CS5542 Big Data Analytics and Apps

Lab Assignment#3

Name: Lakshmi Korrapati

ID: 14

Objectives: Two Objectives

- Image Caption Generator
- Data Analytics based on Unsupervised Learning

Technologies:

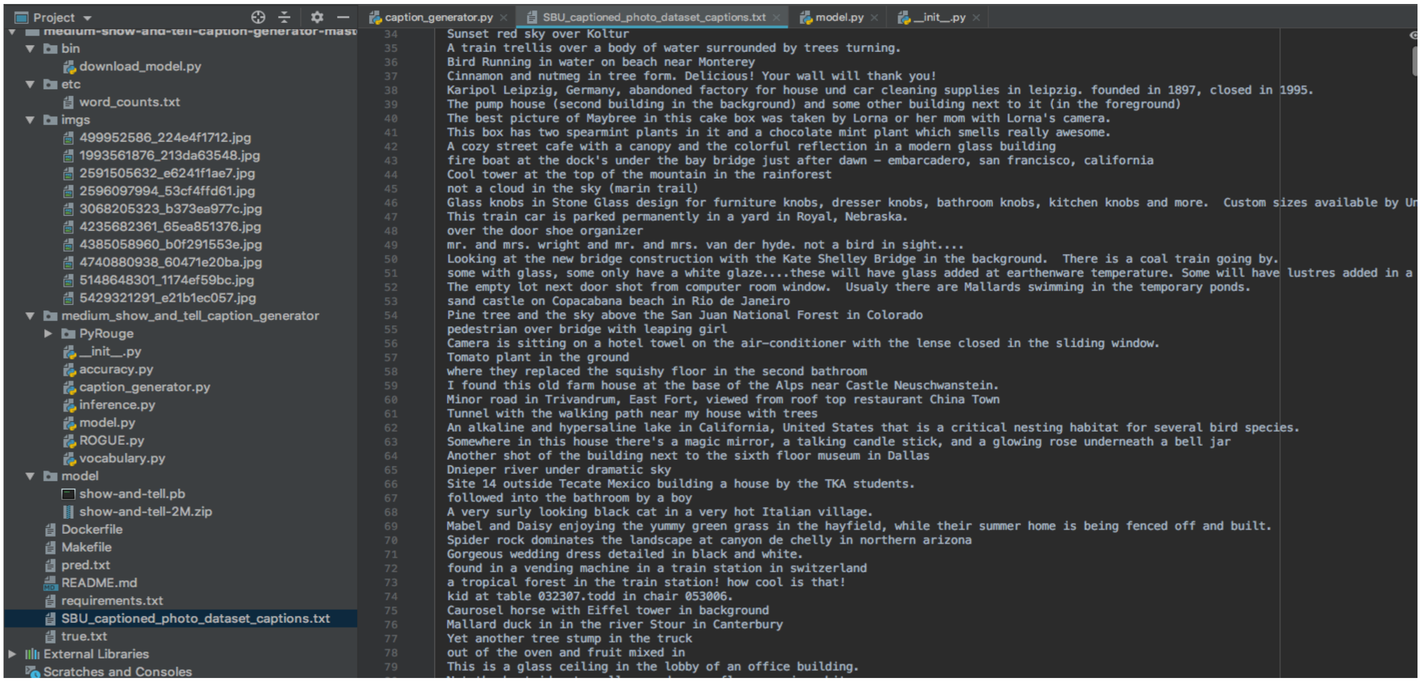
- Pycharm – IDE for executing the python files
- IntelIJ - IDE for executing the Scala files

Used Packages:

- nltk
- opencv-python
- numpy
- matplotlib
- Tensorflow
- BLEU score
- Show and tell model
- PIL
- logging
- Heapq

