|  |
| --- |
| **Impurivision: A High Performance Mobile Application  for Identifying Water Contamination using Deep Learning**  Palash Shah Kartik Chugh |

|  |
| --- |
| Abstract Contaminated water causes disease and poverty in communities of the developing and developed world. The automated, handheld detection of water contamination has the potential to empower the 6 billion global users of mobile phones to be aware of impurities in their water supply. In this project, we develop a mobile application that prompts the user to take two bird’s-eye view photos: one of an empty cup, and another of the same cup filled with water. The app subtracts the former from the latter, generating a “difference image”. A convolutional neural network is built and trained on 8,505 such difference images of water treated with the most common contaminants. Then, the CNN is integrated into the mobile app to allow the user to scan their water for these contaminants. The CNN’s purity classifier detects the presence of contaminants with 97.4% accuracy, and its substance classifier identifies the specific contaminants with 89.6% accuracy. Lastly, a reporting feature and additional tools are included into the app to inform local public health officials of water contamination issues. |

# 

# Table of Contents

[**Abstract**](#_xkmvz9qgcf9i) **1**

[**Table of Contents**](#_2sy2fwio02er) **2**

[**Key Words**](#_rtuny3muj5no) **3**

[**Abbreviations and Acronyms**](#_97aso87b5fa6) **3**

[**Acknowledgements**](#_1aiykfbor49k) **3**

[**Introduction**](#_sxdn5nhh3m8s) **4**

[**Materials**](#_yts8u1b4htvs) **5**

[**Methodology**](#_2h1hflr7t0ar) **6**

[Mobile Application](#_omy2x2rh8bme) 6

[Contaminant Selection](#_ps726w3m9854) 7

[Data Construction](#_kjxhv7ucfb0y) 9

[Data Optimization](#_fbtm6bhd0j87) 11

[Convolutional Neural Network](#_ssnurxsflnzn) 11

[Overview](#_abgbgcytjc30) 11

[Architecture](#_r8p12s2dj3wr) 14

[Training](#_c09fv746yrbb) 15

[**Results**](#_yvzplb9y5deq) **16**

[Data Analysis](#_sc71xfc8127q) 16

[Validation](#_r57n32goy81b) 16

[Raw Difference Purity Classifier](#_y64g6jpv3spo) 17

[Raw Difference Substance Classifier](#_dqmtroh2plxm) 18

[Filtered Difference Classifiers](#_my3590ttzq1c) 19

[**Discussion**](#_w8kpm6k0nz9w) **20**

[Results](#_5l7ltvl5vp6p) 20

[Impacts](#_el2lhalxt583) 20

[**Conclusions**](#_2ws0z9atqmli) **23**

[Findings](#_dwplkbu3y8ia) 23

[Future Improvements](#_196xl9fee2fp) 23

[**References**](#_si5ef6t9e0zq) **24**

# Key Words

Water contamination, groundwater pollution, computer vision, machine learning, deep learning, neural network, convolutional neural network, red-green-blue color model, image compression, k-fold cross validation

# 

# Abbreviations and Acronyms

Red-green-blue (RGB), convolutional neural network (CNN), Matthews correlation coefficient (MCC), true positive (TP), true negative (TN), false positive (FP), false negative (FN), application programming interface (API).

# Acknowledgements

The completion of Impurivision could not have been possible without the help of multiple teachers. First, we would like to thank Ms. Constantino, a biology teacher and the science fair coordinator for Westfield High School, who guided us through the project and helped provide a different perspective on the project.

Additionally, we would like to extend our gratitude towards Ms. Koppel, a chemistry teacher at Chantilly High School who provided invaluable input regarding the selection of substances and the creation of accurate simulants for toxic contaminants.

# Introduction

The planet is facing an immense water crisis: surging population growth worldwide is requiring more water than is available, pushing communities and countries to compete for sources of clean water. As resources are strained by the demographic, economic, and technological demands of Earth’s 7.5 billion inhabitants, access to clean water for drinking, bathing, and cooking is a dire situation. All in all, 2.3 billion people lack such access, and as a result, are often forced to rely on unsafe water sources laden with industrial pollutants and groundwater contaminants. It is tempting to view the issue as unique to the developing world, but aging infrastructure has made developed countries increasingly vulnerable to pollution, which frequently stems from industrial accidents and faulty water cleaning processes.

At the same time, however, mobile device usage is at unprecedented levels, with a stunning 6 billion individuals estimated to have access to a device in 2018. This includes the world’s largest populations, namely India and China, representing a nearly 1.4 billion people market share. In fact, because mobile phones are already ubiquitous in saturated Western markets, their growth is highest in developing countries — about 470% in the past decade alone. As one article in *Mobile Beyond* put it, “Adoption of cell phones among the developing world’s rural poor is exploding, to say the least.” Despite substantial issues with regards to water quality, mobile devices have become an increasingly viable platform for social advancement in the developing world, and have been one in the developed world for quite some time. As such, they represent a unique opportunity by which to address the water crisis. By outlining a deep learning approach to detecting water contamination, as well as a functioning mobile application to make its algorithms useable, this research paper hopes to break ground in the field of water purity analysis.

The following are the three major objectives set prior to beginning work on the project:

* Solution: Detect for the majority of prevalent water contaminants
* Innovation: Demonstrate the viability of a deep learning approach to water purity analysis
* Application: Develop tools to translate our novel approach into real-world impacts

The objectives heavily shaped all parts of the project, from data construction to deep learning, and app development to image compression. The extent to which these objectives were met are discussed in the report conclusion.

# Materials

The majority of tangible materials required for this project were used for the data construction process, in which thousands of images of clean and contaminated water were generated. This required the a variety of cup types, pollutants, and dyes to simulate toxic contaminants.

*Cups*: small white foam cup, medium-sized white paper cup, large red plastic cup (~315 ct. each).

*Pollutants*: gasoline, sediment, paint, sodium, pesticides, fertilizer, insect repellent

*Dyes*: red, orange, yellow, blue, brown, chrome (~2 oz. each)

In addition, several hardware and software technologies were employed to develop the computational basis of our project.

*Programming APIs*: TensorFlow, Glueviz, Python, Java, Android

*Programming Environments*: Android Studio, IntelliJ IDEA, Jupyter Notebook

*Deep Learning Server*: Windows Server 2016; 8 CPUs, 64 GB

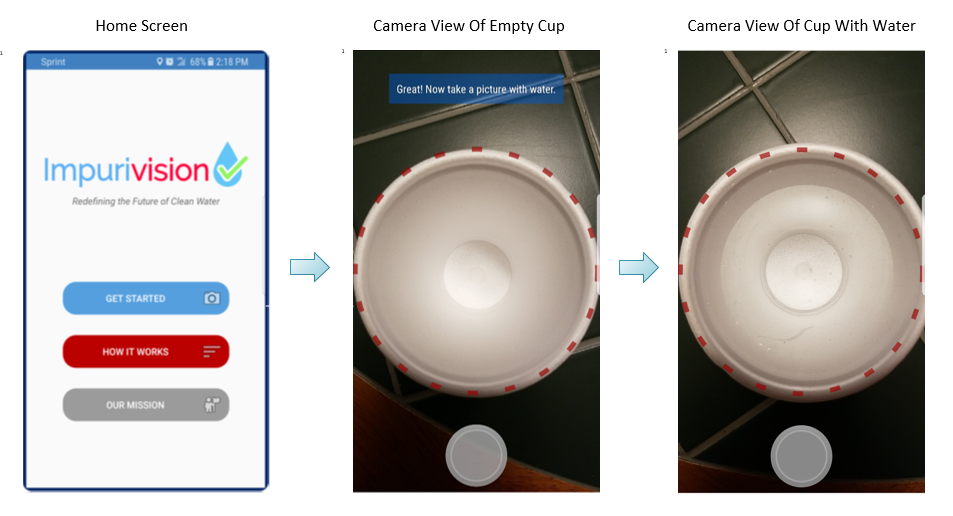
*Mobile Devices*: Samsung Galaxy S8, Samsung Galaxy S4, Google Nexus 7

