```
In [104]: import numpy as np
          import pandas as pd
          import matplotlib.pyplot as plt
          % matplotlib inline
```

READ TEXT DATA

```
In [105]: # read text captions
          def readTextFile(path):
              with open(path) as f:
                  captions = f.read()
              return captions
```

```
In [106]: captions = readTextFile("Flickr_Data/Flickr_TextData/Flickr8k.token.txt")
          print(len(captions.split("\n")))

          40461
```

```
In [107]: captions = captions.split("\n")[0:-1] # last line in file is empty
```

```
In [108]: captions[0] # 0th line

Out[108]: '1000268201_693b08cb0e.jpg#0\tA child in a pink dress is climbing up a set of stairs in an entry way .'
```

```
In [109]: captions

Out[109]: ['1000268201_693b08cb0e.jpg#0\tA child in a pink dress is climbing up a set of stairs in an entry way
          .',
           '1000268201_693b08cb0e.jpg#1\tA girl going into a wooden building .',
           '1000268201_693b08cb0e.jpg#2\tA little girl climbing into a wooden playhouse .',
           '1000268201_693b08cb0e.jpg#3\tA little girl climbing the stairs to her playhouse .',
           '1000268201_693b08cb0e.jpg#4\tA little girl in a pink dress going into a wooden cabin .',
           '1001773457_577c3a7d70.jpg#0\tA black dog and a spotted dog are fighting',
           '1001773457_577c3a7d70.jpg#1\tA black dog and a tri-colored dog playing with each other on the road
          .',
           '1001773457_577c3a7d70.jpg#2\tA black dog and a white dog with brown spots are staring at each other
          in the street .',
           '1001773457_577c3a7d70.jpg#3\tTwo dogs of different breeds looking at each other on the road .',
           '1001773457_577c3a7d70.jpg#4\tTwo dogs on pavement moving toward each other .',
           '1002674143_1b742ab4b8.jpg#0\tA little girl covered in paint sits in front of a painted rainbow with
          her hands in a bowl .',
           '1002674143_1b742ab4b8.jpg#1\tA little girl is sitting in front of a large painted rainbow .',
           '1002674143_1b742ab4b8.jpg#2\tA small girl in the grass plays with fingerpaints in front of a white
          canvas with a rainbow on it .',
           '1002674143_1b742ab4b8.jpg#3\tThere is a girl with pigtails sitting in front of a rainbow painting
```

```
In [110]: print(len(captions))

          40460
```

```
In [111]: # make a dictionary mapping each image to the captions it corresponds to.
          d = {}
```

```
In [112]: for cap in captions:
              first, second = cap.split("\t")
              img_id = first.split(".")[0]
              if d.get(img_id) is None: # check if image_id is already present in dictionary
                  d[img_id] = []
              d[img_id].append(second)
```

```
In [113]: d["1009434119_febe49276a"]
```

```
Out[113]: ['A black and white dog is running in a grassy garden surrounded by a white fence .',
           'A black and white dog is running through the grass .',
           'A Boston terrier is running in the grass .',
           'A Boston Terrier is running on lush green grass in front of a white fence .',
           'A dog runs on the green grass near a wooden fence .']
```

```
In [114]: d
```

