

# Image Caption - flickr8k

XIU ZHEN HUANG  
30.11.2025

# Overview

- Brief Description of Dataset
- Deep Learning Problem
- Exploratory Data Analysis (EDA)
- Data Cleaning
- Model Architecture
- Results and Analysis
- Conclusion
- Future Work
- Reference

The full project locates at the [git repository](#).

# Brief Description of Dataset

This mini project implements an image captioning system on the Flickr8k dataset using Kaggle.

The dataset contains **8,000 images**, and each image is annotated with **five captions**, resulting in a total of 40,456 captions stored in a text file.

There are 40456 images

image\_path /kaggle/input/flickr8k/Images/

	image	caption
1	1000268201_693b08cb0e.jpg	A child in a pink dress is climbing up a set o...
2	1000268201_693b08cb0e.jpg	A girl going into a wooden building .
3	1000268201_693b08cb0e.jpg	A little girl climbing into a wooden playhouse .
4	1000268201_693b08cb0e.jpg	A little girl climbing the stairs to her playh...
5	1000268201_693b08cb0e.jpg	A little girl in a pink dress going into a woo...

===== Image: 1000268201\_693b08cb0e.jpg =====



Caption 1: A child in a pink dress is climbing up a set of stairs in an entry way .  
Caption 2: A girl going into a wooden building .

Caption 3: A little girl climbing into a wooden playhouse .  
Caption 4: A little girl climbing the stairs to her playhouse .  
Caption 5: A little girl in a pink dress going into a wooden cabin .

# Deep Learning Problem

The Flickr8k dataset is commonly used for training and evaluating image captioning models.

In deep learning-based image captioning, the model usually consist of two major components: an encoder to extract visual features from the image and a decoder that generates descriptive sentences word by word.



# Exploratory Data Analysis (EDA)

Caption 1: A boy holding a lightsaber jumps in the middle of the street .

Caption 2: A boy holding a stick jumping Ont he street .

Caption 3: A small boy in a white shirt is jumping with a sword in his hand .

Caption 4: A young boy jumps in the street while holding a stick .

Caption 5: a boy in shorts and white shirt is jumping whilst holding a pink stick in front of a large tree surrounded by sacks .

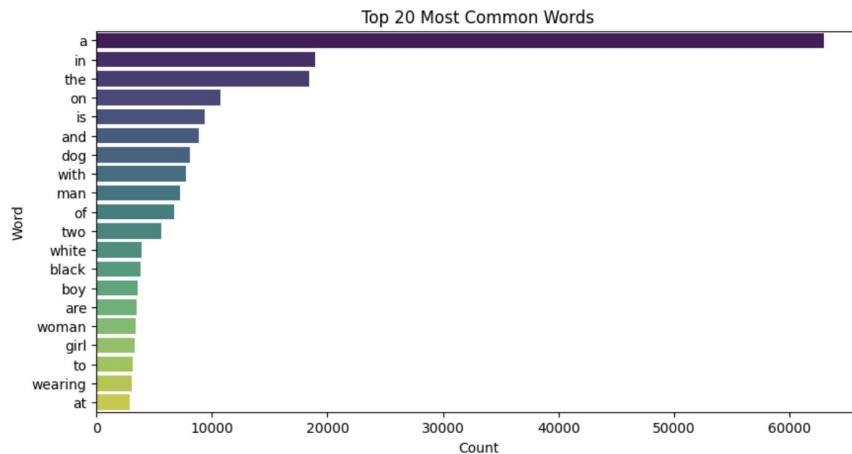
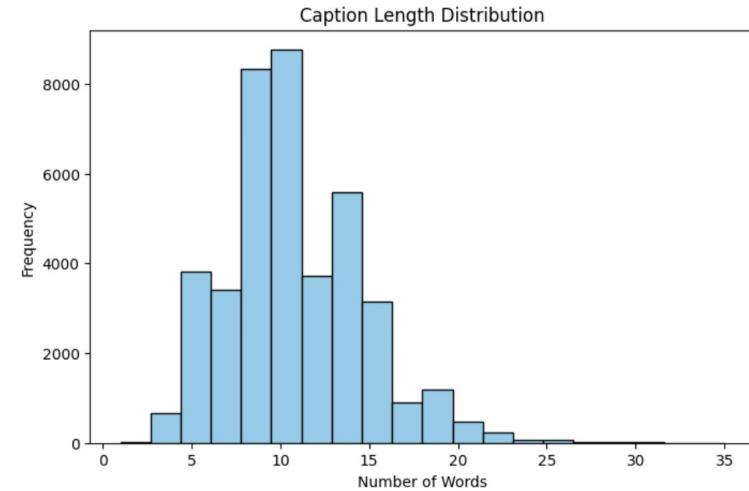


