Deep Learning

AT3

Report

Stefan Hall - 14156968
Huy Nguyen - 14001527
Declan Stockdale - 11214549

Project outline

The following report details the work of our group in developing a deep learning model for captioning images with short descriptions. Each member of the group created a model or series of models, capable of extracting the key features of random images and assigning captions to them. This report will describe the data which was ingested by the models, and then discuss how the data was prepared and used to generate the results. Following this, the various architectures of the different models used will be discussed as well as the different metrics employed to assess the quality of each model for this task. Finally, the best model will be put forward based on the metrics discussed and future recommendations for this project will be presented.

Introduction

Image captions also known as automated image annotation is a field of research in the machine learning domain that combines aspects of computer vision and natural language processing. It aims to produce useful descriptive captions for a given input image.

This is a complex problem as a model needs to not only predict the objects with an image, of which there may be many, it also needs to know the context and possible cardinality of how objects are related to each other (Elhagry 2021) . One example could be an image of a man standing next to a dog. A potential caption could be “a man stands to the left of his dog” where the model not only needs to predict the dog and man, but know they are next to each other and which side they are in respect to one another. From this simple sample, one can appreciate the complexity of the problem.

The Dataset

The dataset used to train the models was the Flickr8k image captioning dataset (kaggle.com). This data is split into two main sections, the Flickr8k_dataset, which is a collection of 8092 unique images of random scenes and the Flickr8k_text, which contains various breakdowns of different descriptions for each image. The descriptions are broken down into a training set of 6000 images and their related descriptions, 1000 validation images and another 1000 testing images.

240583223_e26e17ee96.jpg from the Flickr8k_dataset

The above image for example is one of the over 8000 images available for training and it is paired with the following captions:

240583223_e26e17ee96.jpg#0	A brown dog is running on a dirt path .
240583223_e26e17ee96.jpg#1	A dog is running down a dirt path .
240583223_e26e17ee96.jpg#2	A running golden retriever
240583223_e26e17ee96.jpg#3	A yellow dog running down a sandy path .
240583223_e26e17ee96.jpg#4	The yellow dog is running on the dirt road .

Each image has a collection of 5 descriptions linked to it. While each description is slightly different, they all generally describe the same scene. The following images detail this further:

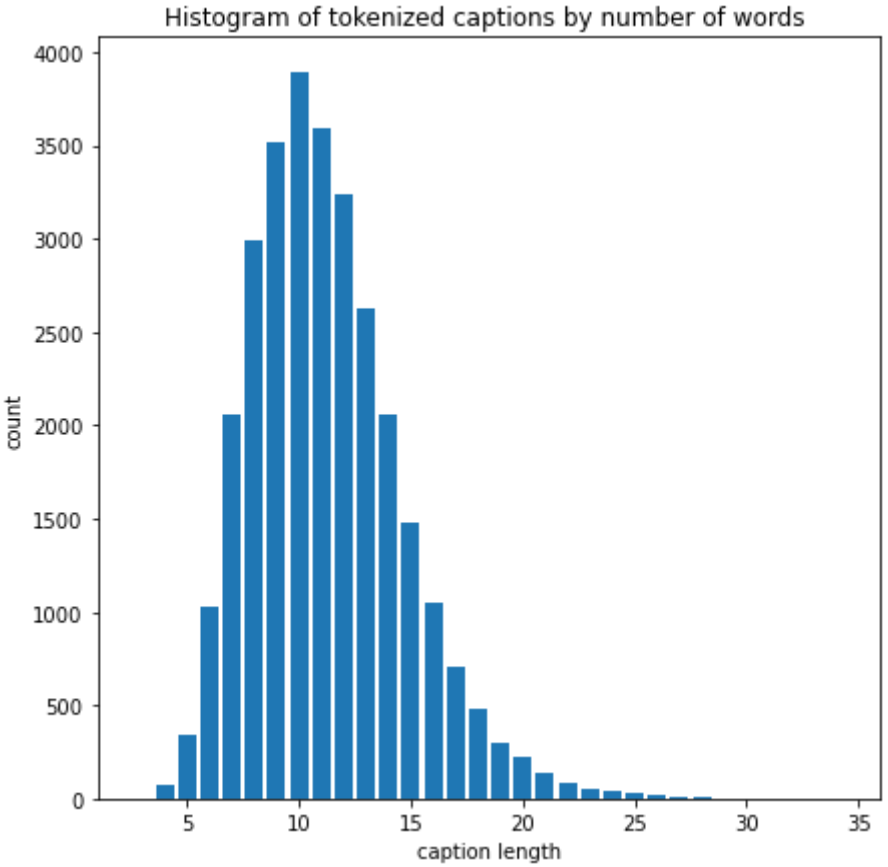
3708266246_97a033fcc7.jpg from the Flickr8k_dataset

