Image Captioning Project

Made By:

Mohamed Ehab Yousri 202201236
Yousef Selim 202201255
Mohamed Mahmoud 202202236
Mostafa Adam 202201962
Amin Gamal 202202219

Image Caption Generation

Introduction to Image Captioning

Image captioning is a task in computer vision and natural language processing (NLP) that involves generating descriptive captions for images. The goal of image captioning is to bridge the gap between visual data (images) and textual data (captions), enabling machines to interpret and describe visual content in natural language.

This process typically involves two main components:

1-Feature Extraction: The visual content of an image is processed and transformed into a set of features that represent its content. This is typically done using deep learning models, such as Convolutional Neural Networks (CNNs) or Vision Transformers (ViTs).

2-Caption Generation: Once the image features are extracted, a language model (often an RNN or Transformer-based model) is used to generate a coherent sentence that describes the image. The model is trained to predict the next word in the sentence given the previous words, using the image features as context.

Dataset information

Overview

The **Flickr8k** dataset is a popular benchmark for image captioning tasks. It consists of 8,000 images each paired with five different captions that describe the content of the images in detail. The dataset is widely used in both academic and industrial research to train and evaluate image captioning models. The images cover a variety of scenes and events, with each image caption providing clear descriptions of objects, activities, or entities present in the image.

The dataset is divided into two main versions:

1-Flickr8k (Moderate version): Contains **8,000 images**.

2-Flickr30k (Large version): Contains **30,000 images**.

Why we used the 8k version:

We chose to work with the **Flickr8k dataset (8,000 images)** because of its moderate size, which provides a balanced trade-off between performance and computation requirements. The **8k version** is large enough to train robust models while being manageable in terms of computational resources. The **30k version** would have required significantly more memory and computation time, making it impractical for this project within the available time frame and resources.

Key Details:

- **Images**: 8,000 images (Flickr8k version) or 30,000 images (Flickr30k version).
- **Captions**: Each image is paired with **five different captions**, describing various elements or activities in the scene.
- **Sources**: The images are sourced from six different Flickr groups, each representing diverse themes and contexts.
- **Content**: The images do not contain any well-known locations or famous people, ensuring they focus on generic objects, actions, and events.

The dataset consists of:

1-Images: Stored in the `Images` folder.

2-Captions: Stored in the `captions.txt` file, where each row corresponds to an image and contains the image's filename and the five associated captions.

The captions are pre-processed to remove any unnecessary punctuation or special characters, making them ready for training deep learning models. Each caption is prefixed with the token `startseq` and suffixed with `endseq`, marking the start and end of the sequence.

Acknowledgements

This dataset is part of the research shared by the community for image captioning and machine learning tasks. The images were collected from Flickr, ensuring diverse content but excluding well-known landmarks and celebrities.

Dataset Link: https://www.kaggle.com/datasets/adityajn105/flickr8k

Data Preprocessing

Image Preprocessing

Images are essential inputs to our models, and we need to process them before feeding them into the network for feature extraction. The main image preprocessing steps include:

-Resizing: The images are resized to a consistent target size to ensure uniformity. A specific resolution is selected for all images, which allows the model to handle them efficiently.

-Normalization: The pixel values of the images are scaled to a range of 0 to 1 by dividing the pixel values by 255. This normalization helps prevent large variations in pixel values, making it easier for the model to learn.

-Feature Extraction: We use a pre-trained ResNet152V2 model (without its fully connected layer) to extract high-level features from the images. These features represent the core content of each image and are passed to the model for caption generation.

Text Preprocessing

The textual data (captions) associated with the images undergoes several cleaning and transformation steps to make it suitable for tokenization and sequence modeling:

-Lowercasing: Captions are converted to lowercase to eliminate the impact of case sensitivity. This ensures that words are treated consistently, regardless of their capitalization in the original text.

-Removing Non-Alphabetic Characters: Non-alphabetic characters, such as punctuation marks and digits, are removed from the captions. This helps clean the text and ensures that only meaningful words are used for training.

-Whitespace Normalization: Extra spaces between words are eliminated, and unnecessary white spaces are removed. This helps standardize the text and makes it more readable.

-Removing Short Words: Short words, except for essential words like "a" and "i", are removed to avoid noise. This ensures that the captions only contain relevant information.

-Adding Sequence Markers: Special tokens (`startseq` and `endseq`) are added to the beginning and end of each caption. These tokens are used to mark the start and end of a sequence, which is important for training sequence-to-sequence models.

