# Image Captioning

Deep Learning Project – Winter 2021

Bilal Majeed
501045997

## Table of Contents

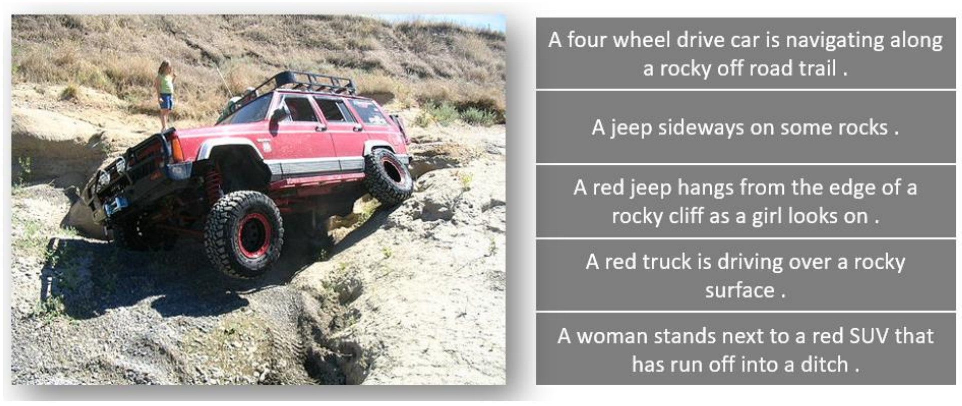
## 1 – Problem Description

The caption generation problem involves training the machine to generate a textual description or caption for the image, given an image. This problem is an interesting one as it combines 2 prevalent areas in data science: Image Feature Extraction and Natural Language Processing. To solve this problem, an image-based model in required, which extracts the features out of the image, and the other is a language-based model, which translates the features given by the image-based model to a natural language sentence. It is expected that the generated caption describes in a sentence what is shown in the image, i.e., the objects present, their interactions, their properties, and the actions being performed. Since this model outputs a sentence, the most appropriate performance metric is the BLEU score (refer to 6.3).

There are a few use cases, some of which have already been implemented: Text to speech of images using an image caption generator to obtain text for visually impaired persons, social media post caption generator, photo editing software edit recommendations based on image descriptions, and visual video content descriptions based on frame-by-frame description generation.

## 2 – Data Source

The data source being used is the Flickr8k dataset which contains 5 captions per image. The dataset contains a folder with 8091 images, each with 224 pixels x 224 pixels x 3 channels, with a unique image identifier as the file name. A separate text file is provided containing 5 captions for each image identifier. The Flickr8k dataset is selected to contain portion of each of the six different groups of the larger Flickr dataset. This dataset is not expected to be generate an effective model due to the size, but this size is chosen due to processing power limits. The dataset can be found on multiple GitHub repositories but there is a no redistribution clause provided by the University of Illinois. The dataset link is provided by the university to researchers after the following form is submitted: [https://forms.illinois.edu/sec/1713398](https://forms.illinois.edu/sec/1713398).



A four wheel drive car is navigating along a rocky off road trail .

A jeep sideways on some rocks .

A red jeep hangs from the edge of a rocky cliff as a girl looks on .

A red truck is driving over a rocky surface .

A woman stands next to a red SUV that has run off into a ditch .

# 3 – Literature Review

## 3.1 – Show and Tell: A Neural Image Caption Generator (2015) [3]

### 3.1.1 – Background

A description includes objects, how they relate to each other as well as their attributes and the activities they are involved in. In addition, this information then needs to be expressed in natural language, which means that a language model is required as well. This paper was inspired from advances in machine translation, where the task is to transform an input from the source language, into its translation T in the target language using the encoder-decoder architecture.

### 3.1.2 – Aim and Methodology

Propose a model for image caption generation by adopting the encoder-decoder architecture from the machine translation advances. This is done through utilizing the idea of the encoder-decoder architecture but replacing a RNN with a deep CNN to produce informative representations of the input image, and then pass the sequence of extracted features to the decoders to generate the caption. Also, using state-of-the-art sub-networks, which have been pre-trained, will help produce state-of-the-art results for the problem at hand.

### 3.1.3 – Results and Conclusions

The most commonly used metric has been the BLEU score, which is precision of word n-grams between generated and reference sentences. Due to recent advances, it is more meaningful to report BLEU-4 (4-gram comparisons), which is the standard in machine translation. Besides the BLEU score, perplexity and human evaluation were used to confirm the metrics. When evaluating captions against human evaluation, the model performs more poorly, but the model had a 27.2 BLEU-4, which is quite a good score for BLEU-4. But, due to poor performance for human evaluation, it was shown that BLEU is not a perfect metric for sequence generation tasks.

## 3.2 – Show, Attend and Tell: Neural Image Caption Generation with Visual Attention (2015) [1]

### 3.2.1 – Background

The main reason that caption generation models are difficult to design is that the model must be powerful enough to determine which objects are in an image, but they must also be capable of expressing object relationships in natural language. Due to advances in both convolutional neural networks to obtain representation of images and recurrent neural networks to decode those representations into sentences, the quality of generated captions has increased.

### 3.2.2 – Aim and Methodology

The aim of the paper is to suggest 2 approaches for effective caption generation that attempt to incorporate “hard” or “soft” attention. Soft attention is an approach that allows the use of standard back propagation as the model is kept differentiable by utilizing the attention mechanism introduced by Bahdanau et al. (2014) [13]. Hard attention introduces the use of a multinoulli distribution for intermediate latent variables. Also, due to attention, the goal is show how this benefits visualization for what the model “sees”.