3708266246_97a033fcc7.jpg#0	A black and white dog is going through an obstacle course .
3708266246_97a033fcc7.jpg#1	A black and white dog jumps over a bar .
3708266246_97a033fcc7.jpg#2	A black dog leaping over a hurdle .
3708266246_97a033fcc7.jpg#3	A border collie jumping over a hurdle .
3708266246_97a033fcc7.jpg#4	Black and white dog jumping over a blue obstacle

2428275562_4bde2bc5ea.jpg from the Flickr8k_dataset

2428275562_4bde2bc5ea.jpg#0	A
2428275562_4bde2bc5ea.jpg#1	A black and white dog is climbing down a hill .
2428275562_4bde2bc5ea.jpg#2	A black and white dog is sliding down a sandy hill .
2428275562_4bde2bc5ea.jpg#3	A black and white dog wearing a red collar is digging in the dirt .
2428275562_4bde2bc5ea.jpg#4	A black dog running on a sand mound .

Since there were nearly 5 descriptions per image, this meant that the corpus was made up of ~40000 different descriptions, each of varying lengths. The following diagram shows the distribution of descriptions, by caption length.



Training process

Training of the models was mostly performed on Google Colab using the automatically allocated resources. In addition to this, one of our team members had a substantial GPU capable of running these models relatively quickly. We were then able to bypass Google's limitations surrounding their resource allocations, as required. All the models used were made available through TensorFlow and its various libraries.

Data Preparation

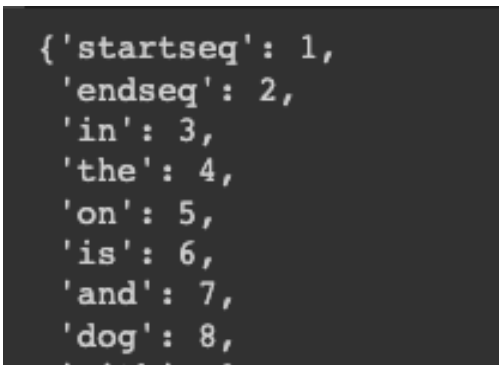
In order to train this model, two forms of data preparation were required. Firstly, the image data needed to be configured into a readable format for a convoluted neural network. Secondly, the text descriptions needed to be cleaned and tokenised to create a complete vocabulary of the corpus.

While each member of the team employed different models in their attempts at this project, all of the images were processed through a convolutional Neural Network (CNN). As such, each image needed to be converted into a tensor, with the necessary target size for the specific model being used.

Some team members ran the initial CNN on the images to extract the relevant features of all the images. These features were then stored in a pickle file to be accessed by the model later, during the compilation process. Fortunately the keras module allows for quick and easy preprocessing, which converts an image into a normalised, numerical multidimensional array, readable by the specific model being employed. As such, the preparation for the image data is very straightforward.