# Exploratory Data Analysis (EDA)

For caption, I showed some EDA to get more understanding about the dataset.

Some words appear more frequency than others, and some words do not.

It indicates that maybe we should do some data cleaning or add others features for our database to make the model better.



# Data Cleaning

- Remove shorter captions to ensure meaningful training input
- Remove the words that appear only once to reduce vocabulary noise
- Added a special <unk> token to handle unseen word during training and inference
- Converted captions to integer sequences and padded them to a fixed maximum length for uniform model input
- Created index-to-word mapping to enable caption decoding during inference

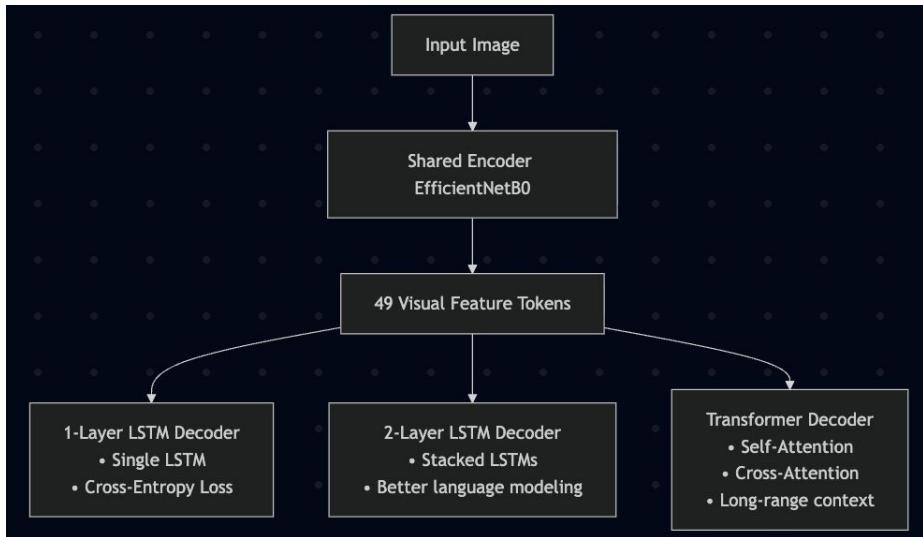
```
frequent_words count: 5198  
before removing shorter caption count: 40456  
after removing shorter caption count: 40451
```

```
Padded captions shape: (40451, 37)  
index 1 -> word: <unk>  
index 2 -> word a
```

# Model Architecture

In this project, I used one shared encoder and compare three decoders. The encoder is responsible for extracting visual feature from images.

Then I varied three decoders and compare how different sequence-generation models affect caption quality.



# Model Architecture - Encoder

I used EfficientNetB0 to extract  $7 \times 7 \times 1280$  feature maps from the input image, reshapes them into 49 visual tokens, and convert each token to a 256-dimensional embedding for the caption decoder.

Layer (type)	Output Shape	Param #
image (InputLayer)	(None, 224, 224, 3)	0
efficientnetb0 (Functional)	(None, 7, 7, 1280)	4,049,571
reshape (Reshape)	(None, 49, 1280)	0
dense (Dense)	(None, 49, 256)	327,936

# Model Architecture - 1 layer LSTM

The 1-layer LSTM decoder is to generate vocabulary logits for image caption generation.

Image features provide the initial hidden states, the decoder generates each word step-by-step and produce a complete caption aligned with the visual content.

