National College of Ireland

Deep Learning for Image Caption Generation for the Visually Impaired

MSc Research Project
Data Analytics

Amit Vajpeyee
Student ID: x19218397

School of Computing
National College of Ireland

Supervisor: Dr. Abid Yaqoob

National College of Ireland
Project Submission Sheet
School of Computing

Student Name:	Amit Vajpeyee
Student ID:	x19218397
Programme:	Data Analytics
Year:	2022
Module:	MSc Research Project
Supervisor:	Dr. Abid Yaqoob
Submission Due Date:	15/12/2022
Project Title:	Deep Learning for Image Caption Generation for the Visually Impaired
Word Count:	6382
Page Count:	23

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature:	
Date:	1st February 2023

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST:

Attach a completed copy of this sheet to each project (including multiple copies).	☐
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	☐
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	☐

Assignments that are submitted to the Programme Coordinator office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Deep Learning for Image Caption Generation for the Visually Impaired

Amit Vajpeyee
x19218397

Abstract

Individuals with challenges related to vision have to deal with the pervasiveness of complete or partial unemployment amongst their peer group. World Health Organization (WHO) had estimated that people with headcount surpassing 2 billion are going through vision related impairments around the globe and approximately half of those are having moderate to serious visual problems. Scant studies have been attempted towards creation of employment for such people. Majority of them are restricted in scope and incorporate just a few aspects. This research has attempted to amalgamate the approaches of image-to-caption generation using the encoder (deep learning) – attention mechanism – decoder (natural language processing) architecture, caption-to-speech generation, and robotic process automation. This integrated ecosystem of diverse technical domains would allow the automatic download of the dataset, training and testing the architecture and then allow end user to use their voice to search keywords on Facebook.

1 Introduction

As per WHO estimates (Fricke et al.; 2018), 'Presbyopia' (far sightedness due to old age) and 'Myopia' (near-sightedness) were responsible for USD 269.4 billion worth of losses in productivity globally. On the individual level mental distress, worry, seclusion in the community take a huge toll psychologically. The daily routine activities that an average person takes for granted (like going from one place to another, preparing food and reading) are a difficult task for such people to carry out. Same logic can be applied to understand the reason behind their lower participation in the labour market. The faculty of sight plays a vital role in the execution of even the most basic tasks at work. Any disability in the sight can prove to be the biggest stumbling block in carrying out most of the blue- or white-collar work.

Individuals with severe sight related impairments are unable to visually observe, distinguish and admire the beauty embedded in the images. People upload a plentiful of information on social media including pictures. In 2016, Facebook had floated an accessibility project to aid such people in comprehending the information in the images uploaded by the members of their household or friends. If there was an image displaying a person with their pet dog, the Facebook app's speech output might have been like: "The given picture may contain a man walking in a park with his dog".

The ultimate goal of this paper is to use technology to find ways to generate jobs for people who are suffering from vision related issues. The approach can be broken down into following 6 broad domains:

1. deep learning (image-to-feature vector generation)
2. attention mechanism (feature vector-to-context vector generation)
3. natural language processing (context vector-to-caption generation as well as caption-to-speech generation)
4. robotic process automation
5. web scraping
6. application of marketing strategies

Robotic process automation was used to solve two purposes:

1. to automatically downloading the dataset
2. take speech input from the user and convert it in to the keyword, then log in to Facebook, and search the keyword in Facebook

Due to the lack of explicit approval from Meta (Facebook's parent company), last two steps were not implemented.

Prior to this paper, research work had focused only on a few aspects and none has tried the integration of aforementioned technologies and strategies, aimed at the job creation (especially for the people with visual disability). A lot of work with the goal of image to caption generation can be found in the literature (containing the encoder-attention-decoder, or more recently, transformer architecture - (Vaswani et al.; 2017)). A few papers have proposed solutions to aid the blind in steering their way from one geographical location to other.

Initial solutions for image-to-caption generation were based on the encoder-decoder architectures. (Bahdanau et al.; 2014) hypothesized the attention mechanism to overcome the inefficiencies in the encoder-decoder methodology. However, their work was based on the language translation (English to French). It was (Xu et al.; 2015) who explored the use of attention mechanism (along with encoder-decoder format) for image-to-caption generation. They used Flickr 8k, Flickr 30k and Microsoft Common Objects in COntext (COCO) datasets. This paper is inspired from their work and has used Flickr 8k dataset only. This paper has attempted to answer the following research question:

'How well can encoder-attention-decoder architecture generate image relevant captions?'

This can further be broken down in to the following research objectives:

1. Generation of image relevant captions.
2. Obtaining the best possible values of the evaluation metrics

In the traditional encoder-decoder mechanisms, entire feature map was fed to the decoder (usually Recurrent Neural Networks) at each timestamp (Xu et al.; 2015). For example, consider an image of a horse running in the racecourse. Suppose that at the current timestamp, the following caption has been generated: 'A'. The next word should be 'horse' but the decoder is being fed the entire feature map containing the encoding of the entire image. This usually results in increased computation time as unnecessary information needs to be decoded. Moreover, the output caption might not be efficient.

On the other hand, attention mechanism depicted in Figure 1 passes only the relevant spatial information in the form of context vector to the decoder. This overcomes the shortcomings of the traditional encoder-decoder models. Again considering the previous example, at the current timestamp, the caption generated is 'A'. At this moment, the context vector having the maximum attention weights to the spatial information of the horse (while remaining spatial information would have least attention weights) would be fed to the decoder. Thus, decoder will take less time to generate the caption at the next timestamp: 'A horse'.

We can consider it like this: in the traditional models, the spatial information of the feature vector or feature map were averaged together. Attention mechanism introduces a weight to spatial position as per its 'perceived' importance at that particular timestamp.

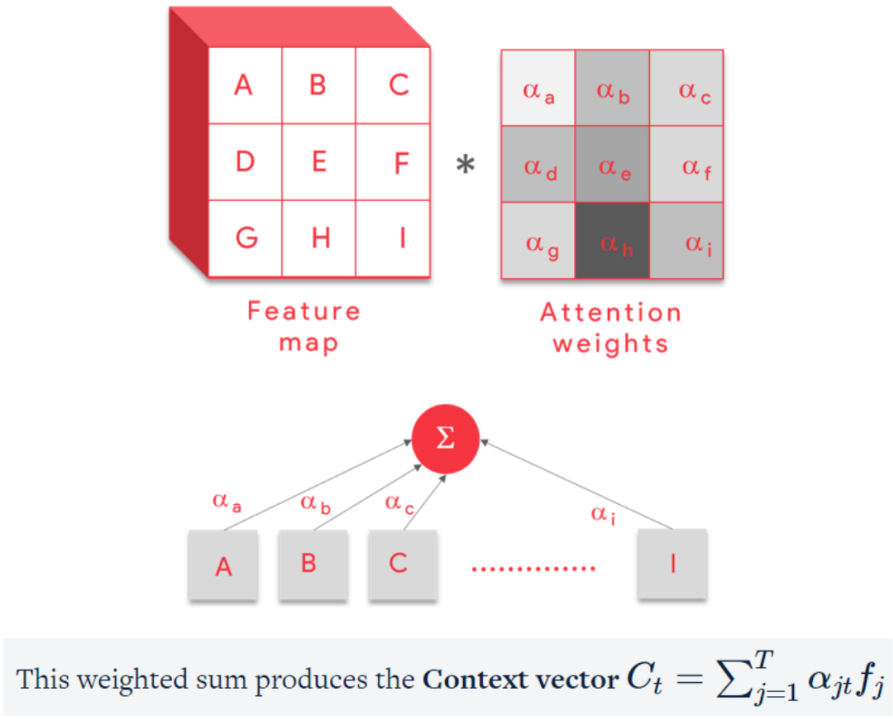

Figure 1: Working of the Attention Mechanism

2 Related Works

This section comprises the gauging of the work that has been conducted till date in the domain of caption generation from images. On collocating and contrasting the literature, a cogent way forward was established for the execution of this research. Primarily, the modeling for captioning the image can be segregated in to three categories: neural network based techniques, attention mechanism strategies and reinforcement learning based procedural methods. (Khan et al.; 2022) found that methodologies based on attention mechanism were more effective than rival techniques.

