each pixel has a RGB value. Due to the Bayer filter and chromatic distortions, the sampling interval is not directly determined by pixel size. Instead, we use the Bayer pattern to give a more reliable indication of the sampling interval (with the caveat that we could achieve a smaller sampling interval depending on the spectrums of the illuminating light and the object under inspection). If we removed the Bayer filter, we would achieve a better digital resolution with the tradeoff of not having any color information (which our current algorithm doesn't use anyway). Green pixels have a sampling interval of $pixel\ size * \sqrt{2}$ and red/blue pixels have a sampling interval of $pixel\ size * 2$. The spatial resolution estimates are given in Table 3.

Based on Table 3, we can see that the optical resolutions and spatial resolutions are closest at the 515nm, which makes sense since the image sensor has the highest density of green pixels.

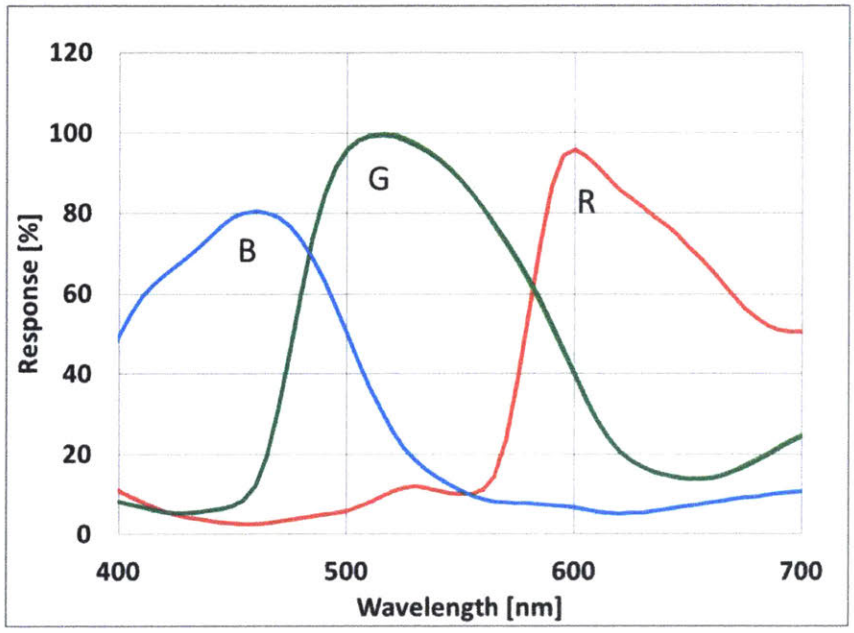



*Figure 3-5. The spectral response of the per pixel RGB filters in the IMX219*[17].

### *3.2.2.3 Experimentally-Determined Resolution*

In this section, we will determine the end-to-end system resolution experimentally. Due to practical limitations (like aberrations in the lens, contrast and appropriate focus), the experimental resolution is usually worse than than the estimated resolution of an ideal system. To determine the resolution experimentally, we used the 1951 target, which is composed of a group of elements, where each element consists of three horizontal and three vertical bars (see Figure 3-6). Each element has a fixed line spacing, thus creating a high contrast object that maps to a defined resolution. As defined by the Rayleigh criterion, the experimental resolution of the system is the resolution of the element where the contrast between the black and white lines is less than 20%. To determine the contrast of an element, we averaged the values of all the pixels across each column that corresponding to the black and white lines of an element. The resulting graph, with pixel intensity vs. relative column position can be seen in Figure 3-6. From this graph, we determined the contrast of the element with our imaging system. The highest element that still had a contrast of over 20% determined the resolution.

The measured resolution is worse than both the estimated spatial and optical resolution. First, this means that we are not limited by the spatial resolution of the image sensor, which could have been readily improved if necessary. Second, the reason for the decrease in resolution is probably optical distortions. Specifically, misalignment between the two conjoined lenses can cause significant decreases in resolution quality. The manual assembly of the lenses makes the misalignment likely. Other issues, such as illumination and manufactured lens aberrations, can cause an additional degradation in resolution. Switz et al showed a measured resolution that aligned more closely with their calculated limiting resolution. Using a different set of reversed lenses, they showed a measured resolution of between 3.9-6.2um, which overlapped with the range of their digital resolution (4.8-6.8um)[7].

As can be seen in Figure 3-6, the resolution decreases as we move from the center of the field of view towards the edges of the field of view. This is expected because of spherical aberrations that alter the focal distance as we move away from the optical axis.

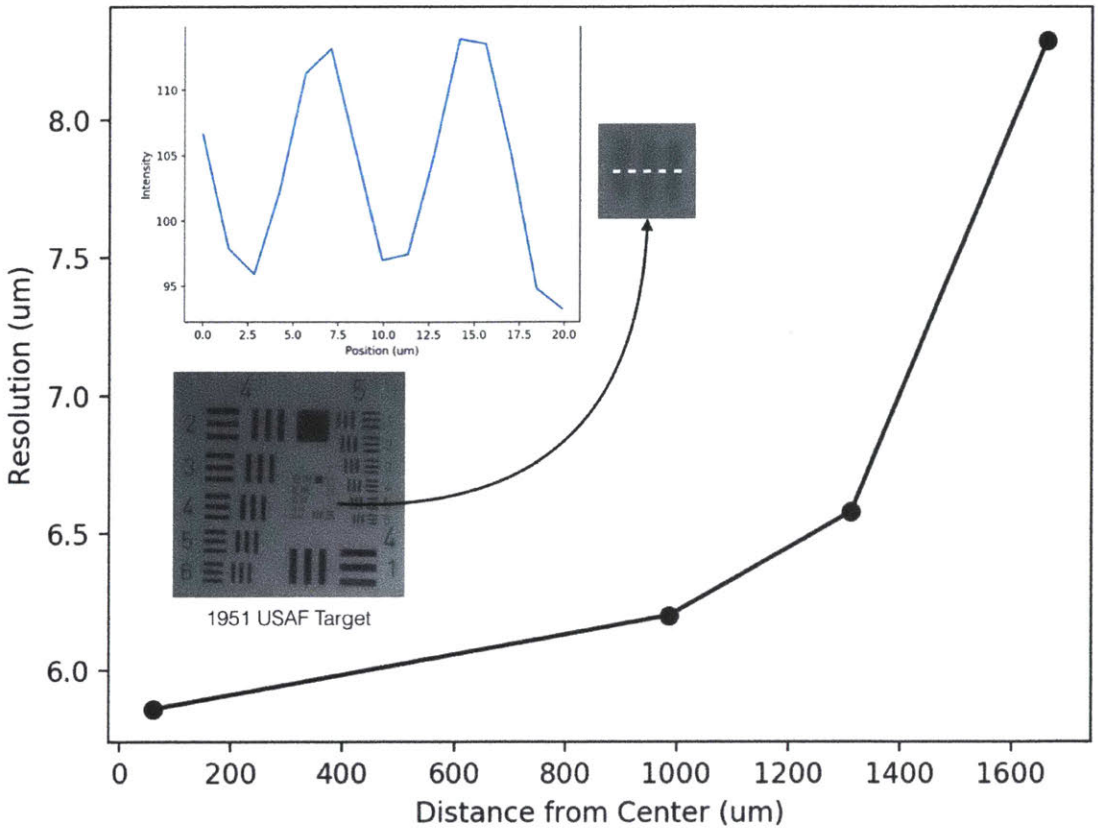



*Figure 3-6. The practical end-to-end resolution of the AutoScope imaging system measured at different points on the field of view.*

# 4 Particle Classification

In this chapter, we will describe the AutoScope's particle classification algorithm. First, we will introduce general techniques for the effective training of convolutional neural networks (CNNs). This will be leveraged through out the rest of our work. Second, we describe the base CNN architecture on which all future particle classification networks will be based on. Third, we will discuss an experiment that shows the feasibility of classifying urine particles at the resolution of the AutoScope. Finally, we will demonstrate that the AutoScope can accurately classify red blood cells and white blood cells.

## 4.1 Background: Preparing Data for Training

### 4.1.1 Training, Validation and Test Sets

In order to train CNNs, the data should be separated into a training set, a validation set and a test set. The majority of the data is placed in the training set, which is used during the SGD to update parameters of the CNN. In contrast, the test and validations sets are never used to update the CNN's parameters. The validation set is used to evaluate the networks accuracy during training, helping to tune the hyperparameters and identify when the CNN starts to overfit to the training data. Overfitting occurs when the predictive power of a CNN no longer generalizes to new inputs, but instead starts learning features that are unique to the training set. Overfitting can be identified when the accuracy of the training dataset continues to increase while the accuracy of the validation dataset remains flat, or even decreases. The test set is used for testing the final solution, evaluating the predictive power of the network with inputs it has never seen before. In our case, the test set is the final clinical validation done with synthetic urine, which we discuss in Chapter 6.

CNNs are trained on batches of images. So, for each forward and backpropagation through a CNN, we send a set of images through the network simultaneously. The benefit of using a larger batch size is a decrease in the variability of the loss gradient, improving the stability of the convergence. The benefit of using a smaller batch size is the increased speed of the forward/backward propagation.

### 4.1.2 Preprocessing

Before an image is passed through the CNN, we perform three transformations on the image: resizing, data normalization and augmentation. Here, we introduce the over-arching concepts behind these transformations. We will discuss the specific implementation used with each algorithm once we describe the corresponding algorithm.

#### *4.1.2.1 Resizing Image Dimensions*

In order to feedforward/feedbackward an entire batch of images simultaneously, each image in a batch must have the same dimensions. This enables 4-D matrix multiplication, improving the computational efficiency of the CNN.

#### *4.1.2.2 Data Normalization*

Data normalization is a set of transformations that normalize the distribution of the input data. Without a normalized data distribution, the CNN would focus on absolute feature values, and implicitly emphasize higher intensity pixels. With a normalized data distribution, the CNN focuses on relative values, and thus is more likely to converge on a single set of filters that can be applied across a diverse dataset.

Common approaches to data normalization include mean-subtraction, rescaling, feature standardization and PCA/whitening. The most common data normalization approach is mean-subtraction, where a mean intensity value is subtracted from each color channel of a pixel. The second most common approach is rescaling, where the value of each pixel is rescaled by either a standard deviation of pixel values, a maximum or a fixed value.

#### *4.1.2.3 Augmentation*

A trained CNN must be invariant to certain image transformations, often including translation, reflection, and rotation. Furthermore, the size of training datasets in medical imaging is often leaner than necessary for robust model training. Data augmentation is a technique that applies geometric transformations on images that the CNN should be invariant to. The result is a larger dataset that improves the CNN's invariance. In this work, we augment the dataset by applying random transformations each time an image is passed through the CNN. Specifically,

we perform rotations, mirrors and translations, with the magnitude of these transformations randomly selected from a predetermined range.

## 4.2 Base CNN Architecture

In this section, we will introduce the base CNN architecture on which the more complex architectures in this work were based. Again, we will present the unique implementations of the base CNN architecture as we introduce the specific algorithms. We have discussed each building block of the CNN previously, and here we will combine these building blocks into an architecture used for classifying image inputs into different types of cells.

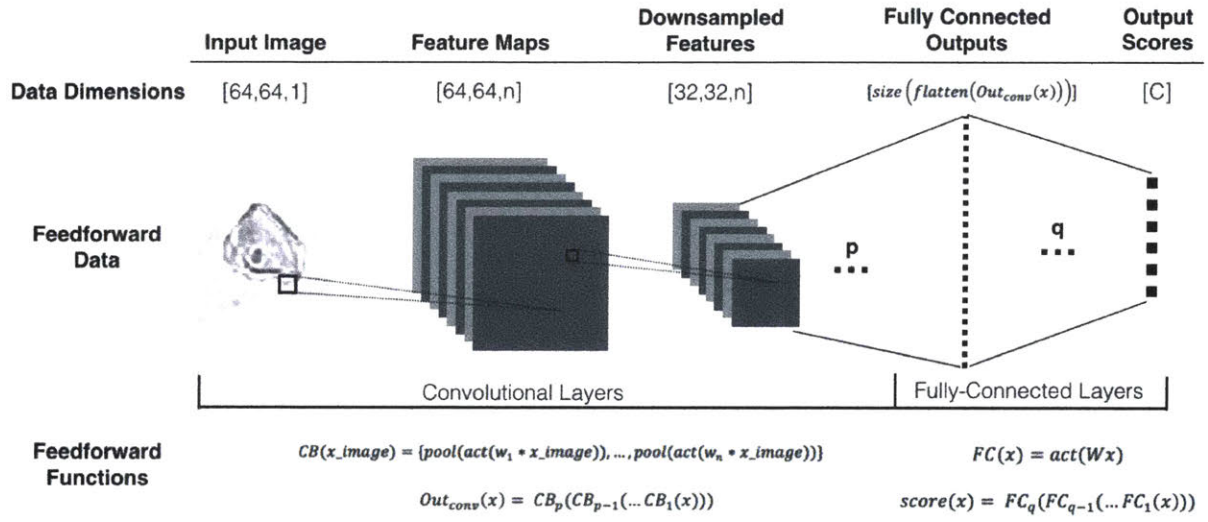



*Figure 4-1. The figure presents both the feedforward data pipeline and feedforward functions of the base CNN architecture. The feedforward data pipeline is a visual representation of a single input image being processed by the CNN. The base architecture can be customized with p (the number of convolutional blocks), q (the number of fully connected layers), n (the number of filters for a convolutional block) and C (the number of output classes). The feedforward function representation was inspired by Van Valen et al*[26].

Using a network of feedforward functions parameterized with learned weights, the CNN maps an input image into a set of output scores, producing a single score for each of $C$ classes. For our CNNs, we use $p$ convolutional blocks, where $p$ usually is between 2 to 4. Within each convolutional block, we have filter layer(s), with each filter layer having $n$ filters. Sometimes, a single filter layer ($w_1$-$w_n$ in Figure 4-1) is split into two back-to-back filter layers consisting of

smaller filters, which reduces the size of a network without sacrificing functionality. Following the filter layer(s) is an element-wise activation layer, implementing a ReLU function across each element. The final layer for each convolutional block is the max pool layer, which uses a 2x2 window with a stride of 2. The input image is resized to 64x64 pixels, with a depth of 1 channel.

The convolution blocks are followed by $q$ fully-connected layers, where $q$ usually is between 2-4. All fully-connected layers use the ReLU function as the activation function, except for the output fully-connected layer where $act(x) = x$. The input to the first fully-connected layer is the output of the last convolutional block, flattened into an array of features. The final output is an array of $C$ scores, with each score mapping to one of the $C$ classes.

In this work, all CNNs were implemented in Python using a combination of Tensorflow, Keras, Numpy, and Scipy[27–30].

## 4.3 Resolution Experiment

The system presented in this work will need to accurately classify particles at significantly lower resolutions than existing systems. The goal of this experiment is to quantify the classification accuracy of urine particles as the resolution of those particles is decreased. Specifically, we aim to demonstrate a proof of concept algorithm that shows that we can accurately classify urine particles at a resolution at or below the image acquisition system presented in Chapter 3. We used a pre-labeled dataset provided by Iris Diagnostics to train and validate the CNN. In the end, we obtained an accuracy of 84.7%-90.0% for the resolutions that we expect with the AutoScope system.

### 4.3.1 The Dataset

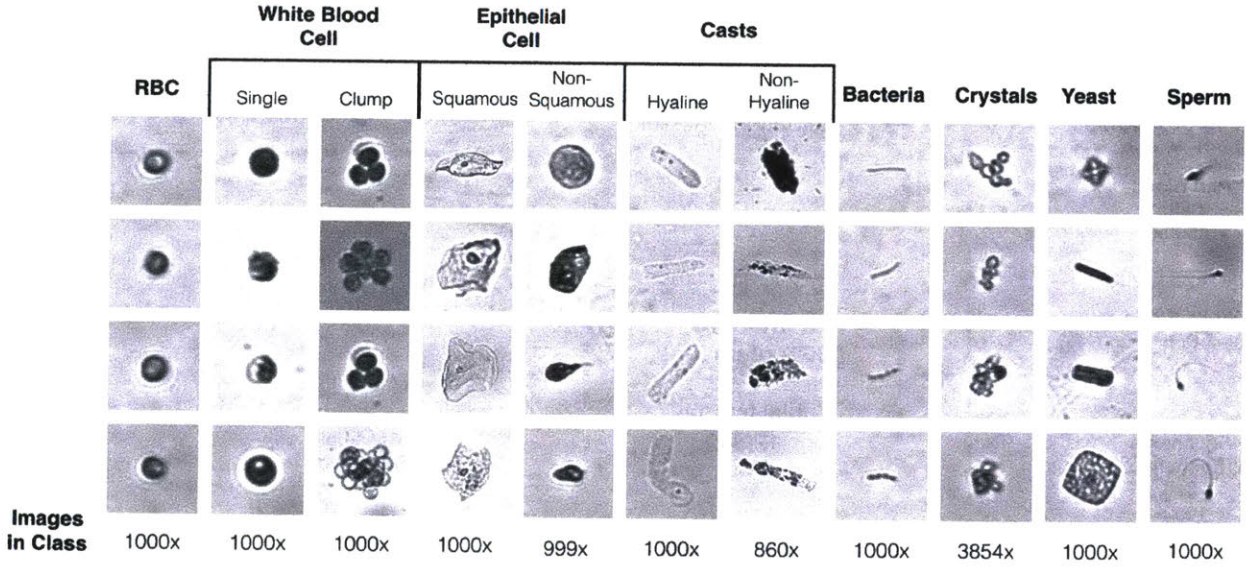



*Figure 4-2. From Iris Diagnostics, we received 13,713 prelabeled urine particles. A few representative examples are shown in this figure.*

The dataset is composed of 13,713 grayscale images of urine particles, with dimensions ranging from 36x36 pixels to 250x250 pixels[31]. Each image is pre-labeled with one of sixteen classes. Figure 4-2 showcases images from the main particle classes, including (from left to right): red blood cells, white blood cells, epithelial cells, casts, bacteria, crystals, yeast and sperm. The images are provided by Iris Diagnostics, and acquired with the iQ-200, a urinanalyis device with a retail price of $100,000 to $150,000. The iQ-200 has an estimated digital resolution of 1.36um. With a 20x objective and no eyepiece, the iQ-200 might have an optical resolution that is around the digital resolution.

