

```
In [1]: # imports
from keras.applications.resnet50 import ResNet50
from keras.preprocessing import image
from keras.applications.resnet50 import preprocess_input
from keras.models import load_model
from matplotlib import pyplot as plt
import numpy as np
import pickle
```

Using TensorFlow backend.

```
In [2]: # Load the neural network model
model = load_model('weights_v2/version_v2_23.h5')

#Load the cnn model
cnn_model = ResNet50(weights='imagenet',include_top=False)

# Load the index file
with open('search_index.pickle', 'rb') as f:
    indexes = pickle.load(f)
```

```
In [3]: print(len(model.layers))
```

8

```
In [4]: # helper function to index the query image
# inputs - the image path and the cnn-model loaded above
# outputs - the flattened feature map of the query image extracted by the cnn-model

def index_query(image_path, cnn_model):

    img = image.load_img(image_path, target_size=(224, 224))
    x = image.img_to_array(img)
    x = np.expand_dims(x, axis=0)
    x = preprocess_input(x)
    query = cnn_model.predict(x)
    query = query.flatten()

    imgplot = plt.imshow(img)
    plt.show()

    return query
```

```
In [35]: # helper function to do the retrieval
# inputs - the query index computed from the cnn_model,
#           the indexes list
#           and the number of results to retrieve (topn)
# outputs - a list of tuples containing the retrieved image paths and their respe

def find_query(query_index, indexes, topn):

    retrieved = []

    for idx in indexes:
        search_image = idx[0]
        search_idx = idx[1]
        diff = query_index - search_idx
        diff = diff**2
        diff = np.reshape(diff,(1,2048))
        match_score = model.predict(diff)
        retrieved.append((search_image, match_score))

    sorted_retrieval = sorted(retrieved, key=lambda x: x[1])

    selected = sorted_retrieval[:topn]
    return selected

# for the purpose of comparison, I have used l2 norm distance as scores

def find_query_l2(query_index, indexes, topn):

    retrieved = []

    for idx in indexes:
        search_image = idx[0]
        search_idx = idx[1]
        diff = query_index - search_idx
        diff = diff**2
        match_score = sum(diff)
        retrieved.append((search_image, match_score))

    sorted_retrieval = sorted(retrieved, key=lambda x: x[1])

    selected = sorted_retrieval[:topn]
    return selected

# helper function to plot the topn results retrieved
def plot_results(selected_images, highlight = None):

    n = len(selected_images)

    fig = plt.figure(figsize=(20,10))

    for i in range(n):
        x = round(n / 5)
        a = fig.add_subplot(x,5,i+1)
        image_path = selected_images[i][0]
```

```

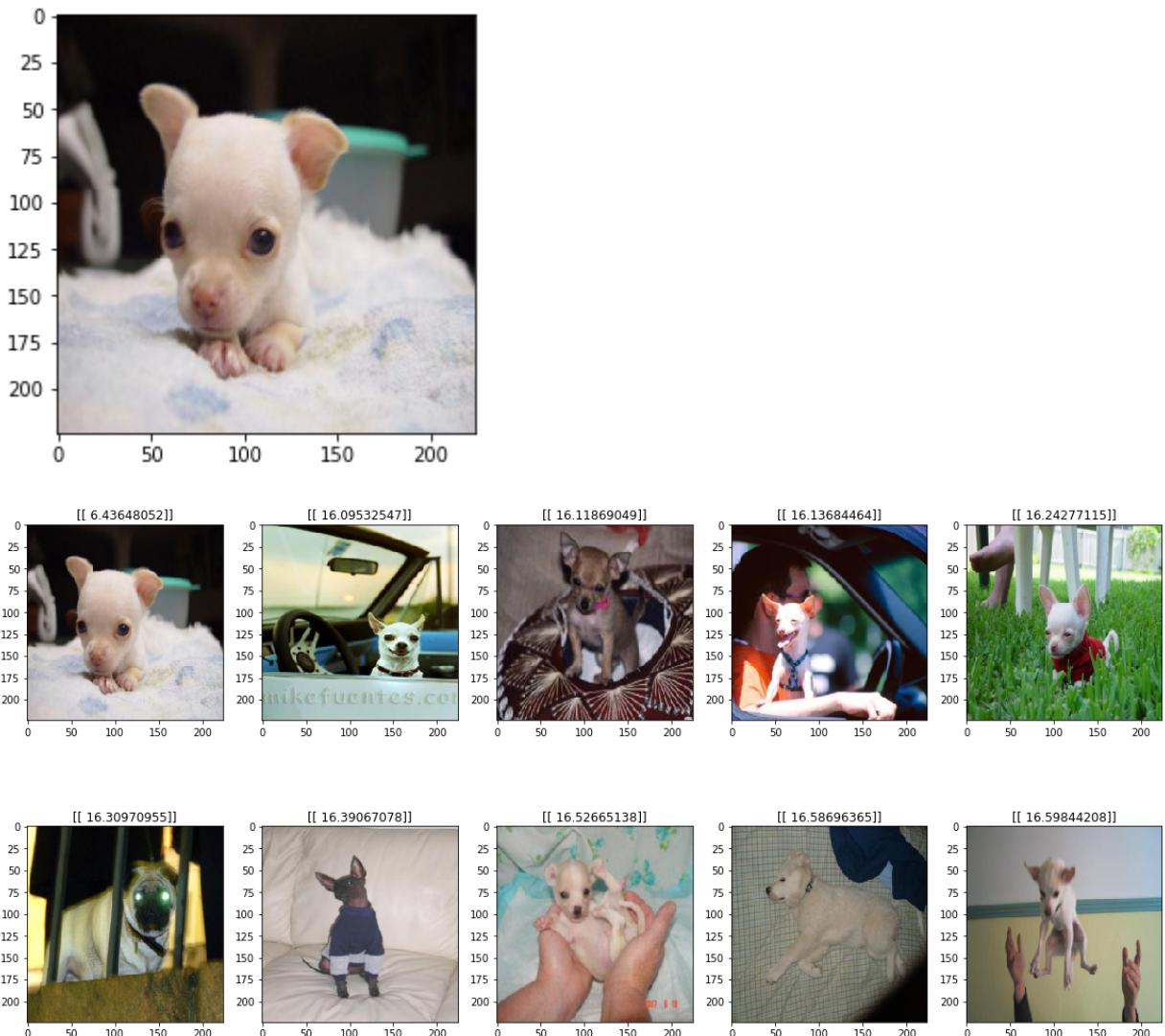
img = image.load_img(image_path, target_size=(224, 224))
plt.imshow(img)
if i == highlight:
    a.set_title("RELEVANT")
else:
    a.set_title(str(selected_images[i][1]))

plt.show()

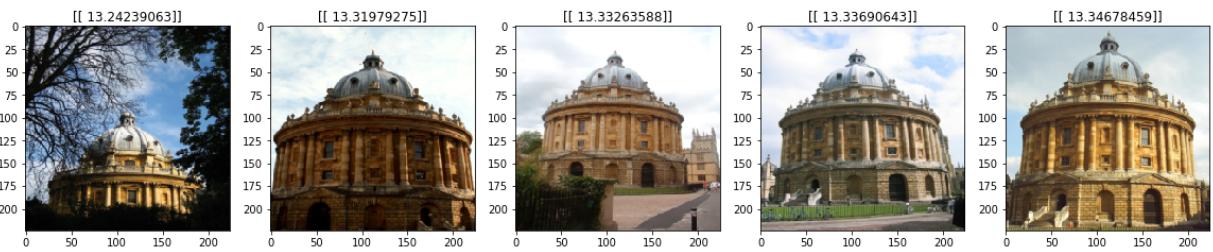
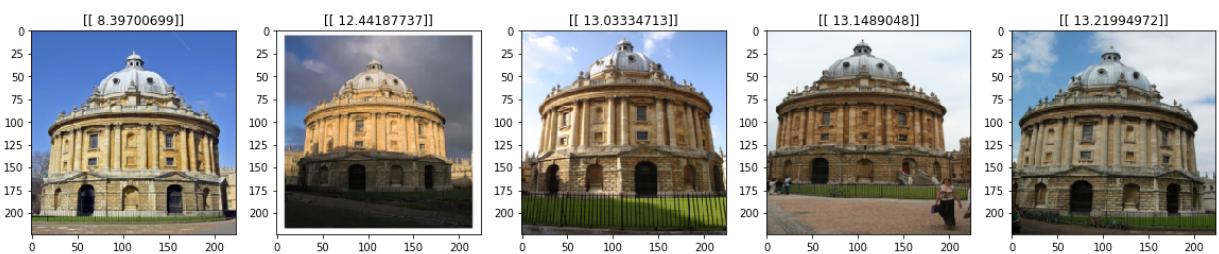
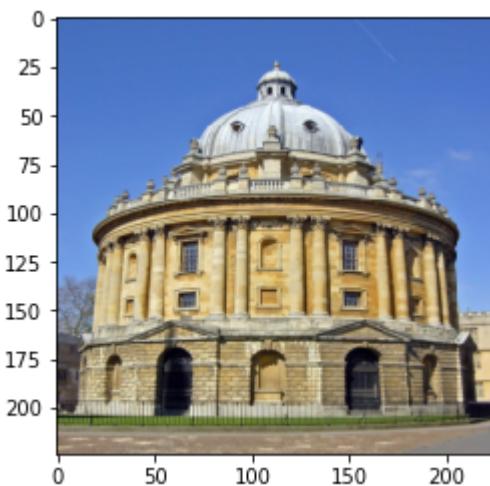
```

