ASSIGNMENT 2: REPORT *(TEAM 06)*

PART 1: British UFO Sightings Data

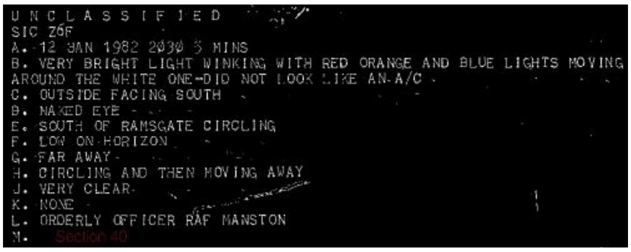
Imagemagick

We tried a lot of options while using Imagemagick. We also tried running various scripts from this blog: http://www.fmwconcepts.com/imagemagick/index.php. We decided on the following command to convert our split *.pdf*s to *.jpg*:

```
magick -density 300 $filename_no_extension/$the_file -depth 8 -background white -flatten
-alpha OFF -resize 50% $filename_no_extension/tiff/$the_file_noext.jpg
```

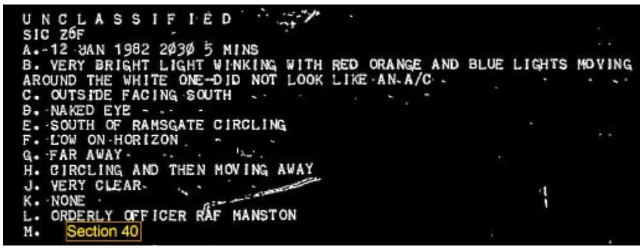
*magick is the Windows equivalent of convert

However, the quality of the image was suboptimal, and tesseract did not perform well. So, we used the **Windows Picture Manager** for setting the values of ***brightness (-19), contrast (39) and midtone values (-100)*** on the *.jpg* files generated from Imagemagick. Running the OCR on these doctored images gave us much better results.



UNCLASSIFIED
SIC Z6F
A. 12 JAN 1982 2030 5 MINS
B. VERY BRIGHT LIGHT WINKING WITH RED ORANGE AND BLUE LIGHTS MOVING AROUND THE WHITE ONE-DID NOT LOOK LIKE AN A/C
C. OUTSIDE FACING SOUTH
D. NAKED EYE
E. SOUTH OF RAMSGATE CIRCLING
F. LOW ON HORIZON
G. FAR AWAY
H. CIRCLING AND THEN MOVING AWAY
J. VERY CLEAR
K. NONE
L. ORDERLY OFFICER RAF MANSTON
M. Section 40

Fig 1(a): Before Windows Picture Manager



UNCLASSIFIED
SIC Z6F
A. 12 JAN 1982 2030 5 MINS
B. VERY BRIGHT LIGHT WINKING WITH RED ORANGE AND BLUE LIGHTS MOVING AROUND THE WHITE ONE-DID NOT LOOK LIKE AN A/C
C. OUTSIDE FACING SOUTH
D. NAKED EYE
E. SOUTH OF RAMSGATE CIRCLING
F. LOW ON HORIZON
G. FAR AWAY
H. CIRCLING AND THEN MOVING AWAY
J. VERY CLEAR
K. NONE
L. ORDERLY OFFICER RAF MANSTON
M. Section 40

Fig 1(b): After Windows Picture Manager

OCR with Tesseract

Tesseract provided good outputs on typewritten reports. However, it performed poorly on handwritten texts, especially cursive. We re-built tesseract with an LSTM model using language models from the tesseract repository, and ran commands in this fashion:

```
tesseract $filename_no_extension/$the_file $out_file_no_extension/$the_file_noext - --oem 1
-l eng
```

Noise Removal

This involved removing punctuation (except `[,:/]`) and other unintelligible text. We also used the `autocorrect` spellcheck library in Python to correct spelling errors. This is handled by the `parser_main.py` script.

```
A«~12 -4AN 1982 2030 5 MINS


B. VERY BRIGHT LIGHT WINKING WITH RED ORANGE AND BLUE LIGHTS noving

AROUND THE WHITE ONE-DID NOT LOOK
C. OUTSIDE FACING SOUTH --

B. NAKED EYE - .- _-

E. -SOUTH OF RAMSGATE

F. LOW ON-HORIZON . e

G. FAR AVAY- B Ins

H. CIRCLING AND THEN MOV ING AWAY
Je VERY CLEAR .; - Pe
K. «NONE - a*

L. orbercy orpicer RAF manston
M. (Section 40
```