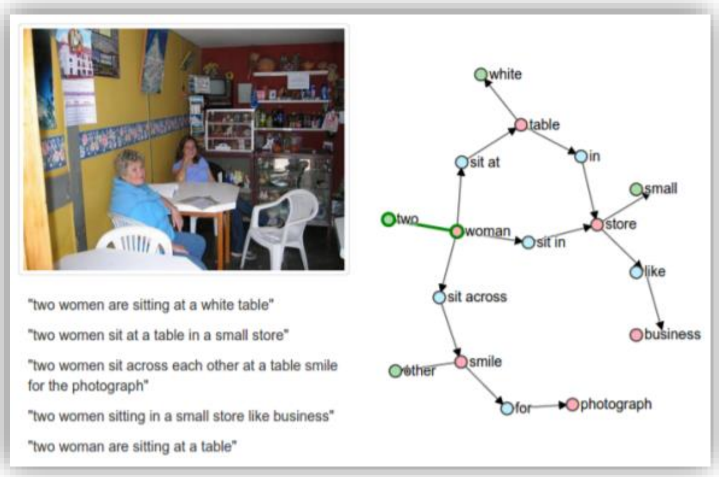
### 3.2.3 – Results and Conclusions

Flickr8k BLEU-4 score was 21.3, Flickr30k BLEU-4 score was 19.9, and COCO BLEU-4 score was 25.0 using Hard attention, while Flickr8k BLEU-4 score was 19.5, Flickr30k BLEU-4 score was 19.1, and COCO BLEU-4 score was 24.3 using Soft attention. It is recommended to use an attention-based approach as it gave state of the art performance on three benchmark datasets using the BLEU and METEOR metric at the time. This attention method also helped make the models more interpretable and showed that the learned attention corresponds well to human intuition. For example, an image of a man playing a violin, the attention vector puts a spotlight on the violin to visualize what the model is focusing on.

## 3.3 – SPICE: Semantic Propositional Image Caption Evaluation (2016) [4]

### 3.3.1 – Background

Interest in image captioning has been driven largely by the development of new benchmark datasets such as Flickr 8K, Flickr 30K and MS COCO, but existing metrics are unable to effectively measure performance. One major problem with metrics like BLEU, ROUGE, CIDEr or METEOR to evaluate generated captions, is that these metrics measure n-gram overlap, i.e., if a n-gram from the actual caption exists in the prediction, a high similarity score is produced. The problem with n-gram


"two women are sitting at a white table"
"two women sit at a table in a small store"
"two women sit across each other at a table smile for the photograph"
"two women sitting in a small store like business"
"two woman are sitting at a table"

sensitivity is not just scoring incorrect predictions well, but also scoring similar predictions poorly due n-gram matching.

### 3.3.2 – Aim and Methodology

Propose a new metric called SPICE specifically designed for caption generation, show how SPICE outperforms current metrics, and how SPICE can be used to conclude on questions like "Which caption generator understands colors best". SPICE disregards the image and makes the comparison of the generated caption to the available actual captions a linguistic task. The metric focuses entirely on the semantic meaning using "scene graphs". Scene graphs are generated for the actual captions and the predicted captions. Then, the similarity of the graphs is calculated using the F-score.

### 3.3.3 – Results and Conclusions

Captions were generated for all the benchmark datasets and all the metrics mentioned were calculated, including the SPICE score. Using human evaluation as the ground truth, scores were graphed against the human evaluation to see the correlation. SPICE significantly outperformed the other metrics by having a 0.88 correlation with the human evaluation scores compared to 0.43 for CIDEr and 0.53 for METEOR. Also, a human generated caption was provided to each metric to judge the scores assigned, for which SPICE also scored higher than the model generated captions unlike the other metrics. Therefore, SPICE is a novel way to capture human judgement and effectively score for model generated captions.

## 3.4 – Where to put the Image in an Image Caption Generator (2018) [12]

### 3.4.1 – Background

To transform a simple neural language model to a caption generator, the model needs to accept as input not only the current sentence input, but also the image that is being captioned. "This raises the question: At which stage should image information be introduced into a language model?" [12]. It is important to determine where the relationship between the image and the caption is strongest.

### 3.4.2 – Aim and Methodology

Explore at which step of the neural caption generator should the image features be inserted. There are 2 proposed methods of introducing the image in the architecture: conditioning by injecting and conditioning by merging. There are 2 problems being tackled here: identifying if early inclusion of image features helps the model learn better and understanding if changing image features after the prediction of each word to adjust the model focus impacts the model performance.

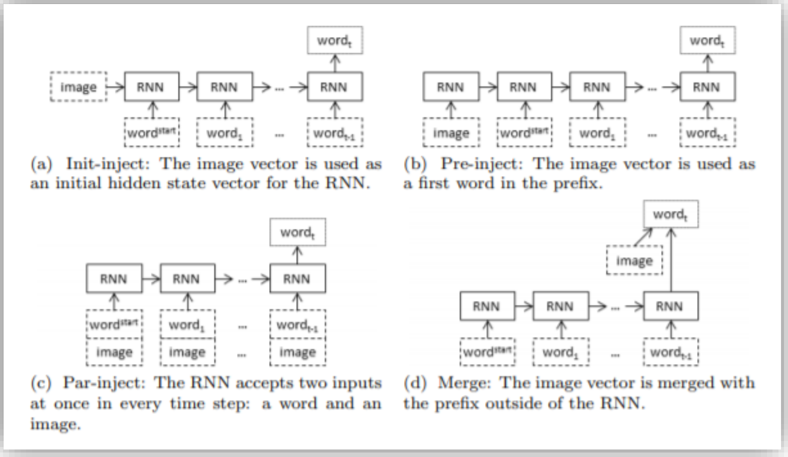


image
RNN
word$_t$
word$^{start}$
word$_1$
word$_{t-1}$
(a) Init-inject: The image vector is used as an initial hidden state vector for the RNN.
(b) Pre-inject: The image vector is used as a first word in the prefix.
(c) Par-inject: The RNN accepts two inputs at once in every time step: a word and an image.
(d) Merge: The image vector is merged with the prefix outside of the RNN.