The text data on the other hand requires significantly more set up. First, all associated descriptions need to be clearly mapped to a specific image. With some minor data wrangling, eventually a dictionary with the jpg as the key and a list of all descriptions as the values is created. Next, the text needs to be cleaned and tokenised. The cleaning process is straightforward. First each unique word is placed in a list and then all words are set to lower case, all punctuation is removed, hanging s and a's are removed, and any numeric characters are removed. Then, before being fed into the Recurrent Neural Network (RNN) model, this list of cleaned unique words is tokenised, resulting in a numbered list of unique words.

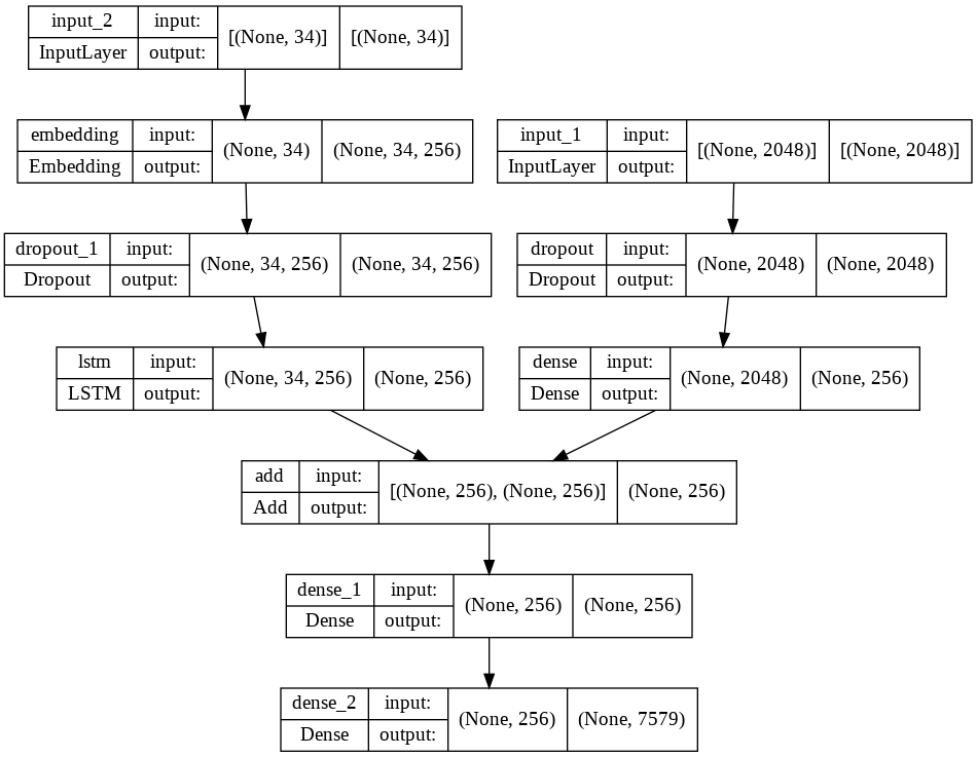
Once the list of cleaned and tokenised, the last step was to set up the padding for a sequence so that all token vectors have a uniform length. The max length caption length for this data set was 34 so all captions less than 34 words long had additional 0's added at the end to ensure uniform length..



Words cleaned and tokenised

Architectures Tested

A variety of models available from keras were tested for this project. Each model used both a CNN and an RNN. The CNN models were chosen as they performed well on the ImageNet database which comprises a 1000 unique classes. We also preload the ImageNet weights for all our models essentially allowing transfer learning to occur. The CNN serves to extract the features from the images using preloaded ImageNet weights, whereas the RNN generates the sequence of words that make up the caption. The first layer after the input layer in the RNN is an embedding layer. This layer provides a vector space which serves to map the proximity of various words to each other. This is then used to create a viable string of text to caption each image. The following diagram shows the high level architecture of all of our models.



In the above diagram, input 1 feeds into the CNN and its feature extraction, whereas input 2 feeds into the RNN, which generates the sequence of words. The two models are then combined through an addition layer and run through the final fully connected layers to the output.

