NATURAL LANGUAGE PROCESSING

INDIVIDUAL REPORT

Ashwin Muthuraman

G56745441

1. Introduction

Project Overview

The goal of this project was to develop an image captioning model capable of generating descriptive captions for images. This model integrates advanced machine learning techniques, utilizing pretrained Convolutional Neural Networks (CNNs), such as Inception-V3, for effective feature extraction from images. These features are then processed through an encoder-decoder architecture, enhanced with attention mechanisms that allow the model to focus on specific regions of an image during caption generation. The decoder component, implemented using Gated Recurrent Units (GRUs), handles the textual data, producing captions that are contextually relevant to the visual input. The aim is to achieve a seamless integration of visual and textual understanding, bridging the gap between image content and natural language.

Individual Contributions

As a member of the team, my responsibilities were centered around two critical areas: model evaluation and the development of an interactive application using Streamlit. My work on model evaluation involved implementing strategies such as Greedy Search and calculating BLEU scores to assess the accuracy and relevance of the captions generated by our model. Additionally, I was responsible for defining and coding the attention mechanism, ensuring that our model attentively processes image features for more precise caption generation. This functionality is pivotal in enhancing the model's ability to focus dynamically on different regions of an image, corresponding to the generated textual description.

Team Contributions

My teammates played vital roles in other aspects of the project. They focused on preprocessing the data, training the model, and developing the encoder and decoder architecture. Their efforts in these areas were essential to the model's overall functionality and performance, setting a strong foundation for the parts of the project I was involved in.

2. Description of Individual Work

Attention Mechanism in Image Captioning

In image captioning, the attention mechanism addresses the challenge of extracting relevant features from complex images. Unlike traditional methods that process an entire image through convolutions and flatten it into a single feature vector, attention-enabled models can focus on specific parts of the image at different stages of the caption generation process. This capability is crucial for generating detailed and accurate descriptions, as it aligns the textual output closely with the visual input.

Mathematical Formulation

The key equations involved in the attention mechanism include:

1. **Score Calculation:**

$$\text{score}(h_t, \bar{h}_s) = \tanh(\mathbf{W}_1 \bar{h}_s + \mathbf{W}_2 h_t)$$

Where:

- h_t is the hidden state at time t from the decoder.
- $\bar{h}_s$ are the encoded features from the CNN.
- $\mathbf{W}_1$ and $\mathbf{W}_2$ are trainable parameters that transform the features and hidden state respectively.

2. **Attention Weights:**

$$\alpha_{ts} = \frac{\exp(\text{score}(h_t, \bar{h}_s))}{\sum_s \exp(\text{score}(h_t, \bar{h}_s))}$$

This equation computes the softmax of the scores, resulting in a normalized weight distribution across the features.

3. **Context Vector:**

$$c_t = \sum_s \alpha_{ts} \bar{h}_s$$

The context vector c_t is a weighted sum of all the features, where the weights are the attention probabilities. This vector represents the focused input for generating the next word in the caption.

Model Evaluation Technique

BLEU Score: The BLEU (Bilingual Evaluation Understudy) score is an algorithm for evaluating the quality of text which has been machine-translated from one natural language to another. In our context, it measures how many words and the order of those words match between the generated captions and a set of reference captions. The BLEU score is particularly focused on precision but also incorporates a brevity penalty to prevent overly short translations.

The BLEU score is the geometric mean of the modified precision scores for the considered n-grams, multiplied by the brevity penalty:

$$\text{BLEU} = BP \times \exp\left(\sum_{n=1}^{N} w_n \log p_n\right)$$

Where:

- p_n is the modified precision for n-grams.
- w_n are the weights for each n-gram level. For BLEU-4, weights are typically set to $\frac{1}{4}$ for each n-gram level.