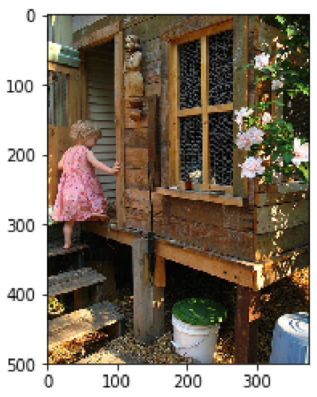
```
Out[114]: {'1000268201_693b08cb0e': ['A child in a pink dress is climbing up a set of stairs in an entry way
          .',
            'A girl going into a wooden building .',
            'A little girl climbing into a wooden playhouse .',
            'A little girl climbing the stairs to her playhouse .',
            'A little girl in a pink dress going into a wooden cabin .'],
           '1001773457_577c3a7d70': ['A black dog and a spotted dog are fighting',
            'A black dog and a tri-colored dog playing with each other on the road .',
            'A black dog and a white dog with brown spots are staring at each other in the street .',
            'Two dogs of different breeds looking at each other on the road .',
            'Two dogs on pavement moving toward each other .'],
           '1002674143_1b742ab4b8': ['A little girl covered in paint sits in front of a painted rainbow with he
          r hands in a bowl .',
            'A little girl is sitting in front of a large painted rainbow .',
            'A small girl in the grass plays with fingerpaints in front of a white canvas with a rainbow on it
          .',
            'There is a girl with pigtails sitting in front of a rainbow painting .',
            'Young girl with pigtails painting outside in the grass .'],
           '1003163366_44323f5815': ['A man lays on a bench while his dog sits by him .',
            'A man lays on the bench to which a white dog is also tied .',
```

```
In [115]: len(d)*5
```

```
Out[115]: 40460
```

```
In [116]: IMG_PATH = "/Flickr_Data/Images/"
          import cv2
          import matplotlib.pyplot as plt
```

```
In [117]: img = cv2.imread("C:\ANACONDA\Scripts\MINOR PROJECT IMAGE CAPTIONING" + IMG_PATH+"1000268201_693b08cb0e.j
          img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
          plt.imshow(img)
          plt.show()
```



```
In [118]: d["1000268201_693b08cb0e"]
```

```
Out[118]: ['A child in a pink dress is climbing up a set of stairs in an entry way .',
           'A girl going into a wooden building .',
           'A little girl climbing into a wooden playhouse .',
           'A little girl climbing the stairs to her playhouse .',
           'A little girl in a pink dress going into a wooden cabin .']
```

DATA CLEANING

```
In [119]: import re
```

```
In [120]: def clean_text(sentence):
              sentence = sentence.lower()
              sentence = re.sub("[^a-z]+"," ",sentence)
              sentence = sentence.split()

              sentence = [s for s in sentence if len(s) > 1]
              sentence = " ".join(sentence)
              return sentence
```

```
In [121]: # clean all captions
          for key, caption_list in d.items():
              for i in range(len(caption_list)):
                  caption_list[i] = clean_text(caption_list[i])
```

```
In [122]: d["1000268201_693b08cb0e"]
```

```
Out[122]: ['child in pink dress is climbing up set of stairs in an entry way',
           'girl going into wooden building',
           'little girl climbing into wooden playhouse',
           'little girl climbing the stairs to her playhouse',
           'little girl in pink dress going into wooden cabin']
```

```
In [123]: #store the data in text file
          with open("descriptions_1.txt", "w") as f:
              f.write(str(d))
```

CREATE VOCAB OF UNIQUE WORDS

```
In [124]: import json
          descriptions = None
          with open("descriptions_1.txt") as f:
              descriptions = f.read()

          json_acceptable_string = descriptions.replace("'", "\"")
          descriptions = json.loads(json_acceptable_string)
```

```
In [125]: print(type(descriptions))
```

```
          <class 'dict'>
```

```
In [126]: descriptions['1000268201_693b08cb0e']
```

```
Out[126]: ['child in pink dress is climbing up set of stairs in an entry way',
           'girl going into wooden building',
           'little girl climbing into wooden playhouse',
           'little girl climbing the stairs to her playhouse',
           'little girl in pink dress going into wooden cabin']
```

```
In [127]: vocab = set()
          for key in descriptions.keys():
              [vocab.update(sentence.split()) for sentence in descriptions[key]]
```

In [128]:
```
vocab
```
Out[128]:
```
{'campaign',
 'curious',
 'smacks',
 'airtime',
 'bright',
 'vinyl',
 'stool',
 'miniskirts',
 'stuntman',
 'pitched',
 'pagent',
 'moves',
 'hindu',
 'artifacts',
 'caravan',
 'flyaway',
 'evident',
 'hulk',
 'store',
```
In [129]:
```
print(len(vocab))
```
```
8424
```
In [130]:
```
# total no of words across all sentences
total_words = []
for key in descriptions.keys():
    [total_words.append(i) for des in descriptions[key] for i in des.split()]
print(len(total_words))
```
```
373837
```
In [131]:
```
print(total_words[:10])
```
```
['child', 'in', 'pink', 'dress', 'is', 'climbing', 'up', 'set', 'of', 'stairs']
```
REMOVE INFREQUENT WORDS - pick only those words out of vocab that appear atleast 10 times in total_words

