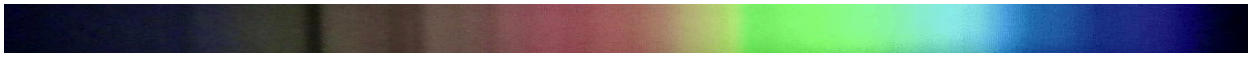
DIY Spectroscopy with Assessment for Food Identification and Quality Control Tasks (Draft)

Joshua Brewster March 22, 2024

Project Sponsors:

Schmidt Marine Org, San Francisco, CA
Newport Aquarium and OSU, Newport OR
Yassine Santissi @ Fishazam LLC.



Contents:

Abstract	1
Repositories	2
Background	2
General Design Considerations	5
Spectrometer Design	6
Camera Selection	10
Software Design	12
Test Setup	17
Spectral Classification Test Results	17
Comparing Results With Previous Work	20
Important Design Refinement Considerations	23
Hyperspectral Imaging Test	24
Conclusion	27
Citations	28

Abstract:

This paper demonstrates the development of an accessible, handheld mobile spectroscopy device designed for students and entry level use, aimed at democratizing the collection of reliable spectral image data for scientific analysis. Leveraging machine learning, in our case XGBoost for spectral classification, this project offers an affordable entry point into spectroscopy for students and citizen scientists facing budget constraints. We describe promising applications in industrial and regulatory settings, particularly in substance identification and quality assurance tasks such as determining the freshness, ripeness, and presence of contaminants in food products. Our preliminary classification study addresses the widespread issue of fish filet mislabeling, which is estimated to affect nearly half of all fish sold in the market. Through the creation of a comprehensive 3D printable kit and web software pipeline, using off the shelf components and supplemented with instructional materials for dataset training, this initiative has produced preliminary data that aligns with existing research, confirming the potential of cost-effective, off-the-shelf tools in addressing significant regulatory and conservation challenges in the food industry. Additionally, the project outlines a pathway towards the development of more advanced, yet still low-cost, spectroscopy solutions, based on insights gained from this endeavor.

Repositories:

3D printable Digital Spectroscope with assembly manuals and demo:
https://github.com/joshbrew/Digital-Spectroscope

Spectral & Camera Identification demo:
https://github.com/joshbrew/cameraId-wonnx-wasm

XGBoost Python Notebooks:
https://github.com/joshbrew/XGBoost_ONNX_Training_Conversion

Puny Pi Cortex A53 Module Drafts:
https://github.com/joshbrew/PunyPi

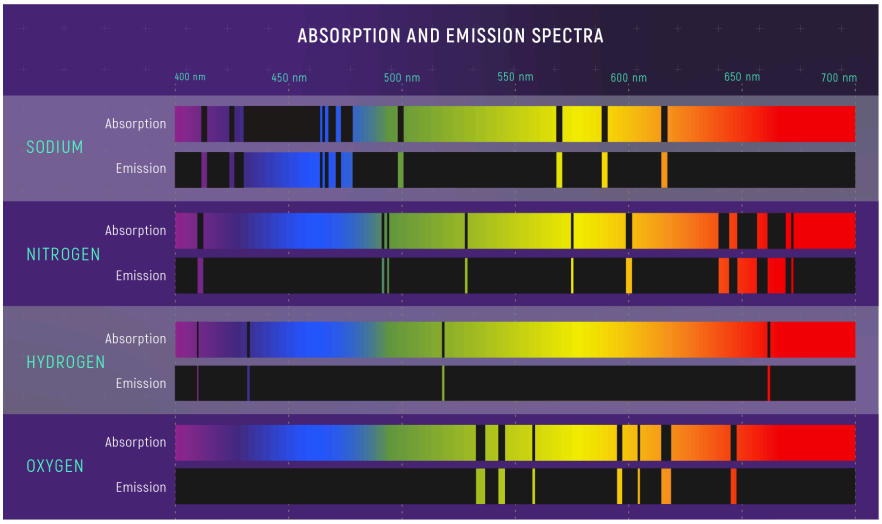
Background:

Spectroscopy is the science of analyzing different wavelengths of light as they are absorbed and reflected by molecules in the substances being identified. This branch of analysis is known as chemometrics, which involves a whole host of tried and true statistical and multivariate methods to pull apart the information. It is used in astronomy, chemistry, materials science, food and drug safety, and more as a ubiquitous research and analysis tool. There are many kinds of spectrometers, some use sophisticated lasers and high speed optical sampling chips, others might shoot radiation at the substance to image the result.

We chose the most low cost and feasible method, known as optical or reflectance spectroscopy, and we are measuring the absorption of light in a material between UV around 350nm and Near Infrared around 1100nm, resulting in a unique profile based on the molecular makeup of that material. While it might be a narrower band than an Interferometer-based Fourier-Transform Spectroscope which typically look from UV starting at 100nm to Far Infrared up to around 10000nm, the precision and uniqueness within that band is all that really matters, and more details will pop out as you look at finer gradations in the wavelengths of light. Our method is fairly low fidelity with what cheap camera sensors we could find off the shelf, but already shows promise in precise identification tasks.

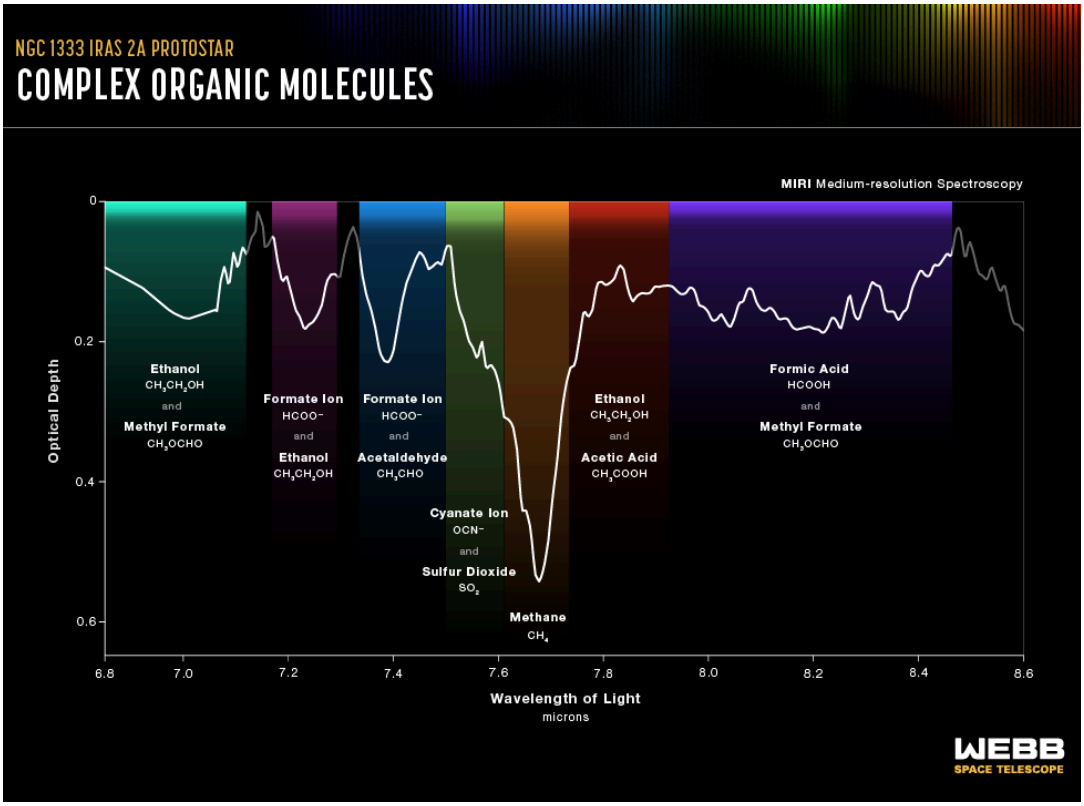
Our project centers around separately identifying different fish where the filets may be indistinguishable at a glance. This lets us limit substitution, and can potentially go much further with freshness and other quality or contaminant markers with larger datasets, meaning the food industry could find itself with a low cost tool to add more valuable data into the process control. This is important as our food systems continue to expand and enforcement gets harder. From data shown by Bartek Rajwa at Purdue, you can even judge, say, which side of the Alps a type of cheese came from, and the lowest end devices he tested (still hundreds of dollars) are able to do it. [2]

When taking images, a rainbow appears like in the diagram below, which is an amplitude spectrum of the light being diffracted into a constructive interference pattern. This image then is the resulting concentration gradient of whatever substance or gas etc. being imaged after being lit up by something, creating an absorption spectrum unique to the molecular profile of that substance. In astronomy for example, we use starlight to image gas clouds or transiting planets in front of them, and use known differences to tell what nebulas, protostars, and planets are made of. In material science we can determine precisely what something is made of. An emission spectrum otherwise is the result of direct light from a substance.



Various absorption and emission spectra.
Source: Webb Space Telescope

For complex organic molecules, which is our concern, rather than precise lines you will see a concentration gradient, which is less precisely identifiable but still affords patterns that a statistical program can pick apart if you know what you are looking for. This is used for quality control in foods or medicine and can identify things like contaminants or authenticity and origin.



A complex concentration gradient of a protostar system.
Source: Webb Space Telescope

This makes our task of identification more difficult but with a little elbow polish we can get promising results with DIY off-the-shelf parts with minimal system requirements and only a few minutes of setup.