Fig 2(a): Output from Tesseract

```
A 12 AN 1982 2030 5 MIND

B. VERY BRIGHT LIGHT WINKING WITH RED ORANGE AND BLUE LIGHTS MOVING

AROUND THE WHITE ONE DID NOT LOOK

C. OUTSIDE FACING SOUTH

B. NAKED EYE

E. SOUTH OF RAMSGATE

F. LOW ON HORIZON . E

G. FAR AWAY B IN

H. CURLING AND THEN MOVE ING AWAY

JE VERY CLEAR . PE

K. NONE A
```

Fig 2(b): Output after noise removal/cleaning

Parsing OCR Output with Tika

The strategy for the TIKA parser was to come up with 3 MIME Types and 3 parsers for parsing content from 3 different types OCR output. We built the TIKA parser after introducing [`UFOParser.java`](UFOParser.java), [`UFOGenericParser.java`](UFOGenericParser.java) and [`UFOTabularParser.java`](UFOTabularParser.java). We leveraged TIKA's `AutoDetectParser` and its batch processor to generate XML output for each `*.<type>.ufo` file. We defined three new glob patterns in [`tika-mimetypes.xml`](tika-mimetypes.xml). TIKA's `AutoDetectParser` automatically assigns each of the `*.<type>.ufo` file to its corresponding parser in the parsing pipeline. [`parser_main.py`](parser_main.py) invokes TIKA's batch processor for every OCR output text file. Regular expressions were used for parsing the dates and

```
<mime-type type="text/listufo">
  <_comment>Used for Parsing UFO List
  <glob pattern="*.list.ufo"/>
  <sub-class-of type="text/plain"/>
</mime-type>

<mime-type type="text/genericufo">
  <_comment>Used for Parsing UFO Gene
  <glob pattern="*.generic.ufo"/>
  <sub-class-of type="text/plain"/>
</mime-type>

<mime-type type="text/tabularufo">
  <_comment>Used for Parsing UFO Gene
  <glob pattern="*.tabular.ufo"/>
  <sub-class-of type="text/plain"/>
</mime-type>
```

duration. Stanford CoreNLP was used for extracting location. We used a custom list of shapes to identify shapes in the input.

Fig 3(a): edited `tika-mimetypes.xml`

```
    <meta content="59.list.ufo" name="tika_batch_fs:rela
    <meta content="59.list.ufo" name="resourceName"/>
    <meta content="612" name="Content-Length"/>
    <meta content="b09948d059a92be0bdbb59ff674c8d9d" nam
    <meta content="text/listufo" name="Content-Type"/>
    <title>
    </title>
</head>
<body>
    <description>A HUSH OR VERY BRIGHT REO LIGHTS NONE N
    <duration>""</duration>
    <location>GLASTONBURY SOMERSET,GLASTONBURY</location
    <report-date>25 DEC 81</report-date>
    <sighting-date>DEC 81</sighting-date>
    <shape>BRIGHT</shape>
```

Fig 3(b): Output of our custom TIka parser

[UFO.java]() and [UFODate.java]() are created for capturing the UFO data.

Our Thoughts on the OCR Pipeline

- Imagemagick was a challenge since we needed to experiment quite a bit to settle on the best commands to use.
- The default version of Tesseract doesn't perform well, so we had to re-build tesseract with LSTM version 4.0.0, and used additional testdata_best models to achieve better recognition using Tesseract.
- The British UFO dataset is very noisy and inconsistent. We primarily identified two types of reports - lists and tabular data, and had to write some heavy preprocessing code and multiple Tika parsers to cater to each type of report.

