

Molegazer

Astronomy meets dermatology

Chris Frohmaier - PyData Southampton - 17 Oct 2023

Dr Chris Frohmaier

- University of Southampton
- Senior Research Fellow in Physics and Astronomy
- Research Software Engineer: Legacy Survey of Space and Time

 cfro.astro@gmail.com

 chrisfrohmaier



Context

Astronomy and Dermatology?

Get more data!

Legacy Survey Space Time



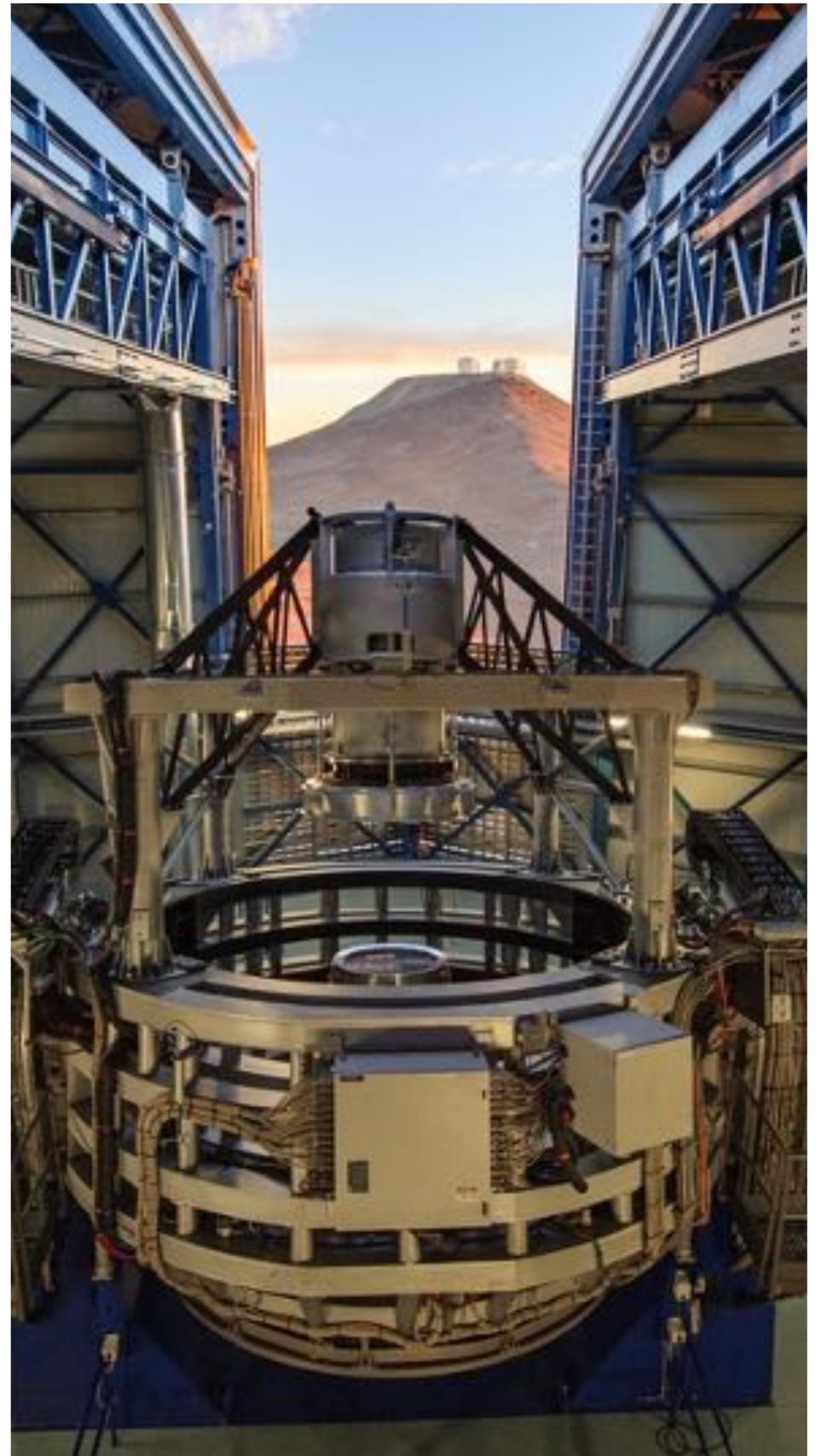
Discover exploding stars (supernova)

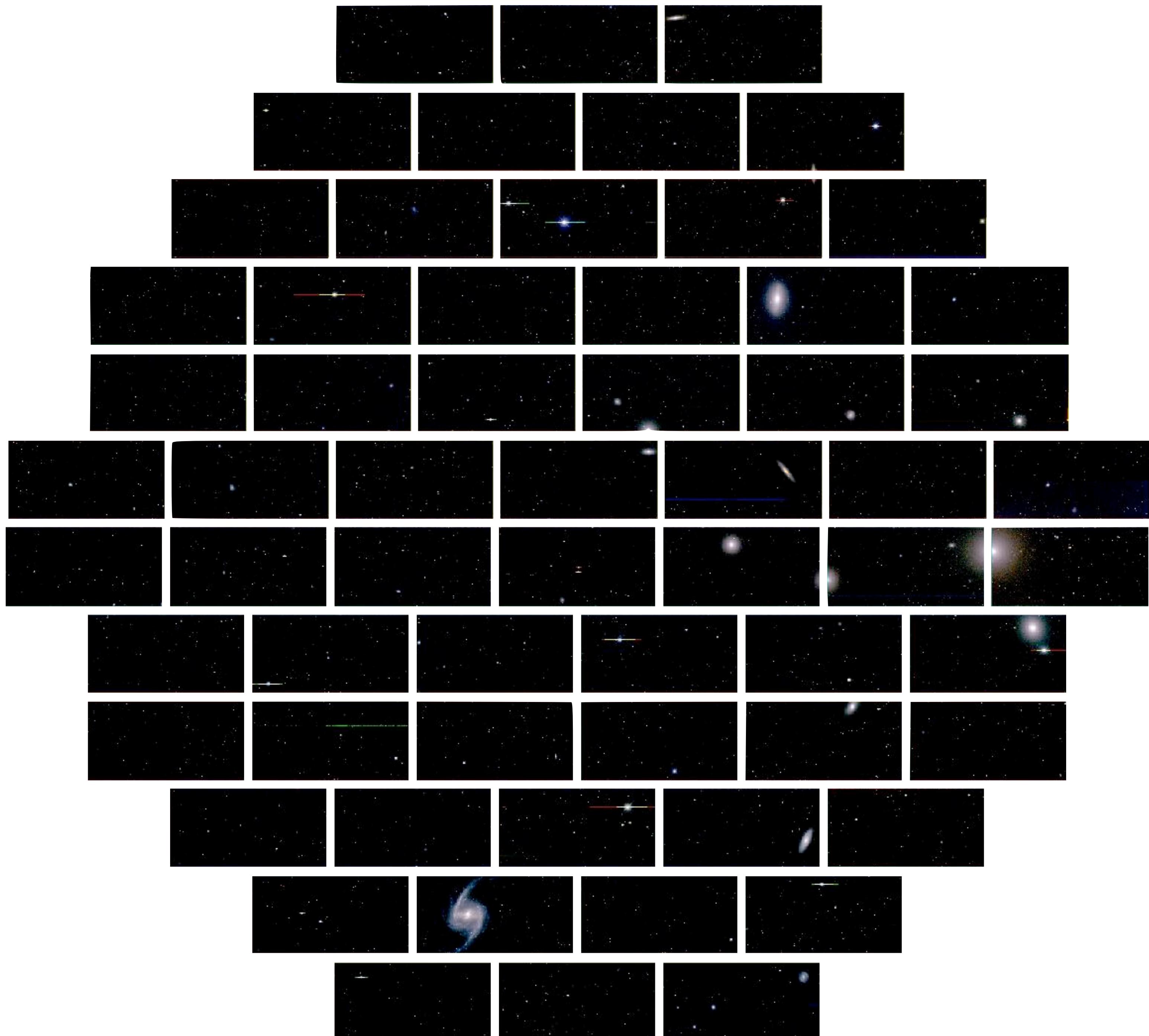


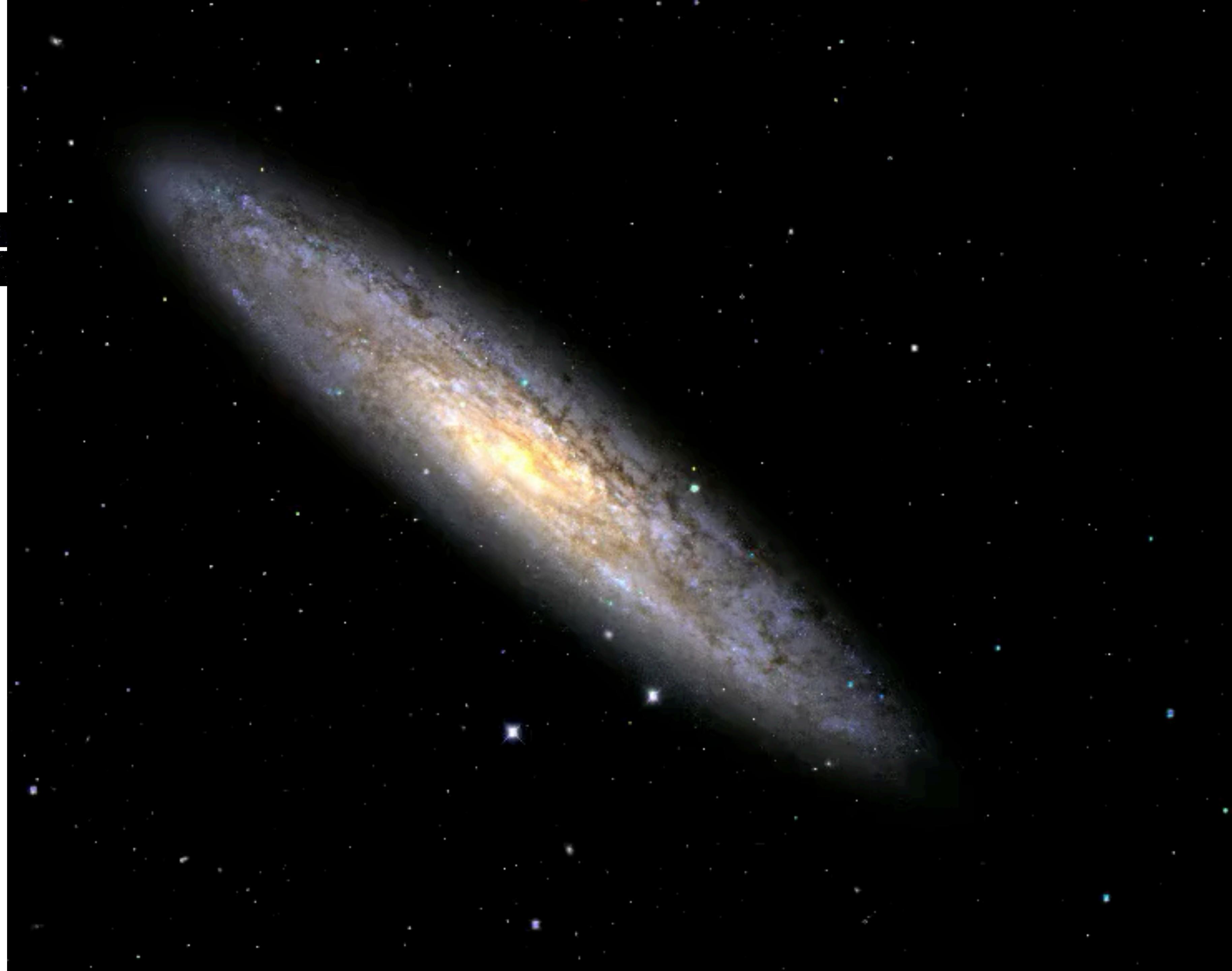
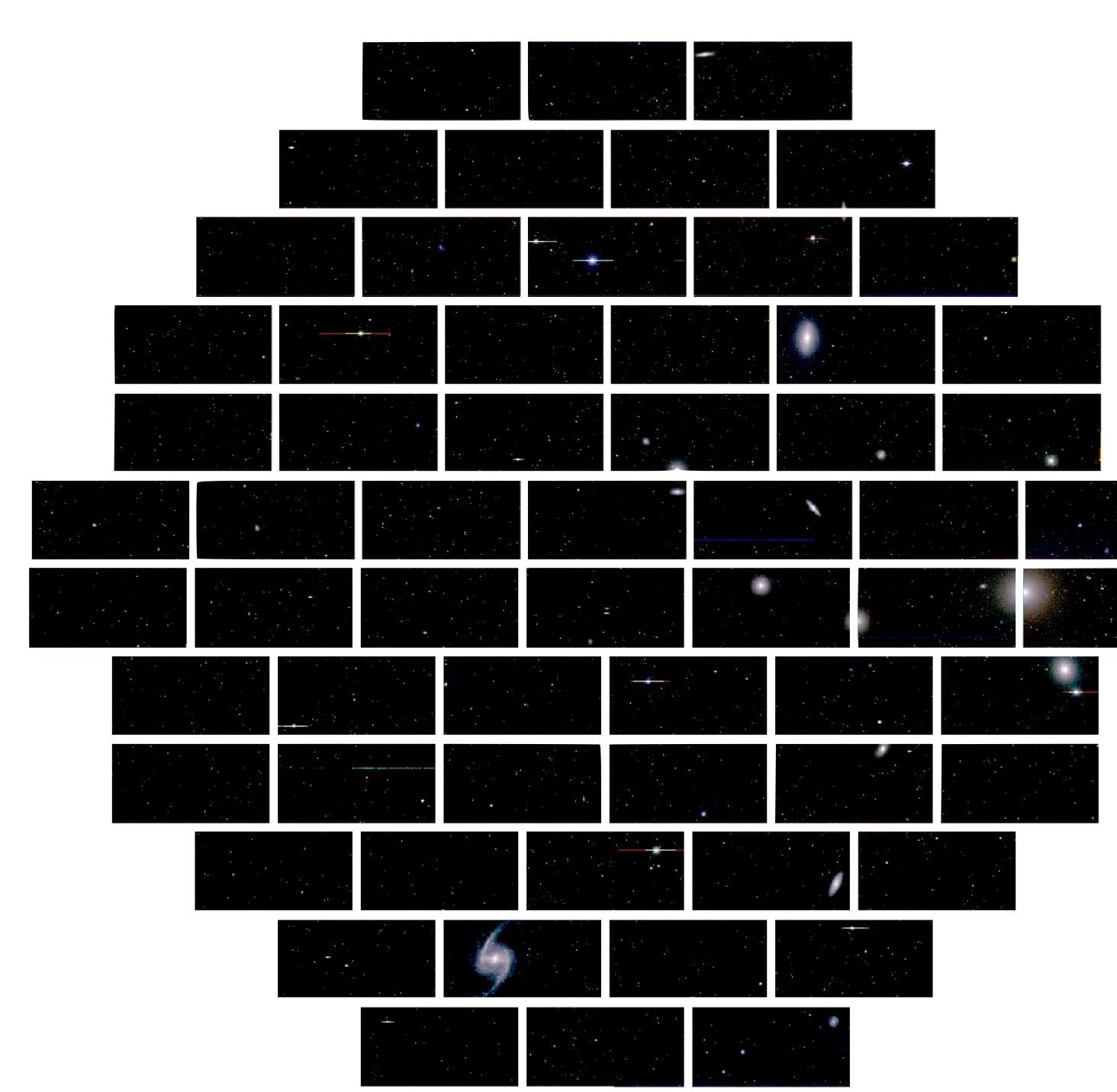
20 TB/night - I orchestrate with prefect.io