### 4.3.2 Data Preparation

For the resolution experiment, the dataset was split into six classes: red blood cells, white blood cells, squamous epithelial cells, casts, bacteria and other. These classes were selected due to their clinical relevance and popularity in urine testing. As can be seen in Figure 4-3, 90% of the images in each class were used for the training dataset, and 10% of the images were used for validation. During the preprocessing phase of data preparation, we performed the normal steps, including image resizing, augmentation, and normalization. For the normalization, we performed

zero-mean subtraction based on the mean pixel value of the color channel. Resampling and normalization is done to all images, while augmentation is only done with the training dataset.

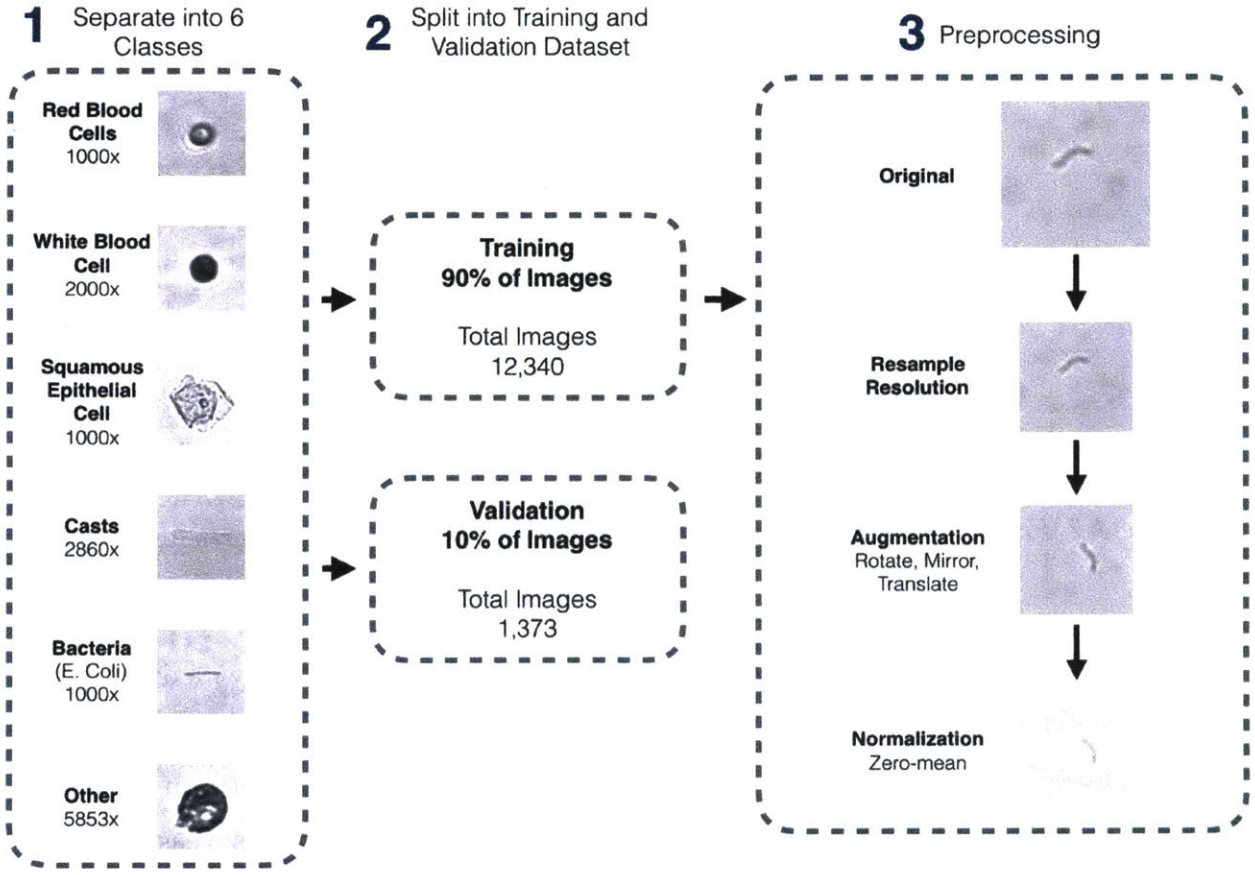



*Figure 4-3. The dataset was 1) split into 6 classes of urine particles, 2) split into a training and validation set and 3) fed through a preprocessing function during training.*

During the resizing step, we performed additional transformations on the image to reduce the image resolution. Specifically, the image was transformed to the target digital resolution by first filtering the image with an antialiasing filter, and then downsampling the image based on the proportion of the original resolution to the target resolution. The resulting image was upsampled again, using bilinear resampling, to a final dimension of 64x64px. This final upsampling step ensures consistent dimensions for each batch of images. It's important to note that downsampling an image that is limited by it's digital resolution permanently removes information represented by the image. Thus, when a downsampled image is upsampled, the practical resolution of the image remains the same. In other words, the information contained in the upsampled image is the same the same as the information contained in the image prior to upsampling.

We plan to evaluate the classification accuracy at 5 different resolutions: 1.36um (the estimated accuracy of the iQ-200), 2.50um (the approximate resolution of a 100x microscope), 4.56um (the best-case resolution of the AutoScope), 10um (the worst case resolution of the AutoScope) and 25um. The 25um resolutions were selected as a sanity check, and to test the limits of the CNNs performance.

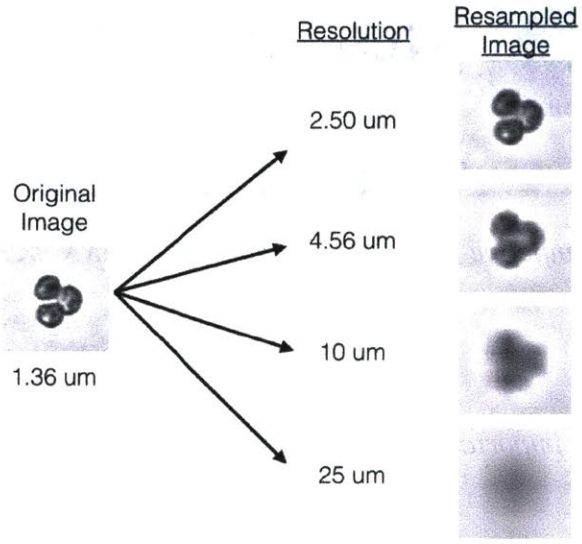



*Figure 4-4. To decrease the resolution of the original images, we resampled the images to produce 5 datasets with unique resolutions.*

### 4.3.3 CNN Architecture

The CNN architecture stems from the base CNN architecture presented previously in this chapter. The exact data pipeline for the architecture is shown in Figure 4-5. Note that the first convolutional block has 32 filters and the second convolutional block has 64 filters.

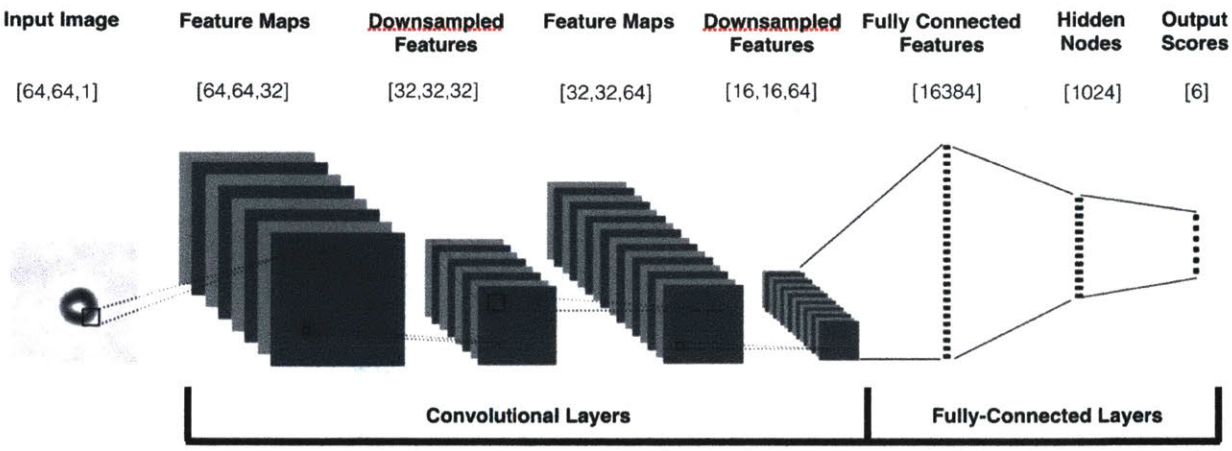



*Figure 4-5. The data pipeline for a single image for the CNN architecture utilized in the resolution experiment.*

*Table 4. A summary of the important parameters for the CNN implemented in the resolution experiment.*

| Parameters | Values |
|---|---|
| Convolutional Blocks (p) | 2 |
| Fully-Connected Layers (q) | 2 |
| Batch Size | 64 |
| Type of Optimizer | Adam Optimizer |
| Learning Rate | 0.0001 |
| Color | Grayscale |
| Image Dimensions | 64x64 pixels |
| Total Parameters (weights + biases) | 16,847,332 parameters (~64MB) |

### 4.3.4 Results

First, we will discuss the classification accuracies of the CNN without any reduction in particle resolution. In this case, the CNN achieved a 91.4% accuracy in classifying the original image into the correct class. During training and validation, each class is equally represented in a batch. Thus, selecting classes at random would provide an accuracy of 16.7%. We see a significant improvement from this accuracy.

In Figure 4-6, the accuracy of each class is shown during the training of the CNN. Specifically, as the CNN is trained, the validation dataset is intermittently used to determine the predictive power of the CNN. As can be seen, the CNN achieves the highest accuracy when classifying the bacteria, probably due to the unique shape of the E. Coli. The CNN achieves the lowest accuracy for the classes that have significant intra-class variability, including "Other", "Cast" and "Squamous Epithelial" classes. The confusion matrix provides a visual representation of the CNN's misclassifications. For the confusion matrix, we use the average results for the final few validation batches.

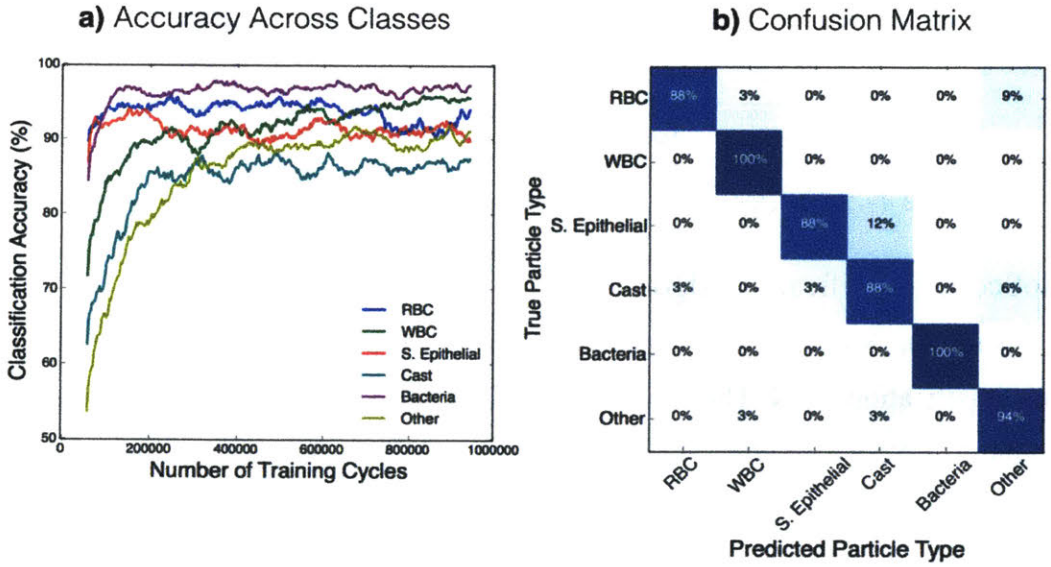



*Figure 4-6. (a) The classification accuracy across each class of particles at a resolution of 1.36um. (b) The confusion matrix, showing the average results across classes for the final few validation batches.*

Next, we present the accuracy of the CNN as the image resolutions are decreased. As can be seen in Figure 4-7, the overall accuracy of a CNN clearly decreases as the resolution decreases. The accuracy decreased by 1.4% from 91.4% at resolution of 1.36um to 90.0% at a resolution of 4.56um. The resolution is reduced by a factor of 3.35 while the accuracy is only reduced by 1.4%. This result implies that majority of features necessary to accurately classify the particles is still included in the information represented in images with a resolution of 4.56um. Similar results hold for a resolution of 10um at 84.7% accuracy. More importantly, these results

imply that the the AutoScope should have sufficient resolution to classify the clinically relevant urine particles at an accuracy of at least 84.7%.

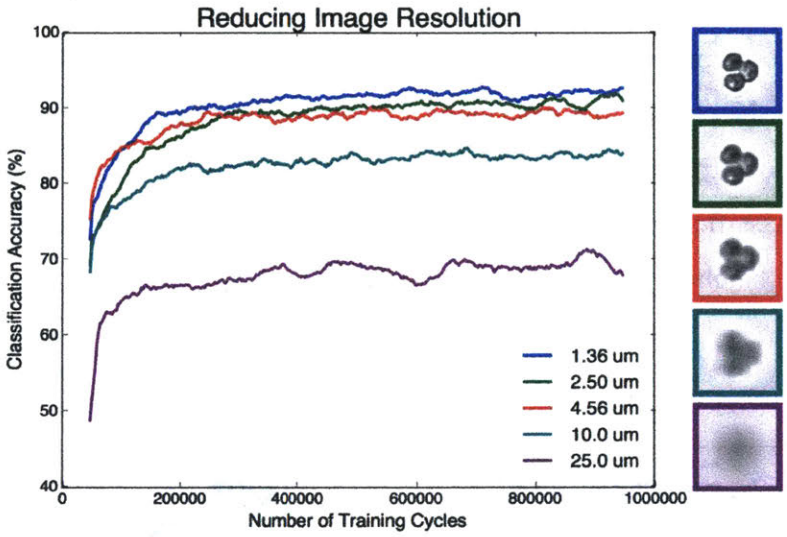



*Figure 4-7. The accuracy results across different particle resolutions.*

## 4.4 AutoScope Classification Algorithm

In this section, we will present the design, training protocol and results from the AutoScope classification CNN. The goal of this CNN is to accurately classify a crop of a particle taken from the AutoScope into one of four particle classes. To train the CNN, we use human labeled particle crops. In the end, we achieve a classification accuracy of 89.3%, which is in the ranged predicted by the resolution experiment.

### 4.4.1 The Dataset

For the end-to-end automated urinalysis, we selected the following target particles: red blood cells, white blood cells and 10um microbeads. Red blood cells (RBCs) and white blood cells (WBCs) were selected because they are commonly requested as a diagnostic indicator by clinicians and are difficult to detect/differentiate. We included 10um microbeads because their size and shape mimic white blood cells. The RBCs were obtained from fresh human whole blood provided by Research Blood Components (in Boston, MA). The WBCs were BA/F3 cells from Thermo Fisher Scientific. BA/F3 cells are lymphocytes, a sub-type of leukocytes (or white blood cells). Specifically, BA/F3 cells are B cells, which produce antibodies that identify invading germs. The BA/F3 cells were cultured in-house in a medium that included fetal bovine serum.

The fetal bovine serum included numerous stray particles, which increased the classification complexity. Specifically, the fetal bovine serum introduced significant contamination that the AutoScope system needs to classify into the 'other' classification category. The microbeads were obtained from Polysciences, Inc.

*Table 5. Summary of the characteristics of the target particles that the AutoScope is trained on.*

| | Red Blood Cells | White Blood Cells | Microbeads |
|---|---|---|---|
| Type of Particle | Erythrocytes | Lymphocytes (B Cells) | 10um Fluoresbrite Carboxylate |
| Source | Human | Mouse | N/A |
| Diameter | 6-8um | 10-16um | 9-11um |
| Vendor | Research Blood Components, LLC | Thermo Fischer Scientific | Polysciences, Inc |
| Vendor Information | N/A | BA/F3 Cells | Catalogue #: 18142 |

To train the CNN, we needed to generate a labeled training and validation dataset. We accomplished this in a 3 step process. First, we used the Autoscope to take images of a slide with only a single particle on it. Second, we manually identified the location of each particle within the AutoScope images using a python GUI. Third, we labeled each segmented particle as either the target particle or other with another python GUI. We knew the identity of the target particle since we knew the specific particle the slide contained. Using this process, we obtained crops of labeled particles on which the CNN could be trained on.

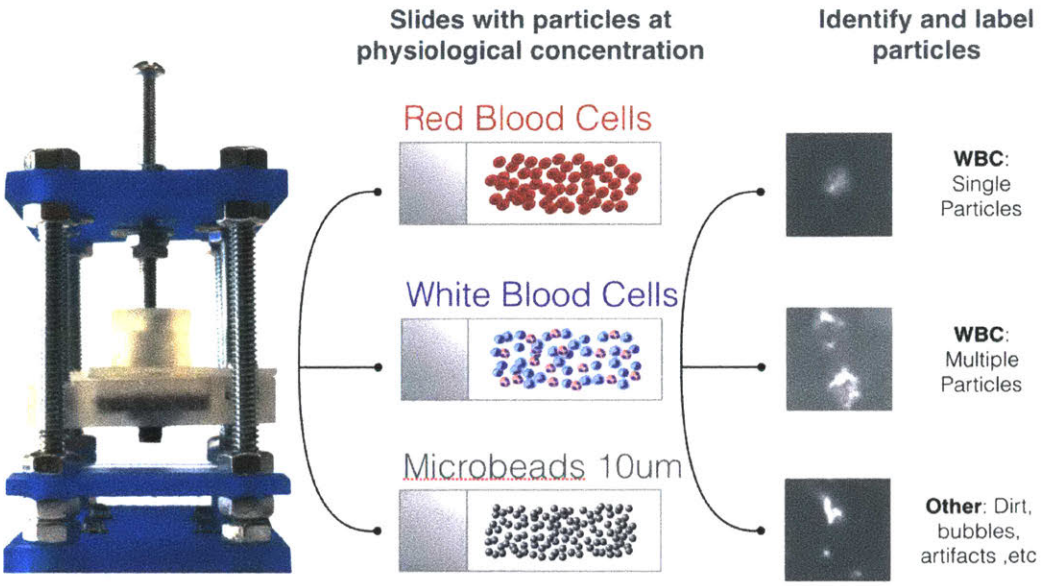



*Figure 4-8. We manually labeled particles to train the CNN by preparing slides that included only a single target particle.*

### 4.4.2 Data Preparation

The dataset includes a total of 2,442 crops of particles, with the per class breakdown shown in Table 6. 90% of each class was allocated for training while 10% of each class was allocated for validation. Figure 4-9 shows examples of a few particles across the 4 classes. These images are shown in grayscale since the CNN has the best performance with single color channel images. The AutoScope generates color images that we then converted to grayscale. In future work, we will be able to obtain higher-resolution images by dropping the color filer in the image sensor, and thus directly obtaining a higher-resolution grayscale image.