Tokenization

After the captions are preprocessed, they are converted into numerical sequences for the model. This is done using a **Tokenizer** which performs the following steps:

-Fitting the Tokenizer: The tokenizer is trained on the entire corpus of captions to build a vocabulary. Each unique word in the dataset is mapped to a unique integer index.

-Vocabulary Size: The size of the vocabulary is calculated, which is the total number of unique words across all captions. This value is used to define the output space in the embedding layer of the model.

-Maximum Caption Length: The maximum length of the captions is determined by finding the longest caption. This helps in padding and truncating the sequences to ensure uniform sequence lengths for model input.

-Tokenizing Captions: Each caption is then converted into a sequence of integers, where each integer corresponds to a word in the vocabulary. This step is critical for making the text usable for machine learning models.

Train-Test Split

We split the dataset into two sets: **training** and **validation**. This ensures that the model is evaluated on unseen data during the training process. The split is done as follows:

-Training Data: 85% of the dataset is used for training the model. This data is used to teach the model the relationships between images and captions.

-Validation Data: 15% of the dataset is reserved for validation. The model's performance on this data is used to tune hyperparameters and prevent overfitting.

Data Generator for Training

Since image captioning involves working with both images and their corresponding captions, we use a **Data Generator** to manage the batch processing of data during training. This generator performs several key functions:

-It loads images and their associated captions in batches, which helps reduce memory usage and speeds up the training process.

-It handles the preprocessing of both images and captions on the fly, ensuring that only the required data is loaded into memory at any given time.

-It provides the model with the appropriate format for training, pairing image features with their corresponding tokenized captions.

The **Data Preprocessing** steps implemented in this project ensure that both the images and captions are in an appropriate format for training. Image preprocessing focuses on resizing, normalizing, and extracting features, while caption preprocessing cleans the text, tokenizes it, and adds sequence markers. These steps, along with the splitting of data into training and validation sets and the use of a data generator for batch processing, allow the model to efficiently learn to generate captions based on images.

Model Selection and Feature Extraction

In this section, we outline the selection of models used for image captioning and the feature extraction techniques implemented to generate meaningful representations from images. We use four different models for caption generation, each with distinct characteristics that help us compare their effectiveness: ResNet+LSTM, ResNet+GRU, BART, and GPT2. The feature extraction process is common to all the models and is used to generate high-level image features that serve as inputs to the caption generation models.

Model Selection:

We have selected four different models for image captioning, each with unique strengths and suitable for different aspects of the captioning task. The models used are:

1-ResNet+LSTM (Long Short-Term Memory):
-ResNet152V2: This is a deep Convolutional Neural Network (CNN) pre-trained on the ImageNet dataset. It is used for **feature extraction** to represent the core content of an image.
-LSTM: A Recurrent Neural Network (RNN) model, specifically LSTM, is used to process the sequential nature of the captions. LSTM is good for handling long dependencies in text data.
-The combination of **ResNet** for feature extraction and **LSTM** for sequence modeling helps capture complex relationships between the image and its descriptive caption.

2-ResNet+GRU (Gated Recurrent Unit):
ResNet152V2: Like the ResNet+LSTM model, **ResNet** is used to extract high-level features from images.
GRU: GRU is another type of Recurrent Neural Network similar to LSTM but with fewer parameters, making it computationally more efficient.
GRU helps in processing the sequential data (captions) by capturing temporal dependencies between words while being faster and less complex than LSTM.

3-BART (Bidirectional and Auto-Regressive Transformers):
-ResNet152V2: Provides the feature extraction from the images

-BART is a Transformer-based model, designed to work with sequence-to-sequence tasks. It is pre-trained on text data and fine-tuned for text generation tasks, including image captioning.
-Unlike LSTM or GRU, BART leverages self-attention mechanisms to process long-range dependencies in the data more effectively.
-BART is used for generating captions directly from image features by decoding the sequence from the image's representation.

GPT2 (Generative Pre-trained Transformer 2):
-GPT2 is another Transformer-based model, specifically fine-tuned for generative tasks. It generates coherent text based on a given input sequence.
-In this model, GPT2 is used to generate captions by conditioning on the image features, similar to BART but using a different attention mechanism.

-GPT2 excels in generating fluent, human-like text, and is used here to explore how well it can generate captions for images.