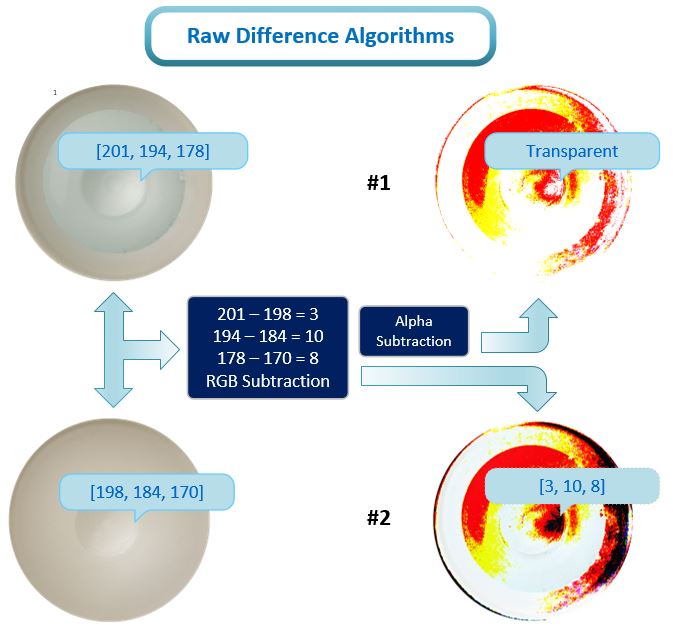
The specific use and functions of these materials are explained in the remainder of this report.

# 

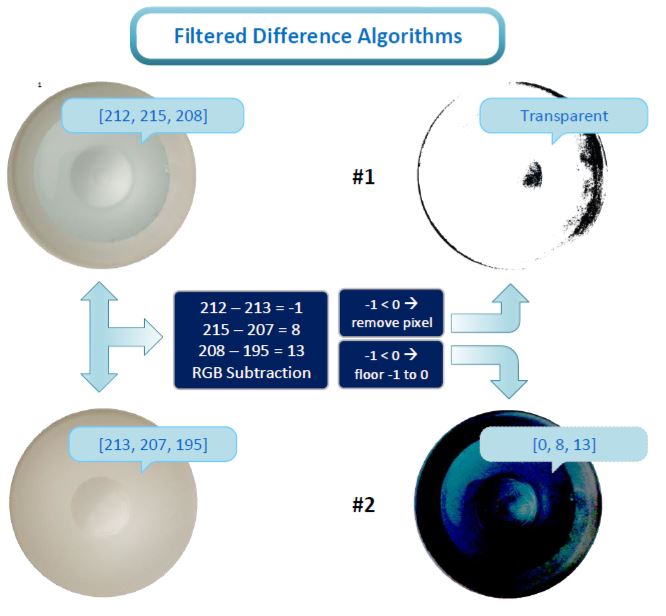
# Methodology

## Mobile Application



The Impurivision mobile application was developed to allow the user to send photos of their water to the CNN, but was also used to create the CNN’s dataset. The RGB color model used extensively by the mobile application contains three component values — red, green, and blue, as implied — stored in computers as integer numbers ranging from 0 to 255, inclusive. The app requires the user to take two pictures: one of an empty cup, and another of that cup filled with the water they intend to test. Subsequently, it performs pixel-by-pixel image “subtraction” by isolating each of the pixels’ RGB channels and computing the difference between them. The result is three RGB “difference components”, which are encoded into a single “difference image” using bit shifting. By subtracting the empty cup image from the water cup image, the “difference image” represents a method by which the CNN can evaluate the user’s water with minimal interference from lighting, cup types and colors, and other variable conditions. The difference image is classified by the purity classifier and the substance classifier, both of which were trained on thousands of difference images pertaining to clean water, 11 substances, and 4 substance mixtures. The results of the classification are made available to the user within minutes, notifying them of whether their water is safe or unsafe to drink, and if indeed the latter, identifying the substance(s) detected.

The original difference algorithm operated in a simplistic fashion; it computed the difference between the two images’ color integers. Because Android color integers are encoded in ARGB format (alpha, red, green, blue), in many cases, alpha values were subtracted out. Alpha represents opacity, with 0 being fully transparent, and 255 being fully opaque; in a camera-taken photo, all pixels are fully opaque. Consequently, raw difference images often had transparent pixels where none ought to have existed, introducing a large amount of error. The difference algorithm was revised to extract RGB components from each pixel of the two images, subtract these components, and encode the difference components into a raw difference image. This retained all subtractive data, as opposed to the first algorithm.

In addition to the “raw” difference algorithm, two “filtered” difference algorithms were created to attempt to correct for a phenomenon known as RGB underflow. In computer science, arithmetic underflow occurs when a number is less than the minimum possible value, resulting in a “flow” to the maximum value. In the case of RGB, the range of possible components values is [0, 255], so -1 becomes 255, -2 becomes 254, and so on. Underflow is usually irrelevant in image processing since any images taken on a camera, phone, or other device are already constrained to the acceptable range. However, it was pertinent in this project due to the subtraction of two images, which makes it possible for the value of RGB components to initially be negative, and then underflow. When the difference between the water cup and empty cup images is a small magnitude below 0, it underflows to a value of large magnitude above 0 on the RGB scheme; -15, for instance, underflows to 241. As a result, most raw difference images are characterized by extremely bright aberrations, since these wrapped around values are abnormally high. In theory, this has a distortive effect on the difference images with the potential to impair deep learning, necessitating an algorithm to filter out the faulty data.

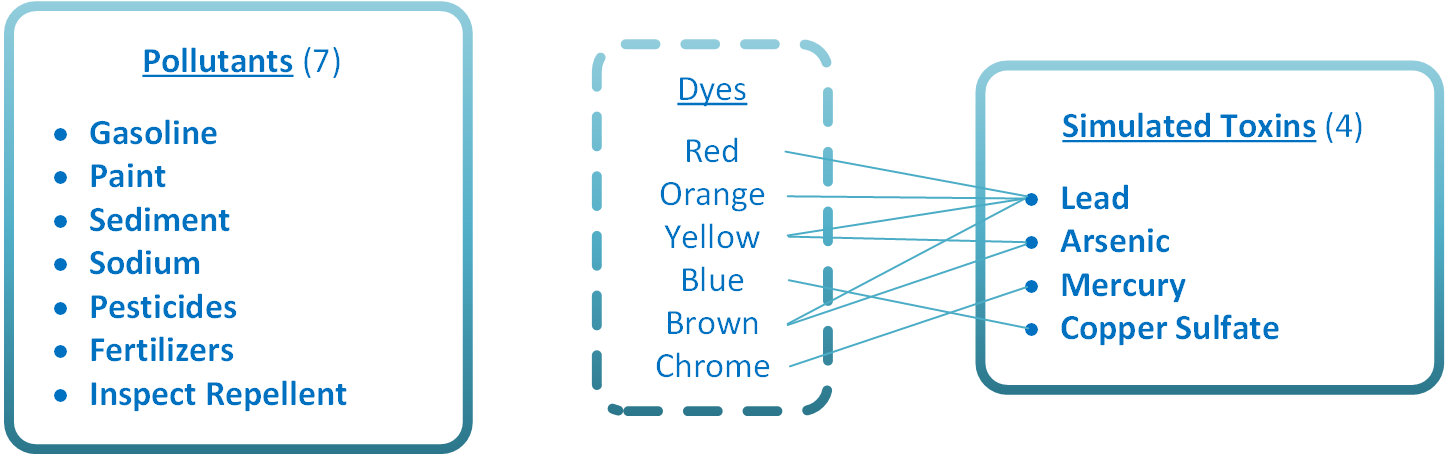
The first filtration algorithm was designed to remove all pixels in which any of the RGB component values resulted from underflow. As seen in the image, substantial amounts of data were eliminated under this regime, resulting in primarily empty (transparent) filtered difference images. A second filtration algorithm revised the approach of the original by “flooring” individual RGB components to 0, resulting in filtered difference images with less data loss but little data variation.

