Video Captioning

(Machine Learning)

Mid Term Report

Department Of Computer Engineering & Application

Institute Of Engineering & Technology

SubmittedTo:

Mr. Vaibhav Diwan
(Assistant Professor)

SubmittedBy:

Kaustubh Sisodia (181500318)
Aman Vikrant
Garg (181500080)
Shivam Dubey(181500667)
Sudheer Kumar(181500731)

Acknowledgement

We owe special debt of gratitude to Mr. Vaibhav Diwan,Assistant Professor Department of CEA , for providing us with an encouraging platform to develop this project , which thus helped us in shaping our abilities towards a constructive goal and for his constant support and guidance to ourwork.
His sincerity, Thoroughness and perseverance is been a constant source of inspiration for us. We believe that he will shower us with all his Extensively experienced ideas and insightful comments at differentstages of the project & also taught us about the latest industry-oriented technologies.

We also do not like to miss the opportunity to acknowledge the contribution of all faculty members of the department for their kind guidance and co-operation.

Kaustubh Sisodia (181500318)
Aman Vikrant Garg (181500080)
Shivam Dubey(181500667)
Sudheer Kumar(181500731)

Table of Contents

Video Captioning

(Machine Learning)

Deep Learning is a very rampant field right now – with so many applications coming out day by day. And the best way to get deeper into Deep Learning is to get hands-on with it. Take up as much projects as you can, and try to do them on your own. This would help you grasp the topics in more depth and assist you in becoming a better Deep Learning practitioner.

In this article, we will take a look at an interesting multi modal topic where we will combine both image and text processing to build a useful Deep Learning application, aka Image Captioning. Image Captioning refers to the process of generating textual description from an image – based on the objects and actions in the image.

This process has many potential applications in real life. A noteworthy one would be to save the captions of an image so that it can be retrieved easily at a later stage just on the basis of this description.

you know the basics of Deep Learning and have previously worked on image processing problems using CNN. If you want to brush up on the concepts, you can go through these articles first:

Fundamentals of Deep Learning – Starting with Artificial Neural Network

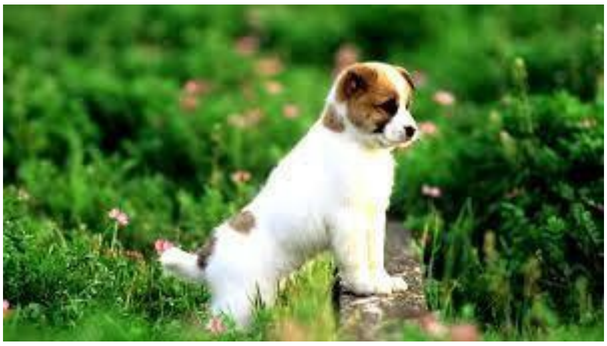
Architecture of Convolutional Neural Networks (CNNs) demystified

Tutorial: Optimizing Neural Networks using Keras (with Image recognition case study)

Essentials of Deep Learning – Sequence to Sequence modelling with Attention (using python)

About the Project

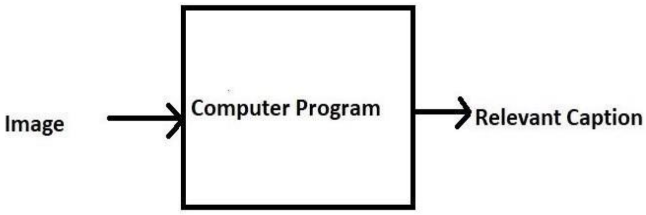
What do you see in the below picture?



Well some of you might say "**A white dog in a grassy area**", some may say "**White dog with brown spots**" and yet some others might say "**A dog on grass and some pink flowers**".

Definitely all of these captions are relevant for this image and there may be some others also. But the point I want to make is; it's so easy for us, as human beings, to just have a glance at a pictureanddescribeitinanappropriatelanguage.Evena5yearoldcoulddothiswithutmost ease.

But,canyouwriteacomputerprogramthattakesanimageasinputandproducesarelevant caption asoutput?



Just prior to the recent development of Deep Neural Networks this problem was inconceivable even by the most advanced researchers in Computer Vision. But with the advent of Deep Learning this problem can be solved very easily if we have the required dataset.

This problem was well researched by Andrej Karapathy in his PhD thesis at Stanford [1], who is also now the Director of AI atTesla.

The purpose of this blog post is to explain (in as simple words as possible) that how Deep Learning can be used to solve this problem of generating a caption for a given image, hence the name Image Captioning.
To get a better feel of this problem, I strongly recommend to use this state-of-the-art system created by Microsoft called as Caption Bot. Just go to this link and try uploading any picture you want; this system will generate a caption for it.
A quick glance is sufficient for you to understand and describe what is happening in the picture. Automatically generating this textual description from an artificial system is the task of image captioning.