Test with trained images

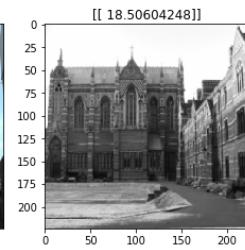
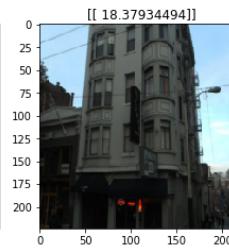
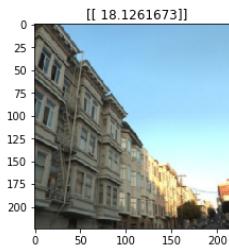
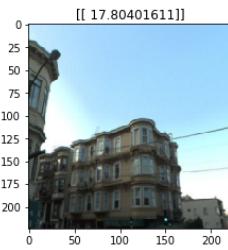
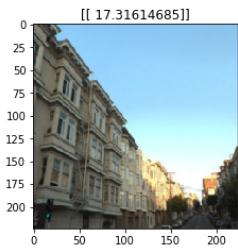
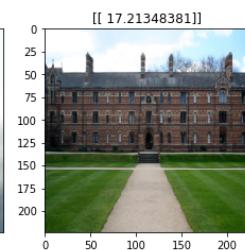
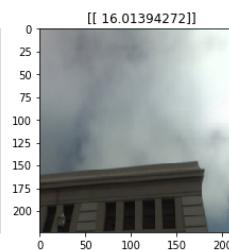
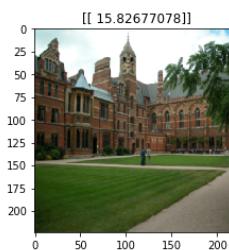
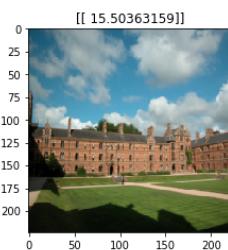
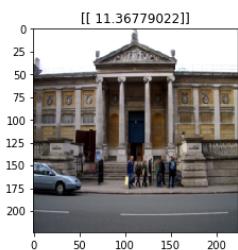
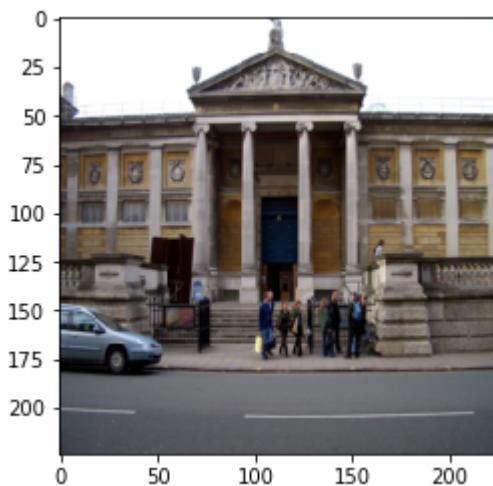
In [8]: `query_index = index_query('Dogs/n02085620-Chihuahua/n02085620_712.jpg', cnn_model)
results = find_query(query_index, indexes, 10)
plot_results(results)`



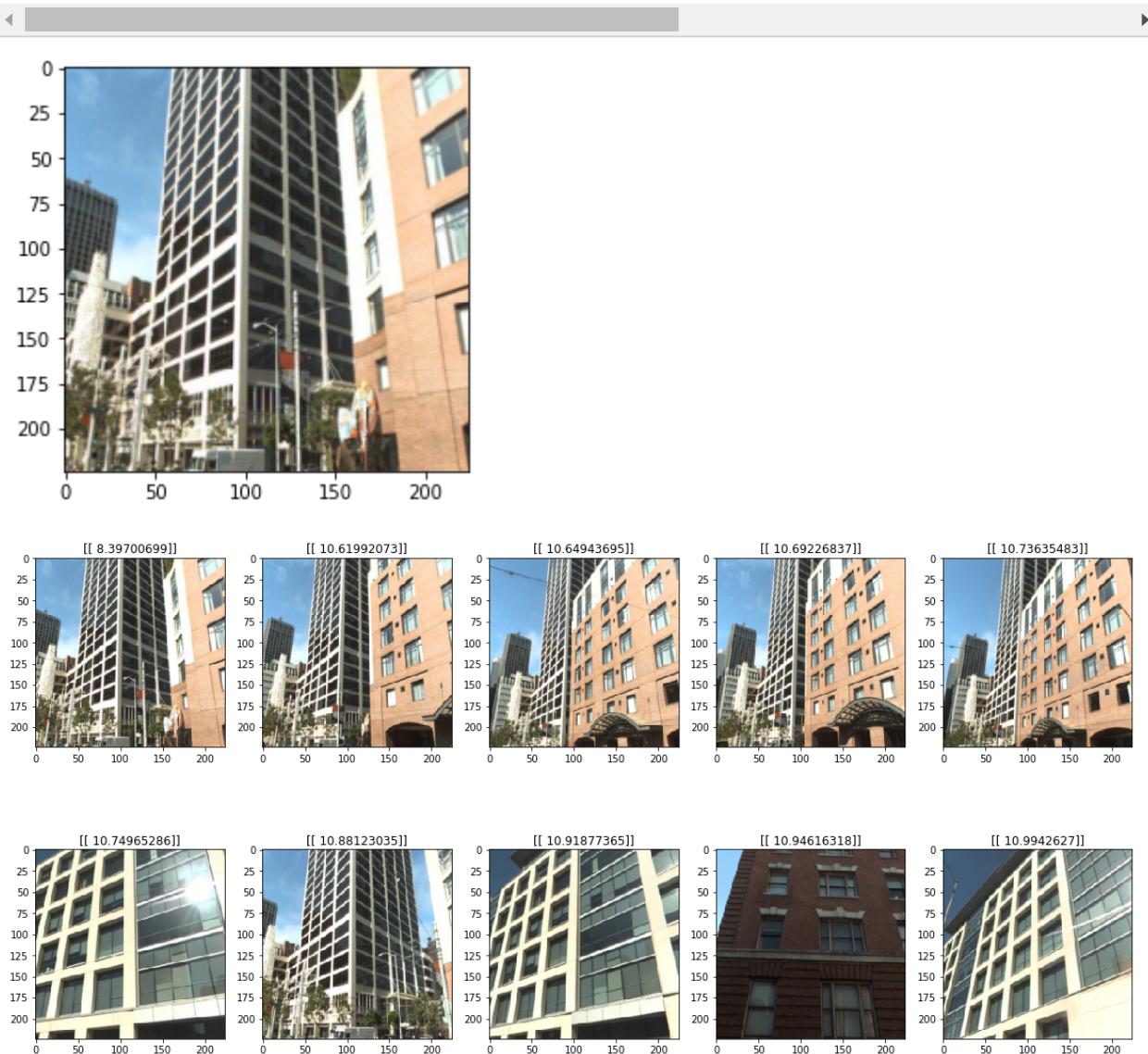
```
In [17]: query_index = index_query('oxford_selected/radcliffe_cam/radcliffe_camera_000027')
results = find_query(query_index, indexes, 10)
plot_results(results)
```



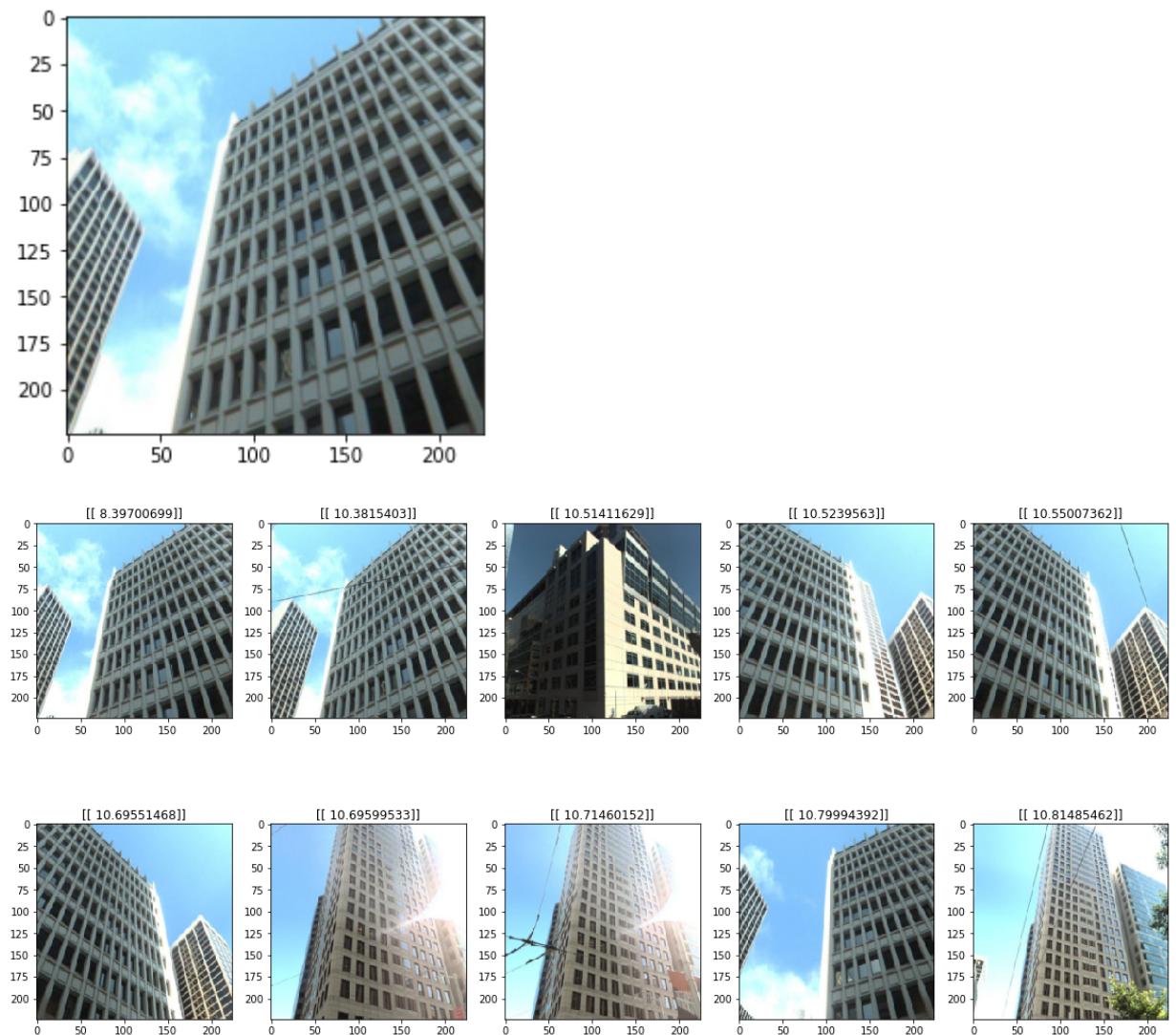
```
In [23]: query_index = index_query('oxford_selected/ashmolean/ashmolean_000000.jpg', cnn_m
results = find_query(query_index, indexes, 10)
plot_results(results)
```