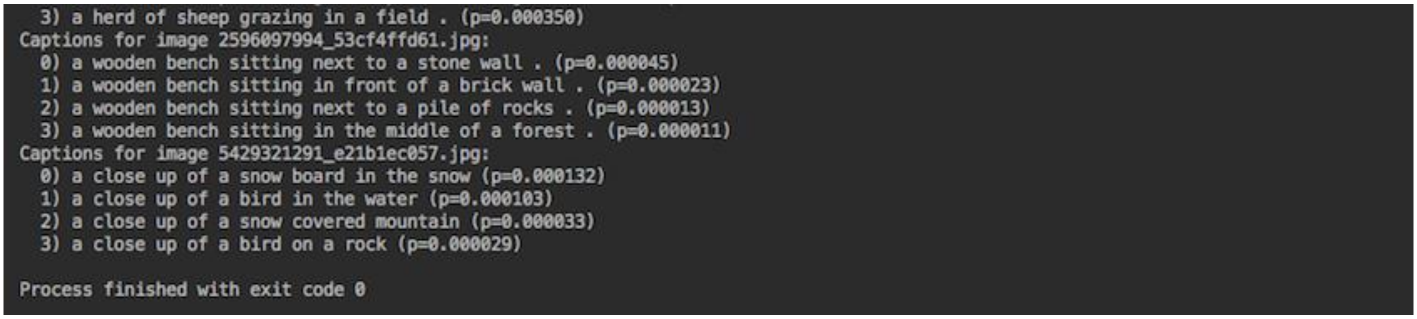
The dataset I have chosen is SBU. I have chosen SBU as it is adaptable for getting to the information. The SBU dataset have an inscription content document alongside a picture URL record which can be gotten to effectively.

Dataset:


Project
caption_generator.py
SBU_captioned_photo_dataset_captions.txt
model.py
__init__.py
medium-show-and-tell-caption-generator-mast
bin
download_model.py
etc
word_counts.txt
imgs
499952586_224e4f1712.jpg
1993561876_213da63548.jpg
2591505632_e6241f1ae7.jpg
2596097994_53cf4ffd61.jpg
3068205323_b373ea977c.jpg
4235682361_65ea851376.jpg
4385058960_b0f291553e.jpg
4740880938_60471e20ba.jpg
5148648301_1174ef59bc.jpg
5429321291_e21b1ec057.jpg
medium_show_and_tell_caption_generator
PyRouge
__init__.py
accuracy.py
caption_generator.py
inference.py
model.py
ROGUE.py
vocabulary.py
model
show-and-tell.pb
show-and-tell-2M.zip
Dockerfile
Makefile
pred.txt
README.md
requirements.txt
SBU_captioned_photo_dataset_captions.txt
true.txt
External Libraries
Scratches and Consoles

```
Sunset red sky over Koltur
A train trellis over a body of water surrounded by trees turning.
Bird Running in water on beach near Monterey
Cinnamon and nutmeg in tree form. Delicious! Your wall will thank you!
Karipol Leipzig, Germany, abandoned factory for house und car cleaning supplies in leipzig. founded in 1897, closed in 1995.
The pump house (second building in the background) and some other building next to it (in the foreground)
The best picture of Maybree in this cake box was taken by Lorna or her mom with Lorna's camera.
This box has two spearmint plants in it and a chocolate mint plant which smells really awesome.
A cozy street cafe with a canopy and the colorful reflection in a modern glass building
fire boat at the dock's under the bay bridge just after dawn - embarcadero, san francisco, california
Cool tower at the top of the mountain in the rainforest
not a cloud in the sky (marin trail)
Glass knobs in Stone Glass design for furniture knobs, dresser knobs, bathroom knobs, kitchen knobs and more.  Custom sizes available by Ur
This train car is parked permanently in a yard in Royal, Nebraska.
over the door shoe organizer
mr. and mrs. wright and mr. and mrs. van der hyde. not a bird in sight....
Looking at the new bridge construction with the Kate Shelley Bridge in the background.  There is a coal train going by.
some with glass, some only have a white glaze....these will have glass added at earthenware temperature. Some will have lustres added in a
The empty lot next door shot from computer room window.  Usualy there are Mallards swimming in the temporary ponds.
sand castle on Copacabana beach in Rio de Janeiro
Pine tree and the sky above the San Juan National Forest in Colorado
pedestrian over bridge with leaping girl
Camera is sitting on a hotel towel on the air-conditioner with the lense closed in the sliding window.
Tomato plant in the ground
where they replaced the squishy floor in the second bathroom
I found this old farm house at the base of the Alps near Castle Neuschwanstein.
Minor road in Trivandrum, East Fort, viewed from roof top restaurant China Town
Tunnel with the walking path near my house with trees
An alkaline and hypersaline lake in California, United States that is a critical nesting habitat for several bird species.
Somewhere in this house there's a magic mirror, a talking candle stick, and a glowing rose underneath a bell jar
Another shot of the building next to the sixth floor museum in Dallas
Dnieper river under dramatic sky
Site 14 outside Tecate Mexico building a house by the TKA students.
followed into the bathroom by a boy
A very surly looking black cat in a very hot Italian village.
Mabel and Daisy enjoying the yummy green grass in the hayfield, while their summer home is being fenced off and built.
Spider rock dominates the landscape at canyon de chelly in northern arizona
Gorgeous wedding dress detailed in black and white.
found in a vending machine in a train station in switzerland
a tropical forest in the train station! how cool is that!
kid at table 032307.todd in chair 053006.
Caurosel horse with Eiffel tower in background
Mallard duck in in the river Stour in Canterbury
Yet another tree stump in the truck
out of the oven and fruit mixed in
This is a glass ceiling in the lobby of an office building.
```