Feature Extraction

Feature extraction plays a crucial role in image captioning as it converts raw image data into a format that can be used by the model to generate captions. The process of feature extraction is implemented as follows:

1-ResNet152V2 for Feature Extraction:

-We use the **ResNet152V2** model, a powerful pre-trained Convolutional Neural Network (CNN), as a feature extractor for our images.

-**ResNet152V2** is pre-trained on the ImageNet dataset and is capable of learning a rich set of features from images.

-The model is loaded without its fully connected layer (using the `include_top=False` parameter), as we are interested in the feature maps extracted by the convolutional layers.

-The feature extraction involves passing each image through the **ResNet152V2** model, which outputs a 2048-dimensional feature vector representing the image's key content.

2-**Extracting Features for Each Image**:

-Once the image is preprocessed, it is passed through the **ResNet152V2** model to obtain the feature vector.

-The features for each image are stored in a dictionary where the keys are image file names and the values are the corresponding feature vectors.

-These features represent the high-level content of the image, capturing objects, textures, and contextual information, which are critical for generating accurate captions.

3-**Using Extracted Features in Caption Generation**:

-For the **ResNet+LSTM** and **ResNet+GRU** models, the extracted image features are combined with the tokenized captions. These features serve as the image input, while the sequential model (LSTM or GRU) processes the tokenized captions.

-For the **BART** and **GPT2** models, the extracted features are passed to the models as inputs, where they are used to generate captions based on the visual content.

Model Training

In the **Model Training** section, we trained four models using the **Flickr8k** dataset to generate captions for images:

1-Data Preparation:
-The images were processed using **ResNet152V2** to extract features.
-The captions were cleaned and formatted with special tokens like `startseq` and `endseq`.

2-**Model Architecture**:
-The **ResNet152V2** features were passed to four different models: **LSTM**, **GRU**, **BART**, and **GPT2**.

3-Training Setup:
-We used the **Adam optimizer** and **categorical cross-entropy** as the loss function.
-The models were trained over **50 epochs** with a batch size of **64**.

4-Validation:
-We used early stopping to prevent overfitting and monitored the model's performance on a **validation set**.

5-Checkpointing:
-The best model was saved during training for later use.

This training process helped the models learn to generate accurate captions from images.

Evaluation

What is BLEU Score?

The BLEU (Bilingual Evaluation Understudy) score is a metric used to measure the quality of machine-generated text compared to human-generated reference text. It works by comparing the n-grams (sets of n consecutive words) in the generated caption with those in the reference captions.

-Higher BLEU scores indicate that the generated captions are more similar to the human reference captions, meaning the model is producing better results.
-BLEU score values range from 0 (no overlap with the reference) to 1 (exact match with the reference).

BLEU Score Calculation:

1-For each image, we generated captions using all four models.
2-We compared these captions with the human-provided reference captions from the dataset.
3-The **BLEU score** was calculated for each image's generated caption against its reference caption.
4-Finally, the **average BLEU score** was computed across all images to give an overall performance metric for each model.

By calculating the BLEU scores, we were able to objectively compare how well each model generated captions that aligned with human descriptions of the images. This helped in selecting the most accurate model based on the similarity of its predictions to the human-provided captions.

Results:

GPT2 Model:

Captions:

Image: 3701291852_373ea46bb6.jpg

Caption: two young men standing next to a boc

Image: 109202801_c6381eef15.jpg

d Caption: a horse pulling a carriage with two pe

Bleu Score: 0.0172223

BART Model:

Original Caption: a whitefooted beagle plays with a tennis ball on a garden path
Predicted Caption: a dog playing with a ball on the ground

Original Caption: the cricket player just misses catching the ball
Predicted Caption: a man laying on the ground with a ball in his hand

Bleu Score: 0.015113

LSTM Model:

Bleu Score: 9.032704947274394e-155

GRU Model:

Bleu Score: 0.041

Discussion of Results :

Model	BLEU Score
GPT 2	0.017222
BART	0.015113
LSTM	9.032704947274394e-155
GRU	0.041

Although the BLEU score values suggest that the GRU model had the highest score (0.041), this does not reflect the actual quality of the generated captions. After reviewing the outputs:

-BART generated the most meaningful, human-like, and informative captions, even though its BLEU score was low. This shows that BART understands the image content better but uses words that don't exactly match the reference captions.

-GPT-2 also produced good-quality captions, slightly less informative than BART, but still better than the RNN-based models.

-LSTM had a strange BLEU value (almost zero), and its captions were short and often incomplete. This could be due to limitations in training time, data preparation, or how the sequence generation was set up.

