Individual Report

1 Introduction

An image without a caption can sometimes lead viewer to get the wrong impression of what is going on in a picture. A good caption can provide the viewer with necessary information, focus on key details, and enhance the overall understanding of the image by using just a few words. This project demonstrates machines' capability to generate captions for pictures in natural language sentence format by using a machine learning model. The model using TensorFlow based is built on the architecture described in the Show, Attend, and Tell Paper. We evaluate the model using BLUE score that compares the official caption with the automated generated caption.

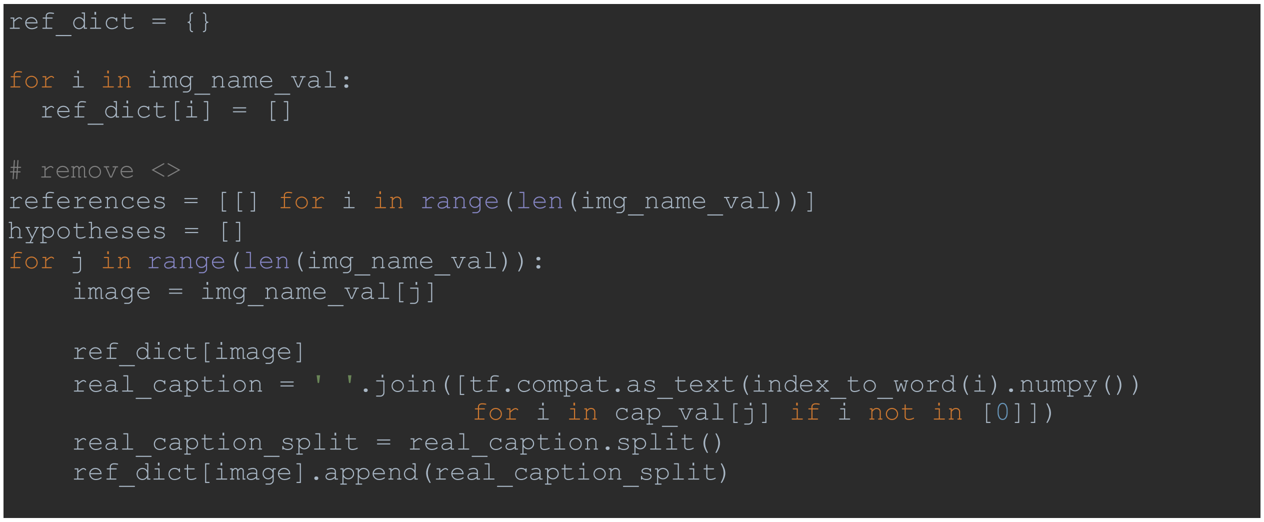
2 Description of Individual Work

My portion was divided into gaining a deeper understanding of the concepts we decided to cover in the project and implementing my portion of the code. Here is a list of the things I did:

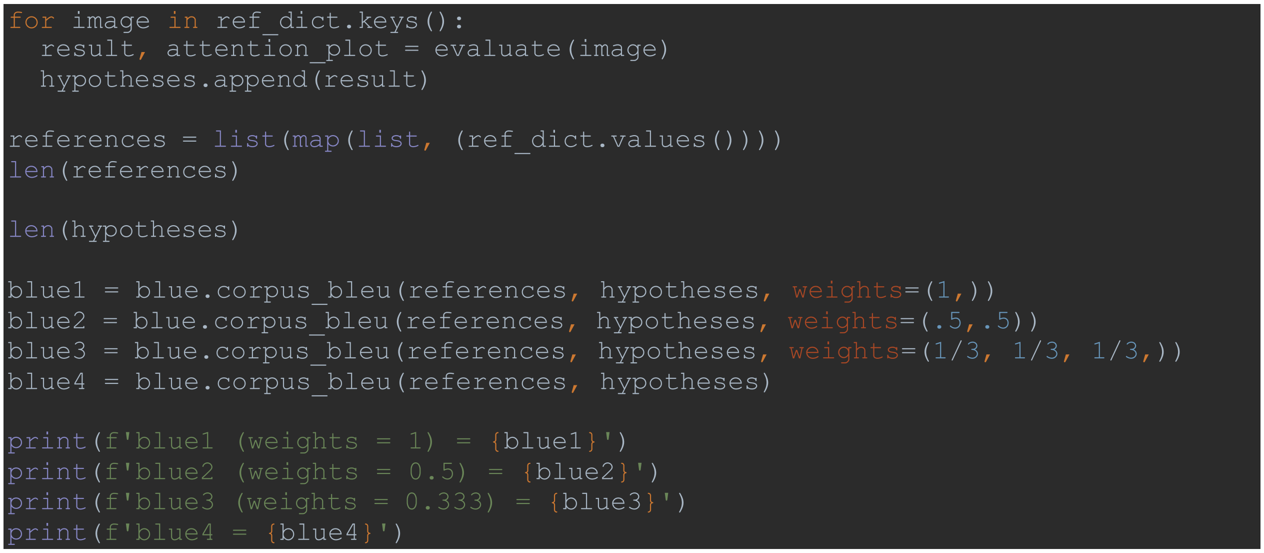
- Select MS-COCO dataset
- Looked for tutorials in PyTorch and TensorFlow
- Read Show, Attend, and Tell paper
- Read BLEU: a Method for Automatic Evaluation of Machine Translation
- Implement BLEU score for our model

3 Details of Individual Work

We divided our group project based on our individual strengths. Alice's undergraduate is in mathematics, Adel's major is data analytics, and my background is in coding, so it made sense that my focus was on implementing image captioning model in TensorFlow. We used the notebook offered by TensorFlow tutorial in Image Captioning as a jumping off point. Our initial class is called image_captioning.py. I added the BLEU implementation.



```
ref_dict = {}

for i in img_name_val:
  ref_dict[i] = []

# remove <>
references = [[] for i in range(len(img_name_val))]
hypotheses = []
for j in range(len(img_name_val)):
    image = img_name_val[j]

    ref_dict[image]
    real_caption = ' '.join([tf.compat.as_text(index_to_word(i).numpy())
                             for i in cap_val[j] if i not in [0]])
    real_caption_split = real_caption.split()
    ref_dict[image].append(real_caption_split)
```



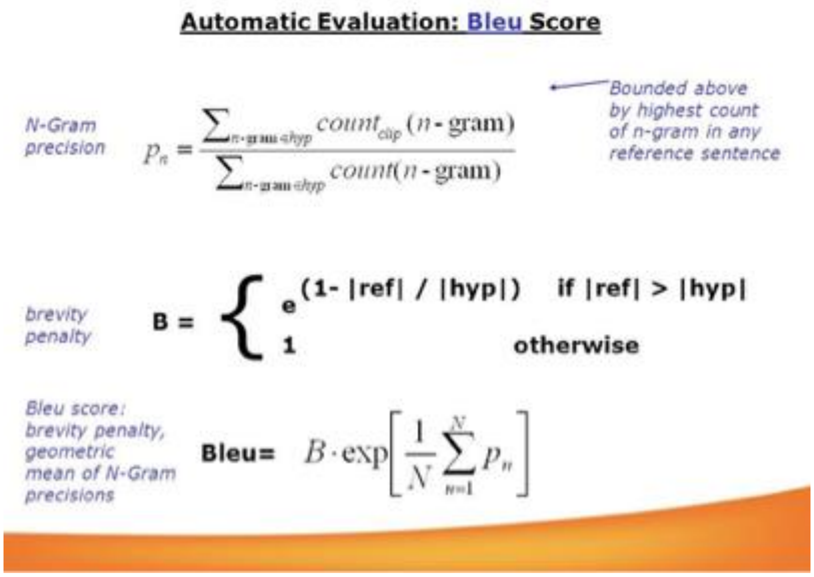
```
for image in ref_dict.keys():
  result, attention_plot = evaluate(image)
  hypotheses.append(result)

references = list(map(list, (ref_dict.values())))
len(references)

len(hypotheses)

blue1 = blue.corpus_bleu(references, hypotheses, weights=(1,))
blue2 = blue.corpus_bleu(references, hypotheses, weights=(.5,.5))
blue3 = blue.corpus_bleu(references, hypotheses, weights=(1/3, 1/3, 1/3,))
blue4 = blue.corpus_bleu(references, hypotheses)

print(f'blue1 (weights = 1) = {blue1}')
print(f'blue2 (weights = 0.5) = {blue2}')
print(f'blue3 (weights = 0.333) = {blue3}')
print(f'blue4 = {blue4}')
```

BLEU (Bilingual Evaluation Understudy) score is an evaluation metric used to assess the quality of a machine-generated text against reference text (Brownlee, 2019). It was initially developed for translation, but it has been adopted to evaluate other text generating tasks such as image captioning. It ranges from 0 to 1, with 1 being a perfect match of the reference text and 0 being a complete mismatch of the reference. The BLEU score is a simple metric calculation that offers its own strengths and weaknesses. For its strengths, it is inexpensive to calculate, it is widely adopted, and most importantly it correlates highly with human evaluation (Brownlee, 2019). We will be able to understand the flaws of BLEU when we describe how it is calculated.

The BLEU score calculation is simple; count matching n-grams of the candidate text to n-grams in the reference text. Matches are independent of word order. Then, you calculate the precision using the following equation (Khandelwal, 2021).

$$\text{Precision} = \frac{\text{No. of candidate translation words occuring in any reference translation}}{\text{Total no. of words in the candidate translation}}$$

This basic approach falls prey to candidate texts that contain common words like “the” since they will create matches but do not necessarily do a good job of reflecting the reference text (Khandelwal, 2021). Modified n-gram precision clips the counts of n-grams to the number of occurrences in the reference. In addition to clipping, a brevity penalty was also introduced by the authors of BLEU to prevent machine-generated text that is too short compared to the reference text. The brevity penalty and the modified n-gram precision together make up the BLEU score. Cumulative n-gram scores are used in our project; this score is the weighted geometric mean of individual n-gram scores from 1 to n (in our case, n=4). Below is the complete BLEU score formula (Vashee, 2017).



Automatic Evaluation: Bleu Score

N-Gram precision

$$p_n = \frac{\sum_{n\text{-gram} \in hyp} count_{clip}(n\text{-gram})}{\sum_{n\text{-gram} \in hyp} count(n\text{-gram})}$$

Bounded above by highest count of n-gram in any reference sentence

brevity penalty

$$B = \begin{cases} e^{(1- |ref| / |hyp|)} & \text{if } |ref| > |hyp| \\ 1 & \text{otherwise} \end{cases}$$

Bleu score: brevity penalty, geometric mean of N-Gram precisions

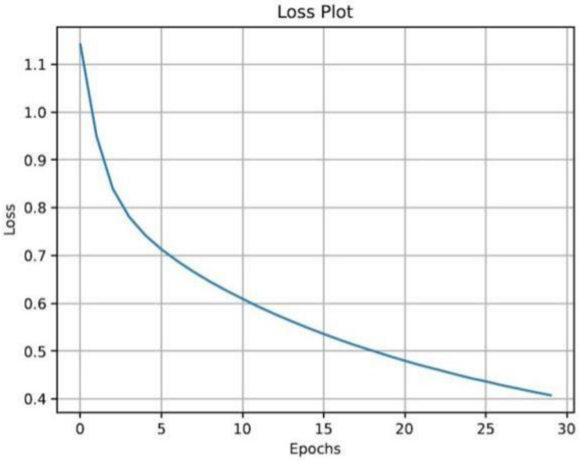
$$\text{Bleu} = B \cdot \exp\left[\frac{1}{N}\sum_{n=1}^{N} p_n\right]$$

The Show, Attend, and Tell paper includes BLEU scores for several different models. Below, we compare our model to the Soft-attention and Hard-attention models featured in the paper.

	BLEU-1	BLEU-2	BLEU-3	BLEU-4
Our model	46.4	23.9	12.3	6.3
Soft-attention	70.7	49.2	34.4	24.3
Hard-attention	71.8	50.4	35.7	25.0

4 Results

We trained the model on a subset of the COCO 2014 dataset. This resulted in 40,018 training images and 200,090 training captions because there are five captions per image. For test, we have 10,003 images and 50,015 captions. We trained using 50, 65, 85, and 100 epochs. Here is the loss plot when we trained using 25 epochs using Adam optimizer. The plot shows a steady decline in loss and approaching 0.4.



These are the BLEU scores our model produced after I added the BLEU implementation:

blue1 (weights = 1) = 0.46400499862953876
blue2 (weights = 0.5) = 0.23941421457469383
blue3 (weights = 0.333) = 0.12275177644615522
blue4 = 0.06272706738679507

5 Summary and Conclusions

We were able to build a model capable of producing captions for an image. Although the captions were not the most accurate, it is a step in the right direction. This project gave me further exposure to the attention mechanism, encoder-decoder models, transfer learning, image captioning, and the BLEU score. In the future, I would attempt to tweak the model by making it bigger or adding different types of augmentations to improve the model.

Total lines of code = 333
Total lines of code
modified = 0 Total lines of
code added = 27
Percentage of code = (333-0)/ (333+27) * 100 = 92.5%

6 References

COCO - Common Objects in Context. (n.d.). COCODataset. Retrieved April 11, 2022, from https://cocodataset.org/

Image captioning with visual attention | TensorFlow Core. (2022, January 26). TensorFlow. Retrieved April 15, 2022, from https://www.tensorflow.org/tutorials/text/image_captioning

Xu, Ba, Kiros, K. J. R. (2016, April 19). *Show, Attend and Tell: Neural Image Caption Generation with Visual Attention*. Arxiv.Org. Retrieved April 4, 2022, from https://arxiv.org/pdf/1502.03044.pdf?msclkid=97863852c49311ecb15903abb2f078b2

Khandelwal, R. (2021, December 13). *BLEU — Bilingual Evaluation Understudy - Towards Data Science*. Medium. Retrieved April 26, 2022, from https://towardsdatascience.com/bleu-bilingual-evaluation-understudy-2b4eab9bcfd1

Brownlee, J. (2019, December 18). *A Gentle Introduction to Calculating the BLEU Score for Text in Python*. Machine Learning Mastery. Retrieved April 26, 2022, from https://machinelearningmastery.com/calculate-bleu-score-for-text-python/

Papineni, Roukos, Ward, Zhu (2022, July 1). BLEU: a Method for Automatic Evaluation of Machine Translation. Dl.acm.org. Retrieved April 26, 2022, from https://aclanthology.org/P02-1040.pdf

Vashee, Kirti. (2017, April 11). *The Problem with BLEU and Neural Machine Translation*. Blogger.Com. Retrieved April 26, 2022, from http://kv-emptypages.blogspot.com/2017/04/the-problem-with-bleu-and-neural.html