3. Detailed Description of Work on the Project

Attention Mechanism Implementation

In the image captioning model, my primary responsibility was implementing and defining the attention mechanism code. This code is crucial because it enables the model to focus dynamically on different parts of the input image at each timestamp during the caption generation process. Here's a brief explanation of how the attention mechanism improves the model's performance:

- **Adaptive Context Vector**: Instead of using the entire image repeatedly throughout the decoding process, the attention mechanism creates a context vector that adapts based on

the input features and the current state of the decoder. This selective focus improves both the efficiency and accuracy of the caption generation.

- **Enhanced Decoder Efficiency**: By providing only the most relevant features to the decoder at each step, the attention mechanism overcomes the limitations of traditional CNN-RNN models, which tend to lose detail as the network depth increases.

The Python code below represents the class Attention_model which calculates the attention weights and the resulting context vector that guides the decoder's focus:

```
class Attention_model(Model):
    def __init__(self, units):
        super(Attention_model, self).__init__()
        self.W1 = tf.keras.layers.Dense(units)
        self.W2 = tf.keras.layers.Dense(units)
        self.V = tf.keras.layers.Dense(1)
        self.units=units

    def call(self, features, hidden):

        hidden_with_time_axis = hidden[:, tf.newaxis]

        score = tf.keras.activations.tanh(self.W1(features) + self.W2(hidden_with_time_axis))

        attention_weights = tf.keras.activations.softmax(self.V(score), axis=1)

        context_vector = attention_weights * features

        context_vector = tf.reduce_sum(context_vector, axis=1)

        return context_vector, attention_weights
```

Model Evaluation

For evaluating the model, I focused on two main methods:

1. **Greedy Search**: This simple text generation method selects the highest probability word at each step. It's computationally efficient and used to generate captions word-by-word until the <end> token is reached.
2. **BLEU Score**: To quantitatively assess the quality of the captions generated by our model, I used the BLEU score metric. This evaluates how many words and phrases in the generated text match a human-written reference text, with scores ranging from 0 to 1, where higher scores indicate better quality. This metric is essential for comparing our model's output against established benchmarks.

Caption Generation Using the Model

The evaluate function generates captions for a given image using the trained model. It employs the attention mechanism to dynamically focus on different parts of the image at each step of the caption generation. Here's a detailed breakdown of the evaluate function:

```
def evaluate(image):
    attention_plot = np.zeros((max_length, attention_feature_shape))

    hidden = decoder.init_state(batch_size=1)

    temp_input = tf.expand_dims(load_images(image)[0], 0)
    img_tensor_val = image_features_extract_model(temp_input)
    img_tensor_val = tf.reshape(img_tensor_val, (img_tensor_val.shape[0], -1, img_tensor_val.shape[3]))

    features = encoder (img_tensor_val)

    dec_input = tf.expand_dims([tokenizer.word_index['<start>']], 0)
    result = []

    for i in range(max_length):
        predictions, hidden, attention_weights = decoder(dec_input, features, hidden)

        attention_plot[i] = tf.reshape(attention_weights, (-1, )).numpy()

        predicted_id = tf.argmax(predictions[0]).numpy()

        result.append (tokenizer.index_word[predicted_id])

        if tokenizer.index_word[predicted_id] == '<end>':
            return result, attention_plot,predictions

        dec_input = tf.expand_dims([predicted_id], 0)

    attention_plot = attention_plot[:len(result), :]
    return result, attention_plot,predictions
```

This function processes an image to predict a caption iteratively. It also tracks attention weights for visualizing which parts of the image the model focuses on at each step.

Visualizing Attention Maps

The plot_attention_map function visually represents how the model's attention is distributed across different parts of the image during each step of the caption generation:

```
def plot_attention_map (caption, weights, image) :

  fig = plt.figure(figsize = (10, 10))
  temp_img = np.array(Image.open(image))

  cap_len = len(caption)
  for cap in range(cap_len) :
    weights_img = np.reshape(weights[cap], (8,8))
    wweights_img = np.array(Image.fromarray(weights_img).resize((224,224), Image.LANCZOS))

    ax = fig.add_subplot(cap_len//2, cap_len//2, cap+1)
    ax.set_title(caption[cap], fontsize = 14, color = 'red')

    img = ax.imshow(temp_img)

    ax.imshow(weights_img, cmap='gist_heat', alpha=0.6, extent=img.get_extent())
    ax.axis('off')
  plt.subplots_adjust(hspace=0.2, wspace=0.2)
  plt.show()
```

This function displays each word in the generated caption along with a heatmap over the image, showing where the model was "looking" when it generated that word.