## Contaminant Selection

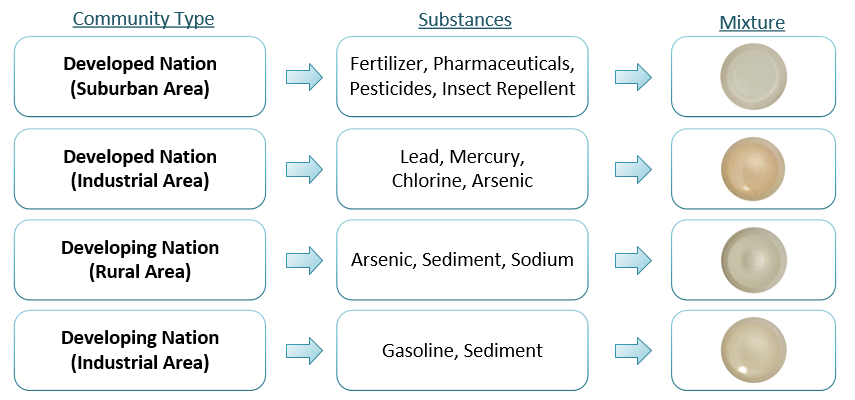
To identify the most common pollutants in developing and developed nations, extensive research was done. Substances that could be visually identified were selected based upon their prevalence in the real world. While some substances inherently manifest as slight visual alterations in water, others result in visible indicators such as pipe rusting. For instance, while arsenic itself might not be yellow, a yellow tint in water is a known sign of arsenic contamination.

Created by The Fiscal Times 

Another crucial substance chosen was lead, due to its adverse health effects, and prevalence in the real world. While small amounts of contamination will not harm humans, over time, the accumulation of lead in the human body will cause diseases like dementia, and lung cancer. Lead can enter drinking water in a variety of different ways. For example, service pipes that are made out of lead often corrode, polluting the water that flows through them. This corrosion is what causes the lead to have a reddish-orange tint as depicted above. The rest of the substances were researched and identified using the official list of major contaminants that the Environmental Protection Agency provides, information from the United Nations, and extensive media reporting.





Eleven substances were identified as major contaminants, in accordance with the objective of addressing the majority of contamination situations. Although the selected substances constitute the most frequent and widespread contaminants, it would be unreasonable to assume that they always arise in water independently. As such, substance mixtures, or groups, were determined to be necessary to expand the realistic applications of Impurivision. Rather than forming random groupings, however, four logical substance “profiles” were developed to simulate the water in a range of community types.

By including data profiles based on real world manifestations of water contamination, the CNN is better equipped to handle real world use through a comprehensive dataset that includes difference images of individual contaminants as well as mixtures.

