**DL Course Project - CSL4020**

# Image Captioning with Deep Learning: A Comparative Study of LSTM, GRU, and Attention-based Models

**Arnava Srivastava | Rohan Regar | Chahat Baranwal | Atharva Kamble**

Prof. Angshuman Paul

## 1. Introduction

Image captioning represents a fundamental challenge at the intersection of computer vision and natural language processing, requiring systems to understand visual content and generate coherent, contextually appropriate textual descriptions. This task has numerous practical applications, including assisting visually impaired individuals, content-based image retrieval, and enhancing human-computer interaction systems.

Traditional approaches to image captioning relied on template-based methods or retrieval-based techniques that matched images to existing captions. However, these methods lacked flexibility and struggled with novel images. The advent of deep learning has revolutionized image captioning by enabling end-to-end trainable models that can generate more natural and diverse captions.

Recent advances in encoder-decoder architectures have shown promising results for image captioning. Typically, these systems use a Convolutional Neural Network (CNN) as an encoder to extract visual features from images, followed by a Recurrent Neural Network (RNN) as a decoder to generate captions. While Long Short-Term Memory (LSTM) networks have been widely adopted as decoders due to their ability to capture long-term dependencies, alternative architectures such as Gated Recurrent Units (GRUs) and attention mechanisms have emerged as potential improvements.

In this report, we present a comparative study of three decoder architectures for image captioning: LSTM, GRU, and LSTM with attention. Our contributions are threefold:

1. We implement and evaluate three distinct decoder architectures using a consistent encoder framework based on ResNet-50
2. We conduct a systematic comparison of these architectures using both quantitative metrics and qualitative analysis
3. We provide insights into the strengths and limitations of each architecture for the image captioning task

Our study provides valuable insights into the relative performance of different recurrent architectures and contributes to the ongoing research in multimodal learning.

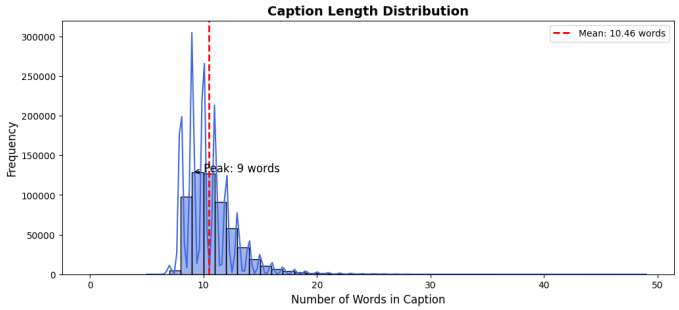


**Figure 1.** Caption length

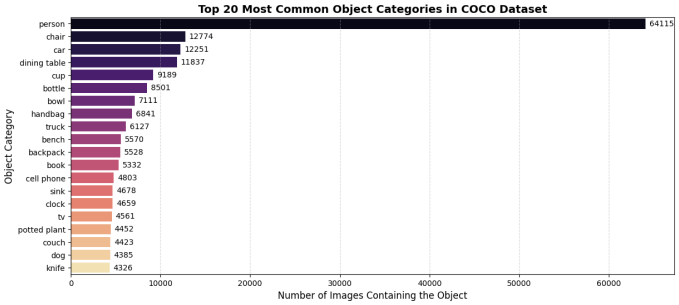


**Figure 2.** Objects

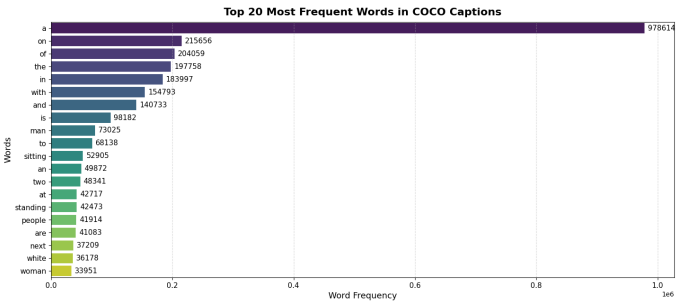


**Figure 3.** Word Count

### 1.1. Dataset

We use the Microsoft Common Objects in Context (MS COCO) dataset, which contains over 120,000 images, each annotated with at least five captions. Our analysis of the dataset reveals several key characteristics:

- The dataset contains a diverse range of images depicting everyday scenes and objects
- Caption lengths follow a normal distribution with a mean of approximately 10 words
- The most frequent words in captions include common objects like "person," "dog," and "car"
- The dataset exhibits a long-tailed distribution of object categories, with some categories appearing much more frequently than others

### 1.2. Preprocessing

#### *1.2.1. Image Preprocessing*

For image preprocessing, we apply the following transformations:

- Resizing the smaller edge of each image to 256 pixels

- Randomly cropping to 224×224 pixels during training
- Applying random horizontal flipping with 50% probability during training
- Normalizing using ImageNet mean and standard deviation values (0.485, 0.456, 0.406) and (0.229, 0.224, 0.225)

#### 1.2.2. Text Preprocessing

For text preprocessing, we:

- Converting all captions to lowercase
- Removing punctuation
- Tokenizing captions into words
- Building a vocabulary with words appearing at least 5 times in the training set
- Adding special tokens: `<start>`, `<end>`, and `<unk>` (unknown)

### 1.3. Model Architecture

#### 1.3.1. Encoder

For all three model variants, we used a pre-trained ResNet-50 as the encoder:

$$\mathbf{f} = \text{EncoderCNN}(\mathbf{I}) \tag{1}$$

where $\mathbf{I}$ is the input image and $\mathbf{f} \in \mathbb{R}^d$ is the encoded feature vector with dimension $d = 256$.

The encoder removes the final classification layer of ResNet-50 and adds a linear layer to map the features to the desired embedding size:

$$\mathbf{f} = W_e \cdot \text{ResNet}(\mathbf{I}) + b_e \tag{2}$$

where $W_e \in \mathbb{R}^{2048 \times d}$ and $b_e \in \mathbb{R}^d$ are learnable parameters.

#### 1.3.2. LSTM Decoder

The LSTM decoder processes the encoded image features and generates captions word by word:

$$\mathbf{h}_t, \mathbf{c}_t = \text{LSTM}(\mathbf{x}_t, \mathbf{h}_{t-1}, \mathbf{c}_{t-1}) \tag{3}$$

$$\mathbf{p}_t = \text{softmax}(W_o \mathbf{h}_t + b_o) \tag{4}$$

where $\mathbf{x}_t$ is the input at time step $t$ (either the image feature vector or the embedding of the previous word), $\mathbf{h}_t$ and $\mathbf{c}_t$ are the hidden state and cell state at time step $t$, and $\mathbf{p}_t$ is the probability distribution over the vocabulary.

#### 1.3.3. GRU Decoder

The GRU decoder follows a similar structure to the LSTM decoder but uses GRU cells instead:

$$\mathbf{h}_t = \text{GRU}(\mathbf{x}_t, \mathbf{h}_{t-1}) \tag{5}$$

$$\mathbf{p}_t = \text{softmax}(W_o \mathbf{h}_t + b_o) \tag{6}$$

The GRU architecture simplifies the LSTM by merging the cell state and hidden state and using only two gates (reset and update) instead of three.

#### 1.3.4. LSTM with Attention Decoder

The attention-based decoder incorporates an attention mechanism that allows the model to focus on different parts of the image when generating each word:

$$\mathbf{a}_t = \text{softmax}(W_a \tanh(W_f \mathbf{f} + W_h \mathbf{h}_{t-1})) \tag{7}$$

$$\mathbf{c}_t = \sum_i a_{t,i} \mathbf{f}_i \tag{8}$$

$$\mathbf{h}_t, \mathbf{c}_t = \text{LSTM}([\mathbf{x}_t; \mathbf{c}_t], \mathbf{h}_{t-1}, \mathbf{c}_{t-1}) \tag{9}$$

$$\mathbf{p}_t = \text{softmax}(W_o \mathbf{h}_t + b_o) \tag{10}$$

where $\mathbf{a}_t$ is the attention weight vector, $\mathbf{c}_t$ is the context vector, and $[\mathbf{x}_t; \mathbf{c}_t]$ denotes the concatenation of the input and context vectors.

The attention mechanism computes weights for different parts of the image features based on the current hidden state, allowing the model to focus on relevant regions.

### 1.4. Training

We train all three models with the following configuration:

- Embedding size: 256
- Hidden size: 512
- Batch size: 64
- Learning rate: 0.001 with Adam optimizer
- Loss function: Cross-Entropy Loss
- Number of epochs: 5

During training, we monitor the loss and perplexity to track the models' progress. We save checkpoints after each epoch to enable model evaluation at different stages of training.

