EXPERIMENT - 6

Topic: Image Caption Generator

1. PDF Document Analysis

The provided sources extensively discuss advancements in **image caption generation**, a critical interdisciplinary task merging Computer Vision (CV) and Natural Language Processing (NLP). The goal is to automatically create meaningful and precise textual descriptions for visual content, with diverse applications ranging from social chatbots and aiding the visually impaired to content indexing, search engines, and medical report generation.

Traditional and Early Approaches: Historically, image captioning relied on **Convolutional Neural Networks (CNNs)** for extracting visual features and **Recurrent Neural Networks (RNNs)**, particularly Long Short-Term Memory (LSTM) units, for generating sentences based on these features. Early models like the one pioneered by Vinyals et al. used CNNs pre-trained on ImageNet for encoding and LSTMs for decoding. To address limitations where image features diminished in longer sentences, **attention mechanisms** were introduced, allowing the decoder to focus on relevant image regions when generating words.

The Rise of Transformer Models: Motivated by the success of **Transformer architectures** in NLP, researchers adopted them for image captioning due to their computational efficiency, parallelisation, and scalability. This led to their use as decoders, initially still relying on CNNs for encoding. More recently, **Vision Transformers (ViT)** emerged, adapting the original Transformer architecture directly for image-related challenges and demonstrating significant success. ViTs divide images into fixed-size patches and use self-attention to identify long-range correlations among them, capturing global context across the entire image.

Key Models and Architectures Discussed:

- **Memory-Enhanced Transformer (Afroze et al.)**:

  This model proposes using a **Vision Transformer** for visual feature extraction, replacing traditional CNNs. It is coupled with a **memory-enhanced transformer** for caption generation that leverages both low and high-level image features and incorporates prior knowledge through memory vectors. The decoder features **meshed cross-attention**, which uniquely utilises every intermediate activation from the encoder, rather than just the last layer's outputs. This approach **outperforms conventional "CNN-LSTM" and "CNN-Transformer" methods** on the Flickr30k and MSCOCO datasets, showing improvements in metrics like METEOR and SPICE.

- **Bangla Image Caption Generator (Sarker et al.)**:

This work focuses on **Bangla image captioning**, a challenging task due to the language's complex features. It proposes a novel approach combining a **Vision Transformer (ViT) as a feature extractor** with a **customised encoder-decoder architecture** for Bangla language generation. The model was trained on the augmented BNATURE dataset, which includes 8,000 images with five Bengali descriptions each, and an additional 500 custom images for variety. The ViT-based model **outperforms other feature extractors** such as Xception, ResNet101, ResNet50, and InceptionV3, achieving state-of-the-art performance for Bangla image captioning with BLEU, ROUGE-L, and METEOR scores of 0.6572, 0.6218, and 0.4513 respectively.

- **Enhanced Dense Image Captioning (Goswami et al.)**:

  This research explores **narrative generation** by fusing **GRiT (Generative Region-to-text Transformer)** for dense image captioning with **GPT (Generative Pre-trained Transformer)** for story generation. GRiT extracts detailed object descriptions and their localisations from images, while GPT constructs cohesive storylines based on these descriptions. The architecture uses a ViT as the backbone for the visual encoder, a foreground object extractor, and a 6-layer Transformer as the text decoder. The integration aims to bridge visual perception and textual expression, creating contextually rich content. The model is evaluated on MS COCO and Visual Genome datasets.

- **CNN and Transformers for Image Caption Generation (Babu et al.)**:

  This project integrates **EfficientNetB0 (a CNN)** as an encoder for extracting high-level visual features with a **Transformer encoder-decoder model** for generating word-by-word captions. It focuses on producing context-aware, semantically rich, and syntactically correct descriptions. The model, trained on the Flickr8k dataset, achieved an **accuracy of 86.21%** and a loss of 2.08% over 150 epochs. Its lightweight architecture is noted for computational efficiency.

- **Vision Transformer and GPT-2 Architecture (Mishra et al.)**:

  This work introduces a novel model employing a **Vision Transformer (ViT) as the encoder** and **Generative Pretrained Transformer 2 (GPT-2) as the decoder** within a Seq2Seq framework. The goal is to enhance caption quality, especially for complex visuals. GPT-2 is fine-tuned using an attention-based Seq2Seq approach. The model achieved high performance on the Flickr8k dataset with a **BLEU-4 score of 39.76 and a METEOR score of 52.30**, and similar strong results on Flickr30k.