## 

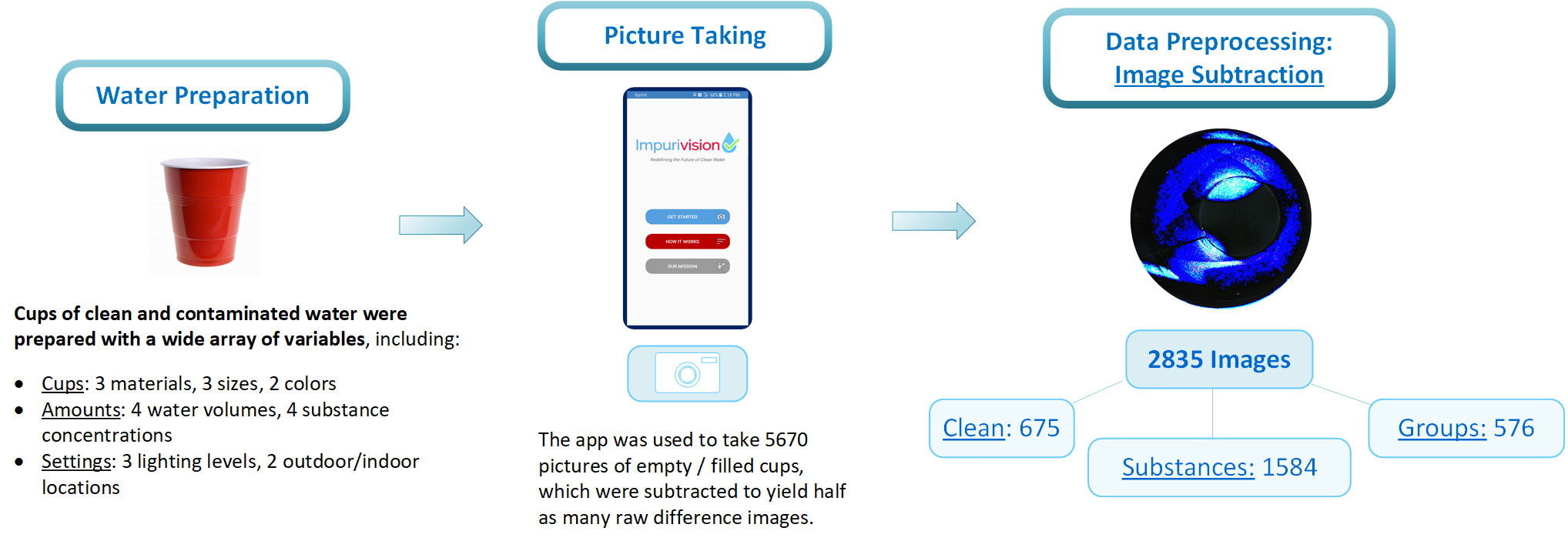
## 

## 

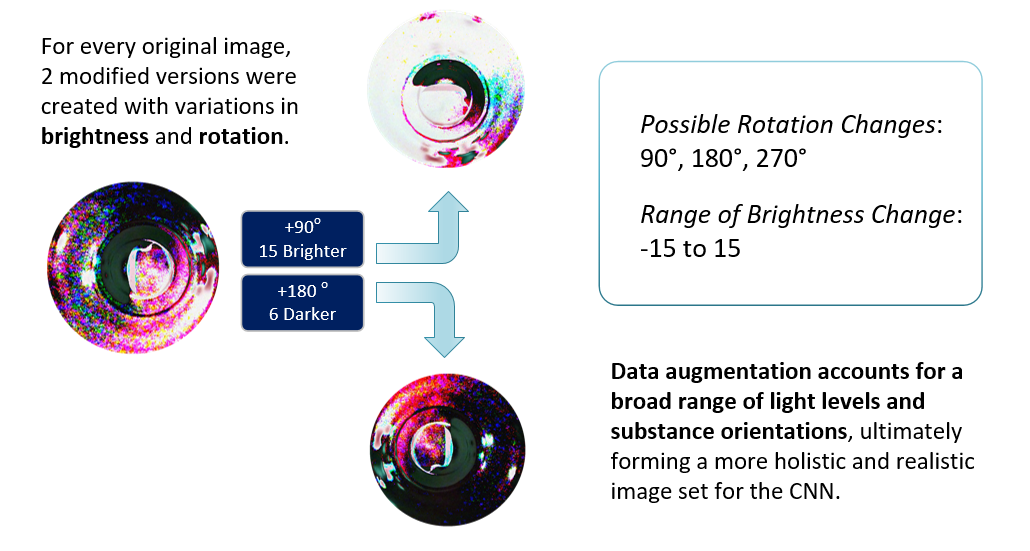
## 

## 

## Data Construction

Following the completion of the substance list, difference images were compiled to form the dataset on which to train the CNN. Gallons of water — both clean and contaminated — were prepared for this purpose.   
The mobile application was used to first take a picture of an empty cup, then to take a picture of the cup with water poured inside. The app generated a difference image, which was downloaded, tagged either “clean” or the contaminant name, and added to the dataset. Water volumes and contaminant concentration were altered precisely, in addition to varying cup types and lighting conditions. This way, 144 distinct images were generated for each substance and substance group, as well as 675 for clean water, for a total of 2835 difference images. Next, the dataset was run through a Java-developed filtration algorithm intended to correct for RGB “underflow”, a rare but observable source of error in the difference

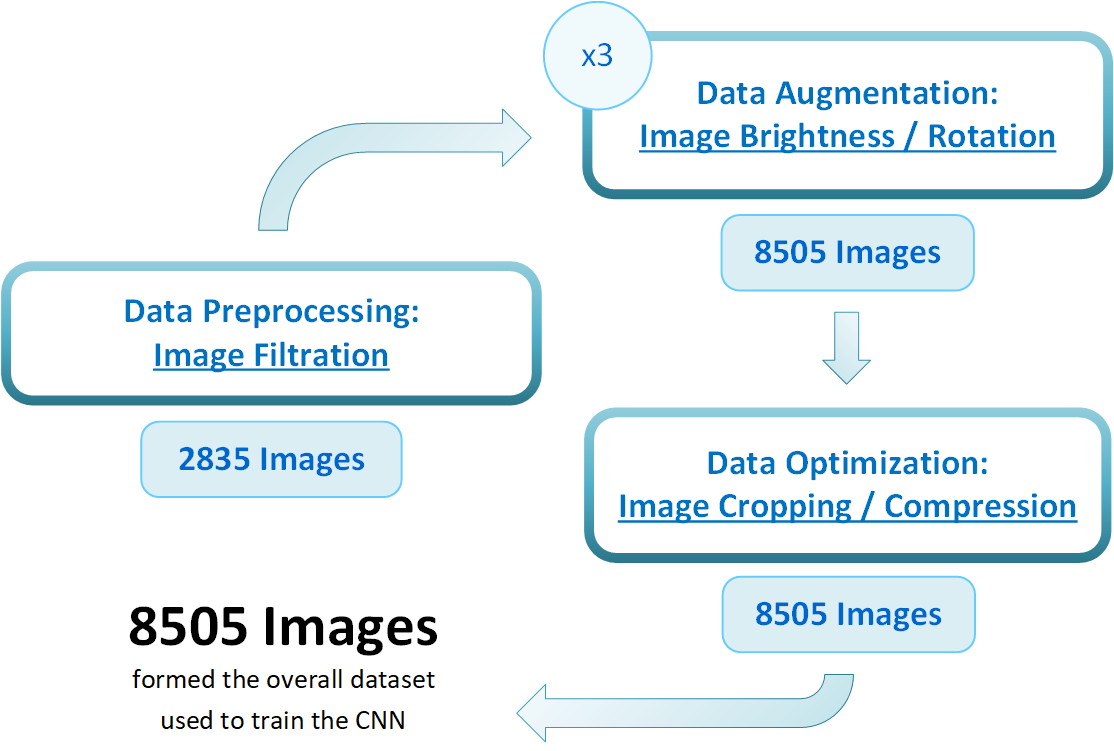
images.





As stated previously, the need to construct a large dataset and to account for environmental variables were determined to be crucial project objectives, necessary for high-performance convolutional neural networks. To accomplish both of these goals, an augmentation algorithm was developed in Java and was used on the 2835 difference images.



For each difference image, two additional ones were “cloned” and then modified, through variations in brightness and rotation. Consequently, a dataset of 8505 difference images was achieved. The data augmentation procedure effectively accounts for a broad range of light levels and substance orientations, ultimately forming a more holistic image set on which to train the CNN. Finally, the images were optimized through a Java program developed to crop and compress them.

## 

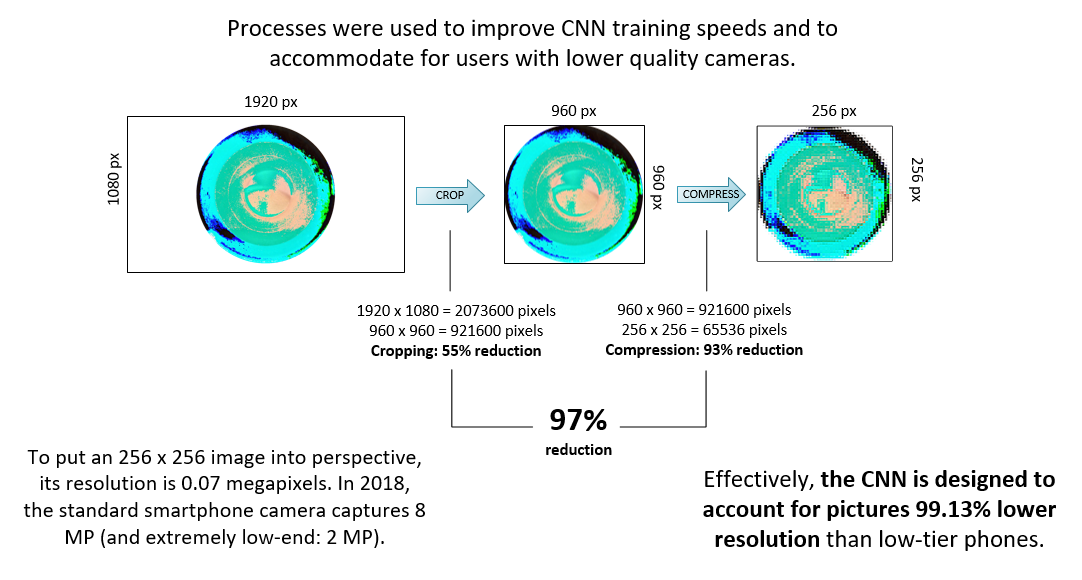
## 

## 

## Data Optimization

Cropping and subsequent compression of the difference images was used to improve training speeds for the CNN and more importantly, to accommodate for users with lower quality cameras.





Additionally, the Impurivision app was partially redeveloped to make it so that all difference images were cropped and then converted to the 256 x 256 pixel versions, before being run on the CNN. This ensures that all smartphones — those with high, medium or low-end cameras — are running on the same, consistent platform. The implications of data optimization are significant: the app is able to be used worldwide, in both developed and developing nations, since its underlying neural network and program code is designed to accommodate for low-end phone cameras.