Output for the show and tell model:



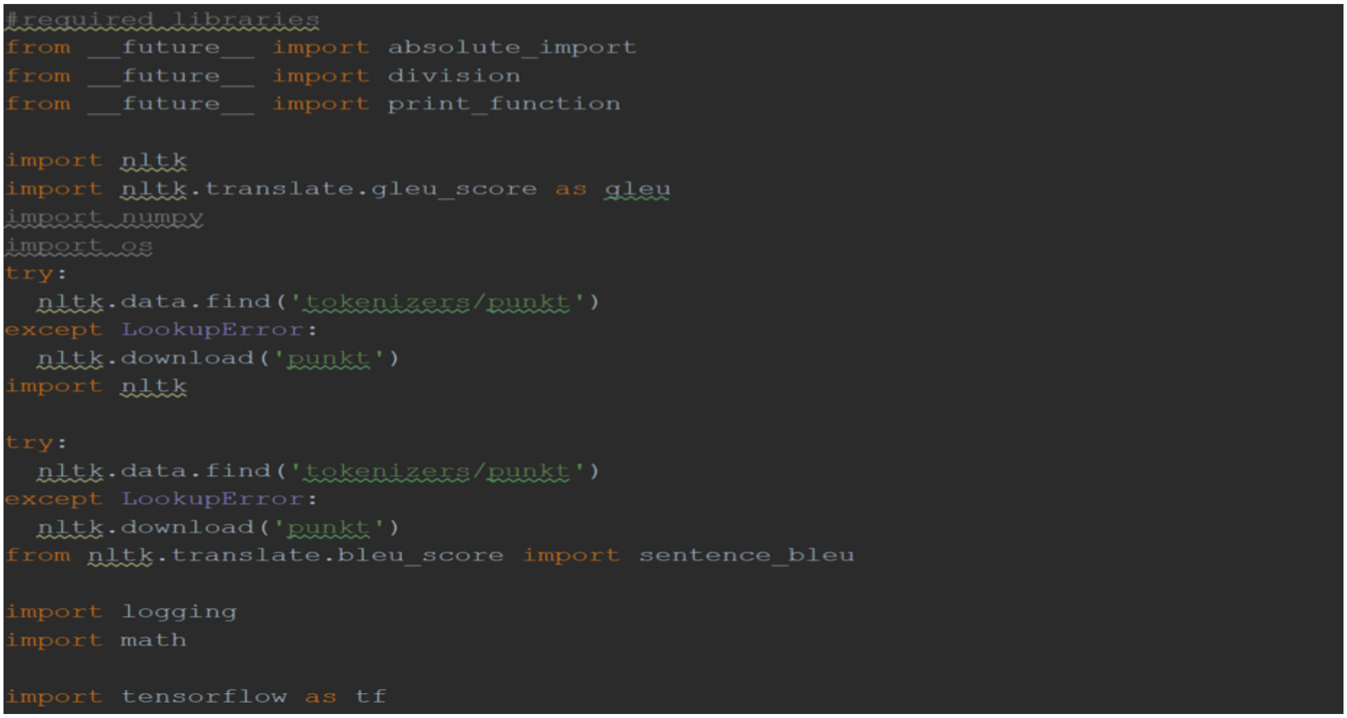
```
  3) a herd of sheep grazing in a field . (p=0.000350)
Captions for image 2596097994_53cf4ffd61.jpg:
  0) a wooden bench sitting next to a stone wall . (p=0.000045)
  1) a wooden bench sitting in front of a brick wall . (p=0.000023)
  2) a wooden bench sitting next to a pile of rocks . (p=0.000013)
  3) a wooden bench sitting in the middle of a forest . (p=0.000011)
Captions for image 5429321291_e21b1ec057.jpg:
  0) a close up of a snow board in the snow (p=0.000132)
  1) a close up of a bird in the water (p=0.000103)
  2) a close up of a snow covered mountain (p=0.000033)
  3) a close up of a bird on a rock (p=0.000029)

Process finished with exit code 0
```

