Image caption generation

1st Umang Mehta
Student ID: 1117269
Lakehead University

2nd Setul Patel
Student ID: 1104360
Lakehead University

3rd Kushal Patel
Student ID: 1104361
Lakehead University

4th Nishchay Trivedi
Student ID: 1106924
Lakehead University

***Abstract*—Image captioning is defined as generation of caption for a given image. Recognizing attributes, important objects and their relationship is main goal for image caption generation. Image captioning has a variety of applications, including guidelines for editing software, use of virtual assistants, image indexing, visually disabled people, social media, and many other applications for natural language processing. Unlike other models, the main parameters to evaluate an image is about having accurate caption for given image not any measures or accuracy. Image captioning is also well-researched area in computer vision. We optimized a deep neural network architecture to get optimal results for generated caption. We discussed here different results using different databases such as MSCOCO, fliker8k and also several deep learning and neural network techniques such as LSTM (Long short term memory), multi-task learning and RNN (Recurrent neural network).**

***Index Terms*—Image caption generation, deep learning, convolutional neural network, RNN, LSTM, machine learning, multi-task learning, natural language processing**

I. Introduction

Ever after researchers began focusing on object identification in pictures, it became clear that presenting only the names of the known objects does not give such a strong impression as a complete explanation of a human being. Before machines think, speak and act like humans, natural language explanations must remain a problem to solve. The function of generating image description or image captioning (IC) is to produce a text description for a given image automatically. There is no question about the benefits and potential of human-like technologies; from allowing robots to communicate with people to unique uses for infant schooling, health assistants for the disabled or visually challenged, and much more. In a single sentence, the generated text is required to explain what is physically represented in the image, such as the entities / objects present in the image, their characteristics, the actions / activities done, the relations with the entity / object (including quantification), the location / scene, etc. (e.g. "a horse is running"). Since there are so many possibilities for practical applications in society, it is not shocking that many experiments have already sought to get more detailed definitions to make computers think like humans. Machines, however, often lack the normal way to interact with humans and this remains a difficult problem to overcome. As an image captioning example, figure 1 shows one of the step by step way for image captioning.

Most modern image captioning systems, inspired by neural machine translation, use an encoder-decoder method in which an input image is encoded into an intermediate representation

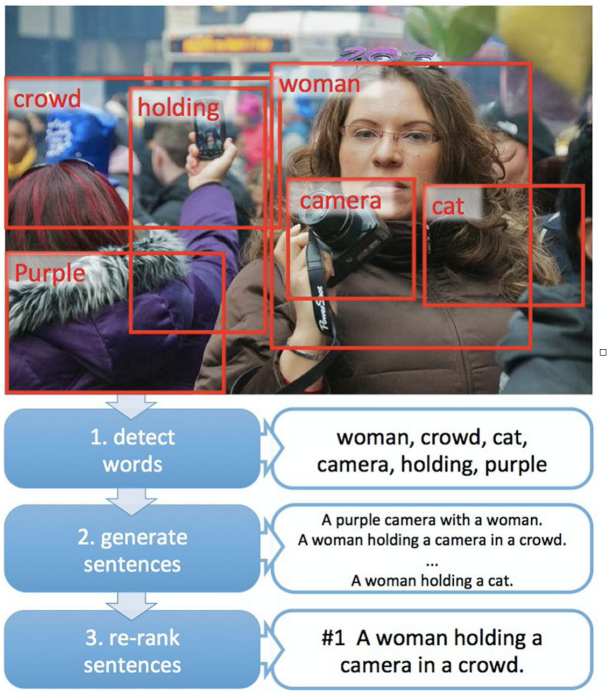

Fig. 1. Image caption generation

Source: https://phys.org/

of the information stored within the image, and then decoded into a concise text sequence. Such encoding can consist of a single output vector of a CNN, or multiple visual features obtained from different regions within the file. Although these detection-based encoders reflect the state of the art, information on the spatial relationships between the sensed objects, such as relative location and height, is currently not being used. However, this knowledge can also be crucial to interpreting the meaning of an image, which is used by humans in real world reasoning. For example, relative location can aid in distinguishing "a girl riding a horse" from "a girl standing next to a horse". Likewise, relative size can help to differentiate between "a woman playing the guitar" and "a woman playing the ukelele". The integration of spatial interactions has been shown to enhance object detection efficiency itself.