In [132]:
```
# find out frequency counts in total_words
import collections
counter = collections.Counter(total_words)
freq_cnt = dict(counter)
print(freq_cnt)
```
```
{'child': 1545, 'in': 18987, 'pink': 739, 'dress': 348, 'is': 9345, 'climbing': 507, 'up': 1302, 'se
t': 109, 'of': 6723, 'stairs': 109, 'an': 2432, 'entry': 1, 'way': 53, 'girl': 3328, 'going': 149, 'i
nto': 1074, 'wooden': 284, 'building': 511, 'little': 1768, 'playhouse': 6, 'the': 18420, 'to': 3176,
'her': 1178, 'cabin': 4, 'black': 3848, 'dog': 8138, 'and': 8863, 'spotted': 38, 'are': 3505, 'fighti
ng': 133, 'tri': 14, 'colored': 221, 'playing': 2008, 'with': 7765, 'each': 430, 'other': 773, 'on':
10746, 'road': 398, 'white': 3959, 'brown': 2578, 'spots': 29, 'staring': 57, 'at': 2916, 'street': 9
44, 'two': 5643, 'dogs': 2125, 'different': 46, 'breeds': 5, 'looking': 744, 'pavement': 48, 'movin
g': 41, 'toward': 146, 'covered': 372, 'paint': 62, 'sits': 577, 'front': 1386, 'painted': 64, 'rainb
ow': 22, 'hands': 246, 'bowl': 30, 'sitting': 1368, 'large': 1237, 'small': 1278, 'grass': 1622, 'pla
ys': 526, 'fingerpaints': 3, 'canvas': 6, 'it': 401, 'there': 304, 'pigtails': 14, 'painting': 43, 'y
oung': 2630, 'outside': 791, 'man': 7275, 'lays': 56, 'bench': 375, 'while': 1968, 'his': 2357, 'by':
1249, 'him': 403, 'which': 51, 'also': 20, 'tied': 15, 'sleeping': 60, 'next': 749, 'shirtless': 104,
'lies': 43, 'park': 508, 'laying': 189, 'holding': 1324, 'leash': 131, 'ground': 357, 'orange': 745,
'hat': 682, 'starring': 8, 'something': 346, 'wears': 115, 'glasses': 206, 'gauges': 2, 'wearing': 30
62, 'blitz': 1, 'beer': 45, 'can': 39, 'crocheted': 1, 'pierced': 6, 'ears': 69, 'rope': 251, 'net':
58, 'red': 2691, 'roping': 2, 'climbs': 201, 'bridge': 141, 'grips': 2, 'onto': 211, 'ropes': 38, 'pl
ayground': 201, 'running': 2073, 'grassy': 474, 'garden': 54, 'surrounded': 178, 'fence': 340, 'throu
gh': 2032, 'boston': 9, 'terrier': 31, 'lush': 8, 'green': 1234, 'runs': 925, 'near': 1026, 'shakes':
37, 'its': 925, 'head': 377, 'shore': 170, 'ball': 1783, 'edge': 170, 'beach': 1046, 'feet': 87, 'sta
```

In [133]:
```
len(freq_cnt.keys())
```
Out[133]:
```
8424
```
In [134]:
```
# sort the dictionary acc to freq counts
sorted_freq_cnt = sorted(freq_cnt.items(), reverse = True, key = lambda x : x[1])
sorted_freq_cnt
```
Out[134]:
```
[('in', 18987),
 ('the', 18420),
 ('on', 10746),
 ('is', 9345),
 ('and', 8863),
 ('dog', 8138),
 ('with', 7765),
 ('man', 7275),
 ('of', 6723),
 ('two', 5643),
 ('white', 3959),
 ('black', 3848),
 ('boy', 3581),
 ('are', 3505),
 ('woman', 3403),
 ('girl', 3328),
 ('to', 3176),
 ('wearing', 3062),
 ('at', 2916),
 ('people', 2887)
```
In [135]:
```
threshold = 10
sorted_freq_cnt = [x for x in sorted_freq_cnt if x[1] > threshold]
sorted_freq_cnt
```
Out[135]:
```
[('in', 18987),
 ('the', 18420),
 ('on', 10746),
 ('is', 9345),
 ('and', 8863),
 ('dog', 8138),
 ('with', 7765),
 ('man', 7275),
 ('of', 6723),
 ('two', 5643),
 ('white', 3959),
 ('black', 3848),
 ('boy', 3581),
 ('are', 3505),
 ('woman', 3403),
 ('girl', 3328),
 ('to', 3176),
 ('wearing', 3062),
 ('at', 2916),
 ('people', 2887)
```
In [136]:
```
total_words = [x[0] for x in sorted_freq_cnt]
```

```
In [137]: total_words

Out[137]: ['in',
           'the',
           'on',
           'is',
           'and',
           'dog',
           'with',
           'man',
           'of',
           'two',
           'white',
           'black',
           'boy',
           'are',
           'woman',
           'girl',
           'to',
           'wearing',
           'at',
```

```
In [138]: len(total_words)

Out[138]: 1845
```