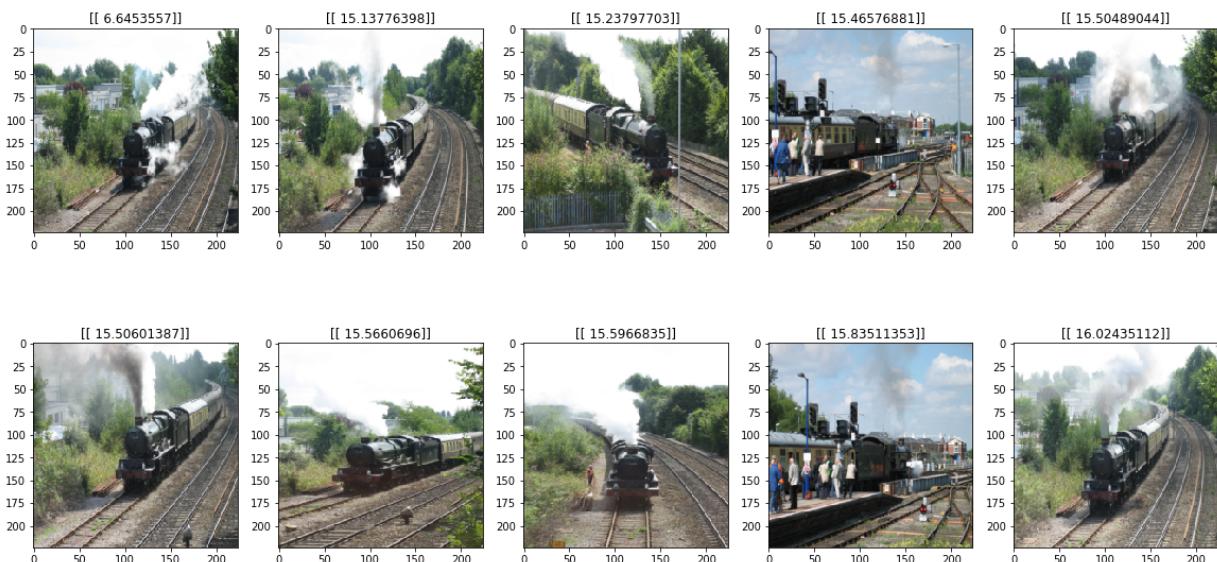
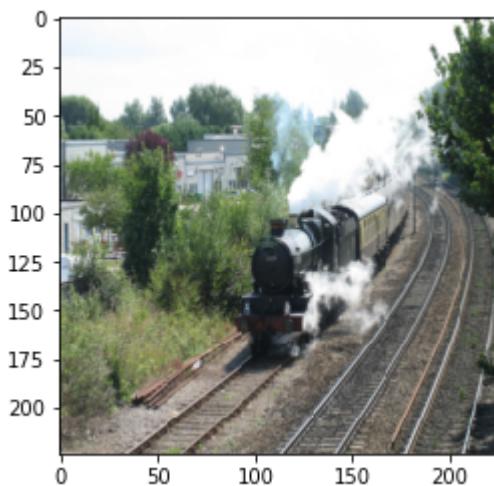
```
In [19]: query_index = index_query('san_fran_selected/18/PCI_sp_12357_37.793396_-122.39295')
results = find_query(query_index, indexes, 10)
plot_results(results)
```



```
In [20]: query_index = index_query('san_fran_selected/16/PCI_sp_12318_37.792333_-122.39429')
results = find_query(query_index, indexes, 10)
plot_results(results)
```



```
In [45]: query_index = index_query('oxford_selected/railway/worcester_000131.jpg', cnn_mod)
results = find_query(query_index, indexes, 10)
plot_results(results)
```



test with freshly indexed images

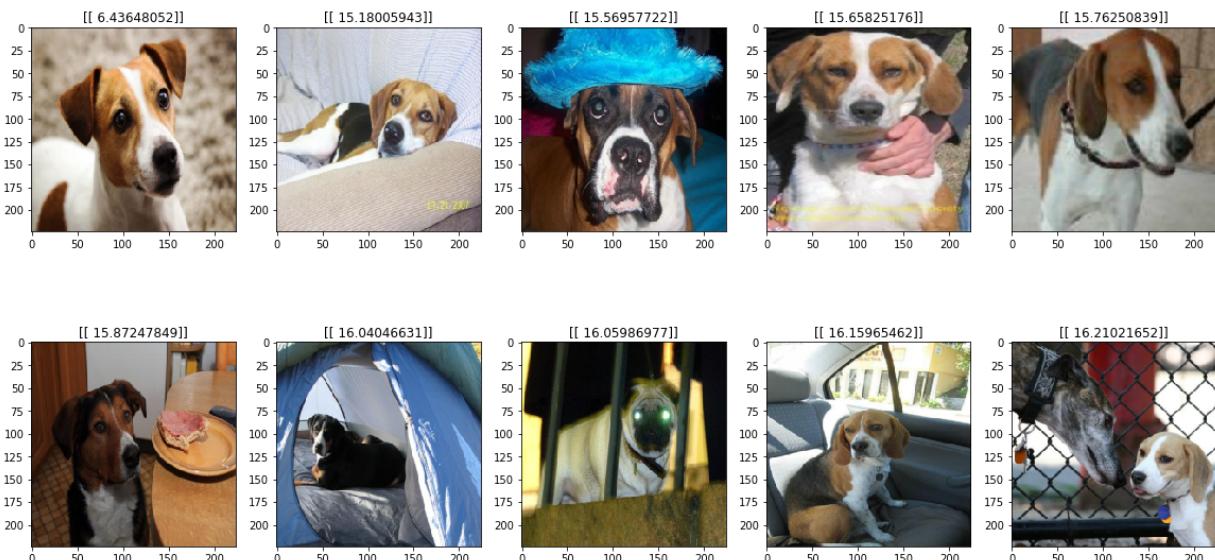
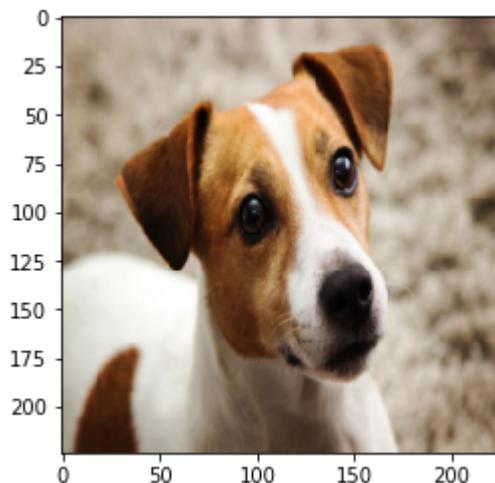
```
In [30]: # read the test indexes
with open('test_index.pickle', 'rb') as f:
    test_indexes = pickle.load(f)
```

```
In [31]: a = indexes
a = a + test_indexes
print (len(a))
print (len(indexes))
```