15 PB after 10 year experiment







Dermatology



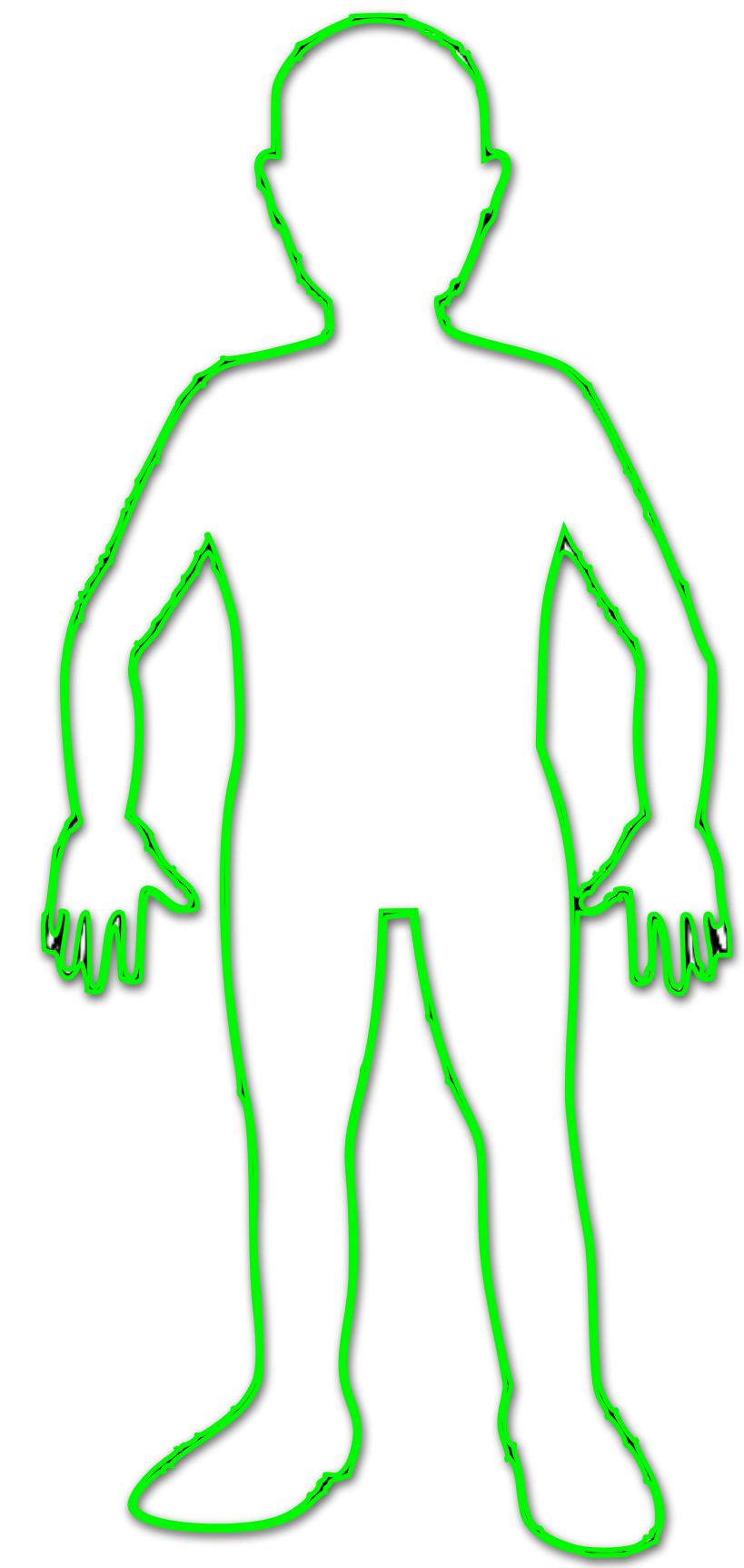
CANCER
RESEARCH
UK



Science and
Technology
Facilities Council



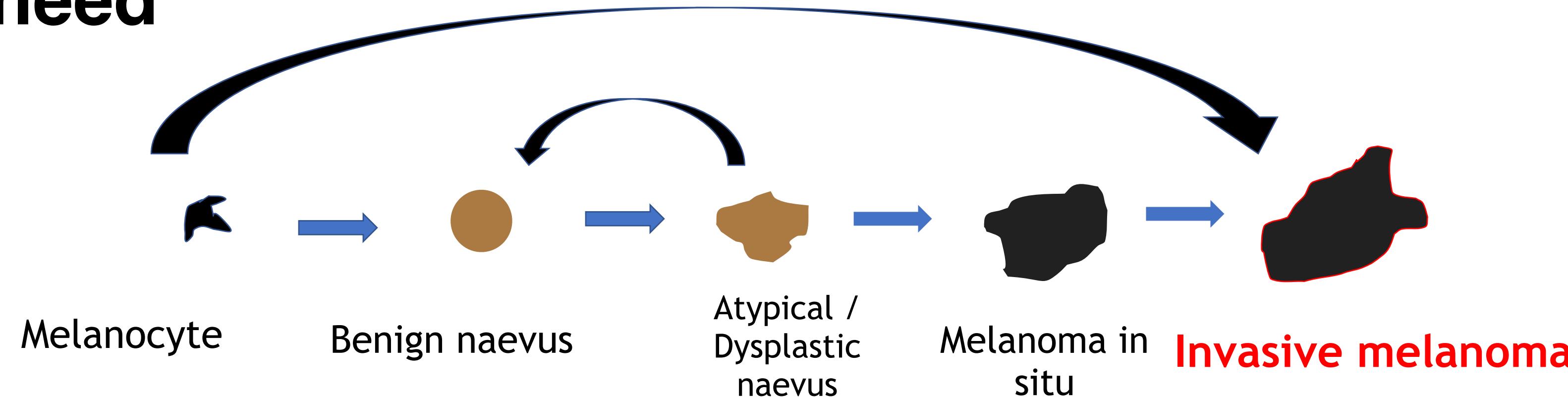
Dr Rubeta Matin PhD FRCP
Consultant dermatologist
Senior Lecturer
Oxford University Hospital Trust





Context 2

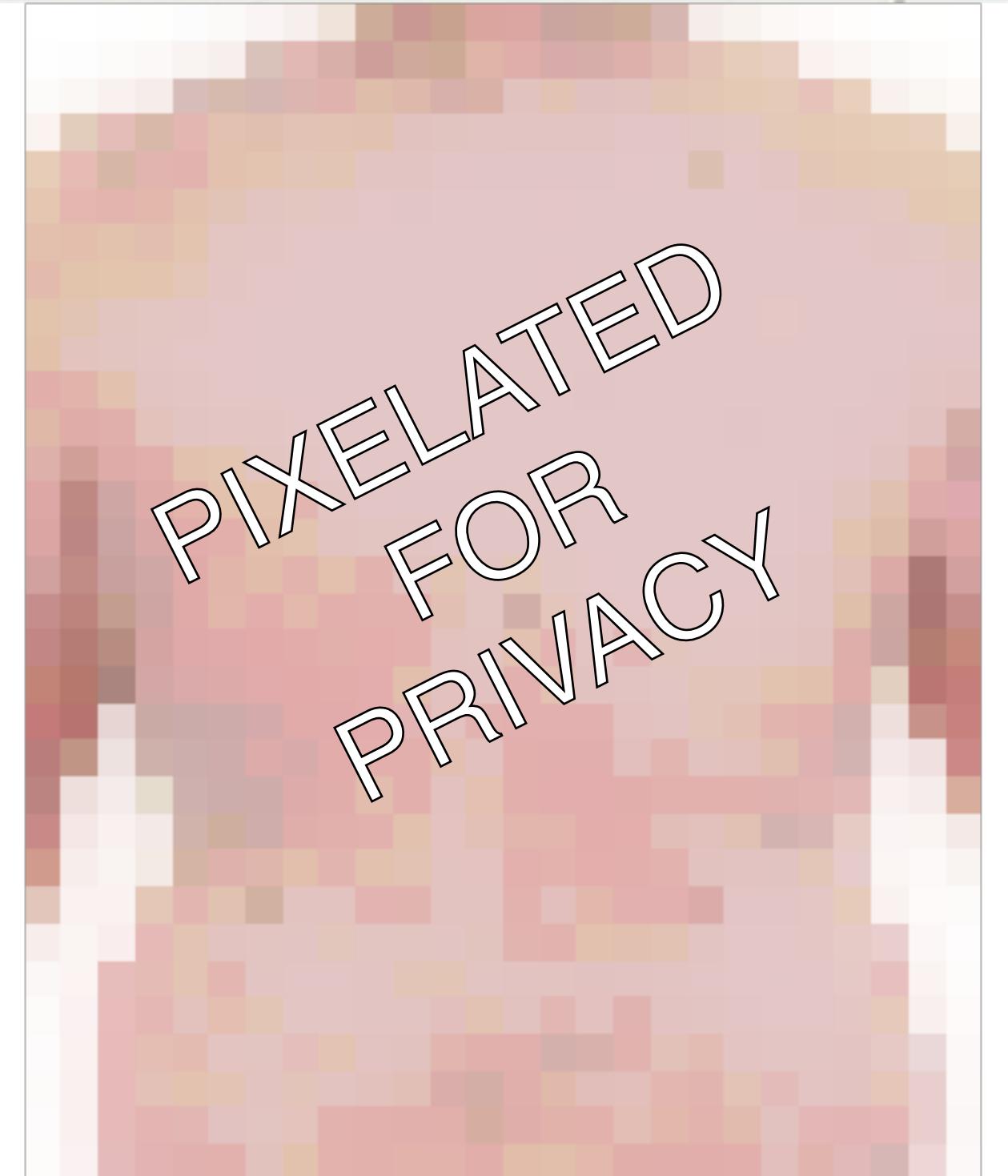
Medical need



- 60% of melanoma comes from pre-existing naevus (mole)
- >6/day die of melanoma in UK
- Sequential monitoring of patients is labour intensive and not routinely performed
 - Moles are checked by eye and compared to a PDF printout by a consultant!