Each type of CNN model required a specific input shape for the images in order to function. The following table lists all the models used and their corresponding input shapes required.

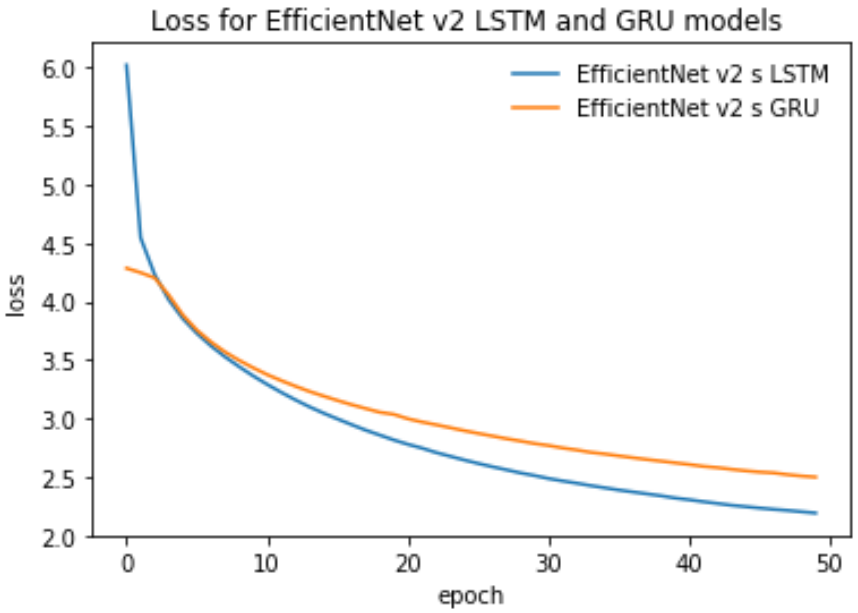
Model	**Input Shape**
VGG19	(224,224)
Inception V3	(299,299)

Efficient Net V2	(384,384)
Mobile Net V3 Small	(576,576)
Inception ResNet V2	(299,299)

As well as experimenting with various CNN models, a few RNN's were also implemented to see if there were any significant improvements. A Long Short Term Memory (LSTM) RNN and a Gated Recurrent Unit (GRU) RNN were both used and the results from each combination were logged.

Optimizer and Loss Function

Adam is the industry standard optimizer in deep learning. The adaptive moment estimation (ADAM) loss function, is simple and efficient and converges much faster than other loss functions available. Therefore all the trials used Adam to reduce the search space and allow for consistent comparison.The loss function used was the categorical cross entropy loss as this is a multiclass classification problem as we're predicting multiple words (Kumar 2021).



The above image shows the difference in loss between LSTM and GRU for the Inception ResNet V2 model

Evaluation Metrics

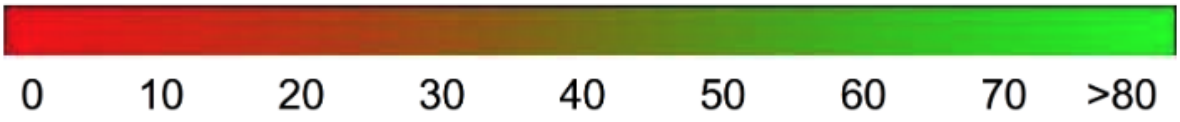
A significant challenge in evaluating automatically generated captions is the sample space of potentially viable solutions. When captioning an image, it is possible to have a variety of different captions which are all technically correct, yet semantically different. Traditional metrics, such as precision, are thus not very useful for assessing how accurate a model is.

This led to the development of alternative metrics to assess model performance. One widely used metric for assessing the accuracy of image captioning is Bi Lingual Evaluation Understanding (BLEU) first published in 2002 (Papineni et al, 2002). This metric is commonly used because it is inexpensive to calculate, language independent and conceptually straightforward. This, along with manually inspecting random samplings of images and their generated captions, is the primary method for evaluation.

