BUILDING IMAGE CAPTION GENERATOR USING NATURAL LANGUAGE PROCESSING

A PROJECT REPORT

Submitted by

DEEPA ANABATHULA – 00831857

SRUTHI BHONAGIRI – 00828687

Under the guidance of

Professor Mr. Khaled Sayed

For the Course DSCI-6004-02

NATURAL LANGUAGE PROCESSING

UNIVERSITY OF NEW HAVEN

WEST HAVEN, CONNECTICUT

SPRING 2024

TABLE OF CONTENTS

ABSTRACT

Nowadays, It is very important to have computer tools that can describe the images. Many programs can detect the image and give the relevant title for the image as output. These programs use deep learning, which is a type of artificial intelligence. In this project, we are working on a program that can understand what's in an image and write the most relevant caption or title of it automatically. This programming tool combines two forms of data which consists of images and text. It's trained to recognize objects or any living things in images using a special kind of neural network called a Convolutional Neural Network (CNN). Then, it uses another part of the text format of information called Long Short-Term Memory (LSTM) to convert words into binary numbers so the computer can understand them better. During training, the program learns to make descriptions that match what's exactly in the image. We tested this program on a dataset called Flickr8k using different types of CNN models. We measured how well it worked using an evaluation methodology score called BLEU score evaluation. The results are accurate and showed that the program did a good job of recognizing objects in images and generating the most relevant caption for the images that are provided as the input.

INTRODUCTION

There is huge progress in using technology for communication, not only in the text form but also in the form of images. Rather than the text, images share a lot of information. As humans our brain immediately starts annotating and labeling when we see an image in front of us, So we can relate to the the image and get the information. Image captioning aims to detect this information by describing the image content through image and text processing techniques. But when it comes to computers, How can the computer decode the image to get the exact relatable information?

Image captioning combines two big fields of research like computer vision, which helps computers see and analyse images, and Natural language processing, which helps them understand and generate to human-like language. So, when a computer generates a caption for any image, it's not just about recognizing what's in the picture. It's also about figuring out how all the different things in the picture relate to each other and putting that into words.

It is not easy for computers because images can be complex, with many different objects and details. But being able to describe images accurately is super important for many things. For example, it can help individuals with visual loss to gain a clearer understanding of their environment.

To train computers how to do image captioning, we use trending technologies like machine learning, where computers learn from instances, and particular methods like Convolutional Neural Networks (CNNs) for understanding images and Long Short Term Memory (LSTM) for understanding and generating text language.

In this project, we're using a combination of CNNs and LSTMs to teach computers how to caption images. CNNs help the computer understand what's in the picture, while LSTMs help it turn that understanding into a sentence. It's like the CNN acts as the "eyes" of the computer, while the LSTM acts as its "mouth”. Image caption generator uses datasets to label images with English keywords. The model uses the ImageNet dataset to train a CNN model called Xception for extracting image features.The extracted features from Xception are then used by an LSTM model to

create the image caption.

To make the image caption generator, we combine the CNN (Xception) with the LSTM.

We'll explain more about how we're doing this and the results we've found in the rest of this paper. We'll talk about what other researchers have done in this area, the specific methods and data we've used, how we've measured our success, and what we've learned from our experiments. Finally, we'll wrap up with a summary of our findings.

CONVOLUTIONAL NEURAL NETWORK (CNN)

CNNs are special computer programs made to understand and sort out images in deep learning. They're like specialized tools for understanding visual information. Think of an image like a giant puzzle made of tiny squares called pixels. CNNs are excellent at examining this puzzle and figuring out what's in the image, even if it's been changed in size, moved, or flipped.

CNN works by scanning the image bit by bit, from left to right and top to bottom. When they scan the image, they notice important things like lines, shapes, and colors. Then, they use all this information to understand what the image is about.

So, when we use CNNs in our project, we're using this technology to help our model understand the images it's looking at. This understanding is crucial for generating accurate and meaningful captions for the images.