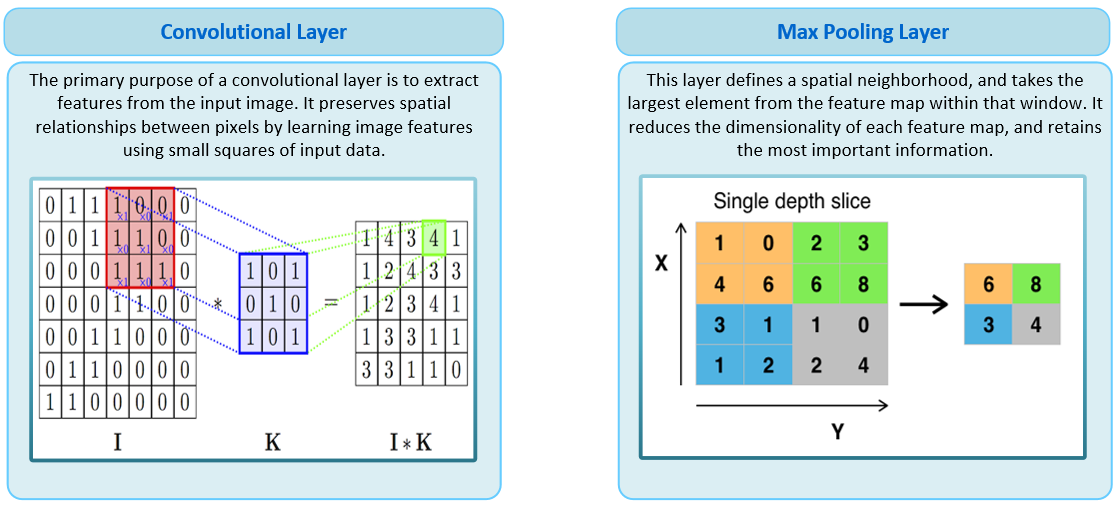
## Convolutional Neural Network

### Overview

A neural network is a type of computer algorithm that is modeled on the human brain and the nervous system. *Convolutional* neural networks represent a field of algorithms that have been widely applied to Image Recognition, and Image Pattern Recognition. These convolutional networks are very similar to general networks: they consist of neurons that have learnable weights and biases. Every neuron receives a certain input, performs a dot product, and applies activation functions to present non-linearity. What changes, is how the network is inherently optimized for image analysis. By explicitly making the assumption that the inputs are images, it allows the developer to encode certain properties into the architecture.

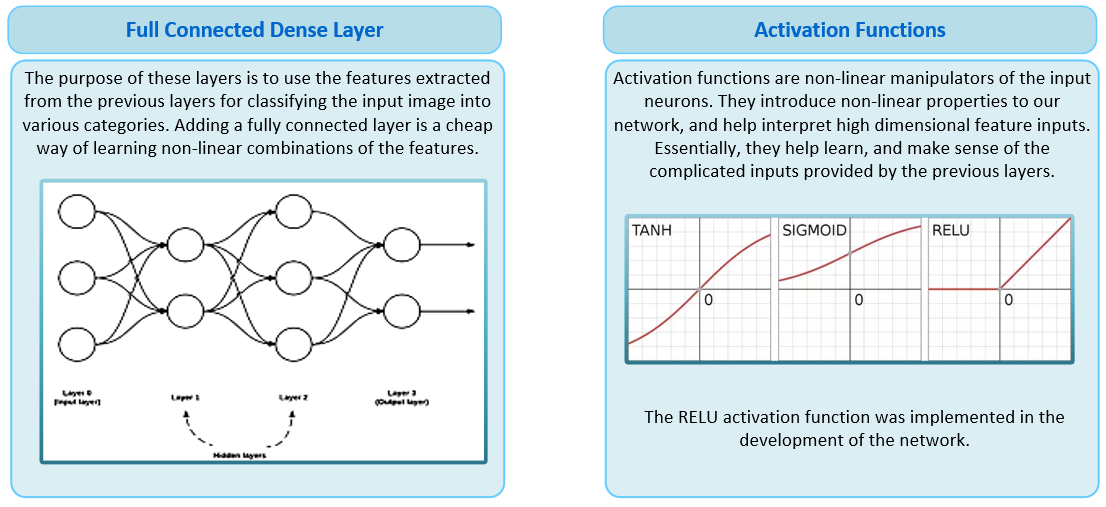
Convolutional neural networks take advantage of the fact that the input will almost certainly always be an image. Specifically, unlike a regular network, these networks have three dimensions: length, width, and height. Through the course of the network, the dimensionality and size of the input images diminishes as a result of the various layers. By the end of a CNN the input image is reduced to a one dimensional matrix which is then fed to the fully connected layer.

Convolutional neural networks are essentially constituted by a sequence of layers, and every layer of the CNN transforms one volume of inputs to another with a differentiable function. Three main layers are used when building these architectures: convolutional layer, pooling layer, and a fully connected layer. First, the input holds this image data in three color channels: red, green, and blue. 



One of the most essential layers of a CNN is the convolutional layer. This layer consists of a set of learnable filters, that extend through the full depth of the input volume. As we slide this filter through the first “forward pass” we produce a two dimensional activation map that gives the appropriate responses to every spatial filter. The network after this first pass learns filters that activate when certain patterns are identified in the input image. By the end of the layer, we will have a set of two dimensional activation maps that are stacked along the various depth dimensions to produce the output volume. This layer will compute the output of neurons that are locally connected in the input image, and then apply a dot product within a small region. Overarchingly, the convolutional layer extracts features from the input image while preserving spatial relationships. For example, one pixel by itself doesn’t tell the network what the image actually is, but a group of pixels can help the algorithm identify the input image.

The convolutional layers have a variety of hyperparameters. For example, when dealing with high dimensionality images, it is impractical to connect all the neurons in one layer to the next. Therefore, generally every neuron is only connected to a local region of the input volume. The spatial extent of this hyperparameter is called the receptive field or the filter size. Stride length is the distance to which one slides the convolutional filters. Generally, the stride length is 1 (filter slides over 1 horizontal dimension). This produces smaller spatial outputs, and improves training speeds. Moreover, the depth of the output volume is a parameter that number of filters we would like to use. The set of neurons that are being manipulated in a sequence in a network is called a depth column. Padding is used to replace the convolved neurons with a specified number (usually 0). In this study, zero-padding was used to preserve the spatial dimensions, and to apply downsampling.

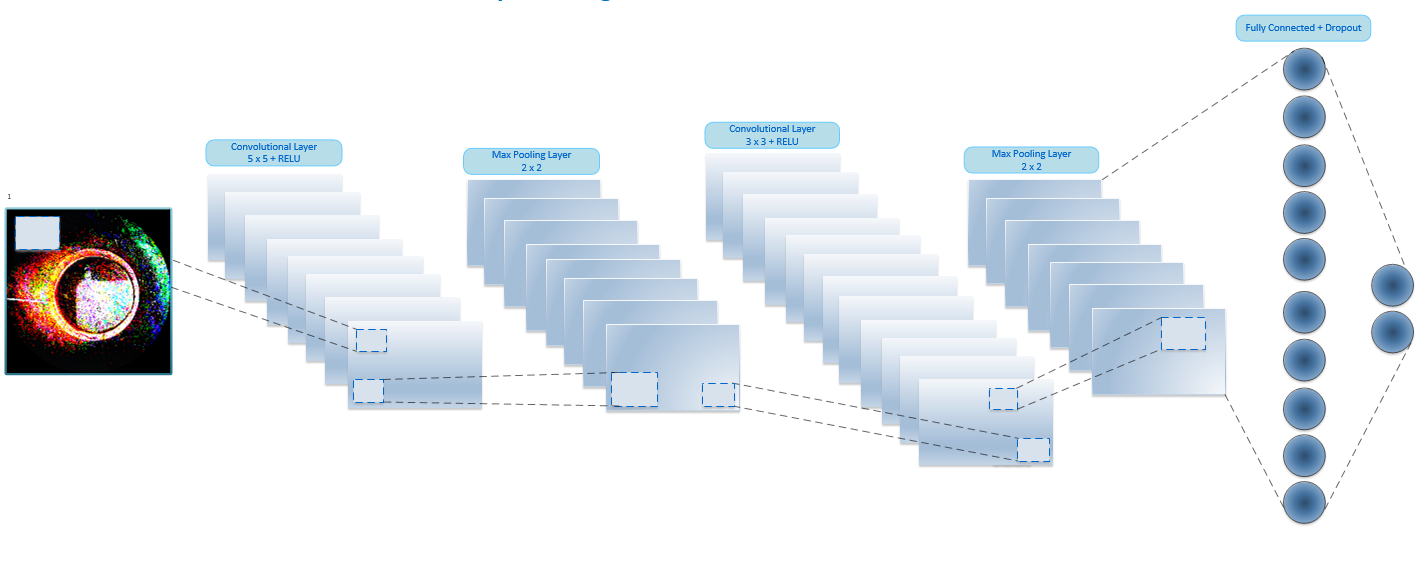
Another essential component of a CNN is the pooling layer which performs a downsampling operation thresholding at zero. This is done to reduce the dimensionality of the input image, and remove distracting noise. While it is not absolutely necessary to include pooling layers, it makes sense and actually improves the performance of the network by generating more robust and optimized features. The most common type of max pooling is one with a filter size of 2 by 2, with a stride length of 2, downsampling 75% of the input volume. In this case, every max pooling operation would be taking one value out of every 2 by 2 “array”.