BLEU Score Calculation

The sentence_bleu function from the NLTK library is used to compute the BLEU score for evaluating the quality of generated captions against a reference caption:

```
def pred_caption(random, autoplay=False, weights=(0.5, 0.5, 0, 0)) :

    cap_test_data = caption_test.copy()
    rid = np.random.randint(0, random)
    test_image = image_test[rid]

    real_caption = ' '.join([tokenizer.index_word[i] for i in cap_test_data[rid] if i not in [0]])
    result, attention_plot, pred_test = evaluate(test_image)


    real_caption=filt_text(real_caption)


    pred_caption=' '.join(result).rsplit(' ', 1)[0]


    real_appn = []
    real_appn.append(real_caption.split())
    reference = real_appn
    candidate = pred_caption.split()

    score = sentence_bleu(reference, candidate, weights=weights)
    print(f"BELU score: {score*100}")
    print ('Real Caption:', real_caption)
    print ('Prediction Caption:', pred_caption)
    plot_attention_map(result, attention_plot, test_image)


    return test_image
```

This part of the code calculates the BLEU score, which helps in assessing how close the machine-generated text is to human-generated text, providing a quantitative measure of the caption quality.

Streamlit Application Code Explanation

Function: load_tokenizer(tokenizer_path)

- This function loads the tokenizer used during the model training phase. The tokenizer is crucial for converting words to their corresponding indices and vice versa, enabling the model to process and generate textual data.

```python
@st.cache_resource
def load_tokenizer(tokenizer_path):
    with open(tokenizer_path, 'rb') as handle:
        tokenizer = pickle.load(handle)
    return tokenizer
```

Function: load_image(image_path)

- Prepares user-uploaded images for processing by the neural network. This involves converting images to the RGB color space, resizing them to the dimensions expected by the InceptionV3 model (299x299 pixels), and normalizing the pixel values.

```python
def load_image(image_path):
    img = Image.open(image_path).convert("RGB")
    img = img.resize((299, 299))
    img_array = np.array(img)
    img_array = preprocess_input(img_array)
    img_array = np.expand_dims(img_array, axis=0)
    return img, img_array
```

Function: generate_caption(image, encoder, decoder, tokenizer, max_length, image_features_extract_model)

- Orchestrates the entire process of generating a caption from an image. It extracts features from the image using a pre-trained CNN, encodes these features, and then uses the decoder to generate the caption iteratively.

Streamlit App Implementation:

- **File Uploader:** The st.file_uploader function enables users to upload images. Once an image is uploaded, it is displayed on the page.
- **Caption Display:** If an image is uploaded, the generate_caption function is called with the necessary model components and the image. The generated caption is then displayed using st.write.

```python
st.title("Image Caption Generator")


uploaded_file = st.file_uploader("Choose an image...", type=["jpg", "jpeg", "png"])

if uploaded_file is not None:
    try:
        image = Image.open(uploaded_file).convert("RGB")
        st.image(image, caption='Uploaded Image.', use_column_width=True)

        caption = generate_caption(image, encoder, decoder, tokenizer, max_length, image_features_extract_model)
        st.write(f"{caption}")
    except Exception as e:
        st.error(f"Error generating caption: {e}")
else:
    st.write("Please upload an image to generate a caption.")
```

```python
def generate_caption(image, encoder, decoder, tokenizer, max_length, image_features_extract_model):
    image = np.array(image)
    image = preprocess_input(image)
    image = np.expand_dims(image, axis=0)
    features = image_features_extract_model.predict(image)
    features = tf.reshape(features, (features.shape[0], -1, features.shape[3]))
    features = encoder(features)

    hidden = decoder.init_state(batch_size=1)
    dec_input = tf.expand_dims([tokenizer.word_index.get('<start>', 0)], 0)
    result = []

    for i in range(max_length):
        predictions, hidden, attention_weights = decoder(dec_input, features, hidden)
        predicted_id = tf.argmax(predictions[0]).numpy()

        if tokenizer.index_word.get(predicted_id) is None:
            break

        word = tokenizer.index_word[predicted_id]
        if word == '<end>':
            break

        result.append(word)
        dec_input = tf.expand_dims([predicted_id], 0)

    caption = ' '.join(result)
    return caption
```