- **Image Captioning with Vision Transformer Encoder Decoder Model (Kareemunissa et al.)**:

  This paper highlights deep learning models, particularly a **CNN (ResNet) as an encoder** and an **RNN (LSTM) as a decoder**, for image captioning. It states that the **ResNet-LSTM model exhibits superior accuracy** compared to CNN-RNN and VGG models. The methodology includes feature extraction by a CNN, followed by processing these features through a decoder to generate descriptive captions. Both Transformer and LSTM decoders can use Teacher Forcing during training and beam

search for caption generation. Applications for such models include autonomous cars and assistive technologies for the visually impaired.

- **Medical Image Captioning (Sanjay et al.)**:

  This research focuses on the specialised domain of **medical image captioning for chest X-rays**, which requires detailed information and an understanding of relationships between objects and clinical findings. It proposes a **concept detection pipeline** using **Vision Transformers (ViT), Graph Convolutional Networks (GCN), and Long Short-Term Memory Networks (LSTM)**. ViT segments images into patches, GCN models spatial relationships, and LSTM generates captions with a specialised medical vocabulary. A separate **caption generation pipeline uses the GPT-2 model**. Fine-tuning GPT-2 on the ROCO-V2 dataset showed improved specificity and accuracy in medical terminology, for example, correcting "alveoli" to "alveolar" and "spine" to "vertebral".

- **Integrated Transformer Models in CLIP for Digital Media Art (Gao et al.)**:

  This study introduces a **New Multi-modal Fusion Attention module (NMFA)** designed to reduce parameter size and computational complexity by over 50% while maintaining performance comparable to traditional fusion mechanisms. It also proposes the **Transformer Fusion CLIP (TFC) model**, a lightweight architecture that utilises the **CLIP (ViT-B/16 version) model** for encoding images and text. The TFC model eliminates the need for object detectors and high-resolution inputs, simplifying preprocessing. A significant contribution is an **enhanced beam search algorithm** that addresses the bias towards shorter captions by incorporating a reward mechanism for longer sequences. This approach **speeded up runtime by eight times and reduced model parameters by over 50%** on the MSCOCO dataset, demonstrating that the TFC model delivers competitive performance with fewer resources. The improved beam search generates longer, richer captions with more nuanced details.

Common Datasets and Evaluation Metrics: Models are frequently trained and evaluated on widely used datasets such as **Flickr8k, Flickr30k, and MSCOCO**. Specialised datasets like **BNATURE** for Bangla captioning, **Visual Genome** for dense captioning, and the **Indiana University Chest X-Ray Collection (IU X-ray dataset) and ROCO-V2** for medical image captioning are also employed. Performance is quantitatively assessed using a range of metrics, including **BLEU, ROUGE-L, METEOR, CIDEr, and SPICE**. Mean Average Precision (mAP) is used for object detection tasks, and human evaluation is also conducted to assess qualitative aspects like grammatical accuracy, sufficiency, readability, and logical coherence.

Future Directions: Future research aims to further enhance model efficiency, handle more complex and intricate images, develop hybrid models combining CNNs and Transformers, expand to cross-lingual captioning, and apply these models to other multimodal domains such as video captioning, visual question answering, recommendation systems, and image-to-3D conversion. There's also a focus on optimising hyperparameters and improving dataset diversity.

2. YouTube Video Analysis

"Captioning Images with a Transformer, from Scratch! PyTorch Deep Learning Tutorial" by Luke Ditria

This video demonstrates how to build an **image captioning model using Transformers** from scratch in PyTorch. The core idea is to **combine a vision encoder** (from a Vision Transformer) **with a text decoder** (from a sequence-to-sequence text model) to generate text conditioned on an image.

Key Aspects:

- **Objective**: To take an image and describe its contents using a Transformer-based model.
- **Dataset**: The **Coco Captions 2014 dataset** is used, including train/validation images and annotations. PyTorch provides a dataset class for convenience. Each image has five captions, but for training, one caption is randomly sampled per image. Random cropping and caption selection help the model learn associations.
- **Tokenization**: The video introduces **Hugging Face's pre-built tokenizers**, specifically from **DistilBERT**, which has over 30,000 tokens in its vocabulary. It handles padding and returns PyTorch tensors, including input_IDs (token indices) and attention_mask (for padding). DistilBERT's classification and separation tokens are repurposed as **start-of-sentence (SOS)** and **end-of-sentence (EOS)** tokens for captioning.
- **Regularization**: **Token dropping** is implemented, replacing certain tokens with a "blank" or "masking" token (an unused token from DistilBERT's vocabulary). This is crucial to prevent overfitting, as the model is large and sequences are relatively short.
- **Model Architecture**:
    - It employs an **encoder-decoder architecture**, where the encoder is a vision encoder and the decoder is a text decoder.
    - Images are **patched** and treated as embeddings, while text uses sinusoidal positional embeddings.
    - The video introduces **PyTorch's pre-built TransformerEncoderLayer and TransformerEncoder** for the vision encoder, and **TransformerDecoderLayer and TransformerDecoder** for the text decoder, simplifying model construction.
    - The decoder uses **self-attention with causal masking** and **cross-attention** to integrate information from the image encoder's output.
    - The model has approximately **18 million parameters**.