II. Literature review

The background of image captioning is being interesting having everyone working on it but not getting the results

wanted for it. Zakir Hossain et. al [1] did a survey on deep learning techniques for image captioning. They did research based on different techniques for types of learning, architecture, number of captions, feature mapping and language models as shown in figure 2. Existing image captioning methods compute log-likelihood scores to evaluate their generated captions. They enclosed the survey using different types of metrics that used to evaluate model for image captioning like BLEU (Bilingual evaluation understudy), ROUGE (Recall-Oriented Understudy for Gisting Evaluation), METEOR (Metric for Evaluation of Translation with Explicit ORdering), CIDEr (Consensus-based Image Descripton Evaluation) and SPICE (Semantic Propositional Image Caption Evaluation). They compared results for these metrics and different models used by such researchers by their papers as reference.

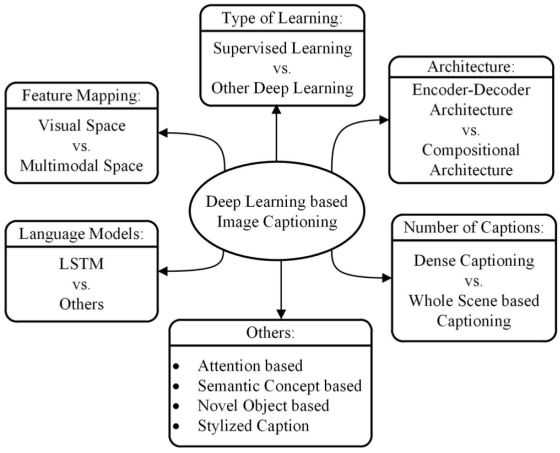

Fig. 2. Deep learning based image captioning

Source: Zakir Hossain et. al[1]

Image caption generation has an application as defining the product for marketing purpose as Philipp Harzig et al.[2] prepared a model based on generating descriptive captions for images which contains particular products of companies. For the model, they used their own database and conducted a series of experiments on 9 different models where all models were trained using the classification-aware loss. Model base was a model initialized with parameters from a vanilla Inception-v3 network. They trained model with base-cls as initialization without unfreezing the encoder network to show the effectiveness of MTL (multi-task learning). They got the accuracy of around 70 percent with the different metrics of evaluation.

III. DATABASE

The main goal of image caption generation application is understanding the visuals of the image. For machine to understand image it requires to identify what kind of object present in image, is it 2D or 3D, relationship between the objects and providing a semantic description of the scene. The dataset plays important role for satisfying such requirements. The Datasets we are using for our application is Microsoft Common Objects in COntext (MS COCO) dataset, which is one of the best image datasets available with 82000 images. The size of the dataset is 13GB. The dataset has more instances so it would be more useful to teach complex models with precise localization. We split the dataset into 80-20 ration, where 80 percentage for training and 20 percentage for validation. After splitting dataset, the length of img_name_train, cap_train, img_name_val, cap_val are 24000, 24000, 6000, 6000 respectively.

IV. MODEL TRAINING

We are using a subset of 30,000 captions and their corresponding images to train our model. Next step is to preprocess the model using inceptionV3 to classify each image in which we will extract features from the last convolutional layer. Now, we will convert the images to the desired format of InceptionV3 by resizing the image to 299px and preprocessing the images using the preprocess input method to normalise the image so that it contains pixels within the range of -1 to 1, which corresponds to the image format used to train InceptionV3. The dataset is converted using numpy in order to work with keras. A numpy array basically is a grid of values, all of the same type and index by tuple of non-negative integers. The rank of the array is defined by the number of dimensions. Hence, the numpy array data is more precise to work with in order to get good results. Now we are going to build a tf.keras model, where the output layer is the last convolution layer in the architecture of InceptionV3. The output form for that layer is 8x8x2048. You use the last convolution layer because, in this case, you are paying attention. During training we will not carry out this initialization because it might turn into a bottleneck. We forward each image over the network and store a dictionary of the resulting vector (image name to feature vector). Pick up the dictionary after all the images are passed through the network, and copy it to the disc.