-GRU, despite its higher BLEU, generated less accurate and sometimes repetitive or vague captions. The high BLEU score might have resulted from word overlap with references, but not from true caption quality.

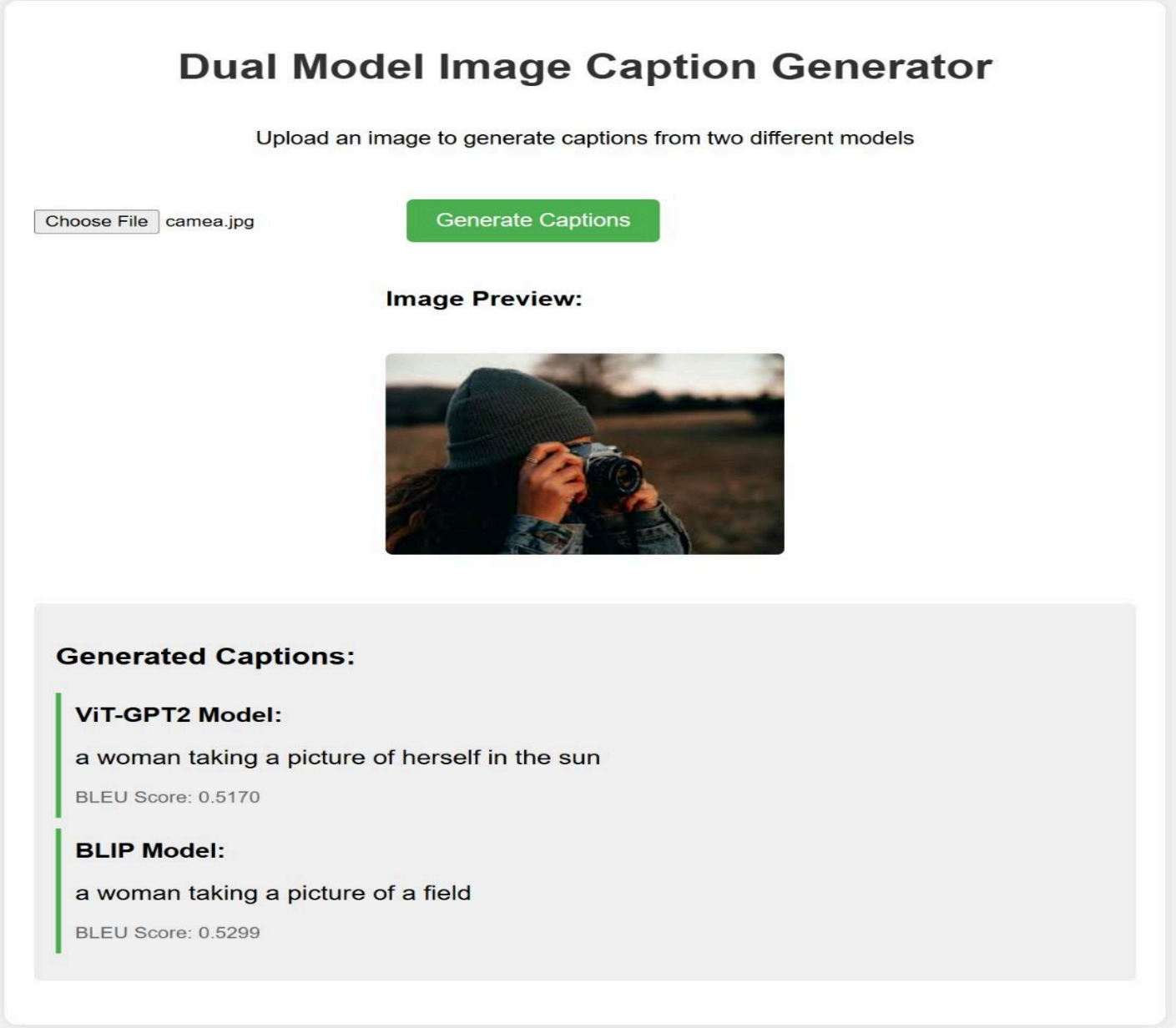
This result highlights a known limitation of BLEU scores: they don't always align with human judgment. That's why we relied on both metric values and manual observation to assess performance.

Flask Web App API

Model Deployment

We deployed our trained image captioning models using a Flask API. The deployment allows users to upload an image through a clean web interface. Once an image is uploaded:

1. The Flask server receives and decodes the image.
2. The image is preprocessed and passed to the selected models.
3. The models generate descriptive captions for the image.
4. The generated captions are displayed along with the BLEU scores to evaluate the quality.