As with the resolution experiment, we augmented and normalized the images as they were fed into the CNN. We augmented the dataset by applying random rotations, shifts, flips and mirrors. We performed zero-mean subtraction based on the mean of each image, and rescaled the pixel values by 128 (the maximum magnitude after a zero-mean is applied to each pixel). We experimented with multiple different approaches to preprocess the images, including RGB images, varying the intensity of the augmentation, performing mean-zeroing based on the entire

dataset instead of each image, etc. Out of all the variations, we selected the highest performing implementation.

*Table 6. Summary of the number of images for each target particle.*

| Class | Total Images |
|---|---|
| Red Blood Cells | 675 |
| White Blood Cells | 226 |
| 10um Beads | 538 |
| Other | 1,003 |
| Total | 2,442 |

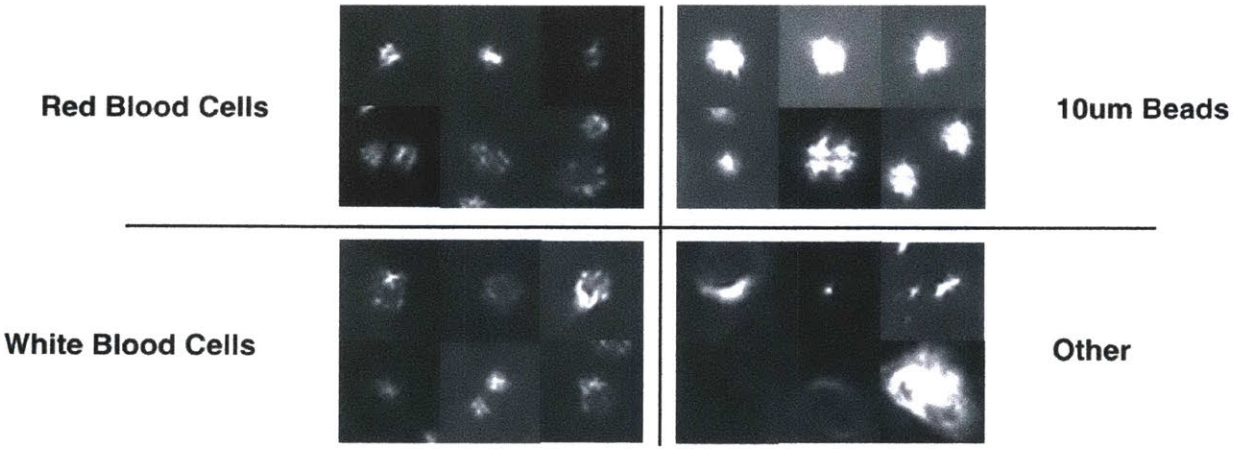



*Figure 4-9. Example particles used for CNN training across the 4 classes.*

### 4.4.3 CNN Architecture

The CNN architecture used to classify particles with the AutoScope is nearly identical to the CNN architecture used in the resolution experiment. In fact, the parameters in Table 4 transfer directly to the AutoScope classification architecture (except for the number of trainable parameters which now is 16,853,500).

The main difference is that we manually calculate custom features that we feed into the fully-connected layers of the CNN. These custom features are characteristics of the particles that the CNN cannot implicitly determine. Specifically, we provide the location of the particle within the

main AutoScope image. The illumination (both the intensity and angle of illumination) is highly dependent on its location within the main AutoScope image. By using only the cropped particles, the CNN cannot readily pinpoint this location. Thus, with these custom features, the CNN can weight different features depending on the location of the particle within the field-of-view, improving classification accuracy. We provide the location of the particle in terms of both the polar coordinates and the Cartesian coordinates.

We tried multiple different architectures, including CNNs that used batch normalization, CNNs with more convolutional blocks, CNNs that used transfer learning from the original VGG16 ImageNet, a variety of hyperparameters, etc. From all of these iterations, we chose this architecture because it provided high accuracy at a low-complexity.

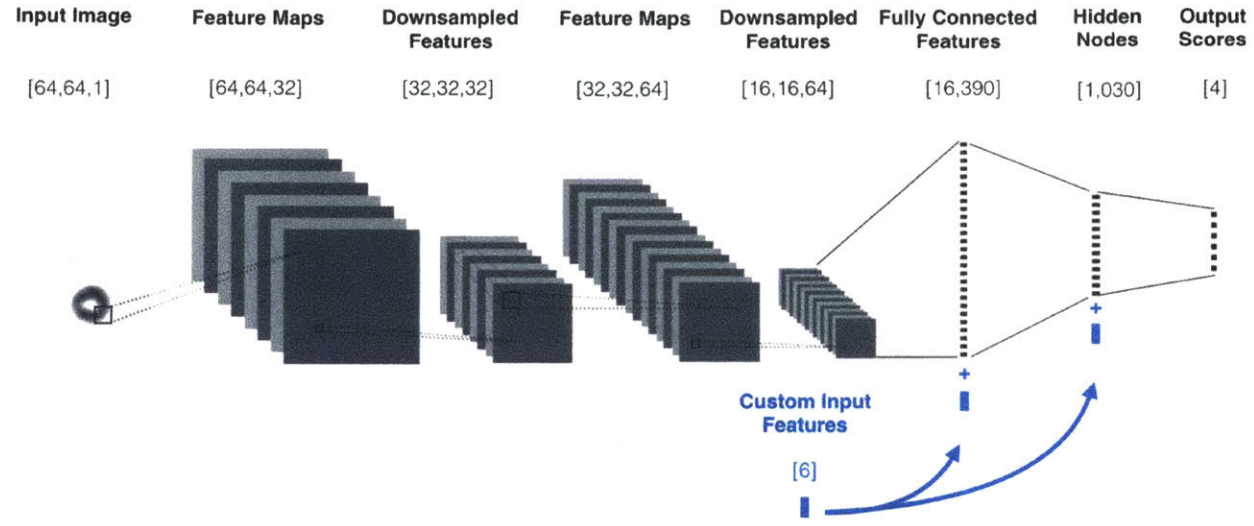



*Figure 4-10. The data pipeline for a single image for the CNN architecture utilized in the AutoScope classification algorithm.*

### 4.4.4 Results

As with all CNNs in this work, we trained the CNN on GPUs in the AWS cloud. Figure 4-11 shows the training results across the number of training cycles. If we randomly sorted particles into classes, we would achieve an accuracy of 25%. During training and validation, the accuracy of the CNN converged to 89.3%. Since the the training accuracy and the validation accuracy does not diverge, we can infer that the CNN is not overfitting to the training data, but still generalizes to untrained inputs. The final accuracy of 89.3% is between the predicted

accuracies of the resolution experiment: 84.7% (for 10um) to 90.0% (for 4.56um). Thus, these results align with the results from the resolution experiment.

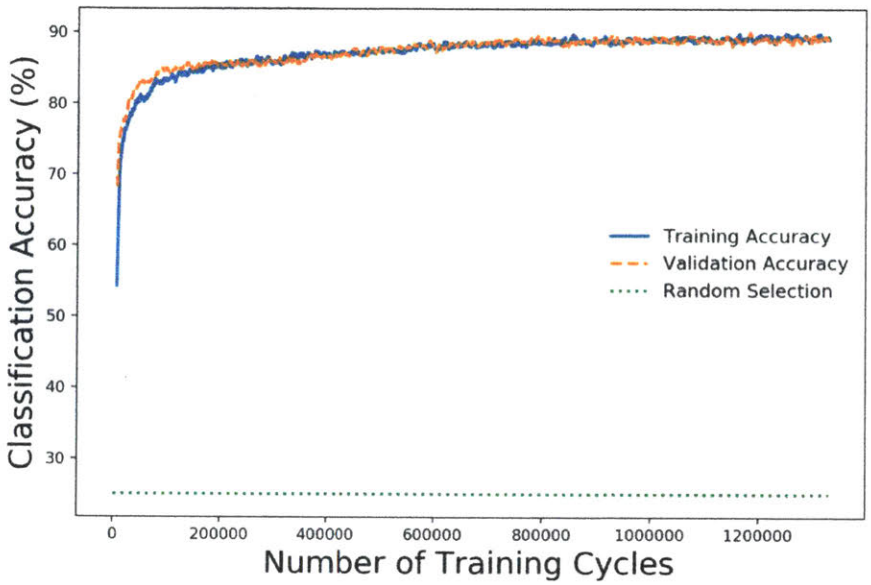



*Figure 4-11. The classification accuracy during training of the AutoScope particle classification algorithm.*

Figure 4-12 shows the accuracies across different classes of particles. Most importantly, we observe the lowest accuracies with white blood cells, having an average final accuracy of 64%. A likely explanation for this result is the medium from which we obtain the white blood cells from. Specifically, the BA/F3 white blood cells are obtained from a medium (a culture medium with fetal bovine serum) that includes many contaminating particles. When labeling the WBCs, it is difficult to differentiate between the actual WBCs and the many contamination particles. Thus, the WBC labels will have an increased amount of human error, leading to a system poorly trained in identifying WBCs. This issue transfers to later experiments since 1) the CNN is poorly trained to identify WBCs and 2) the the WBC medium introduces these contamination particles in all future experiments. Figure 4-13 shows an AutoScope image of the WBC reference solution with just WBCs. In the zoomed in image, we can observe WBCs surrounded by contamination particles labeled as others. In this image, the 'other' particles seem more clear, but some are ambiguous to the naked eye.

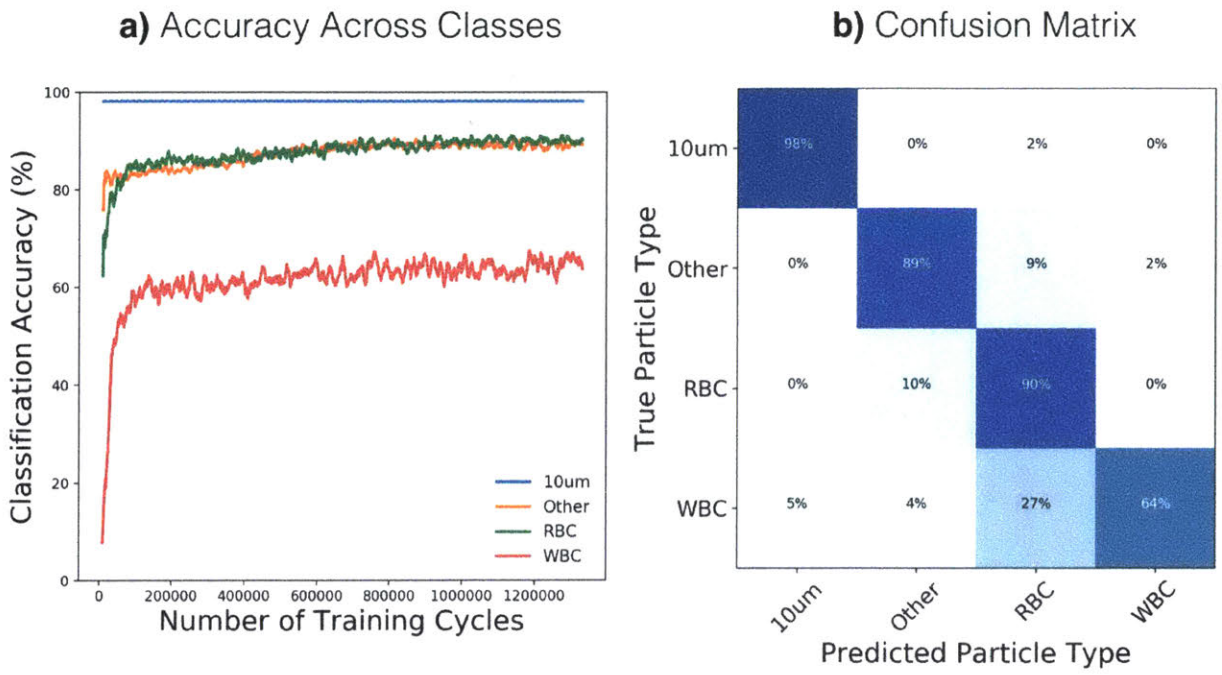



*Figure 4-12. (a) The accuracy for the different particles classes for the AutoScope classification CNN. (b) The confusion matrix, showing the average results across classes for the final few validation batches.*

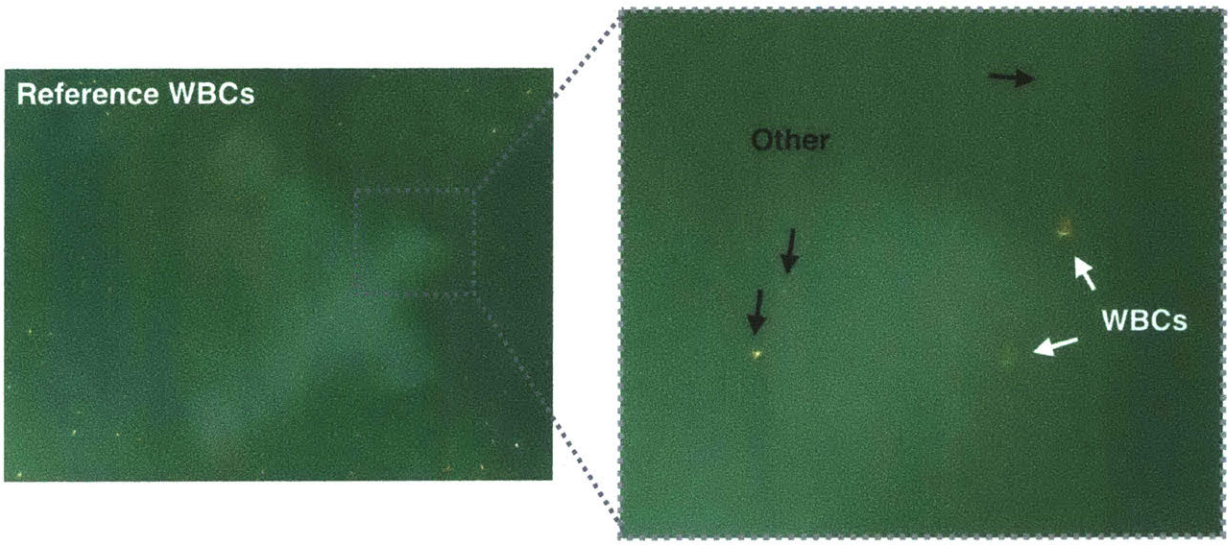



*Figure 4-13. The WBC reference solution contains contamination particles from the medium that the WBCs are cultured in. This decreases the labeling accuracy and thus the networks predictive power.*

# 5 Particle Segmentation

Image segmentation is the process of assigning each pixel in an image a specific class. Within image processing, segmentation is important since it allows us to localize objects of interest within a larger image. In this work, we implemented and evaluated two separate approaches to image segmentation. First, we implemented a standard approach, which includes a combination of thresholding, morphological algorithms, and component analysis. Next, we implemented semantic segmentation, a novel technique enabled by CNNs. In this chapter, we will start by providing background on the Fully-Convolutional Network architecture, which we leverage in our semantic segmentation algorithm. In the rest of the chapter, we will briefly discuss both approaches, and then demonstrate the superior results of semantic segmentation.

## 5.1 Background: The Fully-Convolutional Network

CNNs have demonstrated significant improvements in image classification tasks within the past decade[32–34]. For image classification tasks, CNNs take fixed-sized images as inputs, and use nonlinear regression to predict a class for each image. This process transforms highly-dense, spatial features into a single score for each class. In contrast, for semantic segmentation, CNNs are used to transform highly-dense, spatial features into highly-dense, spatial scores, providing a score vector for every input pixel. Thus, the output of a semantic segmentation CNN is most often an image with the width and height of the input image, and a depth equal to the number of classes.

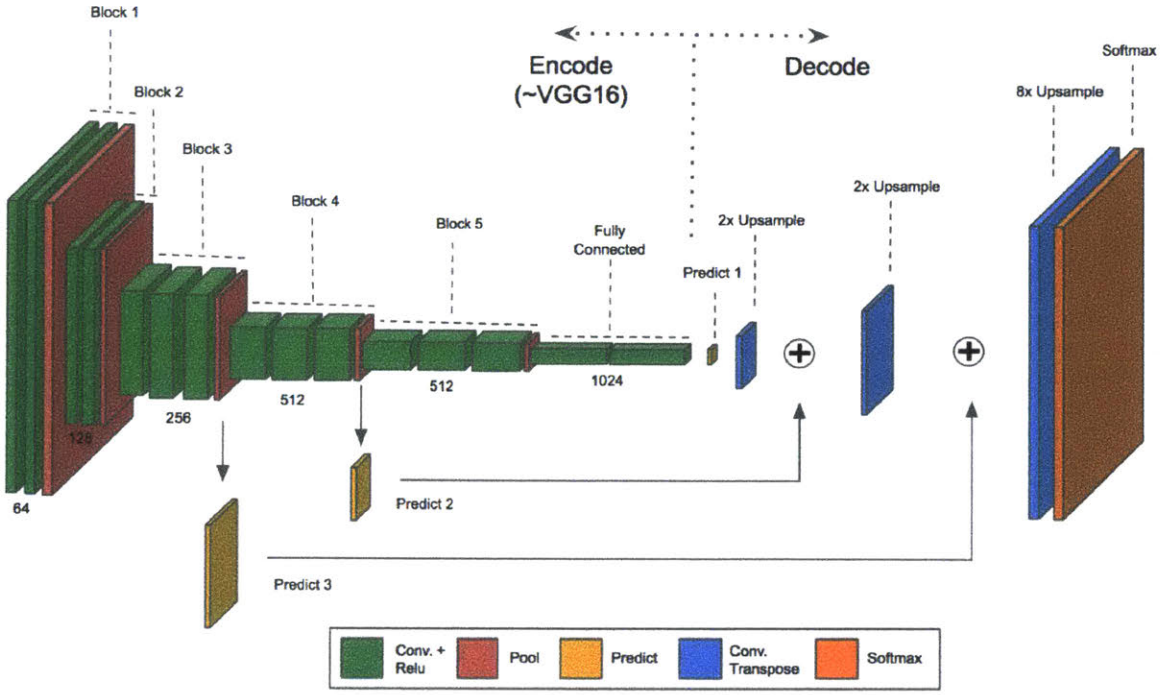



*Figure 5-1. A diagram of the architecture of the fully connected network used for semantically segmenting the background of the AutoScope images from the particles (the foreground). Visualization inspired by diagram in Tai et al*[35].