Show and Tell model:

- At first when the show and tell demonstrate is executed a pb expansion document is produced which contains the model parameters

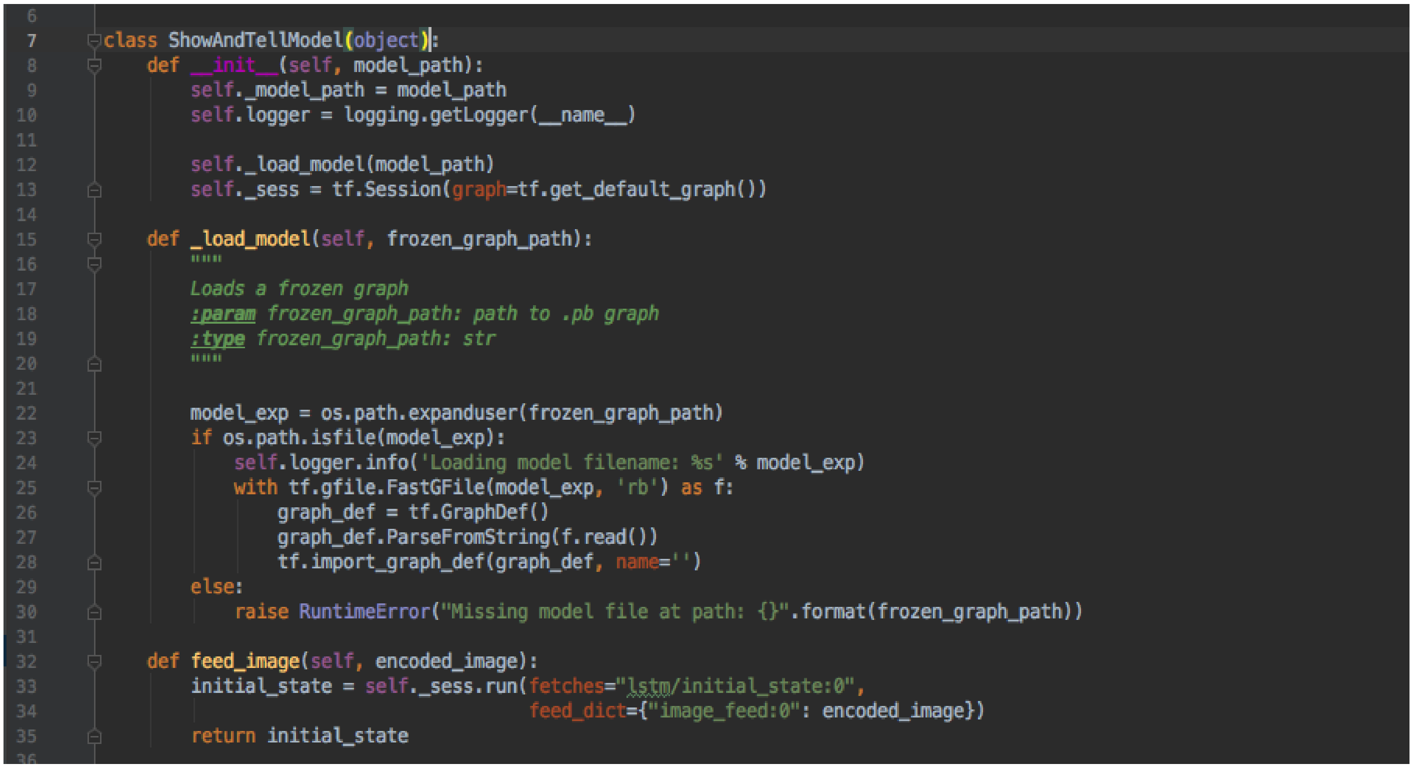
- Following stage is that we need to prepare the model which needs the vocabulary record that contains every one of the words required in for the inscription.

