Snakes Species Classification: With Python

Gianna Pedroza, Max Cabilangan, Nikhil Tripathi,
Rishvik Kambhampati, Audrey Huang,
Sophia Santos, Rohan Saklani, Helen Yuan,
Maxwell Chu, Aman Sehra

Introduction

Goals & Motivations

★ Given an image of a snake, **identify** its species

★ **Classify** whether the snake is venomous

Steps

1. **Web scrape** images of different snake species
2. Convert images to a **dataset**
3. **Classify** images via augmentation and build a model
4. Use our model to **identify** the snake species and whether it's venomous

Web Scraping with Selenium

Selenium is a web driver that automates various web applications.

We used Selenium to automate the process of scrolling through Google Chrome's image search of each snake species,

programming it to parse through each image HTML and find the "class" code associated with the snake's species.

```
def get_images(delay, max, url):
    driver = webdriver.Chrome()
    def scroll_down(driver):
        driver.execute_script("window.scrollTo(0, document.body.scrollHeight);")
        time.sleep(delay)

    driver.get(url)

    image_urls = set()
    skips = 0
    thumbs = 0

    while len(image_urls) < max:
        print(f"Processed {thumbs} images so far.")
        curr_height = driver.execute_script("return document.body.scrollHeight")
        scroll_down(driver)
        if (driver.execute_script("return document.body.scrollHeight") == curr_height): # Reached bottom
            break
```

```
        thumbnails = driver.find_elements(By.CLASS_NAME, 'mNsIhb')
        for img in thumbnails[thumbs:max]:
            thumbs += 1
            try:
                img.click()
                time.sleep(delay)
            except:
                continue

            containers = driver.find_elements(By.CLASS_NAME, "YsLeY")
            for container in containers:
                images = container.find_elements(By.TAG_NAME, "img")
                for image in images:
                    src = image.get_attribute('src')
                    if src and (re.search(r"\..{3}$", src) or re.search(r"\..{4}$", src)) and (src not in image_urls)
                        image_urls.add(src)
                    else:
                        max += 1
                        skips += 1
    print(f"There are now {len(image_urls)} unique images.")
    driver.quit()
    return image_urls
```

Web Scraping with Selenium Cont.

After we specify a number of images we want the scraper to search for, the program downloads that many images (or as many as it can) to a local folder, which we can then use to create our dataset.

```python
watersnake_urls = get_images(1, 10000, watersnake)
urls = set()
urls = watersnake_urls
```

Python

```python
def download_img(download_path, url, file_name):
    try:
        headers = {
            "Accept": "image/*",
            "User-Agent": "Mozilla/5.0 (Macintosh; Apple Mac OS X 13_0_1) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/114.0.0.0 Safari/537.36",
        }
        content = requests.get(url, headers = headers)
        content.raise_for_status()
        file = io.BytesIO(content.content)
        image = Image.open(file)
        path = download_path + file_name + ".png"
        with open(path, "wb") as f:
            image.save(f, "PNG")
    except Exception as e:
        print("Failed -", e)
```

Python

```python
count = 746
for url in urls:
    download_img("/Users/merid/Downloads/snaks/", url, str(count))
    count += 1
```

Python

Web Scraping with BeautifulSoup

BeautifulSoup is a library that parses through the html of the page from the webdriver.