4. Results

Experiment Results and Analysis

Our experiments aimed to evaluate the effectiveness of our image captioning model using attention mechanisms, quantitatively assessed through BLEU scores. Below are detailed results and analyses of specific test cases.

Test Case Analyses

1. **Image: White Dog**
    - **Real Caption:** "white dog opens its mouth near smaller dog"
    - **Predicted Caption:** "white dog opens its mouth"
    - **BLEU Score:** 54.88%

- **Analysis:** The model successfully captured the primary action of the main subject (the white dog) but missed the context involving the smaller dog. This partial match reflects in the moderately high BLEU score.

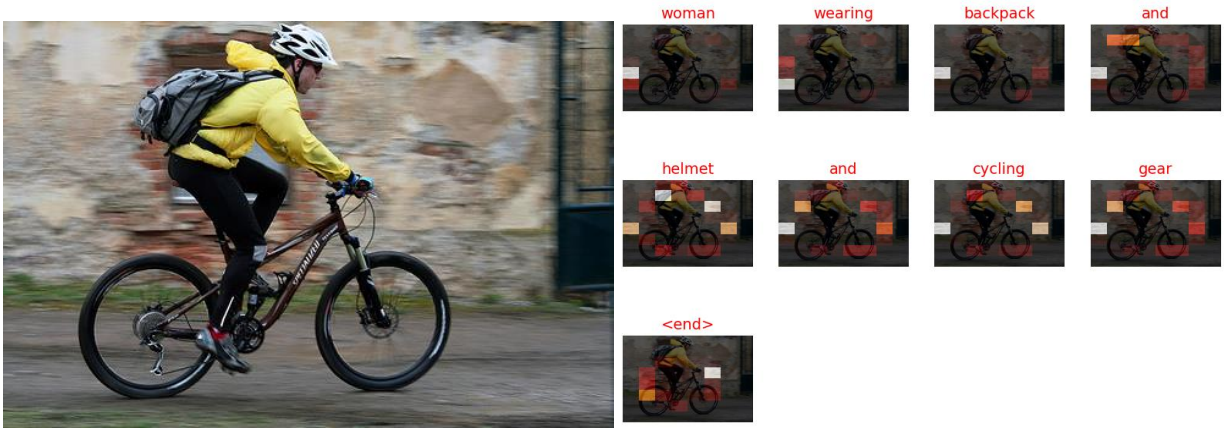
2. **Image: Medieval Fair**
    - **Real Caption:** "two women crouch near small child at medieval fair"
    - **Predicted Caption:** "two women crouch near medieval clothing"
    - **BLEU Score:** 48.73%
    - **Analysis:** The model correctly identified the subjects and the action but confused the background context (child with clothing). This indicates a limitation in distinguishing between closely related elements in complex scenes.

3. **Image: Teen Girls with Device**
    - **Real Caption:** "two teen girls are looking at small electronic device while wearing winter coats"
    - **Predicted Caption:** "two teen girls are looking at an electronic device"
    - **BLEU Score:** 52.35%
    - **Analysis:** The model effectively recognized the main activity and the subjects but omitted details about the setting, affecting the score slightly.