In recent years, many CNN architectures have been proposed to implement semantic segmentation, including U-Net, SegNet, DeepLab, etc[36–38]. In this work, we decided to use the Fully-Convolutional Network (FCN), because it provides high-accuracy for our application with relatively low architectural complexity[39]. The fully-convolutional network used in this worked is diagramed in Figure 5-1. Broadly, the FCN works through convolutional encoding layers that extract coarser features from the input image followed by convolutional decoding layers that upsample these decoded features to per-pixel class predictions. Specifically, there are four important techniques that help describe the FCN architecture, and were popularized by the original FCN paper. These techniques describe the network presented in Figure 5-1.

### 5.1.1 Transfer Learning

First, the FCN paper uses transfer learning, a technique of transferring the weights of the convolutional layers from pre-trained networks. Although the pre-trained networks learned from

other classification problems, it turns out that the feature-extracting filters are highly robust across different applications. In this work, we use the the pre-trained weights from the VGG16 network submitted to the ImageNet classification competition[40]. Since we use a network that has been pretrained on RGB images, we will use RGB AutoScoe images.

### 5.1.2 Converting Fully-Connected Layers into Convolutions

Second, the fully-connected layers are re-interpreted as convolutional layers. A fully-connected layer is mathematically equivalent to a convolutional layer with receptive fields of the same size as the input region. The major benefit of implementing fully-connected layers as convolutional layers is that the entire network can be represented as a convolution (which is where the 'fully-convolutional network' name stems from). Instead of the CNN only working with fixed-dimension inputs, the CNN can now be convolved across images of any size. For instance, the FCN can be trained with augmented crops, and then provide predictions on the entire image. In fact, the predictions on the entire image is more efficient since calculations can be reused during forward/backpropagation.

### 5.1.3 Learned Interpolations

Third, after the encoding layers, the extracted features must be decoded to create the final prediction image. Where the encoding layers downsampled the image with a max-pool layer, the encoding layers upsample the image with a transposed or strided convolution. Transposed convolutions invert the relationship between the dimensions of the input/output as compared to a direct convolution. Transpose convolutions are implemented by representing the direct convolution in terms of a single matrix multiplication, and then taking the transpose of this matrix. Another way to visualize a transpose convolution is through an upsampling scheme where the input is interleaved with zero values, and then convolved with a filter. The benefit of transpose convolution is that we uspample the input image with weights learned through gradient descent. Thus, the interpolation during upsampling can be learned.

### 5.1.4 Skip Connections

Fourth, and finally, skip connections are introduced that connect the decoding/upsampling layers with their corresponding encoding/downsampling layers. The

encoding layers reduce the dimensionality of the data by extracting features. However, in the decoding phase, it would be beneficial to upsample by not only using the coarser features but also using the more dense features from the encoding layers. This is accomplished through skip connections that pass the features from the encoding layers to the decoding layers for more spatially precise classifications.

## 5.2 The Standard Segmentation Algorithm

Within this work, the goal of segmentation is to localize all the particles within the microscope image. Specifically, segmentation needs to provide the coordinates of all particles and/or clumps of particles (this includes RBCs, WBCs, microbeads and other particles). Using these coordinates, the microscope image is cropped into many smaller images, which are then fed into the CNN classification algorithm discussed in Chapter 4.

We will first introduce the standard segmentation algorithm. In the standard algorithm, we use common image segmentation techniques to find particle coordinates. First, we apply an adaptive thresholding algorithm to transform the greyscale image to a binary image. Each pixel is labeled as either a 0 (background) or 1 (foreground). Adaptive thresholding functions determine a new threshold for each pixel based on the values of the surrounding pixel. By doing this, adaptive thresholding takes into account localized variations, for example, in the illumination of the image. Second, a connected component analysis is used in order to isolate each group of connected foreground pixels. For each connected component identified, we remove components with a size below 4 foreground pixels. This component removal technique filters out stray foreground pixels, and we assume the remaining foreground pixels map to actual particles. Then, in order to combine clumps of particles into a single coordinate, we use the close operator followed by the dilation operator. Finally, we identify the centroid of each remaining connected component, and use this to produce a crop to be classified.

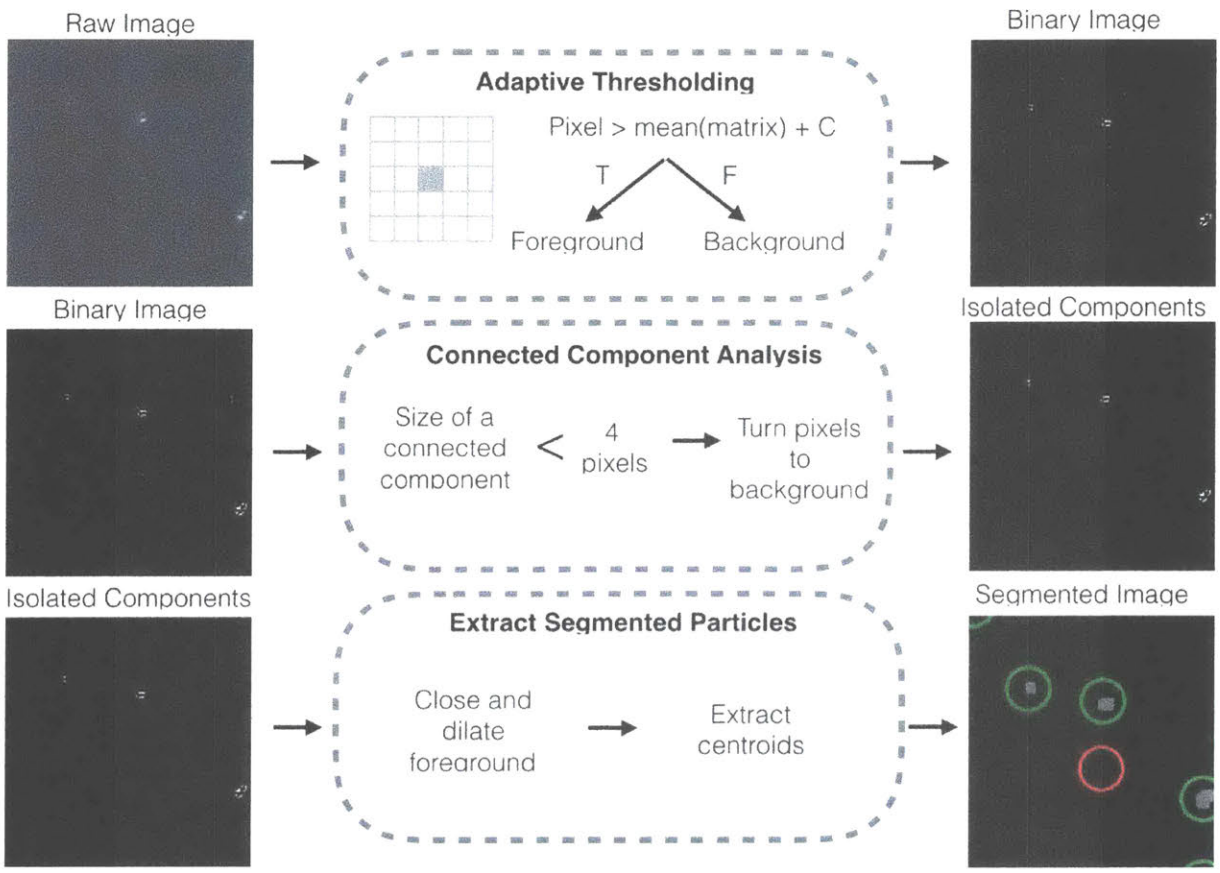



*Figure 5-2. The standard segmentation algorithm uses thresholding, connected component analysis and morphological operations in order to predict the centroid of each particle.*

## 5.3 The Semantic Segmentation Algorithm

For the semantic segmentation algorithm, we use a FCN to threshold the image followed by simple image processing techniques to identify particle coordinates. The FCN is trained to produce an image with each pixel segmented into either the foreground or background. The broad FCN architecture was described earlier in this chapter and the specific training hyperparameters are given in Table 7. Functionally, this operation produces the same type of outputs as adaptive thresholding. However, the quality of the results is superior with the FCN approach as compared to standard thresholding. Once we have an accurate binary image, we use component analysis to identify each particle or clump of particles in the thresholded image. These coordinates are used to produce crops, which are then classified into particle types by the AutoScope classification CNN.

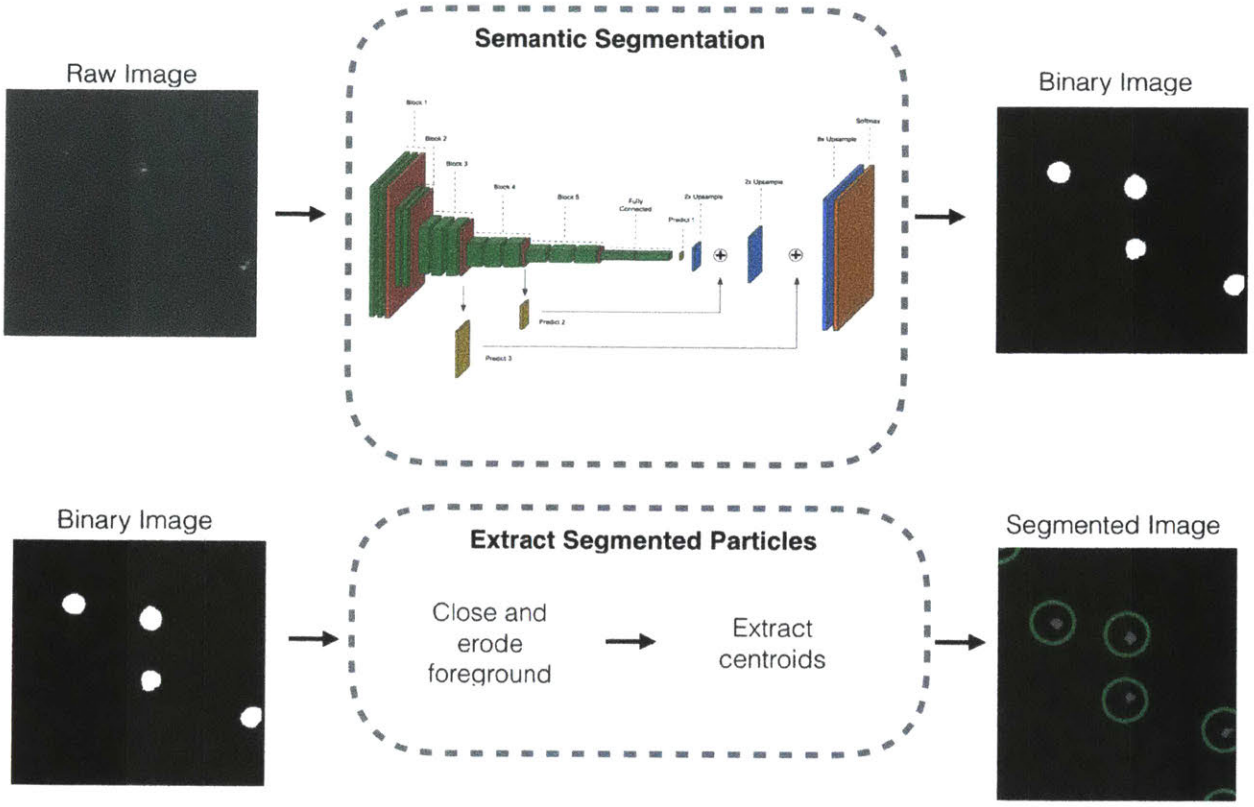



*Figure 5-3. The semantic segmentation algorithm used a trained FCN network to threshold the image, and then extracts the particle centroids from the binary image.*

*Table 7. Characteristics of the FCN architecture used for the AutoScope semantic segmentation.*

| Parameters | Values |
|---|---|
| Encoding Convolutional Blocks | 5 |
| Fully-Connected Layers | 2 |
| Decoding Convolutional Blocks | 3 |
| Skip Layers | 2 |
| Batch Size | 16 |
| Type of Optimizer | Adadelta Optimizer |
| Color | RGB |
| Image Dimensions | 480x480 pixels |
| Total Parameters (weights + biases) | 41,460,166 (~158MB) |

### 5.3.1 Data Preparation

The training data for the FCN was generated from the same process/data used to create the training data for the classification network in Chapter 4. Essentially, the particle centroids were manually selected from images that included a single target particle. Based on these coordinates, we made the ground truth image by placing a circle of foreground pixels at each of these labeled coordinates. The result is a mask with a white foreground circle at each of the particle locations. The foreground circles have a radius of 20 pixels, which is independent of and larger than the size of most underlying particles. The main reasoning behind a large radius is to increase the loss penalty for missed particles (since we want to eliminate false negatives) and still reward the network even if the predicted centroid isn't exactly aligned with the ground truth centroid. We split all the labeled images into a training set (90%) and a validation set (at least 10%).

As in previous CNN-based algorithms, we pre-process the images prior to feeding them through the network. To normalize each image, we apply zero-mean subtraction on each color channel, using the same values used by the original pre-trained VGG16 network. Next, we need to resize the images from the original full-scale AutoScope images (3280x2464 px) to more manageable dimensions (like 480x480px). The reason for this is that the GPU has a limited amount of working memory, and we want to fit an entire training session onto a single GPU to optimize our training efficiency. Furthermore, we want to reduce the dimensions of the full-scale image without reducing the resolution of the image. We accomplish this by taking 480x480 pixel crops from the full-scale image at random locations. Through this process, we generate a batch of training images with a size of 480x480px at the resolution of the AutoScope imaging system. Furthermore, there was no need for additional augmentation since the random cropping of the full-scale image provided sufficient training data augmentation. As a final note, even though we use crops during training, we still can use the full-scale AutoScope images when we later apply the FCN network after training.

### 5.3.2 Training the Network

We trained the FCN on an AWS EC2 instance with a single GPU. During training, we calculated the accuracy through the following equation.

$$Per\ Pixel\ Accuracy = 100 * \frac{Accurately\ Labeled\ Pixels}{Total\ Pixels}$$

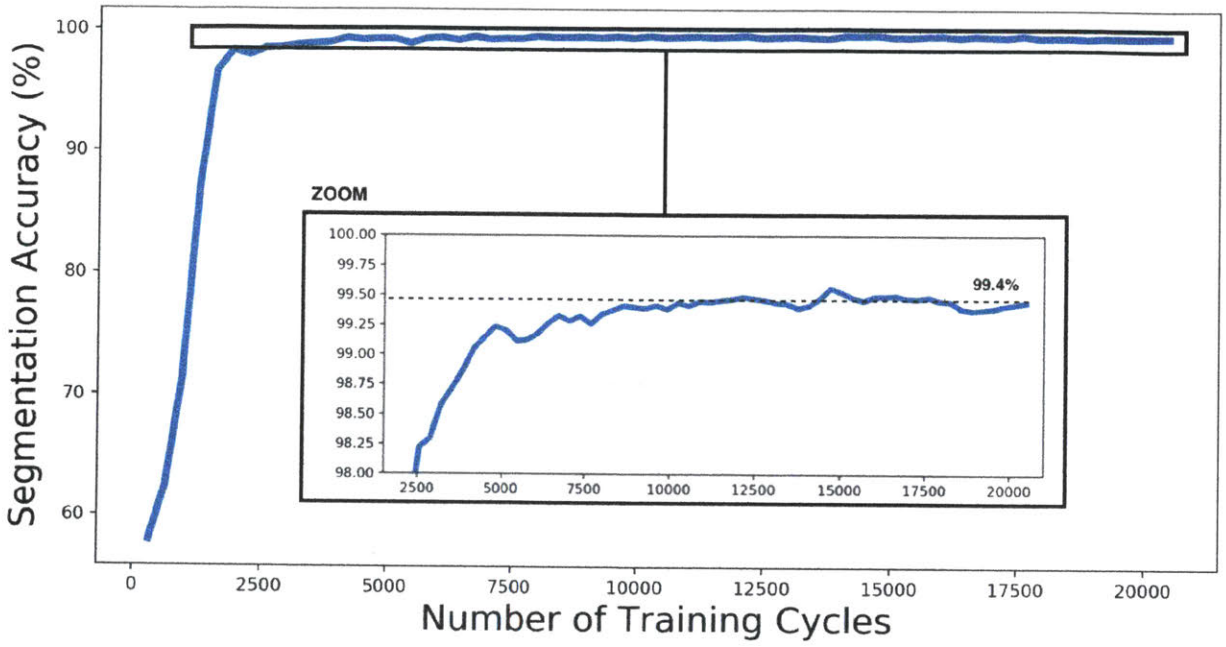



*Figure 5-4. The per pixel training accuracy of the FCN network during training.*

It's important to note that the per pixel segmentation accuracy used for training is not the same as the per particle accuracy that we will later use to characterize algorithm performance. Figure 5-4 shows the per pixel accuracy across training cycles. Although the system seems to plateau rapidly, it's critical to allow the FCN to train to completion. The performance gains in accurately identifying particles are significant as we increase the per pixel accuracy from 98% to +99%. As can be seen in Figure 5-5, this +1% increase in per-pixel accuracy changes the predicted output from foreground blobs interspersed with background pixels to filled foreground circles.