The Data

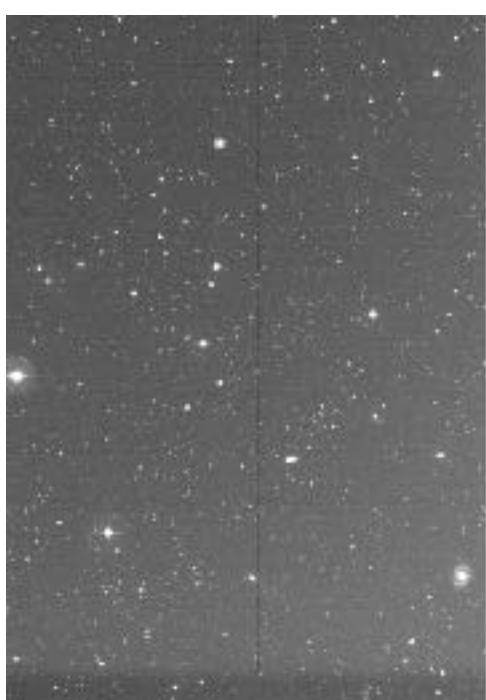
- 50 NHS patients (Oxford cohort)
 - High risk of developing melanoma
- Clinical setting
- Total Body Photography ~3 months collected over 3 years
- 500 patients with single epoch snapshot photography
- >8,000 individual images taken to-date
- All identifying images are removed before sending to Southampton
- Nikon NEF format



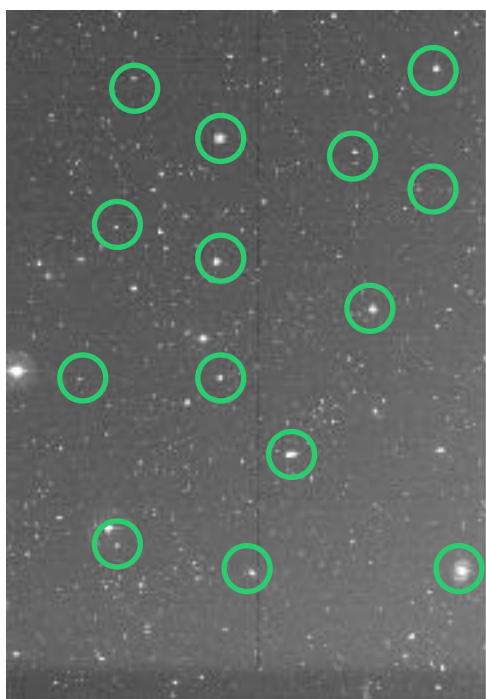
Project Aim:
Predict which naevi will
evolve into melanoma

(work-in-progress)

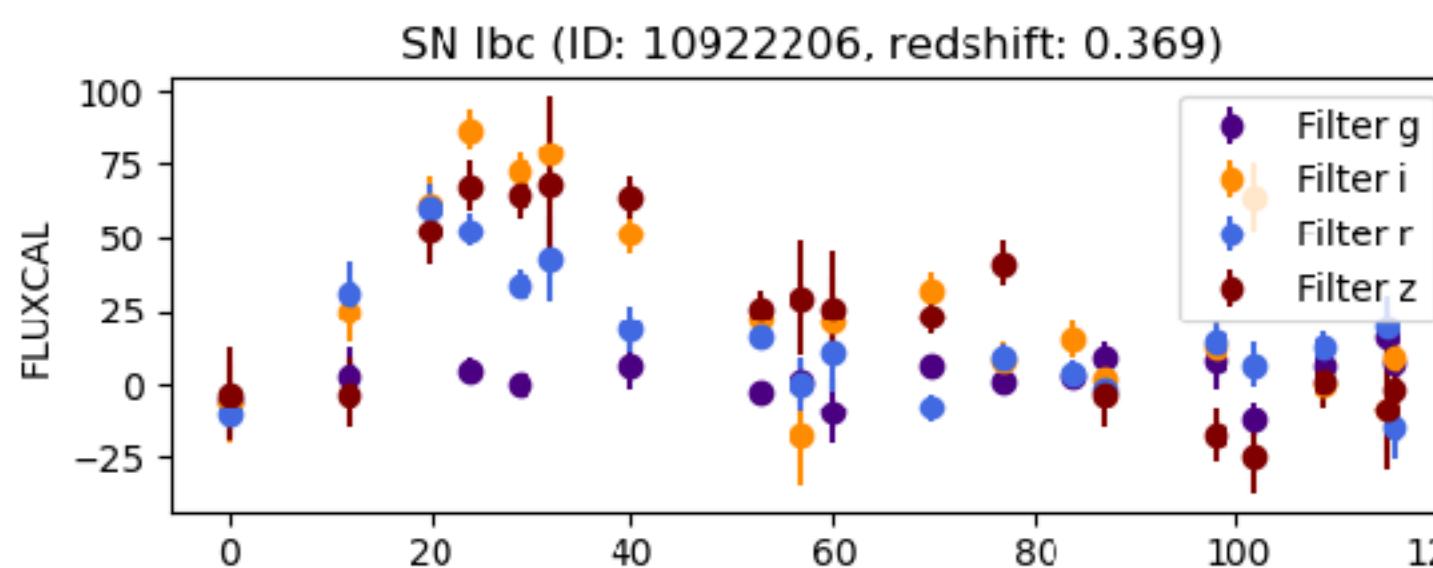
Aquire Image



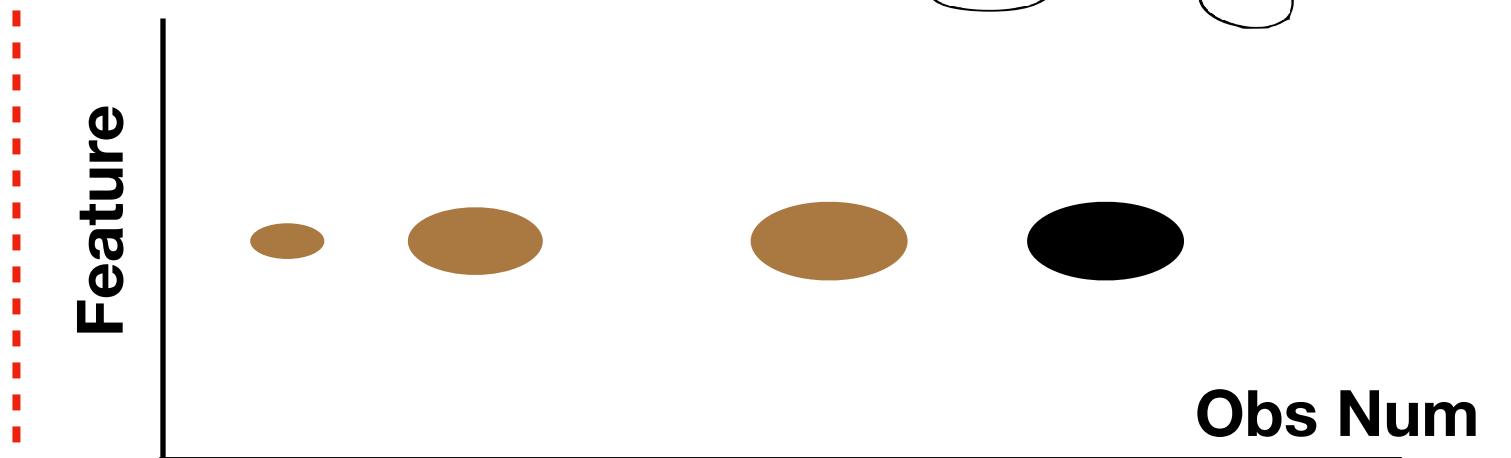
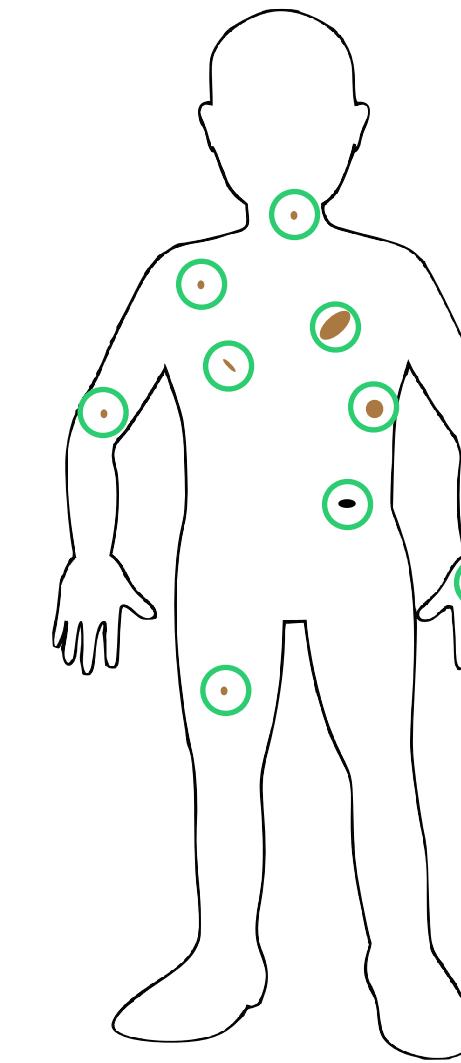
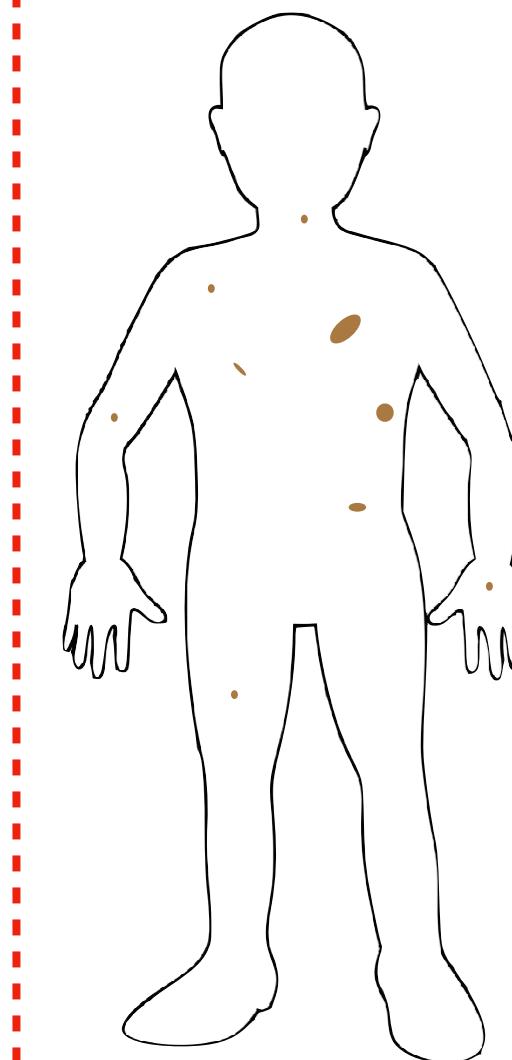
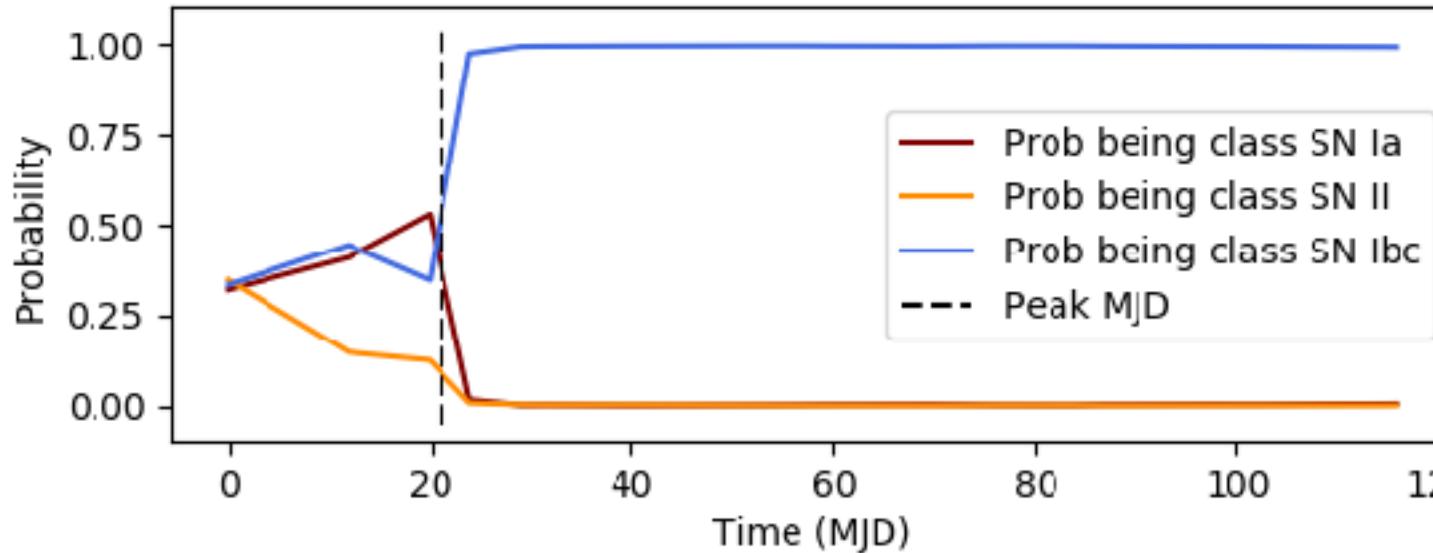
Source Detection



