Machine Learning

2228 - CSE 6363 - SEC 002

HW4 - REPORT

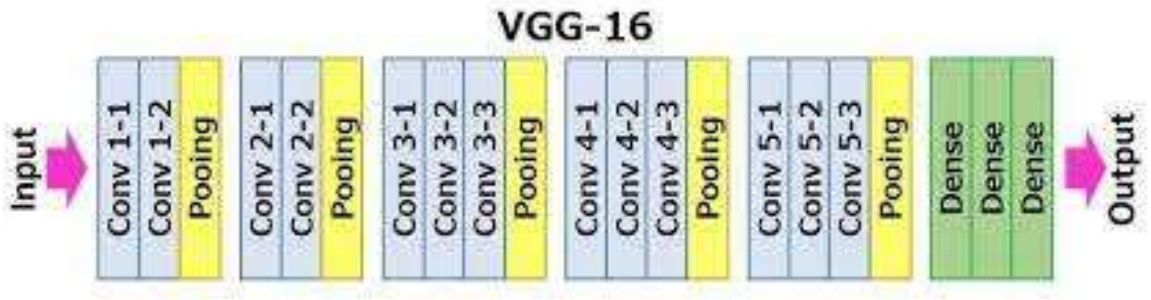
Names of Group members

1. Pratik Antoni Patekar (1001937948)
2. Amrita Singh (1001937490)
3. Ruthvik Kumar Myadam (1002026231)
4. Prathibha Lakkidi (1001962876)
5. Harshini Kandimalla (1001960046)

Results

- **a CNN model to encode image into a feature vector v**

The CNN model used is VGG-16:



VGG-16 is a convolutional neural network that is 16 layers deep. You can load a pre trained version of
the network trained on more than a million images from the ImageNet database.

```
import tensorflow as tf

vgg_model = tf.keras.applications.vgg16.VGG16(weights='imagenet')
VGG16Model = Model(inputs=vgg_model.inputs, outputs=vgg_model.layers[-2].output)
```

The output of the above cells is the layers and the output shape of the model, that is as follows,

```
Model: "model_23"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_43 (InputLayer)       [(None, 224, 224, 3)]     0

 block1_conv1 (Conv2D)       (None, 224, 224, 64)      1792

 block1_conv2 (Conv2D)       (None, 224, 224, 64)      36928

 block1_pool (MaxPooling2D)  (None, 112, 112, 64)      0

 block2_conv1 (Conv2D)       (None, 112, 112, 128)     73856

 block2_conv2 (Conv2D)       (None, 112, 112, 128)     147584

 block2_pool (MaxPooling2D)  (None, 56, 56, 128)       0

 block3_conv1 (Conv2D)       (None, 56, 56, 256)       295168

 block3_conv2 (Conv2D)       (None, 56, 56, 256)       590080

 block3_conv3 (Conv2D)       (None, 56, 56, 256)       590080

 block3_pool (MaxPooling2D)  (None, 28, 28, 256)       0
...
=================================================================
Total params: 134,260,544
Trainable params: 134,260,544
Non-trainable params: 0
_________________________________________________________________
```

We have used CNN model vgg16 to encode all the images. Before encoding we have removed the last two layers of the model. This will encode the image from the size of (224, 224, 3) to (1, 4096).

```
row_no = 502 # enter row no from 0 to 59999
img_name = df_train.iloc[row_no]['Image_path']
image = read_image(img_name)
encoded_image = VGG16Model.predict(image)
print(encoded_image.shape)
```

```
1/1 [==============================] - 0s 388ms/step
(1, 4096)
```

- **An LSTM with word2vec to encode each word in the question q_i**

We created a vocab dictionary of all the words existing in the questions and the answers, and then we sorted these words. We created a mapping of all these words to integers and vice-versa.

```
int_to_word = {}
word_to_int = {}
for int1 in range(len(some_list)):
  word = some_list[int1]
  int_to_word[int1] = word
  word_to_int[word] = int1
```

```
int_to_word[1000]
```

```
'playground?'
```

```
word_to_int['playground?']
```

```
1000
```

Following is the trained LSTM model that was used to convert questions into integers and pass it as input to the ***fusion model***.

```
Model: "model_29"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_48 (InputLayer)       [(None, 18)]              0

 embedding_28 (Embedding)    (None, 18, 300)           447300

 lstm_19 (LSTM)              (None, 18, 64)            93440

 lstm_20 (LSTM)              (None, 18, 64)            33024

 flatten_1 (Flatten)         (None, 1152)              0

 dense_48 (Dense)            (None, 1024)              1180672

=================================================================
Total params: 1,754,436
Trainable params: 1,307,136
Non-trainable params: 447,300
_________________________________________________________________
```