- At the point when the model is prepared it is tried by giving an info picture to the model.



```
#required libraries
from __future__ import absolute_import
from __future__ import division
from __future__ import print_function

import nltk
import nltk.translate.gleu_score as gleu
import numpy
import os
try:
  nltk.data.find('tokenizers/punkt')
except LookupError:
  nltk.download('punkt')
import nltk

try:
  nltk.data.find('tokenizers/punkt')
except LookupError:
  nltk.download('punkt')
from nltk.translate.bleu_score import sentence_bleu

import logging
import math

import tensorflow as tf
```

Model functionality in snippet



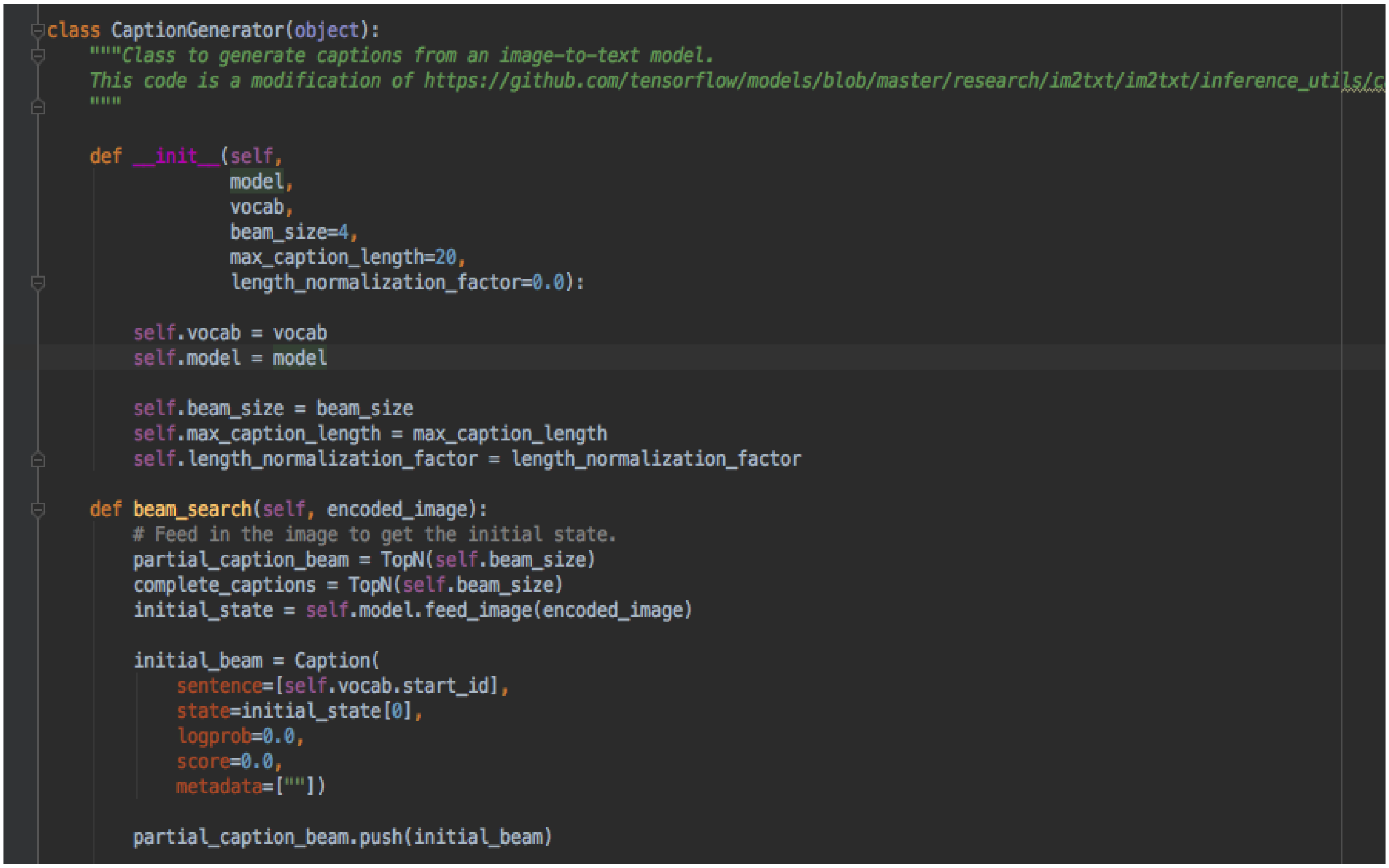
```

class ShowAndTellModel(object):
    def __init__(self, model_path):
        self._model_path = model_path
        self.logger = logging.getLogger(__name__)

        self._load_model(model_path)
        self._sess = tf.Session(graph=tf.get_default_graph())

    def _load_model(self, frozen_graph_path):
        """
        Loads a frozen graph
        :param frozen_graph_path: path to .pb graph
        :type frozen_graph_path: str
        """

        model_exp = os.path.expanduser(frozen_graph_path)
        if os.path.isfile(model_exp):
            self.logger.info('Loading model filename: %s' % model_exp)
            with tf.gfile.FastGFile(model_exp, 'rb') as f:
                graph_def = tf.GraphDef()
                graph_def.ParseFromString(f.read())
                tf.import_graph_def(graph_def, name='')
        else:
            raise RuntimeError("Missing model file at path: {}".format(frozen_graph_path))

    def feed_image(self, encoded_image):
        initial_state = self._sess.run(fetches="lstm/initial_state:0",
                                       feed_dict={"image_feed:0": encoded_image})
        return initial_state
```