Access to professional hardware varies. For low cost you might find yourself not satisfied and only able to do cursory explorations. On the DIY side, Thunder Optics sells a cardboard cutout spectrometer for $20. There are cheap paper tubes for classrooms. Carolina ChemKits sells a pack of 10 plastic spectrometers and slides for $90 for classrooms which is a great deal. There are generic desktop spectrometers for $200-$400. The average professional spectroscope is more than $500. GoSpectro sells a professional phone spectrometer for $525. The cost quickly grows from there. Thorlabs offers their CCS100 for $2300. Wasatch, Oto, Edmund spectrometers and so on approach the $3000-$10000 range once you include accessories. We found after looking at how every type of design was made that there could be a sweet spot on affordability, DIY-ability, and actual real world applicability with some smart camera selection and a 3D printable, adjustable frame.

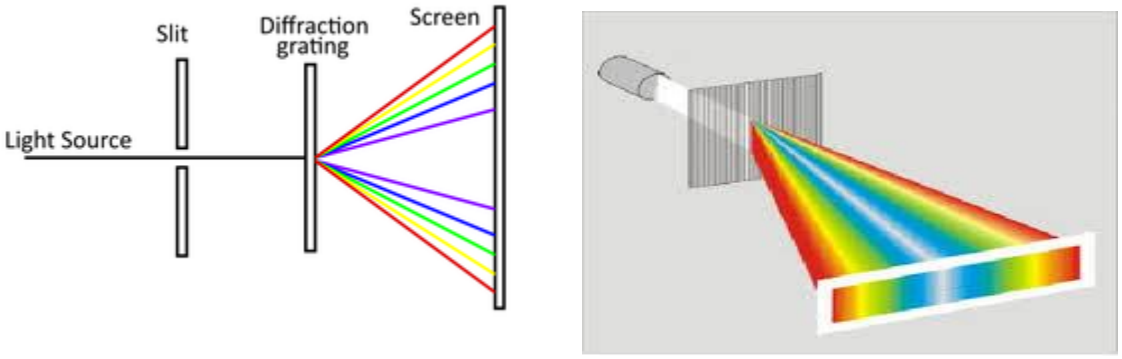
Software support varies as well. Most software tools are designed for the particular brands, and if they offer APIs it's usually fairly limited or vendor-locked and involves a lot of writing your own python or matlab code, which doesn't necessarily scale well to many users. ENLIGHTEN by Wasatch for example is free and has a free trial for an IDing library of 25000 substances but is meant for Wasatch products that cost as much as a car. We found

much larger public datasets that can be scraped from the web for tens of thousands of scanned substances. NIST, USGS, many universities and so on offer large libraries for substance IDing with spectroscopy that can be parsed with a little programming but the formats or, say, the way you can pull these datasets are not on a common standard or accessible in a user-friendly way for interfacing with our tools. There are various free software tools on github with limited applicability, e.g. the odd spectral viewer here and there for making charts. There are also a handful of Android and IOS apps. Vernier Spectral Analysis is free but analyzes a specific file format meant for their devices. The pickings are simply slim and don't offer a robust solution to both collect data and run analysis on the fly. We know how to solve this lack with free modern high performance web programming, and leave others something that is scalable and more modifiable by a wider audience.

We addressed all of these issues with practical spectroscopy in a particularly challenging task: fish filet identification. Food and biological materials are complex and require a softer approach to statistical analysis. We took a modern algorithm for working with this kind of data known as XGBoost, or "eXtreme Gradient Boosting" and showed promising results without fine tuning an off the shelf camera set and on a fairly limited dataset. This lowers the barrier to entry significantly for approaching important tasks like fraud control, where nearly half of filets are thought to be mislabelled even in the United States [3].

General Design Considerations:

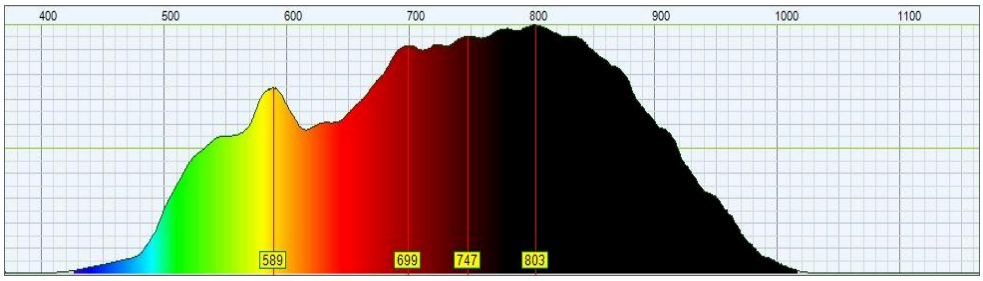
Building a DIY spectrometer is very straightforward, but perfecting it with off the shelf parts is not trivial. We've managed to design working devices that can start being used at home, with caveats, while more work needs to be done to perfect the specs while maintaining its off-the-shelf feasibility.



A basic spectroscope

The basic components of a digital spectrometer are a camera or light sensor array, a diffraction grating which acts like a prism, a narrow slit to allow light in, and a light source. Simple spectroscopes use a tube that is long enough to make the slit appear narrower to the diffraction grating without machine precision sub-millimeter slits. You can achieve this with something as minimal as thick cardboard and tape. We created several 3D

printable, adjustable versions using breakaway blades and levers held by screws for sub-millimeter adjustments that are low profile and hand-tunable. They allow the use of PCB Arducams or your own mobile phone on an adjustable, mountable frame. Prior DIY work showed that a typical HD webcam spectrometer could easily achieve a resolution of **1 nanometer**. [1]



Digital spectrogram of an Incandescent light. Blue to Infrared.
Source: Theremino Spectrometer Tutorial

The chief considerations for both hardware *and software* for us are:

- Ease of Use
- Plug and Play
- Mobility
- Modularity
- Adjustability
- Precision

This leaves us with a probable blueprint for future advances of deployable low cost spectroscopy.

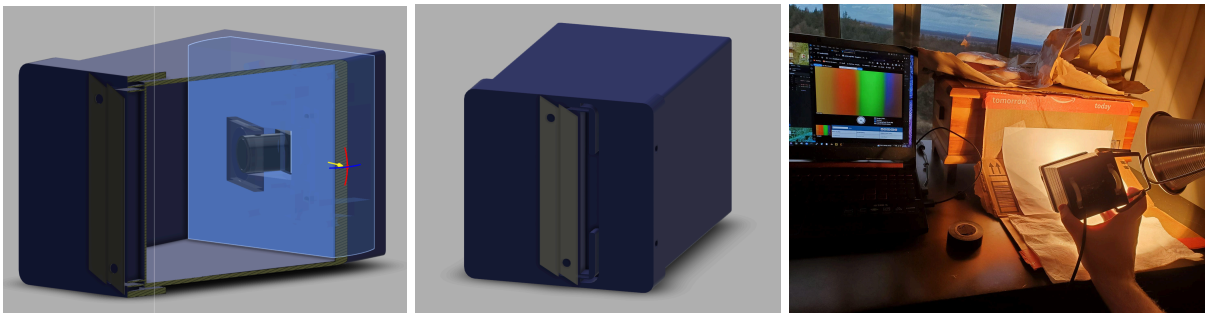
Spectrometer Design:

Our spectrometer designs are do-it-yourself 3D printable designs with off the shelf parts easily sourced from amazon or other online catalogs. It lets you experiment with different PCB cameras and phones with adjustable parts, and you can set your own precision on the light slit using a clever breakaway blade setup with levers you can tighten with screws for fine adjustments. Cameras are limited to 3 wavelengths but we can glean more information using the diffraction grating to assess the full available spectrum according to the Bayer chart. It is highly experimental and we are still settling on the final recommended specs for the best version of this design, but you can build this off the shelf using any similar parts you can get ahold of and it will still give you usable results. We made many key considerations in our designs so they would add value and ease-of-use to the scientific and education community as a deployable do-it-yourself system.

We started designing rough materials, tape, a box, and the first webcam we could find which was a clunky 720p wifi security camera.

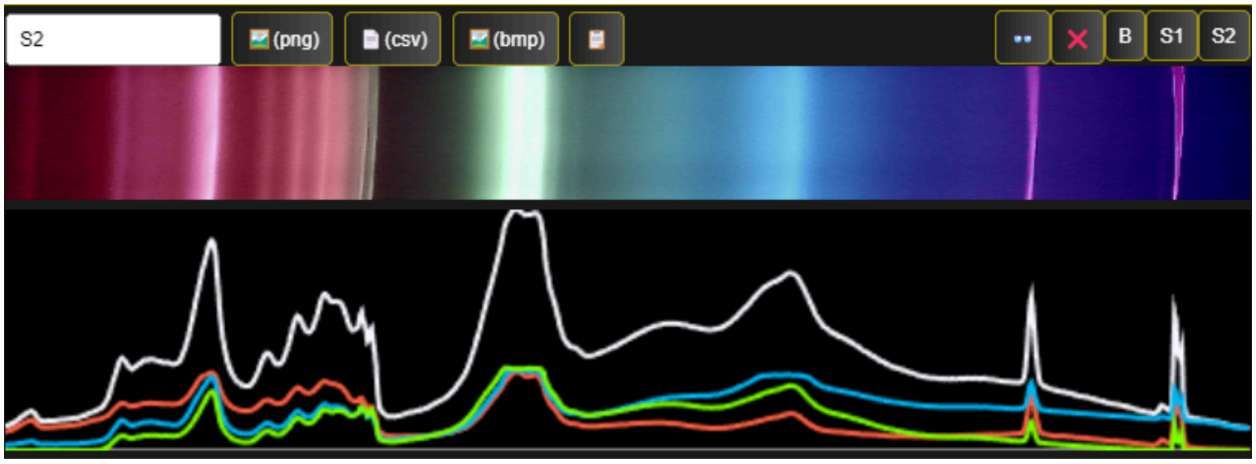
But a pretty decent result! The spectrum on the right is sunlight through my window.