- **Training**:
    - Uses the **Adam optimizer** and **cross-entropy loss** (excluding padding tokens).
    - **Mixed precision** training is applied.
    - Training involves passing images, a shifted target sequence, and padding masks to the model.
    - The model was trained for **200 epochs**, taking 5-6 hours, with token dropping significantly preventing overfitting.
- **Testing and Results**: Captions are generated by prompting the decoder with the SOS token. Examples demonstrate the model generating descriptions for images, though some inaccuracies were noted, potentially due to image size. The video emphasises the ease of combining vision and text within the Transformer architecture for multimodal applications.

"Image Caption Generator using Flickr Dataset | Deep Learning | Python" by Hackers Realm

This video presents a deep learning project to build an **image caption generator using the Flickr 8k dataset**. It leverages a **hybrid CNN-LSTM architecture** implemented with Keras and TensorFlow.

Key Aspects:

- **Objective**: To build a model that can extract features from an image and generate a descriptive caption.
- **Tech Stack**: **Keras with TensorFlow**, requiring a GPU for efficient processing.
- **Dataset**: The **Flickr 8k dataset** is used.
- **Modules and Tools**: Key modules include VGG16 for image feature extraction, Tokenizer and pad_sequence for text processing, LSTM and Dense layers for model architecture, and corpus_bleu from NLTK for evaluation.
- **Data Pre-processing - Image**:
    - **VGG16** is used as a pre-trained **Convolutional Neural Network (CNN)** for feature extraction.
    - The VGG16 model is restructured to **remove its final prediction layer**, retaining only the feature extraction layers.
    - Images are loaded, resized to 224x224, converted to NumPy arrays, and pre-processed according to VGG16 requirements.
    - Extracted features are stored in a **pickle file** (features.pickle) to avoid re-computation.
- **Data Pre-processing - Text (Captions)**:

    - Captions are loaded and mapped to their corresponding image IDs. Each image can have multiple captions.
    - A clean function is applied to captions to **convert them to lowercase, remove digits and special characters, delete extra spaces, and remove single-character words**.
    - **"start" and "end" tags are added to each caption**, which are critical for the model to know when to begin and stop generating text. An initial issue with special characters in tags was rectified for better performance.
    - A **Tokenizer** is fitted on all cleaned captions to create a vocabulary (around 8483 unique words) and determine the max_length of captions (35 words) for padding.
- **Train-Test Split**: The dataset is split into 90% for training and 10% for testing.
- **Data Generator**: A **data_generator function** is implemented to feed data in batches, preventing memory overload, especially on systems with limited RAM. This generator handles splitting captions into input (X, sequence up to a point) and output (Y, the next word), padding input sequences, and one-hot encoding output words.
- **Model Creation**: The model concatenates two distinct feature pathways:
    - **Image Feature Pathway**: Input for VGG16 features (4096 dimensions), followed by Dropout and a Dense layer with ReLU activation.
    - **Text Feature Pathway**: Input for text sequences (max length), an Embedding layer, Dropout, and an LSTM layer.
    - These pathways are combined using an **Add layer**, followed by Dense layers and a final Dense output layer with **Softmax activation** to predict the next word from the vocabulary.
    - The model is compiled with categorical_crossentropy loss and the Adam optimizer.
- **Training**:
    - The model was trained for **15, then 20 epochs**, with a batch size of 64.
    - Training uses the data_generator. The model can be saved as best_model.h5.
- **Prediction and Evaluation**:
    - A predict_caption function is used to generate captions iteratively, starting with the "start" tag and stopping at "end" or max_length.
    - The **BLEU score** is used as the evaluation metric. After rectifying issues with start/end tags and increasing epochs, the model achieved significantly improved scores (e.g., **BLEU-1: 0.54, BLEU-2: 0.31** after 20 epochs), indicating good performance in NLP tasks.

- The video also demonstrates visualizing the results by plotting the image and displaying both actual and predicted captions.

3. Summarizing Dataset in different LLM's

- NotebookLM Can't summarize dataset and can't give any information about the Dataset
- While other LLM's Like Chatgpt, Gemini AI, etc. can summarize the dataset and give information about the dataset

- Lagisetty Bhupala Vignesh - 2022BCSE07AED283