Before the fully connected layers, dropout layers are introduced as regularization techniques for reducing overfitting in neural networks, and are a way to “average” the values of the individual neurons. These layers reduce overfitting by randomly turning neurons on and off, making it difficult for the model to overlearn the input dataset. Finally, the fully connected layers compute the individual class scores, resulting in a flattened matrix. As the name implies, and like a standard neural network, all neurons are interconnected in this layer.

### 

### Architecture

The norm of the deep learning world has been that “deeper” architectures are better for image classification. Though the CNN development was started off by attempting more complicated networks consisting of over 10 layers, simpler architectures were found to be more effective. It seems quite counterintuitive, that these high dimensionality inputs, are being classified without a lot of manipulation. However, the dataset created specifically for Impurivision was already well curated, making a “deep” network useful but not strictly necessary.



The Impurivision architecture consists of two convolutional layers, each coupled with two pooling layers. The convolutional layers have have 64 and 32 filters, in order. Max pooling layers were used to remove dimensionality, and improve training speeds. The size of the pooling layers used in this study were 2 \* 2 as our images started out quite small. Nonetheless, larger pooling layers were tried but seemed to remove too much valuable information. RELU activation functions are embedded within the layers to provide non-linearity to the model. This is followed by dropout, and fully connected layers. The first fully connected layer has 128 nodes, followed by the second layer which is used to compute the individual class scores.. The dropout threshold was set to 0.5, which means that the network will randomly turn off 50% of the neurons, before the transformed data is fed into the fully connected layer. This architecture was applied to train two different models: a purity classifier and a substance classifier.

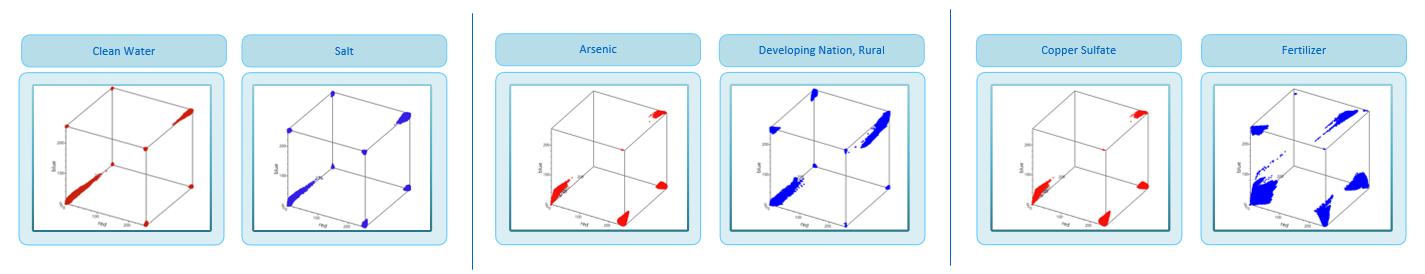
### Training

The CNN was trained on the images using the specialized architecture over a series of several epochs. The training set was determined to be 75% of our dataset, while the rest was excluded from the training set and used for testing. In the case of the purity classifier, the model reached maximum performance around 9 epochs, while the substance classifier reached maximum performance around 20 epochs. For both classification tasks, a standard batch size of 32 was used.

# 

# Results

## Data Analysis

Data Analysis was performed to confirm the validity of our deep learning approach. To the human eye, some substances appear identical to each other, so RGB plots were used to prove that a difference existed from the perspective of the mobile device. 

While these subtle differences may be indistinguishable to the human eye, 3 dimensional RGB plots prove that they do exist on a pixel by pixel basis. As such, it is clear upon an analysis of our dataset that difference images are capable of drawing distinctions between different substances, at least from a deep learning perspective. The images used to develop the above plots are shown below:

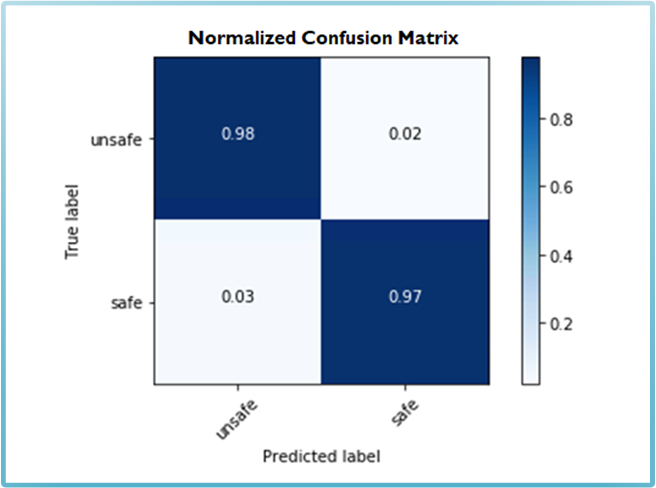
|  |  |
| --- | --- |
| Copper Sulfate | Fertilizer |

## Validation

Several validation techniques were performed to analyze the overall viability of the algorithm. Validation was performed on two different classifiers: one to evaluate the potability of water (purity classifier), and another to identify harmful substances in non-potable water (substance classifier).

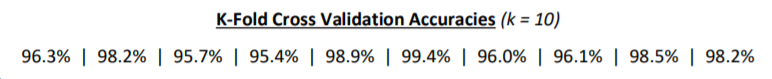
### 

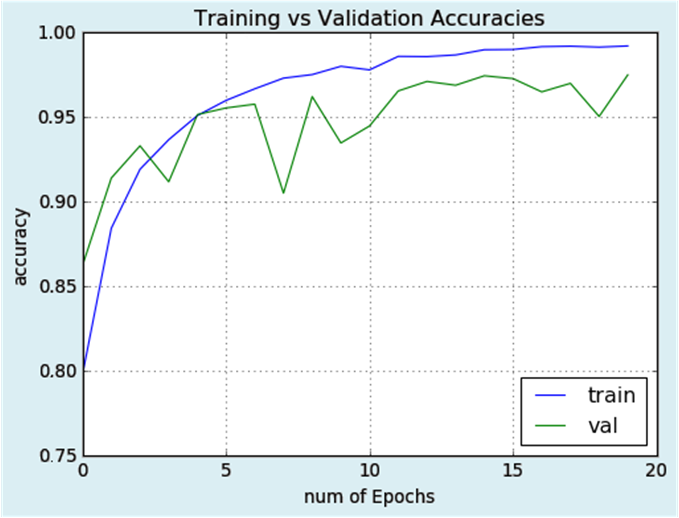
### Raw Difference Purity Classifier



A confusion matrix was developed for the purity classifier model. This matrix displayed a 98% true positive accuracy (true safe prediction), and a 2% false positive accuracy (false safe prediction). The returned false positive accuracy was especially impressive, as falsely predicting safe would be dangerous.