The taped up prototypes were handed off to Bernard Markus from the Netherlands who designed the first iteration based on a foam poster board-based box that we sent to him, which was made based on an old online tutorial we remembered from high school. It was designed to be simple and adjustable so we could fine tune the image before locking in a spec. We used an OV2640-based Arducam for the design, which is compatible with all other Arducam models.



V1 3D model slice, with it in action imaging fish using a desk lamp.

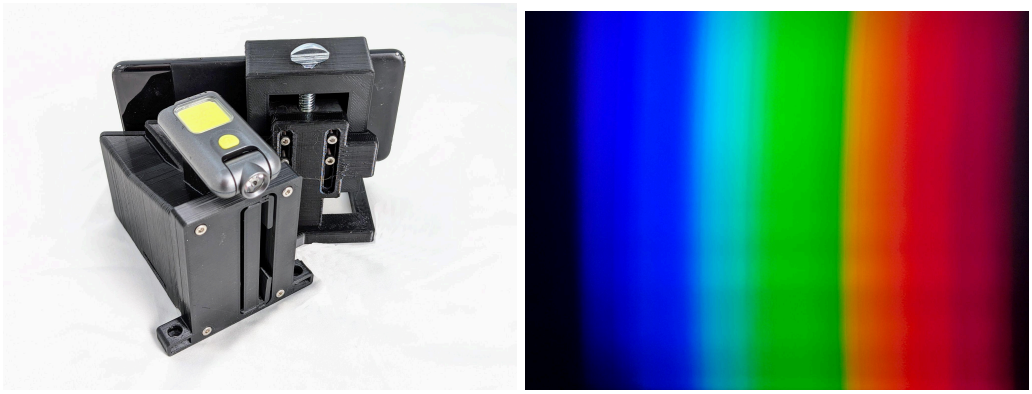
The output with the OV2640 is great after some fine adjustment to the light entry slit. Here is the output for a fluorescent bulb.



Fluorescent light emission spectra with OV2640, slit size tune to <0.1mm

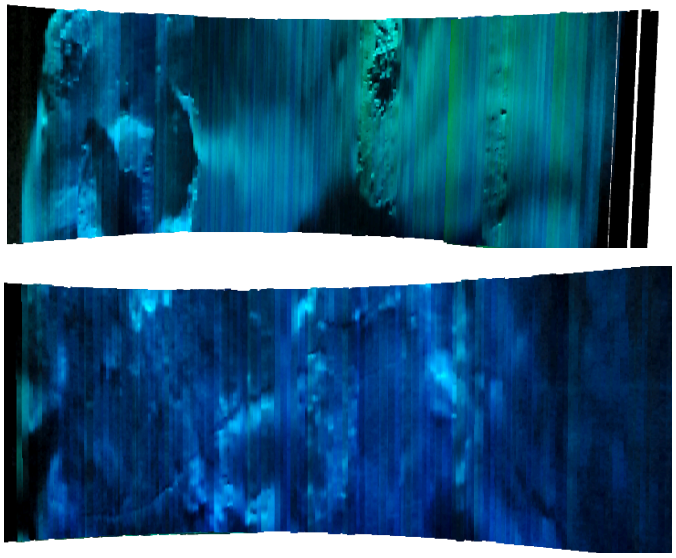
The next designer, Davis Fay out of Colorado, took Bernard's prototype and refined it so it was a better handheld device with a more minimal profile. It added some quality of life features like a cap for isolating the slit, a mount for the light source, an easier frame to mount the PCB camera into with a fixed angle, screw holes for a tripod or gimbal mount, and finally a version made for using with a cell phone. It was a natural adaptation and we are pleased with the results: it looks sleek, it's easy to assemble, low cost, and comes with detailed assembly manuals with diagrams. The slit precision is controllable with screws and allows for matching precision with machine-cut slits at widths less than 100 micrometers. The breakaway blades Bernard chose are very straight and cost pennies.

V2 PCB camera design. The cone allows for a 20 degree field of view lens to fill the image with the spectrogram.



Mobile phone spectrometer design and result with a Samsung Galaxy S10e

We were pleased to find that cell phones take just as good if not better images than the typical PCB camera due to the very high resolution and sophisticated light and color compensation built in, which we can readily take advantage of in our own programs. It showed sharp and subtle absorption lines. The wavelength is generally limited to only the visible range as we cannot modify the lens filters, however. We found it is likely perfectly viable for spectroscopy tasks, but also it unlocks a convenient method for a more advanced type of spectroscopy called Hyperspectral Imaging.



Slice of Limestone (top) versus a chunk of agate (bottom) in a blue-green wavelength. These are hand motion-captured images.

Perhaps the most exciting prospect for the smartphone-based spectroscopy is integrated motion sensing support for creating hyperspectral scans from the raw spectrum shown prior. The images shown are hand-scanned so precision is about as good as our hand movement, but still gives a great result. We found our inspiration from a DIY Smartphone Hyperspectral Imaging paper from 2022 [4] and another DIY hyperspectral paper from 2016 [5], which showed that this kind of design was viable for studying things like peak ripeness in fruit, and gave comparable results to industrial hyperspectral cameras that get into the tens of thousands of dollars very quickly. They used very clunky setups though, while ours works fully handheld and uses generalized software.

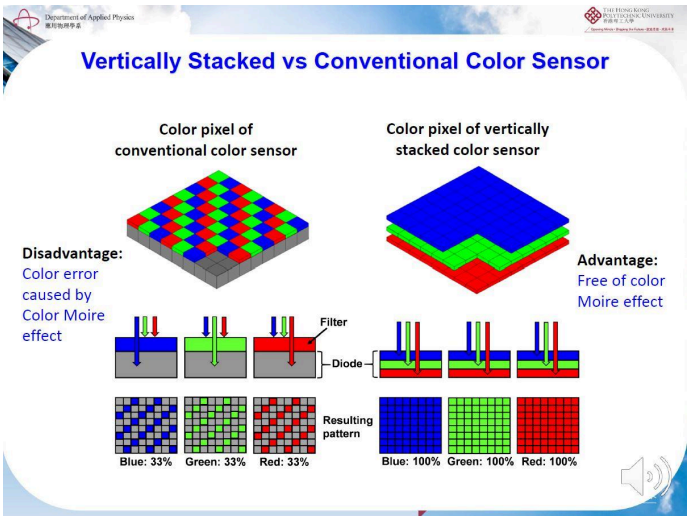
The USB PCB cameras can be made into remote devices as well using single board computers with an internet connection. We tested combining a Raspberry Pi 4 Model B with 4gb of RAM and Apache Guacamole, which gives us a remote desktop through the browser available at a URL on a network. This can then host a fullscreened version of our analysis application running entirely on the small single board computer over the internet. With WiFi or a local area connection this demonstrates a kind of automated remote monitoring system you'd find in big league industrial process control. This is a rough example for how to scale this kind of solution effectively. We are also experimenting with a custom Cortex A53 module with a low cost graphical processor as an entirely self contained automated testing solution with its own operating system and access via tools like Guacamole.

Another alternative is that Android allows you to plug in USB cameras through available ports on phones or tablets, e.g. with a USB-A to USB-C converter. We were able to use our Android S10e plugged to the Arducam model to run the software, and even use the phone's motion sensors in conjunction with moving the camera to capture hyperspectral images. Altogether, this supplies many possible approaches for hardware utilization based

on project constraints. We included a list toward the end of this paper on design considerations for refining these devices from an off the shelf toy into a respectable scientific tool.

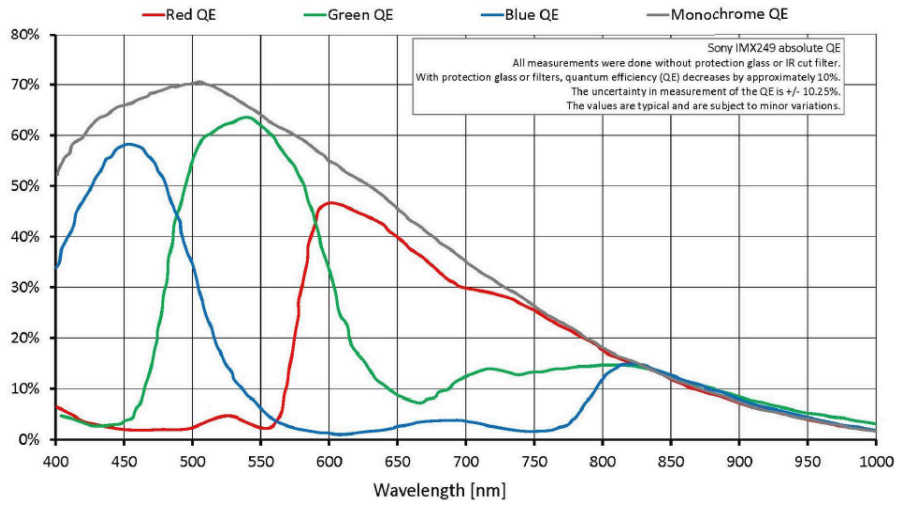
Camera Selection:

Digital cameras are composed of a grid of photodetectors with 3 different color sensitive sensors for Red, Green, and Blue light. The filters are bandpassed for a wide set of wavelengths across the visible spectrum, giving us sensitivity to some UV light up to Near Infrared. Typical digital cameras are only sensitive from partial UV to Near Infrared, or from 350nm to about 1100nm, but with less sensitivity at either end. This is because of the filters used to isolate the visible spectrum on a standard camera.