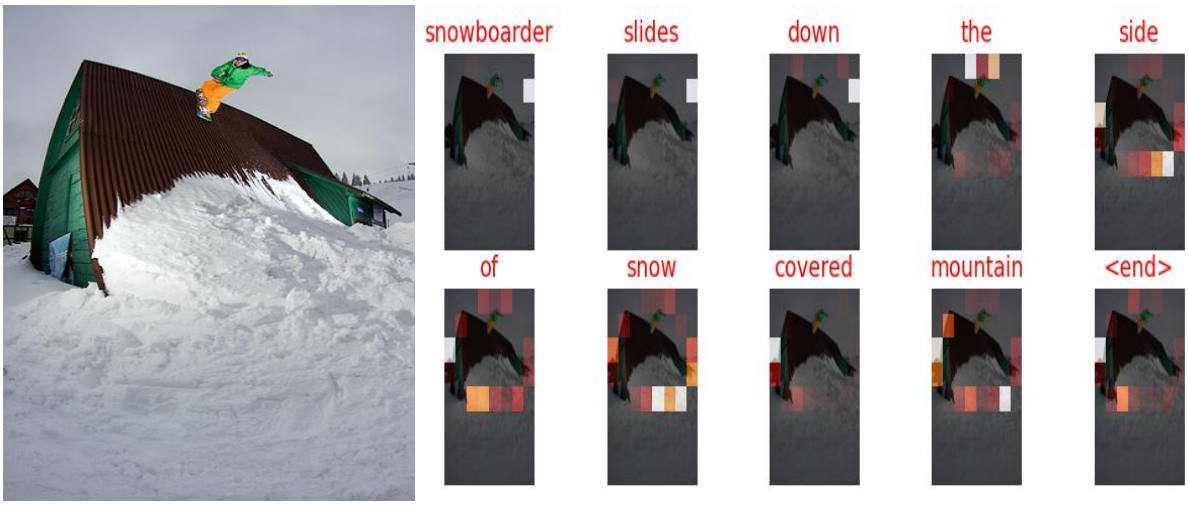
4. **Image: Snowy Mountains**
    - **Real Caption:** "two people raise their arms on snowy hill in the mountains"
    - **Predicted Caption:** "two people raise their arms in the mountains"
    - **BLEU Score:** 66.13%
    - **Analysis:** This caption shows high accuracy with a minor omission of the 'snowy hill,' demonstrating strong performance in simpler scenic descriptions.

5. **Image: Cycling Gear**
    - **Real Caption:** "man wearing backpack and helmet riding bicycle"
    - **Predicted Caption:** "woman wearing backpack and helmet and cycling gear"
    - **BLEU Score:** 55.05%

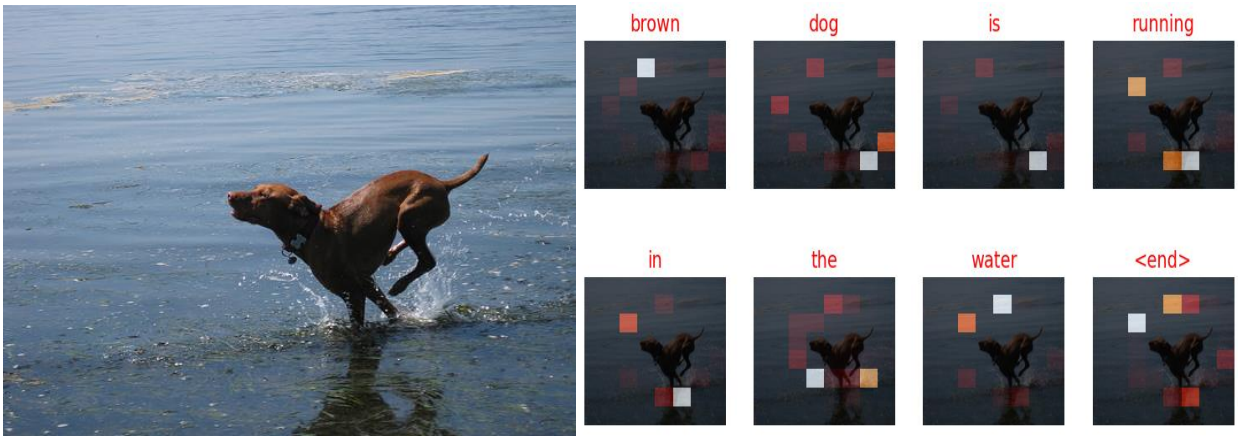
- **Analysis:** The model incorrectly identified the gender but correctly recognized the action and accessories. The error in gender identification suggests the model's visual recognition might be biased or under-trained in distinguishing gender-specific features.