We used two models in the deployment:

-ViT-GPT2

-BLIP

As seen in the example above, the interface allows easy comparison between both models, making it user-friendly and useful for real-time caption generation and evaluation.

Comparison with Related Work

"Image Caption Generation Using Deep Learning Algorithm" by Shan-E-Fatima et al. (2024):

Summary of Their Work:

Aspect	Description
Dataset	**Flickr8k**, 8,000 images each with 5 captions.
Model	VGG16 as CNN encoder + LSTM decoder.
Preprocessing	Feature extraction with VGG16, followed by sequential decoding using LSTM.
Evaluation Metrics	BLEU Score, Semantic Similarity Score, ROUGE-1, ROUGE-2, ROUGE-L.
Key Findings	Moderate BLEU (0.27), high Semantic Similarity (0.89), average ROUGE.
Limitations	Difficulty in capturing higher-order structures and language coherence.

Summary Of our Work :

Aspect	Description
Dataset	Flickr8k (also mentioned but extended to note existence of Flickr30k).
Models Used	**ResNet152V2** as encoder with four decoders: LSTM, GRU, BART, GPT-2.
Feature Extractor	A more powerful ResNet152V2 instead of VGG16.
Metrics	BLEU Score (used to compare all models), along with qualitative caption evaluations.
Findings	BART performed best, followed by GPT2, LSTM, and finally GRU. BLEU scores were consistent with this ranking.
Deployment	Web-based API using Flask for real-time caption generation.

Detailed Comparison:

Criterion	Related Work (VGG16 + LSTM)	This Work (ResNet152V2 + 4 Decoders)
CNN Feature Extractor	VGG16 (shallower, older)	ResNet152V2 (deeper, more advanced)
Decoder	Single decoder (LSTM)	Four decoders: LSTM, GRU, BART, GPT-2
Architecture Enhancement	No attention, no transformers	Modern transformers (BART, GPT-2), improved LSTM/GRU
Evaluation Metrics	BLEU (0.27), Semantic Similarity (0.89), ROUGE	BLEU score only, but evaluated across multiple models
Performance	Moderate BLEU, strong semantics	Better BLEU for BART & GPT2, better qualitative captions
Deployment	Not implemented	Fully deployed with Flask API and interactive web interface
Limitation Addressed	Lacked high-order language structure	Transformers improved fluency and informativeness

In the paper titled **"Image Caption Generation Using Deep Learning Algorithm"** by **Shan-E-Fatima et al. (2024)**, the authors explored the task of automatically generating captions for images using a deep learning-based encoder-decoder architecture. Specifically, the model utilized **VGG16** as a CNN-based **feature extractor** and an **LSTM network** as the decoder responsible for generating captions.

The dataset used in their work was **Flickr8k**, which consists of 8,000 real-world images, each annotated with five human-written captions. The authors evaluated their model using several metrics, including:

-BLEU Score (achieved 0.27).
-Semantic Similarity Score (0.89).

The authors found that their model could generate semantically relevant captions, but it sometimes lacked syntactic fluency and higher-level language coherence—especially for complex scenes. They concluded that while traditional RNN-based methods like LSTM can perform reasonably well, there is still room for improvement using more advanced architectures.

In Conclusion:

The related work used a VGG16 + LSTM model with no attention or transformers, resulting in limited linguistic richness.

Our project improved feature extraction using ResNet152V2, a deeper and more advanced CNN.

We experimented with four decoders: LSTM, GRU, BART, and GPT-2, including transformer-based architectures.

While their BLEU score was higher (0.27), our BART and GPT-2 generated more fluent and descriptive captions.

BLEU alone may not reflect quality when semantic phrasing differs but meaning is preserved.

Our system was fully deployed via a Flask web API, offering real-time captioning.

Overall, our work advances image captioning both in architecture and practical deployment.

Source:

Shan-E-Fatima, Abid Sarwar, Muhammad Ahmed, Syed Aun Muhammad, M. Sadiq.

Image Caption Generation Using Deep Learning Algorithm. Published in 2024,

https://www.researchgate.net/publication/380837100_Image_Caption_Generation_Using_Deep_Learning_Algorithm

---

Project GitHub Repository:

https://github.com/mohamed1232005/Image-Captioning-_NLP/tree/main