Insights and Inferences from the British UFO Dataset

- Since most of the data from the first assignment is based on sightings in the United States, and since most of our inferences were based on queries on the data aggregated by the state, there was no easy way to compare our inferences and conclusions across datasets.
- **New object types identified**
  'Arc', 'Boomerang', 'Bucket', 'Bullet', 'Disc', 'Helicopter', 'Missile', 'Rectangle', 'Square', 'Star'

```
In [30]: df.isna().sum().sum() / df.size
Out[30]: 0.4905708460754332
```

Fig 4: Running pandas query on our sparse data

- **How well the sightings were described**
  1308 records were retrieved from ***1968*** PDF pages (~72% of the data). Unfortunately, this extracted data is sparse - around **50%** of the entries are missing. The columns extracted are *location*, *duration*, *sighted date*, *reported date*, *shape*, and *description.*

PART 2: UFO Stalker

Web Scraping

Using selenium and angularJS, we developed a script that scraped every single event in the history of the UFO Stalker website. We scraped over ***87,000*** records, from which we further extracted over ***1,500*** images to re-train our tensorflow image detection models.

Object Detection and Image Captioning

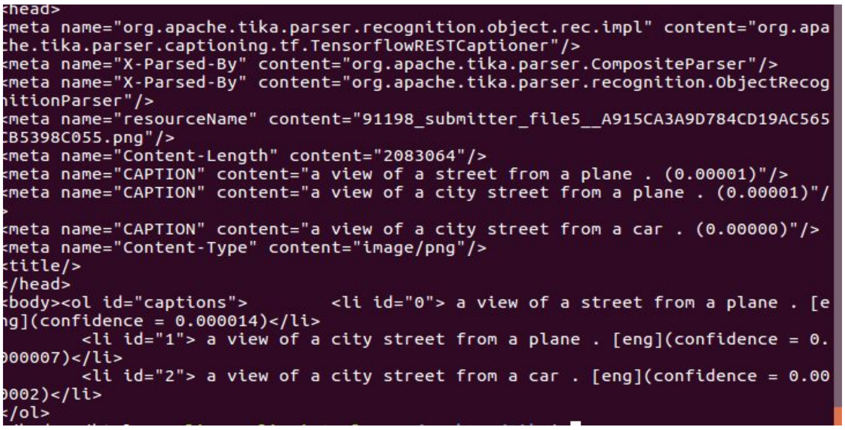
Two scripts were developed for the purpose of running the image detection models - `objects.py` (generates object tags for images) and `captioner.py` (generates image captions). These scripts were run once the corresponding docker images has been built and made to run on localhost (port 8764).Did the runs with and without dockers through tika to generate both HTML and JSON outputs. Variations on parameters in XML file.

1. *Parameters for objects*
   - `topN` - Returns the top N results in descending order of confidence values
   - `minConfidence` measure - Only output results with confidence measure >= this value. The default is **0.015** (1.5% confidence). This measure can be set to as high as **0.03** beyond which all results are filtered out (generating null values).

```
<head>
<meta name="org.apache.tika.parser.recognition.object.rec.impl" content="org.apache.tika.parser.recognition.tf.TensorflowRESTRecogniser"/>
<meta name="X-Parsed-By" content="org.apache.tika.parser.CompositeParser"/>
<meta name="X-Parsed-By" content="org.apache.tika.parser.recognition.ObjectRecognitionParser"/>
<meta name="resourceName" content="91154_submitter_file6__IMG2055.PNG"/>
<meta name="Content-Length" content="2111506"/>
<meta name="OBJECT" content="hook, claw (0.03186)"/>
<meta name="Content-Type" content="image/png"/>
<title/>
</head>
<body><ol id="objects"> <li id="hook, claw"> hook, claw [eng](confidence = 0.031858)</li>
</ol>
```

Fig 5:Object Output [/files_jeud8334j/91154_submitter_file6__IMG2055.PNG]()

