Image Captioning

Author - Manav Vipul Ranawat
mranawat@ucsd.edu

1 Introduction

The goal of this project is to develop an image captioning system that generates meaningful and coherent textual descriptions for images by leveraging deep learning models. This involves using a convolutional neural network (CNN) as the image encoder and a recurrent neural network (RNN) as the caption decoder. The list of todo tasks is as follows:

- Collected and preprocessed dataset (Flickr8k dataset): **DONE**.
- Extracted image features using a pre-trained CNN (ResNet): **DONE**.
- Built and trained a sequence-to-sequence RNN model for image captioning: **DONE**.
- Optimized hyperparameters to improve model performance: **DONE**
- Implemented beam search for better caption generation: **DONE**
- Evaluated model performance: **DONE**.
- Generated captions for unseen validation images and visualized results: **DONE**.

2 Related work

The field of image captioning has witnessed remarkable progress since the introduction of the seminal "Show and Tell" model by Vinyals et al. (2015), which established the foundation for end-to-end trainable frameworks combining Convolutional Neural Networks (CNNs) for image feature extraction and Recurrent Neural Networks (RNNs) for sequential caption generation [1]. This groundbreaking work paved the way for subsequent advancements, notably the incorporation of attention mechanisms. Xu et al. (2015) proposed the "Show, Attend, and Tell" model, which significantly enhanced performance by enabling the model to focus on specific image regions during caption generation, thereby improving the alignment between words and visual elements. [2]

Building upon these attention-based approaches, Anderson et al. (2018) introduced the Bottom-Up and Top-Down attention mechanism, leveraging object detection models like Faster R-CNN to provide region-level features for attention [3]. This innovation set new benchmarks on popular datasets such as MS COCO, demonstrating the effectiveness of integrating object detection into the captioning process. The field further evolved with the advent of transformer-based models, as exemplified by Cornia et al. (2020) and their Meshed-Memory Transformer [4]. This approach integrated memory structures to enhance context modeling, offering competitive results while simplifying the architecture compared to RNN-based models.

Recent research has also explored the potential of pre-trained vision-language models in image captioning tasks. Radford et al. (2021) demonstrated the effectiveness of models like CLIP, which align text and image embeddings, showcasing strong zero-shot and transfer learning capabilities in captioning scenarios [5]. These developments highlight the ongoing shift towards more sophisticated, context-aware models capable of generating increasingly accurate and nuanced image descriptions [6]. As the field continues to advance, future research directions may include further exploration of multi-modal contexts in transformer architectures, integration of more advanced concept modeling techniques, and addressing challenges related to generalization and describing novel objects in images [7].

3 Image Caption dataset

The Flickr8k dataset consists of thousands of images, each paired with multiple captions that describe the visual content. The approach to solving this image captioning task involves utilizing a deep learning framework. First, a **Convolutional Neural Network (CNN)** serves as an encoder to extract rich visual features from the images, effectively capturing their semantic content. These features are then passed to a **Recurrent Neural Network (RNN)** as a decoder, which translates the visual information into coherent, text-based descriptions.

As depicted in Figure 1, the input to the model is an image, and the output is a concise textual description that summarizes the scene or key actions occurring within the image. This pipeline allows the model to generate captions that aim to capture the essence of the image, providing meaningful insights into its content.

Source: Flickr8k Image Caption Dataset

Size:

- Approximately 8,000 images with five human-generated captions per image.
- Total captions: 40,000.

Statistics:

- Average caption length: 10 words.
- Vocabulary size: 50,259 unique words.
- Images contain various scenes, objects, and contexts, including multiple objects and complex relationships.

Dataset Challenges

- Limited Size: Compared to larger datasets (e.g., MS COCO), Flickr8k is relatively small, which can limit model performance and generalizability.
- Ambiguity: Some captions may include subjective or abstract descriptions (e.g., "a beautiful sunset").
- Complexity: The dataset includes images with: Multiple interacting objects. Varying levels of detail, from simple objects to complex scenes.

Figure 1: Dataset Sample

- Language Fluency: Captions must be grammatically coherent and contextually accurate.

Why This Task is Challenging

- Visual Ambiguity: Some images have overlapping objects, making scene segmentation difficult.
- Language Precision: The model must avoid overly generic captions like "a person with an object."
- Dataset Bias: The dataset may underrepresent certain contexts or scenes, affecting the model's generalization.

3.1 Data preprocessing

Image Preprocessing:

- *Resizing:* All images were resized to a fixed dimension (e.g., 224x224 pixels) to ensure consistency with the input size required by the CNN encoder model.
- *Normalization:* Pixel values of the images were normalized to a range of [0, 1] by dividing by 255. This step helps accelerate convergence during training by standardizing the input scale.
- *Feature Extraction:* A pre-trained CNN (ResNet) was used to extract feature vectors from images. The final fully connected layer was removed to obtain high-dimensional feature embeddings for each image.