Compare History

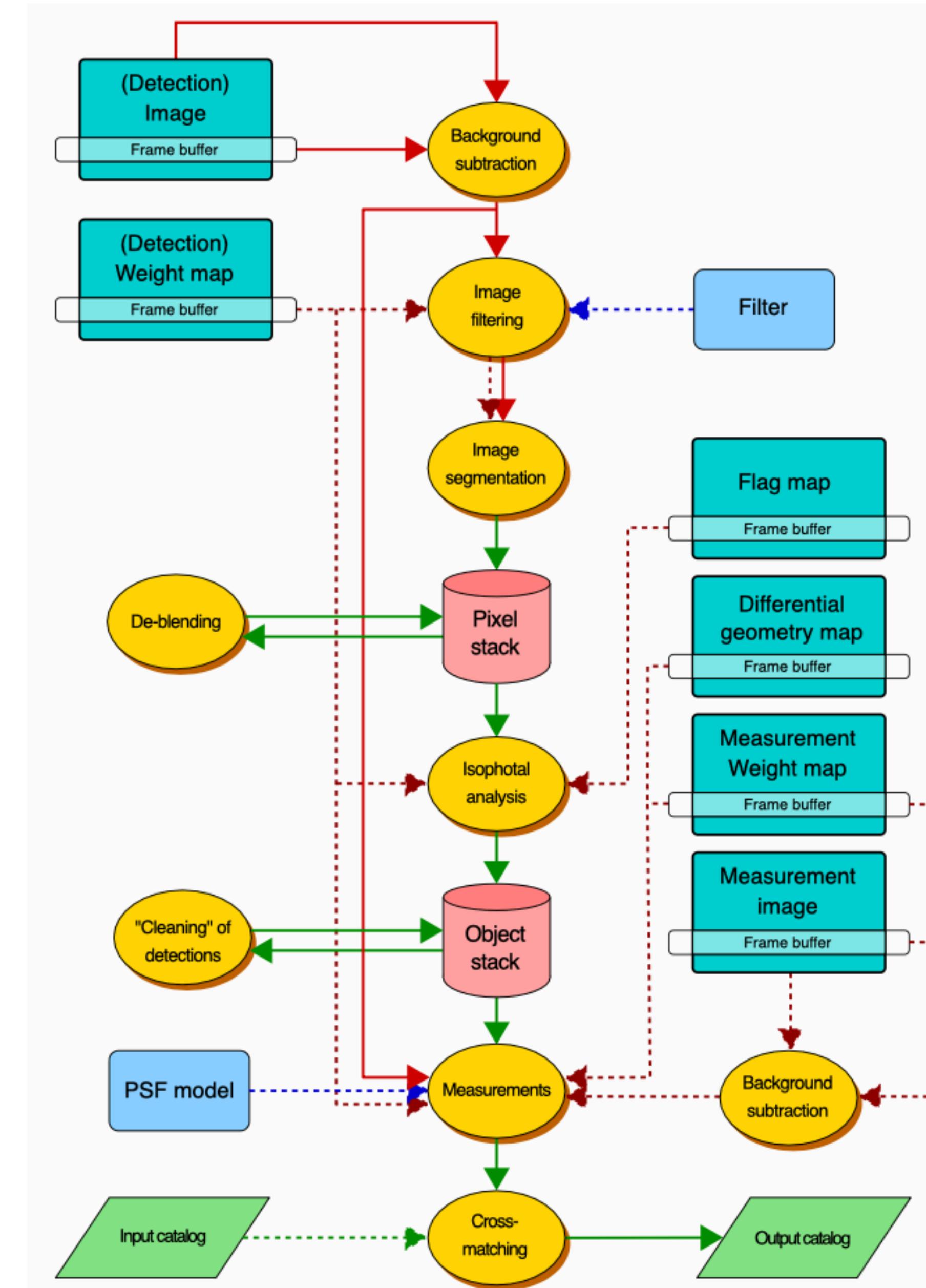
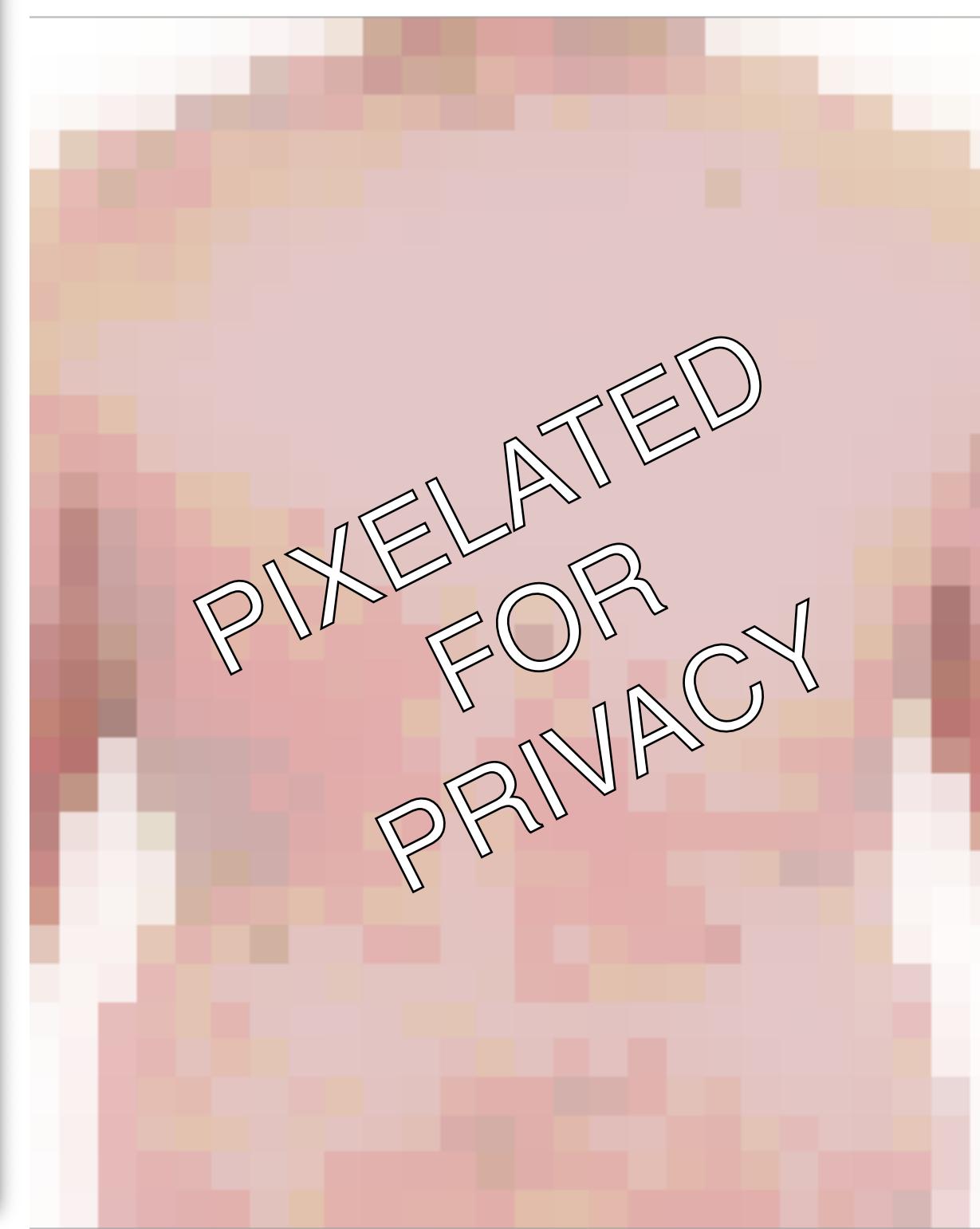