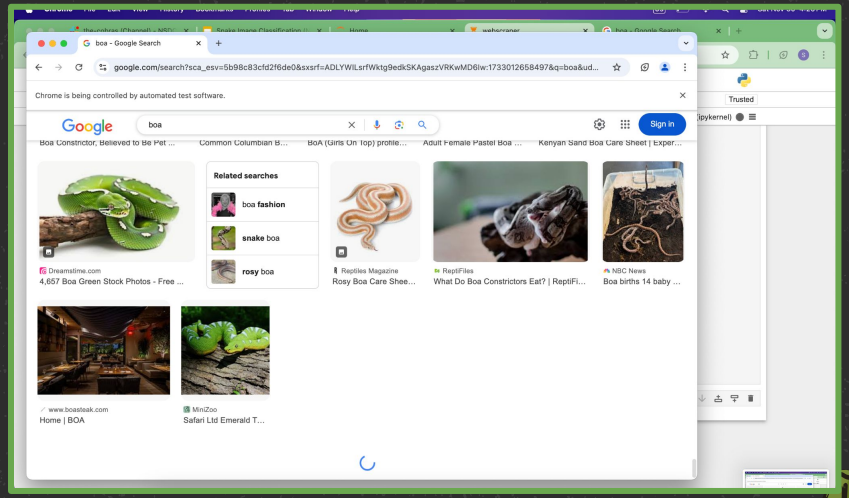
```
wd = webdriver.Chrome()

url = 'https://www.google.com/search?sca_esv=5b98c83cfd2f6de0&sxsrf=ADLYW
wd.get(url)

num_scrolls = 20
for _ in range(num_scrolls):
    wd.execute_script('window.scrollTo(0, document.body.scrollHeight);')
    time.sleep(3)

page = wd.page_source
soup = BeautifulSoup(page, 'html.parser')
boas = soup.find_all('img', class_="YQ4gaf")

boa_img = []

for boa in boas:
    boa_img.append(boa.attrs["src"])
```

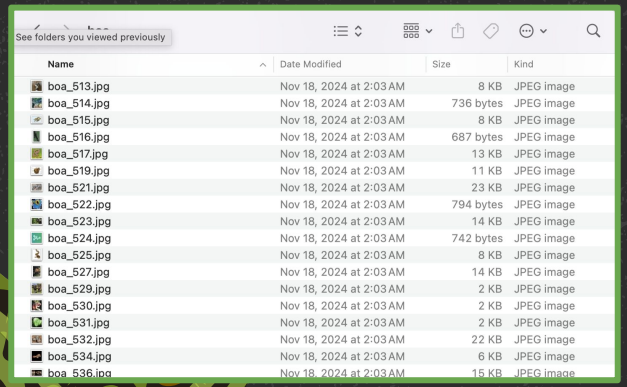
We used this to find all the img tags with the class associated with the snake.

```
<div class="F0uyec" jsdata="j0Opre;9Cf2142kIFKuBM;A9uWV0" jscontroller="qK
rDxc" jsshadow role="button" tabindex="0" jsaction="rcuQ6b:npT2md;h5M12e;j
GQF0b:kNqZ1c;mouseover:UI3Kjd" data-viewer-entrypoint="1" data-vhid="9Cf21
42kIFKuBM">
  <div jsslot>
    <div class="H8Rx8c" style="height:180px"> flex
      <g-img style="max-height:183px;max-width:275px" class="mNsIhb">
        <img id="dimg_rttOZ5SOGvH5kPIP0_HzSA_241" src="data:image/jpeg;base
        64,/9j…EQBERAEREAREQBERAEREB//2Q==" class="YQ4gaf" height="183"
        style="object-position:50% 45%" width="275" alt="Boa Care Sheet"
        data-csiid="rttOZ5SOGvH5kPIP0_HzSA_12" data-atf="1"> == $0
      </g-img>
      <div class="cC9Rib"></div>
    </div>
```

Web Scraping with BeautifulSoup

Using a similar download_img function, Selenium opens and downloads the images from the uri that begins with “data:image/”.