- **a fusion method to integrate v and `qi` into a matrix**

We have used a multiplication layer to fuse the image model and LSTM questions model followed by a bash normalization, dropout and dense.

```
Model: "model_31"
__________________________________________________________________________________________________
 Layer (type)                   Output Shape         Param #     Connected to
==================================================================================================
 input_48 (InputLayer)          [(None, 18)]         0           []

 embedding_28 (Embedding)       (None, 18, 300)      447300      ['input_48[0][0]']

 lstm_19 (LSTM)                 (None, 18, 64)       93440       ['embedding_28[0][0]']

 lstm_20 (LSTM)                 (None, 18, 64)       33024       ['lstm_19[0][0]']

 input_47 (InputLayer)          [(None, 4096)]       0           []

 flatten_1 (Flatten)            (None, 1152)         0           ['lstm_20[0][0]']

 dense_47 (Dense)               (None, 1024)         4195328     ['input_47[0][0]']

 dense_48 (Dense)               (None, 1024)         1180672     ['flatten_1[0][0]']

 multiply_6 (Multiply)          (None, 1024)         0           ['dense_47[0][0]',
                                                                  'dense_48[0][0]']

 batch_normalization_100 (Batch  (None, 1024)        4096        ['multiply_6[0][0]']
 Normalization)

 dropout_6 (Dropout)            (None, 1024)         0           ['batch_normalization_100[0][0]']

 dense_51 (Dense)               (None, 1000)         1025000     ['dropout_6[0][0]']

 dense_52 (Dense)               (None, 38)           38038       ['dense_51[0][0]']

==================================================================================================
Total params: 7,016,898
Trainable params: 6,567,550
Non-trainable params: 449,348
__________________________________________________________________________________________________
None
```

- **Report your accuracy on the validation set of VQA v1 dataset:**

For the final model we have used Adam optimiser with a learning rate of 0.001 with loss function as categorical crossentropy.

On the training dataset we achieved a maximum accuracy of 88.87%.

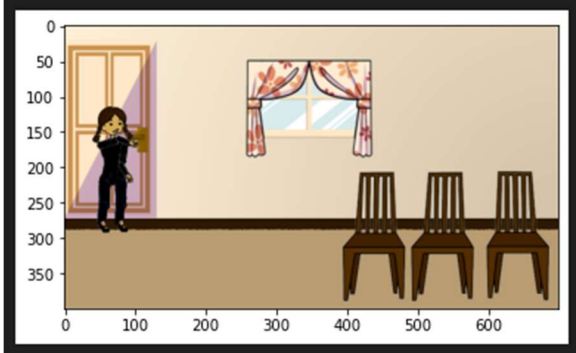
```
hist = FinalModel.fit(generator, steps_per_epoch=steps, epochs=5, verbose=1)
```

```
Epoch 1/5
1053/1053 [==============================] - 145s 133ms/step - loss: 0.6541 - accuracy: 0.7553
Epoch 2/5
1053/1053 [==============================] - 157s 149ms/step - loss: 0.4317 - accuracy: 0.8347
Epoch 3/5
1053/1053 [==============================] - 164s 155ms/step - loss: 0.3705 - accuracy: 0.8596
Epoch 4/5
1053/1053 [==============================] - 161s 153ms/step - loss: 0.3287 - accuracy: 0.8750
Epoch 5/5
1053/1053 [==============================] - 163s 155ms/step - loss: 0.2963 - accuracy: 0.8887
```

In the next section, we have attached screenshots of the results of the validation dataset. We have selected 30 random samples from the validation dataset.

- **Provide 30 randomly chosen examples from the validation set of VQA v1 dataset. Each example consists of a image, a question, the ground truth answer, the predicted answer, the attention map between the answer and words in the question, and the attention map of input image.**

```
How many chairs are on the right?
3
```

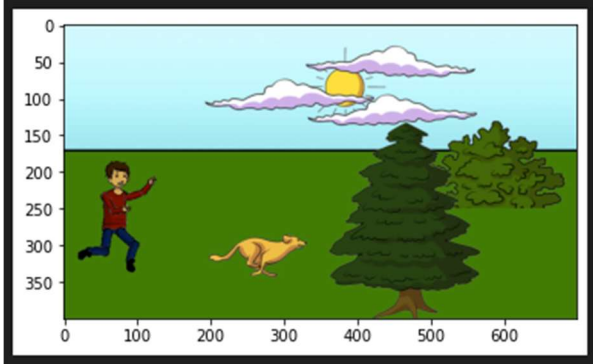



```
1/1 [==============================] - 0s 210ms/step
1/1 [==============================] - 0s 20ms/step
Answer      Prob
     3 0.738884
     1 0.252235
     2 0.008879
     4 0.000002
```

```
How many Teacups are on the table?
3
```

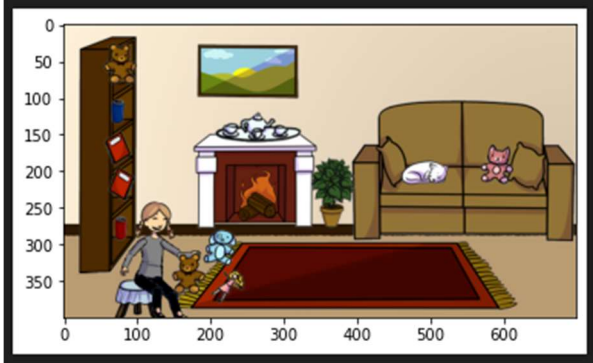

```
1/1 [==============================] - 0s 180ms/step
1/1 [==============================] - 0s 23ms/step
Answer      Prob
     3 0.997198
     2 0.002589
     1 0.000179
     5 0.000025
```

```
How many clouds in the sky?
3
```