Literature review for the first two categories mentioned above was conducted and summarized in subsection 5.1 and subsection 2.2 respectively.

2.1 Neural Network-based Techniques

(Kiros et al.; 2014) used three different dataset. These dataset were Attribute Discovery which contains almost 40K images, SBU dataset which contains 400K images and lastly IAPR TC-12 dataset which contains 20K images. Convolutional Neural Network (CNN) was used for extracting image features.

(Vinyals et al.; 2015) utilized Recurrent Neural Networks (RNN's) as the encoders, author utilized CNN for extracting image features. Once CNN is done, RNN was utilized for caption generation. CNN is widely used for good performance in image recognition. CNN last layer is put into RNN for caption generation, for avoiding the issue of vanishing gradients or exploding, Long Short-Term Memory (LSTM) was used. On Pascal dataset, the BiLingual Evaluation Understudy - 1 (BLEU-1) (Papineni et al.; 2002) score achieved was 59. 66 score was achieved on Flicker8 dataset.

(Karpathy and Fei-Fei; 2015) made use of VGGNet as the encoder, it is one of many available CNN architectures. Encoder is VGGnet and the decoders is bi-directional RNN, Microsoft's Common Objects in COntext (COCO) [1] and Flickr8k dataset [2] were used in this research. Root Mean Squared Propagation (RMSProp) and Stochastic Gradient Descent (SGD) were used as the optimizers and then author found that the last-mentioned performed better.

2.2 Attention Mechanism Strategies

(Bahdanau et al.; 2014) and (Xu et al.; 2015) suggested using CNN as encoders and RNNs as decoders for generating the captions. The former introduced the attention mechanism and used it in translating languages: from English to French. The latter used attention mechanism to process the feature vector, which was gained from encoders. Flickr30K [3], Microsoft's COCO and Flickr8K datasets were used for model training.

(Liu et al.; 2017) asserted to be the pioneers to provide evidence of efficacity of the model hypothesized by (Xu et al.; 2015). Three models were used. The first one was supervised or implicit where, VGGNet-19 having the pre-trained weights of Imagenet was used, Optimizer Adam along with SGD (Stochastic Gradient Descent) was used. Dropouts were used to mitigate overfitting. The decoder of choice was LSTM. The second model was strong supervision, these models were used on Flickr30k dataset. The third model was weak supervision, these models were trained on MS COCO dataset. The BLEU-3 and 4 scores on Flickr30k and MS COCO datasets were 30.2, 21.0, 37.2 and 27.6 respectively.

Attention based encoder which is CNN and the decoder is RNN was postulated by (Xu et al.; 2022), (Alabduljabbar et al.; 2022) and (Khan et al.; 2022). The latter compared the performance of four models. Classifying the images was not the aim of the encoders. Instead, the aim was obtaining the feature vectors, which are then fed to the gated recurrent units (GRU's). These GRU's were used as the decoders. For training the models, MS COCO was utilized. As per this paper, with the Inception V3 which is pre-trained, highest BLEU-1,2,3 and the rouge scores were achieved. However, specific values were not mentioned.

(Alabduljabbar et al.; 2022) postulated that along with encoder-decoder, two atten-

[1] https://cocodataset.org/#download
[2] https://forms.illinois.edu/sec/1713398
[3] https://bryanplummer.com/Flickr30kEntities/

tion models should be tested, first one is for the visual aspect and second is for textual aspect. Flickr30k dataset was used, ResNet101 was applied as an encoder. Adamax was the optimizer, early stopping was also applied in Keras, the evaluation metric used was BLEU. There were four models which were used and then their performance was juxtaposed. In Model 1 was the preliminary model comprising of decoder (LSTM) and encoder (ResNet101). In model 2, model 1 was re-introduced with the visual attention. Model 3 comprised of Model 1 having visual and reflective attentions. The reflective attention had a focus on textual features.

(Xu et al.; 2022) pointed out that the constituents and architecture of CNN and RNN are different, thus the CNN-RNN union is heterogeneous, thereby leading to difficulty in end-to-end training of the overall model. Author proposed that the transformers has overcome the limitation. There is an issue that there was no mathematical/ statistical analysis for backing up their hypothesis.

(Xiao et al.; 2022) hypothesized using attentional LSTM. The said architecture changes the vector which is achieved by the encoder for accommodating the pertinent contextual attributes as well as the spatial ones. MS coco and Flickr30K dataset were made use of for this research.

(Jiang and Hu; 2022) proposed hybrid attention network. HAN utilized the customary approach of attention in conjunction with attention mechanism of another kind. In this approach, model tried to mirror the way a human brain discerns and processes during the phase of captioning. The overall BLEU 4 score achieved is 37.4 by using MS COCO dataset and Flick30k dataset.

(Wang and Gu; 2022) disregarded the traditional mechanisms and did not take any consideration of the two issues which are feature vector noise and secondly, the predictions obtain dissimilar spatial attributes in the channel and image attributes. DBDAF mechanism was proposed in overcoming this issue. The overall scores received on COCO dataset were BLEU-1 - 81.2, BLEU-2 - 65, BLEU-3-50.2, BLEU-4 - 38.9, Semantic Propositional Image Caption Evaluation (SPICE) (Anderson et al.; 2016) - 22.3 and Metric for Evaluation of Translation with Explicit Ordering (METEOR) (Banerjee and Lavie; 2005) - 28.7.

(Popattia et al.; 2022) suggested guided attention method. Following scores were achieved BLEU-4- 34.9, BLEU-3- 467.1, BLEU-2-62.2, BLEU-1- 76.0. When author did the some more testing, author achieved a better result in BLEU-4 scored as compared then HAN however, the results were less than AoA-Net architecture. Author also didn't use the transformer approach.

Wang et al. (2021) used the models which have the attention architecture as foundation, each step is independent of another. It disregarded the information which is gathered in the previous steps. Authors proposed a dynamic attention model for overcoming the above situation. They archived the scores with the help of 'Karpathy Test-split' by using the coco dataset. BLEU-4: 41.1, METEOR: 28.5, Recall-Oriented Understudy for Gisting Evaluation (ROGUE) (Lin; 2005): 58.2 was achieved. These models were prone on the biases which is in the dataset.

(Zhang et al.; 2021) used the traditional attention model which uses focal point on the basis of specific are which is visual, connections were not focus and the interplay which is available in between the areas of visual. There is no arrangement inspect for expressions or words which are relevant. This will lead to inappropriate captioning. In many individual areas, the context which is embedding based was introduced. With the loss-beam of Consensus-based Image De- scription Evaluation 'CIDEr-5' (Vedantam

et al.; 2015), author was able to achieve: BLEU-1: 81.4, BLEU-2: 62.4, BLEU-3: 46.9, BLEU-4: 36.5, METEOR: 24.5, ROUGE-L: 62.6, SPICE: 21.1.