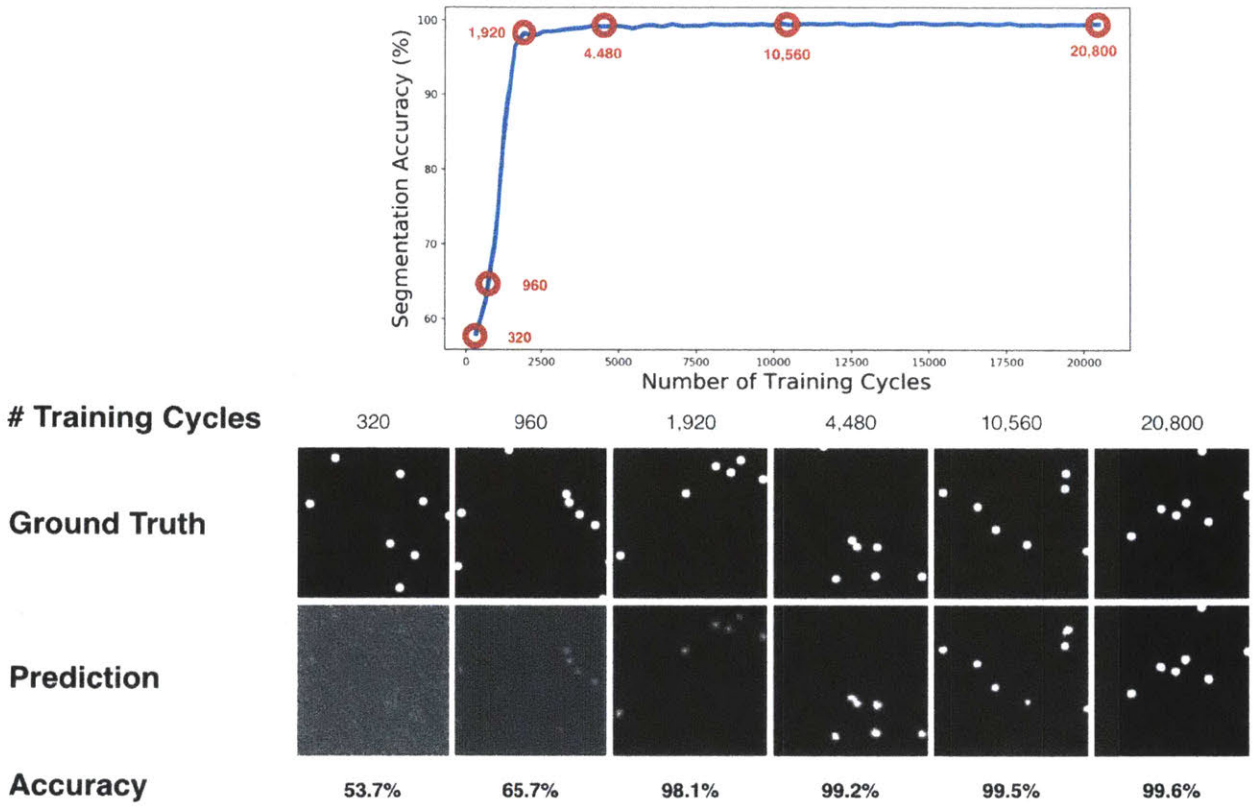



*Figure 5-5. Comparison of the ground truth with representative images predicted by the network at different stages of the training. Each red circle on the accuracy graph maps to a set of ground truth and prediction images shown below it.*

## 5.4 Results: Comparing Semantic to Standard Segmentation

In this section, we will compare the per particle accuracy of the standard segmentation approach to the per particle accuracy of the semantic segmentation approach. We consider a particle to be accurately detected when the centroid of the predicted particles is within 30 pixels of the centroid of the ground truth particles. When quantifying the performance of the segmentation algorithms, we care about three types of predictions: true positives, false negatives and false positives. As can be seen in Figure 5-6, true positives (labeled in green) occur when the algorithm accurately predicts the coordinates of a particle. False negatives (labeled in red) occur when a particle exists in the ground truth image but the algorithm fails to predict its location. Finally, false positives (labeled in blue) occur when the algorithm predicts a centroid which doesn't exist in the ground truth image. Since false positive predictions can still be accurately

classified as 'other' particles during classification, false positives do not affect the segmentation accuracy. On the other hand, it is critical to avoid false negatives since these particles will never even be fed into the classification algorithm. Thus, the per particle accuracy is determined through the following equation.

$$Per\ Particle\ Accuracy = 100 * \frac{True\ Positives}{Total\ Ground\ Truth\ Particles}$$

*Table 8. The per particle accuracy across the different particles using the standard segmentation algorithm compared to the semantic segmentation algorithm.*

| Particles | Standard Segmentation | Semantic Segmentation |
|---|---|---|
| Microbeads | 94.7% | 98.5% |
| Red Blood Cells | 71.3% | 95.3% |
| White Blood Cells | 77.2% | 99.1% |

The results across both algorithms are summarized in Table 8. Each validation image includes only a single target particle and the artifacts that can be found in any solution (the 'other' category). As a reminder, 'other' particles are still predicted and counted during segmentation since the segmentation algorithm is completely particle agnostic. The classification algorithm is better equipped to sort out the 'other' particles as compared to the segmentation algorithms. We observe an increased per particle accuracy across all particle types, with RBCs and WBCs have the largest increases of over 20%. This improvement is significant since a 20% reduction in segmentation accuracy maps to a 20% reduction in cell count accuracies for a system with perfect classification.

Figure 5-6 shows the true positives, false negatives and false positives for an AutoScope image of red blood cells. With the standard approach, there is a significant increase in the false negative particles at the center of the image. This is due to the fact that the particles have much lower contrasts at the center of the image due to the unique illumination pattern at this location. On the other hand, the semantic approach successfully detects most of these low contrast particles.

In conclusion, the semantic segmentation algorithm achieves significant improvements in accurately identifying particles from the AutoScope images.

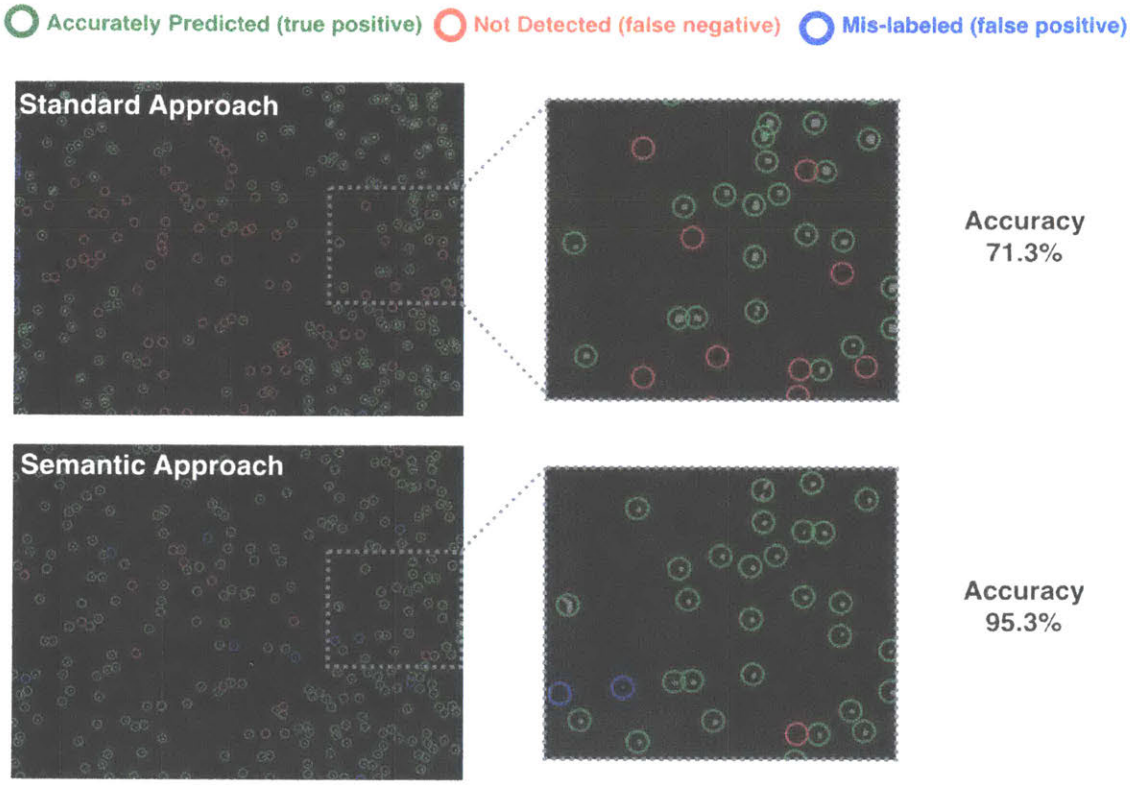



*Figure 5-6. A comparison between the performance of the standard segmentation and semantic segmentation approach. The figure shows true positives, false negatives and false positives for particles on a full-scale AutoScope image taken from a red blood cell solution.*

# 6 End-to-End AutoScope Validation

The goal is to develop an end-to-end, automated microscopic urinalysis system. In Chapter 3, 4 and 5, we presented the technical building blocks of the end-to-end system, including image acquisition (Chapter 3), particle classification (Chapter 4) and particle segmentation (Chapter 5). In this chapter, we will present the experimental results from validating the AutoScope system. First, we will estimate the sensitivity and specificity of the AutoScope system in terms of accurately categorizing urine solutions to have normal/abnormal amount of red blood cells and white blood cells. Second, we will compare the accuracy of the AutoScope's estimated particle counts to known particle counts.

## 6.1 Background

In the background section, we explain the standard reporting techniques used by medical laboratories to summarize urinalysis results. Specifically, we defined a high-powered field and urinalysis reference ranges.

### 6.1.1 High-Powered Field (HPF)

The results from a urinalysis test are usually reported as particles per high-powered field (HPF) or particles per low-powered field (LPF). Since red blood cells (RBCs) and white blood cells (WBCs) are relatively small particles, these are reported as number particles/HPF. A HPF is the field of view observed from a microscope eyepiece with ~400x magnification (usually achieved with a 40x objective followed by a 10x eyepiece). To report final results, technicians average the number of particles they observe across 10 different HPFs on the same slide with the same urine sample.

In order to compare AutoScope results to laboratory results, the AutoScope also summarizes its results in terms of particles/HPF. As can be seen in Table 9, the field-of-view of the AutoScope is 51.7 as large as that of a normal high-powered microscope (at 400x magnification). Averaging results across multiple HPFs increases the certainty of the estimate of the true particle counts. So, by averaging HPFs, we decrease the sample standard deviation in proportion to $\frac{1}{\sqrt{N-1}}$ where N is 10 for medical lab results and 51.7 for the AutoScope.

*Table 9. The area of different fields of view across different microscopes at a specific magnification. The area is given in the actual field-of-view observed on the sample.*

| | Microscope (LFP) | Microscope (HPF) | AutoScope |
|---|---|---|---|
| Field of View | 3.14 mm^2 | 0.20 mm^2 | 10.16 mm^2 |
| Magnification | 100x | 400x | 1x |

### 6.1.2 Reference Ranges for Urine Testing

Normal reference ranges are the range of values for a diagnostic test that 95% of the normal population falls within. Each clinically relevant particle in urine has its own reference ranges. For WBCs, the normal reference range is consistently between 0-5 particles across multiple papers and medical laboratories. However, the reference range for RBCs vary across different sources, with an upper normal range from 2 RBCs/HPF (for Quest Diagnostics) to 5 RBCs/HPF. The most consistent normal reference range is 0-3 RBCs/HPF, which is what we will use in our work (shown in Table 10)[5,41–43].

*Table 10. Clinically relevant thresholds for the number of red blood cells and white bloods cells that can be found in a HPF (on average) during urinalysis.*

| | Normal Range | Abnormal Range |
|---|---|---|
| Red Blood Cells | 0-3 per HPF | >3 per HPF |
| White Blood Cells | 0-5 per HPF | >5 per HPF |

## 6.2 Estimating the Sensitivity and Specificity of the AutoScope

The sensitivity and specificity is a metric commonly used to quantify the performance of a medical test. In order to compare the clinically relevant performance of the AutoScope against the performance of other urinalysis systems, we need to determine the sensitivity and specificity of the AutoScope.

The sensitivity and specificity measure the ability of a test to accurately separate inputs into either a positive output or a negative output. To be precise, sensitivity measures the proportion of positives that are correctly identified as positives, and can be calculated by the following equation:

$$Sensitivity = \frac{True\ Positives}{True\ Positives + False\ Negatives}$$

Specificity measures the proportion of negatives that are correctly identified as negatives, and can be calculated by the following equation:

$$Specificity = \frac{True\ Negatives}{True\ Negatives + False\ Positives}$$

For urinalysis tests, a positive result for a urine particle occurs when the particle count in a sample of urine is in the abnormal range. A negative results occurs when the particle count is in the normal range.

In this section, we will present an experiment that estimates the AutoScope's sensitivity and specificity in determining if a urine sample's particle counts are in a normal or abnormal range (see Table 10). We will accomplish this by generating digital urine that has the same distribution of particles as the samples of urine sent to medical laboratories for testing. In the rest of the chapter, we will describe the experimental protocol and the results from the experiment.

### 6.2.1 Experimental Protocol

Within the experimental protocol, we will first present the process of obtaining a probability density function (PDF) of the number of particles found in urine. Next, we will discuss the process of using these PDFs to create hundreds of samples of digital urine that can be used to estimate the sensitivity and specificity of the AutoScope.

#### *6.2.1.1 Estimating the Distributions of Particles in Urine*

In this section, we will estimate the PDF for the number of particles observed in urine at medical laboratories. The probability density function maps a particle count (represented as a continuous variable) to the relative likelihood that this particle count is found in a sample of urine sent to a medical lab. To estimate these PDFs, we used data from an experiment that sampled 209 urine specimens at a medical laboratory. The experiment used these 209 samples to determine the specificity and sensitivity of semi-automated urinalysis systems (like the Iris iQ-200)[5]. Thus, by accurately modeling the particle counts in these 209 samples, we can can estimate the sensitivity and specificity of the AutoScope. Table 11 summarizes the number of

samples which had particle counts within the provided ranges of RBCs, WBCs and epithelial cells. These counts were determined by manual microscopy.

*Table 11. The number of urine samples that have particle counts within the given ranges. These counts were taken from 209 urine samples evaluated through manual microscopy at a medical laboratory.*

| Particle Counts (particles/HPF) | 0-5 | 6-10 | 11-20 | >20 |
|---|---|---|---|---|
| RBCs | 180 | 7 | 8 | 14 |
| WBCs | 115 | 42 | 22 | 30 |
| Epithelial Cells | 155 | 36 | 12 | 6 |

It took a few steps to convert the data in Table 11 into an estimated PDF. We first converted the data in Table 11 to cumulative probabilities. Then, we used a two exponential model to estimate the cumulative distribution function (CDF) of obtaining a certain particle, represented by the following equation:

$$prob(sample\ count \leq x) = a_1 e^{b_1 * x} + a_2 e^{b_2 * x} + 1$$

where x is an arbitrary particle count and $a_1, b_1, a_2, b_2$ are parameters of the model. At a very large x, the model asymptotes to a probability of 1.0, as indicated by the +1 at the end of the model. The reason we built the models around the CDF instead of the PDF is that the data provided in Table 11 maps a range of particle counts to a cumulative sample count for that range. When data is provided in this format, it's more accurate to build models against the CDFs. The model parameters were determined by using the Levenberg–Marquardt algorithm.

Once we obtained the CDFs, we determined the PDFs by taking the first order derivative of the CDF with respect to x. To validate the fit of the resulting PDFs, we compared the integral of the PDFs between the particle count ranges given in Table 11 to the sample numbers also given in Table 11. These matched closely, deviating by consistently less than 3%. The estimated CDFs and PDFs for the counts of each particle are given in Figure 6-1. The histogram in the figure represents the results from selecting particle counts from the PDFs 10,000 times, and aligns with the theoretical PDF models.

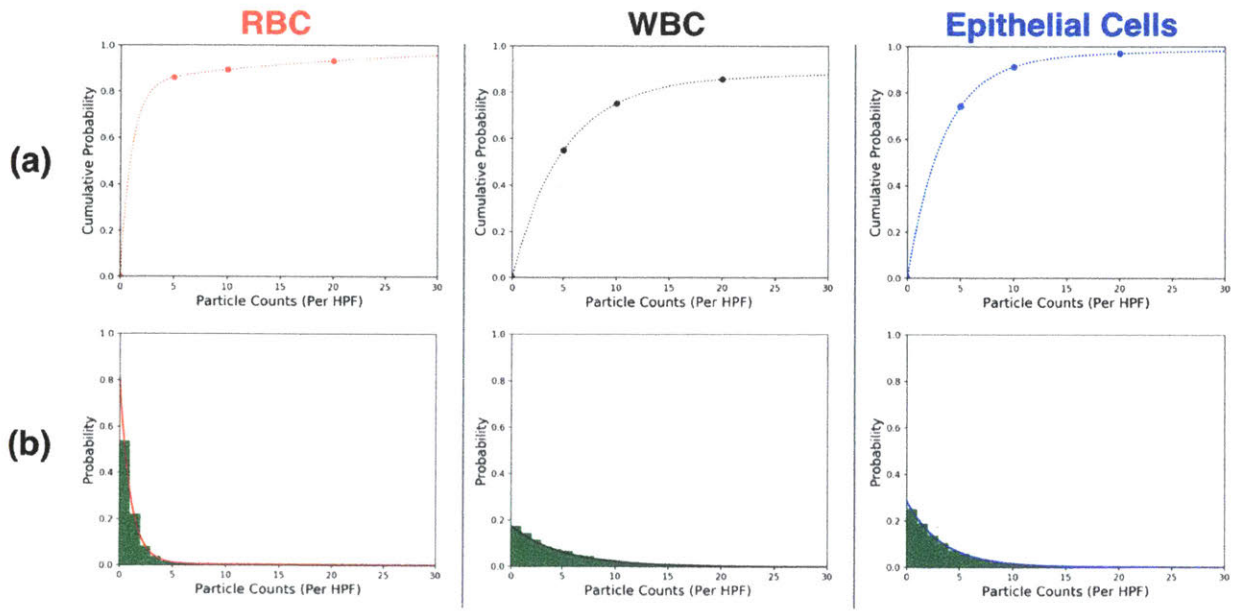



*Figure 6-1. (a) The models of the cumulative distribution functions based on 209 urine samples sent to a medical lab for analysis. All CDFs approach 1.0 at large particle counts, even the white blood cell's CDF. (b) The estimated probability density functions. The line graph represents the derived model. The histogram represents the results from sampling from the probability density functions 10,000 times.*

### *6.2.1.2 Generating Digital Urine Solutions*

We used the PDFs of particles to generate 209 digital urine samples. Each digital urine solution consisted of RBCs, WBCs, microbeads and 'other' particles. To determine a particle count/HPF for a digital urine solution, we sampled the PDF. For the RBC and WBC counts we used the RBC and WBC PDFs. For the microbeads, we used the PDF of the epithelial cells. For the 'other' particles, we also used the PDF of the epithelial cells, but we increased the resulting count drawn from the PDF by 1. So, if we drew a count of 3.1 'other' particles per HPF from the PDF, we used 4.1 'other' particles per HPF for the sample of digital urine. The main reason for this is to ensure that no digital urine solution has zero particles in it, which would bias our results because the algorithm would always achieve 100% accuracy on these solutions (with no particles to classify, there can be no classification errors).