### 3.4.3 – Results and Conclusions

Flickr8k, Flickr30k, and MSCOCO data sets were used to test the suggested architectures using the standard metrics, BLEU, ROUGE, METEOR, and CIDEr. On different datasets, different models performed better using the BLEU-4 score: Flickr8k par-inject architecture performed best, Flickr30k pre-inject architecture performed best, and MSCOCO init-inject architecture performed best. Overall, using the MSCOCO dataset with the init-inject architecture produced the best results. Despite best performance, there was very little difference between the performance of the models, but the inject architectures were proven to produce repeated captions.

## 3.5 – Image Captioning (2018) [5]

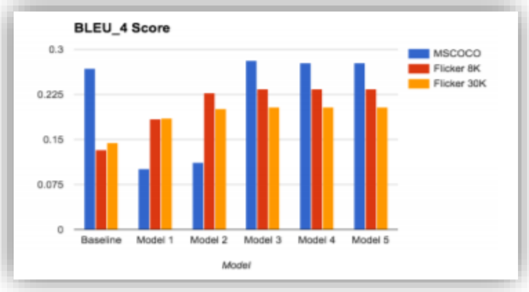
### 3.5.1 – Background

Using the research already done on image caption generators, the goal is to create an architecture to produce state-of-the-art results. The baseline model by A. Karpathy is used to improve the results. The baseline model consists of a feature extraction step using the VGGNet or ResNet architectures and then injecting this feature vector to the RNN. The two parts, CNN and RNN, are joined together by an intermediate feature expander, that feeds the output from the CNN into the RNN.

### 3.5.2 – Aim and Methodology

There were 3 changes made to the baseline model: transfer learning by using the baseline model and further training the model, increasing the number of hidden layers to 2 and 4, and using 101-layer deep ResNet instead of VGGNet. BLEU-4 and CIDEr evaluation metrics were used to test the model performance

### 3.5.3 – Results and Conclusions



BLEU-4 scores for the VGGNet with 2-layer RNN, VGGNet with 4-layer RNN, and ResNet with 1-layer RNN were higher than the baseline model. Captions that were very incorrect were hypothesized to be due to lack of training data, e.g., high co-occurrences of cake and knife but no co-occurrences of cake and spoon.

## 3.6 – Bottom-Up and Top-Down Attention for Image Captioning and Visual Question Answering (2018) [7]

### 3.6.1 – Background

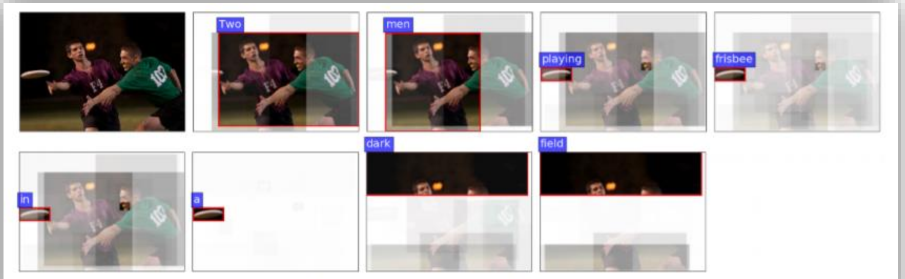
There have been difficulties in combining image and language for tasks like image captioning. In this task, it is required to extract features from the images, that not only describe the objects but the interactions as well. Recently, attention mechanisms have been widely adopted in this task to improve performance by focusing on specific parts of the image that are important. There are 2 types of attention mechanisms: attention driven by non-visual context (top-down), and visual feed-forward attention mechanisms as (bottom-up). Currently, only top-down attention is being used to solve the problems at hand.

### 3.6.2 – Aim and Methodology

Utilize both attention mechanisms (single head) and apply the architecture to the image captioning problem and visual Q&A problem. The R-CNN together with ResNet-101, an object detection model, is used to identify objects within each image using the bottom-up attention model. For the image captioning problem, the model uses a top-down attention mechanism with 2 LSTM layers. The first LSTM layer is considered as the top-down attention model and the second LSTM layer is the language model. The first LSTM layer uses the spatial image features generated by the CNN using the bottom-up attention model to generate an attention vector. This attention vector is concatenated with the image features and is fed to second LSTM layer for the language model.

### 3.6.3 – Results and Conclusions

The bottom-up and top-down attention models applied to the caption generation problem achieved higher BLEU-4 scores than the baseline model on the MSCOCO dataset. The previous best result was 35.6 but this model scored 36.9 on the same dataset, which is a state-of-the-art result. Besides the state-of-the-art results, a comparison was made to the pre-trained ResNet model and the attention model outperformed the ResNet model in object identification. Therefore, for image captioning problems, it is recommended to simply replace the pre-trained CNN features with pre-trained bottom-up attention features as shown in the paper.