Predict classification/
evolution



The astronomy solution

Source Extractor



**We could detect moles... but it wasn't a
scalable solution**



Research
Software
Group



Dr Sam Mangam

Senior Research Software Engineer

University of Southampton



Dr Ed Parkinson

Research Software Engineer

University of Southampton



Dr Sam Mangam



Dr Ed Parkinson

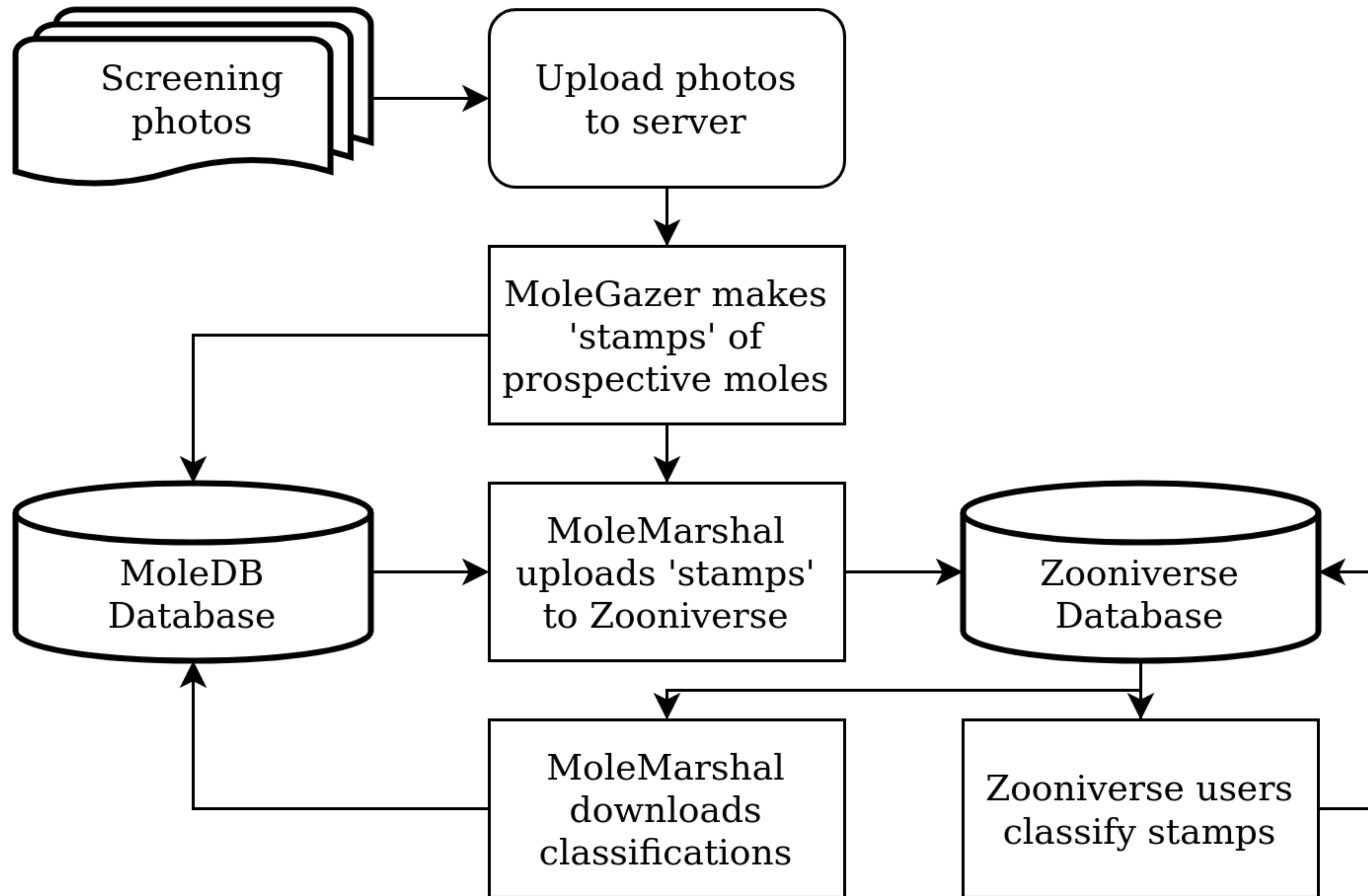
**Improve our mole
detection algorithms
within the Python
framework**

**Create a system for
dermatologists to label
our data to train future
models**

MoleGazer

Mole Marshal

The Moleverse

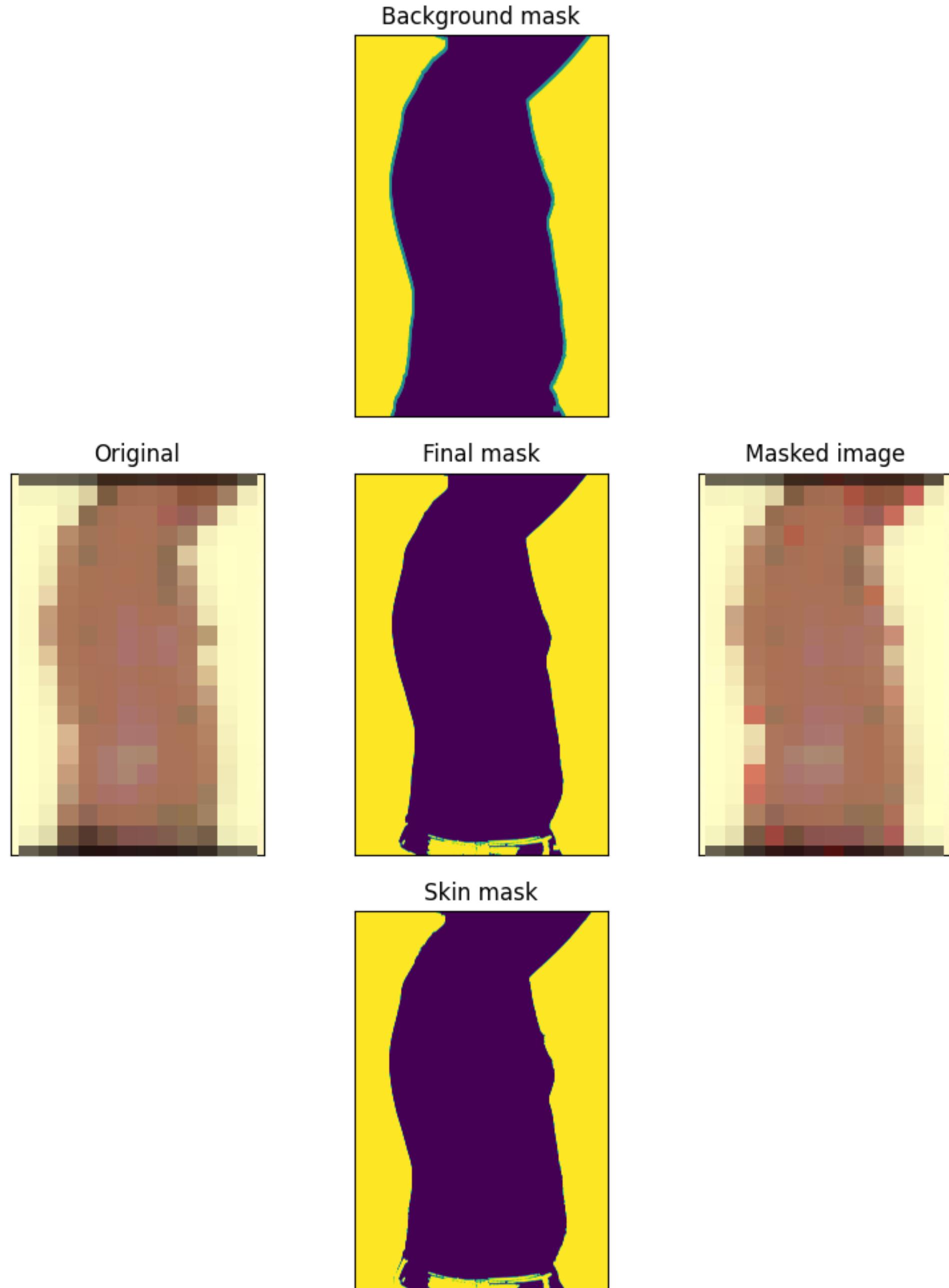


Molegazer

Detection skin via hue and saturation, then detect background colour.