The digital urine samples were generated directly from images taken with the AutoScope. When generating a sample of digital urine, we used three times the particle count drawn from the PDF. The PDF outputs counts in terms of particles per HPF. During manual microscopy in labs, results are averaged across 10 HPFs. The AutoScope averages across 51.7 HPFs. For the purpose of this experiment, we decided to use 3 HPFs due to a limited availability of unique images of particles. We had access to a limited number of particles because (1) we could only use labeled particles and (2) we could use only particles that the neural network had never trained on. With access to a larger labeled dataset, we could average across more HPFs. This would likely improve the AutoScope's results by reducing the variance in the AutoScope's measurements.

After creating the 209 digital urine samples, we compared the distribution of particles in all of these digital urine samples to the expected PDFs. The results of this comparison are shown in Figure 6-2, with the line graph being the theoretical PDF and the histogram being the observed distribution. As expected, the theoretical and observed distributions aligned.

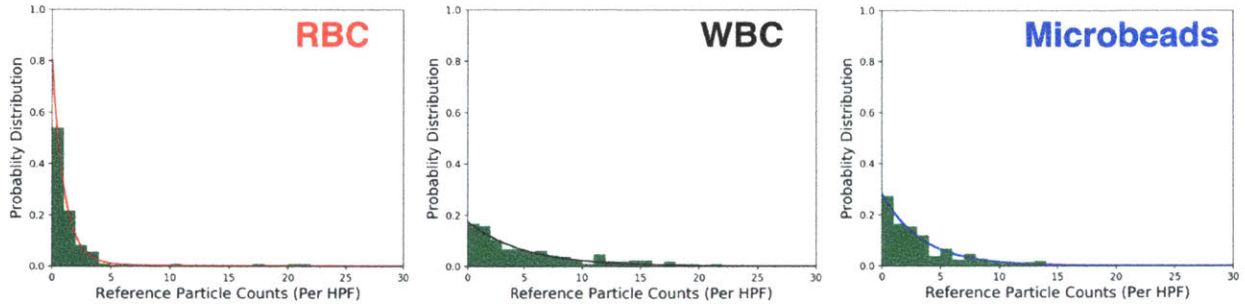



*Figure 6-2. The graphs compare the theoretical PDFs of particle counts in urine (the line graph) to the observed distribution of particle counts in the digital urine (the histogram).*

### 6.2.2 Results and Discussion

In this section, we will present the results from analyzing the 209 digital urine samples with the AutoScope algorithms. Specifically, we will determine the sensitivity, specificity and R-squared value for each particle, and compare them to the results for the iQ-200.

For each digital urine sample, we used the AutoScope algorithms to estimate the particles per HPF. The results across the 209 digital samples are shown in Figure 6-3. Across all target particles, we got a linear best fit line of $y=0.8245x+0.6597$ with R-squared value of 0.933. The

particle counts are clustered between 0-5, as is expected based on the PDFs of the particle counts.

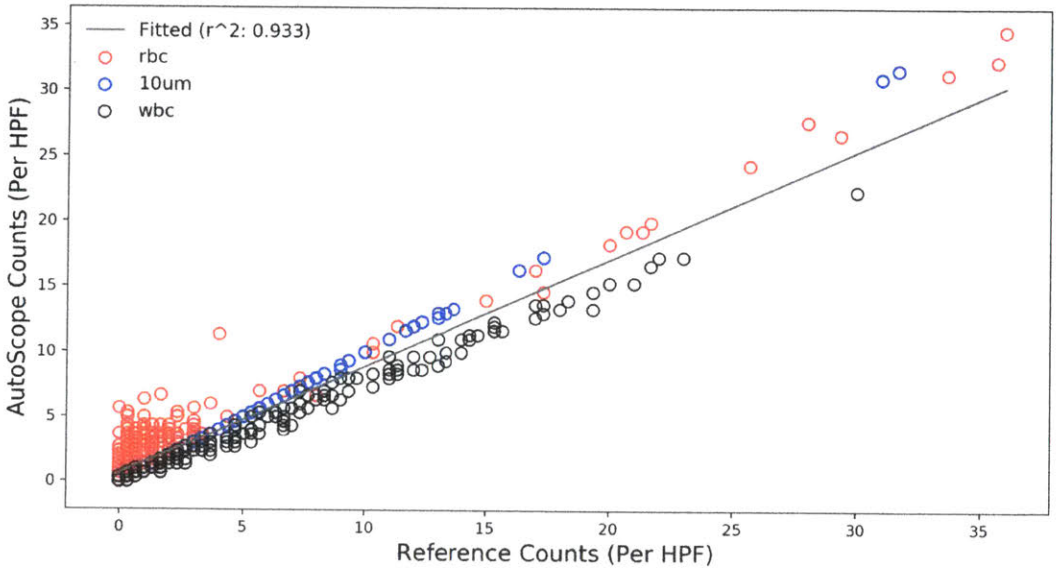



*Figure 6-3. The AutoScope counts of RBCs, WBCs and microbeads compared to the reference counts for the 209 samples of digital urine.*

In order to determine the sensitivity and specificity, we used the normal ranges provided in Table 12, which are based on those used at medical laboratories as well as in other studies (shown in Table 10). The thresholds that the AutoScope used to determine normal/abnormal particle counts were titrated in order to increase AutoScope's sensitivity/specificity (see Table 12 for thresholds). Selecting optimal thresholds is common practice for diagnostic systems, and is accomplished through a receiver operating characteristic curve (ROC curve). The final sensitivity, specificity and R-squared values are also presented in Table 12. RBCs had the lowest sensitivity and specificity, which is due to the distribution of RBC counts compared to WBC, microbead and 'other' particle counts. Specifically, we expect lower RBCs than any other particles. Thus, when we misclassify just a few of the more abundant WBCs, microbeads and 'other' particles as an RBC, we are more likely to incorrectly exceed the RBC abnormal threshold.

*Table 12. The validation results for RBCs, WBCs and microbeads obtained from using the AutoScope algorithm to label 209 samples of digital as having normal or abnormal particle counts.*

| Particle | Normal Range | AutoScope Thresholds | Sensitivity | Specificity | Overall Accuracy | R-Squared |
|---|---|---|---|---|---|---|
| RBCs | 0-3/HPF | 4/HPF | 88% | 89% | 89% | 0.947 |
| WBCs | 0-5/HPF | 4/HPF | 91% | 97% | 95% | 0.989 |
| Microbeads | 0-5/HPF | 5/HPF | 100% | 100% | 100% | 1.000 |

Finally, we compare the validation results of the AutoScope with those of the iQ-200. to The iQ-200 is a semi-automated microscopic urinalysis system with the best performance on the market[5]. The iQ-200 retails for between $100,000 to $150,000. A 2005 paper by Wah et al validated the performance of the iQ-200 by comparing the automated particle counts across 166 urine samples to manual microscopic counts[41]. The comparison is shown in Table 13. The AutoScope achieves better results than the iQ-200 across all metrics, except for the specificity of RBCs (which is 4% lower). It's important to remember that the AutoScope validation results do not include any error from inaccurate particle segmentation within this experiment. However, due to the high accuracy of the segmentation algorithm, we expect this to only reduce the sensitivity and specificity by a small amount.

Since we use different types of references for the iQ-200 and the AutoScope, we need to caveat the comparison between the iQ-200 and the AutoScope. The errors in the reference counts for the AutoScope and iQ-200 can decrease the perceived accuracy of the respective system. The iQ-200 reference is determined through manual microscopy. Wah et al reduced the error of their reference by averaging the counts from two separate technicians performed on two samples of the same specimen across many HPFs. When analyzing the manual counts, they obtained an R-squared value greater than 0.941 for both RBCs and WBCs. For the AutoScope, we make digital urine solutions with a known number of pre-labeled particles. If there is any error in the labeling of the particles, that error can be transferred to the AutoScope validation results. With that caveat, we don't think errors in the reference will significantly alter the outcomes.

*Table 13. Comparison between the validation results of the AutoScope and those of the iQ-200*[41].

| Particle | Instrument | Normal Range | Sensitivity | Specificity | R-Squared |
|---|---|---|---|---|---|
| RBCs | AutoScope | 0-3/HPF | 88% | 89% | 0.947 |
| | iQ-200 | 0-3/HPF | 80% | 93% | 0.920 |
| WBCs | AutoScope | 0-5/HPF | 91% | 97% | 0.989 |
| | iQ-200 | 0-5/HPF | 77% | 96% | 0.884 |

## 6.3 Validating the AutoScope with Synthetic Urine Mixtures

The motivation of this experiment is to validate the end-to-end performance of the AutoScope system in detecting RBCs and WBCs in urine. We will accomplish this by producing synthetic urine with different concentrations of urine particles. Then, we will use our automated urinalysis system to measure the concentration of urine particles in these solutions. We will compare our system's results with the reference results, and quantify how well they correlate.

### 6.3.1 Experimental Protocol

In this section, we will describe the experimental protocol. First, we will discuss the process of making synthetic urine samples. Second, we will present how the synthetic urine samples were processed to generate particle counts.

#### *6.3.1.1 Making Synthetic Urine Samples*

To make synthetic urine solutions, we followed four steps (shown in Figure 6-4). Briefly, we 1) generated reference solutions for each particle, 2) determined the concentrations of these reference solutions, 3) mixed together predetermined amounts of the reference solution, and 4) diluted the mixture with buffer.

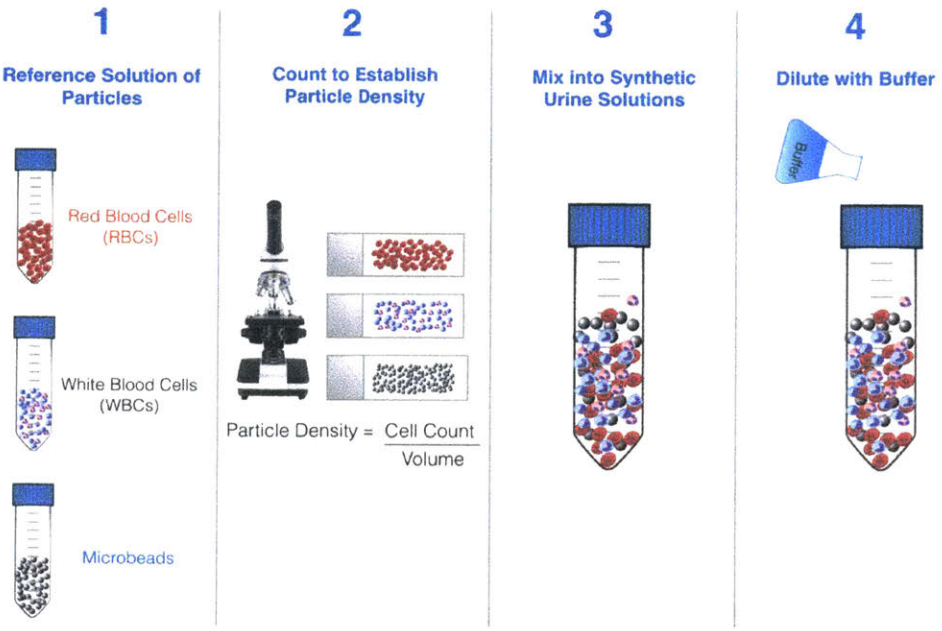



*Figure 6-4. We produced synthetic urine with a mixture of red blood cells, white blood cells, and microbeads using a four-step process. We produced 9 combinations of synthetic urine with known particle concentrations based on 3 reference solutions.*

**Step 1: Generated a single reference solution for each particle type**

We generated a single reference solution for each of the target particles, including RBCs, WBCs and 10um microbeads. Since all synthetic solutions stem from different mixtures and dilutions of these reference solutions, by knowing the particle concentration of the reference solutions, we can calculate the concentrations of the synthetic urine solutions.

**Step 2: Determined the particle concentration of each reference solution**

We determined the particle concentrations of each reference solution by manually counting the particles under a high-powered microscope. To do this, we prepared a slide (using a 22x22mm cover slip) with 20uL of the reference solution. Then, we used a Zeiss microscope to count the number of particles visible in the microscope's field of view, averaging the result across 6-13 different field-of-views. Using this information, we calculated the concentration of the reference solution with:

$$\frac{Particles}{Volume\ (\mu L)} = \left(\frac{Average\ Particle\ Count}{Zeiss\ Field\ of\ View\ (mm^2)}\right)\left(\frac{Area\ of\ Coverslip\ (mm^2)}{Volume\ of\ Solution\ on\ Coverslip\ (\mu L)}\right)$$

where the Zeiss Field of View is $1.05mm^2$. We also calculated the particles per HPF with:

$$\frac{Particles}{HPF\ Unit} = \left(\frac{Average\ Particle\ Count}{Zeiss\ Field\ of\ View\ (mm^2)}\right)\left(\frac{HPF\ Field\ of\ View\ (mm^2)}{HPF\ Unit}\right)$$

where the HPF field of view area is 0.20 $mm^2$.

Table 14 shows the concentrations for each of the reference solutions. Since we were working with living cells and since we ran experiments across different days, we needed to create new reference solutions for each day (indicated by day 1 vs day 2).

*Table 14. The concentration of the reference solutions based on the manual counts obtained from the Zeiss microscope.*

| Reference Solution | Concentration | Particles per HPF |
|---|---|---|
| Microbeads (day 1) | 462.6 particles/uL | 3.75 particles/HPF |
| Red Blood Cells (day 1) | 2642.3 particles/uL | 21.44 particles/HPF |
| White Blood Cells (day 1) | 434.9 particles /uL | 3.53 particles/HPF |
| Microbeads (day 2) | 1728.2 particles/uL | 14.02 particles/HPF |
| Red Blood Cells (day 2) | 3758.6 particles/uL | 30.50 particles/HPF |
| White Blood Cells (day 2) | 167.1 particles/uL | 1.36 particles/HPF |

**Step 3-4: Mix and dilute different proportions of the reference solutions**

We generated 9 synthetic urine solutions by mixing and diluting different proportions of the reference solutions. We accomplished this by first pipetting a predetermined volume of each of the reference solutions into a microtube (Step 3) and then diluting the resulting mixture with PBS (Step 4). Before pipetting any reference, we mixed the reference solution for >60 seconds with a vortex mixer.

In total, we produced 9 synthetic urine solutions, each with a distinct combination of particle concentrations. Out of the 9 synthetic urine solutions, 7 solutions were measured on experimental day one, and 2 solutions were measured on experimental day two.

*Table 15. The reference absolute counts for each solution of synthetic urine. These were the expected particle counts per HPF for each solution of synthetic urine.*

| Sample Solution | Experimental Day | Microbeads/HPF | RBCs/HPF | WBCs/HPF |
|---|---|---|---|---|
| Solution 1 | 1 | 3.75 | 0 | 0 |
| Solution 2 | 1 | 0 | 10.72 | 0 |
| Solution 3 | 1 | 0 | 0 | 3.53 |
| Solution 4 | 1 | 1.88 | 0 | 0.88 |
| Solution 5 | 1 | 0 | 3.57 | 2.35 |
| Solution 6 | 1 | 0 | 7.15 | 1.18 |
| Solution 7 | 1 | 0.63 | 3.57 | 1.18 |
| Solution 8 | 2 | 0 | 8.71 | 0.97 |
| Solution 9 | 2 | 0 | 2.54 | 1.13 |

*Table 16. The reference relative counts for each solution of synthetic urine. These were the expected percentage breakdown of target particles to total particles within each solution of synthetic urine.*

| Sample Solution | Microbeads | RBCs | WBCs |
|---|---|---|---|
| Solution 1 | 100% | 0% | 0% |
| Solution 2 | 0% | 100% | 0% |
| Solution 3 | 0% | 0% | 100% |
| Solution 4 | 68% | 0% | 32% |
| Solution 5 | 0% | 60% | 40% |
| Solution 6 | 0% | 86% | 14% |
| Solution 7 | 12% | 66% | 22% |
| Solution 8 | 0% | 90% | 10% |
| Solution 9 | 0% | 69% | 31% |

*6.3.1.2 Processing Synthetic Urine*

In this section, we will present the process of using the AutoScope to determine particle counts for each synthetic urine solution. To accomplish this, we 1) captured images of the synthetic urine and 2) used the AutoScope's algorithms to automatically calculate particle counts.

**Step 1: Capture images of the synthetic urine with the AutoScope**

After we created the 9 sample solutions, we acquired multiple images of each solution with the AutoScope. To acquire an image of a solution, we first prepared a microscope slide with the synthetic urine. First, we made sure that the synthetic urine solution was thoroughly mixed by using a vortex mixer. Then, we took 20uL of the sample and placed it on a clean glass slide. A 22x22mm coverslip was placed on top of the solution. Finally, we inserted the glass slide into the AutoScope system, and took multiple images of the sample at different, non-overlapping locations on the coverslip. We repeated this process for each sample.

**Step 2: Use the AutoScope's algorithms to automatically calculate particle counts**

Once the images from each synthetic urine solution were acquired, we then passed the images through our automated cell-counting software. The cell-counting software performed automatic particle segmentation, followed by automatic particle classification and then neatly summarized the results for operator use.

In Figure 6-5, we show two example results from different synthetic urine solutions. To be specific, (a) is from Solution 7 and (b) is from Solution 8. Each particle is labeled with a colored circle. Black circles indicate a classification of 'other'. Blue circles indicate microbeads. Red circles indicate RBCs. White circles indicate WBCs. In the grey table at the bottom corner of each image, there is a summary of the AutoScope results compared to the reference results for the corresponding solution.

