Kyle Dalope
ADS508-02-SP23

# Assignment 1.1: Using Amazon Rekognition

**Section 1: The unsafe words detected in the default image.**
For the text detection of the image, in the cell titled *Display list of detected unsafe text,* what are the 3 words that were detected?

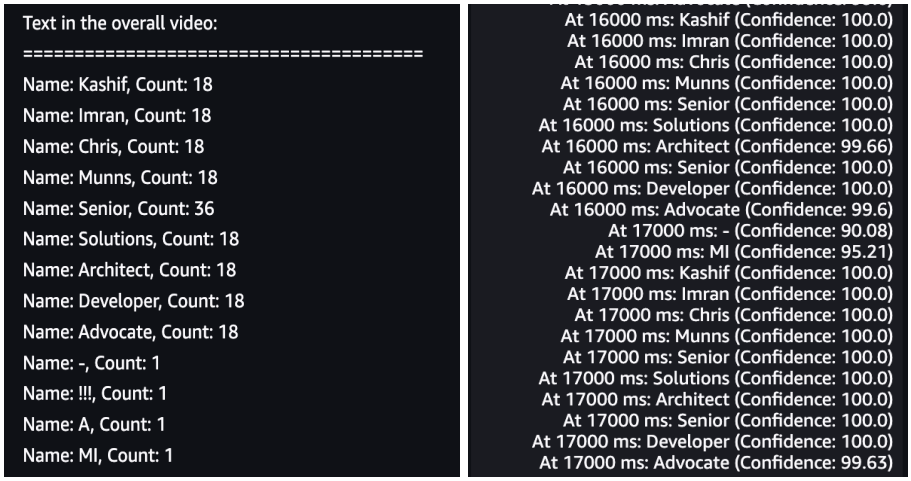
*Section 1 Answer*

- Detected unsafe word: damm
- Detected unsafe word: darn
- Detected unsafe word: crap


**Section 2: The timestamps and confidence of the words AWS and Twitter in the video.**
For the text detection in the video, in the cell titled *Display Recognized Text in the Video*, what are the timestamps and confidence of the word AWS and Twitter in the video?

*Section 2 Answer*

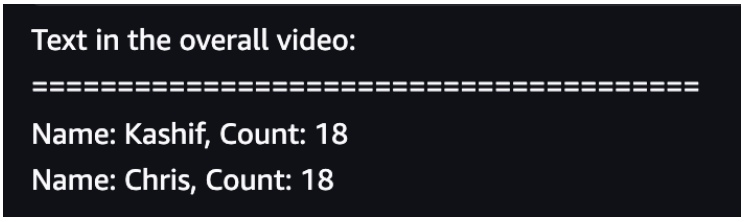
- After initially running the *05_Inappropriate_Text_Detection.ipynb*, the output did not include AWS or Twitter and only observed the following seen below:



- Updated this code block by adding the filters from detect_text (from Image text detection) to this video text detection.

## Call Rekognition to start a job for text detection

```
[23]: startTextDetection = rekognition.start_text_detection(
          Video={
              "S3Object": {
                  "Bucket": bucket,
                  "Name": videoName,
              }
          },
          Filters={
              "WordFilter": {"MinConfidence": 70, "MinBoundingBoxHeight": 0.05, "MinBoundingBoxWidth": 0.02},
              "RegionsOfInterest": [
                  {"BoundingBox": {"Width": 0.1, "Height": 0.05, "Left": 0.01, "Top": 0.01}},
              ],
          },
      )

      textJobId = startTextDetection["JobId"]
      display("Job Id: {0}".format(textJobId))

'Job Id: 74d03c8f655a153b015b12a75fd1b684e9b650df4ee36a79dc25b07581e96b20'
```