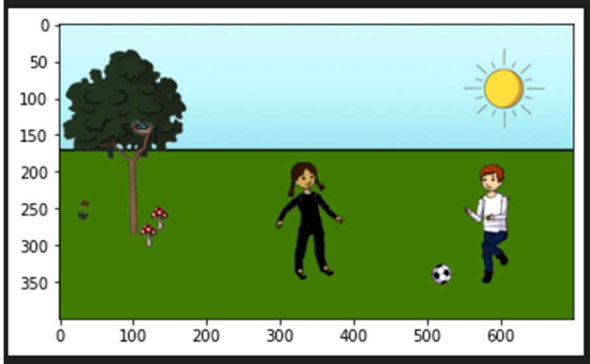
```
1/1 [==============================] - 0s 181ms/step
1/1 [==============================] - 0s 23ms/step
Answer     Prob
     3 0.904056
     2 0.048890
     1 0.046286
     5 0.000362
```

```
How many flames are in the fireplace?
1
```



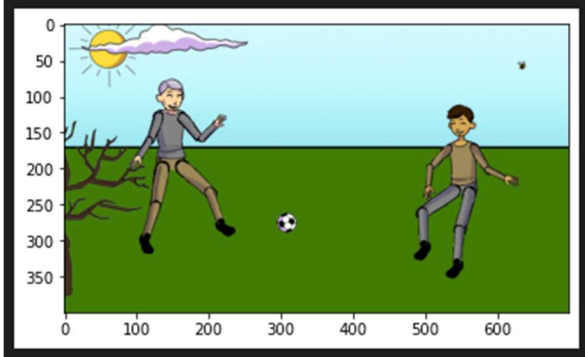
```
1/1 [==============================] - 0s 190ms/step
1/1 [==============================] - 0s 22ms/step
Answer     Prob
     1 0.765568
     2 0.233541
     3 0.000889
     5 0.000002
```

```
How many suns?
1
```



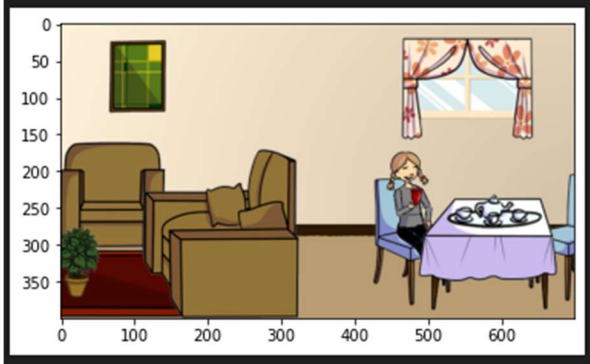

```
1/1 [==============================] - 0s 181ms/step
1/1 [==============================] - 0s 27ms/step
Answer         Prob
     1 9.999542e-01
     2 4.464129e-05
     3 1.165202e-06
     0 5.574016e-13
```

```
How many balls?
1
```



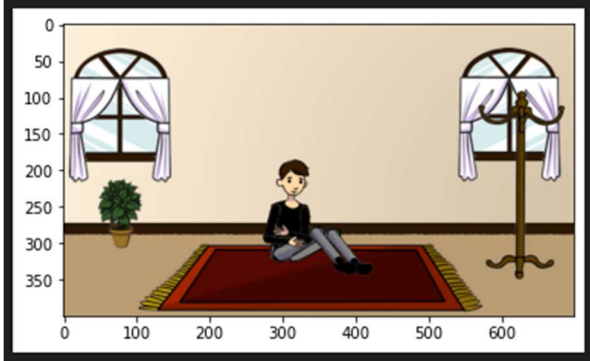

```
1/1 [==============================] - 0s 178ms/step
1/1 [==============================] - 0s 26ms/step
Answer         Prob
     1 9.562705e-01
     3 3.937946e-02
     2 4.349930e-03
     0 7.106563e-08
```

```
How many teapots do you see?
1
```




```
1/1 [==============================] - 0s 169ms/step
1/1 [==============================] - 0s 23ms/step
Answer     Prob
     1 0.976820
     3 0.019751
     2 0.003344
     0 0.000085
```

```
How many windows?
2
```

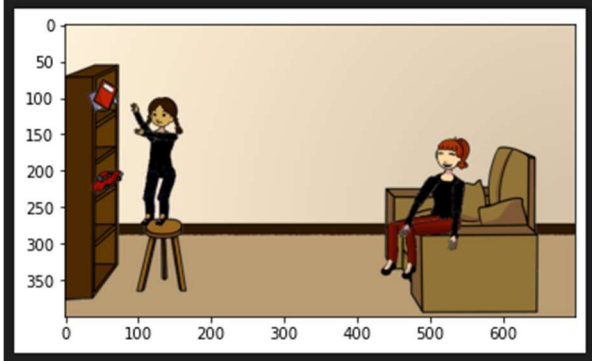



```
1/1 [==============================] - 0s 190ms/step
1/1 [==============================] - 0s 28ms/step
Answer         Prob
     2 5.444509e-01
     1 4.555492e-01
     3 1.012024e-08
     0 2.819592e-14
```

```
How many coats are on the coat rack?
0
```