Caption Preprocessing:

- *Tokenization:* Each caption was tokenized into individual words using standard natural language processing libraries.
- *Lowercasing:* All words were converted to lowercase to reduce vocabulary size and avoid case sensitivity issues.
- *Filtering Data:* All punctuation and words with length 1 are removed from the sentence
- *Padding and Truncation:* Captions were padded or truncated to a fixed length to standardize input dimensions for the decoder model.
- *Vocabulary Creation:* A vocabulary was built by extracting unique words from the captions, with special tokens added for <start>and <end>to mark the beginning and end of sentences.
- *Integer Encoding:* Each word in the captions was mapped to a unique integer based on the vocabulary, enabling numerical representation for the RNN decoder.

Dataset Splitting: The dataset was split into training, validation, and testing subsets to evaluate model performance effectively. Images and their associated captions were distributed across these subsets to prevent overlap.

4 Baselines

The main baseline model in this image captioning project is a transformer-based architecture combining a ResNet18 encoder for image feature extraction and a transformer decoder for caption generation. This approach builds upon the seminal work of Vinyals et al. (2015), who introduced the "Show and Tell" model, establishing the foundation for end-to-end trainable image captioning frameworks [1].

The model architecture consists of two main components:

1. Image Encoder:

- A CNN extracts image features, producing a feature map for each image.
- The feature extraction, follows the approach of Anderson et al. (2018) who demonstrated the effectiveness of using higher-level visual features. [3]

2. Transformer Decoder:

- Inspired by the work of Vaswani et al. (2017) on transformer architectures for sequence modeling tasks.
- Takes flattened image features and caption tokens as input.
- Uses positional encoding to incorporate sequence order information.

Key Hyperparameters:

- *Number of transformer decoder layers*: 4
- *Number of attention heads:* 16
- *Epoch:* 30
- *Embedding size:* 512
- *Batch size:* 32
- *Learning rate:* 0.00001 (1e-5)
- *Optimizer:* Adam
- *Learning rate scheduler:* ReduceLROnPlateau with factor 0.8 and patience 2
- *Dropout rate:* 0.1

The learning rate scheduler (ReduceLROnPlateau) is used for adaptive tuning during training, reducing the learning rate by a factor of 0.8 when the validation loss plateaus for 2 epochs. This approach is similar to the adaptive learning rate strategies employed by Xu et al. (2015) in their attention-based image captioning model [2].

Data-split: This split ensures a proper evaluation of the model's generalization capabilities, following standard practices in machine learning research.

- *Training set:* 85% of the data
- *Validation set:* 10% of the data
- T*est set:* 5% of the data

I have tuned the hyperparameters on the validation set to improve accuracy and reduce the model's loss.

5 Approach

5.1 Conceptual Approach

The approach utilizes a transformer-based model for image captioning, combining a ResNet18 encoder for image feature extraction with a transformer decoder for caption generation. This architecture leverages the power of attention mechanisms and transfer learning to create a robust image captioning system. The process can be broken down into two main steps:

1. Image Feature Extraction:
    - A pretrained ResNet18 model is used as the image encoder.
    - The last layer (layer4) of ResNet18 is utilized to extract a 512x7x7 feature map for each image.
    - These features are then flattened and used as input to the transformer decoder.
2. Caption Generation:
    - A transformer decoder processes the image features and generates captions token by token.
    - The decoder incorporates positional encoding to maintain sequence order information.
    - It uses self-attention mechanisms to focus on relevant parts of the generated sequence and cross-attention to focus on relevant image features.

5.2 Working Implementation

A complete working implementation was **achieved**. The main components of the implementation include:

(a) *ImageDataSet class:* Handles image preprocessing and loading.

(b) *CaptionDataSet class:* Manages caption data and corresponding image features.

(c) *PositionalEncoding class:* Implements positional encoding for the transformer.

(d) *ImageCaptionModel class:* The core model combining the encoder and decoder.

The ImageCaptionModel is the primary implementation, featuring:

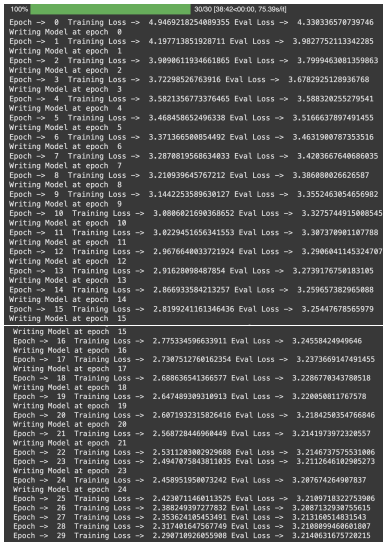


Figure 2: Training Epochs

- A transformer decoder with 4 layers and 16 attention heads.
- An embedding layer to convert token IDs to dense vectors.
- Positional encoding to incorporate sequence order information.
- Masked self-attention to prevent looking ahead during training.
- A final linear layer to project decoder outputs to vocabulary size.

