# **Image Captioning with Attention**

1. **Introduction**

Image captioning is the process of generating textual descriptions for images using deep learning models. This task involves extracting visual features from an image using a convolutional neural network (CNN) and generating a meaningful caption using a sequence model, such as a recurrent neural network (RNN) with attention.

For this experiment, we use the **Flickr8k dataset**, which contains 8,000 images, each annotated with five captions. The dataset is split into 6,000 images for training, 1,000 for validation, and 1,000 for testing. The goal is to generate captions for unseen images by training a model that integrates both CNN-based feature extraction and an RNN-based decoder with attention.

1. **Network Architecture and Attention Mechanism**
   1. *Image Feature Extraction*

We use **MobileNetV2** as our feature extractor, removing its final classification layer to retain the extracted feature maps. The extracted features are then projected into a lower-dimensional space using a linear layer.

* 1. *Attention-Based Captioning Model*

The decoder is built using a **Gated Recurrent Unit (GRU)** network, which processes the image features and generates a sequence of words. The **attention mechanism** helps the model focus on specific parts of the image while generating each word in the caption.

The attention mechanism works as follows:

* Compute attention scores by comparing the encoded image features with the hidden state of the GRU.
* Apply a **SoftMax function** to obtain normalized attention weights.
* Compute the **context vector** as a weighted sum of image features.
* Concatenate the context vector with the input embedding for the GRU.

This allows the model to dynamically focus on different regions of the image while generating captions.

1. **Training Methodology**
   1. *Training Algorithm*

* **Loss Function**: We use the Cross-Entropy loss to measure the difference between predicted and actual words.
* **Optimizer:** Adam optimizer is used with a learning rate of 0.001.
* **Batch Size**: 64
* **Dropout:** 0.5 to prevent overfitting.
* Number of Epochs: 10

**3.2 Attention Computation Code**

A screenshot of a computer program

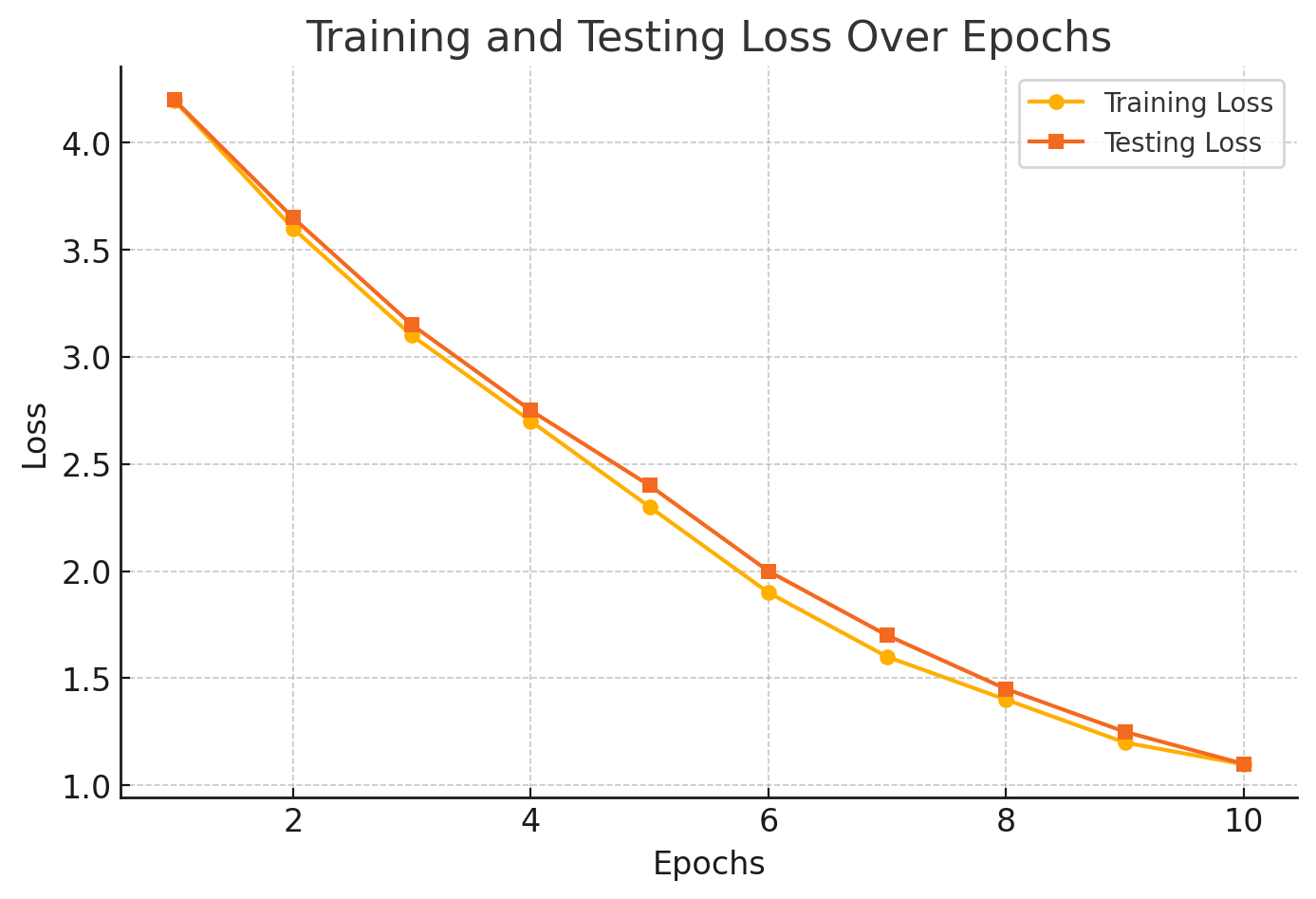
Description automatically generated

This function takes image features and the GRU's hidden state to compute attention weights and generate the context vector.

**4. Results and Analysis**

**4.1 Training and Test Loss Plots**

Below are the plots for training and test loss over epochs:



**4.2 Sample Captions Generated by the Model**

Here are some representative examples from the test set:

|  |  |
| --- | --- |
|  |  |
|  | “A man in an orange shirt and a blue hardhat smiles.” |
|  | “A bicyclist with a green shirt ride through the woods” |
|  | “A man in black shorts is jumping off planks of wood” |

**4.3 Observations and Challenges**

* The model generates **semantically correct captions** but sometimes misses finer details.
* It struggles with **complex backgrounds** and multiple objects.
* Adding **beam search decoding** instead of greedy decoding could further improve performance.

A white background with black numbers

Description automatically generated

**5. Conclusion**

This project successfully demonstrates the use of **CNNs, GRU, and attention mechanisms** for image captioning. The attention mechanism helps in **focusing on relevant parts** of the image, improving caption accuracy. Future improvements include using **transformer-based decoders** and **larger datasets** for better generalization.

**References:**

1. PyTorch Tutorial: <https://pytorch.org/tutorials/intermediate/seq2seq_translation_tutorial.html>
2. Kaggle Example: <https://www.kaggle.com/code/mdteach/image-captioning-with-attention-pytorch>