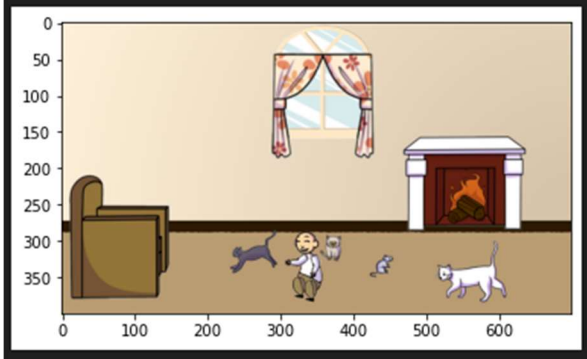

```
1/1 [==============================] - 0s 179ms/step
1/1 [==============================] - 0s 23ms/step
Answer         Prob
     0 1.000000e+00
     3 1.131984e-11
     1 4.659395e-14
    13 1.123400e-15
```

```
How many eyes does the girl on the couch have?
2
```



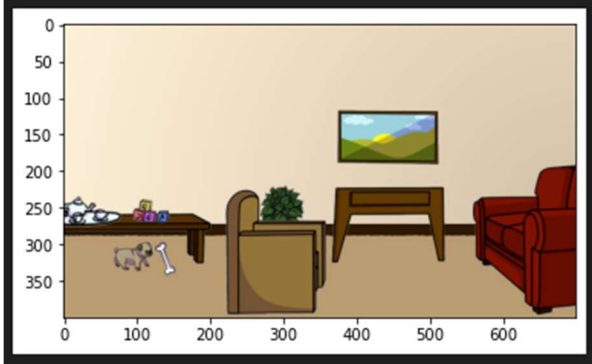

```
1/1 [==============================] - 0s 175ms/step
1/1 [==============================] - 0s 26ms/step
Answer         Prob
     2 9.998621e-01
     1 1.379513e-04
     3 1.745592e-08
   100 2.664812e-09
```

```
How many exits?
1
```



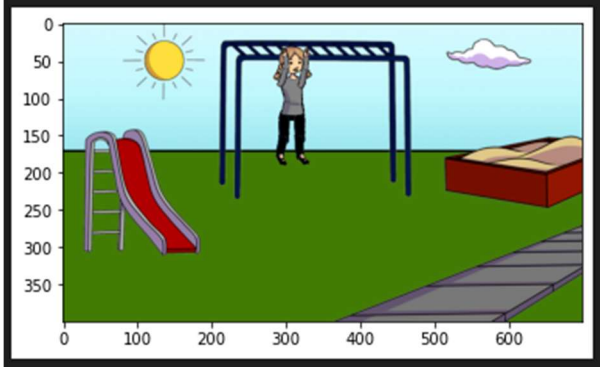
```
1/1 [==============================] - 0s 173ms/step
1/1 [==============================] - 0s 20ms/step
Answer         Prob
     1 9.995425e-01
     2 4.156693e-04
     3 4.188014e-05
     5 1.209849e-10
```

```
How many bones does the dog have?
1
```



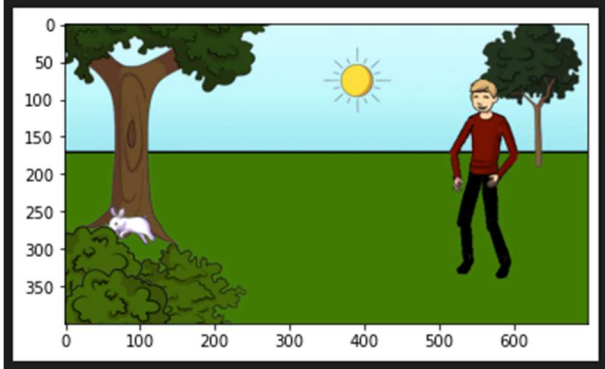
```
1/1 [==============================] - 2s 2s/step
1/1 [==============================] - 0s 110ms/step
Answer     Prob
     2 0.982852
     1 0.015410
     3 0.001710
     0 0.000025
```

```
How many slides?
1
```



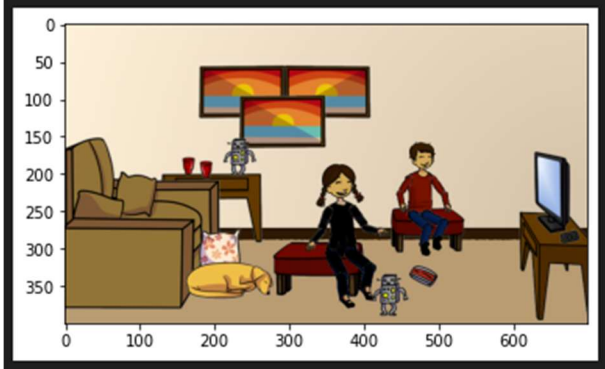
```
1/1 [==============================] - 0s 181ms/step
1/1 [==============================] - 0s 24ms/step
Answer         Prob
     1 9.851781e-01
     2 1.478294e-02
     3 3.899850e-05
     0 4.244011e-11
```

```
How many rabbits?
1
```