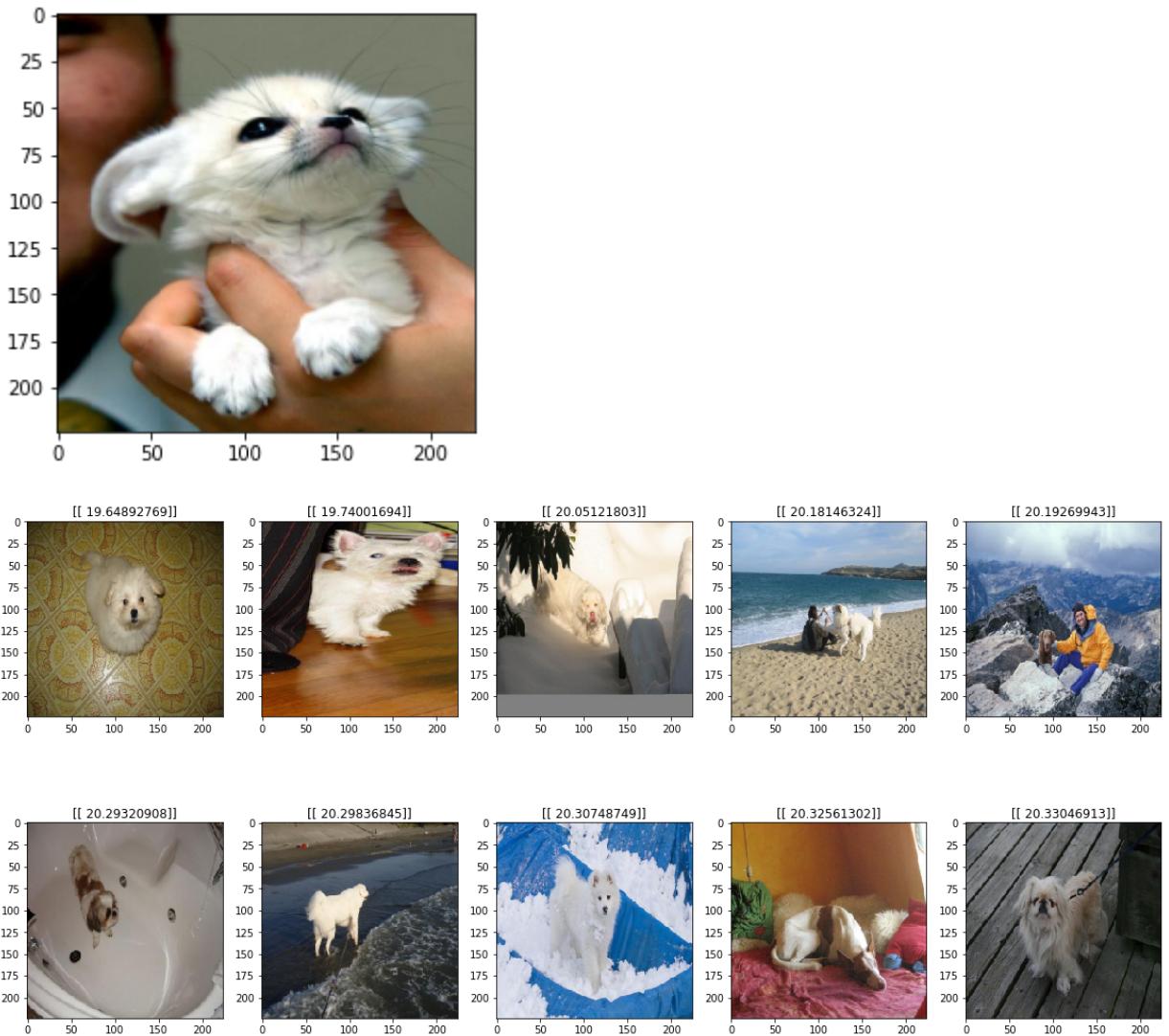
23936
23925

```
In [32]: indexes = a
```

```
In [33]: #image doesnt exist in indexes that were trained on
#retrieving from freshly appended indexes
query_index = index_query('test_images/kukkur.jpg', cnn_model)
results = find_query(query_index, indexes, 10)
plot_results(results)
```

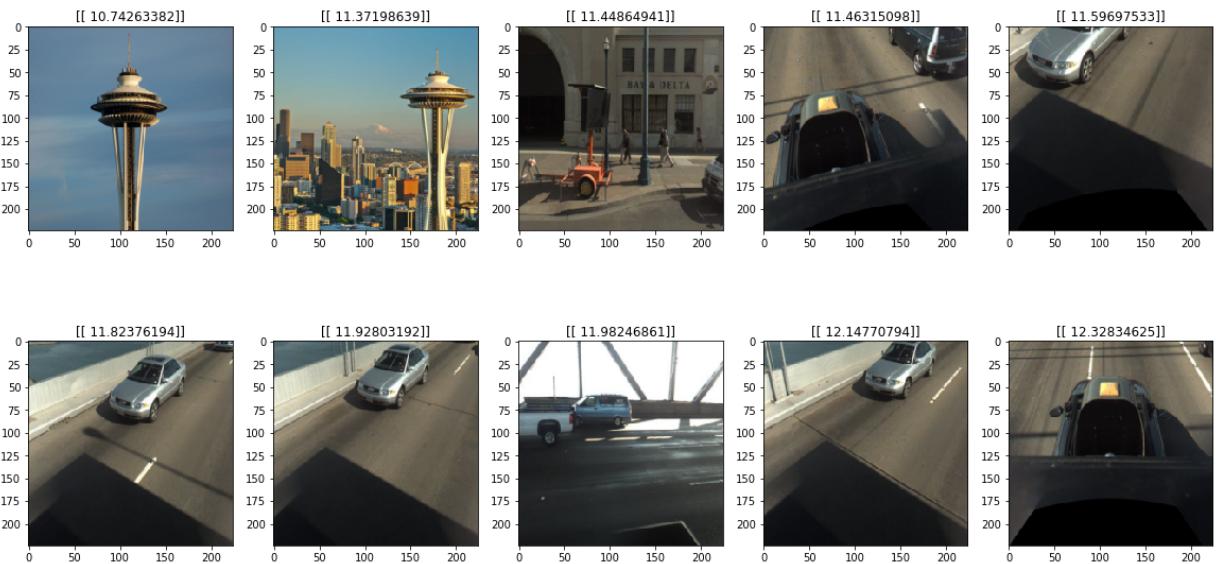
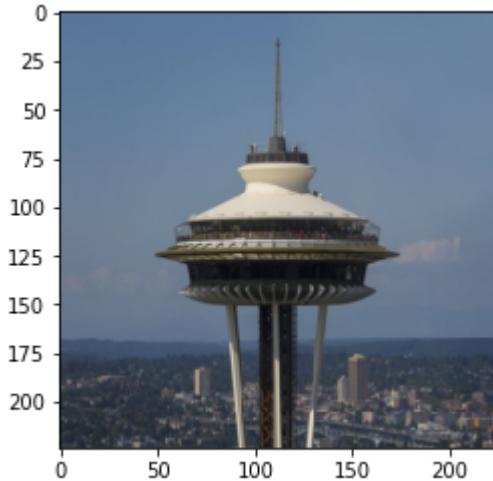


```
In [97]: #image doesnt exist in indexes that were trained on
#retrieving from freshly appended indexes
# exact image did not exist in the database ... image taken from the internet...
query_index = index_query('test_images/kutta.jpg', cnn_model)
results = find_query(query_index, indexes, 10)
plot_results(results)
```

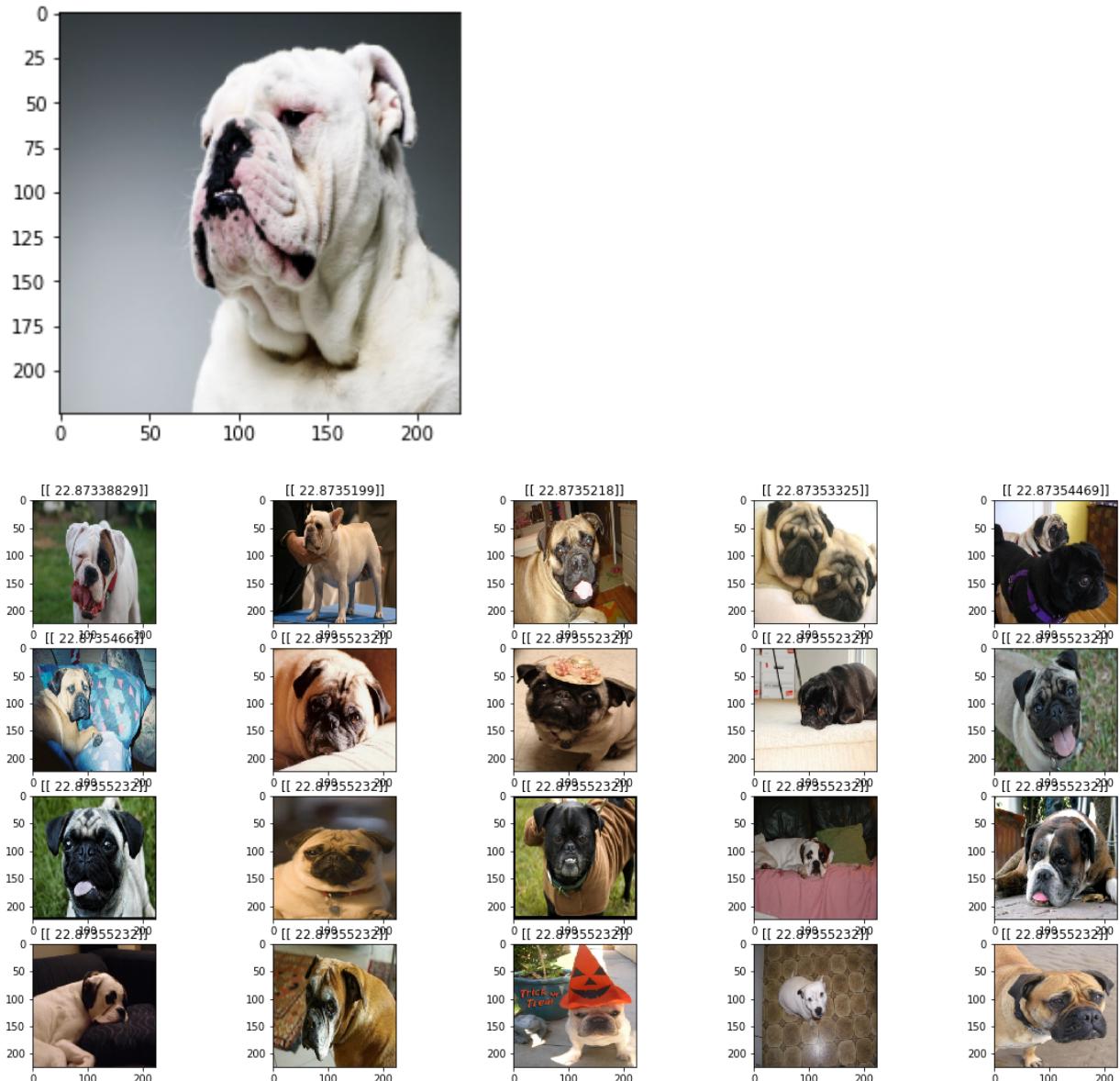


This result is from an intermediate version of the model, the final version results differ as shown during L2 comparison below