Layer (type)	Output Shape	Param #	Connected to
image (InputLayer)	(None, 224, 224, 3)	0	-
efficientnetb0 (Functional)	(None, 7, 7, 1280)	4,049,571	image[0][0]
reshape (Reshape)	(None, 49, 1280)	0	efficientnetb0[0...]
dense (Dense)	(None, 49, 256)	327,936	reshape[0][0]
caption_in (InputLayer)	(None, 36)	0	-
global_average_poo... (GlobalAveragePool...)	(None, 256)	0	dense[0][0]
word_embedding (Embedding)	(None, 36, 256)	1,331,712	caption_in[0][0]
dense_1 (Dense)	(None, 256)	65,792	global_average_p...
dense_2 (Dense)	(None, 256)	65,792	global_average_p...
lstm_layer_1 (LSTM)	(None, 36, 256)	525,312	word_embedding[0][0], dense_1[0][0], dense_2[0][0]
output_dense (Dense)	(None, 36, 5202)	1,336,914	lstm_layer_1[0][0]

# Model Architecture - 2 layer LSTM

A second stacked LSTM uses the sequence representation, enabling deeper language modeling before generating vocabulary logits for each word in the caption.

Layer (type)	Output Shape	Param #	Connected to
image (InputLayer)	(None, 224, 224, 3)	0	-
efficientnetb0 (Functional)	(None, 7, 7, 1280)	4,049,571	image[0][0]
reshape (Reshape)	(None, 49, 1280)	0	efficientnetb0[0...]
dense (Dense)	(None, 49, 256)	327,936	reshape[0][0]
caption_in (InputLayer)	(None, 36)	0	-
global_average_pool (GlobalAveragePool)	(None, 256)	0	dense[0][0]
word_embedding (Embedding)	(None, 36, 256)	1,331,712	caption_in[0][0]
dense_3 (Dense)	(None, 256)	65,792	global_average_p...
dense_4 (Dense)	(None, 256)	65,792	global_average_p...
lstm_layer_1 (LSTM)	(None, 36, 256)	525,312	word_embedding[0...] dense_3[0][0], dense_4[0][0]
not_equal_1 (NotEqual)	(None, 36)	0	caption_in[0][0]
lstm_layer_2 (LSTM)	(None, 36, 256)	525,312	lstm_layer_1[0][...] not_equal_1[0][0]
output_dense (Dense)	(None, 36, 5202)	1,336,914	lstm_layer_2[0][...]

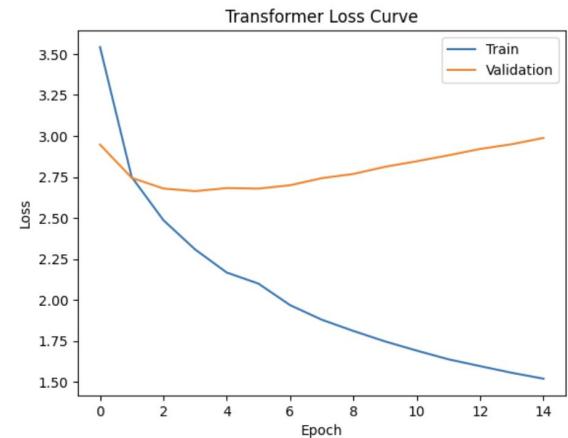
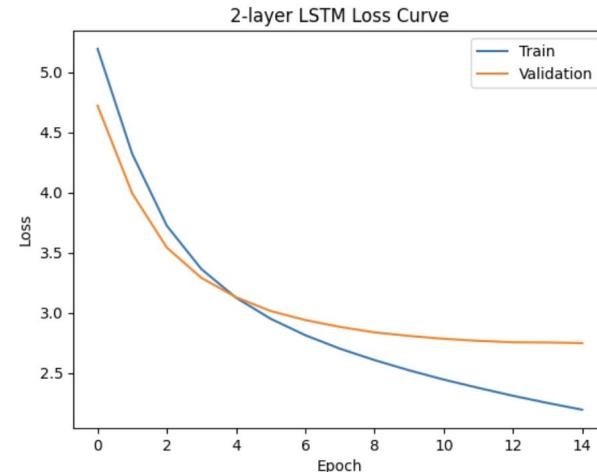
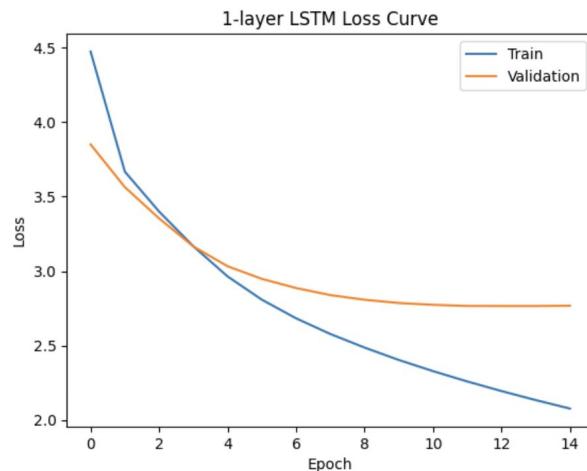
# Model Architecture - Transformer