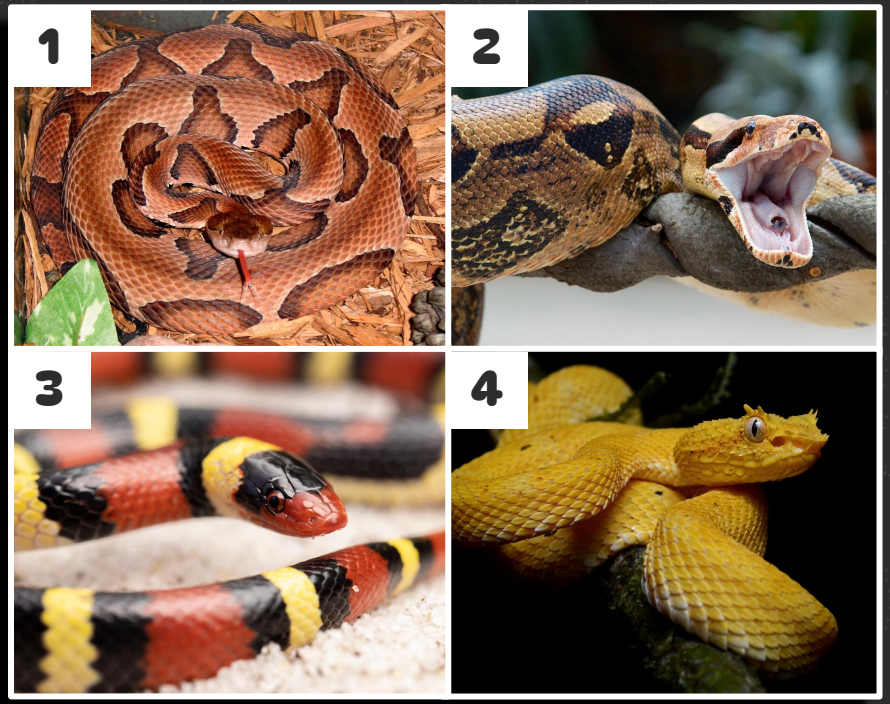




11 Snake Species Collected

1. Copperhead
2. Boa Constrictor
3. Coral Snake
4. Yellow Eyelash Viper

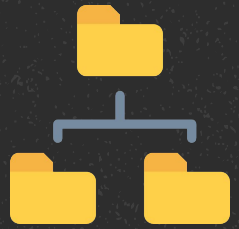
...and 7 more!

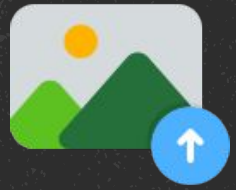


Converting to a Dataset

To be able to input the data into a model, have to convert it to the proper format.

1. Format the images in the folder properly
2. Upload from the directory
3. Rescale all of the images to the same size for consistency
4. Separate into train, validation, and test sets





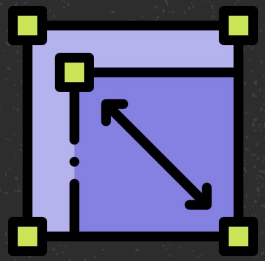
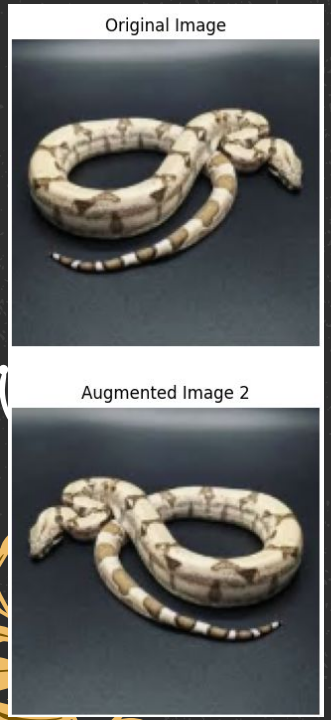


Image Classification: Data Augmentation



Data augmentation is the process of making modifications to existing training data to increase its diversity and exposure