Next, the Matthews Correlation Coefficient (MCC) was calculated for our model. The MCC value is used to measure the quality of a binary classifier. It takes into account true and false positive values and is regarded as a qualifier that can be used even with uneven class sizes. It returns a value between +1 and -1, where +1 represents a “perfect prediction”, 0 represents a random prediction, and -1 represents a complete disagreement between prediction and observation. The implementation of this technique returned a MCC value of 0.946, representing an almost “perfect” prediction.



Another performance metric used to analyze our model is *k*-fold cross validation, in which the original sample size is randomly partitioned into *k* equal sized samples, and a single subsample is used to represent the training data. The training data is rotated, and trained on the 1 subsample left as validation. In the case of Impurivision the dataset was split into 10 different sets, and run ten times. 

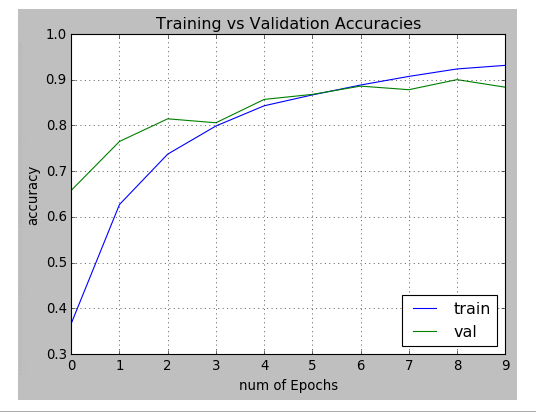
Lastly, the training and testing accuracies were analyzed to make sure the model wasn’t overfitting. The graph on the right shows converging accuracies, as well as negligible overfitting.

### 

### 

### 

### Raw Difference Substance Classifier

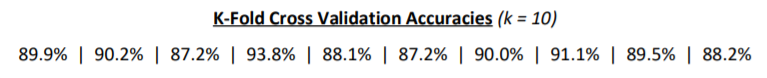
A CNN model was also developed to predict substances that might be in the users water. There were 15 classes (one for each substance) that were included in the prediction. The trained model proved to be verifiably effective for the 15 way classification task. The model’s training and validation accuracies converged quickly, after 10 epochs, proving that overfitting was negligible.

Shown is the distribution of accuracies for individual substances. Substances like pesticides and sodium had lower accuracies because they looked very similar to each other, and the model was occasionally confused between the two. For example, it was found that 71% of substances that were wrongly classified as sodium were actually pesticide.

Below is the distribution of individual substance accuracies for the raw difference substance classifier:

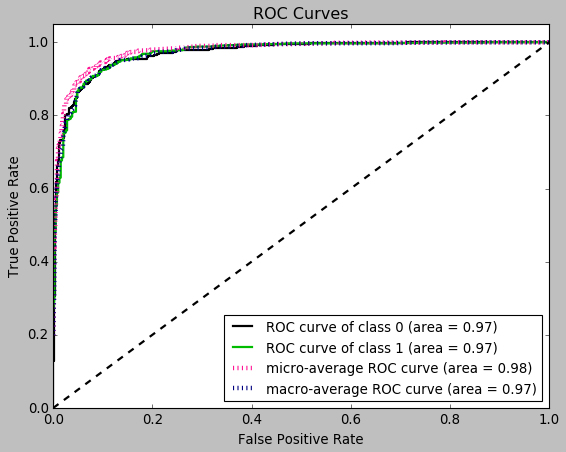
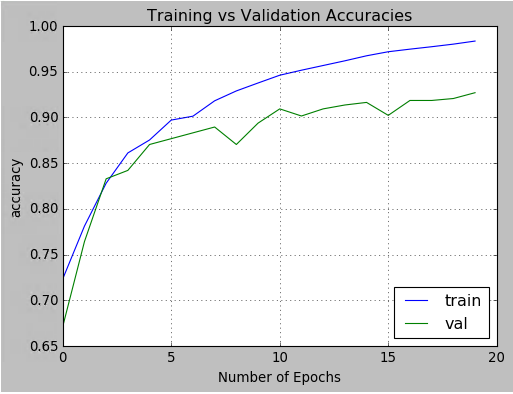


The following are the results of *k*-fold cross validation performed on the substance classifier:

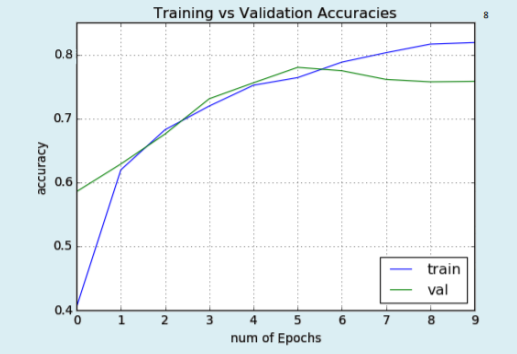


### Filtered Difference Classifiers

Although the filtered difference image dataset was created to correct for possible convolutional neural network error due to RGB underflow, flooring the negative RGB components was shown to adversely affect accuracy. Below are plots for the filtered purity classifier:



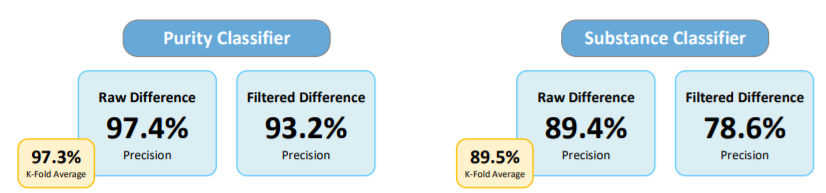
As well as for the filtered substance classifier:



# 

# Discussion

## Results

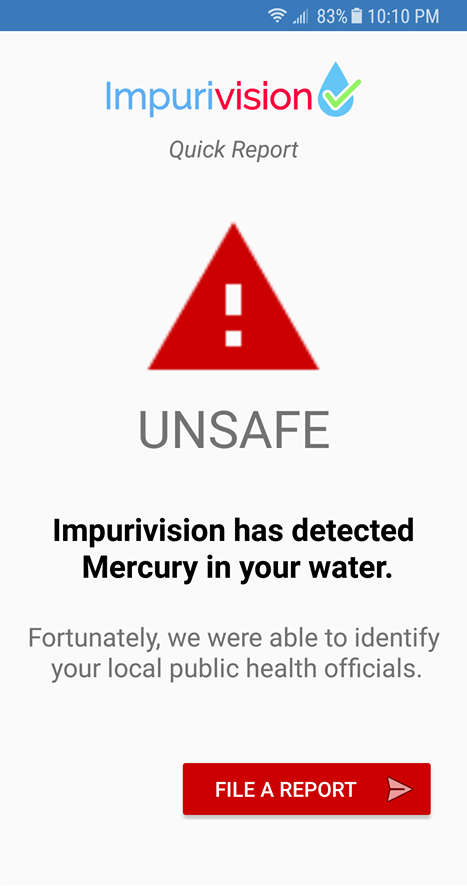
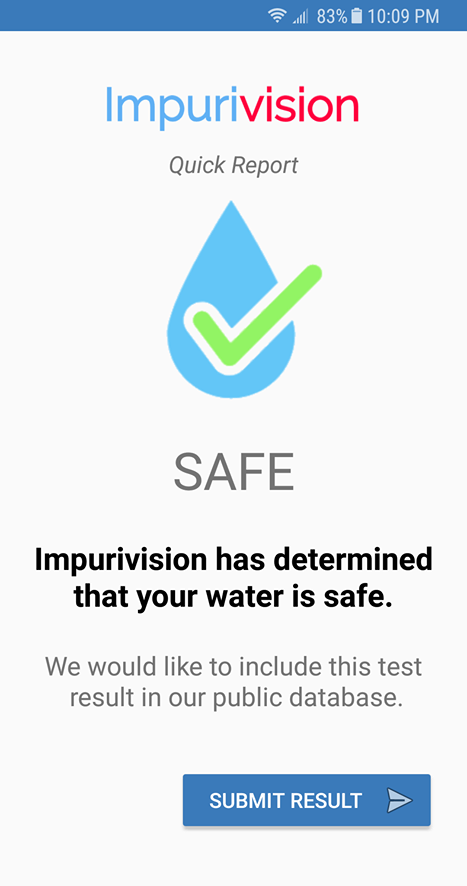
**

