Automated Multilingual Image Captioning

Jasmeet Singh 664405139
Prakhar Srivastava 676173979

IDS 576 Deep Learning and Applications
UIC Business Liautaud Graduate School

Dec 1, 2021

1. Introduction

Computer vision and machine learning researchers are now working on teaching computers to automatically generate meaningful captions for images. This job combines image scene comprehension, feature extraction, and visual representation translation into natural languages. This research has a lot of potential, such as developing assistive technology for persons who are blind and automating caption duties on the internet.

The process of creating a written description of an image is known as image captioning. The captions are generated using both Natural Language Processing and Computer Vision. Image captioning is a one-to-many RNN application. Based on the vocabulary of train data, the model predicts the caption for a particular input image. For this case study, we are considering the Flickr8K dataset. RNNs have become very powerful. Especially for sequential data modeling. There are basically four types of RNNs.

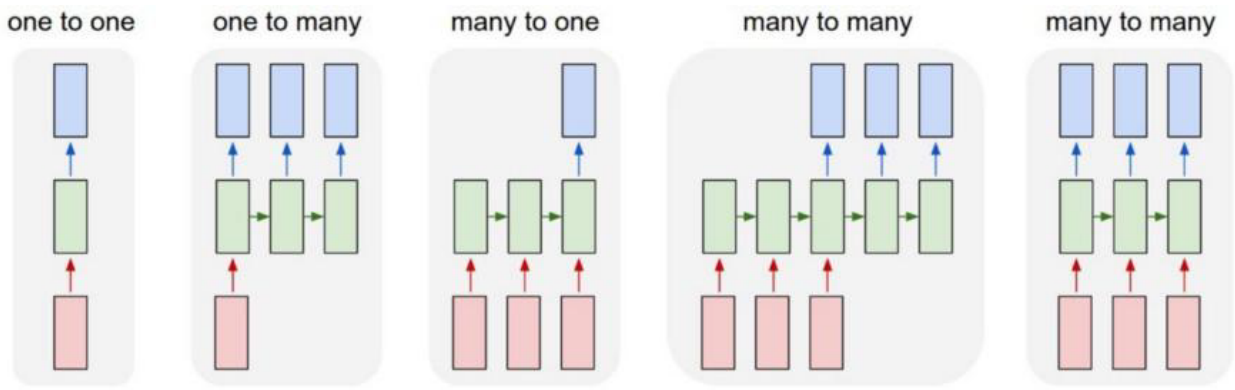

Fig 1: Types of RNNs

Image Captioning is the technique in which automatic descriptions are generated for an image. It can be thought of as an end-to-end Sequence to Sequence problem since it turns images from a series of pixels to a series of words. Both the language or remarks as well as the images must be processed for this purpose. To get the feature vectors, we employ recurrent Neural Networks for the Language component and Convolutional Neural Networks for the Image part.

2. Related Work

This project is built using the base code base available at :
https://github.com/ankitbvs/Automated-Image-Captioning-System#references

Library/code to convert the output caption to audio
https://www.kaggle.com/sumitkg86/eye-for-blind

Library to convert linguistics (English to Hindi)
https://www.geeksforgeeks.org/build-an-application-to-translate-english-to-hindi-in-python/

3. Materials and Methods

3.1. Dataset

Flickr 8K

Flickr8K has 6000 training images, 1000 validation images and 1000 testing images, all in JPEG format. Each image has 5 captions which provide clear descriptions of the salient entities and events.

Total Images - 8091
Total Captions - 40455

The small child climbs on a red ropes on a playground .
A small child grips onto the red ropes at the playground .
A little girl in pink climbs a rope bridge at the park .
A little girl climbing on red roping .
A child playing on a rope net .