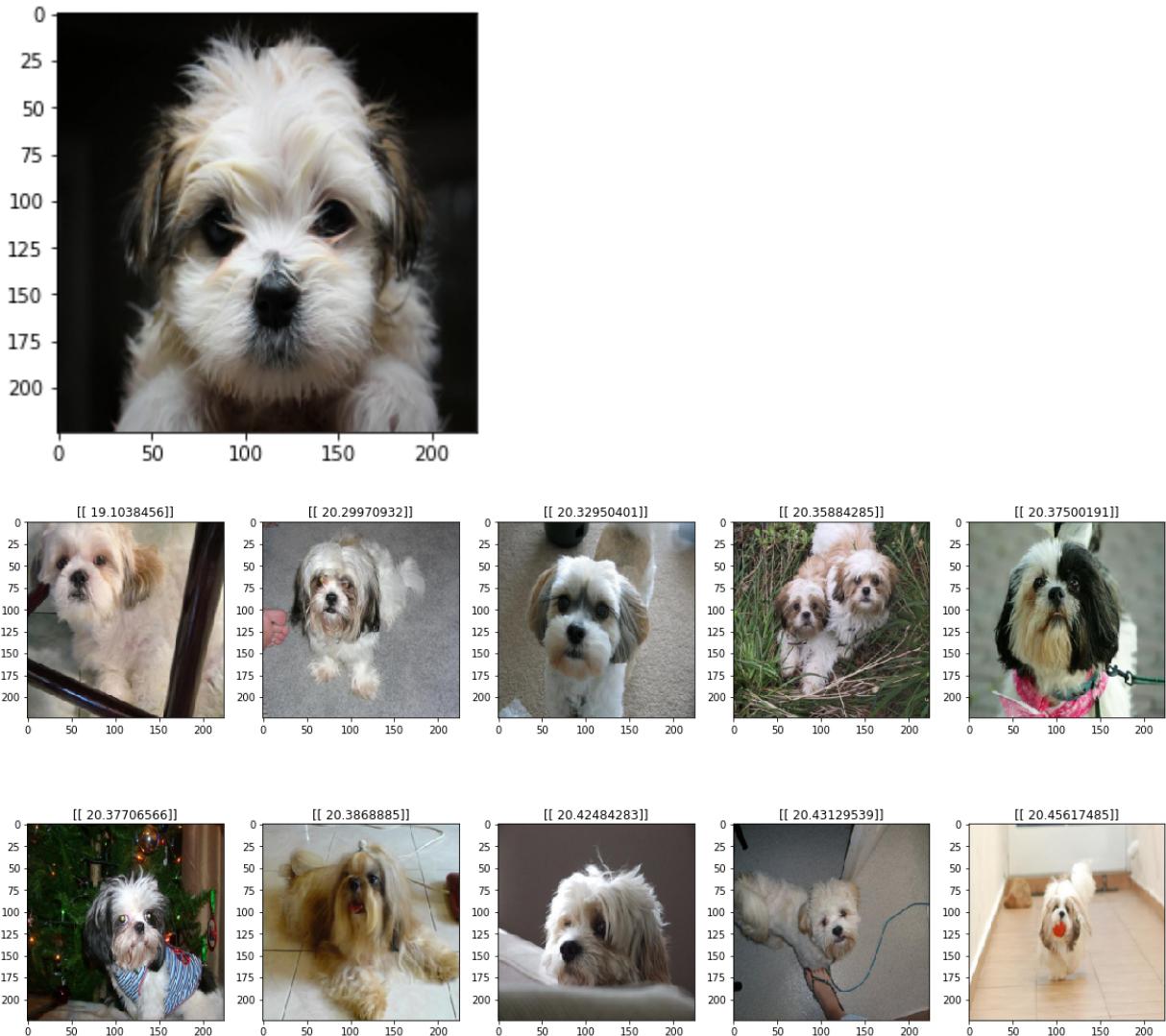
```
In [27]: # EXACT image doesnt exist in indexes that were trained on
#retrieving from freshly appended indexes
# 2 out of 3 retrieved in top 10
query_index = index_query('test_images/space_needle_1.jpg', cnn_model)
results = find_query(query_index, indexes, 10)
plot_results(results)
```



```
In [29]: # image downloaded from internet
# EXACT image doesn't exist in indexes that were trained on
query_index = index_query('test_images/6-face.jpg', cnn_model1)
results = find_query(query_index, indexes, 20)
plot_results(results)
```



```
In [103]: # image downloaded from internet
# EXACT image doesn't exist in indexes that were trained on
query_index = index_query('test_images/puppy_photo_on_flickr.jpg', cnn_model)
results = find_query(query_index, indexes, 10)
plot_results(results)
```



Augmentation test

```
In [40]: # image downloaded from quora
# what explains the match?!
# windows in oxford are vertical but the san-fransisco windows are horizontal

query_index = index_query('test_rotated', cnn_model)
results = find_query(query_index, indexes, 10)
plot_results(results)
```



Evaluation against L2 (Euclidean Distance)

Relevant Results are defined by visually inspecting the results of each search and appending the same artefact as in the query manually into the relevant list

First 10 results are from the Neural Net Model and the last 10 are from L2 Search

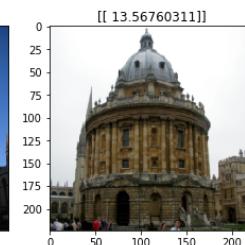
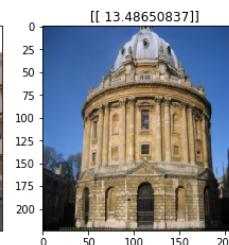
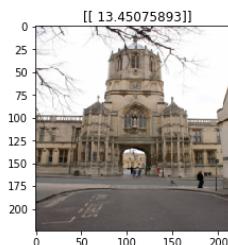
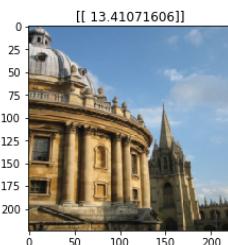
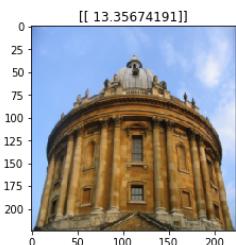
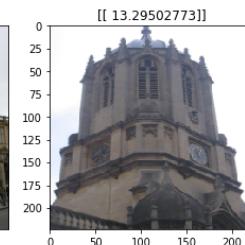
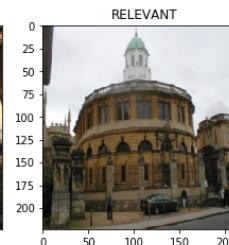
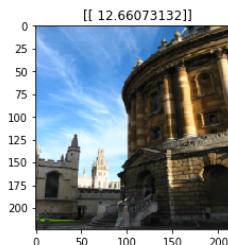
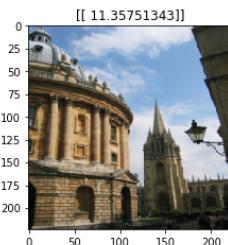
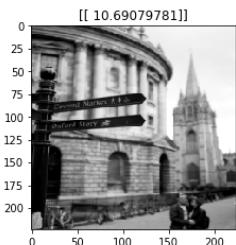
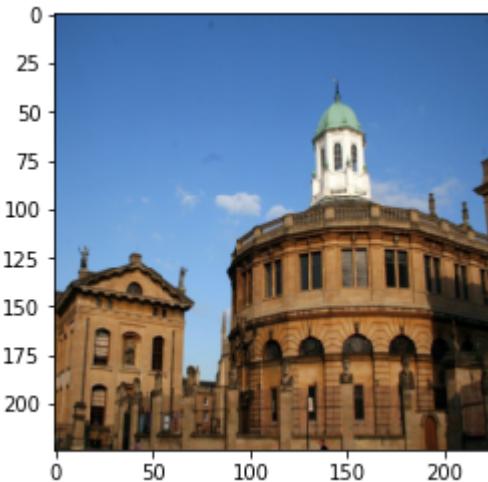
The relevant image with this case is being retrieved by the Neural Network Model, BUT NOT THE EUCLIDEAN DISTANCE SEARCH