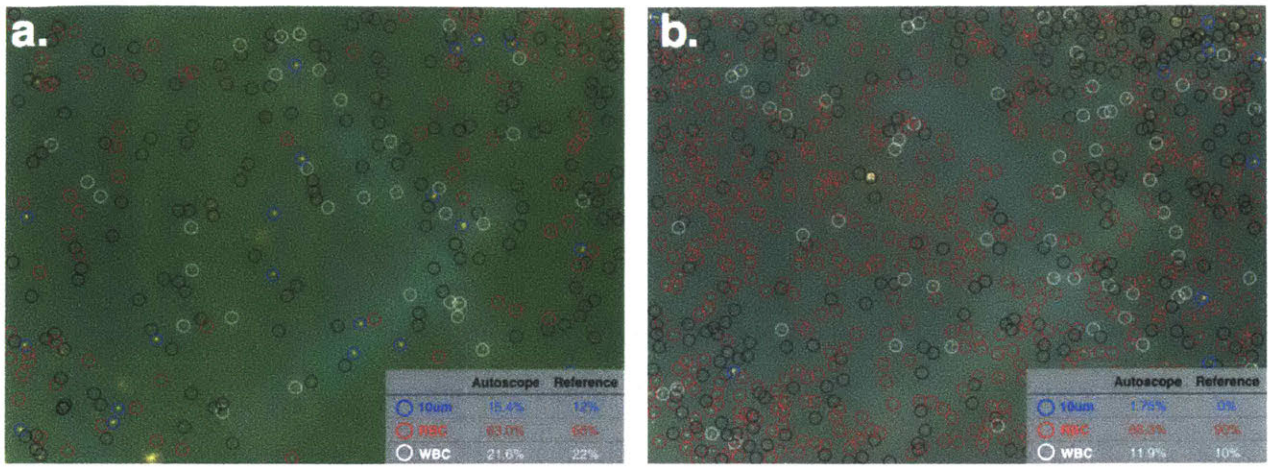



*Figure 6-5. Example images from different solutions. Each particle is classified as a microbead, red blood cell, white blood cell, or other. The table in the bottom right corner of each image provides the AutoScope's relative counts compared to the reference results for that solution. (a) An AutoScope image for solution 7. (b) An AutoScope image from solution 8.*

### 6.3.2 Results and Discussion

In this section, we will present the results of this experiment. First, we will justify using relative particle counts to compare the AutoScope results to the reference results. Second, we will confirm the accuracy of the reference results by comparing them to the results obtained by a medical laboratory on our synthetic urine samples. Finally, we will quantify the accuracy of the end-to-end AutoScope system by looking at the correlation of the AutoScope results with the reference results.

#### *6.3.2.1 Metric Used to Compare Results*

In order to report the measured results for each sample, we considered two separate metrics: absolute particle counts and relative particle counts. The absolute particle count is the average number of particles observed in a high-powered field, measured as a count/HPF. Relative particle count is the proportion of a specific target particle to the total target particles, measured as a percentage. In this section, we will justify using relative particle counts to compare results to the AutoScope.

The ideal metric to validate the AutoScope system is an absolute particle count, since this metric provides a direct indication of the concentration of the synthetic urine. However, there are

a few problems with this metric. The most salient problem is that the particle counts for the same slide significantly vary across different field of views of that slide.

We confirmed this phenomenon across the majority of the slides, including on slides with the reference solution and on slides with the synthetic urine. As an example, Figure 6-6 shows the particle counts per HPF for the reference solutions. As can be seen, we observed significant variation across different fields of views (FoVs), with standard deviations of up to 10.1 particles/HPF.

Multiple factors caused the non-even distribution of the particles across the slide. The particles had the highest concentration at the location where the pipette deposited the synthetic urine. The concentration decreased from this focal point, with additional variation due to bubbles and due to variations in how the coverslip was placed on the slide. Possible options to reduce this particle variation include using a hemocytometer or using a larger volume of fluid on each slide.

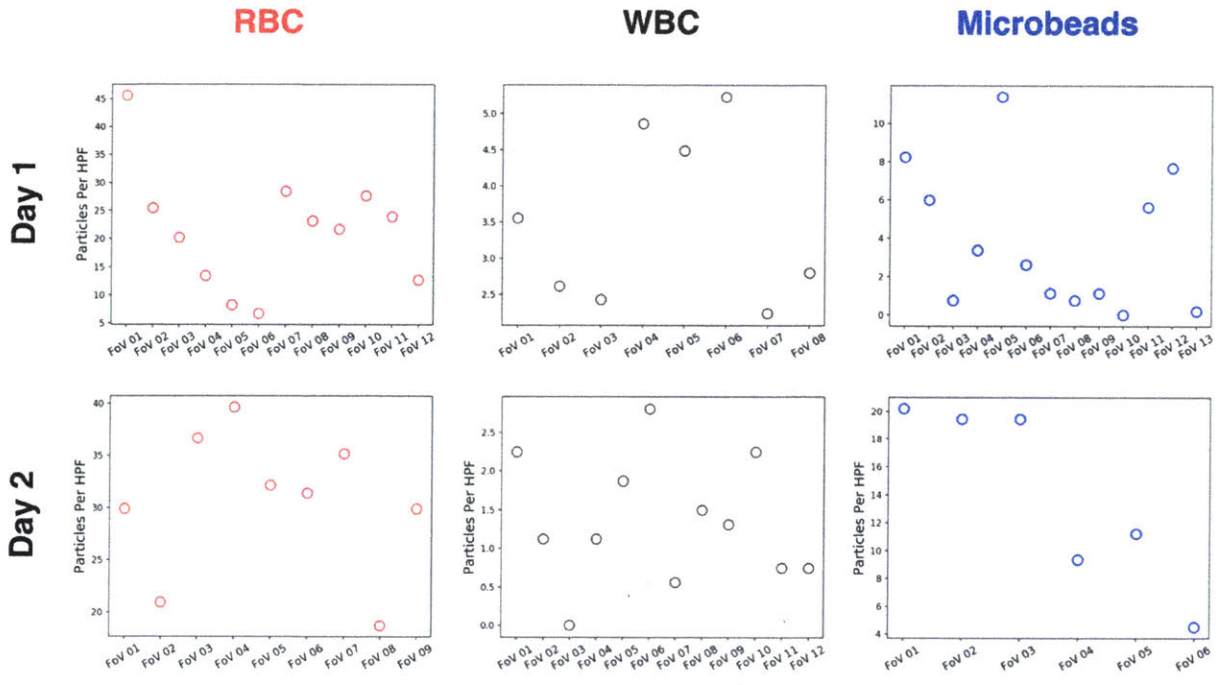



*Figure 6-6. The figure shows the absolute counts per HPF for the reference solutions across different fields of views (FoVs). Note that the y-axis scales differ for each scatter plot. Within each scatter plot, it can bee seen that the absolute counts vary significantly even though they are taken from the same slide.*

Due to the non-uniform distribution of particles on slides, we could not rely on absolute counts as a metric to compare our results. Instead, we used relative particle counts to compare

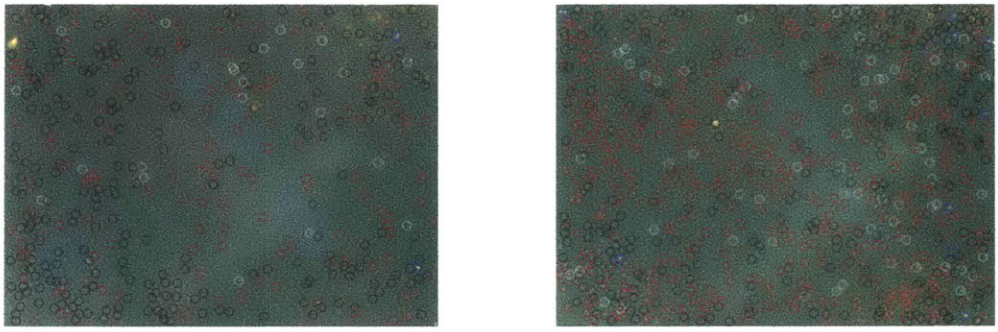
our results. The relative particle counts are given as a percentage, and can be calculated through the following equation:

$$Relative\ Count = 100 * \frac{Target\ Particle\ Count}{Total\ Particle\ Count}$$

The reason that relative particle counts give consistent results is that the particles are evenly mixed across the slide. So, even though the total particles might be different across fields of view, the relative counts will be the same. This is due to the fact the that solution is well mixed and particle diffusion ensures that the relative proportion of particles will be the same through out a solution, on average.

We demonstrate this behavior n Figure 6-7 by comparing absolute particle counts with relative particle counts for the same solution on the same slide. As can be seen, the absolute particle counts vary across different fields of view, as shown in (a) and (b). However, the relative particle counts are consistent across different fields of view.

**a.**




**b.**

| | Absolute | Relative | | Absolute | Relative |
|---|---|---|---|---|---|
| **RBC** | 140 / 168 | 85.4% | **RBC** | 493 / 571 | 86.3% |
| **WBC** | 22 / 168 | 13.4% | **WBC** | 68 / 571 | 11.9% |
| **Microbeads** | 2 / 168 | 1.2% | **Microbeads** | 10 / 571 | 1.8% |

*Figure 6-7. The figure provides an example of the need to use relative particle counts instead of absolute particle counts. Part (a) shows two AutoScope images taken from the same slide with the same synthetic urine solution. Part (b) compares the absolute particle counts with the relative particle counts for the fields of view in part (a). Although the absolute particle counts varied across the two images, the relative particle counts are similar.*

*6.3.2.2 Confirming Reference Results with Medical Lab Results*

In order to validate the accuracy of the AutoScope, we compared the AutoScope results with the reference results (done in the next section - 6.3.2.3). However, since the reference results were determined through a manual process, we first needed a way to confirm the accuracy of the reference results. We accomplished this by sending a few synthetic urine solutions to a medical laboratory for testing. Then, we compared the medical laboratory results with the reference results that we calculated in-house. In this section, we show that the reference results are within the ranges of results provided by the medical laboratory.

We sent three samples (Solution 5, 6 and 8) of the synthetic urine to a clinical urinalysis lab (MIT Medical Laboratory). Without knowledge of the concentrations of the samples, the lab performed a standard microscopic urinalysis test, and reported back RBC/HPF and WBC/HPF for each sample. The medical laboratory results are provided in ranges of particles per HPF (see Table 17). We could not send all synthetic urine solutions to a medical laboratory for testing since some solutions had microbeads, which medical laboratories do not deal with.

*Table 17. The results reported by the medical laboratory for the solutions sent for clinical urinalysis. The medical laboratory did not report microbeads since they are not clinically relevant particles, but we indicate 0 microbeads for each solution for completeness.*

| Sample Solution | Microbeads/HPF | RBCs/HPF | WBCs/HPF |
|---|---|---|---|
| Solution 5 | 0 | 5-10 | 5-10 |
| Solution 6 | 0 | 15-20 | 1-3 |
| Solution 8 | 0 | 10-15 | 1-3 |

We compared the relative counts from the medical laboratory to the relative counts from the reference results for RBCs and WBCs for the 3 solutions (see Figure 6-8). If there is a 1-to-1 relationship between the reference results and the medical laboratory results, then we expect all the data points to fall on the linear identity line. Since the medical laboratory results are given in ranges, we expect the linear identity line to transect these linear ranges. This is the case for all medical laboratory results, including the RBCs (red markers) and WBCs (white markers). Thus, we conclude that the reference results are accurate within the ranges provided by the medical test, and can be used to validate the AutoScope results.

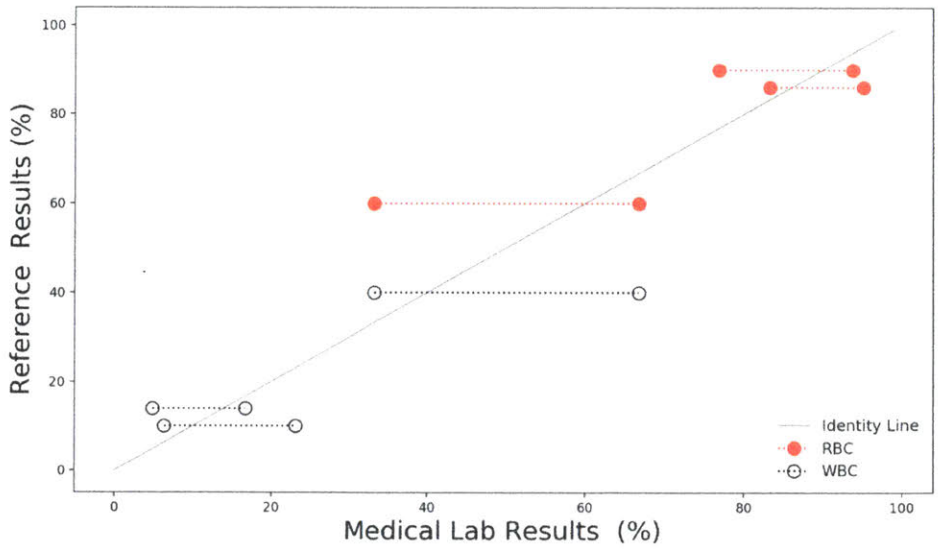



*Figure 6-8. The comparison between relative particle counts of the medical lab results and the reference results. The linear relationship provides evidence that the reference results can be trusted.*

### *6.3.2.3 AutoScope Results*

In this section, we present the end-to-end accuracy of the AutoScope. The accuracy of the AutoScope is determined by quantifying the correlation between the AutoScope and reference results. The R-squared value provides an indication of how well the AutoScope values are predicted by the reference values. The Wald Test allows us to test the hypothesis that the linear relationship between the AutoScope and reference results has a slope of 0 (using a t-distribution for the slope statistic). A slope of 0 indicates that there is no correlation between these variables. So, we use the Wald Test to ensure that there is a correlation. To set expectations, the ideal result occurs when there is a 1-to-1 relationship between the reference and AutoScope relative counts, which would lead to all data-points falling on the identity line of y=x.

First, we compare the average AutoScope results for each solution to the reference (see Figure 6-9). We obtained a final particle count for each solution by averaging across all the images (fields-of-view) taken for that solution (up to 8 different images). Unfortunately, we only had a single image for Solution 3, and thus did not include Solution 3 in Figure 6-9. We still present the results including Solution 3 later in this section.

Across the 8 synthetic urine solutions shown in Figure 6-9, we obtained an R-squared value of 0.98, with a best-fit line of y=0.906x+3.14 and a standard error of 0.027. Using the Wald Test, we rejected the null-hypothesis that the slope is zero (the two-sided p-value was significantly less than 0.01). Thus, we can conclude that there is a clear correlation between the the AutoScope and reference results.

When we include Solution 3, the R-squared value decreases to 0.92. However, these results are significantly biased by the inaccuracy of a single image. Specifically, the results of Solution 3 are fully represented by the inaccuracies of a single image where other Solutions are represented by the average of multiple images. Furthermore, we expect to obtain poor results with Solution 3 since it's composed of 100% WBCs, for which we get the lowest classification accuracy. Even when including Solution 3, we still achieve an R-squared comparable to clinical standards, as seen in Table X.

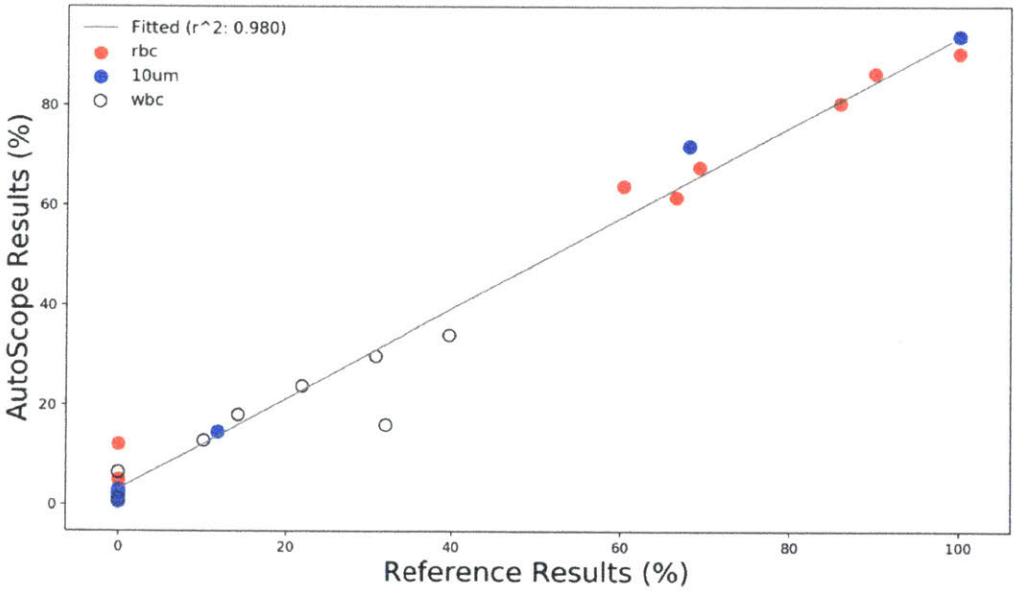



*Figure 6-9. The AutoScope results compared to the reference results in terms of relative particle counts. The AutoScope results are the average across all images taken for a single synthetic urine solution.*

Next, we will separate out the results based on which experimental day they were collected. For each experiment, the entire experimental process was repeated, including making new reference samples and new synthetic urine solutions. The goal of repeating the experiment was to demonstrate repeatability across different days with different reference solutions. Figure

6-10 shows the results across both days. We achieved an R-squared value of 0.974 for Day 1 and 0.999 for Day 2. Thus, we demonstrated accurate results across both days.