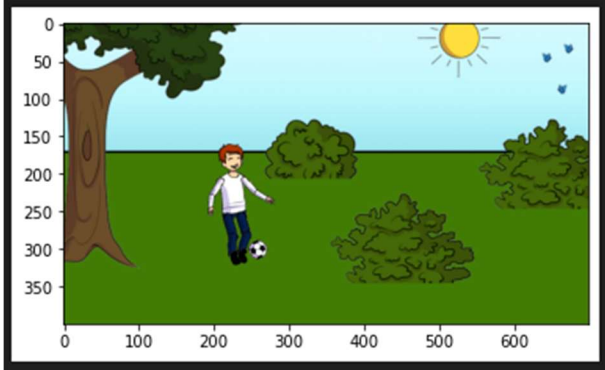
```
1/1 [==============================] - 0s 181ms/step
1/1 [==============================] - 0s 25ms/step
Answer         Prob
     1 9.999120e-01
     3 8.677954e-05
     2 1.045724e-06
     0 1.010709e-07
```

```
How many robots are there?
2
```



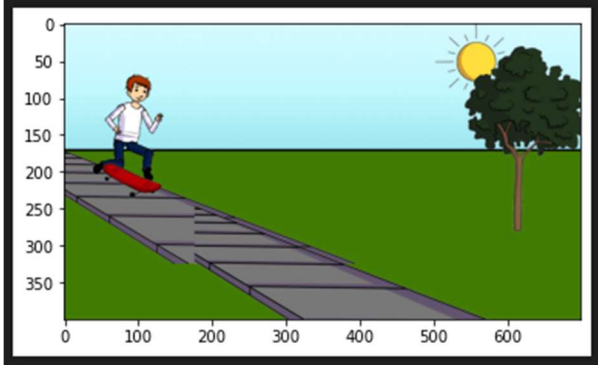

```
1/1 [==============================] - 0s 254ms/step
1/1 [==============================] - 0s 27ms/step
Answer     Prob
     1 0.829879
     2 0.169929
     3 0.000190
     0 0.000002
```

```
How many different types of bushes are there?
2
```




```
1/1 [==============================] - 0s 172ms/step
1/1 [==============================] - 0s 46ms/step
Answer         Prob
     3 9.999830e-01
     2 9.711832e-06
     1 7.404073e-06
     0 8.999800e-12
```

```
How many people is in this picture?
1
```

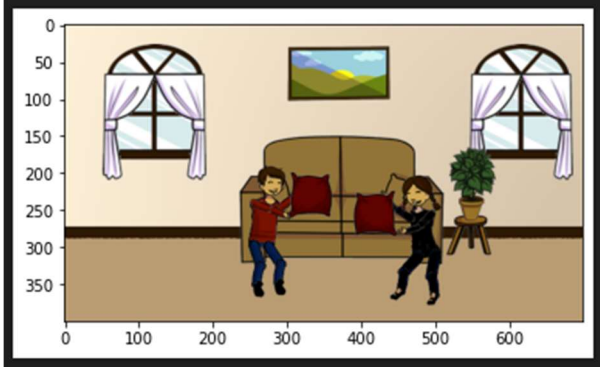



```
1/1 [==============================] - 0s 166ms/step
1/1 [==============================] - 0s 23ms/step
Answer     Prob
     1 0.471106
     2 0.282831
     3 0.196852
     5 0.049125
```

```
How many pets are on the couch?
3
```

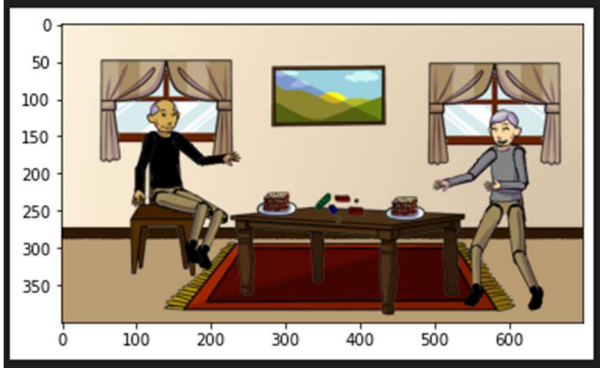

```
1/1 [==============================] - 0s 182ms/step
1/1 [==============================] - 0s 26ms/step
Answer     Prob
     3 0.979979
     2 0.013641
     1 0.006339
     5 0.000036
```

```
How many adults could comfortably sit on the sofa?
2
```




```
1/1 [==============================] - 0s 202ms/step
1/1 [==============================] - 0s 27ms/step
Answer         Prob
     2 9.510669e-01
     1 3.112703e-02
     3 1.780604e-02
     5 1.627491e-08
```

```
How many paintings are on the wall?
1
```