(Yan et al.; 2021) - Usage of feature vector along with attention mechanism. While the words are generating, these are not related to the contents which are visual like is, himself and to. Misleading of the words can take place in this case. Another mechanism was used by the author is task adaptive. this mechanism was proposed to remove these issues by making new vectors which are task adaptive. For generating non-visual words, there was a use of attended in the decoders. They COCO Karpathy test split was achived by the author: BLEU1: 72.0, BLEU-4: 24.7, METEOR: 25.8, ROUGE-L: 52.1, CIDER-D: 92.2 and SPICE: 19.3.

(Shrimal and Chakraborty; 2020) postulated beam search in the process of generating the captions. LSTM is the encoder and ResNet-101 was used as the encoder, author also used attention methods for modeling. Author used Flickr8k dataset and following results were achieved - BLEU-1: 62, BLEU-2: 42.8, BLEU-3: 34.3, BLEU-4: 21.3, METEOR:22, CIDEr: 51.3 and ROUGE-L: 42.1.

(Shambharkar et al.; 2021) utilized beam search and also achieved argmax mechanisms for predictions. Authors used LSTM for troubleshooting the issue of vanishing the gradients when the below one is the weight's value of the parameter. This problem will lead to improper training of the model, the captions generate are not very good. There is the problem of overfitting. RNNs were used for replacing LSTM where the value is one. Authors accomplished the these scores with k set to 3: BLEU-1: 0.61, BLEU-2: 0.344, BLEU-3: 0.245, BLEU-4: 0.123.

2.3 Concluding Statements on Related Work

A comparison of techniques and methodologies hypothesized in the literature pointed towards a host of solutions for the research question of this paper. Images were processed by CNN's acting as encoders. Outputs were feature vectors or feature maps. A vast majority used the tried and tested transfer learning architectures like VGGNet, ResNet, InceptionV3. Two main types of RNN's, viz., GRU's and LSTM's were implemented to tackle vanishing as well as exploding gradients. Several variants of attention mechanism were suggested to effectively tune the feature maps according to the captions generated at a relevant time-stamp. In order to train the model as best as possible, 'teacher forcing' was deployed in some of the related work, while others used some features of Keras like 'Early Stopping' to put an end to further epochs if the performance of the model did not improve for a certain number of successive epochs. Most common metric of choice for evaluation was BLEU.

In this work, InceptionV3 (Yang et al.; 2016) architecture was utilized with the pre-trained 'Imagenet' weights (Ding et al.; 2016) as the base model. Further customizations were done in order to meet the requirements of research question. This has been elaborated in sections 3.4 and 5.

3 Methodology

A host of methodologies were juxtaposed to be utilized on the Flickr 8k dataset to answer the business related issues. Moreover, a perusal of a number of related work who have used the benchmark Flickr 8k dataset, the respective authors have used two types of architectures primarily: encoder-attention-decoder and more recently, transformers. This

paper has used the encoder-attention-decoder architecture, CRISP-DM framework and RPA (Robotic Process Automation).

CRISP-DM (Cross Industry Standard Process for Data Mining) framework furnishes a standard design to execute a data mining project. That design can be decomposed in to 6 prominent stages as depicted in Figure 2: Enterprise comprehension, data insights, fashioning or re-fashioning of data, modeling, assessment and analysis, and, deployment. (Palacios et al.; 2017)

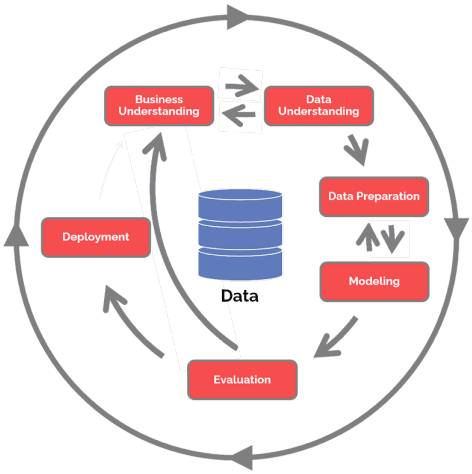

Figure 2: CRISP-DM Framework

The phases of CRISP-DM framework in context of this paper have been elaborated as follows:

3.1 Business Understanding

In the business context, this stage provides a cogent information of what needs to be achieved and how it can be accomplished. One needs to assess the current circumstances and resources at hand and accumulating or acquiring resources in pursuit of the business objectives. According to the research question specified in section 1, to develop and deploy an efficient model that can explain the contents of an image. In this work, an attempt has been made to expand this business objective further. RPA can be used to log in to the social media account of the business, use speech-to-text to generate the keyword and search for the keyword on the social media). Author of this paper conjectures that if web scraping (in compliance with ethics, GDPR and Data Protection Act) and appropriate marketing strategies are used on the social media (especially in the public pages like marketplace), the proposed solution can help in creating jobs for the individuals who are suffering from vision related issues.

3.2 Data Understanding

This stage of CRISP-DM necessitates accumulation of data, its verification for completeness and alignment with the enterprise needs by conforming to the requisite quality.

First of all, data needs to be downloaded, unzipped, unnecessary data should be deleted, remaining data should be manipulated to be in the correct format to be processed

further. All of this was done with the help of RPA. Main tools used were PyCharm Community edition, Google Chrome, Selenium library. The original website is no longer hosting the Flickr 8k dataset at this time. It is available at no cost from the website of 'University of Illinois Urbana-Champaign' [4]. However, one needs to fill the form and then wait for the download link to be received via email. Alternatively, the dataset can be downloaded from Github [5] also.

Once the initial pre-processing is complete, the dataset contains one folder and one text file. The folder contains 8091 images and the text files contains the names of individual images in the folder and 5 relevant captions for each image. The dimensions of images in terms of pixels are not fixed. Total size of this dataset is 1.04 GB. A sample of images can be viewed in Figure 3.

Figure 3: Specimen of Images along with the relevant captions from Flickr 8k Dataset

3.3 Data Preparation

It is the stage of CRISP-DM which consumes a lot of time, in terms of manual efforts required to clean and prepare the input pipelines that feed data in the correct format to the model.

When talking about this study, the dataset contains two different types of data: images and captions. They need to be pre-processed separately. While pre-processing captions, following need to be generated: vocabulary, wordtoindex and indextoword mappings. While pre-processing images, images need to be converted in to batches of tensors

[4]https://forms.illinois.edu/sec/1713398
[5]https://github.com/goodwillyoga/Flickr8k_dataset

so that then can be fed to the base model based on transfer learning. But before it is done, the values of each pixel in each image must first be rescaled between the values -1 and +1, and and each image must also be resized to the shape (299, 299, 3). A custom function 'load_image' was used to serve this purpose. Code for the same can be perused from Figure 4. This is required by the InceptioV3 model in Keras and has been elaborated in section 5.

```
# Typing the code here for creating the function.
# This function should return images & their path

def load_image(image_path):
    #write your pre-processing steps here

    # Read the image:
    given_img = tf.io.read_file(image_path)
    given_img = tf.image.decode_jpeg(given_img, channels = 3)

    # To resize the image in to the shape of (299, 299):
    # given_img = tf.keras.layers.Resizing(299, 299)(given_img)
    given_img = tf.image.resize(given_img, (299, 299))

    # To scale the input pixel values between -1 and 1/ To Normalize
    # the image within the range of -1 to 1, such that it is
    # in correct format for InceptionV3:
    given_img = tf.keras.applications.inception_v3.preprocess_input(given_img)

    return given_img, image_path
```

Figure 4: Code for the custom function load_image

3.4 Modeling

Once the data from the input pipeline is ready, it needs to be processed with the help of an appropriate model. This phase of Crisp-DM is further divided in to 4 jobs: 1.) Selection of the modeling approach, 2.) Splitting the raw data or as in this study, the data from the input pipeline in to train and test datasets.