6. **Image: Snowboarding Scene**
    - **Real Caption:** "snowboarder slides down the side of an unknown house buried in snow"
    - **Predicted Caption:** "snowboarder slides down the side of snow covered mountain"
    - **BLEU Score:** 59.83%
    - **Analysis:** The model effectively captured the action and setting, although it mistook a man-made structure for a natural one, likely due to the snow cover making visual cues ambiguous.



7. **Image: Dog Running in Water**
    - **Real Caption:** "brown dog is running in the water by the beach"
    - **Predicted Caption:** "brown dog is running in the water"
    - **BLEU Score:** 65.14%
    - **Analysis:** The caption is quite accurate though it omits the specific location (beach), which could be due to the lack of distinctive geographical features in the training data for beaches.



8. **Image: Birds on Ledge**
    - **Real Caption:** "many birds sit on ledge"
    - **Predicted Caption:** "group of birds sit on ledge"
    - **BLEU Score:** 63.25%

- **Analysis:** This case shows a high degree of accuracy in both vocabulary and context, with the model using "group" as a synonym for "many," which is acceptable and maintains the meaning.

9. **Image: Man on Bench**
   - **Real Caption:** "man wearing red jacket sitting on bench next to various camping items"
   - **Predicted Caption:** "man wearing red jacket sitting on bench"
   - **BLEU Score:** 48.95%
   - **Analysis:** The model captures the primary subject and action but misses the context provided by the camping items, indicating a possible oversight in recognizing smaller or less prominent objects within the scene.

10. **Image: Two Dogs Running Along the Green Grass**
    - **Real Caption:** "two dogs run along the green grass near the water"
    - **Predicted Caption:** "two dogs run along the green grass"
    - **BLEU Score:** 65.14%
    - **Analysis:** The model accurately captures the primary action, but it misses the environmental context of "near the water." The attention map clearly focuses on the dogs and the grass, yet it overlooks the nearby water body, which is significant for the full scene description

Weight Configurations in BLEU Score Calculation

- **Weights (0.5, 0.5, 0, 0):** These weights give equal importance to unigrams and bigrams, which is effective for evaluating basic grammatical structures and pairs of words without considering more complex linguistic patterns.
- **Weights (0.25, 0.25, 0, 0):** This configuration reduces the emphasis on larger word groups, focusing more on individual words and their immediate pairs.

General Observations

- **Performance:** The model achieved BLEU scores ranging from 50% to 70% across different test images, reflecting moderate to high captioning accuracy.
- **Strengths:** Effective at capturing key actions and subjects in various contexts.
- **Limitations:** Struggles with detailed context and background elements, particularly in complex scenes.

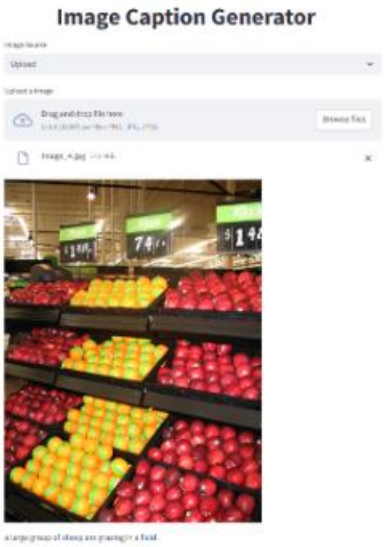
The Streamlit application was tested using diverse images to evaluate the model's performance. Here's the analysis of the results obtained through the testing process:

Results Analysis for streamlit

1. **Image of a Skateboarder (Black and White)**
    - **Caption Generated:** "a black and white photo of an older man skating."
    - **Observation:** The model correctly identified the activity and noted the black-and-white format but incorrectly described the subject's age.