Two men playing frisbee in a dark field.

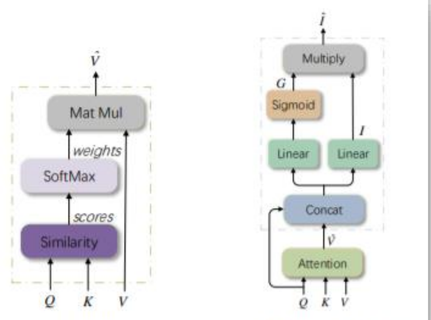
## 3.7 – Attention on Attention for Image Captioning (2019) [8]

### 3.7.1 – Background

Attention mechanisms have been used in current encoder/decoder architectures for visual captioning. An image is encoded to a set of feature vectors using a CNN based network and then decoded to words using an RNN based network, and the attention mechanism helps the decoder generate a weighted average of the feature vectors, helping capture global dependencies. But the decoder doesn't understand the attention result and how it relates to the query, which can produce worse results.

### 3.7.2 – Aim and Methodology

Attention on Attention (AoA) extends the normal attention mechanisms by adding another attention. First, AoA generates an "information vector" and an "attention gate" with two linear transformations. The AoA  is applied to both the image encoder and the caption decoder of the model. first calculates an importance score for each candidate. Instead of directly feeding these vectors to the decoder, a "refining network" is built which contains an AoA module to help



(a) Attention (b) Attention on Attention find interactions between objects in the image, and to measure how they relate. The RNN then takes the input of the embedding, current word, and a visual vector generated using the AoA module.

### 3.7.3 – Results and Conclusions

The MSCOCO dataset is used to test the suggested model and a BLEU-4 score of 37.2 is achieved. The AoA network also counts the number of objects more effectively and is capable to finding better interactions between the objects. For example, the model predicts the difference between “holding a tennis ball” as compared to “hitting a tennis ball”. The base encoder-decoder model predicts “A teddy bear sitting on a book on a book” while the suggested model predicts “A teddy bear sitting on a chair with a book”. To further solidify the results, 30 evaluators were given 100 randomly selected images with 2 captions generated using the AoA model and 1 caption from the base encoder-decoder model. The effectiveness of the AoA model was 49.15% while the base model was 21.2%. This model produced state-of-the-art results and should be applied wherever it is applicable.

## 3.8 – Controlling Length in Image Captioning (2020) [6]

### 3.8.1 – Background

Current captioning models learn using either a LSTM or transformer, which makes it difficult to control the generation. The length of the caption for these models is determined when the <eos> is generated. If the model can give the end users control of output length, it would make the captioning model more robust. By influencing the length, there can be different types of captions generated, i.e., short and simple captions, and long and detailed captions for the same image.

### 3.8.2 – Aim and Methodology

2 models were trained and tested: LenEmb and Marker. The LenEmb controls the length, using an injection method to insert the caption length into the model. Given the current desired length, the remaining length is embedded into a vector that is the same size as the word embedding. Then the word embedding of the previous word is added with the generated length embedding, which is then fed as the input to the LSTM. On the other hand, the Marker model inserts the desired length as a token as the first word, instead of <bos> (beginning of sentence). The model needs to learn to predict the length using the first passed token, which removes the requirement to do anything else to generate a caption with desired length. If no token is provided for length, the model should be capable of producing a result using the normal process.

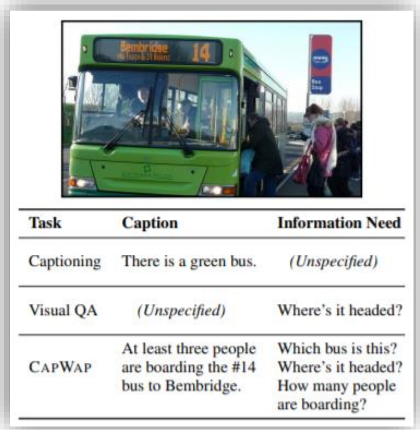
### 3.8.3 – Results and Conclusions

The results of both suggested models were below the baseline model by 0.7 using the BLEU-4 score. Despite that, overall the Marker model performed better than the LenEmb model. Despite the overall performance being worse, the LenEmb model performs better as the caption length increases to 28. Summarizing the results, the suggested models perform better than baseline model when length < 10, but length between 10 - 16 which are the most common lengths, the baseline model performs better.

## 3.9 – CapWAP: Captioning with a Purpose (2020) [10]

### 3.9.1 – Background

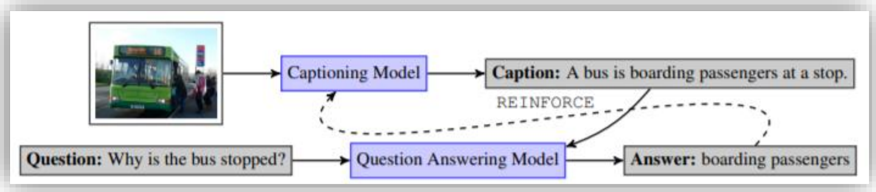
The main issue with image captioning problems are that they are generic, and might not be relatable to all users, as different users will care about different parts of an image. For example, one user might want to emphasize the morning sunrise at a beach while another user might emphasize the waves at the beach. Considering the idea of using question and answering systems, it might make sense to answer user questions about an image rather than simply providing a generic result.