LONG SHORT TERM-MEMORY (LSTM) IN IMAGE CAPTION GENERATOR

LSTM is a type of recurrent neural network (RNN), which is good at predicting what comes next in a series of things, making it great for tasks where you need to predict the next steps in a sequence.

In our project, we're using LSTM to generate captions for images. LSTM is used for predicting the next word in a sequence of text. This is similar to how Google's predictive text feature suggests the next word based on what you've typed until now.

As our model looks at an image, the LSTM part of the model is responsible for processing the information relevant to generating a caption. It chooses which parts are crucial and which ones it can leave out. Once the LSTM has processed the image features extracted by the CNN, it starts generating a description. It uses the relevant information it has learned to piece together a coherent and meaningful caption for the image.It picks out the important bits and leaves behind the less important ones.In our image captioning model, we use CNN to pull out important details from the images and LSTM to write sentences that describe them well. This mix helps our model grasp what's in the pictures and describe it accurately in common language.

DATASET

In this project, we're using the Flickr_8K dataset to train our image caption generators. It's quite large, around 1GB in size.

It is arranged into two main folders: Flicker8k_Dataset and Flickr_8k_text.

The Flicker8k_Dataset folder consists of a total of 8091 images. These images will be utilized to train our models. Inside the Flickr_8k_text folder, we will find many files. The most important one is called Flickr8k.token. This document lists all the names of the images along with their respective captions. Reviewing the information from the Flickr_8k_text folder gives us access to all the necessary details for our project,

including multiple captions for each image, providing a wide range of examples for our models to analyze and learn from.
The key file we'll be using is Flickr8k.token, which pairs each image with its captions. This dataset is very important for training our models to generate accurate captions for images.

DATASET PREPROCESSING

Image captioning is good because it combines computers looking at images and understanding what there are protaiting , then describing them in words like a person would. The Flickr_8K dataset has lots of images along their descriptions. The following are the preprocessing steps.

1. Loading the Dataset:

The first step in our preprocessing pipeline involves loading the Flickr_8K dataset, which is organized into separate folders for images and text data. The images are stored in the Flicker8k_Dataset folder, while the textual information, including captions, is stored in the Flickr_8k_text folder. By reading the captions from the text file (Flickr_8k.token.txt), we create a dictionary mapping each image filename to its associated captions.

2. Cleaning Captions:

To ensure consistency and improve the quality of our text data, we perform cleaning operations on the captions. This involves converting all text to lowercase, removing punctuation marks, and filtering out words containing numbers. By standardizing the format of captions, we facilitate better training of our image caption generator model.

3. Extracting Features:

Next, we leverage a pre-trained Convolutional Neural Network (CNN) model, specifically Xception, to extract features from the images in the dataset. These features

capture the essential visual information present in each image and are represented as feature vectors. By passing each image through the CNN model and obtaining its corresponding feature vector, we prepare the visual data for integration with the textual data during model training.

4. Tokenizing Vocabulary:

Textual data, including captions, needs to be converted into a format that our model can process effectively. To achieve this, we tokenize the cleaned captions, which involves converting words into numerical tokens. This tokenization process creates a mapping between words and their corresponding numerical indices, enabling the model to understand and generate text-based sequences.

5. Creating Input-Output Pairs:

In preparation for model training, we create input-output pairs by combining image feature vectors with sequences of tokens representing captions. These pairs serve as training data for our image caption generator model. Using a data generator function, we generate batches of input-output pairs to feed into the model during training, ensuring efficient memory utilization and scalability.

6. Defining the Model Architecture:

Finally, we define the architecture of our image caption generator model using the Keras Functional API. The model typically comprises a feature extractor (CNN), a sequence processor (LSTM), and a decoder. These components work synergistically to learn the associations between images and their corresponding captions, enabling the model to generate descriptive and contextually relevant captions for unseen images.