- Load demonstrate work is utilized for stacking the model utilizing the os library. If there should arise an occurrence of any mistake that is dealt with by the catch hinder as an exemption.

- Feed_image work for the most part encourages a picture to LSTM display for anticipating the following word in the subtitle age.

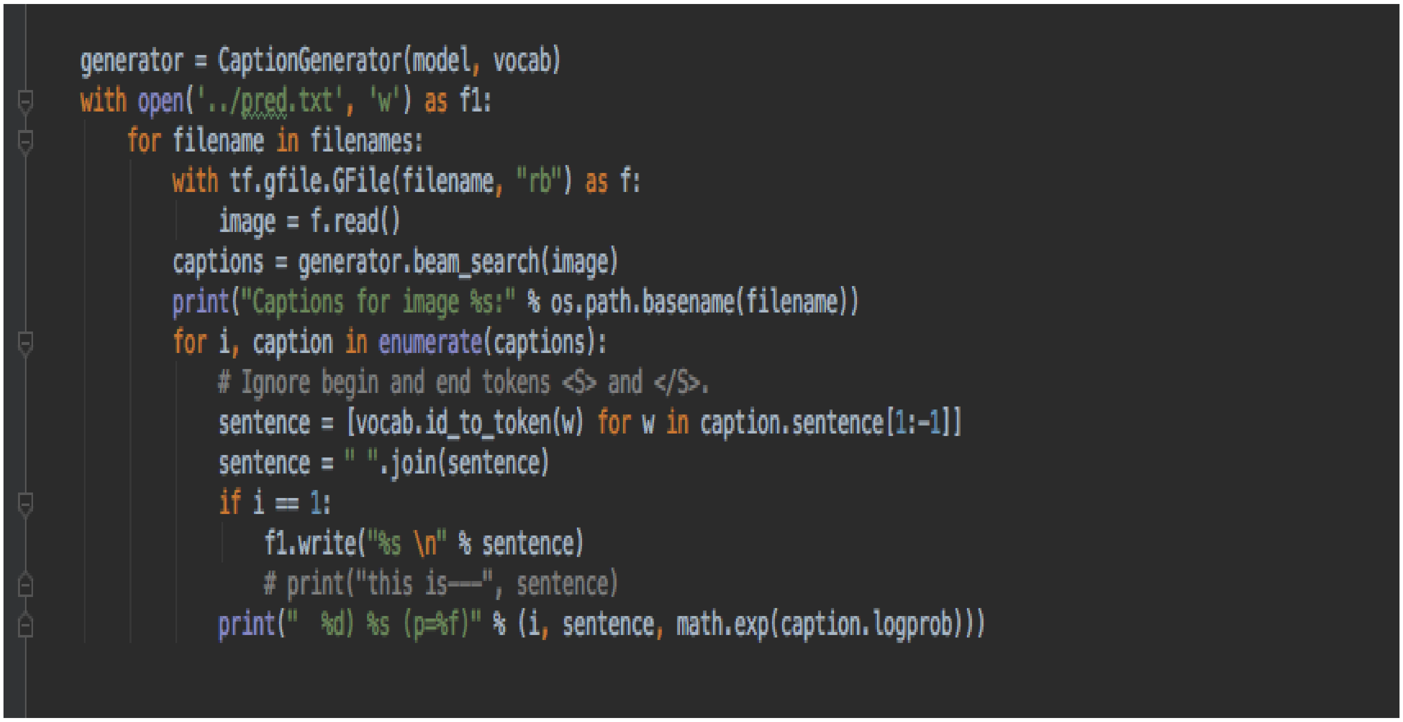
- A long side the over two capacities inference_step work is accessible which is a softmax work usage which is a last stage.