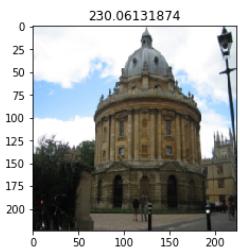
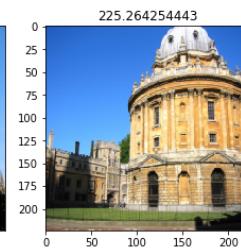
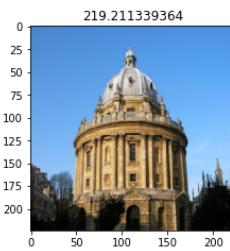
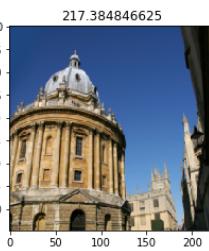
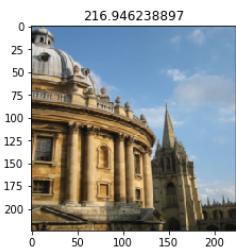
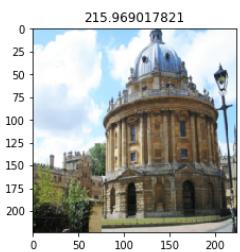
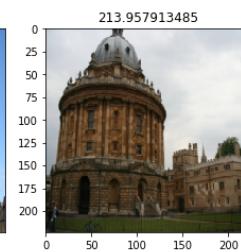
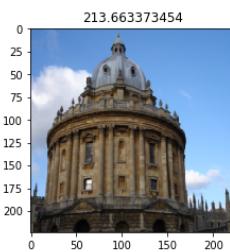
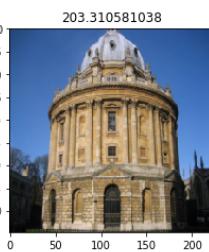
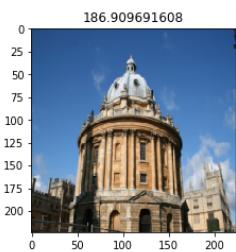
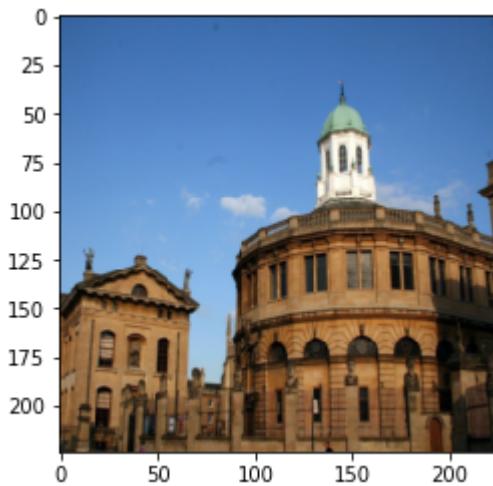
```
In [44]: # EXACT TEST IMAGE DOES NOT EXIST IN THE DATABASE

# RELEVANT IMAGE IS MARKED AS "RELEVANT", USING 'highlight' INSTEAD OF SCORE

query_index = index_query('main-qimg-a2233912a89872a2076e6799392a7f44-c', cnn_mod)
results = find_query(query_index, indexes, 10)
plot_results(results, highlight = 3)

query_index = index_query('main-qimg-a2233912a89872a2076e6799392a7f44-c', cnn_mod)
results_l2 = find_query_l2(query_index, indexes, 10)
plot_results(results_l2)
```

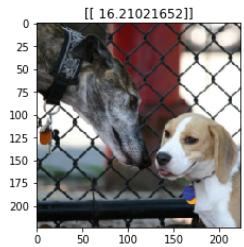
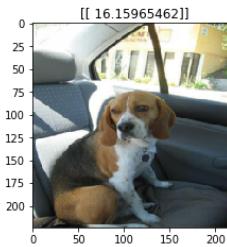
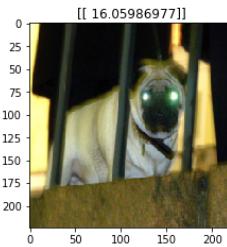
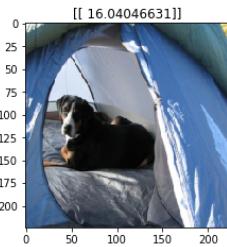
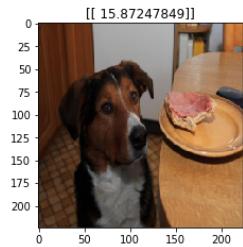
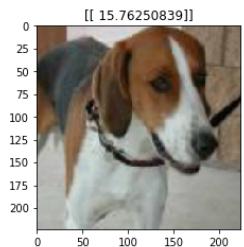
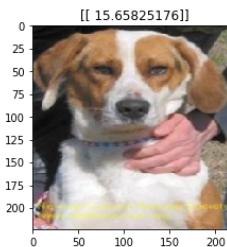
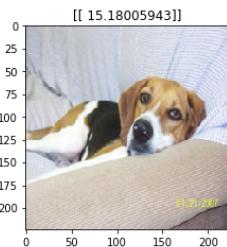
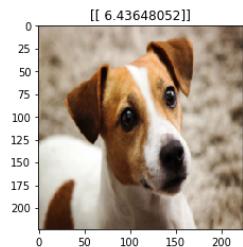
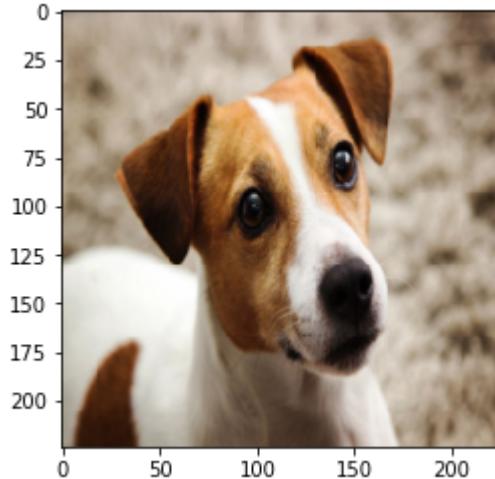


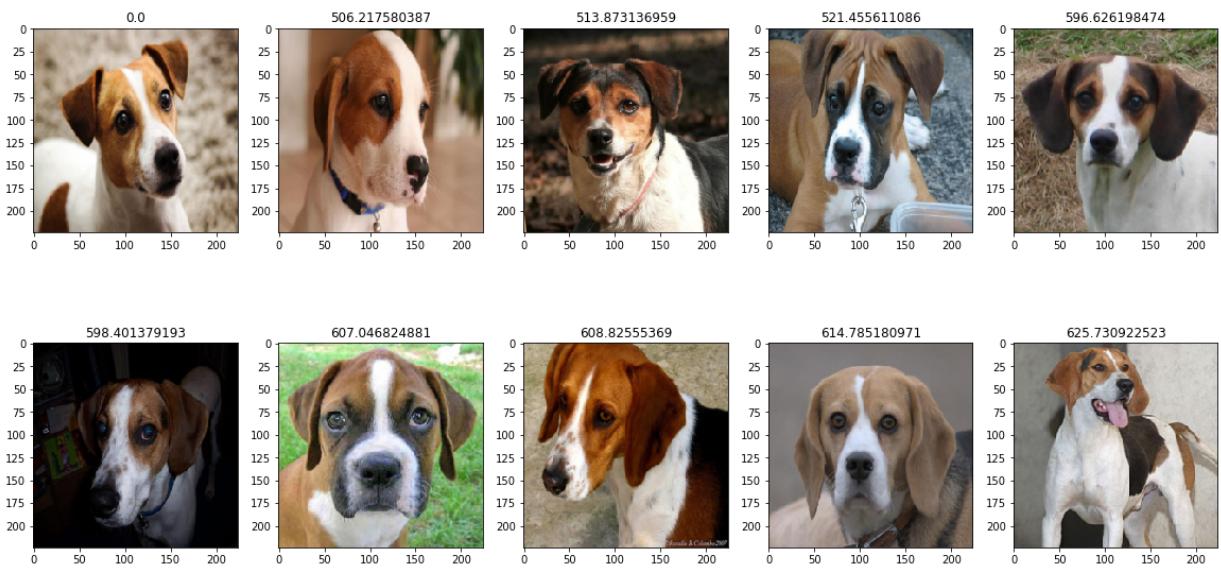
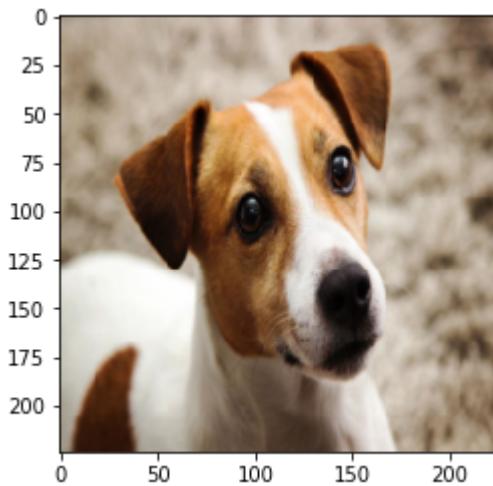


In [41]: *#image doesnt exist in indexes that were trained on
#retrieving from freshly appended indexes*

```
query_index = index_query('test_images/kukkur.jpg', cnn_model)
results = find_query(query_index, indexes, 10)
plot_results(results)
```