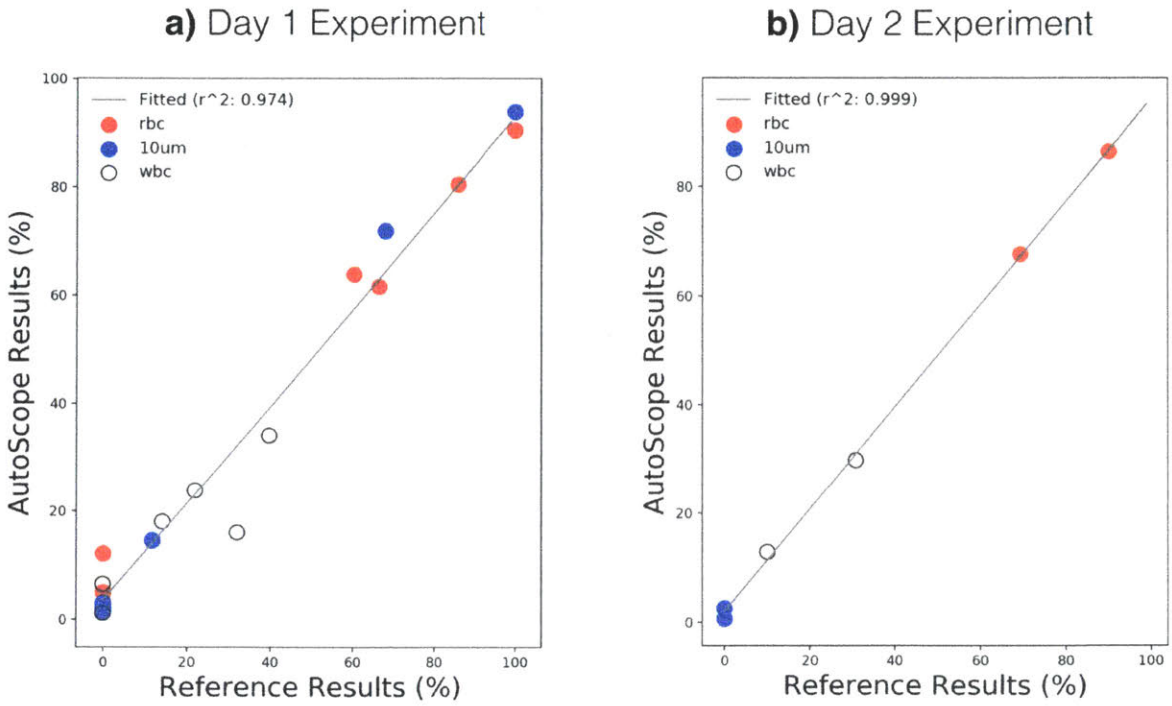



*Figure 6-10. The reference results compared to the AutoScope results across the two experimental days. For each experimental day, the experimental procedure was re-performed from the beginning.*

Finally, we will present the results of each image taken instead of each solution. In Figure 6-9 and Figure 6-10, each marker represented the results for a single synthetic urine solution, using the averages of multiple AutoScope images. In Figure 6-11, each marker represents the results of a single image. The vertical pattern of markers represents different images across the same solution. By looking at the data in this way, we obtain an R-squared value of 0.95. The decreased R-squared value is due to the variation of results across different images (field-of-views) of the same solution. Even with a perfect measurement system, we still expect a variation of results across images due to the random distribution of particles in a mixture. Even with this variation, we still observe a significant correlation. The Wald Test allows us to reject the null hypothesis that the slope is 0 (the two-sided p-value was significantly less than 0.01) and thus conclude that there is a correlation between the results.

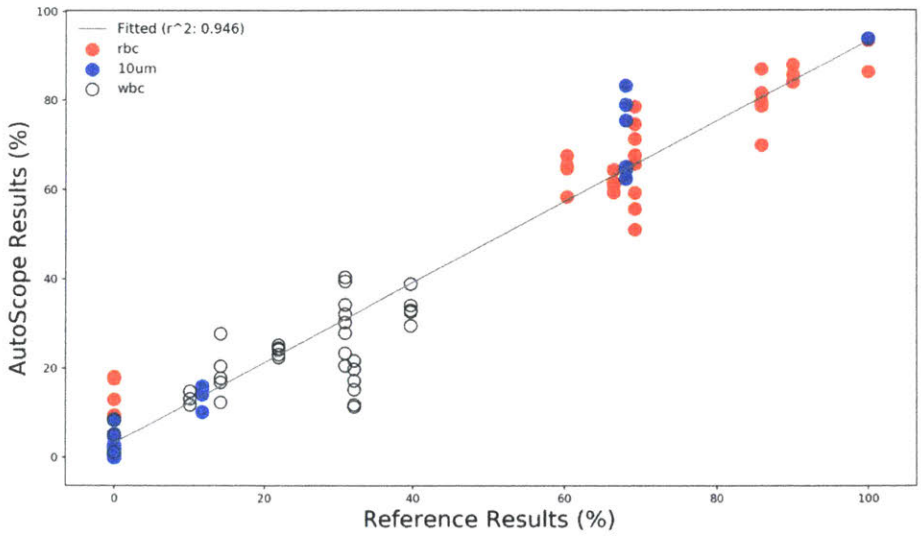



*Figure 6-11. The reference results compared to the AutoScope results in terms of relative particle counts for each image.*

# 7 Conclusion

## 7.1 Summary

In summary, this work introduces the AutoScope, an end-to-end automated, low-cost microscopic urinalysis system able to accurately detect red blood cells and white blood cells in urine. The AutoScope system is composed of three core functional parts: the image acquisition system, the segmentation system and the classification system.

For the image acquisition system, we used a reversed lens approach to achieve an end-to-end resolution of 5.86-8.29um. We obtained this resolution while still maintaining a large field of view (3.68 x 2.76 mm) and a low cost ($6-$12 for the lens). In order to augment the contrast of our semi-transparent biological specimen, we implemented a darkfield illumination lighting scheme. We designed the AutoScope to be low-cost, and were able to attain a total bill of materials of $57-$92.

Next, we developed a convolutional neural network to automatically differentiate red blood cells and white blood cells from interfering particles in the images taken by the AutoScope. First, using pre-labeled images of urine particles from a medical laboratory, we estimated that we can achieve a classification accuracy of between 84.7% to 90.0% at the AutoScope resolutions. This served as a proof of concept of the classification algorithm. Next, we used images of urine particles taken by the AutoScope to demonstrate an overall classification accuracy of 89.3% in differentiating red blood cells, white blood cells and microbeads from other particles.

To achieve automated particle segmentation, we implemented and trained a fully-convolutional network to label each pixel of an AutoScope image as either foreground or background. Leveraging this technique, we achieved a segmentation accuracy of 99.1%, 95.3%, and 98.5% for WBC, RBC, and microbead images, respectively.

Finally, we validated the performance of the AutoScope system by running two experiments. First, we determined the specificity and sensitivity of the AutoScope by generating 209 digital urine specimens modeled after urine specimens received in medical labs. We obtained a sensitivity of 88% and 91% and a specificity of 89% and 97% for red blood cells and white blood cells, respectively. Next, we validated the end-to-end accuracy of the AutoScope

system, from image acquisition to particle segmentation to particle classification. We accomplished this by fabricating 8 synthetic urine samples from white blood cells, red blood cells and 10um microbeads. The reference results were confirmed through a medical laboratory. When we compared the AutoScope's counts to the known reference counts, we achieved an R-squared value of 0.980 across all target particles. To demonstrate repeatability, we performed the full experimental procedure across 2 different days, achieving an R-squared of 0.974 on Experimental Day 1 and 0.999 on Experimental Day 2. The sensitivity, specificity and R-squared values for the AutoScope are comparable (mostly better) than the validation metrics for the iQ-200, a $100,000-$150,000 state-of-the-art semi-automated urinalysis system.

In conclusion, we have shown a proof of concept for an end-to-end automated microscopic urinalysis system that accurately counts red blood cells and white blood cells at a low cost.

## 7.2 Future Work

In future work, we plan on addressing the following topics:

- **Stain particles with a fluorescent marker to create the labeled dataset.** In our current work, we manually label particles using a combination of the researcher's discretion and the knowledge of the target particle that is contained in a solution. However, this process leads to issues when there are many interfering particles in the solution, which is especially true for the white blood cells. In this work, we use BA/F3 cells that are cultured in a medium that include fetal bovine serum, dead BA/F3 cells and other interfering particles. During labeling, these interfering particles get labeled as white blood cells and some white blood cells get labeled as interfering particles. Since a model can only be as good as the training data, we need to make efforts to improve the training data. The solution is to stain the cells with fluorescent dye. The AutoScope can then take white light images of a slide followed by fluorescent imaging of the same slide to generate significantly more accurate labels.
- **Demonstrate accuracy across a brother set of urine particles.** In our current work, we focused on counting red blood cells and white blood cells since these are the most common and most clinically relevant urine particles. Moving forward, we plan to expand

the set of urine particles that the AutoScope can detect. Specifically, we plan to include casts, epithelial cells, and possibly bacteria.

- **Demonstrate accuracy in counting red and white blood cells in blood.** Abnormal cellular counts of blood cells can indicate many diseases, and so complete blood counts (CBCs) are among the most commonly performed blood tests in medicine, routinely performed at physical examinations. CBC are used to diagnose or monitor diseases from anemia to chronic inflammatory diseases to HIV to cancer (during treatment). Today, CBCs are mostly done with a flow cytometer, which can be prohibitively expensive for small clinics and certainly for patients. So, in addition to microscopic urinalysis, we plan to validate the AutoScope's performance on blood based cell counting.
- **Develop a fluidic capsule to replace the glass slides to improve usability of the system.** In our current work, we need to prepare a glass slide in order to count particles in a solution. This will be a significant challenge to using the AutoScope at the point-of-care or in a home setting. To eliminate this obstacle, we plan on developing a disposable fluidic capsule. The fluidic capsule will be designed to take a variable amount of blood/urine and distribute it across a service for imaging.
- **Redesign the mechanical case of the AutoScope to make it more user-friendly.** Currently, the AutoScope's mechanicals were designed for research and development. In order to increase the usability of the system at the point-of-care, we plan to miniaturize the system and increase the system's robustness. Also, in a lab setting, a manual focus is useful since the size/type of slides vary. However, in a commercial system, we can build an autofocus system that only needs to make very fine adjustments in z-height.

# 8 Appendix

The core AutoScope code can be found on my Github at https://github.com/SidneyPrimas/AutoScope.

The code is organized into three main directories:

1. AutoScope_Algos

This folder contains the algorithms for the classification and segmentation AutoScope images. The most important scripts are contained in the core_algo directory. Within this directory, I want to highlight a few workhorse scripts that are starting points for different processes.

- **Data Preparation Scripts**: These scripts put the Autoscope's images into the proper folder structure necessary for model training. They include data_preperation/create_classification_folder_from_labels.py and data_preperation/create_segmentation_folder.py.
- **Training Scripts:** Scripts that train neural networks to perform particle segmentation and classification. These include train_classification_particles.py and train_segment_particles.py.
- **Prediction Scripts:** Scripts that used the trained models to predict the particle segmentation and classification on new AutoScope images. These include: process_urine_classify.py and process_urine_segment.py.

2. Labeling_Algos

This folder contains scripts to build tools that allow a user to manually label the location and type of particle in AutoScope images. This is done to develop a training dataset. The training datasets are fed into the training scripts above.

3. Sub_Tasks

This folder contains scripts to perform other types of analyses needed for my Master's thesis that are not related to classification or segmentation of particles. For example, modeling the illumination pattern of the Autoscope, calculating the end-to-end resolution of the Autoscope system, etc.

# 9 Bibliography


1. Delanghe, J. New screening ciagnostic techniques in urinalysis. *Acta Clin. Belg.* **62,** 155–161 (2007).

2. *Modern Urine Chemistry: Application of Urine Chemistry and Microscopic Examination in Health and Disease*. (2009).

3. Kouri, T. *et al.* Towards European urinalysis guidelines. *Clin. Chim. Acta* **297,** 305–311 (2000).

4. Simerville, J. A., Maxted, W. C. & Pahira, J. J. Urinalysis: A comprehensive review. *American Family Physician* **71,** 1153–1162 (2005).

5. Ince, F. D. *et al.* The comparison of automated urine analyzers with manual microscopic examination for urinalysis automated urine analyzers and manual urinalysis. *Pract. Lab. Med.* **5,** 14–20 (2016).

6. Bogoch, I. I. *et al.* Mobile phone microscopy for the diagnosis of soil-transmitted helminth infections: A proof-of-concept study. *Am. J. Trop. Med. Hyg.* **88,** 626–629 (2013).

7. Switz, N. A., D'Ambrosio, M. V. & Fletcher, D. A. Low-cost mobile phone microscopy with a reversed mobile phone camera lens. *PLoS One* **9,** (2014).

8. D'Ambrosio, M. V *et al.* Point-of-care quantification of blood-borne filarial parasites with a mobile phone microscope. *Sci. Transl. Med.* **7,** 286re4 (2015).

9. Hans, C., Merchant, F. A. & Shah, S. K. Decision fusion for urine particle classification in multispectral images. in *Proceedings of the Seventh Indian Conference on Computer Vision, Graphics and Image Processing - ICVGIP '10* 419–426 (2010). doi:10.1145/1924559.1924615

10. Ongun, G. *et al.* An automated differential blood count system. *Annu. Reports Res. React. Institute, Kyoto Univ.* **3,** 2583–2586 (2001).

11. Shen, M. & Chen, D. Study on urinary sediments classification and identification techniques. in (eds. Sheng, Y., Zhuang, S. & Zhang, Y.) **6027,** 60271A (International Society for Optics and Photonics, 2006).

12. Murasaki, Y. & Tanigiichi, K. Pattern recognition of urinary sediment images applying a fuzzy-neural network. *IEICE Trans.* 2630–2632 (1993).

13. Karpathy, A. & Li, F.-F. Convolutional Neural Networks for Visual Recognition (CS231). (2017). at <http://cs231n.github.io/>

14. Torralba, A. & Freeman, B. Advances in Computer Vision (8.869). (2016). at <http://6.869.csail.mit.edu/fa17/schedule.html>

15. LeCun, Y. A., Bottou, L., Orr, G. B. & Müller, K. R. Efficient backprop. *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)* 9–48 (2012). doi:10.1007/978-3-642-35289-8-3

16. What is AWS - Amazon Web Services. at <https://aws.amazon.com/what-is-aws/>

17. Nomura, Sasaki & Kishimoto. IMX219PQ. at <https://www.sony-semicon.co.jp/products_en/new_pro/april_2014/imx219_e.html>

18. Murphy, D., Chambers, W., Fellers, T. & Davidson, M. MicroscopyU - The Source for Microscopy Education. (2017). at <https://www.microscopyu.com/>

19. Davidson, M., Abramowitz, M., Neaves, S. & Hoffman, R. Molecular Expressions Microscopy Primer. (2015). at <https://micro.magnet.fsu.edu/primer/>

20. *Luxeon K2: Technical Datasheet DS51*. at <https://www.lumileds.com/uploads/54/DS51-pdf>

21. Skandarajah, A., Reber, C. D., Switz, N. A. & Fletcher, D. A. Quantitative imaging with a mobile phone microscope. *PLoS One* **9,** (2014).

22. Steenblik, R. & Steenblik, P. Lenses and uses, including microscopes. (2000). at <https://patents.google.com/patent/US6847480B2/en>

23. Ford, B. *Single Lens: Story of the Simple Microscope*. (William Heinemann Ltd, 1985).

24. Rottenfusser, R., Davidson, M. & Wilson, E. Carl Zeiss Microscopy Online Campus - Education in Microscopy and Digital Imaging. at <http://zeiss-campus.magnet.fsu.edu/>

25. Edmund Optics - Imaging + Microscopy Application Notes. at <https://www.edmundoptics.com/resources/application-notes/imaging/>

26. Van Valen, D. A. *et al.* Deep Learning Automates the Quantitative Analysis of Individual Cells in Live-Cell Imaging Experiments. *PLoS Comput. Biol.* **12,** (2016).

27. Abadi, M. *et al.* TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems. (2016). doi:10.1038/nn.3331

28. Chollet, F. Keras. (2018). at <https://github.com/keras-team/keras>

29. Van Der Walt, S., Colbert, S. C. & Varoquaux, G. The NumPy array: A structure for efficient numerical computation. *Comput. Sci. Eng.* **13,** 22–30 (2011).

30. Jones, Oliphant & Peterson. SciPy: Open Source Scientific Tools for Python, 2001. (2001). at <http://www.scipy.org/>

31. Ranzato, M. *et al.* Automatic recognition of biological particles in microscopic images. *Pattern Recognit. Lett.* **28,** 31–39 (2007).

32. Krizhevsky, A., Sutskever, I. & Hinton, G. E. ImageNet Classification with Deep Convolutional Neural Networks. *Adv. Neural Inf. Process. Syst.* 1–9 (2012). doi:http://dx.doi.org/10.1016/j.protcy.2014.09.007

33. Sermanet, P. *et al.* OverFeat: Integrated Recognition, Localization and Detection using Convolutional Networks. *arXiv Prepr. arXiv* 1312.6229 (2013). doi:10.1109/CVPR.2015.7299176

34. Wu, S., Zhong, S. & Liu, Y. Deep residual learning for image steganalysis. *Multimedia Tools and Applications* 1–17 (2017). doi:10.1007/s11042-017-4440-4

35. Tai, L., Ye, H., Ye, Q. & Liu, M. PCA-aided fully convolutional networks for semantic segmentation of multi-channel fMRI. in *2017 18th International Conference on Advanced Robotics, ICAR 2017* 124–130 (IEEE, 2017). doi:10.1109/ICAR.2017.8023506

36. Ronneberger, O., Fischer, P. & Brox, T. U-Net: Convolutional Networks for Biomedical Image Segmentation. *Miccai* 234–241 (2015). doi:10.1007/978-3-319-24574-4_28

37. Badrinarayanan, V., Kendall, A. & Cipolla, R. SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.* **39,** 2481–2495 (2017).

38. Chen, L.-C., Papandreou, G., Kokkinos, I., Murphy, K. & Yuille, A. L. DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs. (2016). doi:10.1109/TPAMI.2017.2699184

39. Long, J., Shelhamer, E. & Darrell, T. Fully convolutional networks for semantic segmentation. in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition* **07-12-June,** 3431–3440 (2015).

40. Simonyan, K. & Zisserman, A. Very Deep Convolutional Networks for Large-Scale Image Recognition. *Int. Conf. Learn. Represent.* 1–14 (2015). doi:10.1016/j.infsof.2008.09.005

41. Wah, D. T., Wises, P. K. & Butch, A. W. Analytic performance of the iQ200 automated urine microscopy analyzer and comparison with manual counts using Fuchs-Rosenthal cell chambers. *Am. J. Clin. Pathol.* **123,** 290–296 (2005).

42. Altekin, E. *et al.* New generation IQ-200 automated urine microscopy analyzer compared with KOVA cell chamber. *J. Clin. Lab. Anal.* **24,** 67–71 (2010).

43. URINE MICROSCOPIC (1095). at <http://www.questdiagnostics.com/testcenter/BUOrderInfo.action?tc=1095&labCode=QER>