PREPARE TRAIN TEST DATA

```
In [139]: train_file_data = readTextFile("Flickr_Data/Flickr_TextData/Flickr_8k.trainImages.txt")
          test_file_data = readTextFile("Flickr_Data/Flickr_TextData/Flickr_8k.testImages.txt")
```

```
In [140]: print(train_file_data)

          2513260012_03d33305cf.jpg
          2903617548_d3e38d7f88.jpg
          3338291921_fe7ae0c8f8.jpg
          488416045_1c6d903fe0.jpg
          2644326817_8f45080b87.jpg
          218342358_1755a9cce1.jpg
          2501968935_02f2cd8079.jpg
          2699342860_5288e203ea.jpg
          2638369467_8fc251595b.jpg
          2926786902_815a99a154.jpg
          2851304910_b5721199bc.jpg
          3423802527_94bd2b23b0.jpg
          3356369156_074750c6cc.jpg
          2294598473_40637b5c04.jpg
          1191338263_a4fa073154.jpg
          2380765956_6313d8cae3.jpg
          3197891333_b1b0fd1702.jpg
          3119887967_271a097464.jpg
          2276499757_b44dc6f8ce.jpg
```

```
In [141]: print(type(train_file_data))

          <class 'str'>
```

```
In [142]: train = [row.split(".")[0] for row in train_file_data.split("\n")[:-1]]
          print(train[:10])

          ['2513260012_03d33305cf', '2903617548_d3e38d7f88', '3338291921_fe7ae0c8f8', '488416045_1c6d903fe0', '26
          44326817_8f45080b87', '218342358_1755a9cce1', '2501968935_02f2cd8079', '2699342860_5288e203ea', '263836
          9467_8fc251595b', '2926786902_815a99a154']
```

In [143]:
```
test = [row.split(".")[0] for row in test_file_data.split("\n")[:-1]]
print(test[:10])
```

```
['3385593926_d3e9c21170', '2677656448_6b7e7702af', '311146855_0b65fdb169', '1258913059_07c613f7ff', '24
1347760_d44c8d3a01', '2654514044_a70a6e2c21', '2339106348_2df90aa6a9', '256085101_2c2617c5d0', '2807068
62_14c30d734a', '3072172967_630e9c69d0']
```

In [144]:
```
len(train), len(test)
```

Out[144]:
```
(6000, 1000)
```

In [145]:
```
# prepare descriptions for the training data
# add start and end token to the training data
train_descriptions = {}
for img_id in train:
    train_descriptions[img_id] = []
    for cap in descriptions[img_id]:
        cap_to_append = "startseq " + cap + " endseq"
        train_descriptions[img_id].append(cap_to_append)
```

In [146]:
```
train_descriptions
```

Out[146]:
```
{'2513260012_03d33305cf': ['startseq black dog is running after white dog in the snow endseq',
  'startseq black dog chasing brown dog through snow endseq',
  'startseq two dogs chase each other across the snowy ground endseq',
  'startseq two dogs play together in the snow endseq',
  'startseq two dogs running through low lying body of water endseq'],
 '2903617548_d3e38d7f88': ['startseq little baby plays croquet endseq',
  'startseq little girl plays croquet next to truck endseq',
  'startseq the child is playing croquette by the truck endseq',
  'startseq the kid is in front of car with put and ball endseq',
  'startseq the little boy is playing with croquet hammer and ball beside the car endseq'],
 '3338291921_fe7ae0c8f8': ['startseq brown dog in the snow has something hot pink in its mouth endse
q',
  'startseq brown dog in the snow holding pink hat endseq',
  'startseq brown dog is holding pink shirt in the snow endseq',
  'startseq dog is carrying something pink in its mouth while walking through the snow endseq',
  'startseq dog with something pink in its mouth is looking forward endseq'],
 '488416045_1c6d903fe0': ['startseq brown dog is running along beach endseq',
  'startseq brown dog wearing black collar running across the beach endseq',
  'startseq dog walks on the sand near the water endseq',
```