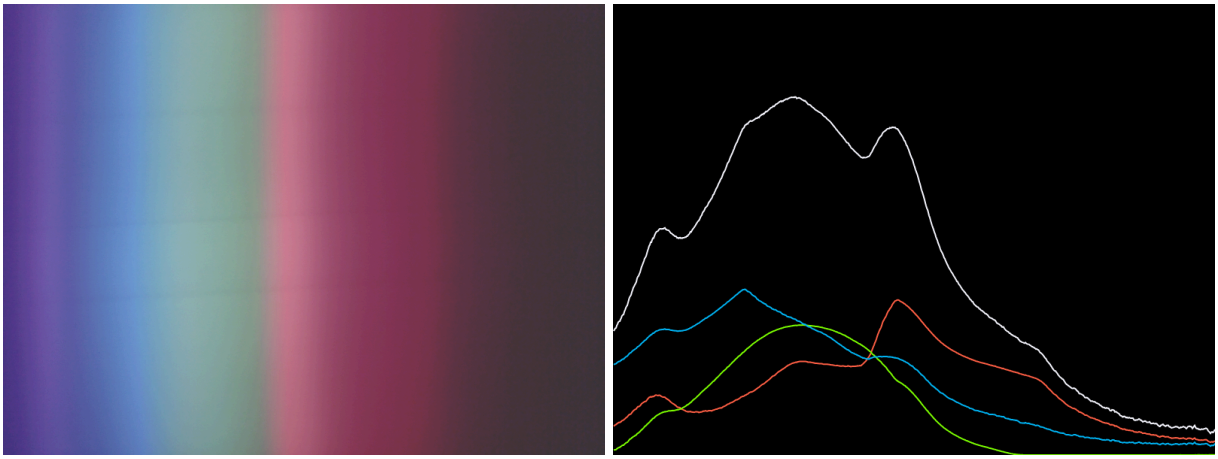


Bayer vs Perovskite Sensors.
Source: Hong Kong Polytechnic Dept of Applied Physics

Camera sensors are either “Bayer” grids of photodetectors or “Perovskite” layered color-sensitive diodes. Our current models are cheaper Bayer grid type cameras. The size of each individual pixel (a 2x2 section of the Bayer grid) and the camera imaging processing will limit the narrowness of the bands that you can sample across your color spectrum and may completely change measurements camera to camera.

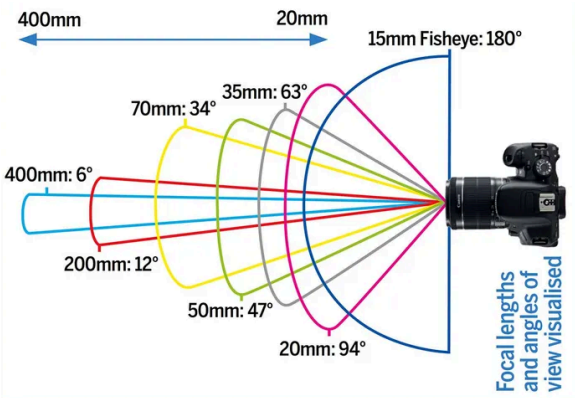


Typical Bayer Filter spectral sensitivities. Source: Sony



Spectrum we created from an averaged image of a continuous color LED using our OV2640 camera.

Notice that in our own raw image the normalized RGB channel sensitivities shown right are relatively matching to the prior Bayer filter spectral chart from Sony. Differences are based on manufacturing, camera settings, and the light source itself which has uneven intensities. The spectral image on the left is uncropped from our camera based on the angle and zoom, which was tuned to give us a full image that makes use of all of the pixels and required the least preprocessing, meaning a much better plug-and-play experience.



Camera used: We modified a night vision OV2640 model with a 20 degree lens and removed the night vision LEDs and IR-Cut filter. It has some of the best presets programmed in of the half dozen or so off the shelf PCB cameras we tried but has to be modified.

Lens diagram source: Digital Camera World

Standard settings like white and color balance are also important to control, some cameras automatically compensate in low light levels which can change your results if you are not paying attention, or you need to expand your dataset to include different combinations of images from different camera sensors with different settings. We recommend just sticking to one model that gives the best results to leave the least amount of question. There has been extensive work done previously covering spectroscopic capabilities in cameras.

So far we've been using OV2640 off the shelf, a Bayer filter model. It is obsolete but still manufactured, however with more time and money it would be best to consult an expert in spectroscopy and camera sensing to identify the easiest and best components to mount and get the finest detailed results we can under 100 dollars. Comparing our results with other spectrometers, ours are comparable in binary classification scenarios, and still satisfactory in multiclass scenarios, which could be improved with a lot more data. We are still identifying camera models and settings to use that keep the device costs around 50-100 dollars, though more expensive components could be swapped in for precision reasons.

Software design:

Web Software Design and Overview

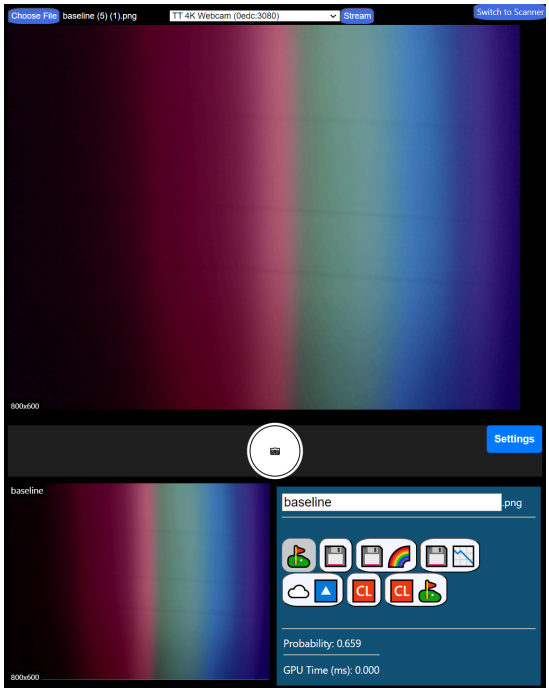
We needed to build our own software from scratch in order to create an easy tool for quick capture, export, and classification of data. This makes for an easy one-click system for ID'ing images with existing models or gathering data for creating and refining new ones. We were able to accomplish this entirely with the freely available tools in python and browser javascript. It runs in real time, has compact app sizes, and classification is instantaneous thanks to some new tools that lets us export trained models into a browser GPU runtime.

The architecture of the software is based on common Web and machine learning standards. This is a combination of many tools in a streamlined fashion:

- Node.js for app bundling and serving.
- HTML5, CSS, and Javascript, with a Typescript framework for developing
- Open Neural Network Exchange (ONNX) WebGPU-based machine learning classification framework, with fallbacks to WebGL and SIMD for multi platform support. Runs entirely client-side.
- Pytorch with user-friendly Jupyter notebook-based execution for training and testing on collected datasets. Results are compiled into the ONNX format for compact applications.
- Simple cloud backup options to your personal Google Drive account, which can be swapped with other cloud backends for your needs.
- Lightweight browser, desktop, and mobile cross platform distribution framework.

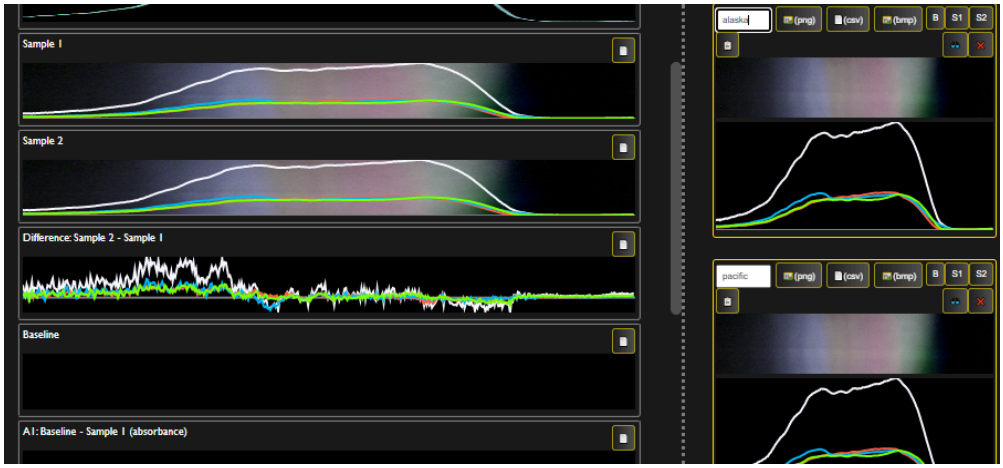
The app is simple: connect the camera, open the app and select it from the dropdown, and start imaging. It's meant to be user friendly and make it easy to gather data and get quick results. This makes it more appropriate for education as well as a model for painless deployment in the field for future scalability. We are strong advocates for modern web development due to its modern status as a community-owned high performance cross platform application framework. Web doesn't just mean online anymore, it's evolved into a general software standard. This format saves everyone time and money, and lowers the barrier to entry for all kinds of future creativity and contributions. This is imperative for moving academic research forward quicker, with more people able to have a collaborative commons for developing real world use cases.