### 1.5. Inference

For caption generation during inference, we implement both greedy search and beam search strategies:

- Greedy search selects the most probable word at each step
- Beam search maintains the top-k most probable sequences at each step, providing more diverse and potentially better captions

## 2. Results and Discussion

We evaluate our three model variants (LSTM, GRU, and LSTM with attention) using both quantitative metrics and qualitative analysis.

### 2.1. Quantitative Evaluation

For quantitative evaluation, we use the BLEU score, which measures the similarity between generated captions and reference captions based on n-gram overlap. The results are presented in Table 1.

**Table 1.** BLEU scores for different decoder architectures

| Model | BLEU Score | BLEU-1 | BLEU-2 |
|---|---|---|---|
| LSTM | 0.184 | 0.587 | 0.408 |
| GRU | 0.205 | 0.597 | 0.426 |
| LSTM with Attention | 0.229 | 0.607 | 0.437 |

The LSTM with attention model achieves the highest BLEU score, followed by the GRU model, with the standard LSTM model showing slightly lower performance. This suggests that the attention mechanism significantly improves caption quality by allowing the model to focus on relevant image regions during caption generation.

### 2.2. Qualitative Analysis

In addition to quantitative metrics, we conduct a qualitative analysis by examining the captions generated by each model for a set of test images. Figure shows example images with captions generated by all three models.

Our qualitative analysis reveals several patterns:

- The LSTM with attention model generally produces more detailed and accurate captions
- The GRU model tends to generate shorter captions but is sometimes more concise and to the point
- The standard LSTM model occasionally misses important details or focuses on incorrect objects

### 2.3. Model Comparison

Based on our evaluation, we can draw several conclusions about the three decoder architectures:


**Figure 4.** Example images with captions generated by different models

1. **LSTM vs. GRU**: While both models perform reasonably well, the LSTM model generally produces slightly better captions according to BLEU scores. This may be due to LSTM's more complex gating mechanism, which allows it to better capture long-term dependencies in caption generation.
2. **Attention Mechanism**: The addition of an attention mechanism to the LSTM model results in significant improvements in caption quality. The attention mechanism allows the model to focus on relevant parts of the image when generating each word, leading to more accurate and detailed captions.
3. **Computational Efficiency**: The GRU model has fewer parameters than the LSTM models, making it more computationally efficient. This could be an important consideration for applications with limited computational resources.

## 3. Conclusion

In this report, we presented a comparative study of three decoder architectures for image captioning: LSTM, GRU, and LSTM with attention. Our results demonstrate that the LSTM with attention model achieves the best performance, highlighting the importance of attention mechanisms in image captioning tasks.

The key contributions of our work include:

- A systematic comparison of different recurrent architectures for image captioning
- Implementation and evaluation of an attention mechanism for improved caption generation
- Insights into the strengths and weaknesses of each architecture

Our research provides valuable insights into the design of image captioning systems and contributes to the ongoing advancement of multimodal learning techniques.

## Individual Contributions

**Atharva Kamble:** Implemented vocabulary creation system, built the GRU-based decoder model for image captioning, and contributed to writing the comprehensive project report.
**Arnava Srivastava:** Designed and implemented the attention mechanism for the LSTM model, enhancing caption generation with focused visual features, and worked on the application for demonstration purposes.
**Rohan Regar:** Created the core LSTM architecture, implemented the encoder-decoder framework with ResNet-50, developed the training methodology, and worked on the application for project demonstration.
**Chahat Baranwal:** Built the end-to-end model pipeline, developed the dataset preparation pipeline, conducted comparative analysis of all three models, and contributed significantly to the project report documentation.

## References

[1] Lin, T. Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., & Zitnick, C. L. (2014). Microsoft COCO: Common objects in context. In European conference on computer vision (pp. 740-755). Retrieved from https://cocodataset.org/#home

[2] Karpathy, A., & Fei-Fei, L. (2014). Deep visual-semantic alignments for generating image descriptions. arXiv preprint arXiv:1405.0312.

[3] Vinyals, O., Toshev, A., Bengio, S., & Erhan, D. (2014). Show and tell: A neural image caption generator. arXiv preprint arXiv:1411.4555.

[4] Regar, R., Srivastava, A., Kamble, A., & Baranwal, C. (2025). ImageCaptioning [Computer software]. GitHub. https://github.com/RohanRegar/ImageCaptioning