BLEU works by separating potential captions into n-grams, that is, 'n' length sequences of words, and assessing the accuracy of a model against these different levels. This results in 4 standard metrics, for example, BLEU 1 has n-gram length of 1, BLEU 4 has n-gram length of 4, etc. Each n-gram in the generated caption is compared to the reference caption and the number of matching n-grams is divided by the total number of n-grams in the generated caption. It also has an additional penalty for caption length which improves caption brevity. Importantly, a perfect score is impossible for BLEU. Instead, Google provides the following guidelines

BLEU Score	Interpretation
< 10	Almost useless
10 - 19	Hard to get the gist
20 - 29	The gist is clear, but has significant grammatical errors
30 - 40	Understandable to good translations
40 - 50	High quality translations
50 - 60	Very high quality, adequate, and fluent translations
> 60	Quality often better than human

The following color gradient can be used as a general scale interpretation of the BLEU score:



(https://cloud.google.com/translate/automl/docs/evaluate#bleu)

The following tables contain the BLEU scores of the various models trained. It's common to report BLEU values from 1-4 in the literature which is also what we've reported here.

LSTM Results after 50 epochs

Model name	**BLEU-1**	**BLEU-2**	**BLEU-3**	**BLEU-4**
VGG 19	0.526	**0.305**	**0.22**	**0.113**
EfficientNet v2 S	0.432	0.201	0.132	0.054

Inception resnet v2	0.448	0.217	0.144	0.064
Inception V3	**0.528**	0.304	0.219	0.110

Gated Recurrent Unit Results after 50 epochs

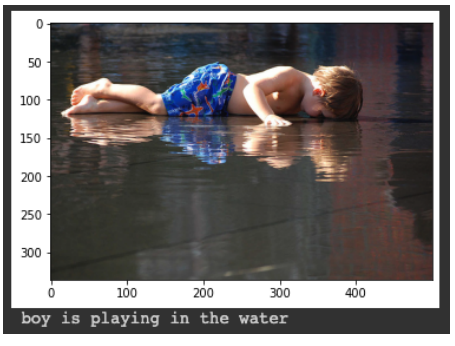
Model name	**BLEU-1**	**BLEU-2**	**BLEU-3**	**BLEU-4**
VGG 19	0.523	0.296	0.211	0.103
EfficientNet v2 s	0.445	0.211	0.143	0.064
Inception resnet v2	0.448	0.209	0.139	0.059
MobileNet v3 small	0.532	0.305	0.220	0.111

Model Performance and Limitations

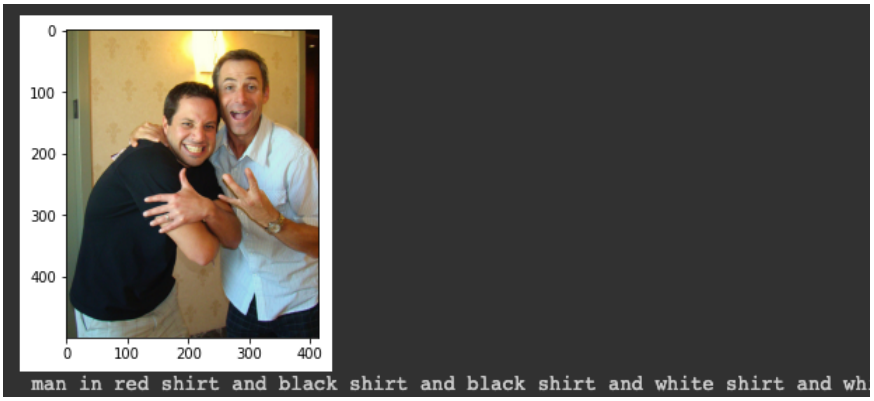
According to the results in the above tables the best performing model appears to be either the VGG19 or the Inception V3 model. Both consistently have the highest results across all BLEU scores, with VGG19 just edging out on top for all scores except BLEU-1.