For spectral imaging, first we take a baseline image. We used a white sheet of paper to get a good full spectrum image of the reflected light for the camera sensor, which will show us more or less the spectral curve of the camera itself based on its filter sensitivity. Our software automatically averages 10 frames off the video feed which will give a naturally smoothed result, where the pixels themselves have random noise we need to compensate for. Now we store this baseline image in local memory for reference by the program using the red flag icon in the menu in the next image. The app also lets you subdivide the image if you aren't using the full camera view. We also want to save the CSV for this image for training and reproducibility. Click the save icons to store data on your machine. You can hover over buttons to read what they do.



Baseline image from white paper using the OV2640 Arducam, with correct classification. It's so fast the timer didn't register it.

After that we take a picture of a fish which will subtract the fish intensities off the baseline intensities, giving us an absorbance spectrum. The machine learning model will attempt to guess what it is based on the labels supplied to it. Benchmarking the WebGPU-based machine learning system to return guesses within 200 nanoseconds to 1ms on average, even on mobile. This means it can process potentially thousands of classes per second on a single user's device. For data collection, however, next we can save both the raw subtracted image and the CSV of the RGB and summed color intensities along the x-axis of the image, by counting up each column of pixels then normalizing the results to unit scale (-1 to 1). The app was pre-set in our test to convert images to an 800x600 format, which then reduces to a CSV with 4 columns and 800 rows for each pixel color component and intensity. This can be set to any resolution your camera supports but we balanced it for performance and battery life.



An older version of the app directly comparing Atlantic and Pacific Cod using an incandescent bulb. Note the curves look identical without correction, but subtracting shows differences.

Each time we take a picture with the app's default settings, it will give you an averaged, baseline-subtracted image. The averaging controls for sensor noise, and the subtraction removes the norm which helps make the results more differentiable. This method assumes the image did not shift in the view. The shifting won't happen on the fixed Arducam setup if screwed in properly. However, modern mobile phone cameras can have some motion compensation built into the cameras, so motion can change the position of the image even if the phone does not shift positions relative to the imaging box.

To summarize the data collection procedure:

- Connect your spectrometer to the application, the camera selection is the same for either the USB camera or mobile camera.
- Adjust color balance levels if needed, on mobile adjust zoom so the image is filled with the spectrum.
- Turn on the flashlight on the spectrometer, use a neutral white surface like a sheet of printer paper to take your first averaged image.
- Click the baseline button to save this image in memory for correcting your next samples.
- Now image your substances - fish in our case - and save the raw image plus the corrected CSV. The raw image will be useful for reproducing results, while the CSVs are the digital absorbance spectrum. Results are normalized relative to the baseline.
- Take several images of each substance, more images is more data. When making use of the classifiers built into the app, you can also average guesses to increase confidence in your results.
- Store images in subfolders named for each class.

Storing all of the CSVs of our corrected samples, including the raw data from the baseline to give us something good to compare to, next we move our dataset into training a classifier. We took about 25 images of each fish we wanted to identify and separated a random selection of those samples into a test set of about 4 samples each. The test set does not get trained on so it can replicate an authentic guess within the program.

XGBoost and ONNX notebooks

XGBoost stands for eXtreme Gradient Boosting. It is a relatively new, award-winning regression technique that generates complex decision trees using a kitchen sink of heuristic methods. This generates functions that represent the bulk of data for each class, which can have unintuitive relationships that these statistical algorithms can find often better than humans. XGBoost is not a neural network but is in the same family.

The Open Neural Network Exchange (ONNX) is a specification that allows neural networks and other kinds of models to be compiled from various environments like PyTorch and Tensorflow into a binary that contains a static version of that model and the weights from training. It can do more than that, but for our purposes we can export from a very bulky environment like Python that requires several gigabytes of dependencies to run PyTorch and other tools into a compact file on the order of kilobytes or megabytes, and then run in a small program through a web environment, which is already supported by all operating systems.

The reason we chose XGBoost is for its compactness and ease of use, while a neural network might only afford marginal improvements at the cost of a lot more processing overhead. An engineer, Mateusz Stankiewicz out of Poland, created our first implementation with off the shelf libraries. On our models trained on a very small sample size (n=~100) the compiled ONNX file with the trained XGBoost program is around 40kb. This will increase relative to the number of samples trained but the relationship is non linear. Compression can be increased further using Principal Component Analysis (PCA) but the raw CSV data already creates very small file sizes. If you were doing something like image recognition, which we create a test program for as well since it was a simple adaptation, then compression gets more important. For example, we trained on about 200 images scaled to 512x512 resolution and ended up with a 7.6Mb XGBoost model, but this could be compressed. It showed promise for identifying raw images of filets as well as the spectrograms but more image preprocessing is needed. The spectral models worked much better out of the box.

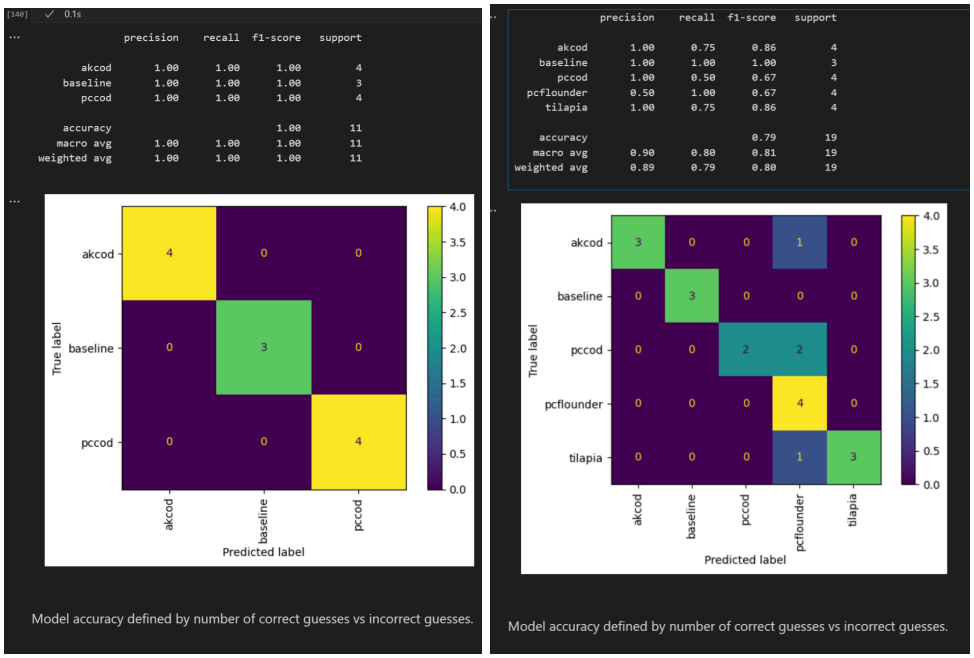
Test Setup:

Here is how we collected a real dataset. The setup is pretty simple. We had our laptop with the spectroscopy IDing web app pulled up, spectrometer connected, and a tray of unfrozen fish. It's important the fish are not too wet or frozen or the water will give you a blander result that is less identifiable since it will just be reflecting the light more, but you can make sure that the angle you are imaging from does not pick up the direct reflection as much to help.

Ad-hoc desk setup for imaging.

Spectral Classification Test Results:

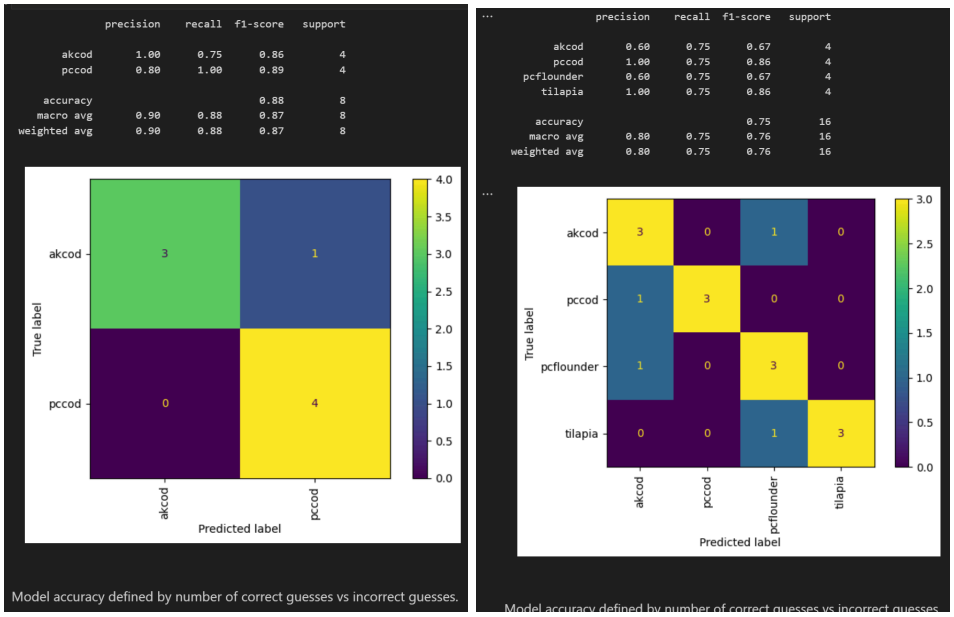
The OV2640 achieved perfect accuracy in our test on a small sample set, about 20 images per species, comparing Alaskan Cod (*Gadus morhua*), Pacific Cod (*Gadus macrocephalus*), and a baseline which was the plain reflection off white paper. The multiclass results with 4 species added Tilapia (*Oreochromis niloticus*) and Arrowtooth Flounder (*Atheresthes stomias*) and scored about 80% accuracy, pretty promising. The Tilapia and Alaskan Cod were from Safeway, the Pacific Cod and Flounder were from Kroger and were all certified. This is good enough that if you took multiple images you could average the guesses and have a fairly confident answer, but this was still a very small data set, with test samples not used in the training set so as to represent a genuine guess.