Use the Laplacian of Gaussian method

- Good for blob detection and edge detection
- Greyscale the image
- Apply a Gaussian blur, then take the second derivative of the pixel intensity map.
- When crosses 0 consider it an edge.





```
from skimage.feature import blob_log
from skimage.color import rgb2gray
from skimage.io import imread
import numpy as np
import matplotlib.pyplot as plt

webb_deep = imread('./Webb_s_first_deep_field.png') #Open JWST deep field
webb = webb_deep[1000:4000,1000:4000] #Small area for demo

webb_gray = rgb2gray(webb) #Convert to Grayscale

#Compute Laplacian of Gaussian
blobs_log = blob_log(webb_gray, max_sigma=30, num_sigma=10, threshold=.1)

#Size is blob is  $\sqrt{2} \times \text{sigma}$  for 2D image
blobs_log[:, 2] = blobs_log[:, 2] * np.sqrt(2)

fig, axes = plt.subplots(1, 1, figsize=(5, 5))
axes.imshow(webb, origin='lower')
for blob in blobs_log:
    y, x, r = blob
    c = plt.Circle((x, y), r, color='palegreen', linewidth=1, fill=False, alpha=0.7)
    axes.add_patch(c)
plt.show()
```



```
from skimage.feature import blob_log
from skimage.color import rgb2gray
from skimage.io import imread
import numpy as np
import matplotlib.pyplot as plt

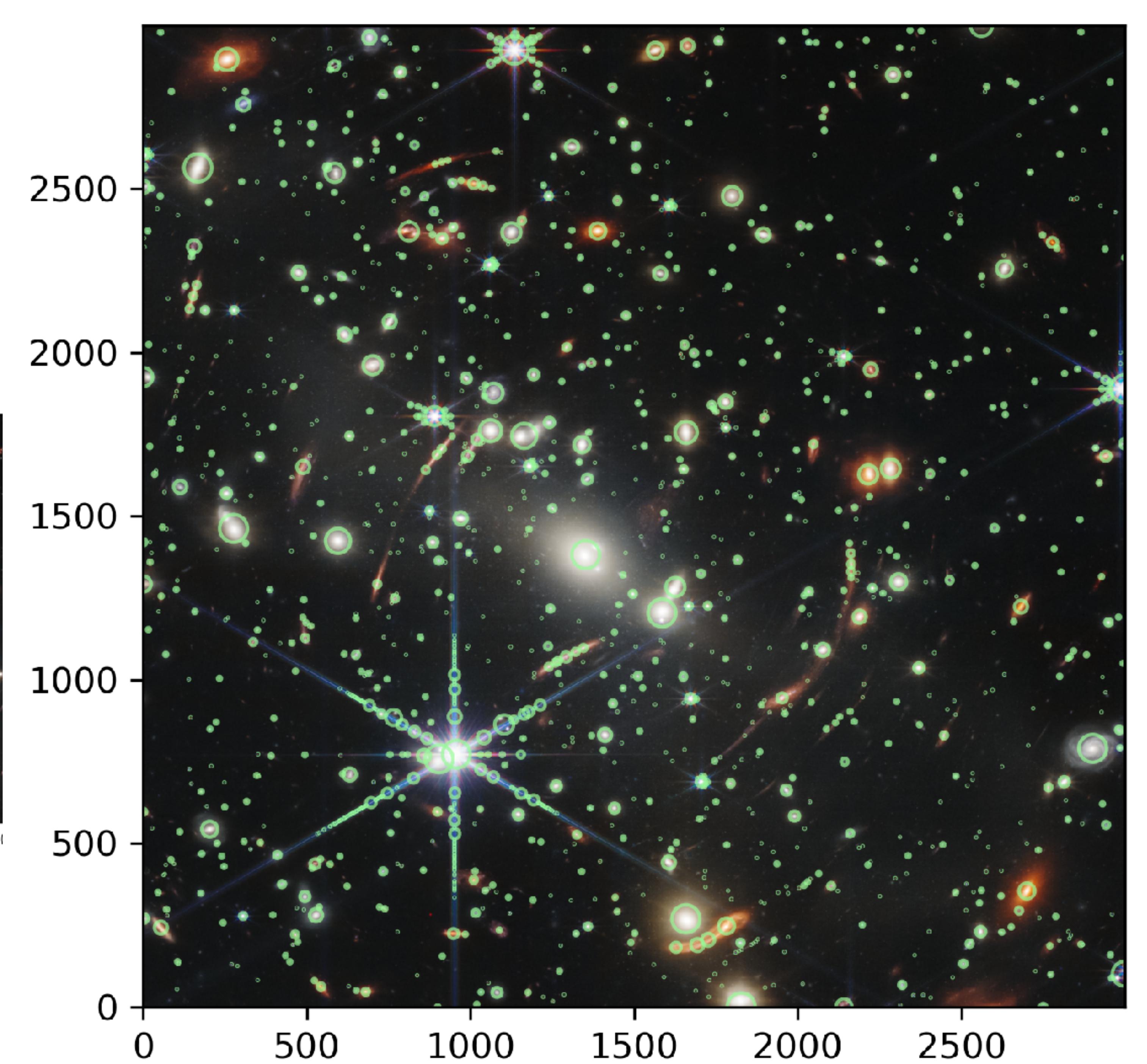
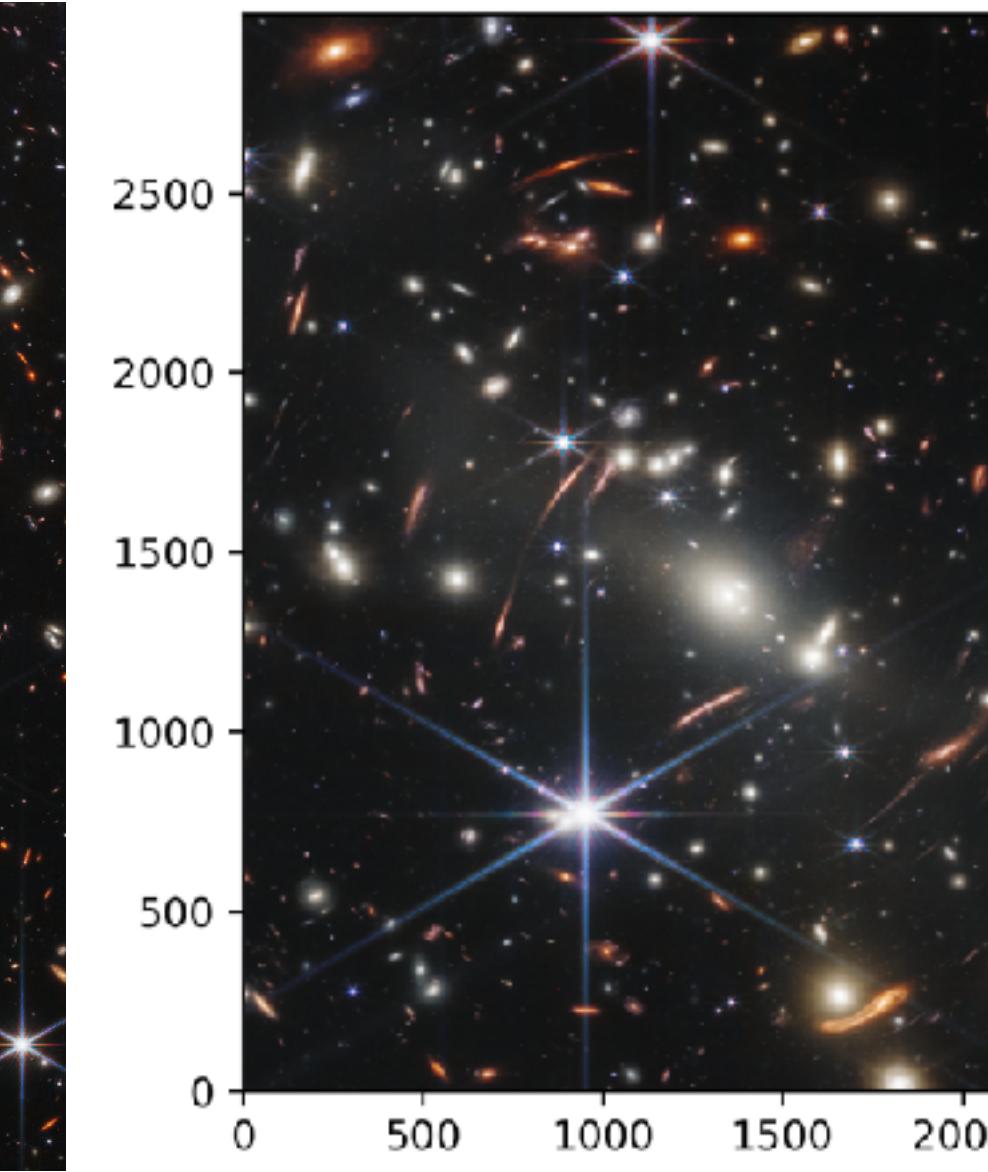
webb_deep = imread('./webb_s_first_deep_field.png') #Open JWST deep field
webb = webb_deep[1000:4000,1000:4000] #Small area for demo

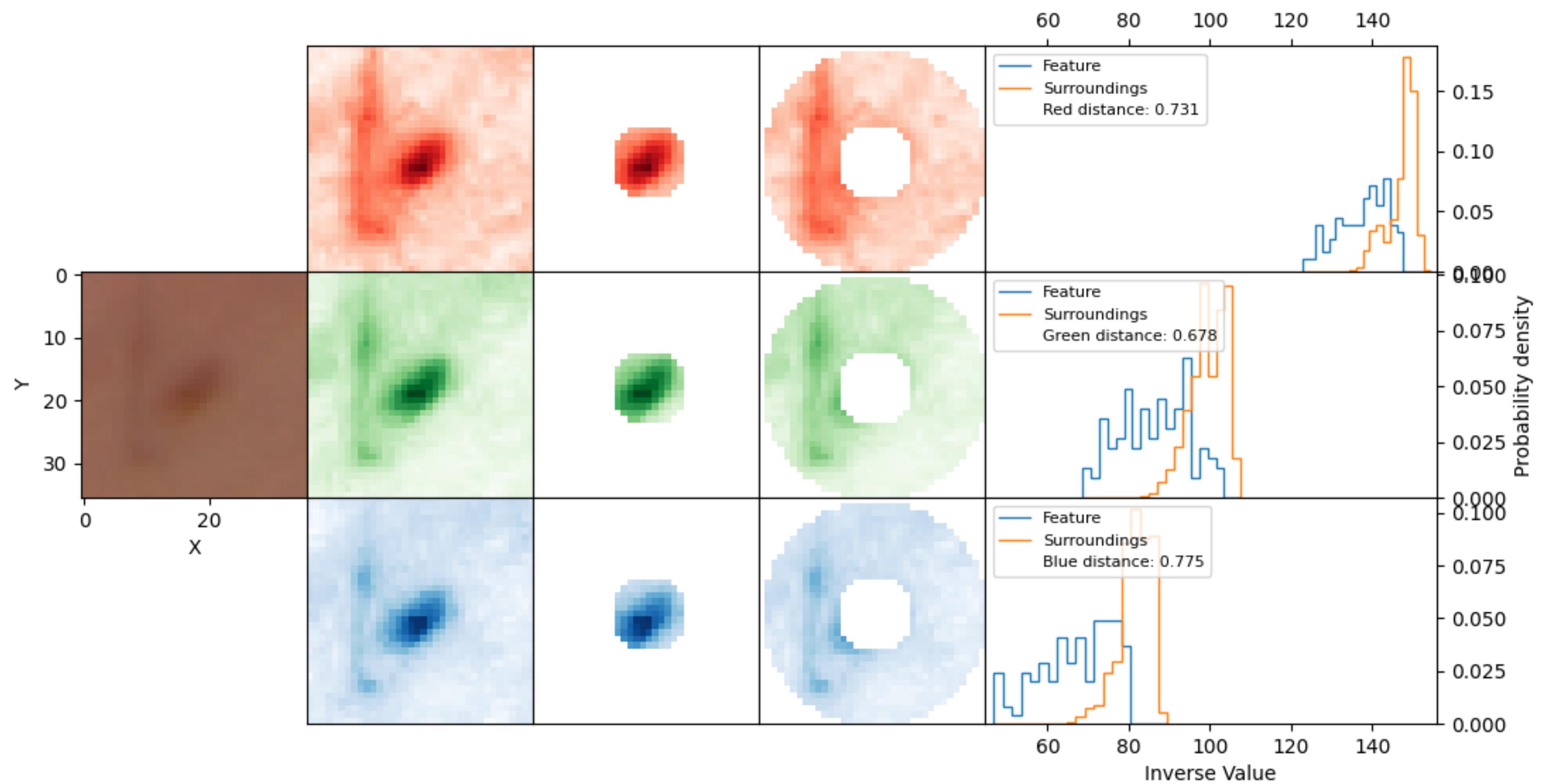
webb_gray = rgb2gray(webb) #Convert to Grayscale

#Compute Laplacian of Gaussian
blobs_log = blob_log(webb_gray, max_sigma=30, num_sigma=10, threshold=.1)

#Size is blob is  $\text{sqrt}(2) * \sigma$  for 2D image
blobs_log[:, 2] = blobs_log[:, 2] * np.sqrt(2)

fig, axes = plt.subplots(1, 1, figsize=(5, 5))
axes.imshow(webb, origin='lower')
for blob in blobs_log:
    y, x, r = blob
    c = plt.Circle((x, y), r, color='palegreen', linewidth=1, fill=False, alpha=0.7)
    axes.add_patch(c)
plt.show()
```





surroundings_difference_min@

Difference: $\mu_1 - \mu_2$

A pixelated background featuring a grid of brown, tan, and yellow squares. Overlaid on the image is the text "PIXELATED FOR PRIVACY" in a white, sans-serif font. The text is rotated diagonally from the bottom-left towards the top-right. There are small, solid-colored squares (pink, yellow, brown) scattered across the text and the background grid.

MoleMarshal

Labelling the data



- Citizen Science platform engaging the public with research
- Made famous for asking public to label images of galaxies
- MoleMarshal is a private project
 - Patient sensitive data
 - Only accessed from behind UoS firewall
- Consultant dermatologists can look through the data and label it
- (I am not allowed to demo this - patient confidentiality)

Interface

The screenshot displays the MoleMarshal-Development web application interface. At the top left is a user icon with a circular arrow symbol. Next to it is the text "MoleMarshal-Development". Along the top right are navigation links: ABOUT, CLASSIFY (which is underlined), TALK, COLLECT, RECENTS, and LAB.

The main content area features a large image of a skin lesion with a red circle highlighting a specific area. Below this image is a smaller row of three smaller images, each with a red circle, likely showing different views or stages of the mole.

To the right of the images is a task panel. It includes tabs for "TASK" and "TUTORIAL", with "TASK" currently selected. The task question "Is there a mole in the top image?" has two options: "No" and "Yes". A blue rectangular box labeled "NEED SOME HELP WITH THIS TASK?" contains these buttons. At the bottom of the panel are three buttons: "Done & Talk" (light blue), "Done" (green), and a settings gear icon.

At the very bottom center of the page are small icons for information, heart, and list.

Interface



MoleMarshal-Development

ABOUT CLASSIFY TALK COLLECT RECENTS LAB



Is there a mole in the top image?

TASK **TUTORIAL**

No

Yes

NEED SOME HELP WITH THIS TASK?

Done & Talk Done ⚙

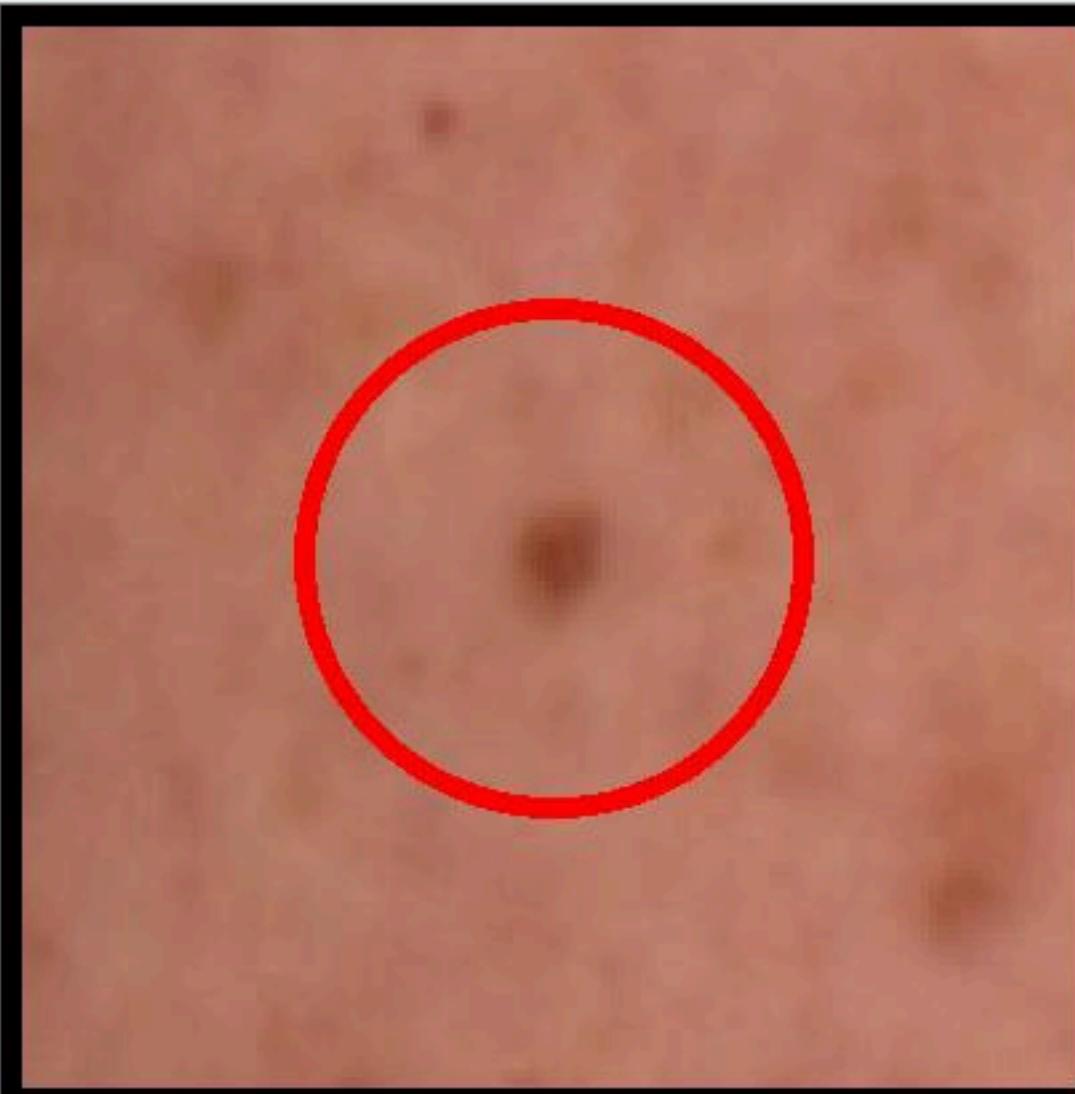
SWITCH TO DARK THEME

Interface



MoleMarshal-Development

ABOUT CLASSIFY TALK COLLECT RECENTS LAB



How severe is this mole, in comparison to the moles in the lower images?