The task is straightforward – the generated output is expected to describe in a single sentence what is shown in the image – the objects present, their properties, the actions being performed and the interaction between the objects, etc. But to replicate this behaviour in an artificial system is a huge task, as with any other image processing problem and hence the use of complex and advanced techniques such as Deep Learning to solve the task.

Content-based image retrieval, also known as query by image content (QBIC) and content-based visual information retrieval (CBVIR), is the application of computer vision techniques to the image retrieval problem, that is, the problem of searching for digital images in large databases (see this survey[1] for a recent scientific overview of the CBIR field). Content-based image retrieval is opposed to traditional concept-based approaches (see Concept-based image indexing).

"Content-based" means that the search analyzes the contents of the image rather than the metadata such as keywords, tags, or descriptions associated with the image. The term "content" in this context might refer to colors, shapes, textures, or any other information that can be derived from the image itself. CBIR is desirable because searches that rely purely on metadata are dependent on annotation quality and completeness.

Having humans manually annotate images by entering keywords or metadata in a large database can be time consuming and may not capture the keywords desired to describe the image. The evaluation of the effectiveness of keyword image search is subjective and has not been well-defined. In the same regard, CBIR systems have similar challenges in defining success. "Keywords also limit the scope of queries to the set of predetermined criteria." and, "having been set up" are less reliable than using the content itself.

Project Description

```
importstring
importglob

fromtensorflow.keras.applications.inception_v3import InceptionV3
importtensorflow.keras.applications.inception_v3


#from tqdm import tqdm
importtensorflow.keras.preprocessing.image
importpickle
#from time import time
importnumpyasnp
fromPILimport Image
fromtensorflow.keras.modelsimport Sequential
fromtensorflow.keras.layersimport LSTM, Embedding, TimeDistributed, Dense,
RepeatVector,\
                         Activation, Flatten, Reshape, concatenate, Dropout,
BatchNormalization
fromtensorflow.keras.optimizersimport Adam, RMSprop
fromtensorflow.kerasimport Input, layers
fromtensorflow.kerasimport optimizers

fromtensorflow.keras.modelsimport Model

fromtensorflow.keras.layersimport add
fromtensorflow.keras.preprocessing.sequenceimport pad_sequences
fromtensorflow.keras.utilsimport to_categorical
importmatplotlib.pyplotasplt

START ="startseq"
STOP ="endseq"

root_captioning ="D:\\datasets\\image_caption_data"



encode_model = InceptionV3(weights='imagenet')
encode_model = Model(encode_model.input, encode_model.layers[-2].output)
WIDTH =299
HEIGHT =299
OUTPUT_DIM =2048
preprocess_input = tensorflow.keras.applications.inception_v3.preprocess_input
WARNING:tensorflow:From D:\anaconda\lib\site-
packages\tensorflow\python\ops\resource_variable_ops.py:435: colocate_with (from
tensorflow.python.framework.ops) is deprecated and will be removed in a future version.
Instructions for updating:
```

```
Colocations handled automatically by placer.

defencodeImage(img):
# Resize all images to a standard size (specified bythe image encoding network)
    img = img.resize((WIDTH, HEIGHT), Image.ANTIALIAS)
# Convert a PIL image to a numpy array
    x = tensorflow.keras.preprocessing.image.img_to_array(img)
# Expand to 2D array
    x = np.expand_dims(x, axis=0)
# Perform any preprocessing needed by InceptionV3 or others
    x = preprocess_input(x)
# Call InceptionV3 (or other) to extract the smaller feature set for the image.
    x = encode_model.predict(x) # Get the encoding vector for the image
# Shape to correct form to be accepted by LSTM captioning network.
    x = np.reshape(x, OUTPUT_DIM )
return x
```

```
test_path = os.path.join(root_captioning,"data",f'Vocab.pkl')
withopen(test_path, "rb") as fp:
        vocab = pickle.load(fp)


idxtoword = {}
wordtoidx = {}

ix =1
for w in vocab:
    wordtoidx[w] = ix
    idxtoword[ix] = w
    ix +=1

vocab_size =len(idxtoword) +1
max_length =34
```

In [3]:

```
test_path = os.path.join(root_captioning,"data",f'embedding_matrix.pkl')
withopen(test_path, "rb") as fp:
        embedding_matrix = pickle.load(fp)

print(embedding_matrix)
embedding_matrix.shape
[[ 0.          0.          0.         ...  0.          0.
   0.        ]
 [ 0.          0.          0.         ...  0.          0.
   0.        ]
 [ 0.49340001 -0.0085568  -0.4605     ... -0.55435002 -0.57477999
```

```
  -0.013045  ]
 ...
 [-0.021736    0.15308     0.11436    ...  0.078342   -0.39166999
   0.12937   ]
 [ 0.20298    -0.10048    -0.33627    ...  0.40496999 -0.19888
  -0.10866   ]
 [ 0.41156    -0.25863001  0.016209   ... -0.25419    -0.27496001
   0.67840999]]
```

