Image Captioning

Group 6 – Yunhong Yang, Ziyue Li, Yutao Chen, Xiaodan Lu

Table of Contents

Introduction

What do you see in this picture?

------ “A young girl is eating broccoli in cream sauce.”

Image Source: MS Coco Dataset

Introduction

- Image Captioning is the task of describing the content of an image in words.
- Image captioning is an intricate endeavor bridging computer vision and natural language processing (NLP).
- In this, the input to the model is an image and the model's output is a caption generated in natural language processing.

Motivations

The first thing to ponder is the significance of the image captioning problem statement. Image captioning has a huge amount of application. Let's explore a few noteworthy examples:

- Aid to the Blind
- Autonomous vehicles
- Google Image Search

MS COCO Dataset

The MS COCO (Microsoft Common Objects in Context) dataset is a comprehensive collection encompassing object detection, segmentation, key-point detection, and captioning tasks. This extensive dataset comprises more than 200,000 images.

“A man wearing earphones doing a trick on a skateboard ramp.”

“There is a surfer wearing a body suit riding a wave.”

“The teenagers are standing together on the sidewalk.”

Image Source: MS Coco Dataset

Data Preprocessing

Data Cleaning

- Converted text to lowercase.
- Remove extra white spaces.
- Removed punctuation and special characters.
- Added Start and End Tokens.

Data Splitting

- Train Set: 80%
- Validation Set: 10%
- Test Set: 10%
- Sizes: Training (7228), Validation (903), Test (904).

Tokenization & Padding

- Map tokens to integers.
- Utilized padding to ensure consistent sequence length.

Resizing Images

- Resized each image to the specified dimensions to ensure consistent input size for neural networks.

Methodology

- Encoder
- Embedding Layer
- Decoder
- Evaluation Metrics

Encoder

InceptionV3 Model

- Inception v3 is a convolutional neural network for assisting in image analysis and object detection, and got its start as a module for GoogLeNet.
- The InceptionV3 model is loaded with pre-trained weights, and its layers are frozen to prevent further training.

RepeatVector

- Why: Single image encoding may not capture enough dynamic details
- Purpose: Replicate image encoding
- Effect: Ensures consistent image context for every word
- Benefits:
  - Enhanced model understanding.
  - Improved generation of contextually relevant captions.

Source: https://towardsdatascience.com/step-by-step-understanding-lstm-autoencoder-layers-ffab055b6352

Embedding

How it works

- Mapping: Converts words to vectors in a continuous space.
- Contextualization: Embeddings capture semantic relationships.
- Dropout: Introduces regularization, preventing overfitting.

Source: https://www.codingninjas.com/studio/library/embedding-layers-in-keras

Encoding & Embedding Process

Encoding → **Encoding** → **Encoding** → **Embedding** → **Concatenating**

Encoding

Preprocess input and add dense layer for image encoding with ReLU activation

Encoding

Flatten the encoding to be used as input in subsequent layers

Encoding

Repeat the flattened encoding for each time step in the caption

Embedding

Embedding layer for the teacher forcing input and apply dropout

Concatenating

Concatenate the flattened encoding and embedded teacher forcing input

LSTM: Decoder

Why?

- Decoding the fixed-length vector and outputting the predicted sequence

In the encoder part: only one vector in the last time step and neglecting all the others

In the decoder part: an output vector at every time step so the Dense layer can make a prediction.

Dense Layer

Why?

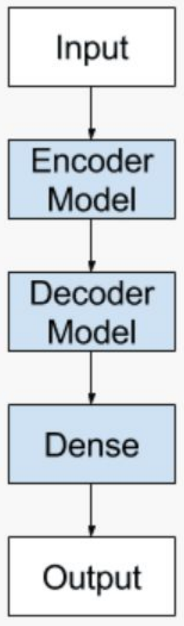
- The decoder is just a language model conditioned on the initial states.

last step:
Predict the Caption.

The number of units is the shape of the output vector

Source:
https://towardsdatascience.com/how-to-build-an-encoder-decoder-translation-model-using-lstm-with-python-and-keras-a31e9d864b9b

Summary



Encoder-Decoder LSTM Model Architecture

Input: Input Text data(Caption) and Image Data

Encoder: InceptionV3 Model as encoder to extract features from the image. processes an input sequence and generates an encoded state

Decoder: LSTM as decoder. uses the encoded state to produce an output sequence.

Dense: Predict the Caption

Output: A sentence of the image caption

Source: https://machinelearningmastery.com/encoder-decoder-long-short-term-memory-networks/

Loss Function

Cross entropy serves as the loss function for model training.