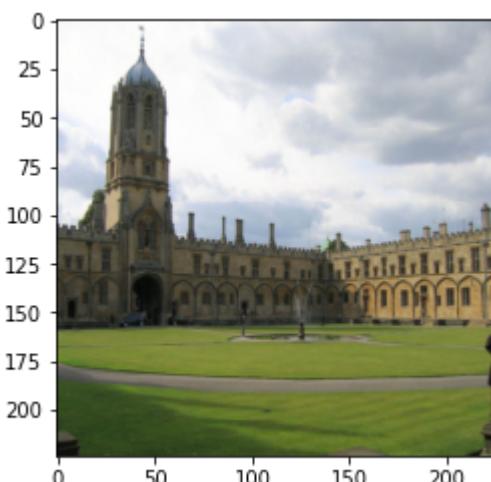
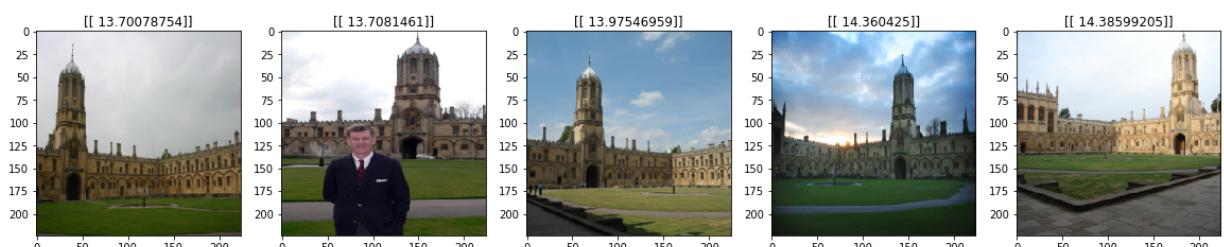
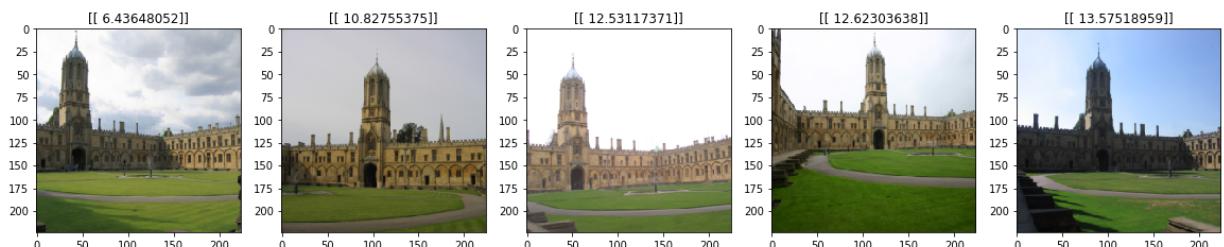
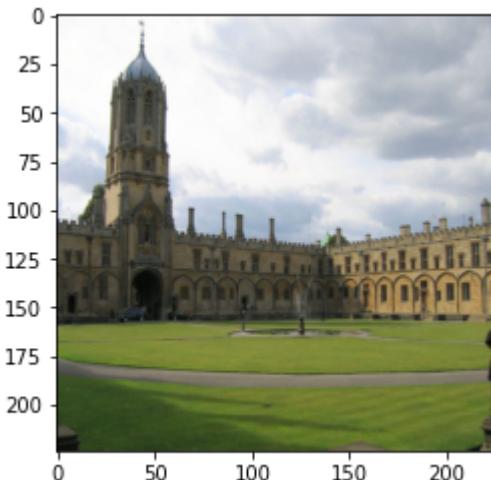
```
query_index = index_query('test_images/kukkur.jpg', cnn_model)
results_12 = find_query_12(query_index, indexes, 10)
plot_results(results_12)
```



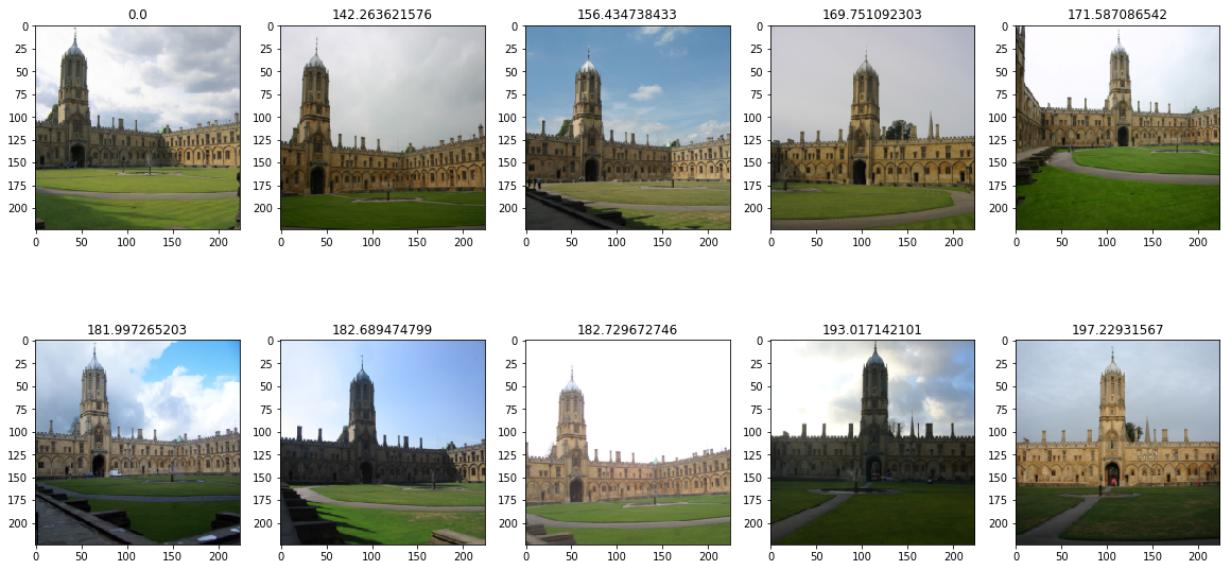


```
In [43]: query_index = index_query('oxford_selected/Christ_Church/christ_church_000435.jpg')
results = find_query(query_index, indexes, 10)
plot_results(results)

query_index = index_query('oxford_selected/Christ_Church/christ_church_000435.jpg')
results_12 = find_query_12(query_index, indexes, 10)
plot_results(results_12)
```



execute test



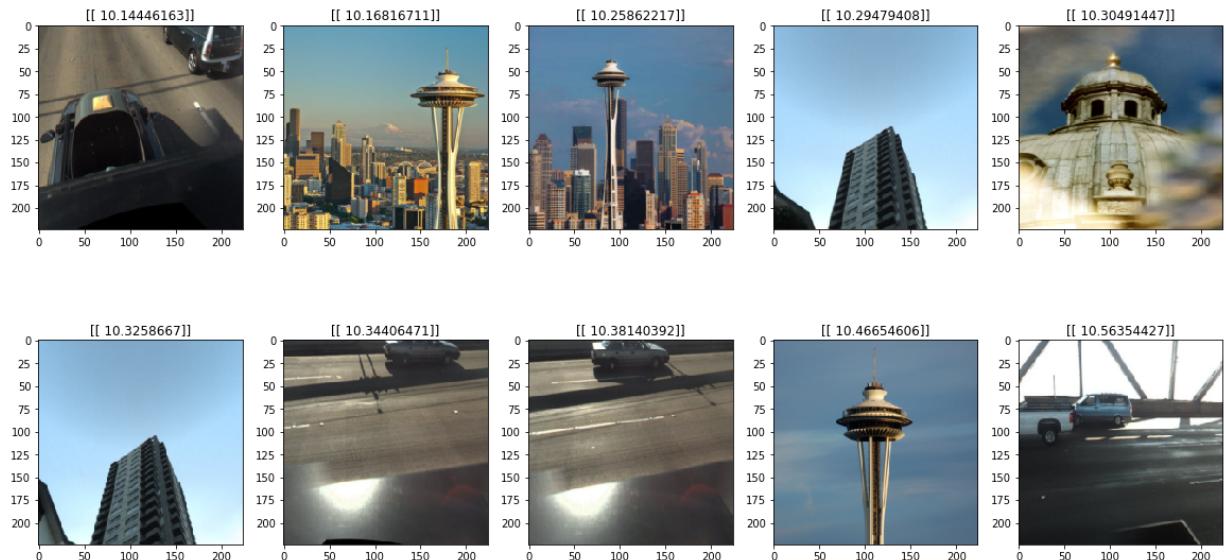
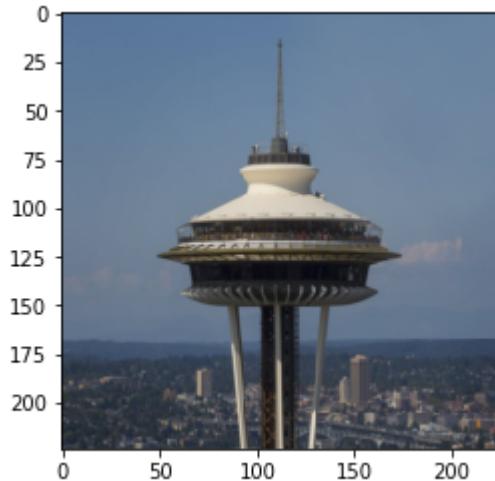
ALL THREE SPACE NEEDLE IMAGES ARE NOW BEING RETRIEVED BY THE NEURAL NETWORK MODEL

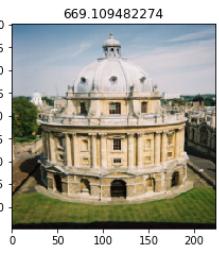
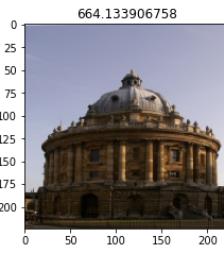
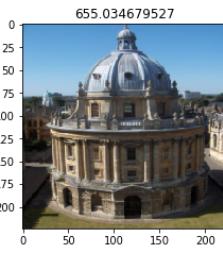
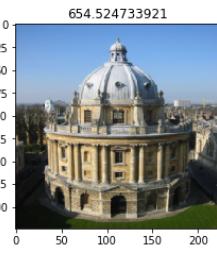
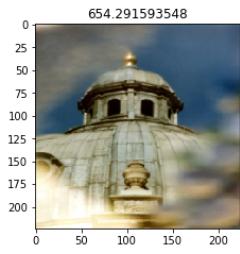
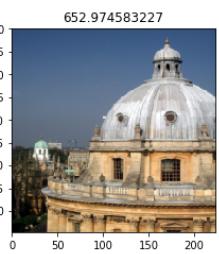
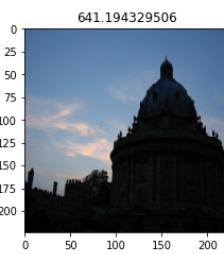
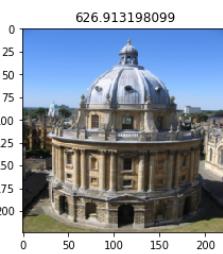
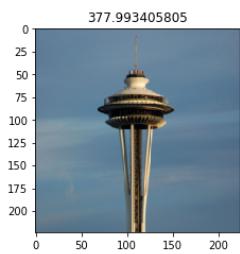
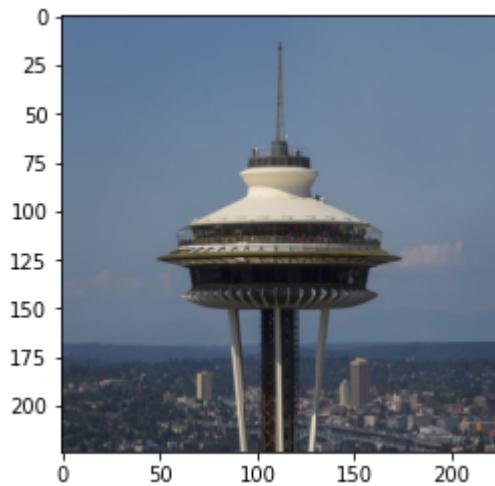
NOT TRUE FOR L2 SEARCH