Out[3]:

```
(1652, 200)
```

In [4]:

```
max_length =34
embedding_dim =200
inputs1 = Input(shape=(OUTPUT_DIM,))
fe1 = Dropout(0.5)(inputs1)
fe2 = Dense(256, activation='relu')(fe1)
inputs2 = Input(shape=(max_length,))
se1 = Embedding(vocab_size, embedding_dim, mask_zero=True)(inputs2)
se2 = Dropout(0.5)(se1)
se3 = LSTM(256)(se2)
decoder1 = add([fe2, se3])
decoder2 = Dense(256, activation='relu')(decoder1)
outputs = Dense(vocab_size, activation='softmax')(decoder2)
caption_model = Model(inputs=[inputs1, inputs2], outputs=outputs)
```

```
WARNING:tensorflow:From D:\anaconda\lib\site-
packages\tensorflow\python\keras\layers\core.py:143: calling dropout (from
tensorflow.python.ops.nn_ops) with keep_prob is deprecated and will be removed in a
future version.
Instructions for updating:
Please use `rate` instead of `keep_prob`. Rate should be set to `rate = 1 - keep_prob`.
```

In [5]:

```
caption_model.summary()
```

```
________________________________________________________________________________________
___________
Layer (type)                    Output Shape         Param #     Connected to
========================================================================================
===========
input_3 (InputLayer)            (None, 34)           0
________________________________________________________________________________________
___________
input_2 (InputLayer)            (None, 2048)         0
________________________________________________________________________________________
___________
embedding (Embedding)           (None, 34, 200)      330400      input_3[0][0]
________________________________________________________________________________________
___________
dropout (Dropout)               (None, 2048)         0           input_2[0][0]
________________________________________________________________________________________
___________
dropout_1 (Dropout)             (None, 34, 200)      0           embedding[0][0]
```

```
_______________________________________________________________________________________
___________
dense (Dense)                    (None, 256)          524544      dropout[0][0]
_______________________________________________________________________________________
___________
lstm (LSTM)                      (None, 256)          467968      dropout_1[0][0]
_______________________________________________________________________________________
___________
add (Add)                        (None, 256)          0           dense[0][0]
                                                                  lstm[0][0]
_______________________________________________________________________________________
___________
dense_1 (Dense)                  (None, 256)          65792       add[0][0]
_______________________________________________________________________________________
___________
dense_2 (Dense)                  (None, 1652)         424564      dense_1[0][0]
=======================================================================================
===========
Total params: 1,813,268
Trainable params: 1,813,268
Non-trainable params: 0
_______________________________________________________________________________________
___________
```

In [6]:

```
caption_model.layers[2].set_weights([embedding_matrix])
caption_model.layers[2].trainable =False
caption_model.compile(loss='categorical_crossentropy', optimizer='adam')

model_path = os.path.join(root_captioning,"data",f'caption-model.hdf5')
caption_model.load_weights(model_path)
```

In [7]:

```
defgenerateCaption(photo):
    in_text = START
for i inrange(max_length):
        sequence = [wordtoidx[w] for w in in_text.split() if w in wordtoidx]
        sequence = pad_sequences([sequence], maxlen=max_length)
        yhat = caption_model.predict([photo,sequence], verbose=0)
        yhat = np.argmax(yhat)
        word = idxtoword[yhat]
        in_text +=' '+ word
if word == STOP:
break
    final = in_text.split()
    final = final[1:-1]
    final =' '.join(final)
returnfinal
```

Requirements:

a) **Hardware Requirements (Minimum):**

i3 processor based computer
4GB Ram
Web Cam
5 GB Hard Disk Space

B) Software Requirements (Minimum):

Windows 7
Python 3.7
Python Modules

1. OpenCV2
2. Pandas
3. Numpy
4. Tensorflow
5. Keras

Technology Used:

A. Open ComputerVision
B. Python & MachineLearning
C. Classification
D.Convolution Neural Network (CNN/CovNets)
E. TransferLearning
F. InceptionV3Model
G. GRU’sLSTM

References

1. https://cs.stanford.edu/people/karpathy/cvpr2015.pdf
2. https://arxiv.org/abs/1411.4555
3. https://arxiv.org/abs/1703.09137
4. https://arxiv.org/abs/1708.02043
5. https://machinelearningmastery.com/develop-a-deep-learning-caption-generation-model-in-python/
6. https://www.youtube.com/watch?v=yk6XDFm3J2c
7. https://www.appliedaicourse.com/

Faculty guidance:

Mr. Vaibhav Diwan
(Assistant Professor)