TRAINING & TESTING

As we see all image captions are available in the Flickr 8k.token file of the Flickr_8k_text folder. If you analyze this file carefully, you can drive the format of image storing, each image and caption separated by a new line and carry 5 captions numbered from 0 to 4 along with.

Now we are going to use the pre-trained model called Xception which is already trained with large datasets to extract the features from these models. Xception was trained on an imagenet dataset with 1000 different classes to classify the images. We can use keras.applications to import this model directly. We need to do a few changes to the Xception model to integrate it with our model. The xception model takes 299*299*3 image size as input so we need to delete the last classification layer and extract out the 2048 feature vectors.

Extract_features() function is used to extract these features for all images. At the end we will put the features dictionary into a pickle

Loading dataset for model training

A file named “Flickr_8k.trainImages.txt” is present in our Flickr_8k_test folder. This file carries a list of 6000 image names that are used for the sake of training.

Machines are not familiar with complex English words so, to process model’sdata they need a simple numerical representation. That’s why we map every word of the vocabulary with a separate unique index value. An in-built tokenizer function is present in the Keras library to create tokens from our vocabulary.

For training the model as a supervised learning task we need to feed it with input and output sequences. Total 6000 images with 2048 length feature vector and the caption represented as numbers are present in our training sets. It’s not possible to hold such a large amount of data into memory so we are going to use a generator method that will

yield batches.

Define the CNN-RNN model,From the Functional API, we will use the Keras Model in order to define the structure of the model. It includes:

Feature Extractor –With a dense layer, it will extract the feature from the images of size 2048 and we will decrease the dimensions to 256 nodes.

Sequence Processor – Followed by the LSTM layer, the textual input is handled by this embedded layer.

Decoder – We will merge the output of the above two layers and process the dense layer to make the final prediction.

We will generate the input and output sequences to train our model with 6000 training images

After successful model training, our task is to test the model accuracy by inputting test image data.

EVALUATION METHODOLOGY

The evaluation methodology for our image caption generator model involves two main approaches: automatic evaluation metrics and human evaluation.

Automatic Evaluation Metrics:

We will utilize two key metrics:

BLEU Score: This metric compares the generated captions with human-written reference captions. It measures how closely the generated captions match the reference captions in terms of word sequences.

METEOR Score: METEOR evaluates the quality of machine-generated text by considering not only exact word matches but also synonyms and paraphrases. It provides a more comprehensive assessment of the similarity between generated and reference captions.A higher score in both BLEU and METEOR indicates better quality captions produced by the model.

Human Evaluation:

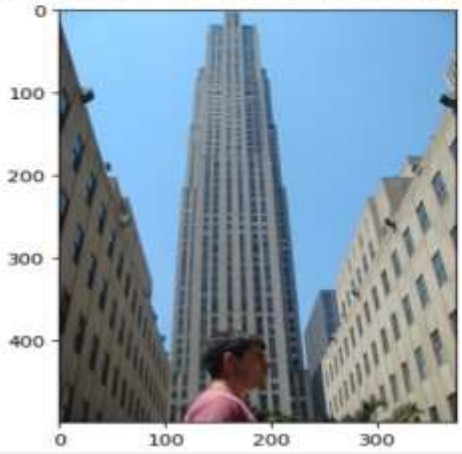
Apart from automatic metrics, we will involve human evaluators to provide feedback on the generated captions. They will assess whether the captions are coherent, relevant, and accurately describe the content of the images. Human evaluation adds a qualitative dimension to the assessment process, capturing aspects that automated metrics may overlook. By combining automatic evaluation metrics with human evaluation, we aim to thoroughly evaluate the performance of our image caption generator model. This comprehensive approach allows us to gain insights into both the quantitative and qualitative aspects of the generated captions, providing a more understanding of the model's effectiveness