PREPROCESSING FOR CAPTIONS

In [147]:
```
len(total_words)
```

Out[147]:
```
1845
```

In [148]:
```
word_to_idx = {}
idx_to_word = {}
for i, word in enumerate(total_words):
    word_to_idx[word] = i+1
    idx_to_word[i+1] = word
```

In [149]:
```
word_to_idx['dog']
```

Out[149]:
```
6
```

In [150]:
```
idx_to_word[6]
```

Out[150]:
```
'dog'
```

In [151]:
```
print(len(idx_to_word))
```

```
1845
```

```
In [152]: idx_to_word[1846] = 'startseq'
          word_to_idx['startseq'] = 1846
```

```
In [153]: idx_to_word[1847] = 'endseq'
          word_to_idx['endseq'] = 1847
```

```
In [154]: print(len(idx_to_word))
```

```
1847
```

```
In [155]: vocab_size = len(word_to_idx) + 1
          print(vocab_size)
```

```
1848
```

```
In [156]: # find out the length of longest sentence
          max_len = 0
          for key in train_descriptions.keys():
              for cap in train_descriptions[key]:
                  max_len = max(max_len,len(cap.split()))
```

```
In [157]: max_len
```

```
Out[157]: 35
```

TEXT FEATURE EXTRACTION USING TRANSFER LEARNING (GLOVE EMBEDDINGS)

```
In [158]: f = open("glove.6B.50d.txt", encoding = "utf8")
```

```
In [159]: # create a dictionary mapping each word in the glove embedding text file to its corresponding glove embedd
          embedding_index = {}
```

```
In [160]: for line in  f:
              values = line.split()
              word = values[0]
              word_embedding = np.array(values[1:], dtype = 'float')
              embedding_index[word] = word_embedding
          f.close()
```

```
In [161]: embedding_index['house']
```

```
Out[161]: array([ 0.60137  ,  0.28521  , -0.032038 , -0.43026  ,  0.74806  ,
                  0.26223  , -0.97361  ,  0.078581 , -0.57588  , -1.188    ,
                 -1.8507   , -0.24887  ,  0.055549 ,  0.0086155,  0.067951 ,
                  0.40554  , -0.073998 , -0.21318  ,  0.37167  , -0.71791  ,
                  1.2234   ,  0.35546  , -0.41537  , -0.21931  , -0.39661  ,
                 -1.7831   , -0.41507  ,  0.29533  , -0.41254  ,  0.020096 ,
                  2.7425   , -0.9926   , -0.71033  , -0.46813  ,  0.28265  ,
                 -0.077639 ,  0.3041   , -0.06644  ,  0.3951   , -0.70747  ,
                 -0.38894  ,  0.23158  , -0.49508  ,  0.14612  , -0.02314  ,
                  0.56389  , -0.86188  , -1.0278   ,  0.039922 ,  0.20018  ])
```

```
In [162]: def get_embedding_matrix():
              emb_dim = 50
              matrix = np.zeros((vocab_size, emb_dim))
              for word, idx in word_to_idx.items():
                  embedding_vector = embedding_index.get(word)
                  if embedding_vector is not None: # if word is not in glove.txt file, take it as all zeros
                      matrix[idx] = embedding_vector
              return matrix
```

```
In [163]: embedding_matrix = get_embedding_matrix()
In [168]: embedding_matrix[1847], embedding_matrix[1846] # glove vector for startseq and endseg - all 0s
```

```
Out[168]: (array([0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
                  0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
                  0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.]),
           array([0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
                  0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.,
                  0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0.]))
```

```
In [164]: embedding_matrix.shape
```

```
Out[164]: (1848, 50)
```