1. *Parameters for captions*
    - `Captions` - number of captions returned in descending order of confidence measure.descending order of confidence values
    - `maxCaptionLength` - maximum length which defaults to 15
    1. a view of a street from a plane.[eng](confidence = 0.000014)
    2. A view of a city street from a plane. [eng](confidence = 0.000007)
    3. A view of city street from a car.[eng] (confdence = 0.000002)



```
head>
meta name="org.apache.tika.parser.recognition.object.rec.impl" content="org.apa
che.tika.parser.captioning.tf.TensorflowRESTCaptioner"/>
meta name="X-Parsed-By" content="org.apache.tika.parser.CompositeParser"/>
meta name="X-Parsed-By" content="org.apache.tika.parser.recognition.ObjectRecog
nitionParser"/>
meta name="resourceName" content="91198_submitter_file5__A915CA3A9D784CD19AC565
CB5398C055.png"/>
meta name="Content-Length" content="2083064"/>
meta name="CAPTION" content="a view of a street from a plane . (0.00001)"/>
meta name="CAPTION" content="a view of a city street from a plane . (0.00001)"/

meta name="CAPTION" content="a view of a city street from a car . (0.00000)"/>
meta name="Content-Type" content="image/png"/>
title/>
/head>
body><ol id="captions">        <li id="0"> a view of a street from a plane . [e
ng](confidence = 0.000014)</li>
        <li id="1"> a view of a city street from a plane . [eng](confidence = 0.
000007)</li>
        <li id="2"> a view of a city street from a car . [eng](confidence = 0.00
0002)</li>
/ol>
```

Fig 6: Caption Output /files_jeud8334j/91154_submitter_file6__IMG2055.PNG

Q Did the image captions accurately describe the UFO object types? What about the identified objects?
The captions were generic and not accurate as they were generated from the bag of words based on the corresponding labeled object. The object was identified in the image but not labeled accurately since the ImageNet corpus on which it has been trained does not include specific labels for UFO-type objects.
Eg: a large jetliner flying over a city under a cloudy sky .
It can be intuitively observed that the object is being identified but being labeled inaccurately leading to the captioner returning a more generic result based on the object identified from the limited dataset it is trained on.

Q Of the UFO images, how many of the images actually generated image captions and/or objects that described the UFO and not just the background scenery?
It was moderately fair in detection of the object and not just the background scenery, however it was not able to label it correctly in most cases(eg: balloon,parachute,jellyfish,nematode) owing to the fact that the ImageNet dataset on which it has been trained does not include labeling for the UFO-type objects. In some cases it detected just the background scenery like lakeside, shore etc. This is further reflected in the captions too.

Q Include your thoughts about and Image Captioning/Object identification what was easy about using it? What wasn't?
Tensorflow is well integrated in Tika and can be utilized with the Docker Images efficiently. However the setup and integration of docker alongwith the tika libraries was challenging and required some rework.

PART 3: Inferences

Inferences from our Combined Dataset

We noticed that by adding the new dataset, we can get more insights into how people over the world view UFOs. Right from duration of observation to the climatic conditions during which they were observed(like in the British UFO files!). These can help us look at similarities in sightings based on shape the UFO was observed. Moreover a trend can be observed as to how many days people wait till they report the UFO they saw
To summarize global sightings among,the following bar graph shows top 10 countries with reported sightings