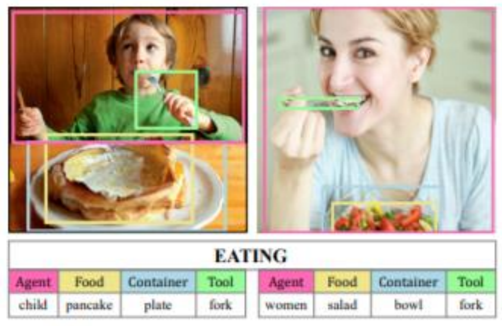
| Task | Caption | Information Need |
|---|---|---|
| Captioning | There is a green bus. | *(Unspecified)* |
| Visual QA | *(Unspecified)* | Where's it headed? |
| CapWAP | At least three people are boarding the #14 bus to Bembridge. | Which bus is this? Where's it headed? How many people are boarding? |

### 3.9.2 – Aim and Methodology

The proposed model for captioning is the Captioning with a Purpose (CAPWAP) model. The goal is to map images to text using answers to possible questions for the given image. For predictions, the model should be capable of assuming user questions for a new image, implicitly answer them, and generate a description. Therefore, the CAPWAP model is given an image, questions and answers to learn possible questions that can asked about a given feature vector and generates a caption that answers these questions. Since an existing model cannot be adopted to apply to this approach, a reinforcement learning approach is employed where the captioning model receives a reward if the generated caption helps the Q&A model answer the question.


Captioning Model
Caption: A bus is boarding passengers at a stop.
REINFORCE
Question: Why is the bus stopped?
Question Answering Model
Answer: boarding passengers

### 3.9.3 – Results and Conclusions



| EATING | | | | | | | |
|---|---|---|---|---|---|---|---|
| Agent | Food | Container | Tool | Agent | Food | Container | Tool |
| child | pancake | plate | fork | women | salad | bowl | fork |

**Figure 2:** Two image examples of a verb and its semantic roles. The verb `eating` captures the scope of the activity, and `agent`, `food`, `container`, `tool` are all reasonable semantic roles for this activity.

The 2 metrics used to evaluate the models are EM (exact match) and the F1 score. There were 4 baseline models generated to test the suggested model against: human reference to show how closely provided captions can answer the questions, 2 other published models, and a generic baseline model. There were 4 datasets, each of which contain questions, answers, and images (MSCOCO) in the data, used to test the models: CapVQA, CapGQA, CapVisual7W, and CapVizWiz. The suggested model performs much better as compared to the other models in all datasets for both metrics. For example, the suggested model produced an EM of 24.2 as compared to 16.8 (max of other models) for the VQA dataset. The main goal of this model is to uncover the information requirement in understanding the image and generating the caption, which is accomplished using the Q&A approach.

## 3.10 – Human-like Controllable Image Captioning with Verb-specific Semantic Roles (2021) [9]

### 3.10.1 – Background

Due to recent release of several large datasets and more research on encoder-decoder architectures, captioning models have produced “super-human” results in accuracy-based evaluation metrics. But papers have suggested that these models produce generic descriptions and are unable to generate captions like humans by referring to different interests or parts of the image. This is done by designing different control signals used as constraints to generate captions, called Controllable Image Captioning (CIC). Premature CIC models focused on subjective constraints like sentiment and emotions, while recent CIC models focus on objective constraints. There are 2 types of objective constraints: content-controlled and structure-controlled. Content-controlled constraints are signals provided that include the contents of interest, e.g., man, wave and surf, while structure-controlled constraints are signals that include semantic structure of sentences, e.g., length and POS.

### 3.10.2 – Aim and Methodology

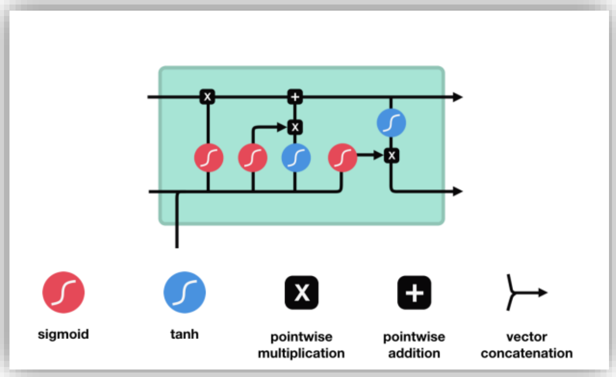
Despite the progress of CIC models, there are 2 control signals that are ignored: Event-compatible (contents referred to in a single sentence should be compatible with the described activity) and Sample-suitable controls (control signals should be suitable for the specific image). To capture these 2 missing signals, an event-oriented control signal: Verb-specific Semantic Roles (VSR). Before a caption model can be generated, a semantic role classifier must be trained and a planner to rank the verbs and semantic roles. Since no data is available with semantic roles, semantic parsing toolkits were used to generate annotations. Then, in the image caption generator, 2 LSTM layers are used to keep track of shifting semantic roles. The datasets used were the MSCOCO and Flickr30k.