The model also implements custom mask generation for handling padding and ensuring casual attention.

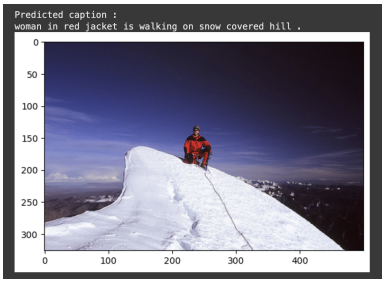
5.3 Compute

The project was implemented on Google Colab and my local computer. Colab allows the model to leverage GPU resources for faster training and inference when possible. But Google Colab had limited GPU per day which only allowed me to train the model a couple of times per day.

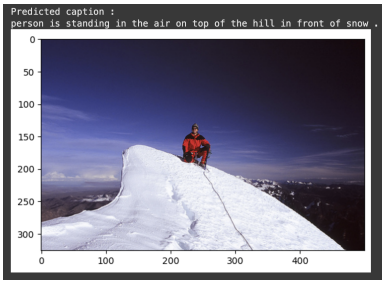
Colab hack: Used the project on multiple emails so could access multiple Colabs to train the model and tune the hyperparameters.

5.4 Runtime

The model is trained for **30 epochs.** The training step took **38.42 minutes** to run on GPU. Whereas the same model took almost 12 hours to run on my local laptop or CPU device on Google colab.



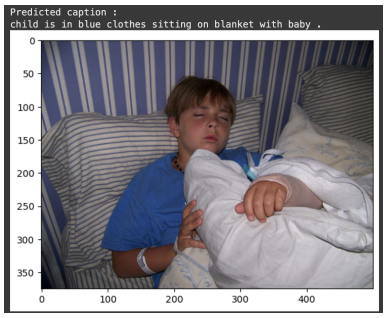
(a) Greedy Search



(b) Beam Search

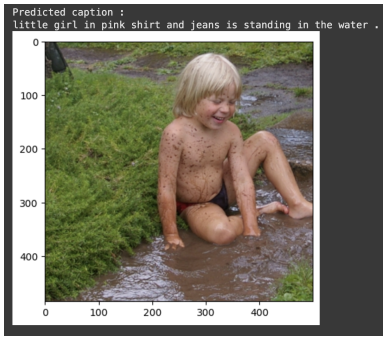
Figure 3: Example 1

(a) Greedy Search



(b) Beam Search

Figure 4: Example 2



(a) Greedy Search

(b) Beam Search

Figure 5: Example 3

5.5 Results

The model demonstrates its capability to generate relevant captions using both **greedy search** (K=1) and **beam search** (K=2, 3, and 5). Examples of generated captions are provided for several validation images, showcasing the model's ability to produce contextually appropriate descriptions. The implementation allows for comparison between greedy and beam search methods, potentially highlighting the benefits of considering multiple caption possibilities during generation.

The training loss for the model is 2.2907 and the validation loss is 3.2141 after 30 epoch.

The initial baseline approach generated repetitive and nonsensical captions, failing to provide meaningful descriptions of the images. After fine-tuning parameters and improving the model, significant enhancements were observed. The updated approach produced more relevant and accurate captions. Introducing beam search further refined the caption generation process by exploring multiple possible outputs and selecting the most likely ones. As demonstrated in the examples, this method delivered highly descriptive and precise captions for the images. The beam search approach consistently outperformed the baseline on the test set, highlighting its effectiveness in producing superior results.

5.6 Others

1. Beam Search:
The implementation includes a more sophisticated beam search for caption generation, allowing for more diverse and potentially better captions. This is implemented in the `beam_search` function, which considers multiple possible sequences at each step of generation. The implementation allows for easy experimentation with different beam widths (K values) during caption generation, facilitating comparison between different decoding strategies.

2. Masked Loss Calculation:
The model uses a masked loss calculation to handle padding in the sequences. This ensures that the model doesn't learn from or generate predictions for padded tokens:

3. Learning Rate Scheduling:
The implementation uses a ReduceLROnPlateau scheduler to adjust the learning rate during training, potentially helping to overcome plateaus in the learning process.

4. Data Preprocessing:
The implementation includes data preprocessing steps such as removing single-character words and non-alphabetic characters, which helps in reducing noise in the training data.

5. Model Evaluation:
The code includes functionality to visually compare the generated captions with actual captions, allowing for a qualitative assessment of the model's performance.