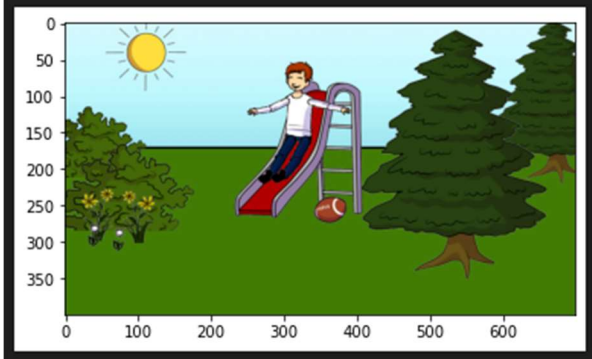



```
1/1 [==============================] - 0s 190ms/step
1/1 [==============================] - 0s 25ms/step
Answer     Prob
     3 0.712297
     2 0.286260
     1 0.001436
     5 0.000005
```

```
How many toys are on the shelf?
4
```

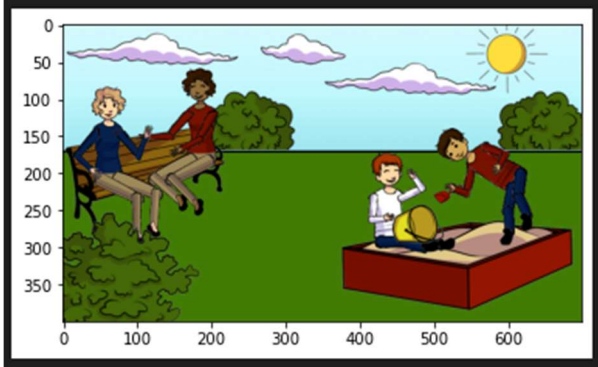

```
1/1 [==============================] - 0s 205ms/step
1/1 [==============================] - 0s 26ms/step
Answer     Prob
     3 0.372444
     2 0.361244
     1 0.261845
     5 0.004292
```

```
How many flowers?
6
```



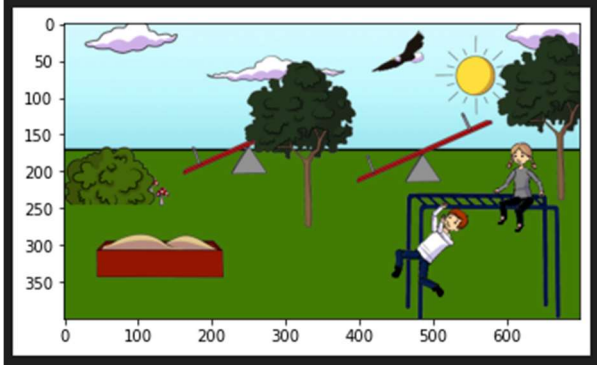

```
1/1 [==============================] - 0s 192ms/step
1/1 [==============================] - 0s 26ms/step
Answer     Prob
     5 0.973845
     4 0.017313
     0 0.002491
     1 0.001767
```

```
How many clouds?
3
```



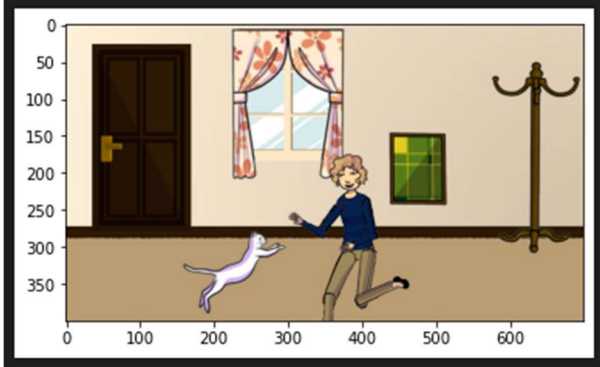

```
1/1 [==============================] - 0s 180ms/step
1/1 [==============================] - 0s 23ms/step
Answer         Prob
     2 6.945001e-01
     3 3.034182e-01
     1 2.081588e-03
     0 2.991119e-08
```

```
How many mushrooms can be seen?
2
```



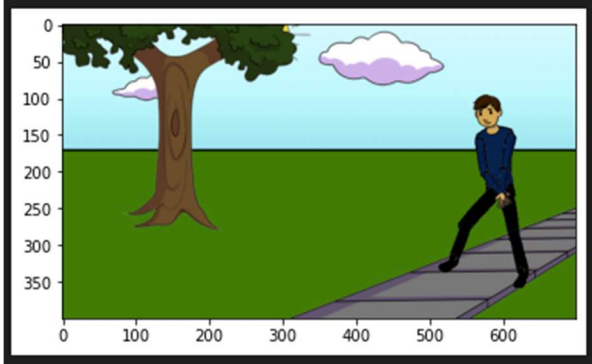

```
1/1 [==============================] - 0s 188ms/step
1/1 [==============================] - 0s 23ms/step
Answer     Prob
     3 0.525351
     4 0.360247
     5 0.087796
     2 0.022750
```

```
How many curtains in the picture?
2
```