Since each generated caption is referenced against at least four other captions and the test captions follow the same structure as the training dataset, the BLEU scores provide a meaningful measure of the accuracy of these models. We can therefore say that for our best performing models, it can very accurately predict individual words that should appear in the image captions. However, when it comes to creating meaningful strings of words, especially strings of length 4, on average our highest performing models will likely have significant grammatical errors.

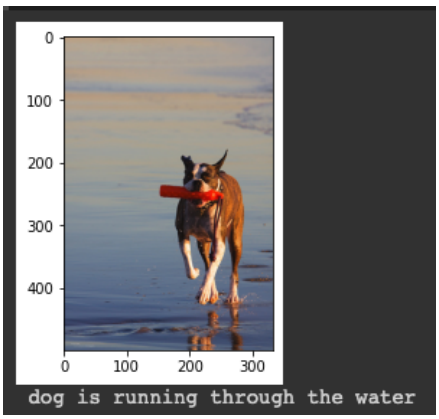
To further evaluate their performance, a random sample of images were run through the model and a caption was generated. The first collection of images shows a collection of images with captions that clearly identify the general description of the scene, though there may be some small grammatical errors (and you may need to zoom in).

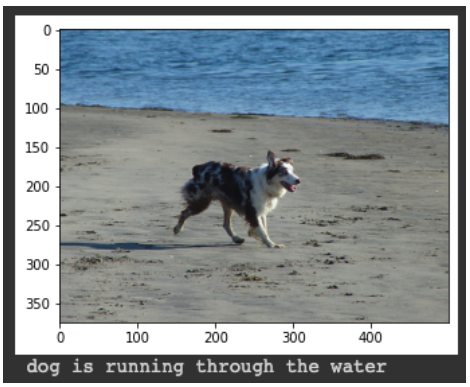


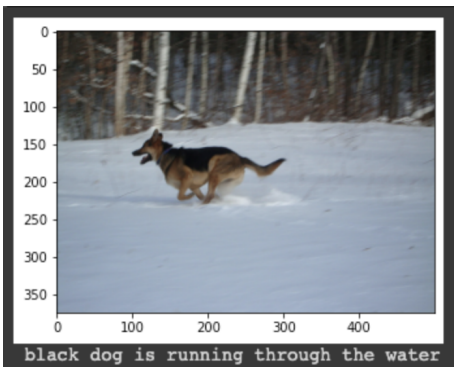
In contrast, the following images all generated very similar (and very wrong) captions. The captions also appear to be a repeating pattern, which may indicate an error within our model. While the model correctly identifies a person, for some reason, it also identifies a red shirt, despite there being none in any of the pictures.



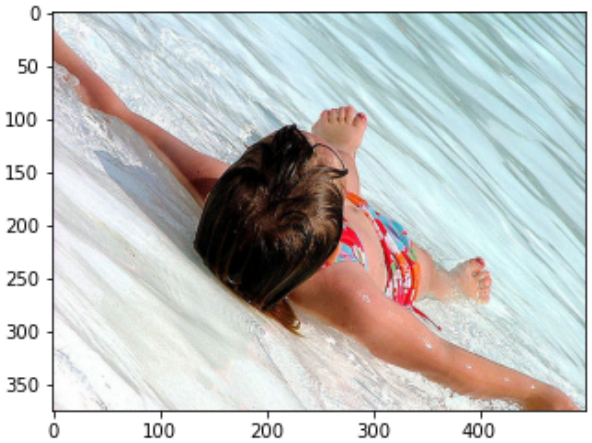
Similarly, the following images have also generated incorrect captions (except, arguably, the first image). These captions are all virtually the same, despite some of the scenes being vastly different (especially the last two).