Also, while designing such customized models, one must remember that images are fed in batches and Flickr 8k dataset has 5 captions for each image. Thus, the output shape at each step must carefully be designed to avoid errors.

3.4.1 Encoder

One of the major learnings from the literature is that one needs to use CNN's to process the images and RNN's to process text. InceptionV3 is widely used transfer learning model to process images of Flickr 8k dataset. However, the result of this model is needed to be fed in to the attention mechanism. Since the goal is not to classify the images, the last layer (Softmax layer) of the InceptionV3 model needs to be removed and the output from its penultimate layer (feature map) should act as an input to the attention mechanism. Thus, model subclassing needs to be used to design custom encoder.

3.4.2 Attention Mechanism

In this research, it was found that the feature map's shape was not in accordance with the shape required for the default attention model in Keras. Thus, model subclassing was needed to be used to design custom attention mechanism. It takes two inputs: The feature

map from the encoder and the hidden state of the RNN (decoder) at that particular time stamp. The yield of the attention mechanism is termed as the 'Context vector'. It needs to be concatenated with the embedding vector obtained from the raw captions.

3.4.3 Decoder

The concatenated vector is then fed to the custom decoder (GRU were used in this research) along with decoder's state in the previous timestamp. The preliminary condition of decoder is a set of zeroes. There are two outputs of decoder: 1.) generated caption at that timestamp, and 2.) hidden state at the timestamp.

3.5 Evaluation

This phase involves the assessment of the predicted captions. Since this is a part of natural language processing, traditional evaluation metrics like accuracy, F1 score, sensitivity, specificity cannot be used. In the literature, evaluation metrics like BLEU, CIDEr, ROUGE and SPICE have been used. For this research, BLEU score was the only metric used. Predictions have been made with 'Greedy search' and 'Beam Search'.

3.6 Deployment

Once the model is finalized, it needs to be saved and deployed so that the target users can start using it to meet the business needs for which it was developed. For this research, the model was deployed on the local device (laptop) as can be viewed in Figure 5. User needs to manually enter the test image and click 'upload'. It takes a few seconds to generate the caption in text. If the user wants to listen the caption, he or she can press the play button on the web page.

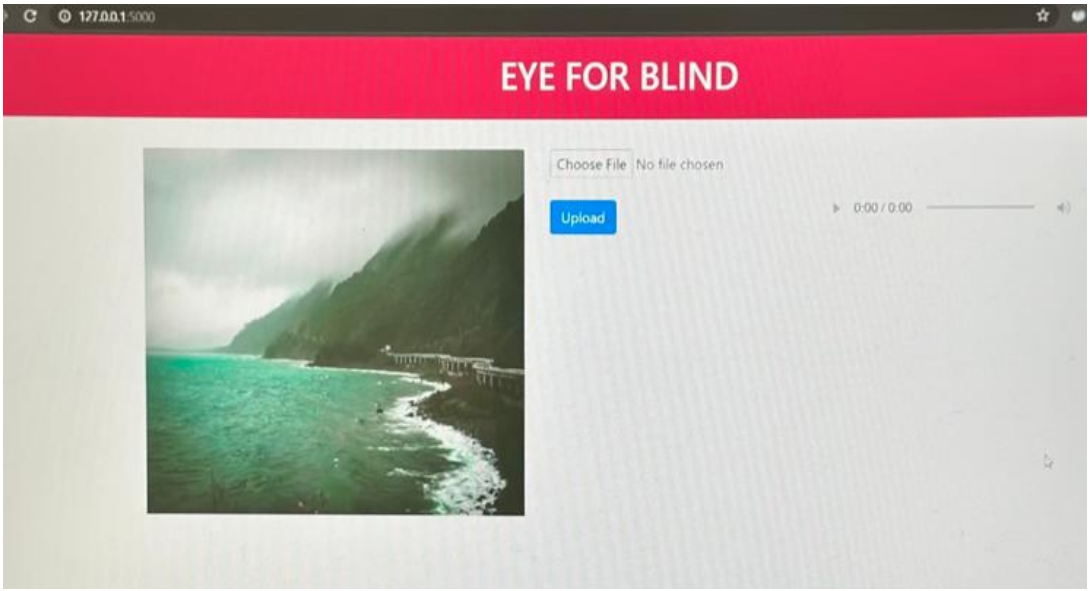

Figure 5: Deployed Model on the Local Host

4 Design Specification

A carefully crafted architecture comprising of image caption generation mechanism and robotic process automation was fabricated keeping in mind the need to find the solution of the research question specified in section 1. Its layout is presented in Figure 6.

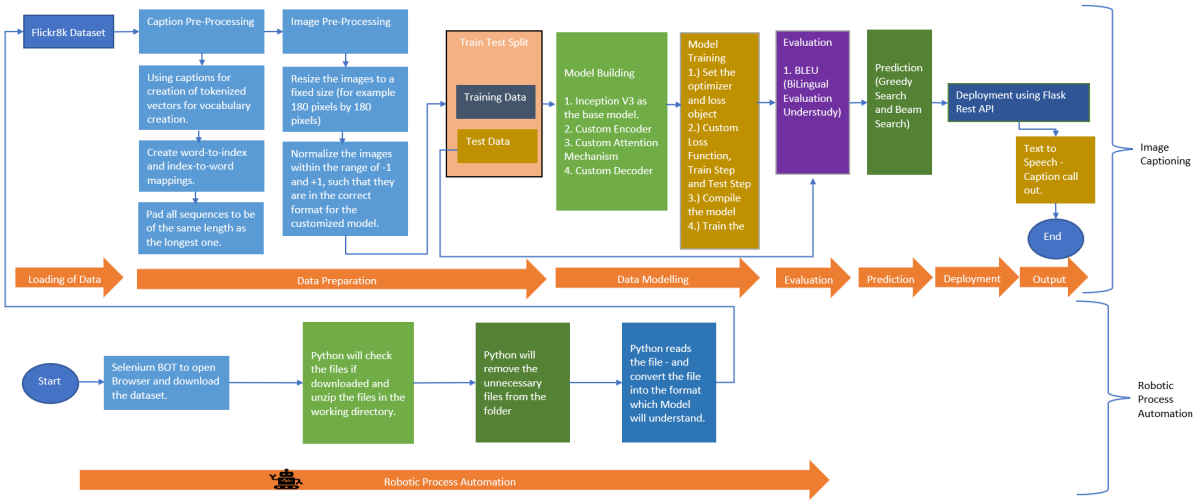

Figure 6: Design Specification for Image to Caption and Speech Generation

Although the code has been further stretched to allow users to make a speech query on his or her Facebook profile. Another RPA model converts the speech in to written keyword that is searched. But since there was no written approval from Facebook, web scraping was not attempted for reasons related to ethics.

Selenium BOT and Python were used to download, unzip and manipulate the dataset. In order to create the input pipeline for the base model, pre-processing was carried out separately on captions and images. Traditionally accepted 80:20 ratio was chosen to split the data in to train and test datasets.

Inception V3 was used as the base model and the encoder-attention-decoder architecture was built using model subclassing in Keras. Even the loss function, training step, testing step and average loss calculation function on the test dataset were customized. BLEU was used as evaluation metric and predictions were made with the greedy search and beam search. Once satisfactory results were obtained, the model was saved and deployed using Flask Rest API and tested using another RPA (Robotic Process Automation) based code on the website of Facebook.

5 Implementation

This section describes how the design specification was implemented in order to achieve the desired outcome as per the research question, in a step-by-step manner.