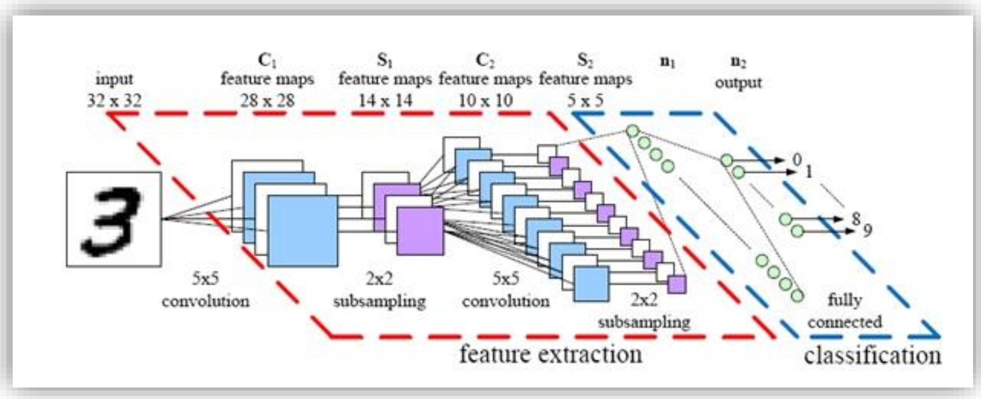
### 3.10.3 – Results and Conclusions

There were 3 baseline models used to compare against: C-LSTM (controllable LSTM), C-UpDn (controllable UpDn – attention model), and SCT (visual regions as a control signal). The proposed model performed better than the given baselines with a BLEU-4 score of 23.1 compared to 18.1 (max of baselines) for the MSCOCO dataset, and 10.7 compare to 10.1 (max of baselines) for he Flickr30k dataset. The paper proved that there have been control signals looked over in past papers but it is needed to further pursue this idea past verb specific semantic roles to better generalize the framework.

## 4 – Technology Utilized and Issues

All the work was done in JupyterLab using a CPU, which caused the most issues for this model as it made testing more difficult. Running 3 epochs took almost 1 hour 30 minutes, which would have to be done every time a new configuration had to be tested. Setting up the Keras GPU library would have been a better option as it would allow for much faster training. I also attempted to use Google Colab, but the free version was performing slower than my machine. As for software, I used Python 3.7 with NumPy, listdir to load image files, pickle to save image features, TensorFlow 2.4.1 and Keras 2.4.0 for applications (VGG19), preprocessing, models, and layers, and NLTK BLEU score for measuring performance. The reason this kind of problem is solvable are the available layers in Keras: LSTMs and CNNs.





Aside for the processing power issue, I also noticed a lack of data. Despite there being 5 captions per image and 85% of the dataset being used as training data, it just was not enough data to train a well performing model. The lack of data was seen in the images and in the test: lack of different types of scenarios in images and lack of vocabulary in the captions. For example, if an image of a cat is given to the caption generator, the decoder never matches the features to a cat and is always predicted as dog. Also, the model overfits due to a lack of data. This can be seen in the test set where similar predictions on similar features are noticed, i.e., any furry animal is predicted as dog, any blue background is predicted as water, any brown background is predicted as beach, and any green background is predicted as grass.

## 5 – Data Preprocessing

Data preprocessing was very limited for this task and contained simple steps to prepare the information for the model.

First, the image was loaded using the Keras library function “load_img” and then the image was converted to an array using the “image_to_array” function. Once that was done, the image was in a 3D array, 224 x 224x 3, which is then passed to the “preprocess_input” function. This function prepares the image array to be passed into Keras models, i.e. turns the image to a 4D array (1, 224, 224, 3) and scales the pixels -1 to 1. Once this has been done, features are extracted from the “fc2” layer of the VGG19 architecture trained on the ImageNet dataset, which is a 1D vector of 4096 length. Using the second last fully connected layer allows the VGG19 model to extract as much details from the image as possible using the convolution blocks in the architecture.

Second, the text had to be cleaned by removing punctuation and special characters using regular expressions. After cleaning, the tokens “<START>” and “<END>” were added to each caption at the beginning and end respectively. This allows the machine to distinguish where the caption starts and ends. Using the Keras tokenizer, the captions are tokenized and then sequenced using the generated vocabulary from the tokenizer. Then, the maximum length of captions was identified for padding using the "post" parameter, meaning all padding is done after the caption.

Lastly, the output is a single word, next word, represented as a sparse vector of vocabulary size, where 1 is the index of the predicted word in the vocabulary and 0 otherwise.

Using the 2 preprocessing steps, there are 2 input sequences and 1 output word generated:

| X1 | X2 (sequences, equal length) | Y (sparse) |
|---|---|---|
| Image features (1 x 4096) | [<START>, 0, …, 0] | A |
| Image features (1 x 4096) | [<START>, A, 0, …, 0] | jeep |
| Image features (1 x 4096) | [<START>, A, jeep, 0, …, 0] | sideways |
| Image features (1 x 4096) | [<START>, A, jeep, sideways, 0, …, 0] | on |
| Image features (1 x 4096) | [<START>, A, jeep, sideways, on, 0, …, 0] | some |
| Image features (1 x 4096) | [<START>, A, jeep, sideways, on, some, 0, …, 0] | rocks |
| Image features (1 x 4096) | [<START>, A, jeep, sideways, on, some, rocks, 0, …, 0] | <END> |

Array of sequences X2 is shown in text form to visualize the data inputs. Therefore, from 1 image and caption pair, there are 5 training samples generated.

# 6 – Model and Results

## 6.1– Architecture Design

### 6.1.1 – Why is Deep Learning Required

Using deep neural networks for any task is a resource intensive approach, hence the direction must be justified. To design an architecture for the image caption generator, it is required to look at both parts of the architecture: Image Feature Extraction and Description Generation.