1. converts the y_true tensor to integer type, which is suitable for handling class indices.
2. ignore loss contributions from padded values (where y_true equals 0)
3. The y_true tensor is one-hot encoded. The words + 1 represents the number of classes, and each class is assigned a unique one-hot vector.
4. used to compute the cross-entropy loss between the predicted logits (y_preds) and the true labels (y_true_one_hot).

```
# loss function
def sparse_it_up(y_true, y_preds):
    # Cast true labels to integers
    y_true = tf.cast(y_true, tf.int32)
    # Create a mask for non-zero elements in true labels
    mask = tf.math.logical_not(tf.math.equal(y_true, 0))
    mask = tf.cast(mask, dtype=tf.float32)
    # Convert true labels to one-hot encoding
    y_true_one_hot = tf.one_hot(y_true, words + 1)
    # Calculate          categorical cross-entropy with masking
    loss = tf.reduce_mean(tf.nn.softmax_cross_entropy_with_logits(y_true_one_hot, y_preds) * mask)
    return loss
```

Evaluation Metrics

Metrics used: BLEU, METEOR and ROUGE scores

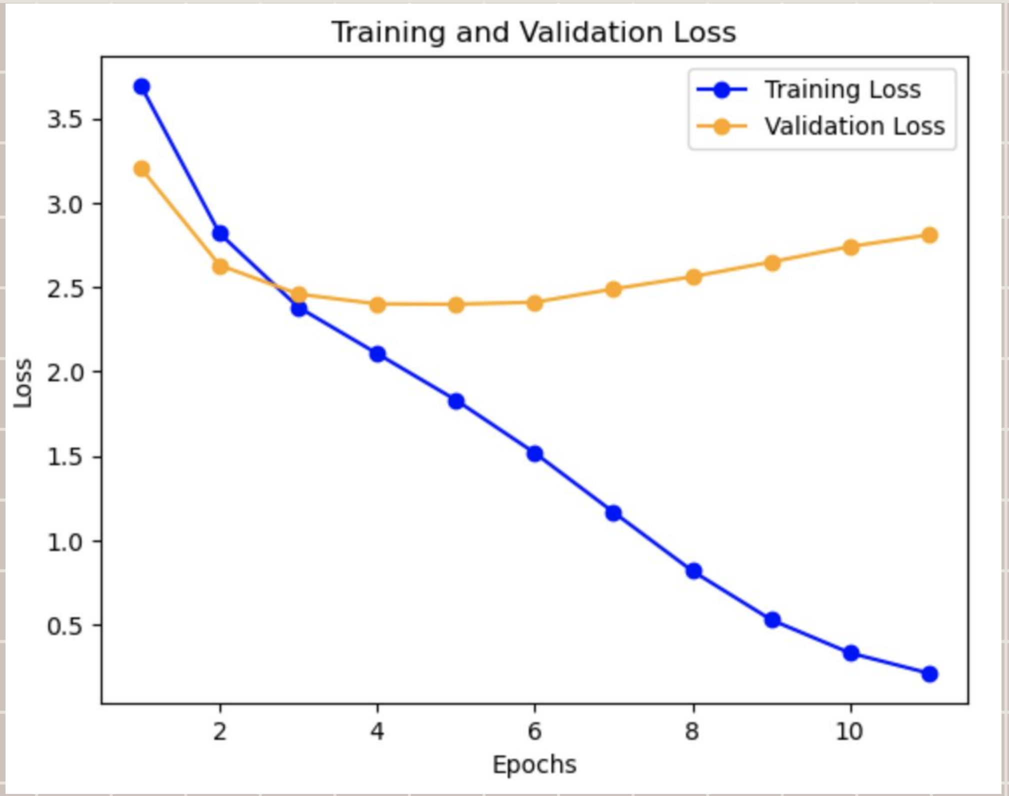
Metric	Problem Solved	Method
ROUGE	Text summarization	Measures overlap of N-grams and LCS between system-generated and reference summaries
BLEU	Machine translation	Measures N-gram precision between candidate and reference translations
METEOR	Machine translation	Harmonic mean of unigram precision and recall, with a penalty for length mismatches

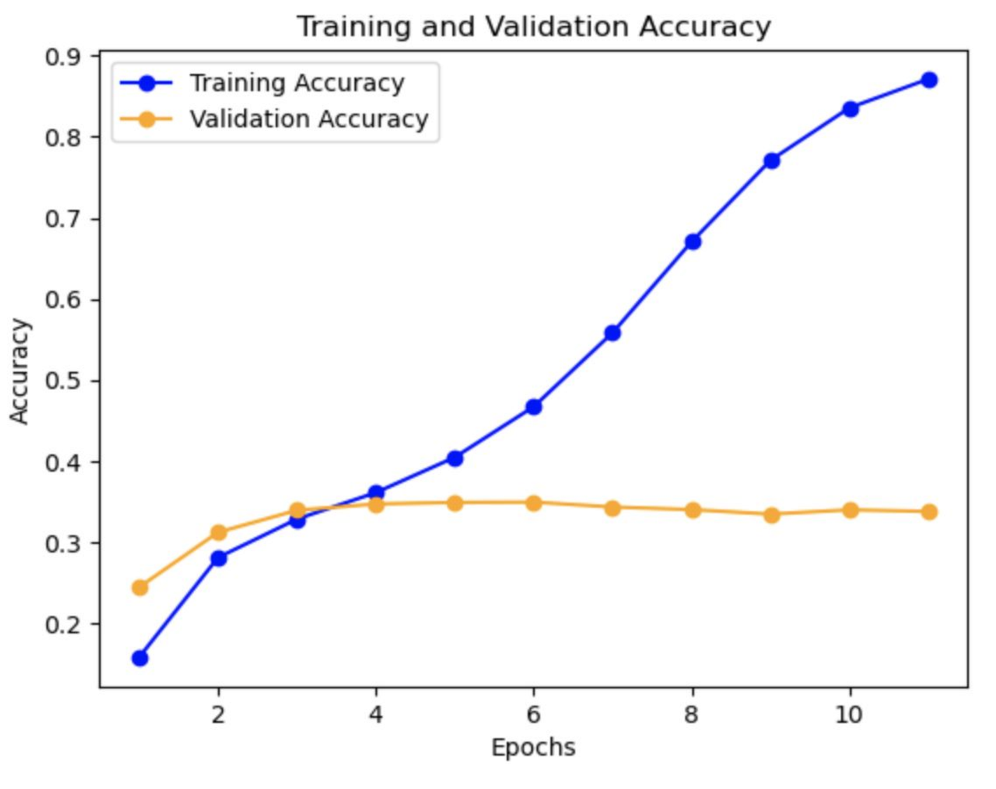
Image source

Results

Result

Loss Function



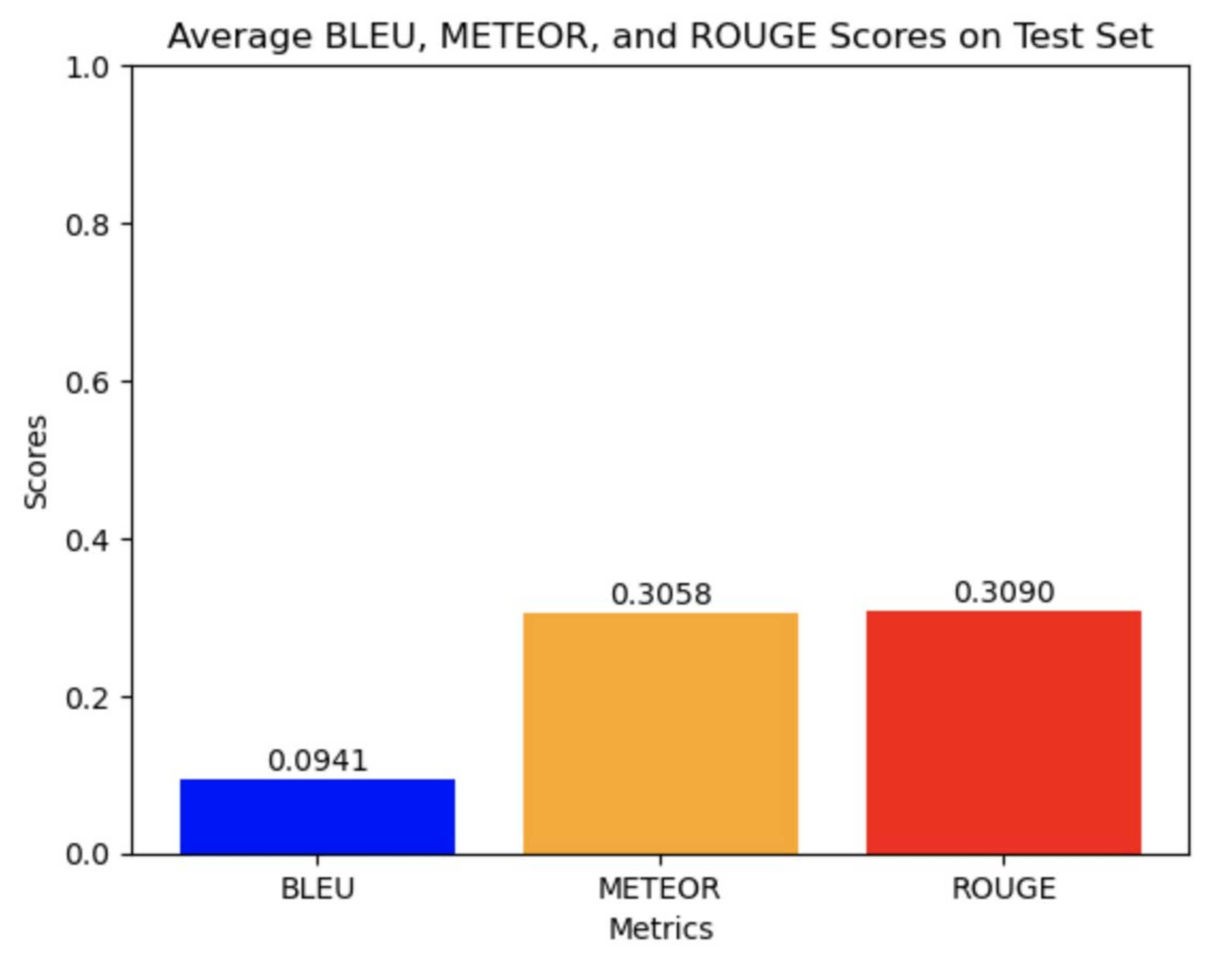


Result

Average BLEU: 0.0941

Average METEOR: 0.3059

Average ROUGE: 0.3090



Conclusion

Conclusion

Label: a man riding a skateboard down a sidewalk
Predicted: a man riding a skateboard while a skateboard

Label: a man on a snowboard who is performing a jump
Predicted: a person riding a snowboard with is skiing a snowboard

Label: a child laying on a bed in a room
Predicted: a small is on a bed with a bed

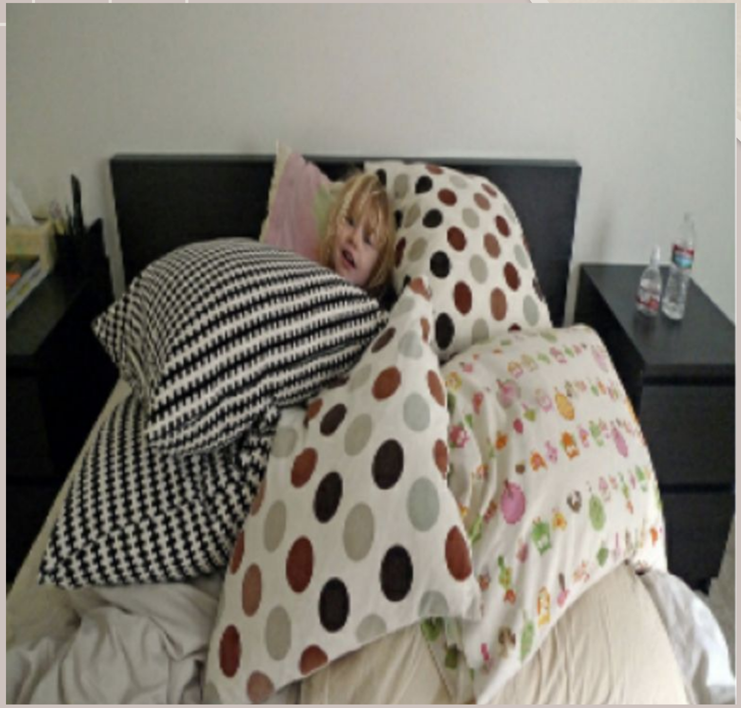


Image Source: MS Coco Dataset

Limitation & Future Thoughts

Limitation

Only 10,000 datasets about “person” were selected in coco dataset.

Future Thoughts

Exploring more diverse datasets
Improving its robustness and versatility

Image source

Thank you!

Q & A

References:

- BM, Nechu. “How to Build an Encoder Decoder Translation Model Using LSTM with Python and Keras.” *Medium*, Towards Data Science, 21 Oct. 2020, towardsdatascience.com/how-to-build-an-encoder-decoder-translation-model-using-lstm-with-python-and-keras-a31e9d864b9b.
- Dhote, Pradeep. “Seq2Seq-Encoder-Decoder-LSTM-Model.” *Medium*, Medium, 20 Aug. 2020, pradeep-dhote9.medium.com/seq2seq-encoder-decoder-lstm-model-1a1c9a43bbac.
- “Rouge Score.” *TorchMetrics*, torchmetrics.readthedocs.io/en/stable/text/rouge_score.html. Accessed 30 Nov. 2023.
- Awsaf. “Coco 2017 Dataset.” *Kaggle*, 4 Sept. 2020, www.kaggle.com/datasets/awsaf49/coco-2017-dataset.
- Mansuy, Raphael. “Evaluating the Performance of Natural Language Processing (NLP) Models Can Be Challenging.” *Medium*, 27 Nov. 2023, ai.plainenglish.io/evaluating-the-performance-of-natural-language-processing-nlp-models-can-be-challenging-ce6f62c07c35.