5.1 Data Download and Manipulation Using RPA and Python

As indicated in section 3.2, the dataset was downloaded from Github [6] [7] using RPA. The raw dataset contains two zipped folders. The zipped file for text contains a folder for MAC OS and 8 text files. Its size is 2.23 MB. Out of those, only the text file with the name 'Flickr8k.lemma.token' is useful. But its contents are not in the desirable format. The zipped file for images contains a folder of 8091 images. Its size is 1 GB. Python was then used to

1. Unzip and move the folders and files to the working directory
2. Delete unnecessary files
3. Make necessary changes in the 'Flickr8k.lemma.token.txt' file
4. Make a copy of this text file, rename that copy as 'captions.txt'
5. Delete the original 'Flickr8k.lemma.token.txt' file

5.2 Caption Pre-Processing

Following steps were performed:

1. Image captions were used to create tokenized vectors for vocabulary creation.
2. Punctuation marks were removed and vocabulary (of 5,000 words) was again created.
3. Wordtoindex and indextoword mappings were created.
4. Length of the longest sequence was calculated.
5. Remaining sequences were padded to match the length of the longest sequence.

5.3 Image Pre-Processing

As explained in section 3.3, a custom function 'load_image' was used

1. Resize the images to the fixed size of (299, 299, 3) as required by the base model - InceptionV3.
2. Normalize the values of individual pixels in all images within the range -1 and +1, again required by the base model.

5.4 Creation of Input Pipeline for the InceptionV3 Model

Since Keras has InceptionV3 architecture based model defined already, it was adopted in this work as a base model. However, it does not accept the images and captions in the form in section 5.2. Keras-based InceptionV3 model takes EagerTensor datatype as input. In order to obtain such a data type from the Numpy ndarray data type, an input pipeline needs to be created. Figure 7 depicts the flow of the input pipeline.

[6] https://github.com/jbrownlee/Datasets/releases/download/Flickr8k/Flickr8k_text.zip
[7] https://github.com/jbrownlee/Datasets/releases/download/Flickr8k/Flickr8k_Dataset.zip

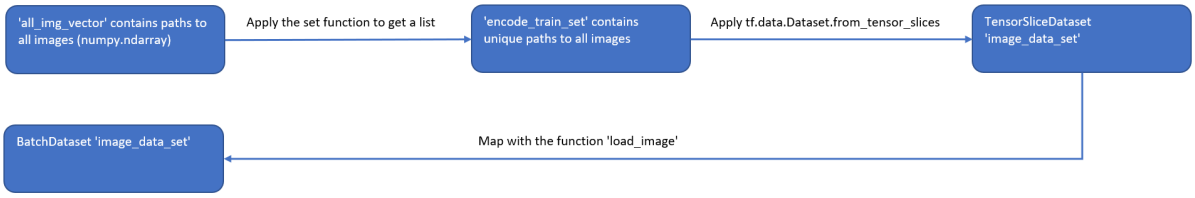

Figure 7: Flow of the input pipeline for Keras InceptionV3 Model

First of all, Numpy ndarray object is created to accommodate paths to all the images in the dataset. Each element inside this object is of string datatype. Since there are 5 captions for each image, set function is applied to get a list object containing the unique paths to all the images. Each element inside this object is of string datatype as well. Then the Keras function from_tensor_slices is used to obtain an object of datatype TensorSliceDataset. This object is then mapped with the custom function load_image to obtain an object of BatchDataset type. The function 'load_image' returns two components: images in the form of EagerTensor data type and image paths in the form of Tensor data type.

The shapes of all but one images EagerTensor are (256, 299, 299, 3). The last image EagerTensor's shape is (155, 299, 299, 3). It is because total number of images are 8091 and the batch size is 256. Dividing 8091 by 256 yields a remainder of 155. Similarly, the shape of all but one Tensor for image paths are 256 and that of the last Tensor for image paths is 155. Each element of the tensor for image paths is of the data type byte objects.

5.5 Data Processing via the Base Model

InceptionV3 model is instantiated from Keras. An empty dictionary was created to extract features and image paths. The EagerTensor is fed to this model in batches. The output was reshaped to (batch_size, 8x8, 2048) that is (256, 64, 2048). And the Tensor containing the image paths was first decoded so that individual image paths can be transformed from byte object data type to numpy objects. The dictionary for features and image paths is then populated with both the types of outputs for the entire dataset.

5.6 Train-Test Split on the Image Path and Captions

The numpy ndarray containing the paths to all the images and the caption vector were split in to 80:20 ratio with the seed value of 42. Overall, the raw dataset contains 8,091 images and 40,455 captions (5 captions for each image). After the train-test split, the size of the train list for images is 34,364 and the size of the test list for images is 8,091. If one adds these two values (that is 32,364 + 8,091) he or she would obtain 40,455.

5.7 Dataset Creation and Train-Test Split

The dictionary obtained from section 5.5 does not contain the captions but the unique paths to all the images. A custom function defined to map the image paths to their features.

Another custom function was built to generate the dataset. Since the maximum size of the data is 32,364, the buffer size was chosen to be 34,365. The function from_tensor_slices

was applied on the numpy ndarrays of images and captions to obtain the train and test datasets. The first element of the output datasets is of the datatype Tensor and is of the shape (256, 64, 2048) except the last one, whose shape is (108, 64, 2048). It is because, maximum size of train data is 32,364 and the batch size is 256. So, after creating 126 batches, the remaining quantity of images in the last batch were 32,364 - (256 x 126) = 32,364 - 32,256 = 108. Each individual component is of the data type EagerTensor.

The second element of the output datasets is of the datatype Tensor and of the shape (256,39) except the last one, whose shape is (108, 39). 39 is the maximum length of the captions. Each individual component of this element is also of the data type EagerTensor.

5.8 Customized Modeling Using Model Sub-classing

Following parameters were set before the customized encoder, attention model and decoder were built using the model sub-classing:

1. Embedding dimensions (quantity of neurons contained in the dense layer) were set to be 256
2. Units (number of neurons) were set to be 512
3. Vocabulary size was set to be 5,001 (Top 5,000 words in the captions + 1)
4. Number of steps in training was set to be equal to the quantity of images in the train set (that is 32,346) divided by the batch size (that is 256). Thus quantity of steps in training was equal to 126 (remainder was discarded).
5. Number of steps in the testing was set to be equal to the quantity of images contained in the test set (that is 8,091) divided by the batch size (that is 256). Thus quantity of steps in training was equal to 31 (remainder was discarded).

5.8.1 Customized Encoder

Since the goal is not the classification of the images, the last layer of the superclass model in Keras (that is, the Softmax layer) was needed to be removed. Moreover, the parameters defined in 5.7 were also needed to be incorporated. A custom function was defined with these goals in mind for the encoder. The input was of the shape (256, 64, 2048). The shape of output is (batch size, 8 x 8, embedding dimensions), that is (256, 64, 256). Once the custom encoder function was ready, it was instantiated.

5.8.2 Customized Attention Model

The output of the encoder needs to be fed to the attention model. Attention model also takes an input in the form of hidden state of the GRU at that timestamp. There are two outputs from this custom attention model: attention weights with the shape (batch size, 8 x 8, 1) that is (256, 64, 1), and the context vector with the shape (batch size, embedding dimensions) that is (256, 256). The context vector needs to be concatenated with the embedding vector obtained from the captions before being fed to the decoder (GRU).

5.8.3 Customized Decoder

Decoder takes attention weights, context vector and previous timestamp's hidden state. By default, the decoder tends to allocate importance to the corresponding token identifiers, inclusive of the zero values which were stuffed in the execution of padding process described in point number 5 of section 5.2. These zeroes must be removed during the loss calculation otherwise the model will incur an additional penalty, which would lead to erroneous calculations down the line. This necessitates the inclusion of mask in decoder for mitigation of the added penalty.