Left: Binary classification test results with similar species (n=11). Right: Multiclass test result with 4 species and a baseline (n=19). The baseline helps with improving classification results greatly in the binary test case.

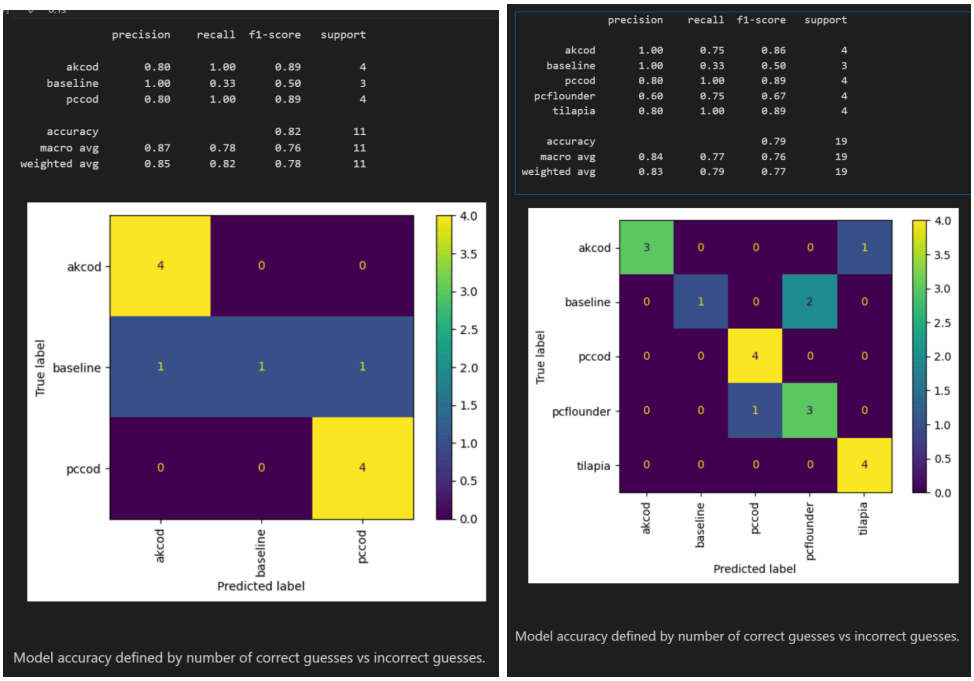
These “confusion matrices” are read as: y axis is the true label, the x axis is the predicted label, a diagonal result is best. XGBoost still works well on non-uniform sample sizes for different classes but those should generally be matched to avoid biasing. We were not especially careful here.

Next are OV2710 binary and multiclass XGBoost results, missing baseline dataset we got about 75% accuracy. The actual output on this camera was completely different from the OV2640. These results are passable but with more data and baseline comparison it would improve. This is with unedited camera settings which can be a large contributor to poor or blurry data, and we will have to compare much more down the line.



Left: Binary classification results with similar species (n=8). Right: Multiclass result with 4 species (n=12). Our baseline results were omitted in this test due to poor quality.

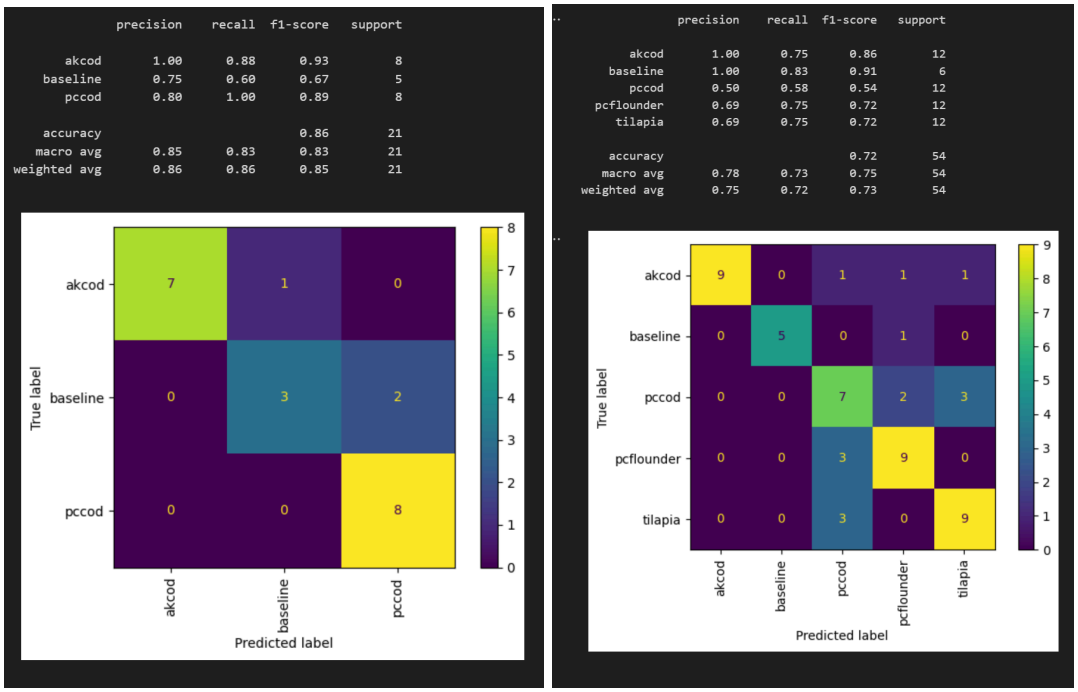
Next is the Samsung Galaxy S10E front zoom camera result, one of 3 cameras on the phone. Our baseline data seemed to be bad in this test but it still perfectly classified Alaskan Cod versus Pacific Cod.



Left: Binary classification results with similar species (n=11). Right: Multiclass result with 4 species (n=19).

Now combining the datasets, we do get less accuracy than in individually trained cases - but still comparable. The model could likely be fine tuned to handle many different camera capabilities in a larger dataset. This

potentially could scale a singular model in an application to many existing phones and PCB cameras that could attach to the imaging box designs. It's a convenient thought, anyway. Notably the binary classification case did as well as the cumulative results of the XGBoost models trained individually on each camera though with slightly different outcomes. We don't know how more data might affect that result.



It's either use a very precise set of hardware and tuning, which is more formally replicable and rigorous, or use a more general dataset with many kinds of hardware and try to account for variances as best we can. We don't even need to always know the exact wavelengths as this is doing simple pattern matching, though it could always help improve results as another dimension. The former is better for forensic science while it's possible our generalized approach here could give anyone these capabilities with their phones and off the shelf cameras - with enough data, that is. It's possible this could be coordinated as part of a more general educational project.

For a very small sample set, these are satisfactory non-random results on various white fish meat plus the redder tilapia as a good comparison that should hopefully also stand out more to our gradient descent model. It struggled the most with misidentifying things as frozen flounder which was very translucent, oily, and white compared to the rest, but more images should help fix that problem. It's obviously not perfect but with more data and testing, it could be made reliable. This will mainly require making a more predictable set of hardware, or otherwise scaling the datasets to match the number of different devices and settings used.

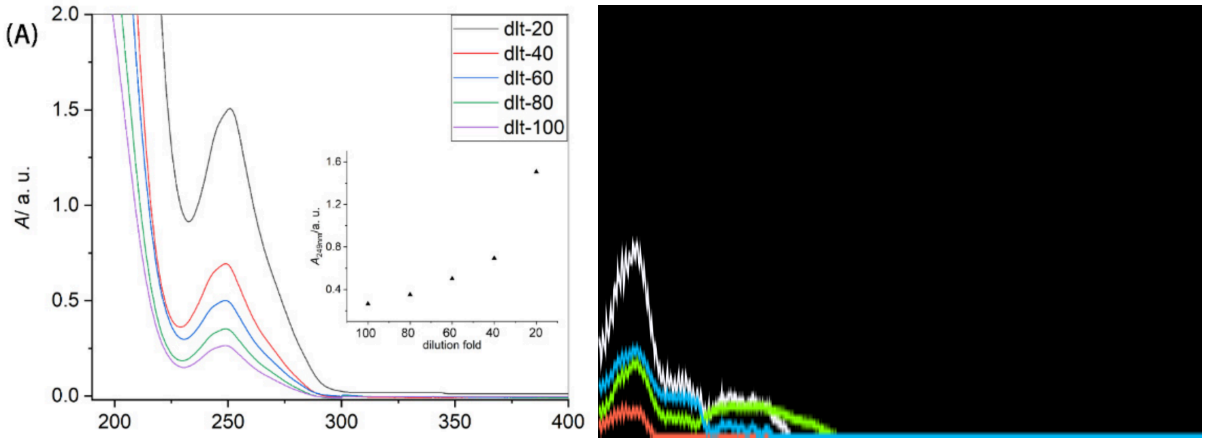
Comparing Results With Previous Work:

There are a few other papers that tested if spectroscopy was viable for identifying fish filets and other body parts. How did we do with our small dataset? How do our results actually look in comparison? The good news is we are on the right track. Our results are not random even with embarrassingly small sample sizes, and we believe

that some relatively simple refinements in the image quality via camera choice and settings should be all it takes to get greater than 95% confidence as shown in many prior works.