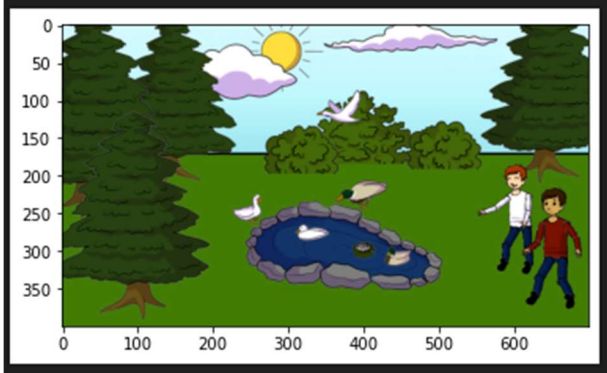

```
1/1 [==============================] - 0s 199ms/step
1/1 [==============================] - 0s 24ms/step
Answer     Prob
     2 0.973981
     1 0.024342
     3 0.001667
    10 0.000010
```

```
How many trees?
1
```



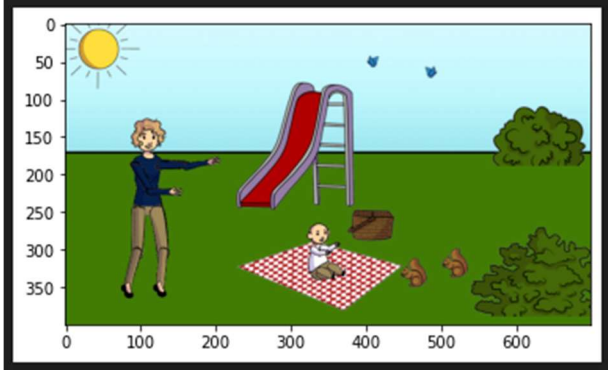

```
1/1 [==============================] - 0s 200ms/step
1/1 [==============================] - 0s 24ms/step
Answer         Prob
     1 9.999988e-01
     2 1.228862e-06
     3 1.861995e-09
     0 1.039742e-10
```

```
How many ducks are flying?
1
```



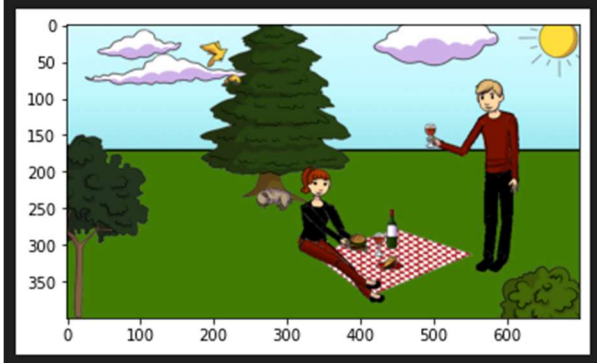
```
1/1 [==============================] - 0s 207ms/step
1/1 [==============================] - 0s 31ms/step
Answer     Prob
     1 0.972742
     2 0.015257
     3 0.007916
     4 0.003245
```

```
How many squirrels are there?
2
```



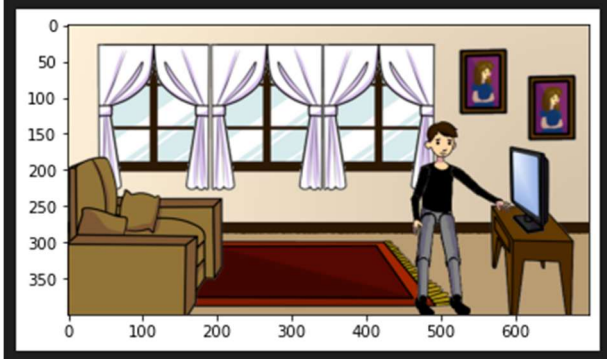
```
1/1 [==============================] - 0s 219ms/step
1/1 [==============================] - 0s 33ms/step
Answer     Prob
     2 0.755087
     1 0.224433
     3 0.017047
    12 0.003087
```

```
How many bottles on the blanket?
1
```




```
1/1 [==============================] - 0s 230ms/step
1/1 [==============================] - 0s 32ms/step
Answer     Prob
     1 0.999203
     3 0.000790
     4 0.000004
     2 0.000003
```

```
How many paintings?
2
```

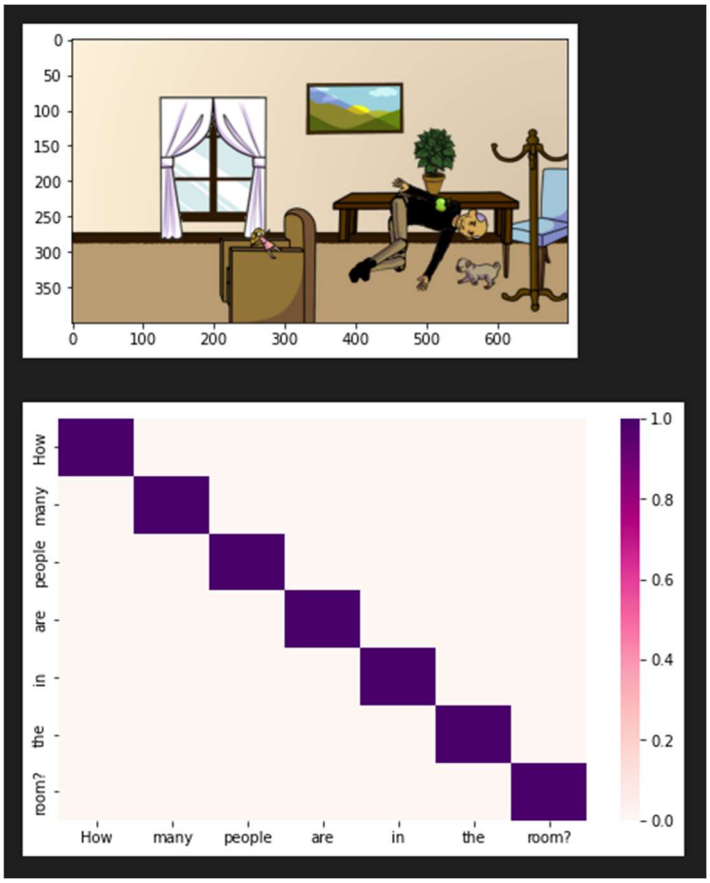



```
1/1 [==============================] - 0s 245ms/step
1/1 [==============================] - 0s 29ms/step
Answer     Prob
     2 0.914900
     1 0.080678
     3 0.004138
     4 0.000260
```

For the above 30 samples from the validation data set, we received the correct output for 22 samples. Therefore the accuracy of the validation data set is (22/30)*100 = 73.33%.

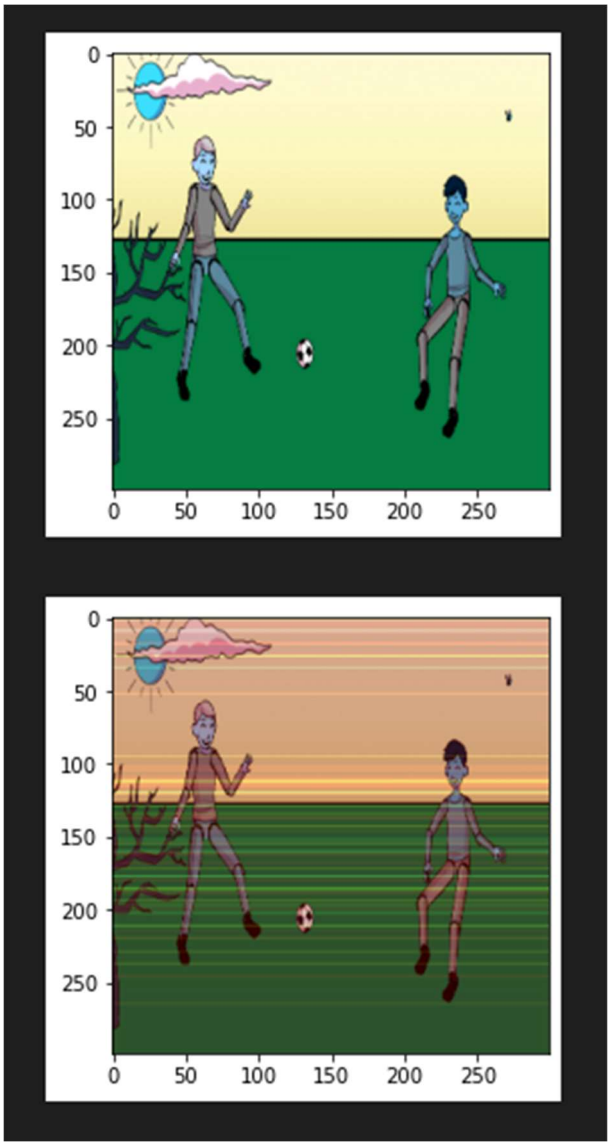
- **The attention map between answer and question words in the previous step indicates correlations among words in the question related to The answer. An example is shown in your homework, you only need to plot each word against the whole image instead of regions.**

We used a novel model for VQA based on the data, one that combines inferential attention with semantic space mapping. A joint embedding of a question and the corresponding image is mapped and clustered around the answer exemplar, which has two key features: a semantic space shared by both labeled and unlabeled answers is built to learn new answers, and a novel inferential attention model is created to simulate the learning process of human attention to explore the correlations between the image and question. It concentrates on the question's key words and relevant areas of the images. Analyses on two open VQA datasets.



- **the attention map of input image means a heatmap that can highlight key regions related to the answer.**

A heatmap of an image shows the areas of the input image that are most important for making predictions. The reasoning behind a VQA model may be understood and potential areas for development can be found using image heatmaps as follows,


0
50
100
150
200
250
0
50
100
150
200
250
0
50
100
150
200
250
0
50
100
150
200
250