Functionality in Beam Size



```
class CaptionGenerator(object):
    """Class to generate captions from an image-to-text model.
    This code is a modification of https://github.com/tensorflow/models/blob/master/research/im2txt/im2txt/inference_utils/c
    """

    def __init__(self,
                 model,
                 vocab,
                 beam_size=4,
                 max_caption_length=20,
                 length_normalization_factor=0.0):

        self.vocab = vocab
        self.model = model

        self.beam_size = beam_size
        self.max_caption_length = max_caption_length
        self.length_normalization_factor = length_normalization_factor

    def beam_search(self, encoded_image):
        # Feed in the image to get the initial state.
        partial_caption_beam = TopN(self.beam_size)
        complete_captions = TopN(self.beam_size)
        initial_state = self.model.feed_image(encoded_image)

        initial_beam = Caption(
            sentence=[self.vocab.start_id],
            state=initial_state[0],
            logprob=0.0,
            score=0.0,
            metadata=[""])

        partial_caption_beam.push(initial_beam)
```

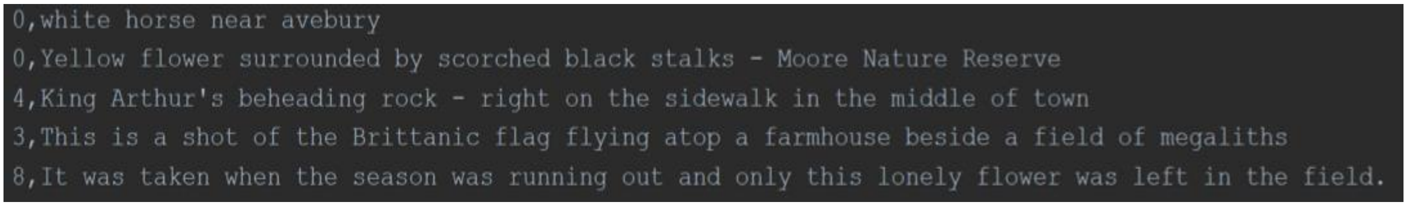
Next feture is BLEU score which for the most part decides a measurement for assessing the created sentence which shifts somewhere in the range of 0 and 1.



```python
generator = CaptionGenerator(model, vocab)
with open('../pred.txt', 'w') as f1:
    for filename in filenames:
        with tf.gfile.GFile(filename, "rb") as f:
            image = f.read()
        captions = generator.beam_search(image)
        print("Captions for image %s:" % os.path.basename(filename))
        for i, caption in enumerate(captions):
            # Ignore begin and end tokens <S> and </S>.
            sentence = [vocab.id_to_token(w) for w in caption.sentence[1:-1]]
            sentence = " ".join(sentence)
            if i == 1:
                f1.write("%s \n" % sentence)
                # print("this is---", sentence)
            print("  %d) %s (p=%f)" % (i, sentence, math.exp(caption.logprob)))
```