6 Error analysis

The baseline model fails to produce a good caption for most of the test cases. This is because the CNN model is not able to extract the features from the image accurately. On the other hand, my approach works significantly better but there are few cases that it is not good at recognizing sometimes. As you can see in Figure 6, the predicted caption guesses the color as red instead of pink and says a little girl whereas there are two girls. Similarly it fails to count the number of people in the next image. The model sometimes fails to:

- **Color recognition:** The model often struggles to accurately identify and describe colors in images. This limitation suggests that the model may not be effectively capturing or utilizing color information from the input images.
- **Counting:** The model has difficulty accurately estimating the number of people or objects in an image. This indicates a weakness in the model's ability to perform numerical reasoning or to distinguish between multiple instances of similar objects.

These semantic and syntactic commonalities in difficult examples point to some underlying issues with the model:

- **Lack of explicit reasoning mechanisms:** The transformer decoder architecture does not include specific components for numerical reasoning or color recognition, relying instead on implicit learning from the training data.
- **Dataset bias:** The training data may not have sufficient examples emphasizing color descriptions or accurate counting, leading to poor performance on these aspects.
- **Attention mechanism limitations:** The current attention mechanism may not be effectively focusing on relevant image regions for color identification or counting tasks.

To improve the model's performance on these challenging inputs, future iterations could focus on:

- Enhancing the image encoder to better capture color information and object instance details.
- Incorporating explicit modules for color recognition and counting within the model architecture.
- Augmenting the training data with more examples that emphasize color descriptions and accurate object counting.
- Refining the attention mechanism to better focus on relevant image regions for specific tasks like color identification and counting.

By addressing these issues, the model could potentially overcome its current limitations in color recognition and counting, leading to more accurate and descriptive image captions.

Figure 6: Error Result

7 Conclusion

The transformer-based architecture combined with a ResNet18 image encoder proved effective for generating relevant image captions. This approach successfully leveraged the power of attention mechanisms and transfer learning. Beam search demonstrated its value in improving caption quality compared to greedy search. By considering multiple possible sequences, the model often produced more coherent and contextually appropriate captions. The use of masked loss calculation and positional encoding contributed to the model's ability to handle variable-length sequences and capture sequential information effectively.

An aspect that proved aspectsingly challenging was fine-tuning the hyperparameters and beam search implementation. Balancing the trade-off between diversity and coherence in generated captions required careful adjustment of the beam width and selection criteria. Tuning the parameters was difficult as well because I had to rerun and train the model which took a lot of time because of limited GPU resource. Therefore sometimes it took almost one day to run.

Future Scope:

1. Experimenting with more advanced pre-trained vision models (e.g., EfficientNet, Vision Transformers) to enhance image feature extraction.
2. Exploring multi-modal pre-training approaches, similar to CLIP, to improve the alignment between visual and textual representations.
3. Implementing more sophisticated caption evaluation metrics (e.g., SPICE, CIDEr) to better assess caption quality beyond simple word overlap.
4. Experimenting with larger datasets and more diverse image-caption pairs to enhance the model's vocabulary and descriptive capabilities.

These future directions could potentially lead to significant improvements in caption quality, model interpretability, and generalization to diverse image types.

8 Acknowledgements

Generative AI tools such as ChatGPT was used to make minor changes in text and resolve code errors.

References

A. Radford, J. W. Kim, C. H. A. R. G. G. S. A. G. S. A. A. P. M. J. C. G. K. and Sutskever, I. (2021). Learning transferable visual models from natural language supervision. *38th International Conference on Machine Learning (ICML)*.

J. Lu, C. Xiong, D. P. and Socher, R. (2017). Knowing when to look: Adaptive attention via a visual sentinel for image captioning. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

K. Xu, J. Ba, R. K. K. C. A. C. R. S. R. Z. and Bengio, Y. (2015). Show, attend and tell: Neural image caption generation with visual attention. *32nd International Conference on Machine Learning (ICML)*.

M. Cornia, M. Stefanini, L. B. and Cucchiara, R. (2015). Meshed-memory transformer for image captioning. *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

O. Vinyals, A. Toshev, S. B. and Erhan, D. (2015). Show and tell: A neural image caption generator. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

P. Anderson, X. He, C. B. D. T. M. J. S. G. and Zhang, L. (2018). Bottom-up and top-down attention for image captioning and visual question answering. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Z. Gan, C. Gan, X. H. Y. P. K. T. J. G. L. C. and Deng, L. (2017). Semantic compositional networks for visual captioning. *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

1. (O. Vinyals and Erhan, 2015)
2. (K. Xu and Bengio, 2015)
3. (P. Anderson and Zhang, 2018)
4. (M. Cornia and Cucchiara, 2015)
5. (A. Radford and Sutskever, 2021)
6. (J. Lu and Socher, 2017)
7. (Z. Gan and Deng, 2017)