Attention model has been instantiated in the decoder. There are three outputs of the decoder:

1. Output of shape (batch size, 1, embedding dimensions) or, (256, 1, 512).
2. Hidden State of the shape (batch size, embedding dimensions) or, (256, 512).
3. Attention weights of the size (256, 64, 1) as described in section 5.8.2.

Once the decoder is instantiated with the embedding dimensions (256), units (512) and vocabulary size (5,001), the final output's shape becomes (256, 5001). This final output is the captions that are generated with each timestamp until the decoder encounters the end token.

A sample image of the overall model used in this work can be seen in Figure 8.

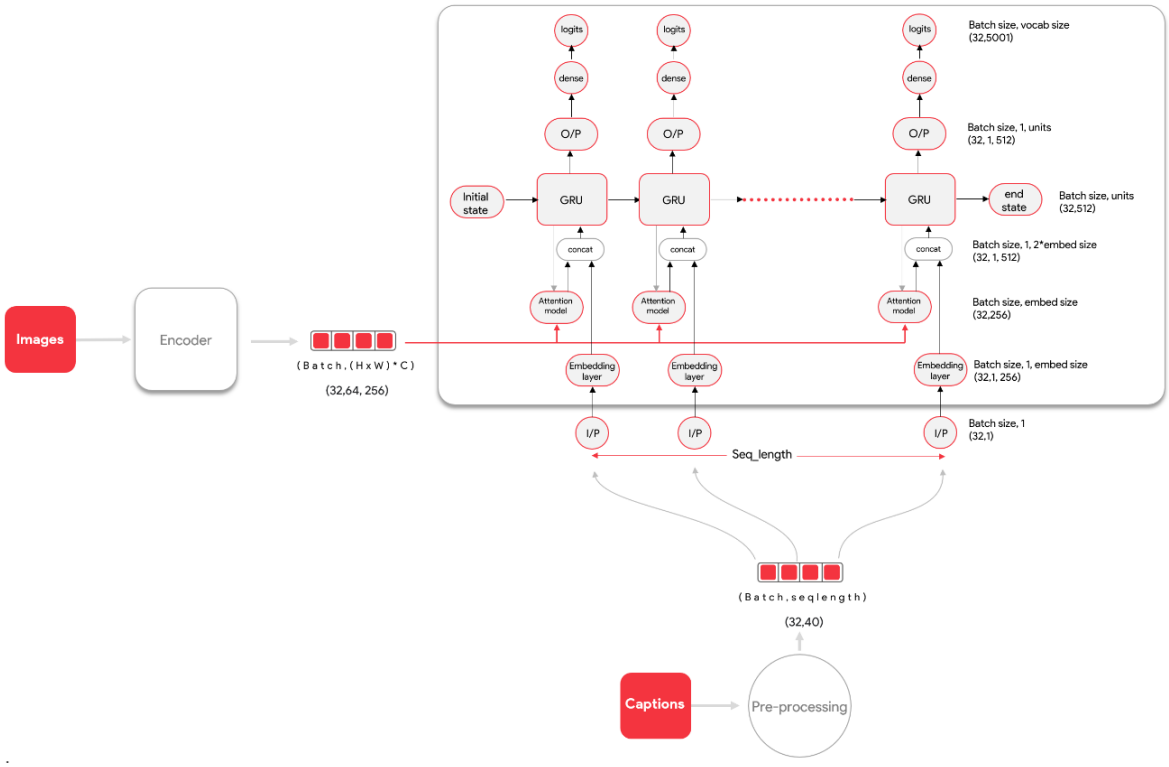

Figure 8: Sample Encoder-Attention-Decoder used in this work

The only difference in this sample model and the one used in this study is the batch sizes of 32 and 256 respectively. Also, the shape of caption vector in this study is (256, 39).

5.9 Model Training and Optimization

Optimizer was set to be ADAptive Moment estimation (ADAM) (Kingma and Ba; 2014) and loss object was defined with SparseCategoricalCrossentropy function of Keras with the value of the parameter from_logits set to be true. This true value signifies the values of loss are not transformed in to set of data within a given range of scale. This needs to be used in this study as the encoder does not have the Softmax function. Softmax pushes the output (probabilities) to be in the range 0 and +1.

The custom loss function needs to take in to account the mask in the decoder. Setting the value of loss to be equal to loss x mask resets the additional penalties back to 0.

Keras Checkpoints were used to save the final model so that it can be deployed at a later stage.

Train step was customized to accommodate the customized encoder, attention model, decoder and loss function. In order to predict the first word, decoder requires the output of the encoder (Feature map) and start token. After the generation of the first word of the caption, teacher forcing is used to pass the corresponding word from the corresponding real training caption at that time-stamp. This is required for model to be trained as desired.

Custom test step was also defined on the same lines as that of the custom train step. But this step does not include teacher forcing as this step is needed to evaluate the model. It would generate a list of predicted captions and corresponding scores at each time-stamp. Argmax function was used to extract the predicted caption with the highest score at that time stamp.

Training during the epochs numbered 48, 49 and 50 are depicted in Figure 9:

```
96%|█████████ | 48/50 [30:01<01:12, 36.06s/it]

For epoch: 48, the train loss is 0.244, & test loss is 3.551
Time taken for 1 epoch 36.05985450744629 sec

98%|█████████▊ | 49/50 [30:37<00:36, 36.06s/it]

For epoch: 49, the train loss is 0.238, & test loss is 3.684
Time taken for 1 epoch 36.05583739280701 sec

100%|██████████| 50/50 [31:13<00:00, 37.48s/it]

For epoch: 50, the train loss is 0.234, & test loss is 3.633
Time taken for 1 epoch 36.07086133956909 sec
```

Figure 9: Model training and calculation of the losses in Epochs 48, 49 and 50

Once all the relevant custom functions have been defined, the train dataset was subjected to the train and test steps with count of epochs set to 50. Once the model was trained, average test losses were calculated on the test dataset.

6 Evaluation

Figure 10 depicts the plot of train (blue) versus validation (orange) losses. It can be observed that while the train losses are coming down with each epoch, the same is opposite in the case of test losses. This does not mean that the model is over-fitting as the methodology undertaken in the train step was of teacher forcing while it was not present during the test step.