A unique approach for water impurity detection was developed: training a convolutional neural network on thousands of images of water with trace concentrations of contaminants, and integrating the CNN into a mobile application to be used with user-provided water images. The CNN developed for this project proved to be verifiably effective as it was able to achieve an accuracy of 97% at evaluating water potability and 89% at identifying specific contaminants and contaminant groups. More specifically, the safe/unsafe classifier identified contaminated water 98% of the time and clean water 97% of the time, while the substance classifier had individual substance accuracies ranging from 69% to 100%. It was found that filtered classifiers were much less effective compared to their raw counterparts. This was determined to be a result of the loss of important information. As such, these positive results establish the viability of the Impurivision app for real-world purposes. App-integrated deep learning is demonstrably superior to current methods, which primarily rely on expensive physical equipment and human analysis.

## 

## Impacts

Two additional features were added to the Impurivision app to provide it with a more powerful real-world applicability: *Quick Report*, which prompts the user to report possible water contamination to local public health officials, and the *Results Map*, which plots each water test result on a publicly accessible map.

Quick Report is active upon completion of water analysis by the CNN. The page shows the results of the safe/unsafe classifier, and if the water was determined to be the latter, the substances detected by the substance classifier as well. A determination of safe will allow the user to submit the results to a database for public access. A determination of unsafe will prompt the user to submit the results, and file a report with local public health officials if they are able to be identified. Thus, Quick Report aggregates all test results while reporting detected contamination to the appropriate authorities.



(Developed by Kartik & Palash)

The database is utilized by the Results Map, which plots each test at the site where it was conducted, and with a symbol representing the results. Additionally, hues indicate the accuracy of the tests; multiple safe determinations across different households result in an icon with bolder shades of green, while repetitive unsafe determinations result in stronger shades of orange. When contamination reports are confirmed by officials or independent organizations, the icon changes to dark red, warning individuals that the water in the area is verifiably unsafe to drink.

Because of its accurate algorithms and useful features, the Impurivision app carries immense potential to translate robust accuracies into real-world benefits.

# 

# Conclusions

## Findings

1. The Impurivision app is capable of producing difference images of sufficient quality, in spite of optimization (compression) techniques, allowing for a deep learning approach to water purity analysis
2. The convolutional neural network utilized by Impurivision is extremely accurate at purity evaluation and highly accurate at substance identification. This finding is corroborated by several validation techniques.
3. The filtration algorithms developed removed valuable information that could be used for the classification task.
4. Due to the use of major contaminants and contaminant mixtures prevalent in diverse communities, Impurivision is able to meet its objective of detecting for the majority of water contamination around the globe (based on an aggregation of contamination data from multiple sources)
5. Additional features in the mobile application dramatically enhance its potential for real-world use

## Future Improvements

* Publish app on the Android app store for use around the globe
  + Partner with municipalities, NGOs, and researchers to raise awareness
* Develop text-to-report system to reach cell users lacking stable Internet
  + Since many individuals in underdeveloped countries lack access to internet, but have cellular connection, this system will allow users can send images to a phone number, and receive an analysis in return.
* Build a larger, more robust dataset that accounts for more information
  + Increase the number of substances and groups, and curtail the use of stimulants
  + Produce thousands more images using generative adversarial neural networks through unsupervised clustering.

# References

Research Sources

1. “Aluminium in Drinking-water.” *World Health Organization*. <http://apps.who.int/iris/bitstream/10665/75362/1/WHO_SDE_WSH_03.04_53_eng.pdf>
2. “Android.graphics.Color.” *Android Developers*, 3 Apr. 2018, <http://developer.android.com/reference/android/graphics/Color.html>.
3. “Cadmium in Drinking-water.” *World Health Organization*. [http://www.who.int/water\_sanitation\_health/dwq/chemicals/llarcadmium.pdf](http://www.who.int/water_sanitation_health/dwq/chemicals/cadmium.pdf)
4. “Chloride in Drinking-water.” *World Health Organization*. <http://www.who.int/water_sanitation_health/dwq/chloride.pdf>
5. “Color” *Android Developers*.  
   <https://developer.android.com/reference/android/graphics/Color.html>
6. “Common Hidden Contaminants.” *Water Quality Association*, <http://www.wqa.org/learn-about-water/common-contaminants>.
7. *CS231n Convolutional Neural Networks for Visual Recognition*, <http://cs231n.github.io/convolutional-networks/>.
8. Lettier, David. “You Need to Know about the Matthews Correlation Coefficient by David Lettier.” *Lettier*, <http://lettier.github.io/posts/2016-08-05-matthews-correlation-coefficient.html>.
9. “Monitoring Arsenic in Water.” *United Nations Children's Fund*. <https://www.unicef.org/supply/files/Monitoring_Arsenic_in_Water.pdf>
10. “Mobile Phone Growth Exploding in Developing Nations.” *Mobile Beyond*. <https://www.mobilebeyond.net/mobile-phone-growth-exploding-in-developing-nations/>
11. “Planet Is Running out of Clean Water, New Film Warns.” *CNN*, Cable News Network, www.cnn.com/2008/TECH/science/09/19/water.crisis/index.html.
12. “Sodium and Chloride in Drinking Water.” *New Hampshire Department of Environmental Services*, 2010, <https://www.des.nh.gov/organization/commissioner/pip/factsheets/dwgb/documents/dwgb-3-17.pdf>
13. S.V. Panno, K.C. Hackley, H.H. Hwang, S. Greenberg, I.G. Krapac, S. Landsberger and D.J. O’Kelly. “Source Identification of Sodium and Chloride Contamination in Natural Waters: Preliminary Results.” *Water Research Center* <http://www.water-research.net/Waterlibrary/privatewell/nacl.pdf>
14. Thomson, Lowell. “Adding Water to Anhydrous Copper II Sulfate.” *YouTube*. <https://www.youtube.com/watch?v=UjmcHP7WFTA>
15. “Unsafe Lead Levels in Tap Water Not Limited to Flint.” *New York Times*, 8 Feb. 2016, <https://www.nytimes.com/2016/02/09/us/regulatory-gaps-leave-unsafe-lead-levels-in-water-nationwide.html?_r=0>
16. Ziordano, Robert. “Design215.Com Menu.” *Design215 Megapixels Comparison and Maximum Print Size Charts*

Image Citations

1. Developed by Kartik & Palash
2. “Worried About Lead in Your Water? Here's a Solution That's Working in Flint.” *The Fiscal Times*, www.thefiscaltimes.com/2016/02/15/Worried-About-Lead-Your-Water-Here-s-Solution-s-Working-Flint.