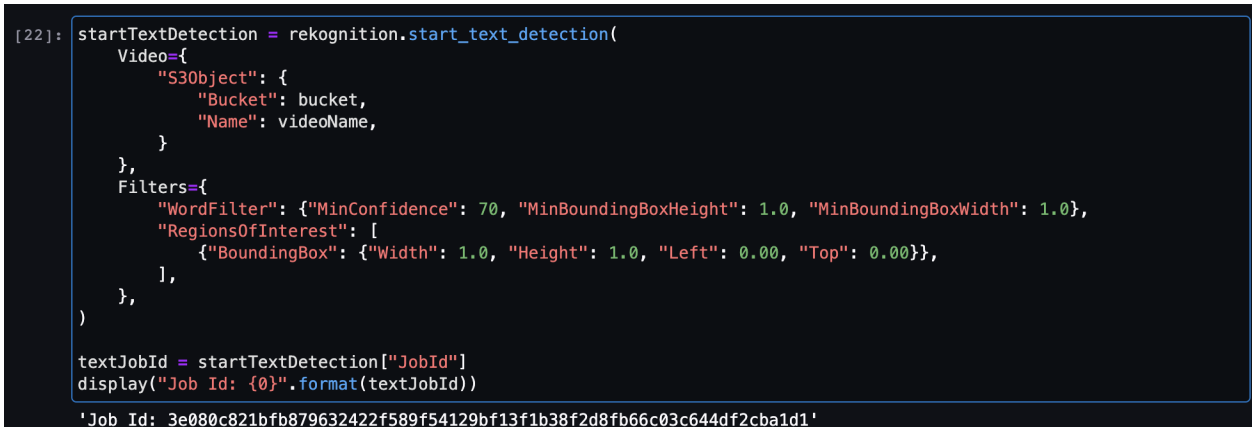
  - I've attempted to remove the "RegionsOfInterest" of the bounding box to see if I would be able to identify the words within the "WordFilter" but it only resulted as:
  - 


- Next, I will adjust the bounding box to see if that will help:

```
[20]: startTextDetection = rekognition.start_text_detection(
          Video={
              "S3Object": {
                  "Bucket": bucket,
                  "Name": videoName,
              }
          },
          Filters={
              "WordFilter": {"MinConfidence": 70, "MinBoundingBoxHeight": 0.05, "MinBoundingBoxWidth": 0.02},
              "RegionsOfInterest": [
                  {"BoundingBox": {"Width": 1.0, "Height": 1.0, "Left": 0.00, "Top": 0.00}},
              ],
          },
      )

      textJobId = startTextDetection["JobId"]
      display("Job Id: {0}".format(textJobId))

'Job Id: ede8608db7f5e86d3d94b2ccf94e853e63777186146ccfaf382570fff74eb593'
```

  - Same results.

- Adjusted the code more and the output contains no words now.



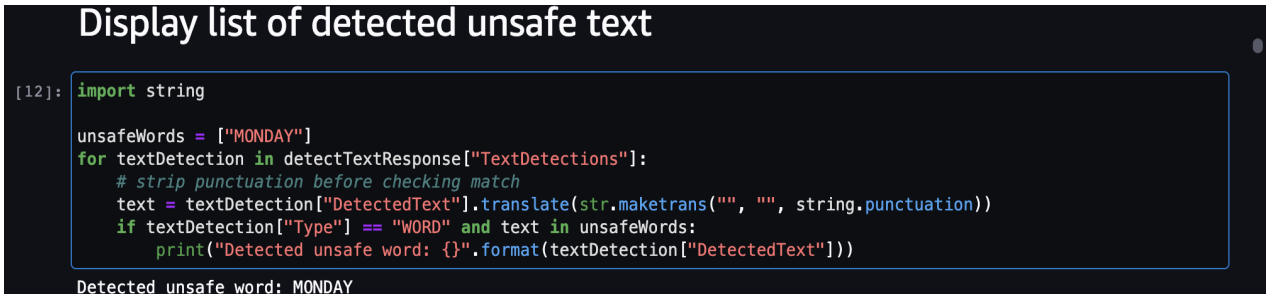
```
[22]: startTextDetection = rekognition.start_text_detection(
          Video={
              "S3Object": {
                  "Bucket": bucket,
                  "Name": videoName,
              }
          },
          Filters={
              "WordFilter": {"MinConfidence": 70, "MinBoundingBoxHeight": 1.0, "MinBoundingBoxWidth": 1.0},
              "RegionsOfInterest": [
                  {"BoundingBox": {"Width": 1.0, "Height": 1.0, "Left": 0.00, "Top": 0.00}},
              ],
          },
      )

      textJobId = startTextDetection["JobId"]
      display("Job Id: {0}".format(textJobId))

      'Job Id: 3e080c821bfb879632422f589f54129bf13f1b38f2d8fb66c03c644df2cba1d1'
```

- 

**Section 3: After adding MONDAY to the unsafe world list, insert a screenshot of the cell titled Display list of detected unsafe text.**

Adjust the unsafe word list for the image text recognition to add MONDAY as an unsafe world. Do this by adjusting the cell titled *Display list of detected unsafe text*, and add MONDAY to the list unsafeWords (be sure you add MONDAY as all caps). Execute the cell and take a screenshot of the cell + the output, and add it to your Word document.

*Section 3 Answer*



Display list of detected unsafe text

```
[12]: import string

      unsafeWords = ["MONDAY"]
      for textDetection in detectTextResponse["TextDetections"]:
          # strip punctuation before checking match
          text = textDetection["DetectedText"].translate(str.maketrans("", "", string.punctuation))
          if textDetection["Type"] == "WORD" and text in unsafeWords:
              print("Detected unsafe word: {}".format(textDetection["DetectedText"]))

      Detected unsafe word: MONDAY
```

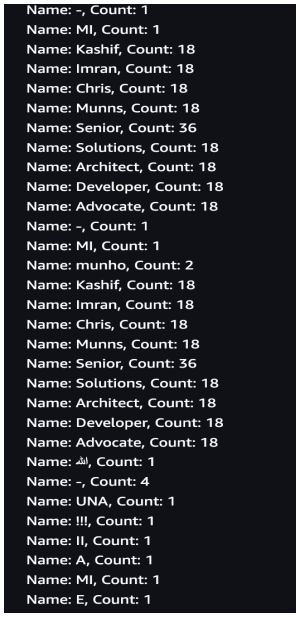
**Section 4: After adjusting the flagged text word list for the video text recognition in the cell titled Display Recognized Text in the Video, insert a screenshot of the cell Display Recognized Text in the Video.**

Adjust the flagged text word list for the video text recognition to only flag Firehose. Do this by adjusting the cell titled *Display Recognized Text* in the Video and updating the list flaggedTextInVideo to have only one value, Firehose. Be sure you add Firehose as the title case (capital F lowercase irehose). Execute the cell and take a screenshot of the cell + the "Flagged text output," and add it to your .PDF.

*Section 4 Answer*

Attempted with MinConfidence of 50:

- 

- 

Attempted to adjust the filters and confidence again (Confidence: 80):

- [66]:
```
flaggedTextInVideo = ["Firehose"]

theLines = {}

# Objects detected in each frame
for obj in getTextDetection["TextDetections"]:
    if obj["TextDetection"]["Type"] == "WORD":
        ts = obj["Timestamp"]
        cconfidence = obj["TextDetection"]["Confidence"]
        oname = obj["TextDetection"]["DetectedText"]

        if oname in flaggedTextInVideo:
            print("Found flagged text at {} ms: {} (Confidence: {})".format(ts, oname, round(cconfidence, 2)))

        strDetail = strDetail + "At {} ms: {} (Confidence: {})<br>".format(ts, oname, round(cconfidence, 2))
        if oname in theLines:
            cojb = theLines[oname]
            theLines[oname] = {"Text": oname, "Count": 1 + cojb["Count"]}
        else:
            theLines[oname] = {"Text": oname, "Count": 1}

# Unique objects detected in video
for theLine in theLines:
    strOverall = strOverall + "Name: {}, Count: {}<br>".format(theLine, theLines[theLine]["Count"])

# Display results
display(HTML(strOverall))
```
  Text in the overall video:
  ---

- [63]:
```
startTextDetection = rekognition.start_text_detection(
    Video={
        "S3Object": {
            "Bucket": bucket,
            "Name": videoName,
        }
    },
    Filters={
        "WordFilter": {"MinConfidence": 80, "MinBoundingBoxHeight": 0.05, "MinBoundingBoxWidth": 0.02},
        "RegionsOfInterest": [
            {"BoundingBox": {"Width": 0.1, "Height": 0.1, "Left": 0.1, "Top": 0.1}},
        ],
    },
)

textJobId = startTextDetection["JobId"]
display("Job Id: {0}".format(textJobId))
```
  'Job Id: f6ba11ce49c57af4f90e681e472303722dfa3f1e83f90b57adf235d58f246e4d'

---

Name: Kashif, Count: 18
Name: Chris, Count: 18
Name: Kashif, Count: 18
Name: Imran, Count: 18
Name: Chris, Count: 18
Name: Munns, Count: 18
Name: Senior, Count: 36
Name: Solutions, Count: 18
Name: Architect, Count: 18
Name: Developer, Count: 18
Name: Advocate, Count: 18
Name: -, Count: 1
Name: MI, Count: 1

Text detected in video
---
At 0 ms: Kashif (Confidence: 99.94)
At 0 ms: Chris (Confidence: 100.0)
At 1000 ms: Kashif (Confidence: 99.94)
At 1000 ms: Chris (Confidence: 100.0)
At 2000 ms: Kashif (Confidence: 99.95)
At 2000 ms: Chris (Confidence: 100.0)
At 3000 ms: Kashif (Confidence: 99.94)
At 3000 ms: Chris (Confidence: 100.0)
At 4000 ms: Kashif (Confidence: 99.94)
At 4000 ms: Chris (Confidence: 100.0)
At 5000 ms: Kashif (Confidence: 99.95)
At 5000 ms: Chris (Confidence: 100.0)
At 6000 ms: Kashif (Confidence: 99.93)
At 6000 ms: Chris (Confidence: 100.0)
At 7000 ms: Kashif (Confidence: 99.92)
At 7000 ms: Chris (Confidence: 100.0)
At 8000 ms: Kashif (Confidence: 99.92)
At 8000 ms: Chris (Confidence: 100.0)
At 9000 ms: Kashif (Confidence: 99.91)
At 9000 ms: Chris (Confidence: 100.0)
At 10000 ms: Kashif (Confidence: 99.94)
At 10000 ms: Chris (Confidence: 100.0)
At 11000 ms: Kashif (Confidence: 99.94)
At 11000 ms: Chris (Confidence: 100.0)
At 12000 ms: Kashif (Confidence: 99.94)
At 12000 ms: Chris (Confidence: 100.0)
At 13000 ms: Kashif (Confidence: 99.94)
At 13000 ms: Chris (Confidence: 100.0)
At 14000 ms: Kashif (Confidence: 100.0)
At 14000 ms: Chris (Confidence: 100.0)
At 15000 ms: Kashif (Confidence: 99.96)
At 15000 ms: Chris (Confidence: 100.0)
At 16000 ms: Kashif (Confidence: 100.0)
At 16000 ms: Chris (Confidence: 100.0)
At 17000 ms: Kashif (Confidence: 100.0)
At 17000 ms: Chris (Confidence: 100.0)

In summary, I found updating the code block for the image easier to adjust rather than the video. I found it a little more difficult to put the correct bounding box parameters to locate the words of interest. I will have to perform more examples and review the references provided further to be more accurate with the parameters when using Rekognition with video files.