An extensive review paper *How Fishy Is Your Fish? Authentication, Provenance and Traceability in Fish and Seafood by Means of Vibrational Spectroscopy* [4] from 2020 compiled dozens of results of spectrometer based identification tasks for fish species and freshness factors. One study used a handheld near-infrared spectrometer using Linear Discriminant Analysis (LDA) on a binary (two species) classification task with a sample size of 80-90 images per species, and achieved around 75% correct classification. Just off the bat, we seem to already have comparable or better results to their paper, where 2 out of 3 of our cameras were able to identify 100% of the Atlantic and Pacific Cod samples correctly in our small test set (11 test samples total, with 15 training samples per species), and the third camera got 7 out of 8 guesses correct on the binary test while all cameras scored around 75-80% on the 4 species test, with the combined result just scoring a little lower around 73%. Again, this is with no fine tuning of the camera parameters beyond changing and focusing the lens zoom. This speaks largely to the versatility of the XGBoost algorithm, which is designed to be self improving especially with more data.

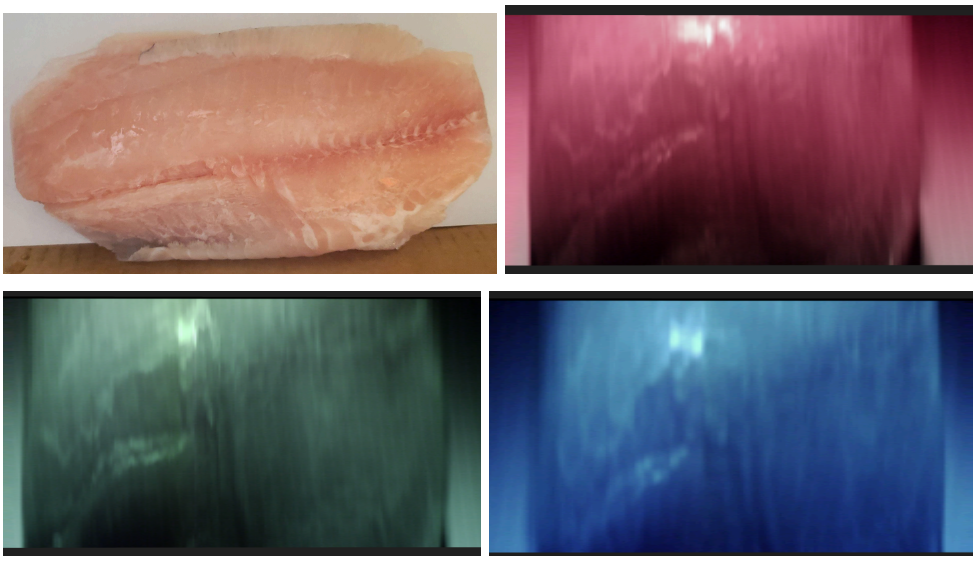
Directly comparing our results to the spectrums shown in an extensive review paper *Rapid Identification between Two Fish Species Using UV-Vis Spectroscopy for Substitution Detection* [7] shows that we are not far behind in the spectral output of our own devices from their professional lab grade versions. We aren't sure if our method is generating the same manner of results though so similarity may be purely visual, as other images can look completely different based on color settings. Using PCA they were generally able to get >95% confidence on binary classification tasks using filet tissue and other parts, while multi-species classification was less differentiable. They specifically note that identifying sub-species is feasible, which is what we also showed in our Atlantic Cod (*Gadus morhua*) versus Pacific Cod (*Gadus macrocephalus*) tests.



Left: The paper's results of several species compiled. Right: Our result for a single species. The RGB in ours are not the same as the different species lines in their chart but generally we seem to be on the right track. [7]

Most of the studies cited in this review achieved accuracies of 80% or greater for identification and freshness tasks (e.g. age since harvest, place of origin, previously frozen versus never frozen) when using visible and infrared wavelengths, sometimes only single wavelengths. They all use methods such as PCA, LDA, and the more advanced ones typically employ Support Vector Machines (SVM) and k-nearest-neighbor (kNN). XGBoost for the most part is a superior and more modern method for general analysis, while components and support vectors can still be added into the training data as preprocessing and/or compression steps. Often more ways to compare the data altogether is better as a kitchen-sink approach. This is something you'll find often in the field of chemometrics, which employs many kinds of control systems and analysis tools to arrive at often extremely accurate results with >95% confidence on tasks like we've mentioned.

Another paper in the review looked at visible and near infrared hyperspectral images and was able to easily conclude which halibut samples were frozen and thawed with greater than 97% accuracy. In our own hyperspectral imaging tests we found it was easy to tell what was frozen or parts were frozen simply by how much light was reflected relative to the baseline, since ice will just reflect the light back and not penetrate as much. These are easy tasks you could accomplish with a regular camera too. One such study used LDA to distinguish haddock and cod filets, and with specific image processing techniques were able to achieve >86% accuracy (n=90), and cited other studies that achieved >95% accuracy with other computer vision methods [8]. This is potentially a much leaner solution to a spectrometer for identification tasks but conditions still need to be controlled in general. We included a basic version of the XGBoost program for training and classifying images that runs very fast on desktop and mobile. More image preprocessing needs to be added to make it effective, following recommendations from the prior filet image analysis work. This image analysis version of the XGBoost can also be appropriated for hyperspectral imaging classification.



Here is a hyperspectral scan of a half-frozen tilapia filet using the OV2640 and a fixed time step for drawing scan lines that we simply synced our hand movements with as best we could. It wasn't perfect, which is why it looks compressed. This image is created in real time and in as many simultaneous wavelengths as specified, in this case

simply 3. Some of the details are apparent from the plain image like the line from the spine. There is a bright patch in the center where ice reflected the flashlight. Previously frozen would need to look at subtler patterns that the XGBoost algorithm should be able to suss out with enough samples.

Overall our results seem to look good. We did our best to control for obvious outliers from poor data collection and followed a specific method to create consistent images, so it is an authentic if limited test of the pattern recognition capabilities of this low cost off-the-shelf hardware. We are pleased and with some fine tuning, as outlined next, results should be able to consistently meet accepted standards for scientific food identification and quality control.

Important Design Refinement Considerations:

There are many important factors to consider when refining this design into an easily repeatable, mass manufacturable design. The designs we presented to you are more or less toy spectroscopes that are still clearly viable for analysis tasks but are hard to scale and get identical results every time. Many of the following factors will confound your datasets if not controlled for. It would take a long time to break down each and every one of these things, so here is a list to summarize:

1. **Camera Sensor Quality and Filtering**: The choice of camera sensor is critical as variations in UV and IR sensitivity and color spectrum fidelity directly affect the accuracy of spectral data. Identifying low-cost sensors that offer a balance between broad spectral range and high-resolution imaging is essential for consistent results. **Different camera sensors will completely change results**.

2. **Camera Exposure and Color Settings**: Adjusting camera settings such as exposure, color balance, and resolution is necessary to optimize image capture for spectral analysis. These settings affect the clarity and comparability of spectral data across different samples. **It will completely change your results**.

3. **Diffraction Grating Density and Quality**: The diffraction grating plays a pivotal role in dispersing light into its spectral components. Selecting a grating with the appropriate density and bandwidth ensures accurate separation of wavelengths, crucial for detailed spectral analysis. In our case with digital cameras, we want UV-Infrared ranged diffraction grating, and 1000 lines/mm gives good results and is still affordable. **Again, this will completely change results.**

4. **Lens Zoom and Focus**: A lens with adjustable zoom and focus capabilities is necessary to ensure sharp images of the sample. Autofocus is nice if it works but manual focus is likely more precise. Our version requires zero image preprocessing based on the aperture and imaging angle we chose, which is very convenient as a developer. There are other options for a tube that can be straight on to the camera

instead. See the PySpectrometer project, which we didn't know about prior, however, it uses a 90 dollar spectroscope tube. [9]

5. **Light Slit Narrowness**: The width of the light slit affects the amount of light entering the spectrometer and thus the clarity of the resulting spectral image. A balance must be struck between allowing sufficient light for analysis while maintaining narrowness to enhance resolution. 0.1mm is a good benchmark. Under 50 micrometers you need brighter light sources or more sensitive sensors for consistency.

6. **Box Opacity and Roughness**: The material of the spectrometer's housing should be non-reflective and opaque to prevent external light interference and reflections within the device, which can degrade the quality of spectral data.

7. **Light Source for Sampling**: The choice of light source influences the spectral range and intensity of illumination on the sample. A consistent and suitable light source is vital for repeatable and accurate spectral measurement.

8. **Distance from Sample**: The distance between the light source, sample, and sensor affects the intensity and distribution of light on the sample. Maintaining a consistent distance ensures uniform light exposure and reliable spectral data collection.

9. **Software Sample Averaging and Baseline Comparison Method**: Employing a simple yet effective method for data processing, such as subtracting the average of multiple frames from a baseline average, allows for the generation of a rough absorbance spectrum. This pragmatic approach balances simplicity with functional accuracy. Moving averages can also help smooth results but will lose information.