A. Preprocessing

Tokenizing is the task of chopping the document unit into pieces called tokens which eventually means to throw away the useless characters.There are several types of tokenization as well including unigrams, bigrams and trigrams. The main purpose of it is to calculate the probability of sequence of tokens. Tokenizing the captions (such as separating on spaces, for example) is the next step. It gives us a vocabulary of all the special terms in the data (e.g., ”surfing,” ”football” etc). Next , we will restrict the size of your vocabulary to the top 5,000 words (to conserve memory). We then construct mappings from word-to-index and from index-to-word. Finally, let us pad all sequences to be the same length as the longest one.

B. Encoder - Decoder

The model here works on encoder- decoder method to work with the framework and architecture. Our model takes a single raw image and produces a y-encoded caption as a sequence of encoded 1-of-K words as shown in equation 1. Where K

is the vocabulary size, and where C is the caption frequency. We use a CNN to generate a collection of feature vectors that we refer to as vec- tors for annotation. The extractor generates L vectors, each representing a D-dimensional image representation corresponding to a part of the image.

$$y = (y_1, ..., y_C), y_i \in R_K \tag{1}$$

To obtain a correspondence between the vectors of the function and portions of the 2-D image, we extract features from a lower convolution layer unlike previous work that used a fully connected layer instead. This enables the decoder to concentrate selectively on certain parts of an image by weighing a subset of all the vectors in the function. An RNN encoder-decoder takes an input sequence and produces another output sequence. A sentence in English, for example, can be considered as a sequence of words that can be input, and a French translation of the sentence is generated as a result that is again a sequence of words. The encoder-decoder model basically is related to the model used by Kelvin Xu et al.[3]

The features contained in the respective.npy files are extracted and then passed through the encoder. The output of the encoder, secret state (initialised to 0), and the input of the decoder (which is the starting token) are transferred to the decoder. The decoder returns the predictions and the hidden state of the decoder. The hidden state of the decoder is then passed back to the model, and the predictions are used to measure loss. Use the forcing teacher to determine the next decoder entry.

```
EPOCHS = 20

for epoch in range(start_epoch, EPOCHS):
  start = time.time()
  total_loss = 0

  for (batch, (img_tensor, target))
  in enumerate(dataset):
   batch_loss, t_loss =train_step(img_tensor,target)
   total_loss += t_loss

  if batch % 100 == 0:
    print ('Epoch {} Batch {} Loss {:.4f}'.format(
    epoch + 1, batch, batch_loss.numpy() /
    int(target.shape[1])))
    # storing the epoch end loss value to plot later
    loss_plot.append(total_loss / num_steps)

    if epoch % 5 == 0:
      ckpt_manager.save()

    print ('Epoch {} Loss {:.6f}'.format(epoch + 1,
                                total_loss/num_steps))
    print ('Time taken for 1 epoch {} sec\n'.format
                (time.time() - start))
```

Listing 1. Training the model

Teacher forcing is the strategy by which the target word is transferred to the decoder as the next data. Calculating the gradients and applying it to the optimizer and backpropagate is the final step. Next step is to train the model for which the code is shown below in listing 1.

V. Evaluation and Results

The evaluation is the technique used to check how well our model will perform on new and unseen data. By evaluating it we can test our model whether it works accurately and the able to make good decision on future samples. There are basically three types of data to test: 1. Training set is subset of the dataset to train our model and used to build our model for predication. 2. Validation set is a subset of dataset to assess the performance of our model in the training phase. 3. Test set is a new or unseen dataset to assess the likely performance of the model in future.