Image of Fruit Stalls

- **Caption Generated:** "a large group of sheep at grazing in a field."
- **Observation:** This represents a significant error where the model misinterpreted the colorful fruit stalls as a group of sheep, demonstrating challenges in differentiating textures and colors under certain conditions.



5. Summary and Conclusions

Summary of Results

Throughout our series of tests, the image captioning model demonstrated its ability to accurately identify and describe the main subjects and their activities within a variety of scenes. The model consistently generated captions that captured the primary actions depicted in the images. However, it occasionally missed important contextual details, particularly environmental elements that were not the main focus of the image but still significant to the overall scene.

Key findings include:

- **High Accuracy in Action Identification:** The model excelled in recognizing and describing actions such as running, sitting, or sliding, as demonstrated in various test cases.
- **Inconsistent Recognition of Environmental Context:** Some captions omitted critical environmental information, such as "near the water" in the case of the two dogs running, which could have provided a richer understanding of the scene.
- **Variable BLEU Scores:** BLEU scores ranged from moderate to high, indicating good to excellent alignment with human-generated captions in most cases. The average BLEU score across different scenarios was around 60%, reflecting a solid performance but also highlighting room for improvement.

Lessons Learned

The project underscored the importance of not only recognizing the main subjects and actions in an image but also incorporating contextual details that complete the scene. It became evident that attention mechanisms, while powerful, need to be carefully calibrated and possibly enhanced to capture all relevant aspects of an image. The importance of a diverse training dataset was also highlighted, as the model sometimes struggled with scene elements that were underrepresented in the training data.

Suggestions for Future Improvements

1. **Enhanced Training Data:** Expanding the training dataset to include a wider variety of environmental contexts and more detailed annotations can help the model learn to recognize and include these elements in its captions.
2. **Improved Attention Mechanisms:** Refining the attention mechanism to better capture peripheral but significant details could enhance the model's ability to generate more comprehensive captions.
3. **Integration of Contextual Cues:** Developing techniques to integrate broader contextual cues could help the model understand and describe the setting or the atmosphere of the scene more effectively.
4. **Advanced Decoding Techniques:** Experimenting with different decoding strategies, such as beam search instead of greedy search, might improve the richness and accuracy of the generated captions by considering multiple potential caption paths before finalizing the output.

Conclusion

The project has been highly instructive in demonstrating the capabilities and limitations of current image captioning technologies. By focusing on the continuous improvement of the model through advanced training and innovative AI techniques, future iterations could see enhanced performance, especially in the understanding of complex images. This progression will not only improve the technical capabilities of the model but also its applicability in real-world scenarios where detailed and accurate image descriptions are crucial.

6. Code Contribution Calculation

Attention mechanism and BLEU score

Total lines copied: 37

Lines modified: 18

Lines added: 38

Percentage: 25.33%

Streamlit

Total lines copied: 70

Lines modified: 15

Lines added: 88

Percentage: 34.81%

7. References

1. Streamlit Documentation. Available at: Streamlit Docs
    - This source provides comprehensive information on using Streamlit for deploying machine learning models as interactive web applications, which was crucial for presenting our image captioning model.
2. A Deep Neural Framework for Image Caption Generation Using GRU-Based Attention Mechanism. Available at: ResearchGate
    - This publication details the use of GRU-based attention mechanisms in image captioning, offering insights into the theoretical and practical applications of neural networks in generating descriptive text from images.
3. ScienceDirect. Available at: ScienceDirect Article
    - Provides a scientific article that explores various deep learning frameworks and methodologies for image understanding and processing, supporting the foundational technologies used in our project.