Fig 2: An image with its 5 caption

3.2. Data Preprocessing and Feature Engineering

Data preprocessing involves loading and resizing image data into (N × 3 × 229 × 229) dimension and normalizing pixel value to be within range [0, 1] with mean value of [0.440, 0.417, 0.384] and std of [0.286, 0.280, 0.290]

Steps Involved

- To making captions distinguishable, they are differentiated by tags "< start >" , "< end >"
- For the decoder part, we first tokenized the words and converted to lower cases and then we had to convert all the words into indexes to get the word embeddings
- We constructed a vocabulary of words which consists of frequent words in the training caption data along with which we added few special words like , ,
- We have also added the word where if the word is not present in the vocabulary it is represented as <unk>
- In the RNN part of the model, each word in the caption is passed to the model one by one along with the corresponding image

- Initially, the image is passed into the model along with the first word and it is mapped to generate the corresponding second word.
- This process is repeated until is encountered it stops generating sentence and it marks the end of the caption
- Two input arrays are passed into the model, one for passing features of image and the other one for passing text data in encoded format
- The output of the model is the encoded next word of the sequence
- Finally, when the model is used to generate descriptions for an image, the words generated in the previous sequence are concatenated and fed into the model to generate the next word.

3.3. Data Exploration

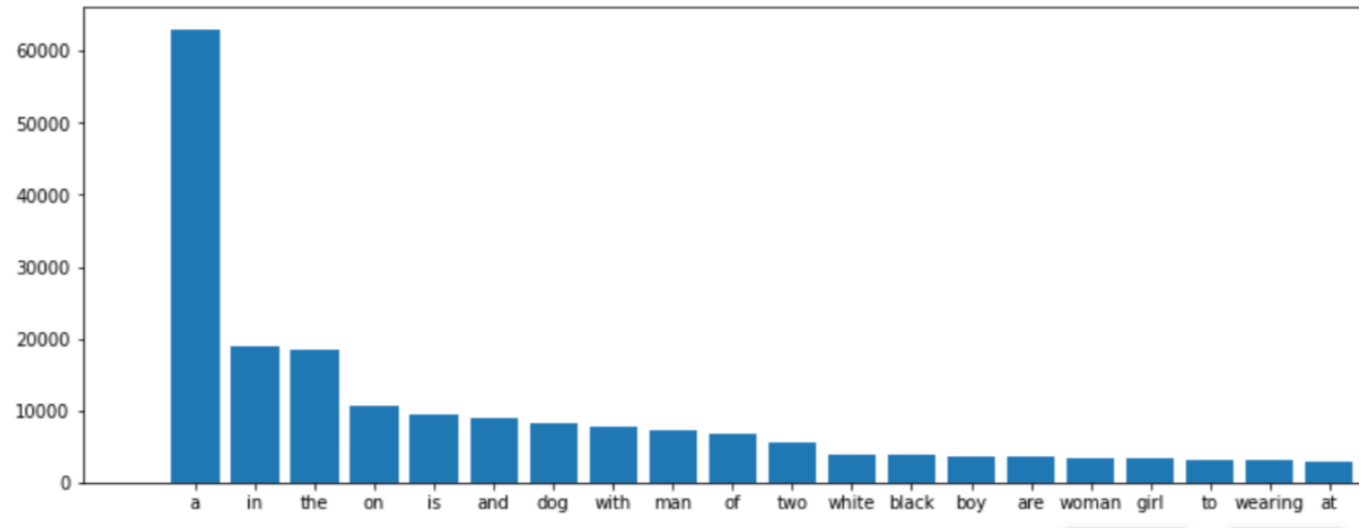

Fig 3 : words frequency in captions

Majority of the words which occur most frequently are the helping (auxiliary) words like a , in , on, is. These words themselves do not explain the image and need other words to make sense and explain the image. This provides us with the intuition of the N-gram models and its weightage.

4. Modeling

4.1. Train-Test Split

The size of the dataset is 8091 images with 40455 captions , around 5 captions each for an image. The data set is then divided into training and test data sets. Training set is 80% of the data and the test and validation set is the other 20%.

4.2. Model Parameters

To make the system understand the image, we need to disintegrate the image into chunks to extract the features. Image Captioning makes use of three primary components.

- **Image Feature Encoder**
  This takes the source photo as input and produces an encoded representation of it that captures its essential features.

- **Sequence Decoder**
  This takes the encoded representation of the image and outputs a sequence of tokens that describes the photo.

- **Sentence Generator**
  The job of the Sentence Generator is to take the sequence of tokens and output a caption which is a sentence of words in the desired language that describes the photo.

Conventional encoder-decoder architectures encoded each source sentence into a fixed-length vector, regardless of its length, from which the decoder would output a translation. As a result, the neural network struggled to cope with extended words, resulting in a performance bottleneck. The Bahdanau attention was proposed to address the performance bottleneck of conventional encoder-decoder architectures, achieving significant improvements over the conventional approach.The main components in use by the Bahdanau encoder-decoder architecture are the following:

- s_{t-1} is the *hidden decoder state* at the previous time step, $t - 1$.
- c_t is the *context vector* at time step, t. It is uniquely generated at each decoder step to generate a target word, y_t.
- h_i is an *annotation* that captures the information contained in the words forming the entire input sentence, $\{x_1, x_2, \dots, x_T\}$, with strong focus around the i-th word out of T total words.
- $\alpha_{t,i}$ is a *weight* value assigned to each annotation, h_i, at the current time step, t.
- $e_{t,i}$ is an *attention score* generated by an alignment model, $a(.)$, that scores how well s_{t-1} and h_i match.

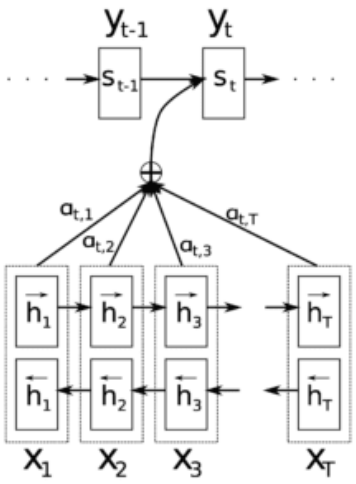

Fig 4: The Bahdanau Architecture

The Bahdanau Attention Algorithm

1. The encoder generates a set of annotations, h_i, from the input sentence.
2. These annotations are fed to an alignment model together with the previous hidden decoder state. The alignment model uses this information to generate the attention scores, $e_{t,i}$.
3. A softmax function is applied to the attention scores, effectively normalizing them into weight values, $\alpha_{t,i}$, in a range between 0 and 1.
4. These weights together with the previously computed annotations are used to generate a context vector, c_t, through a weighted sum of the annotations.
5. The context vector is fed to the decoder together with the previous hidden decoder state and the previous output, to compute the final output, y_t.
6. Steps 2-6 are repeated until the end of the sequence.

4.3. Evaluation Metrics

BLEU

The Bilingual Evaluation Understudy, or BLEU, is a score that is used to compare a candidate text translation against one or more reference translations. Although it was designed to evaluate text generated for translation, it can also be used to evaluate text generated for a variety of natural language processing applications.

The approach works by counting matching n-grams in the candidate translation to n-grams in the reference text, where 1-gram or unigram would be each token and a bigram comparison would be each word pair. The comparison is made regardless of word order. The counting of matching

n-grams is modified to ensure that it takes the occurrence of the words in the reference text into account, not rewarding a candidate translation that generates an abundance of reasonable words. The weights are specified as a tuple where each index refers to the gram order. The weightage can be tuned for the 1 to 4 gram model based on the results required. If we just want to work with a 1 gram model then weights would be (1,0,0,0).

5. Experimental Results & Model Evaluation

We get the following results with our models, After running for 5 epochs, training loss was decreasing but validation loss started increasing post epoch 4. So the model gives the best results on epoch 4 with the train loss 0.950, & test loss 6.378

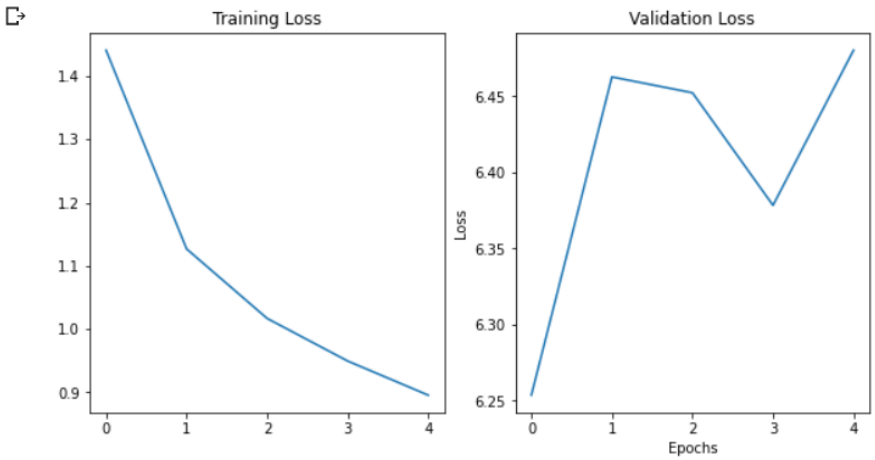

Fig 5 : Training and Validation loss

Then we switched to check the random image and compare the caption generated by the model with the actual caption. Below is the random image which shows a dog playing in the water. The parameters taken for BLEU for N-gram weightage is (0.15, 0.25, 0.40, 0.20)

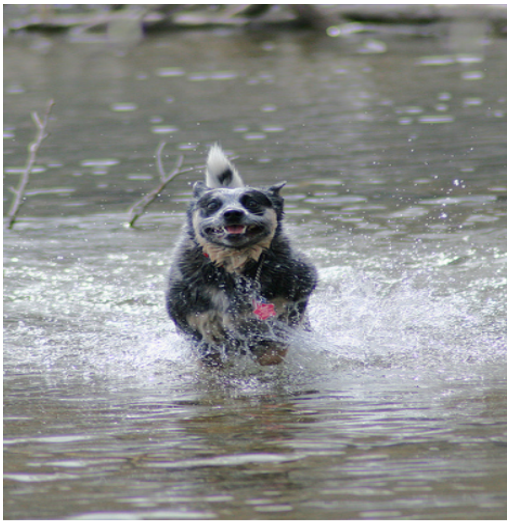

Fig 6 :Test Image

Actual Caption: <start> there is a black dog running in a shallow river <end>

Predicted Caption : <start> the black dog is wading in the water with park <end>

BLEU Score: 45.62

6. Applications

- This project can be used to caption an image to help blind people, so that they can also get the feel of the image. By converting the model output (caption) to an audio file.
- This project can serve a greater audience with multilingual caption generation. We converted the model output caption which was in English to Hindi.

  Output Caption : the black dog is wading in the water with park

  Hindi Translation : काला कुत्ता पार्क के साथ पानी में तैर रहा है

7. Conclusion and Discussion

Deep Learning is a burgeoning topic right now, with a slew of new applications being released every day. And getting hands-on with Deep Learning is the greatest approach to learn more about it. Image Captioning refers to the process of generating textual description from an image – based on the objects and actions in the image.This process has many potential applications in real life. A noteworthy one would be to save the captions of an image so that it can be retrieved easily at a later stage and can be converted into various other usable forms like audio /multilingual texts.Also, we could have tuned the hyperparameters more to get better results. Overall working on this project has helped us develop more interest in understanding the Deep Learning concepts and we would like to explore more on this project in future.

8. References

[1] Pradeep Miriyala “Image Captioning for Flickr8k Images” ,October 5, 2021. https://www.kaggle.com/pradeepmiriyala/image-captioning-for-flickr8k-images

[2] Sumit Gupta “Eye_For_Blind,” May 31, 2021. https://www.kaggle.com/sumitkg86/eye-for-blind

[3] Ankit Bvs “Automated-Image-Captioning-System: Automated Image Captioning System”, September 12, 2020 https://github.com/ankitbvs/Automated-Image-Captioning-System#references

[4] Roy “A Guide to Image Captioning”, December 9, 2020 https://towardsdatascience.com/a-guide-to-image-captioning-e9fd5517f350

[5] Jason Brownlee “A Gentle Introduction to Calculating the BLEU Score for Text in Python”, Nov 20,2017 https://machinelearningmastery.com/calculate-bleu-score-for-text-python/