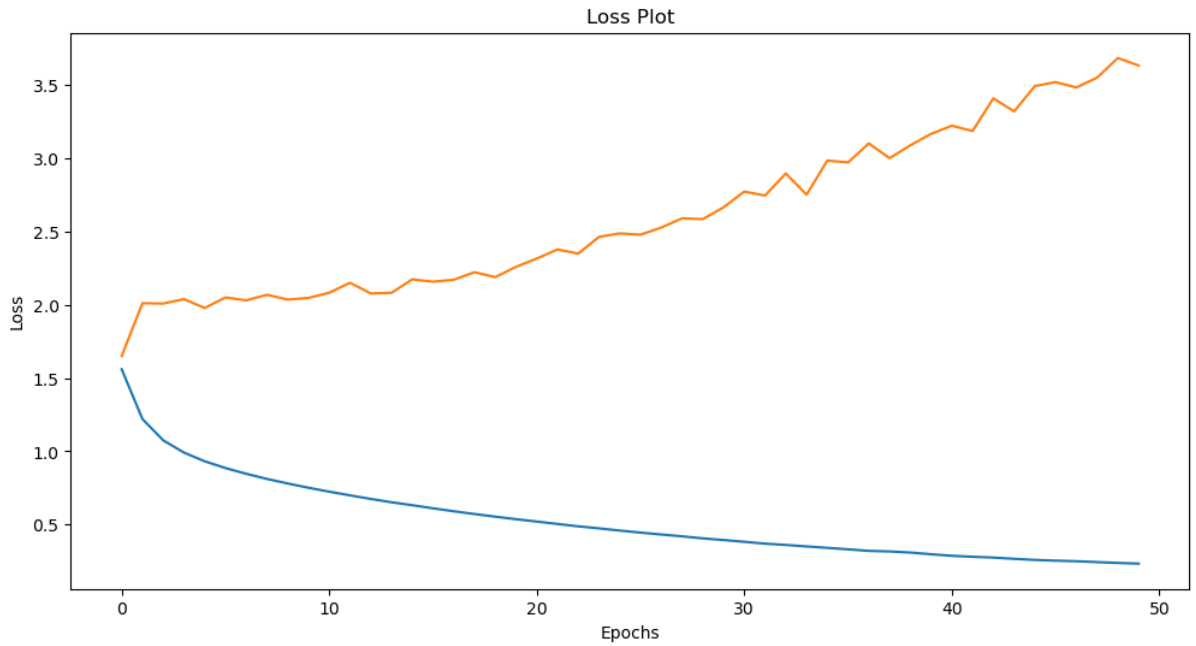

Figure 10: Train loss (blue) versus test loss (orange)

Greedy search (Klerk et al.; 2021) and Beam search (Lowerre; 1976) were used to make the predictions. Two images from the test dataset were tried: Figure 11 and Figure 12.

To give a better context against the predicted output captions, only 1 real caption (out of the total of 5 each) has been stated below for test images 1 and 2 respectively:

1. a man holds a trophy on the stage
2. some men try to load what looks like a cannon

The output of the greedy search on the two test images as can be observed from Figure 13 and Figure 14 respectively.

The output of the beam search were:

1. a racer displaying his trophy and waving
2. a group of men watch

The predicted captions of beam search were better in the test 1 as the count of epochs for training were 50 while the count of epochs for training were only 15 for the test 2.

Ideally, the range of beam search is between 0 and 1. Former means that the prediction is not good while the latter means that the prediction is perfect.

The values of BLEU score on test 1 and test 2 images are as follows, respectively:

Figure 11: Test Image 1

Figure 12: Test Image 2

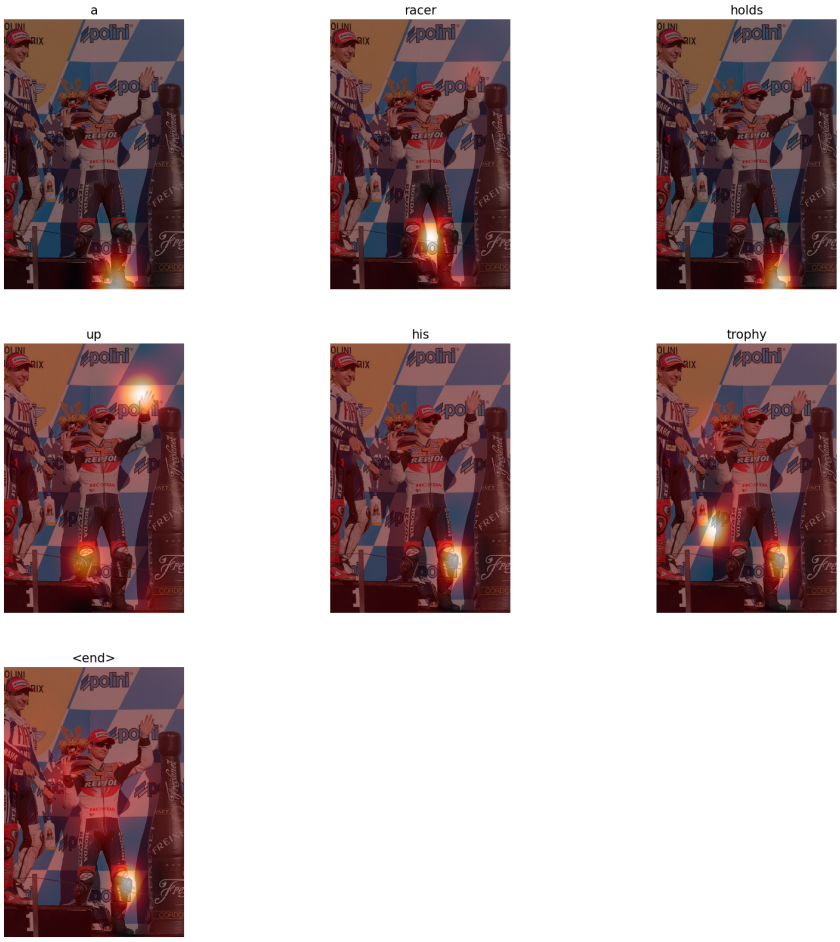

Figure 13: Output of Greedy Search on Test Image 1

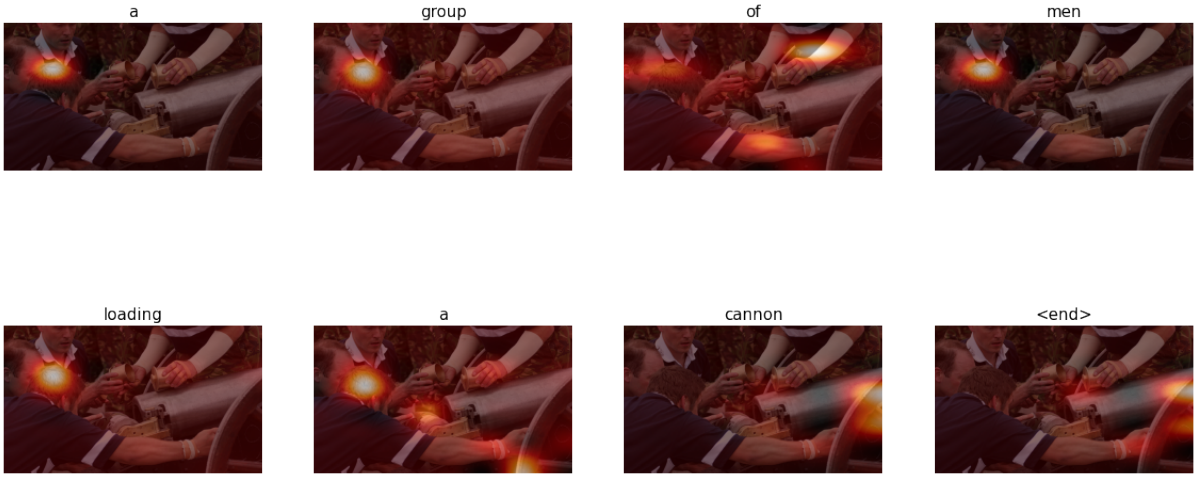

Figure 14: Output of Greedy Search on Test Image 2

1. 1.0977058971259434e-229

2. 7.558611808210718e-184

While the predicted captions were good, especially after training the model for 50 epochs, from the human perspective, the Bleu score were negligible (close to 0).

6.1 Discussion

While the related works have used the n-gram versions of the BLEU scores, this work has used the plain BLEU score. Thus, direct comparison is not possible. However, it can safely be concluded that the resulting BLEU scores in this work need a lot of refinement in order to be closer to the value of 1. This may be due to the four weights assigned during the calculation of the BLEU scores were all equal to 0.25 (all the four weights must add up to 1).

7 Conclusion and Future Work

Restating the research question:

'How well can encoder-attention-decoder architecture generate image relevant captions?'