TASK

TUTORIAL

1

2

3

4

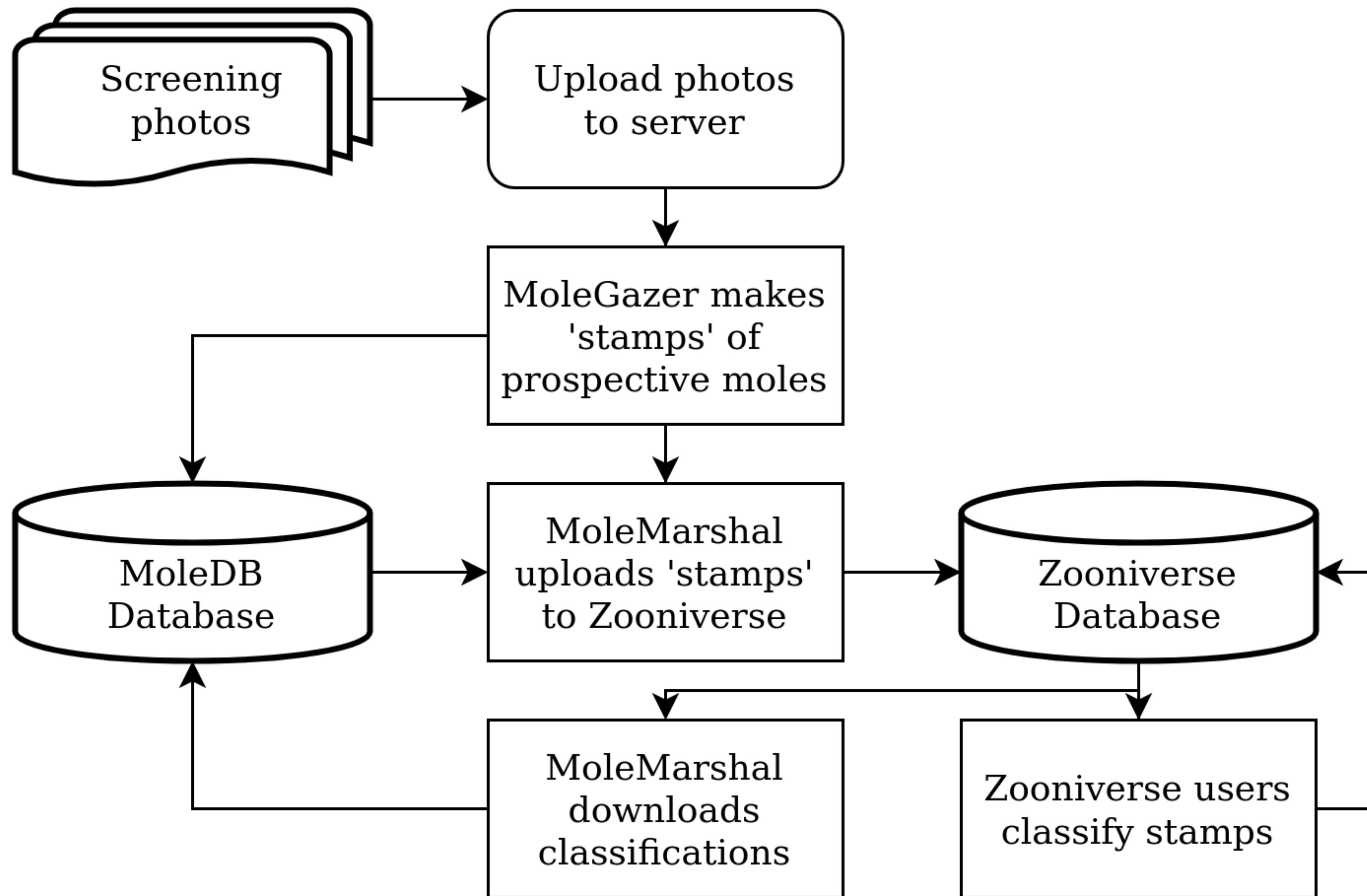
5

N/A

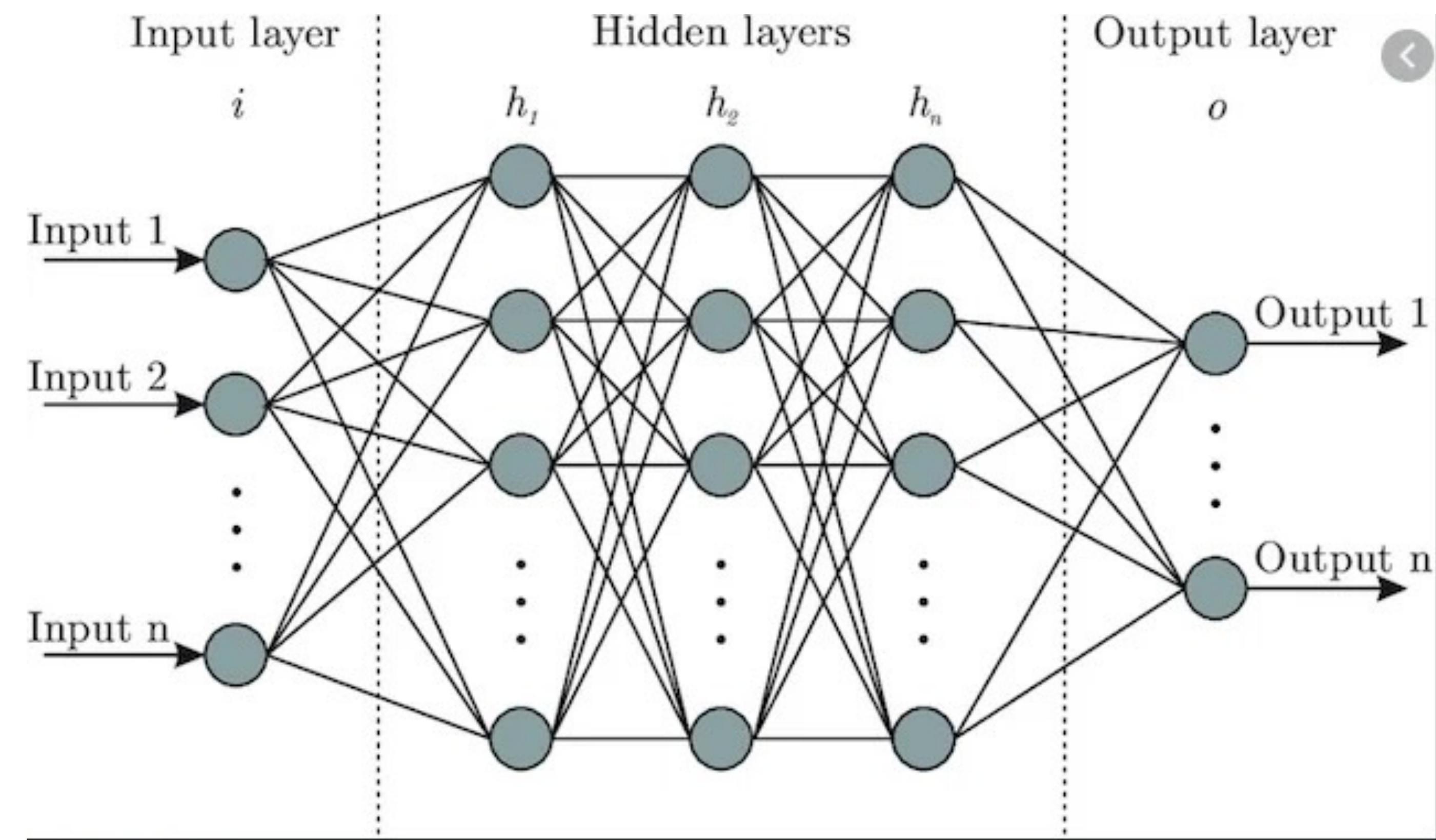
NEED SOME HELP WITH THIS TASK?