As such, our models have shown that they are capable of generating captions which can occasionally correctly identify and describe the scene depicted in various images. Though there is still significant room for improvement, as is clearly shown by the BLEU-n scores and the mismatched captions in the later images.



```
vgg19_LSTM_model_0.h5  the child is wearing red hat and holding the face of the water
vgg19_LSTM_model_9.h5  child in red jacket is swimming in the ocean
vgg19_LSTM_model_19.h5  the baby is swimming in the clear ocean
vgg19_LSTM_model_29.h5  girl in pink coat is lying in the ocean
vgg19_LSTM_model_39.h5  girl in pink coat is lying on the sand in pool
vgg19_LSTM_model_49.h5  girl in pink goggles is swimming in the snow
```

We can also observe that the models are learning and improving the quality of the caption with increasing number of epochs. From the image above which used the VGG19 LSTM model, we can see the caption from the model after 1 epoch results in the caption “the child is wearing red hat and holding the face of the water” while the caption fo the model trained for 50 epochs is “girl in pink goggles is swimming in the snow”. It appears that the models are learning information as it was able to identify water but the last model predicted snow which isn’t correct.

Remaining Issues and Recommendations

This report details the process and the results of training an image captioning model within the time allotted for this task. With more time, there are some avenues that would have been further explored, especially in the hope of increasing the BLEU-4 score on the Flickr8k dataset and creating a model capable of accurately captioning the majority of images.

Firstly, the feature extraction from the CNN models could be further fine tuned to be better suited to this particular dataset. Though a method of validation would be required in order to assess the accuracy. In a similar vein, while the key metric for the evaluation of this project has been BLEU, it may be worthwhile to explore other metrics to give further depth in the assessment of the results.

Another avenue to explore would be the method of word embedding and tokenizing the vocabulary. Currently, the embedding is being handled by keras. However, the results may differ if another form of embedding is used, like for example GloVe or word2vec.

The last immediate path for exploration, is the possible effect of augmenting the incoming image data, as is normally done for image classification to avoid overfitting. In order to potentially better generalise the models, the results of this augmentation could be explored.

As for immediate recommendations for the model in its current state, there is a slight difference in the scores between the models trained with LSTM and GRU. It is recommended that, if ever this model is released to production, a trade off between speed and accuracy needs to be defined. Since GRU is more efficient and quicker, but is generating slightly lower scores, if the model requires speed over accuracy, then the GRU is a good candidate.

Though, to further develop and explore these ideas, the current limitation of computational resources available should ideally be solved. For the development of this report, the majority of the team used Google Colaboratory. While this is a powerful tool with a host of packages already available to use, the limitations on time and GPU usage make training deep learning models very difficult. Fortunately one team member had their own dedicated GPU, capable of training the models at a similar speed to the Colab notebooks. However, ideally all team members should be able to work in an IDE that does log them out after a time limit.

References

https://aclanthology.org/P02-1040.pdf

https://towardsdatascience.com/foundations-of-nlp-explained-bleu-score-and-wer-metrics-1a5ba06d812b

https://daniel.lasiman.com/post/image-captioning/

https://towardsdatascience.com/image-captioning-with-keras-teaching-computers-to-describe-pictures-c88a46a311b8

https://machinelearningmastery.com/develop-a-deep-learning-caption-generation-model-in-python/

Elhagry, A., Kadaoui. K, A Thorough Review on Recent Deep Learning Methodologies for Image Captioning, Computer Vision and Pattern Recognition, 28, July 2021

Kaggle, Flickr 8k Dataset, Flickr8k Dataset for image captioning., https://www.kaggle.com/datasets/adityajn105/flickr8k, accessed on 15th May 2022

Kumar, A., Keras – Categorical Cross Entropy Loss Function, Data Analytics Data, Data Science, Machine Learning, AI, Accessed on 15th May 2022

Papineni K., Roukos R., Ward T., Zhu W.J., BLEU: a Method for Automatic Evaluation of Machine Translation, Computational Linguistics, July 2002

Many of the above blog posts provided the guidance for developing the code in our team's notebooks.