Restating the research objectives:

1. Generation of image relevant captions.

2. Obtaining the best possible values of the evaluation metrics

7.1 Conclusion

The work was designed and implemented in the context of research question and research objectives. The predictions of the greedy and beam searches improved with the increase in the number of epochs of model training. This was in congruence with the first objective.

However, the resulting evaluation metric (BLEU Score) were of the magnitude close to zero. This was not in the expected lines and needs to be further investigated and improvised. Thereby it can be concluded that this study has failed to meet the expectations in the second objective.

The overall consequence of this study is that it cannot be commercially implemented till the values of BLEU score achieve the values close to 1.

This work was restricted to the evaluation metric of BLEU Score. However, usage of evaluation metrics like CIDEr, ROUGE and SPICE could have pointed to another outcome.

7.2 Future Work

The base model was chosen to be InceptionV3 and the decoder of choice was GRU. In future DenseNet will be used along with Bidirectional LSTM (Lu et al.; 2021) .

This work used Adam optimizer with loss object of SparseCategoricalCrossentropty defined in Keras. Changing how the optimization takes place can significantly change the overall outcome. Future improvisation would entail the incorporation of these. Moreover,

deep learning models tend to need a large amount of training (Chilimbi et al.; 2014). Thus, this study will be extended to implement on much larger benchmark datasets like Flickr 30k and MS COCO.

Lastly, n-gram BLEU Score along with the remaining benchmark evaluation metrics (CIDEr, ROUGE and SPICE) will be implement to test the overall efficacy.

References

Alabduljabbar, G., Benhidour, H. and Kerrache, S. (2022). Image captioning based on feature refinement and reflective decoding, *arXiv preprint arXiv:2206.07986* .

Anderson, P., Fernando, B., Johnson, M. and Gould, S. (2016). Spice: Semantic propositional image caption evaluation, *European conference on computer vision*, Springer, pp. 382–398.

Bahdanau, D., Cho, K. and Bengio, Y. (2014). Neural machine translation by jointly learning to align and translate, *arXiv preprint arXiv:1409.0473* .

Banerjee, S. and Lavie, A. (2005). Meteor: An automatic metric for mt evaluation with improved correlation with human judgments, *Proceedings of the acl workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization*, pp. 65–72.

Chilimbi, T., Suzue, Y., Apacible, J. and Kalyanaraman, K. (2014). Project adam: Building an efficient and scalable deep learning training system, *11th USENIX symposium on operating systems design and implementation (OSDI 14)*, pp. 571–582.

Ding, N., Goodman, S., Sha, F. and Soricut, R. (2016). Understanding image and text simultaneously: a dual vision-language machine comprehension task, *arXiv preprint arXiv:1612.07833* .

Fricke, T. R., Tahhan, N., Resnikoff, S., Papas, E., Burnett, A., Ho, S. M., Naduvilath, T. and Naidoo, K. S. (2018). Global prevalence of presbyopia and vision impairment from uncorrected presbyopia: systematic review, meta-analysis, and modelling, *Ophthalmology* **125**(10): 1492–1499.

Jiang, W. and Hu, H. (2022). Hadamard product perceptron attention for image captioning, *Neural Processing Letters* pp. 1–18.

Karpathy, A. and Fei-Fei, L. (2015). Deep visual-semantic alignments for generating image descriptions, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Khan, R., Islam, M. S., Kanwal, K., Iqbal, M., Hossain, M., Ye, Z. et al. (2022). A deep neural framework for image caption generation using gru-based attention mechanism, *arXiv preprint arXiv:2203.01594* .

Kingma, D. P. and Ba, J. (2014). Adam: A method for stochastic optimization, *arXiv preprint arXiv:1412.6980* .

Kiros, R., Salakhutdinov, R. and Zemel, R. (2014). Multimodal neural language models, *International conference on machine learning*, PMLR, pp. 595–603.

Klerk, W. J., Kanning, W., Kok, M. and Wolfert, R. (2021). Optimal planning of flood defence system reinforcements using a greedy search algorithm, *Reliability Engineering & System Safety* **207**: 107344.

Lin, C. (2005). Recall-oriented understudy for gisting evaluation (rouge), *Retrieved August* **20**: 2005.

Liu, C., Mao, J., Sha, F. and Yuille, A. (2017). Attention correctness in neural image captioning, *Thirty-first AAAI conference on artificial intelligence*.

Lowerre, B. T. (1976). *The harpy speech recognition system.*, Carnegie Mellon University.

Lu, H., Yang, R., Deng, Z., Zhang, Y., Gao, G. and Lan, R. (2021). Chinese image captioning via fuzzy attention-based densenet-bilstm, *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)* **17**(1s): 1–18.

Palacios, H. J. G., Toledo, R. A. J., Pantoja, G. A. H. and Navarro, Á. A. M. (2017). A comparative between crisp-dm and semma through the construction of a modis repository for studies of land use and cover change, *Adv. Sci. Technol. Eng. Syst. J* **2**(3): 598–604.

Papineni, K., Roukos, S., Ward, T. and Zhu, W.-J. (2002). Bleu: a method for automatic evaluation of machine translation, *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pp. 311–318.

Popattia, M., Rafi, M., Qureshi, R. and Nawaz, S. (2022). Guiding attention using partial-order relationships for image captioning, *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4671–4680.

Shambharkar, P. G., Kumari, P., Yadav, P. and Kumar, R. (2021). Generating caption for image using beam search and analyzation with unsupervised image captioning algorithm, *2021 5th International Conference on Intelligent Computing and Control Systems (ICICCS)*, IEEE, pp. 857–864.

Shrimal, A. and Chakraborty, T. (2020). Attention beam: An image captioning approach, *arXiv preprint arXiv:2011.01753* .

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł. and Polosukhin, I. (2017). Attention is all you need, *Advances in neural information processing systems* **30**.

Vedantam, R., Lawrence Zitnick, C. and Parikh, D. (2015). Cider: Consensus-based image description evaluation, *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4566–4575.

Vinyals, O., Toshev, A., Bengio, S. and Erhan, D. (2015). Show and tell: A neural image caption generator, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Wang, C. and Gu, X. (2022). Dynamic-balanced double-attention fusion for image captioning, *Engineering Applications of Artificial Intelligence* **114**: 105194.

Wang, Y., Sun, X., Li, X., Zhang, W. and Gao, X. (2021). Reasoning like humans: On dynamic attention prior in image captioning, *Knowledge-Based Systems* **228**: 107313.

Xiao, F., Xue, W., Shen, Y. and Gao, X. (2022). A new attention-based lstm for image captioning, *Neural Processing Letters* pp. 1–15.

Xu, K., Ba, J., Kiros, R., Cho, K., Courville, A., Salakhudinov, R., Zemel, R. and Bengio, Y. (2015). Show, attend and tell: Neural image caption generation with visual attention, *International conference on machine learning*, PMLR, pp. 2048–2057.

Xu, Y., Li, L., Xu, H., Huang, S., Huang, F. and Cai, J. (2022). Image captioning in the transformer age, *arXiv preprint arXiv:2204.07374* .

Yan, C., Hao, Y., Li, L., Yin, J., Liu, A., Mao, Z., Chen, Z. and Gao, X. (2021). Task-adaptive attention for image captioning, *IEEE Transactions on Circuits and Systems for Video technology* **32**(1): 43–51.

Yang, Z., Yuan, Y., Wu, Y., Cohen, W. W. and Salakhutdinov, R. R. (2016). Review networks for caption generation, *Advances in neural information processing systems* **29**.

Zhang, Z., Wu, Q., Wang, Y. and Chen, F. (2021). Exploring region relationships implicitly: Image captioning with visual relationship attention, *Image and Vision Computing* **109**: 104146.