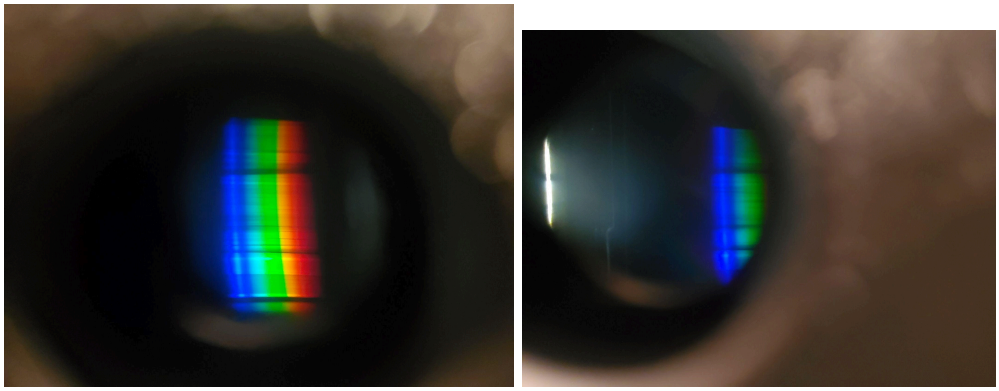
10. **Miniaturization and Scalability**: Designing the spectrometer for easy miniaturization and scalability enhances its portability and durability. This adaptability enables broader application and facilitates scientific exploration in varied environments.

11. **Hyperspectral Imaging Motion Technique and Software**: The quality of hyperspectral imaging is contingent upon the precision of the motion technique and the software's ability to accurately interpret relative perspective. Optimizing both hardware movement and software processing is key to maximizing the fidelity of spectral imaging. Industrial hyperspectral imaging simply uses cameras with extra color channels; ours avoids that by imaging the spectral band itself while moving the image proportionally to the device movement, which creates a ghost image of the scanned substance in targeted wavelengths.

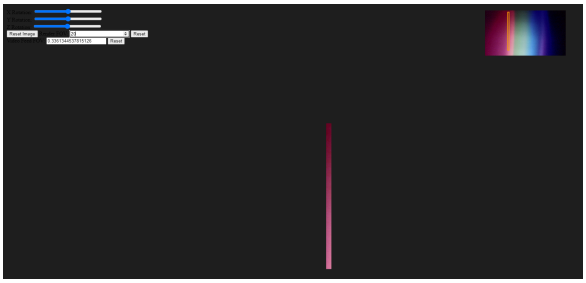
Hyperspectral Imaging Test:

The last piece is inspired by a paper *Low-Cost Hyperspectral Imaging with A Smartphone* [4]. To quote the abstract: "The Hyperspectral Smartphone is capable of accurate, laboratory- and field-based hyperspectral data collection, demonstrating the significant promise of both this device and smartphone-based hyperspectral imaging as a whole." Hyperspectral imaging allows you to image in specific wavelengths based on which color scanline you focus on in the vertical image field. Their method, however, employs a stepper motor and rail with fixed timing for taking an image. This is rather inconvenient.

We tapped the motion sensors built into the phone using generic web APIs and also laid the groundwork for using custom sensors e.g. through a Raspberry Pi with an accelerometer and a web socket. This means the hyperspectral software "just works" once your phone or camera is mounted, and it can be packaged as something on the app store or served freely through the web.



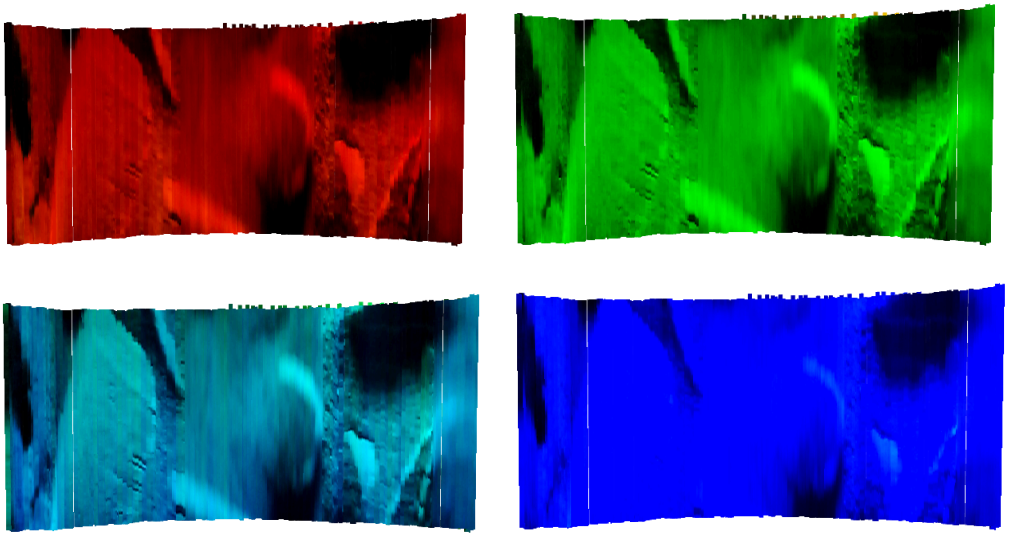
Above: Raw image through phone (pre-zoom). Below: Basic application environment.



The application before imaging is a simple hideable controls display for drawing hyperspectral images with a hack using the efficient 3D library called ThreeJS. The image is painted into a 3D scene from vertical lines cut out of the primary spectral image, and continuously painted with each video frame, creating scanlines for each wavelength. You can use this technique to image a taller spectrum that can cover a wider field of view though your lens and diffraction grating quality can distort this image if not selective. For the regular spectroscopy counting up intensities in each color, this larger spectrum gives much richer data due to a larger imaging area. The application will automatically subdivide images for you or you can specify. You can either use motion controls or set fixed trajectories of your scan line, which is modeled as a rotating face of a sphere from a central position while saving the pixels from each draw call to paint an image as it moves.

Candy wrapper imaged across 5 wavelengths. Notice how the left side text is not highlighted in Red. Magic!

Imaging rocks, which give a nice variation for test imaging.



Fossil results in Red, Green, and Cyan, and Blue wavelengths.

Note the hyperspectral imaging was done entirely handheld using the phone's motion sensors and steady movement. This can be automated on a motorized tripod or, say, drawn from steady movement on a conveyor belt, for more precise and detailed results. The field of view stretching for painting a correctly scaled image works programmatically and needs to be set to match the camera's field of view and zoom level.

Conclusion:

This project demonstrates the viability of DIY spectroscopy in the $50-$100 range for potentially serious scientific and regulatory endeavors, and highlights a path toward cheaply refining it into an industry standard. We hope our work illuminates this better for scrappy scientists and engineers who want to create tools like this that can be built and used from the ground floor. We importantly wanted to highlight the software tools used to develop our testing suite as it contains a model for low cost translational technologies, where user-friendliness and development costs are everything for moving research tools into public discourse.

This is very relevant in our case of looking at fish fraud where we need to immediately scale responses to this global issue with evidence-based approaches that can allow government regulatory bodies to prosecute violations in important conservation laws. Yes it could be as simple as taking a picture with any existing cameras either bare or slotted into a spectrometer box with 3d printed parts if we make the right use of existing tools like XGBoost and either image or color spectral analysis. There are already plenty of other publicly available, community-managed computer vision tools for things like plant and animal identification, and this will only get easier as technology in general evolves. While our dataset was fairly small, our DIY-grade, barebones hardware already shows parity with prior work in the field of spectroscopy-based fish identification, meaning even low fidelity tools are capable of performing these tasks, which should be a good signal to interested groups that this can be done at scale, especially when spectral analysis is already a staple in the food and drug industry. This is also getting harder to regulate as the world gets bigger and funding or effectiveness limits oversight capacity.

Our results were possible with help from around the world and are completely open source and available under the LGPL v3 license to encourage use and for other contributors to share their improvements freely. The Cortex A53 module is under GPL v3 which is a slightly stricter copyleft license for preserving public domain works.

Citations:

1. *Theremino Spectrometer Construction*. Theremino. 2014. <https://www.theremino.com/wp-content/uploads/files/Theremino_Spectrometer_Construction_ENG.pdf
2. Rajwa, Bartek. *Spectroscopy of Deception. Unmasking Food Fraud with Biophysics.* 2024. <https://pswscience.org/meeting/2490/>
3. *Revealed: seafood fraud happening on a vast global scale*. The Guardian. 2021. <https://www.theguardian.com/environment/2021/mar/15/revealed-seafood-happening-on-a-vast-global-scale>
4. Stuart, M. et al. *Low-Cost Hyperspectral Imaging with A Smartphone.* Journal of Applied Sciences 2021. <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8404918/>
5. Wug Oh, S. et al. *Do It Yourself Hyperspectral Imaging with Everyday Digital Cameras*. IEEE. 2016. <https://ieeexplore.ieee.org/document/7780639>
6. Power, A. Cozzolino, D. *How Fishy Is Your Fish? Authentication, Provenance and Traceability in Fish and Seafood by Means of Vibrational Spectroscopy.* Applied Sciences. 2020. <https://www.mdpi.com/2076-3417/10/12/4150>
7. Croce, A. *Rapid Identification between Two Fish Species Using UV-Vis Spectroscopy for Substitution Detection*. Molecules. 2021. <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8587656/>
8. Grassi, S. et al. *Fish Fillet Authentication By Image Analysis.* Journal of Food Engineering. 2018. <https://www.sciencedirect.com/science/article/abs/pii/S0260877418301675?via%3Dihub>
9. Write, L. PySpectrometer 2. <https://github.com/leswright1977/PySpectrometer2>

Special thanks for assistance and ideas to:

Bernard Markus
Davis Fay
Paul Phan
Mateusz Stankiewicz @ Flyps
Yassine Santissi
Daniel Csati
Don Gerhart
Morgan Hough