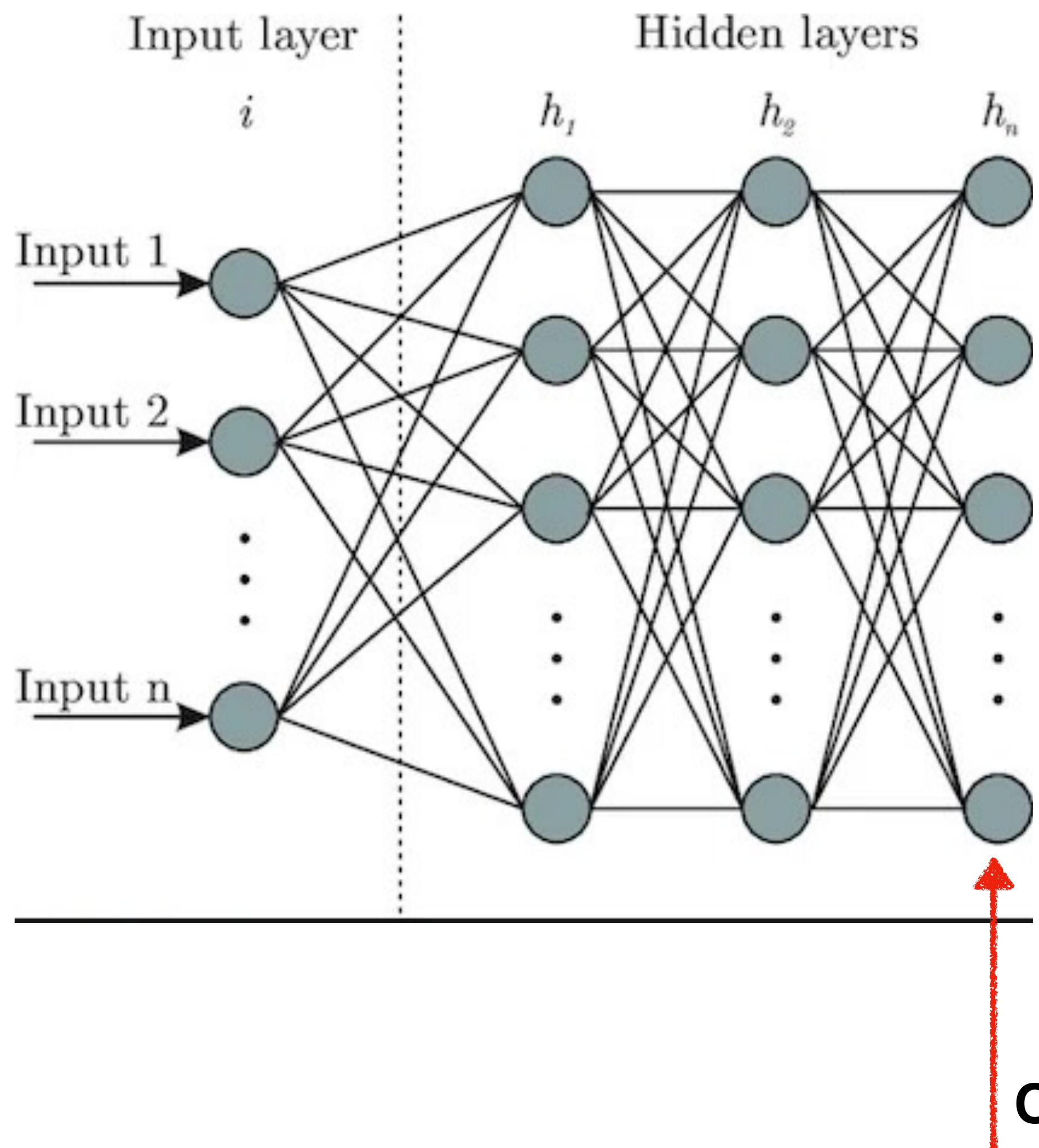
Back Done & Talk Done ⚙

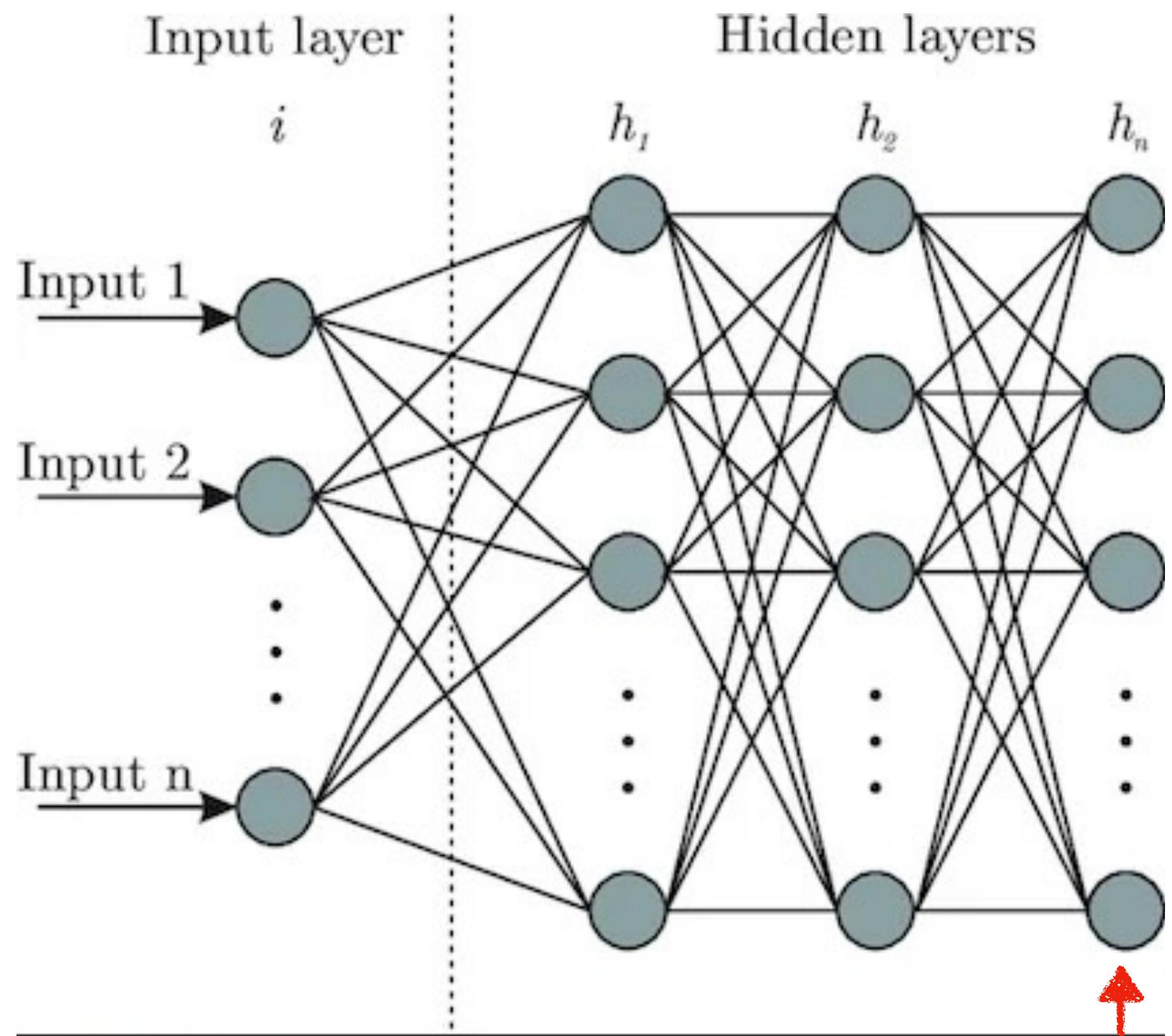
SWITCH TO DARK THEME



**Finding similar moles...
efficiently label data**







Only use the values at these nodes
Similar regions in this nD-space would
be occupied by similar looking moles?



```
# for loading/processing the images
from keras.preprocessing.image import load_img
from keras.preprocessing.image import img_to_array
from keras.applications.inception_resnet_v2 import preprocess_input

# models
from keras.applications.inception_resnet_v2 import InceptionResNetV2
from keras.models import Model

# clustering and dimension reduction
from sklearn.cluster import KMeans
from sklearn.decomposition import PCA

# for everything else
import os
import numpy as np
import matplotlib.pyplot as plt
from random import randint
import pandas as pd
import pickle
import glob
```



```
# load model
model = InceptionResNetV2(
    weights='imagenet', # Load weights pre-trained on ImageNet.
    input_shape=(299, 299, 3),
)
model = Model(inputs=model.inputs, outputs=model.layers[-2].output)
```

```
imageList = glob.glob('/my/very/private/medical/data/stamps/*.png')

def extract_features(file, model):

    img = load_img(file, target_size=(299,299))
    img = np.array(img)
    reshaped_img = img.reshape(1,299,299,3)
    imgx = preprocess_input(reshaped_img)

    features = model.predict(imgx, use_multiprocessing=True)

    return features

data = {}
for mole in imageList:
    feat = extract_features(mole,model)
    data[mole] = feat
```



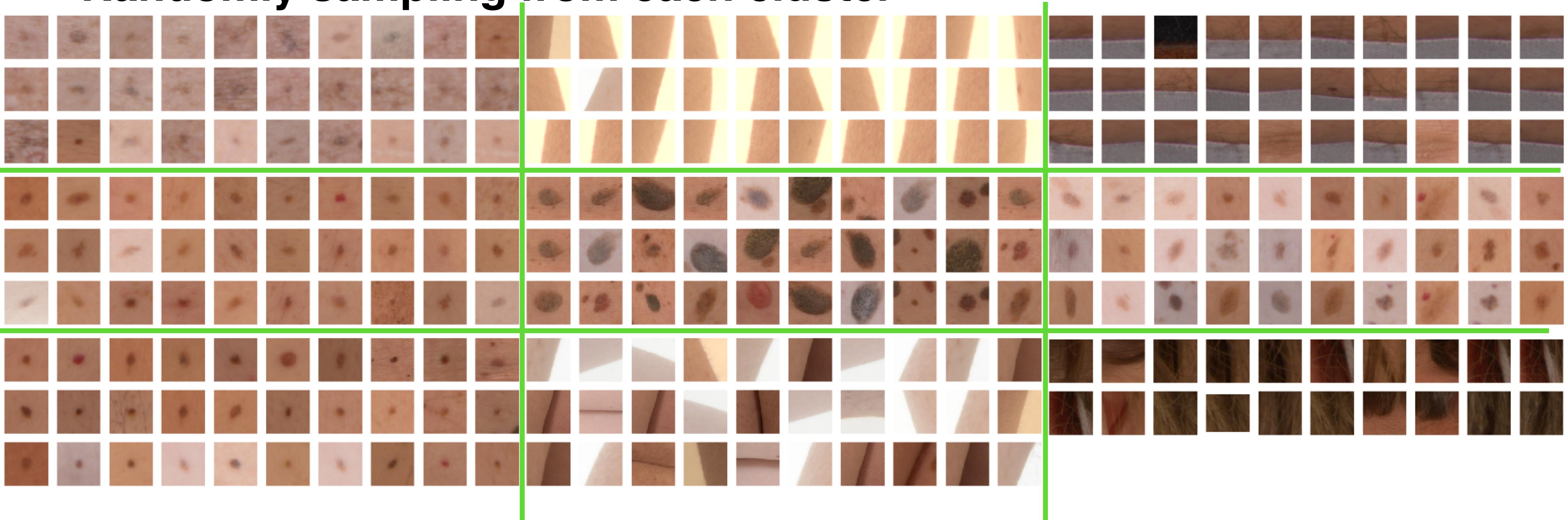
```
pca = PCA(n_components=100, random_state=42)
pca.fit(feat)
x = pca.transform(feat)

numberOfClasses = 10 #Guess how many classes

kmeans = KMeans(n_clusters=numberOfClasses, random_state=42)
kmeans.fit(x)
```

Plot the clusters

Randomly sampling from each cluster



A more efficient labelling system

(For fun)

What did the classifier predict?

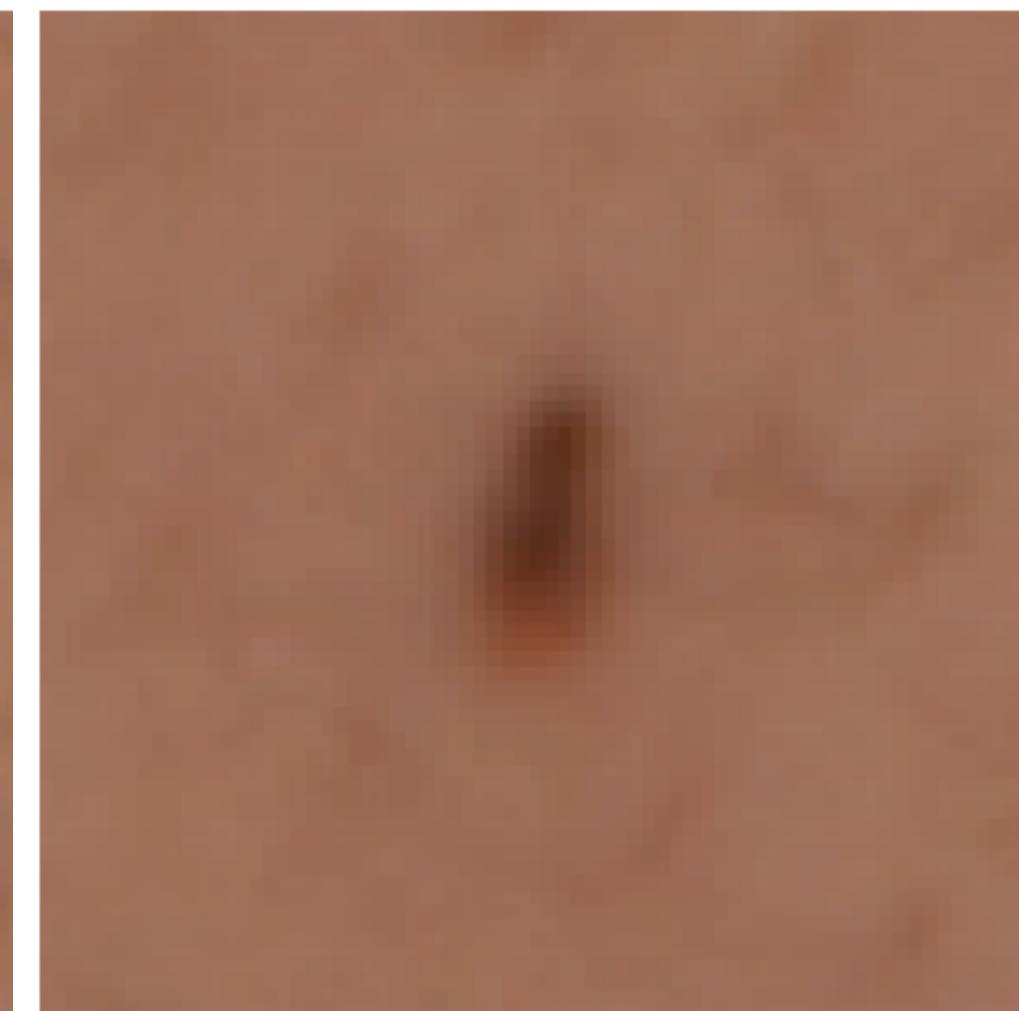
tick, 0.91



tick, 0.96



tick, 0.28



banana, 0.18



mashed_potato, 0.05



Future work

- Things seem to work on single snapshots!
 - We can detect moles on patients
- More diverse cohort!
- Need to build in time series
 - Tracking moles over multiple years
 - How to align patients so we know we are looking at the same moles
 - Astronomy techniques don't work reliably for this problem.

Questions?



Dr Chris Frohmaier

 chrisfrohmaier



cfro.astro@gmail.com



cfrohmaier

