Insta-Docs

Introduction

Our goals here were simple enough:

- find a way to download metadata about Instagram posts that matched our `#dogstagram` hashtag
- upload the images from the posts to Google's Vision API to analyse the images
- create a website that let us display the images/posts and let's us group and filter them by the analysed tags

What follows is an expansion on each of these steps.

Scraping Instagram for posts

The first step involved retrieving the posts from Instagram that matched the hashtag **#dogstagram**.

In order to do this we used phantombuster.

Specifically, we added the Instagram Hashtag Collector from the api-store which let us quickly configure a search to download metadata about posts that we were targeting.

Once we had let it run, we were able to download the results in a `results.json` file which we would use going forward.

Cleaning data for use

Once we had the data, we needed to "clean it" so that it would be easier to use.

As an example, the data currently was a list of posts, where each post data looked similar to:

```
    {
        "postUrl":"...",
        "profileUrl":"...",
        "username":"...",
        "fullName":"...",
        "commentCount": 10,
        "likeCount": 3000,
        "pubDate":"...",
        "description":"...",
        "imgUrl":"...",
        "postId":"..",
        "caption":"...",
        "query":"#dogstagram",
        "timestamp":"...",
        "type":"Photo"
    }
```

We didn't need all this information. We were only really focused on `postUrl` and `description` (this was the caption on Instagram).

Using some `python` we can quickly clean all our data using something similar to:

```
import json

# load the data
with open('results.json') as f:
    data = json.loads(f.read())

# create a new list where each post only has the
# url and description
cleaned_data = [
    {
        'postUrl': x['postUrl'],
        'description': x['description']
    } for x in data
]

# save the cleaned data for later use
with open('cleaned_data.json') as f:
    f.write(json.dumps(cleaned_data))
```

We now have a file `cleaned_data.json` that has a list of post basic post metadata.

Analyse images with Google Vision API

Google's Vision API is a service they offer which, according to their site:

> Google Cloud's Vision API offers powerful pre-trained machine learning models through REST and RPC APIs. Assign labels to images and quickly classify them into millions of predefined categories. Detect objects and faces, read printed and handwritten text, and build valuable metadata into your image catalog.

Simply put, it lets you analyse images using AI.

We would be using this to analyse the images from the Instagram posts that we scraped. This would let us take a look at how the AI identifies each image, but also let us group images that are similar on our website.

> ***NOTE:*** You can test the Vision API by heading over to their demo page. It'll let you upload a picture and analyse it.

In order to get started with Vision API, you can read the docs (as that's usually the best place to learn about a service/software). They include quickstart, how-tos, and in-depth looks at the API.

We used `python`, so we followed the steps for python setup.

Once we were all setup and able to contact the API, we tailer-made our code to suit our needs.

```
# import the google cloud storage client
from google.cloud import storage

# import the google cloud vision client
from google.cloud import vision
import pprint
import json

# create a vision client object that lets us interact with google's vision
# annotator
client = vision.ImageAnnotatorClient()

# load our cleaned data
with open('cleaned_data.json') as f:
    data = f.read()

images = json.loads(data)

annotations = []

total = len(images)
current = 0
# loop through all the images and annotate them
for obj in images:
    try:
        image = vision.types.Image()
        image.source.image_uri = obj['postUrl'] +'media/?size=t'
        response = client.label_detection(image=image)
        labels = response.label_annotations
        obj['annotations'] = [
            {
                'Description': x.description,
                'Likelihood': round(x.score * 100,2)
            } for x in labels
        ]
        annotations.append(obj)
        current += 1
        print '{}/{}'.format(current,total)
    except e:
        print e

# save the analysed data to file
with open('analysed_data.json', 'w') as f:
    f.write(json.dumps(annotations))
```

We now had a file `analysed_data.json` that had all our analysed data and was of the form:

```
    {
        "postUrl": "...",
        "description": "...",
        "annotations": [ ... ],
    }
```

With this, we were ready to create a basic website that would be able to display this information.

Create our Application

For creating our website, we used Vue.js. Without going into too much detail, it's a Javascript framework that lets you build reactive websites with, pretty much, minimal effort (dending on your needs of course).

The idea was simple:

- create a grid of all the images from the posts
- allow us to click on an image, which shows a popup of the full Instagram post
- allow us to filter the images based on the annotations from the Vision API
- allow us to filter the images based on text found in the captions on the posts

All in all, creating the website was simple enough (as long as you have a developer nearby they should be able to get it done). Vue makes the reactivity a breeze, meaning as soon as you filter, the page updates automatically to reflect the changes.