RESULT

OUTPUT OF IMAGE CAPTION GENERATOR

```
generate_caption("1032460886_4a598ed535.jpg")
```

```
---------------------*Actual*---------------------
startseq man is standing in front of skyscraper endseq
startseq man stands in front of skyscraper endseq
startseq man stands in front of very tall building endseq
startseq behind the man in red shirt stands large skyscraper endseq
startseq there is skyscraper in the distance with man walking in front of the camera endseq
--------------------*Predicted*--------------------
startseq man in red shirt stands in front of skyscraper endseq
```




```
generate_caption("1052358063_eae6744153.jpg")
```

```
--------------------*Actual*--------------------
startseq boy takes jump on his skateboard while another boy with skateboard watches endseq
startseq child is performing skateboard trick while another child with skateboard leans on wall endseq
startseq little boy skateboarder is doing trick on his board while another young skateboarder watches endseq
startseq young boy skateboarder jumping on platform on skateboard endseq
startseq two skateboarders endseq
--------------------*Predicted*--------------------
startseq skateboarder wearing black shirt and jeans is doing trick on skateboard endseq
```

CONCLUSION

In this guide, we've constructed a powerful deep learning model using CNN and LSTM. Although we trained our model on a small dataset of 8,000 images, larger datasets containing over 100,000 images are used in business-level models for higher accuracy. As the dataset size increases, so does the accuracy of the model. So, if you aim to create a more precise caption generator, consider using this model with larger datasets.

Training the model for more epochs improves its performance, but it requires more time, careful planning, and possibly specialized hardware. When dealing with extensive datasets like Flickr32k, deeper models are needed to comprehend complex details within the images.

Generating good captions is challenging because the model must understand both the visual content and the textual context. Despite these challenges, our model has achieved good results, showing promise for future improvements.

In essence, this project has demonstrated the effectiveness of using deep learning techniques to generate captions from images. With further research and enhancements, we can continue to refine our model and its applications across various domains.

REFERENCES:

1. W. Abdulla Mask r-cnn for object detection and instance segmentation on keras and tensorflow

2. M.W. Akram, M. Salman, M.F. Bashir, S.M.S. Salman, T.R. Gadekallu, A.R. Javed A novel deep auto-encoder based linguistics clustering model for socialtext Trans. Asian Low-Resource Lang. Inf. Process. (2022)

3. P. Anderson, X. He, C. Buehler, D. Teney, M. Johnson, S. Gould, L. Zhang Bottom-up and top-down attention for image captioning and visual question answering Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2018)

4. M.F. Bashir, H. Arshad, A.R. Javed, N. Kryvinska, S.S. Band Subjective answers evaluation using machine learning and natural language processing IEEE Access, 9 (2021), pp. 158972-158983View at publisher CrossRefView in ScopusGoogle Scholar

5. M. Buric, M. Pobar, M. Ivasic-Kos Ball detection using yolo and mask r-cnn 2018 International Conference on Computational Science and Computational Intelligence (CSCI), IEEE (2018)

6. A. Capuano, A.M. Rinaldi, C. Russo ,An ontology-driven multimedia focused crawler based on linked open data and deep learning techniques Multimedia Tools and Applications, Springer (2019), pp. 1-22

7. H. Fang, S. Gupta, F. Iandola, R.K. Srivastava, L. Deng, P. Dollár, J. Gao, X. He, M. Mitchell, J.C. Platt, et al. From captions to visual concepts and back Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2015)

8. K. He, G. Gkioxari, P. Dollár, R. Girshick Mask r-cnn Proceedings of the IEEE International Conference on Computer Vision (2017)

9. M.Z. Hossain, F. Sohel, M.F. Shiratuddin, H. Laga A comprehensive survey of deep learning for image captioning ACM Comput. Surv., 51 (6) (2019), pp. 1-36

10. P. Hurtik, V. Molek, J. Hula, M. Vajgl, P. Vlasanek, T. Nejezchleba Poly-yolo: higher speed, more precise detection and instance segmentation for yolov3 arXiv preprint