To extract features from images, it is possible to use dimensionality reduction to transform the 224 x 224 x 3 image, i.e., 150,058 fields to a 1D vector of size 4096 using PCA or LDA. Another method is to use a descriptor like BRIEF (Binary Robust Independent Elementary Features), which is power method to extract pixels and neighboring pixels to extract image features. According to the paper A Comparison of CNN and Classic Features for Image Retrieval [11], from the available handcrafted descriptors, SURF had the best overall performance and was competitive with the top CNN approaches. But it can be difficult to do some tasks with a CNN as compared to descriptors, e.g. image rotation. Using the most recent VGG19 pre-trained model will prove to be more effective for quick image feature extraction.

Sequence generation has become more popular since the encoder-decoder neural architecture showed promising results. There are other probabilistic solutions for language modelling that help generate linguistic rules to learn the distribution of words, e.g., n-grams or Hidden Markov Models. For example, n-gram language models can be used to generate grammatically correct, but incoherent, sequences. Therefore, encoder-decoder architectures have proven to be more robust in generating sequences, despite the issue of input context. Since the introduction of Seq2Seq models using the encoder-decoder architecture, there have been a few major improvements in sequence generation, two of which are attention and transformers. Attention and transformers focus on solving the issue of loss of information as the Seq2Seq model progresses forward in long sequence generation, despite the use of LSTMs.

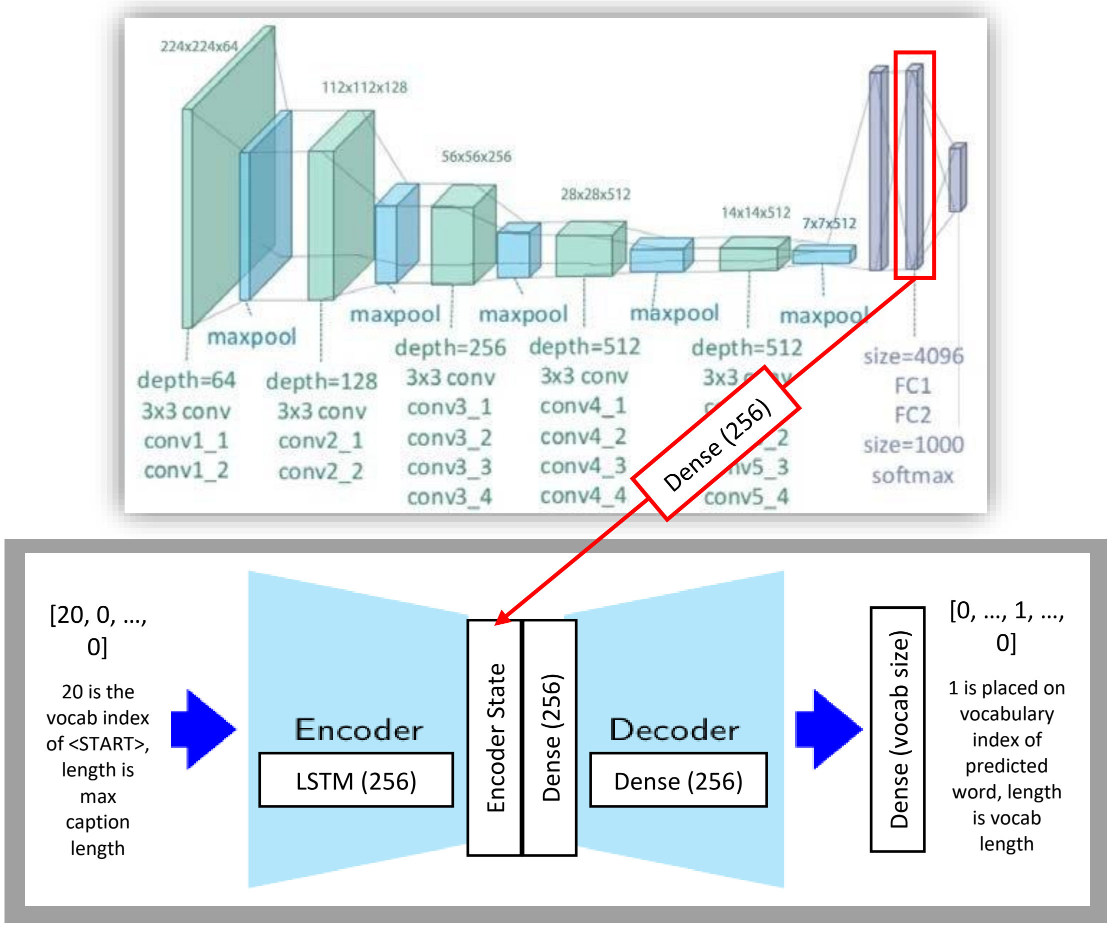
### 6.1.2 – Neural Network Architecture

The network architecture consists of 3 separate sections: image feature transformation, text input encoder, and decoder.

First, the image feature transformation is done through taking the 1D vector of size 4096 (X1 shown above) as input and applying a dense layer with 256 hidden units as this vector will be used in conjunction with the 256 hidden unit output from encoder. A dropout layer was used for regularization with 0.5 as the rate prior to passing the input to the dense layer. This helps reduce the chances of overfitting the model.

Second, the encoder receives an input of the size of the maximum length of the captions (X2 shown above) and an embedding is generated using the vocabulary size. A dropout layer was also used in the encoder to prevent overfitting. Then, the LSTM layer was used with 256 hidden units as it produces state-of-the-art results for sequence or time-series data.