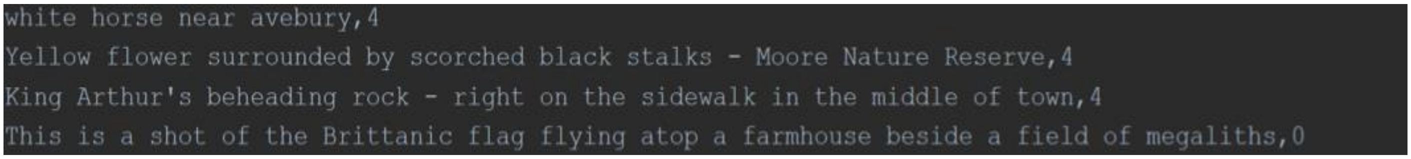
- Different Clustering strategies have been executed in this segment which incorporates K-Means and EM bunching.
- Both speak to the different grouping procedures when connected on unsupervised information bunches them as needs be.

output of KM_clustering



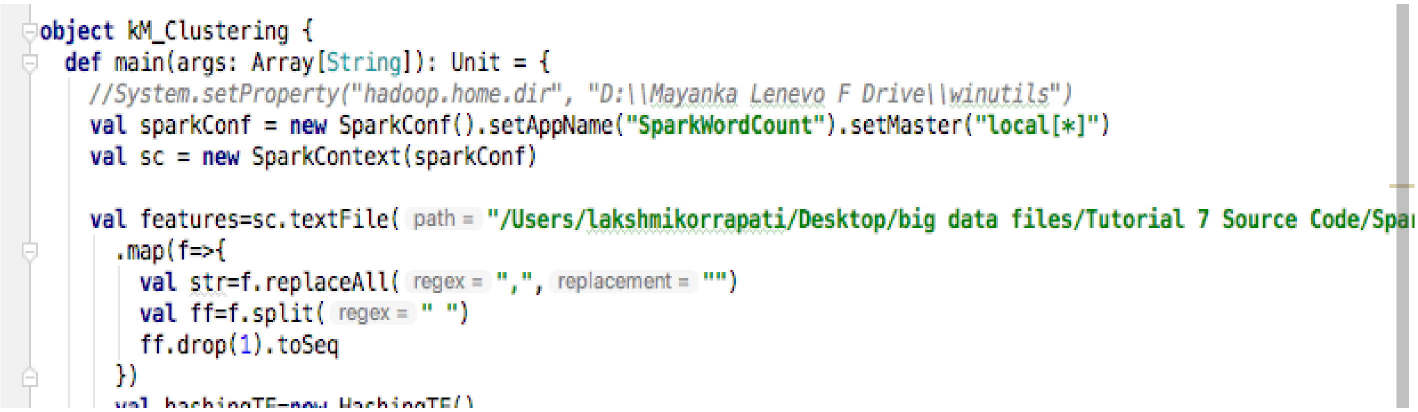
```
0,white horse near avebury
0,Yellow flower surrounded by scorched black stalks - Moore Nature Reserve
4,King Arthur's beheading rock - right on the sidewalk in the middle of town
3,This is a shot of the Brittanic flag flying atop a farmhouse beside a field of megaliths
8,It was taken when the season was running out and only this lonely flower was left in the field.
```

Output for EM_clustering



```
white horse near avebury,4
Yellow flower surrounded by scorched black stalks - Moore Nature Reserve,4
King Arthur's beheading rock - right on the sidewalk in the middle of town,4
This is a shot of the Brittanic flag flying atop a farmhouse beside a field of megaliths,0
```

Snippet for KM_clustering:



```
object kM_Clustering {
  def main(args: Array[String]): Unit = {
    //System.setProperty("hadoop.home.dir", "D:\\Mayanka Lenevo F Drive\\winutils")
    val sparkConf = new SparkConf().setAppName("SparkWordCount").setMaster("local[*]")
    val sc = new SparkContext(sparkConf)

    val features=sc.textFile( path = "/Users/lakshmikorrapati/Desktop/big data files/Tutorial 7 Source Code/Spa
      .map(f=>{
        val str=f.replaceAll( regex = ",", replacement = "")
        val ff=f.split( regex = " ")
        ff.drop(1).toSeq
      })
      val hashingTF=new HashingTF()
```

The code speaks to pushing the subtitles into a hash guide and after that grouping them likewise, at that point putting away the qualities as csv document.