The Transformer decoder embeds caption tokens with **positional encoding** and processes them through **self-attention** and **cross-attention layers**. It attends to the 49 visual tokens extracted by the encoder and generates vocabulary logits for each time step to produce the final caption.

Layer (type)	Output Shape	Param #	Connected to
image (InputLayer)	(None, 224, 224, 3)	0	-
efficientnetb0 (Functional)	(None, 7, 7, 1280)	4,049,571	image[0][0]
reshape (Reshape)	(None, 49, 1280)	0	efficientnetb0[0...]
dense (Dense)	(None, 49, 256)	327,936	reshape[0][0]
caption_in (InputLayer)	(None, 36)	0	-
caption_transformer (CaptionTransformer)	(None, 36, 5202)	7,405,138	dense[0][0], caption_in[0][0]

# Loss Curve

The loss curves indicate that the LSTM models exhibit more stable training behavior, while the Transformer achieves the lowest training loss. However, due to the limited size of the dataset, the Transformer also shows clear signs of overfitting.



# Results and Analysis

Model outputs:

1-layer LSTM : a small boy in a red shirt is walking on the sidewalk

2-layer LSTM : a boy in a blue shirt is running on the sidewalk

Transformer : a young boy in a white shirt and blue jeans is holding a sword



# Results and Analysis

Model outputs:

1-layer LSTM : two girls in a blue and yellow dress are playing in a sprinkler

2-layer LSTM : two young girls are playing in a park

Transformer : two little girls in green boots playing with a toy



# Results and Analysis

The results show that the Transformer achieves higher scores than the LSTM models, reflecting stronger language generation and accuracy.

However, its rising validation loss suggests overfitting, implying that a larger dataset would help it generalize better.

Model	BLEU-1	BLEU-2	BLEU-3	BLEU-4	METEOR	CIDEr
1-layer LSTM	0.568	0.383	0.245	0.155	0.185	0.374
2-layer LSTM	0.621	0.436	0.288	0.194	0.202	0.492
Transformer Decoder	0.750	0.605	0.474	0.373	0.300	1.139

# Conclusion

In this project, I compared three fundamental image captioning models using a shared EfficientNetB0 encoder and three different decoders: a 1-layer LSTM, a 2-layer LSTM, and a Transformer decoder.

Based on the **loss curves** and evaluation metrics including **BLEU**, **METEOR**, and **CIDEr**, the model capacity shows a clear impact on caption quality.

For small-scale datasets such as Flickr8k, the **2-layer LSTM** achieves the best balance between stability and performance. Although the **Transformer** decoder obtains the highest language scores, it exhibits noticeable overfitting due to limited data. With a larger dataset, the Transformer architecture is expected to outperform the LSTM models more reliably.

# Future Work

- Using a larger dataset could help reduce overfitting in the Transformer decoder and allow it to fully realize its modeling capacity.
- Analyzing or visualizing cross-attention maps may provide insights into how the model aligns image regions with generated words, potentially guiding further architecture improvements.
- Incorporating pre-trained word embeddings such as GloVe or FastText may enhance the semantic quality of generated captions, especially when training data is limited.

# Reference

- [A survey on image captioning methods](#)
- [Show and Tell: A Neural Image Caption Generator](#)
- [Comparative study of Transformer and LSTM Network with attention mechanism on Image Captioning](#)
- [Image Captioning based on Deep Learning Methods: A Survey](#)
- [From Show to Tell: A Survey on Deep Learning-based Image Captioning](#)