

EX 7 IMAGE CAPTIONING USING CNN-RNN ARCHITECTURE**DATE:****Problem Statement:**

Build a deep Recurrent Neural Network (RNN) to generate captions for images. Combine a Convolutional Neural Network (CNN) for feature extraction with an RNN for sequence generation.

Objectives:

1. Understand image captioning using encoder-decoder architecture.
2. Use a pre-trained Vision Transformer (ViT) as the CNN encoder and GPT-2 as the RNN decoder.
3. Generate meaningful captions for visual content using beam search decoding.
4. Explore the power of vision-language models for real-world applications.

Scope:

This experiment demonstrates the fusion of computer vision and natural language processing through encoder-decoder architectures. Students gain insight into how visual features are mapped to textual sequences, a key component in modern AI systems such as image tagging, accessibility tech, and content generation.

Tools and Libraries Used:

1. Python 3.x
2. PyTorch
3. HuggingFace Transformers
4. VisionEncoderDecoderModel (ViT + GPT-2)
5. PIL (Python Imaging Library)

Implementation Steps:**Step 1: Import Required Libraries**

```
import torch
from transformers import VisionEncoderDecoderModel,
ViTImageProcessor, AutoTokenizer from PIL import Image
from transformers.utils import hub

# Set extended download timeout
hub.HUGGINGFACE_HUB_HTTP_TIMEOUT = 60
```

Step 2: Load Pretrained Image Captioning Model

```

model = VisionEncoderDecoderModel.from_pretrained("nlpconnect/vit-gpt2-
imagecaptioning")
processor = ViTImageProcessor.from_pretrained("nlpconnect/vit-gpt2-
imagecaptioning")
tokenizer = AutoTokenizer.from_pretrained("nlpconnect/vit-gpt2-image-
captioning")

device = torch.device("cuda" if torch.cuda.is_available() else "cpu")
model.to(device)
model.eval()

```

Step 3: Define Caption Generation Function

```

def generate_caption(image_path, max_length=30,
num_beams=4):
    image =
Image.open(image_path).convert("RGB")
    pixel_values = processor(images=image,
return_tensors="pt").pixel_values.to(device)

    output_ids = model.generate(pixel_values,
max_length=max_length,
num_beams=num_beams,
early_stopping=True)

    caption = tokenizer.decode(output_ids[0],
skip_special_tokens=True)
    return caption

```

Step 4: Load Image and Generate Caption

```

if __name__ == "__main__":
    img_path = "download.jpeg" # Replace with any image file
    print("Caption:", generate_caption(img_path))

```

Conclusion:

This experiment showcases how deep learning models can describe visual scenes with human-like captions. By combining ViT for image encoding and GPT-2 for text generation, we observe effective cross-modal learning. Pre-trained models significantly reduce development time and enable quick prototyping.