Both the transformed image vector and the encoded input sequence vector are added using the add layer. Then a dense layer is applied with 256 hidden units using the relu activation function. Once the decoder has combined the image and textual features, a dense layer is applied to produced vector using a softmax activation function with length of vocabulary size.



## 6.2 – Hyperparameter Choices

Hyperparameter choices were relu activation for all layers except the output layer which was softmax. All layers used 256 hidden units, except the output layer which generated an output with vocabulary size. The loss used was the categorical crossentropy and the optimizer was adam. Other configurations like changing hidden units or adding additional layers does not affect the cost because there are larger issues with this dataset, i.e., the size, lack of vocabulary, and lack of types of images.

## 6.3 – Results and Comparison to Current Work

The model was run for 15 epochs with training (85%) and validation sets (15%). The model overfit after 4 epochs as the validation loss stopped around 3.7, but the training loss was decreasing, i.e. the model was overfitting. Accuracy was not used because it didn't provide much information about the model performance. For example, the actual caption is "The cat is running in the grass" and the predicted caption is "The dog is running in the grass", with an accuracy of 6/7, but the context is incorrect.

To use the model to make predicted on new images, use the functions from Keras to load image, transform image to array, preprocess array, and predict image features from VGG19, returning the fc2 layer. Then, initialize the predicted caption with “<START>” and each prediction will be added to this caption until the maximum caption length is reached or the “<END>” token is seen. In the loop, tokenize the predicted caption so far and pad the caption to the maximum length. Use the image fc2 layer output and the padded sequence as inputs to the model, which outputs a vector of length of the vocabulary. Use the argmax function from the numpy library to find the index that has the highest value and locate the word in the vocabulary. Add this predicted word to the predicted caption.

BLEU scores are a better metric to evaluate caption generation as it is capable of creating a comparison using n-grams. For comparison, BLEU scores compared to paper with suggested scores for skillful models: Where to put the Image in an Image Caption Generator [12]. The paper suggests that using the Flickr8k dataset, the following BLEU scores are expected compare to my model performance using 15% of the data as test data:

| BLEU | Expected Score | Actual Score |
|---|---|---|
| BLEU-1 | 0.609 – 0.611 | 0.539 |
| BLEU-2 | 0.421 – 0.424 | 0.287 |
| BLEU-3 | 0.285 – 0.287 | 0.195 |
| BLEU-4 | 0.190 – 0.191 | 0.090 |

BLEU scores are calculated for 1, 2, 3 and 4-grams. BLEU-1 calculates how many of the same words are present in the predicted caption. BLEU-2 does the same for bi-grams, BLEU-3 does the same for tri-grams, and BLEU-4 does the same for 4-grams. The issue with this metric, as the accuracy, is that it does not consider order. This is the reason BLEU-4 is a popular metric for comparison as it looks at 4-gram comparisons between captions, so order is accounted for to some extent.

The best performance currently is produced in the paper: CAPWAP Captioning with a Purpose [10]. The model proposed produces a more descriptive caption as compared to the given caption. For example, the given caption is “A man playing tennis” and the predicted caption is “a man in white shirt and white shorts playing a game of tennis on a grass court” which are all accurate descriptors. Furthermore, they also connected the model to a question-answering model to reinforce the captioning model to train the model.

## 7 – Lessons Learnt

The major lesson learnt from this project was that the processing power required to do image feature extraction and sequence generation is quite high. A GPU would be very useful to run concurrent processes for faster training. Also, the requirement of data is very important to train neural networks, especially for sequence generation. For example, if a word was not seen in the training set or was only seen once, chances are that the predictions would never include those words. To tackle the issue of data, there are larger datasets like Flickr30k or MSCOCO, which help produce much better results for image caption generation. Another approach that could be used to better the prediction results would be use pre-trained BERT embeddings for the encoder instead of generating an embedding using the training data. Besides the issue with sequence generation, there were also gaps in the image feature extraction. A suggestion to combat the issue of feature vectors without rich information is to combine image feature extraction models: Pre-trained CNN Model and Pre-trained Object Detector. For example, combining VGG19 and a pre-trained model like YOLO would provide more information about the details and actions being performed in the images. Lastly, a simple method to prevent overfitting to the training set would be to conduct hyperparameter tuning, specifically learning rate, optimizer and regularizer.

Besides the generation of the model, the literature review provided insight into current research areas in image captioning. This combination model of image analysis and sequence generation has become a very hot topic in research. New architectures are introduced every day and ideas are published to solve nuances in generating human-like captions. Recently, integrating attention mechanisms, control parameters, and Q&A systems has helped produce state-of-the-art results for the image captioning problem.

## 8 – References

[1] https://paperswithcode.com/paper/show-attend-and-tell-neural-image-caption
[3] https://paperswithcode.com/paper/show-and-tell-a-neural-image-caption
[4] https://paperswithcode.com/paper/spice-semantic-propositional-image-caption
[5] https://paperswithcode.com/paper/image-captioning
[6] https://paperswithcode.com/paper/controlling-length-in-image-captioning
[7] https://paperswithcode.com/paper/bottom-up-and-top-down-attention-for-image
[8] https://paperswithcode.com/paper/attention-on-attention-for-image-captioning
[9] https://paperswithcode.com/paper/human-like-controllable-image-captioning-with
[10] https://paperswithcode.com/paper/capwap-captioning-with-a-purpose
[11] https://arxiv.org/pdf/1908.09300.pdf
[12] https://arxiv.org/pdf/1703.09137v2.pdf
[13] https://arxiv.org/pdf/1409.0473.pdf