CODE

```
IMG_SHAPE = (150, 150, 3)
base_model = keras.applications.EfficientNetB0(input_shape=IMG_SHAPE,
                                               include_top=False,
                                               weights='imagenet')
# For MobileNetV2: alpha = 1.4
base_model.trainable = False

i = keras.Input(shape=IMG_SHAPE)
x = base_model(i, training = False)
base_model_layer = keras.Model(inputs = i, outputs = x)

model4 = models.Sequential([
    # augmentation layers
    layers.RandomFlip("horizontal", input_shape=IMG_SHAPE),
    layers.RandomFlip("vertical", input_shape=IMG_SHAPE),
    layers.RandomRotation(0.2),

    base_model_layer,
    layers.GlobalAveragePooling2D(),

    #layers.Dense(512, activation='relu'),
    layers.Dropout(0.3),
    layers.Dense(512, activation='relu'),
    #layers.Dropout(0.1),
    layers.Dense(10, activation='softmax'),  # outputs the final classification
])
```

```
[33]: test_loss, test_accuracy = model4.evaluate(test)
      print(f'Test Accuracy: {test_accuracy * 100:.2f}%')

      10/10 ━━━━━━━━━━━━━━━━━━━━ 16s 754ms/step - accuracy: 0.7778 - loss: 0.7654
      Test Accuracy: 80.00%
```

Image Classification: The Model

Input

Feature Extraction

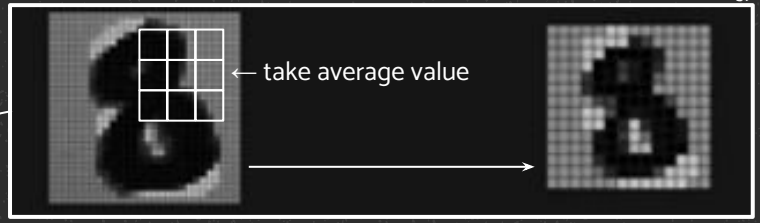
Classification

- Augmentation
- EfficientNetB0 (pre-trained)
- Average Pooling Layer
- 2 Dropout Layers
- 2 Dense Layers

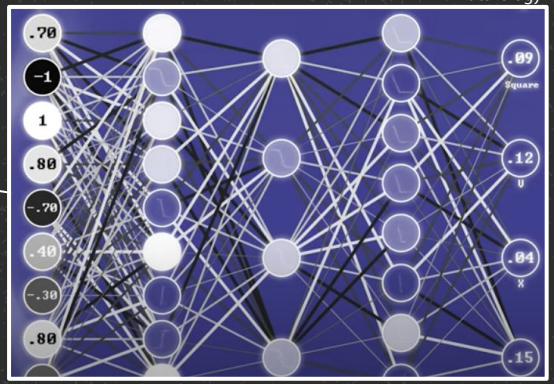
Output: pit viper

3Blue1Brown

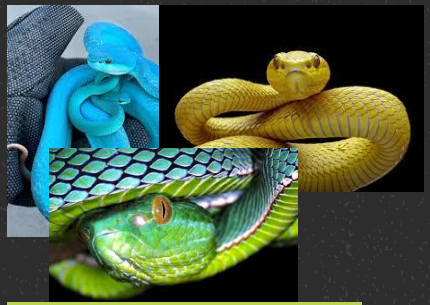
Futurology



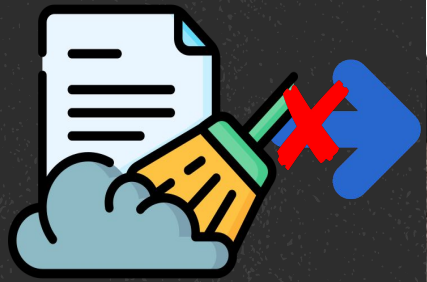
Futurology



Potential Improvements

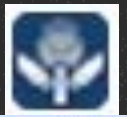


More images!



Clean the data!

```
class="YQ4gaf"
```



```
class="YQ4gaf zr758c"
```

Similar Parent Class!

Conclusion

Using **web scraping** with **selenium** and **image classification**, in this project, we were given an image of a snake and our model could identify which species it was and whether or not it was venomous

Real-world applications of our model:

- Aid in public travel services by quickly identifying venomous vs non-venomous snakes
- Opens up new opportunities for automated animal identification

Resources

- Build a Deep CNN Image Classifier with ANY Images - YouTube
- https://www.youtube.com/watch?v=i_LwzRVP7bg&t=532s