Fig. 3. Input Image

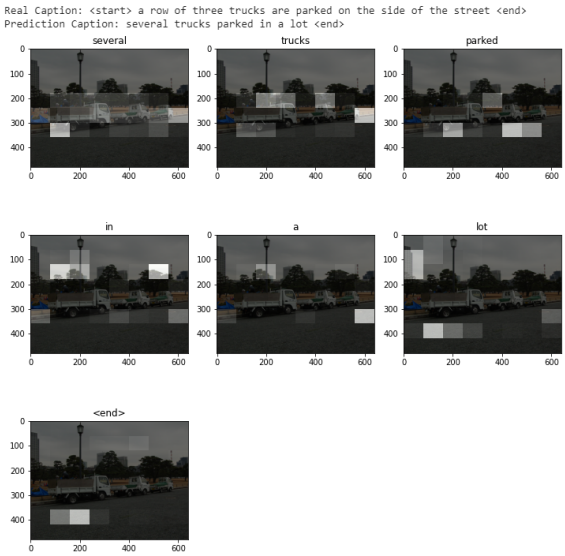

Fig. 4. Testing result

In our model, the input to the decoder is its previous predictions along with the hidden state and the output of the encoder at each time stage. It stores the attention weights for

every time step and will not more predict till the end token. The result is divided in to two main sentences that is: Real caption and prediction caption. Now, during the testing, our model performs well on unseen image and generates caption most likely to the real one. The figure 3 shows the actual image given as an input and figure 4 depicts the caption for that image ”several trucks parked in a lot” as prediction as the actual caption is ”a row of three trucks are parked on the side of the street”.

VI. CHALLENGES

The challenges came during training and testing part was to get model fit and get at least related caption generated for the given image. As the project was running on the google colab, which is either running on a GPU or TPU. The main problem with it was run-time which was eventually high and needed more space because of the database we are working on and also the model we were using was not working on the classes and functions as the final model is working, so the data space was lacking and also the model fit was taking time and sometimes GPU and TPU trashed because of complex structure of the model. So, according to the requirements we changed model to overcome these problems. Another challenge regarding to the model was over-fitting and under-fitting. Underfitting happens when machine learning model can not adequately get the underlying structure of the data. It means the data does not fit properly according to the model.

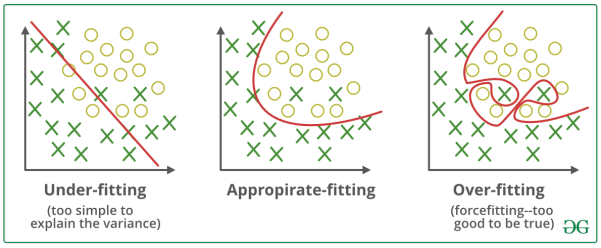

Fig. 5. Overfitting

Source: https://towardsdatascience.com

Overfitting happens when the noise of the data from dataset is captured by a statistical model or machine learning model. As shown in figure 3, underfitting happens when the number of parameters to train the particular model is less and overfitting occurs when it is high and model is trained for more number of epochs As a result, sometimes we were getting the caption unrelated to the image when the number of epochs and batch size were not perfect but some times the caption was repeating again and again because of lacking training of the model.

REFERENCES

[1] A Comprehensive Survey of Deep Learning for Image Captioning, MD. Zakir Hossain, Ferdous Sohel, Mohd Fairuz Shiratuddin, Hamid Laga, Murdoch University, Australia, 2018.

[2] Image Captioning with Clause-Focused Metrics in a Multi-Modal Setting for Marketing, Philipp Harzig, Dan Zecha, Rainer Leinhart, Carolin Kaiser, University of Augsburg, Germany, 2019.

[3] Show, Attend and Tell: Neural Image Caption Generation with Visual Attention, Kelvin Xu, Jimmy Lei Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhutdinov, Richard S. Zemel, Yoshua Bengio, University of Toronto.

[4] Show and Tell: A Neural Image Caption Generator, Oriol Vinyals, Samy Bengio, Dumitru Erhan, Alexander Toshev, google.

[5] Automatic Caption Generation for News Images, Yansong Feng, Mirella Lapata, IEEE.

[6] Lecture notes, Dr.Emad Mohammad, Lakehead University, 2020.

[7] Lecture notes, Dr. Thangarajan Akilan, Lakehead University, 2020.