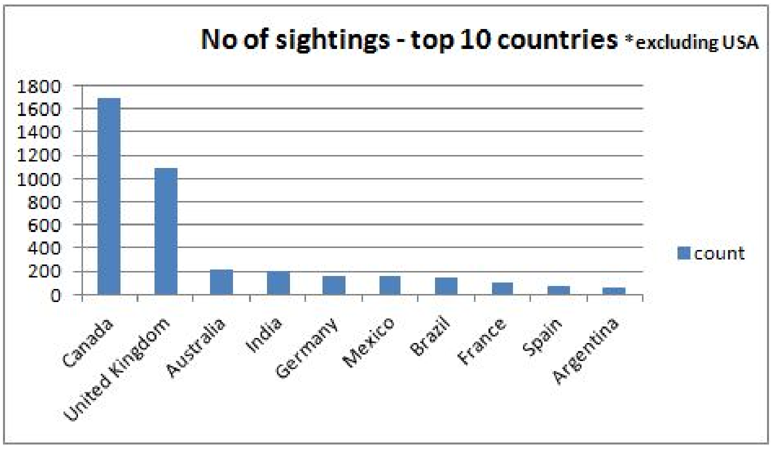


Fig. Global sightings

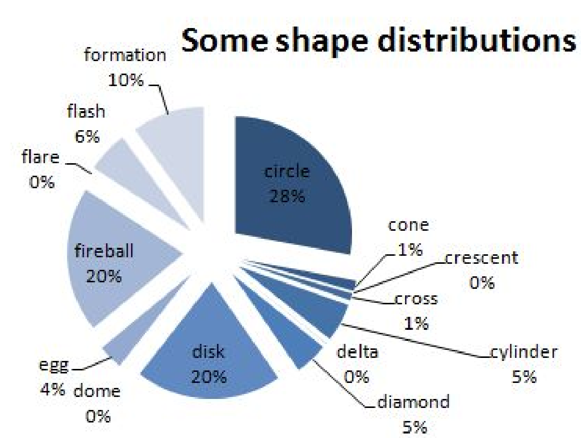


Fig. More insight into different shape type distribution

**Note: All .csv files on GitHub are compressed and need to be decompressed before being used*

PART 4: Tika NER

After receiving the new version of our dataset, we ran CoreNLP with Tika to extract all the Named Entities in the description. A column by name "ner" has been added to our dataset that contains the details about the NER that have been extracted from the description column. This was bundled into *NERRecognizer.jar.* A snapshot of the sample output is as under

```
event_id,ner
16974.0,{PERSON=[eversole]}
16971.0,{LOCATION=[montana]}
16970.0,"{LOCATION=[tennessee, virginia, abingdon], ORGANIZATION=[mufon],
16969.0,"{ORGANIZATION=[nikon], PERCENT=[100%]}"
16967.0,{}
16966.0,"{DATE=[saturday], TIME=[midnight, night]}"
16964.0,{}
16961.0,{DATE=[fall of 2008]}
```

We were able to execute OpenNLP on sample data and the details have been incorporated under part 4 in the readme file. The tika-NER folder contains the .bat files for executing OpenNLP and CoreNLP on sample files.
However, we also encountered instances when CoreNLP failed to recognize NERs (e.g it failed to identify "new mexico" in a description string)

PART 5: TensorFlow Re-Training

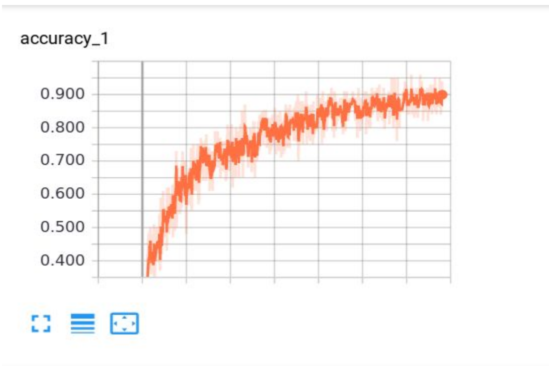


Fig 7:TensorBoard: Accuracy

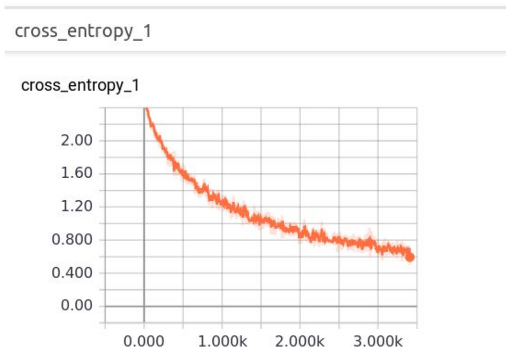


Fig 8.TensorBoard: Cross Entropy

The object detection model is re-trained using the UFO objects Shapes as the new object classifier. The retrained model generates more relevant object tags based on the UFO shapes. The accuracy is visualized in the above TensorBoard output. We have trained on **12 shape classes** with training data size of 1500 images. These include:
blimp, cigar, circle, cylinder, disc, fireball, oval, saturn like, sphere, square, star like, triangle.

Pull Request: https://github.com/USCDataScience/img2text/pull/7