Also a similar search result as L2 here has also been retrieved by an earlier version of NEURAL NETWORK as demonstrated in one of the tests above

```
In [49]: # EXACT image doesnt exist in indexes that were trained on
#retrieving from freshly appended indexes
# 3 out of 3 retrieved in top 10
query_index = index_query('test_images/space_needle_1.jpg', cnn_model)
results = find_query(query_index, indexes, 10)
plot_results(results)

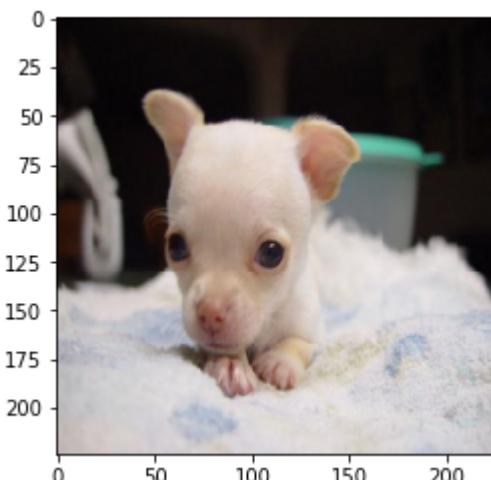
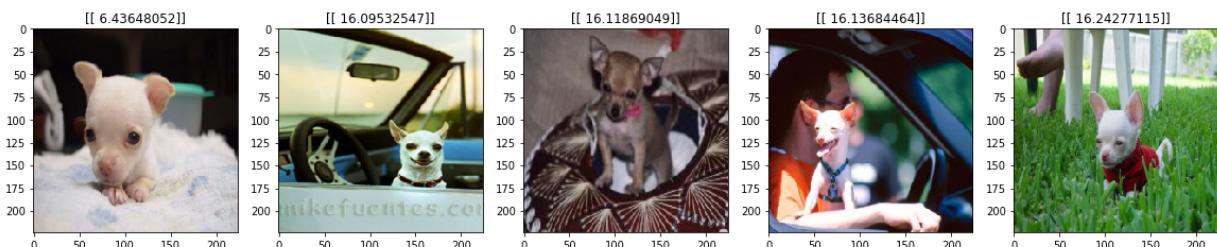
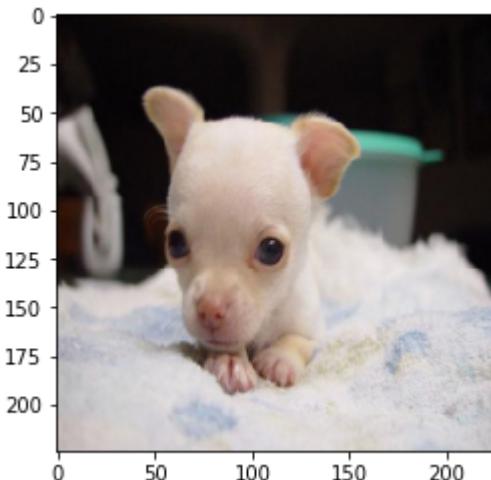
query_index = index_query('test_images/space_needle_1.jpg', cnn_model)
results_12 = find_query_12(query_index, indexes, 10)
plot_results(results_12)
```



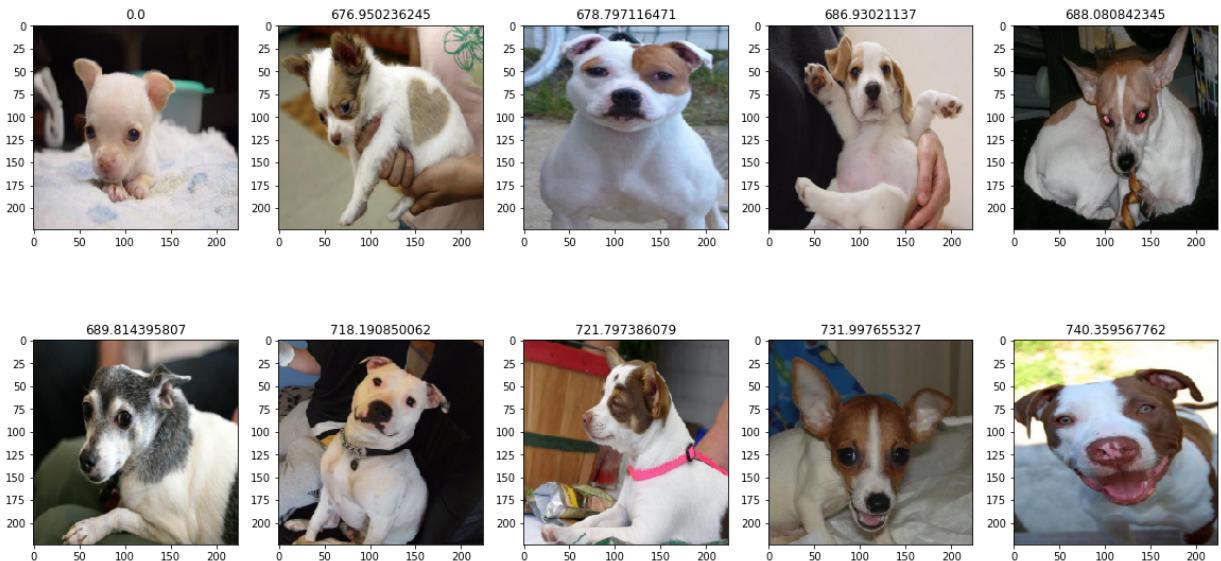


```
In [50]: query_index = index_query('Dogs/n02085620-Chihuahua/n02085620_712.jpg', cnn_model
results = find_query(query_index, indexes, 10)
plot_results(results)

query_index = index_query('Dogs/n02085620-Chihuahua/n02085620_712.jpg', cnn_model
results_12 = find_query_12(query_index, indexes, 10)
plot_results(results_12)
```



execute test



to interpret the above results I have printed the results below in text

NOTICE HOW ALL OF THE FIRST 5 IMAGES RETRIEVED FROM THE NEURAL NETWORK BELONG TO THE SAME CLASS "Chihuahua" AS THE QUERY IMAGE

THE NEURAL NETWORK RETRIEVES 7 IMAGES FROM THE SAME CLASS

L2 ONLY RETRIEVES 4

WITH A CLASS BASED EVALUATION SCHEME NEURAL NETWORK OUTPERFORMS L2 HERE TOO.

In [54]: `for res in results:
 print (res)`

```
('Dogs/n02085620-Chihuahua/n02085620_712.jpg', array([[ 6.43648052]], dtype=float32))
('Dogs/n02085620-Chihuahua/n02085620_12334.jpg', array([[ 16.09532547]], dtype=float32))
('Dogs/n02085620-Chihuahua/n02085620_7292.jpg', array([[ 16.11869049]], dtype=float32))
('Dogs/n02085620-Chihuahua/n02085620_4572.jpg', array([[ 16.13684464]], dtype=float32))
('Dogs/n02085620-Chihuahua/n02085620_431.jpg', array([[ 16.24277115]], dtype=float32))
('Dogs/n02110958-pug/n02110958_6274.jpg', array([[ 16.30970955]], dtype=float32))
('Dogs/n02113978-Mexican_hairless/n02113978_143.jpg', array([[ 16.39067078]], dtype=float32))
('Dogs/n02085620-Chihuahua/n02085620_11337.jpg', array([[ 16.52665138]], dtype=float32))
('Dogs/n02111500-Great_Pyrenees/n02111500_409.jpg', array([[ 16.58696365]], dtype=float32))
('Dogs/n02085620-Chihuahua/n02085620_10621.jpg', array([[ 16.59844208]], dtype=float32))
```

In [52]: `for res in results_12:
 print (res)`

```
('Dogs/n02085620-Chihuahua/n02085620_712.jpg', 0.0)
('Dogs/n02085620-Chihuahua/n02085620_1346.jpg', 676.95023624452733)
('Dogs/n02093256-Staffordshire_bullterrier/n02093256_14608.jpg', 678.79711647121781)
('Dogs/n02088364-beagle/n02088364_17314.jpg', 686.93021137002393)
('Dogs/n02085620-Chihuahua/n02085620_1620.jpg', 688.08084234482544)
('Dogs/n02087046-toy_terrier/n02087046_3462.jpg', 689.81439580714493)
('Dogs/n02093256-Staffordshire_bullterrier/n02093256_2737.jpg', 718.19085006189073)
('Dogs/n02085620-Chihuahua/n02085620_5312.jpg', 721.79738607890465)
('Dogs/n02087046-toy_terrier/n02087046_5386.jpg', 731.9976553271149)
('Dogs/n02093428-American_Staffordshire_terrier/n02093428_2199.jpg', 740.3595677623772)
```

In []:

END OF TESTS

FINAL NOTE: TO UPDATE THE NEURAL NETWORK I ALSO CHANGED THE GROUND TRUTHS OF THE TRAINING DATASET

**.... TO A DIFFERENT SET OF PRIME NUMBERS IN
THE FOLLOWING ORDER : 7, 11, 17, 37 AND
71**

In []: