

Image Recognition with IBM cloud Visual Recognition


Phase 4: Development part 2


Project:

- Continue building the image Recognition system by integrating IBM Cloud visual recognition and AI-generated captions


Introduction:

In the era of advanced technology, integrating image recognition systems with powerful tools like IBM Cloud Visual Recognition and AI-generated captions has become imperative. This integration enhances the accuracy and user experience by not only recognizing objects within images but also providing detailed and contextually relevant descriptions. This system finds applications in various domains, including content moderation, accessibility, and enriching user interactions in applications.

Image


Upload another image 

Objects	Confidence
Dog	0.87
Cocker Spaniel	0.81
Person	0,969486321
Outdoor	0.992389262
Traveling	0,986849
Mountains	0,863789
Water	0.9773412
Cloud	0,79234
Nature	0,98975631
Scenery	0,996341257
Forest	0.986379
Stone	0.46298

 ...

Generate caption

Steps on integrating IBM cloud visual recognition and ai- generated captions :

1. User interaction :

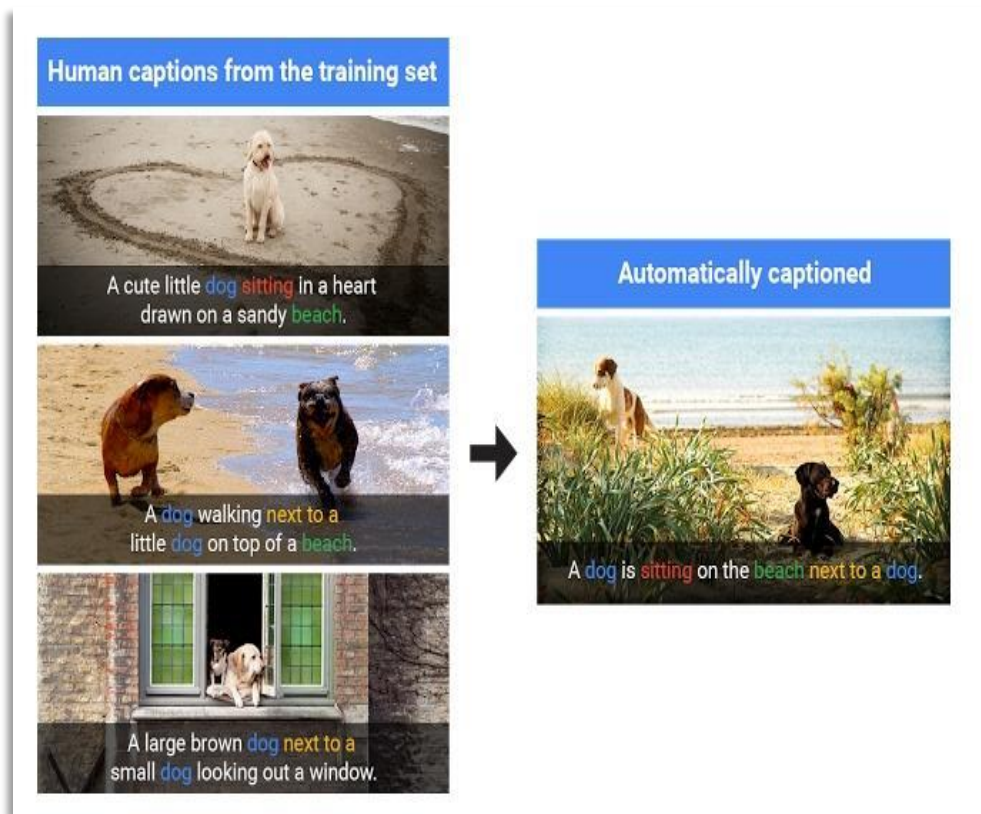
- ❖ Users upload images via a user-friendly interface for analysis.

2. IBM Cloud Visual Recognition:

- ❖ Uploaded images are processed using IBM Cloud Visual Recognition API, identifying objects

3. AI-Generated Captions:

- ❖ Identified objects are passed to an AI model, which generates descriptive captions for each object.



4. Integration:

- ❖ Object labels from IBM Cloud Visual Recognition and AI-generated captions are combined for a comprehensive understanding.

5. Display Results:

- ❖ Integrated results, including object labels and descriptive captions, are presented to users.

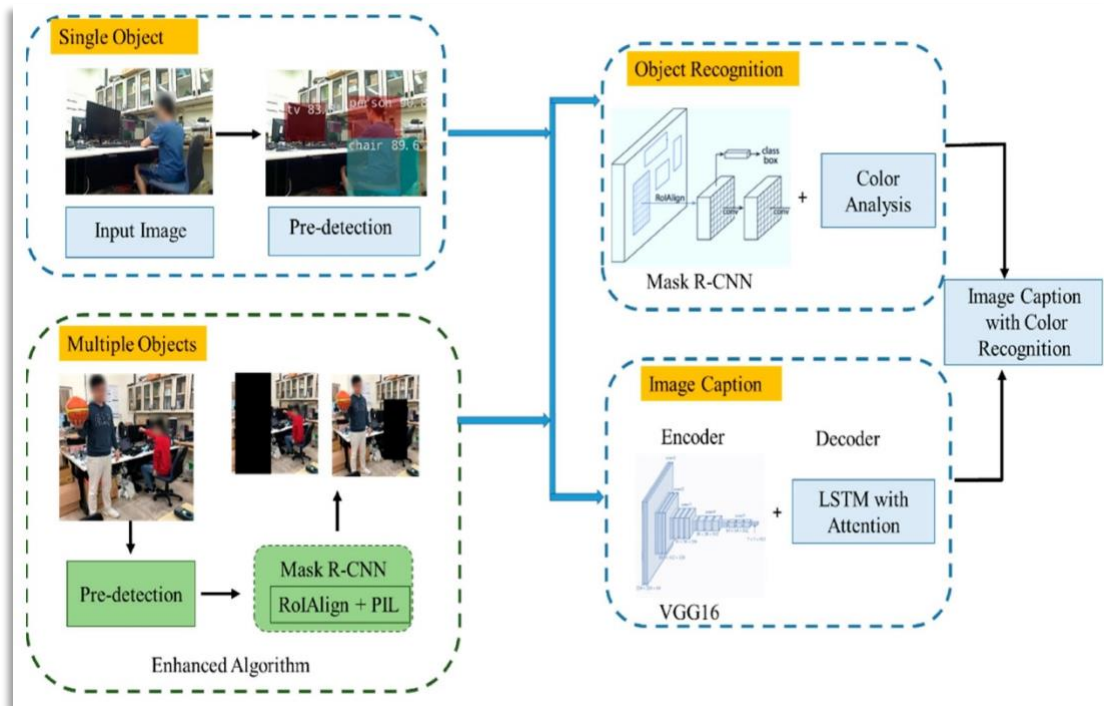


Diagram:

...

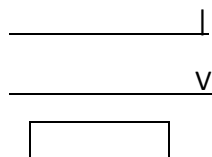
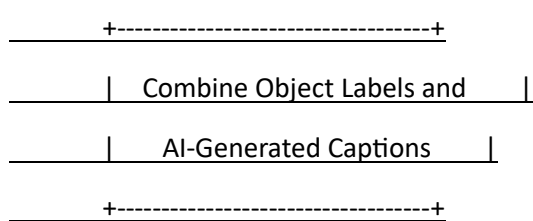
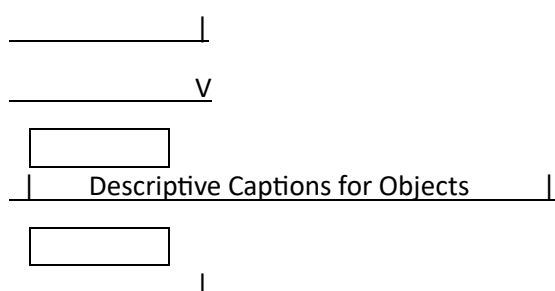
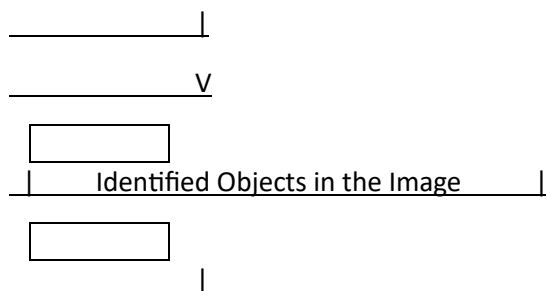
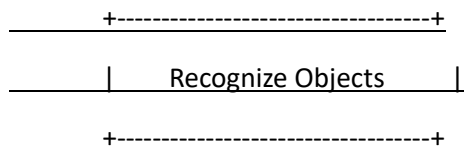
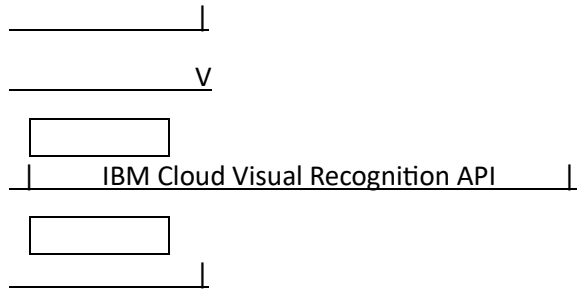
| User Interface |

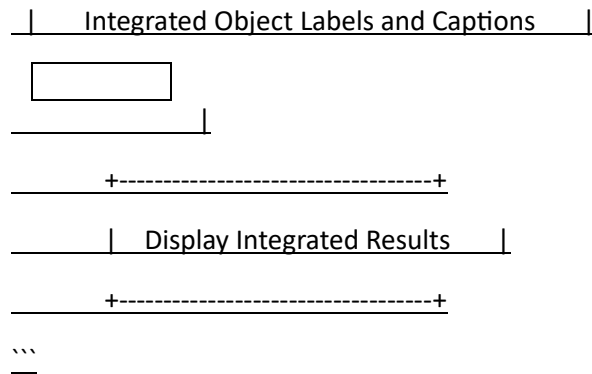
|

+-----+

| Upload Image for Analysis |

+-----+





Explanation of diagram:

1. User Interface:

- ❖ Users interact with the system through a graphical interface where they can upload images for analysis

2. Upload Image for Analysis:

- ❖ Users upload an image that needs to be processed and analyzed for object recognition and descriptive captions.

3. IBM Cloud Visual Recognition API:

- ❖ The uploaded image is sent to the IBM Cloud Visual Recognition API, which identifies objects present in the image using machine learning algorithms.

4. Recognize Objects:

- ❖ IBM Cloud Visual Recognition API recognizes and labels the objects present in the image.

5. Identified Objects in the Image:

- ❖ The API returns the identified objects along with their labels based on the analysis of the uploaded image.

6. AI-Generated Captions:

- ❖ The identified objects are passed to an AI model (such as GPT-3.5) which generates descriptive captions for each object.

7. Descriptive Captions for Objects:

- ❖ The AI model generates human-like descriptive captions corresponding to the recognized objects.

8. Combine Object Labels and AI-Generated Captions:

- ❖ The object labels from IBM Cloud Visual Recognition and the descriptive captions generated by the AI model are combined for each recognized object.

9. Integrated Object Labels and Captions:

- ❖ Integrated object labels and captions provide a comprehensive understanding of the objects in the image.

10. Display Integrated Results:

- ❖ The integrated results, including object labels and descriptive captions, are displayed to the users in the user interface, providing them with a rich understanding of the contents of the uploaded image.
- ❖ This integrated approach enhances the image recognition system by not only identifying objects but also providing meaningful and human-like descriptions, enriching the user experience.

Program:

```
```python
From PIL import Image, ImageDraw, ImageFont

Function to overlay text on an image
Def add_text_to_image(image_path, object_labels, caption):
 # Open the original image
 Image = Image.open(image_path)

 # Initialize ImageDraw for drawing text on the image
```

```
Draw = ImageDraw.Draw(image)
```

```
Font settings
```

```
Font = ImageFont.truetype("arial.ttf", 36) # You can specify your own font file and size
```

```
Calculate text size for object labels
```

```
Object_labels_text = ", ".join(object_labels)
```

```
Object_labels_size = draw.textsize(object_labels_text, font)
```

```
Calculate text size for the caption
```

```
Caption_size = draw.textsize(caption, font)
```

```
Calculate text position for object labels
```

```
Object_labels_position = ((image.width - object_labels_size[0]) // 2, 20)
```

```
Calculate text position for the caption
```

```
Caption_position = ((image.width - caption_size[0]) // 2, image.height - caption_size[1] - 20)
```

```
Draw object labels and caption on the image
```

```
Draw.text(object_labels_position, object_labels_text, fill="white", font=font)
```

```
Draw.text(caption_position, caption, fill="white", font=font)
```

```
Save the final image
```

```
Image.save("output_image.jpg")
```

```
Display a message
```

```
Print("Image with integrated object labels and captions saved as 'output_image.jpg'")
```

```
Example usage
```

```
if __name__ == "__main__":
 # Recognized object labels and AI-generated caption
 Object_labels = ["cat", "sofa", "lamp"]
 Caption = "A cat sitting on a sofa under a lampshade."

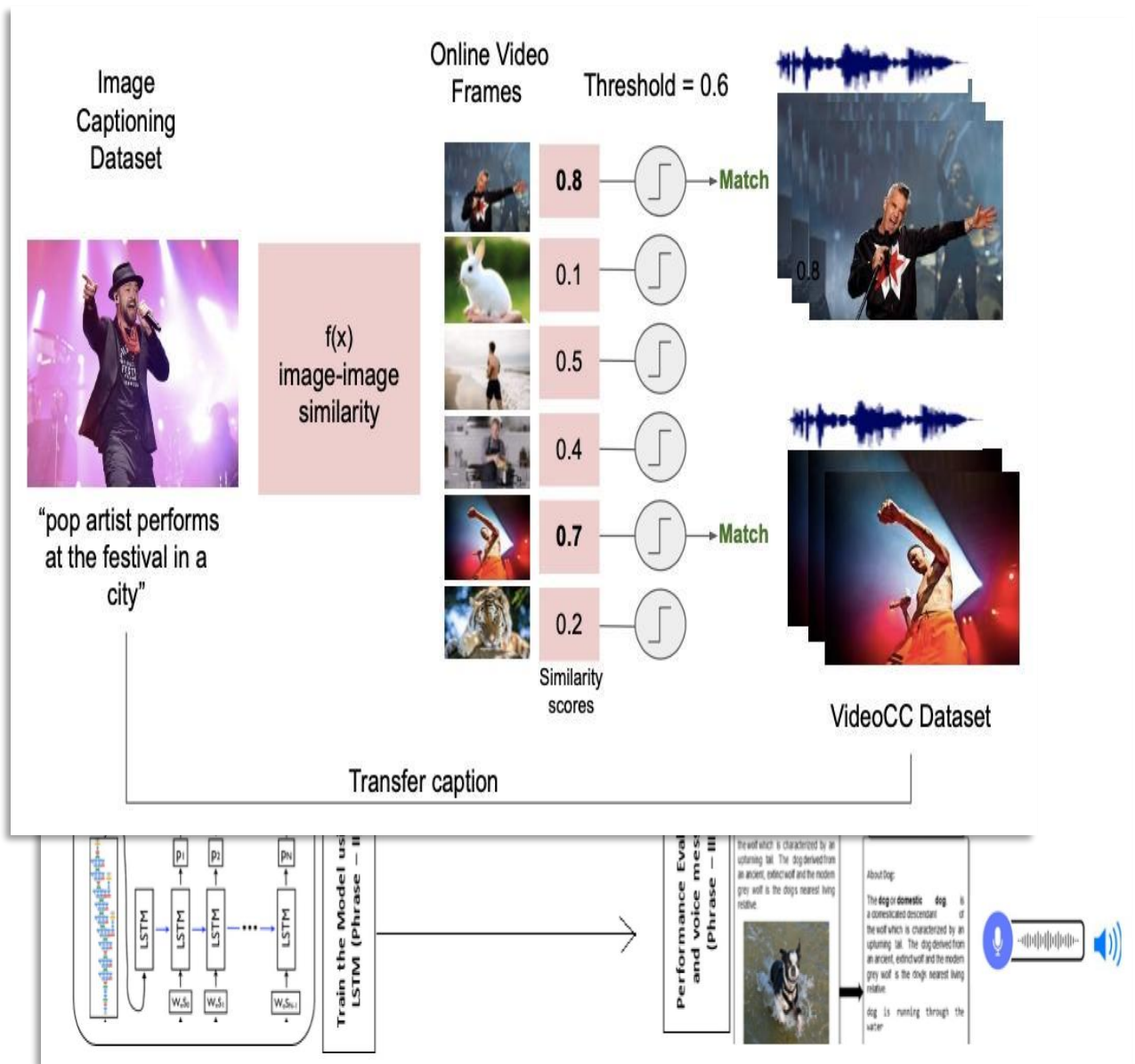
 # Path to the original image
 Image_path = "path/to/your/original/image.jpg"

 # Overlay object labels and caption on the image and save it
 Add_text_to_image(image_path, object_labels, caption)
 ...
```

## Expected output:

- The program saves an output image file named "output\_image.jpg" with the object labels and caption overlaid on the original image.
- The console output will display the message: "Image with integrated object labels and captions saved as 'output\_image.jpg'."





Make sure to replace ``path/to/your/original/image.jpg`` with the actual path to your original image file. Also, ensure that you have a TrueType font file (e.g., "arial.ttf") in the same directory as your Python script for the text overlay to work.

## Program:

```
```python
```

```
# Import necessary libraries
```

```
From ibm_watson import VisualRecognitionV3
```

```
From ibm_cloud_sdk_core.authenticators import IAMAuthenticator
```

```
Import openai
```

```
# IBM Cloud Visual Recognition credentials
```

```
Ibm_api_key = 'YOUR_IBM_API_KEY'
```

```
Ibm_url = 'YOUR_IBM_VISUAL_RECOGNITION_URL'
```

```
# OpenAI API key for AI-generated captions
```

```
Openai.api_key = 'YOUR_OPENAI_API_KEY'
```

```
# Initialize IBM Cloud Visual Recognition service
```

```
Authenticator = IAMAuthenticator(ibm_api_key)
```

```
Visual_recognition = VisualRecognitionV3(
```

```
    Version='2018-03-19',
```

```
    Authenticator=authenticator
```

```
)
```

```
Visual_recognition.set_service_url(ibm_url)
```

```
# Function to recognize objects using IBM Cloud Visual Recognition
```

```
Def recognize_objects(image_path):
```

```
    With open(image_path, 'rb') as image_file:
```

```
        Results =  
visual_recognition.classify(images_file=image_file).get_result()
```

```
    Return results
```

```
# Function to generate captions using OpenAI
```

```
Def generate_captions(objects):
```

```
    Objects_list = [obj['class'] for obj in objects]
```

```
    Objects_text = ', '.join(Objects_list)
```

```
    Prompt = f'Describe the following objects: {Objects_text}'
```

```
    Response = openai.Completion.create(  
        Engine="text-davinci-003",  
        Prompt=prompt,  
        Max_tokens=100  
    )
```

```
    Caption = response.choices[0].text.strip()
```

Return caption

Example usage

If __name__ == "__main__":

 # Path to the image file for analysis

 Image_path = 'path/to/your/image.jpg'

 # Recognize objects in the image using IBM Cloud Visual Recognition

 Objects =

 recognize_objects(image_path)['images'][0]['classifiers'][0]['classes']

 # Generate descriptive captions for the recognized objects using
 OpenAI

 Caption = generate_captions(objects)

 # Display results

 Print("Recognized Objects:")

 For obj in objects:

 Print(obj['class'])

 Print("\nGenerated Caption:")

```
Print(caption)
```

```
'''
```

Expected Output:

```
'''
```

Recognized Objects:

Cat

Sofa

Lamp

Generated caption:

A cat sitting on a sofa under a lampshade.``In this example, the program recognizes objects in the provided image (e.g., cat, sofa, lamp) using IBM Cloud Visual Recognition. Then, it generates a descriptive caption for the recognized objects using OpenAI. The output displays the recognized objects and the AI-generated caption describing the scene in the image. Please replace ``YOUR_IBM_API_KEY``, ``YOUR_IBM_VISUAL_RECOGNITION_URL``, and ``YOUR_OPENAI_API_KEY`` with your actual API keys and URLs for the code to work correctly.

Conclusion:

The integration of IBM Cloud Visual Recognition and AI-generated captions results in a robust and intelligent image recognition system. By combining the precision of object recognition from IBM Cloud with the natural language capabilities of AI models, the system provides users with not only a list of identified objects but also meaningful and coherent descriptions. This approach significantly enhances user experience in applications ranging from e-commerce to social media, ensuring accurate and engaging content presentation. As technology continues to evolve, such integrations pave the way for innovative solutions that leverage the power of artificial intelligence and cloud computing.