Image Captioning with Sequence Encoding/Decoding

Boyu Zhang (SUNetID: bzhang99) Mayukh Roy (SUNetID: rmayukh)

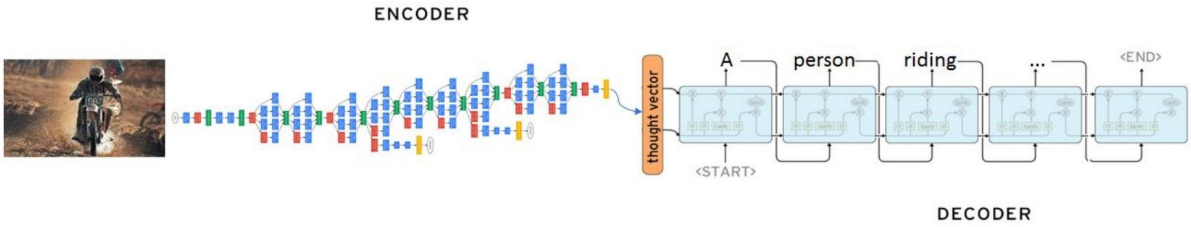
1. Abstract:

In this paper, we are implementing an Image Captioning Model which will convert any input image to a descriptive sentence. Image Captioning can be used in various applications such as a cellphone application to describe surrounding scenes for blind or visually impaired people. We can also use this model to generate video captions for IP cameras which can be used to detect abnormal activities and raise alarm immediately. Searching photos within album app can also benefit from this model since it will be as easy as searching keywords among sentences. In this model we use an Mask-RCNN Encoder to detect images instead of a regular CNN such as ResNet or Inception as found in most of the baseline models and prior works. Image detection allows us to identify different parts of the image and this is very useful since we are also implementing an Attention Model which can then focus on these individual detected objects instead on the whole image. Using this attention model along with GRU decoder we can then generate the sentence associated with the image. We also use Beam Search in our prediction model to achieve improved results and the results were measured having BLEU Score as the evaluation metric. Using our model we were able to improve on the Baseline model and achieve better BLEU Score.

2. Related works:

Encoder and Decoder based captioning

Inspired by machine translation, a typical way to do image captioning is to use an encoder and decoder model. The encoder is a CNN which converts input picture to a feature vector. Decoder is an RNN which then generates the captions based on image feature vector. [1]



(*Figure 1: traditional encoder decoder model*
source:https://medium.com/mlreview/multi-modal-methods-image-captioning-from-translation-to-attention)

Attention based captioning

The problem of the previously described model is the feature vector that CNN generated represents the whole input picture. As a result when the RNN is generating each individual word in the sentence, it is looking at the whole picture. This is inefficient and can also sometimes lead to wrong results. In Xu's paper [2], this problem was mitigated by adding an attention layer between encoder and decoder. The attention layer is a small neural which can learn where to focus in the picture when RNN is generating the target word.

Sequence based captioning

The other problem of the base CNN+RNN model is the imbalance between the representation of the image and the representation of the words. The RNN decoder is generating the caption sentences in different time steps, however the CNN feature of the image is one time step only. In Chang's paper [3], in order to fix this problem, an image is converted into a sequence of detected objects using object detection model. The encoder is changed from CNN to R-FCN. In this way, a

sequence of objects can be translated to a sequence of words using the same model as machine translation.

3. Dataset:

Our dataset is from COCO, where we are getting large scale of images along with multiple captions corresponding to each image. We had a total of 406,095 data points (image, caption pair) with 81219 unique images in total. 80% of the total data was allocated to the training set while 10% of it was used as the validation set and the remaining 10% was assigned to test set. The training set consisted of images with multiple captions while validation and test sets only had unique only image/caption pairs to avoid getting duplicate results. We also used 80/10/10 split instead of 60/20/20 to ensure the training sets represent a wider range of images so the model can train and identify as many objects from the images. Here are some examples of images and captions:

Image A

Image B

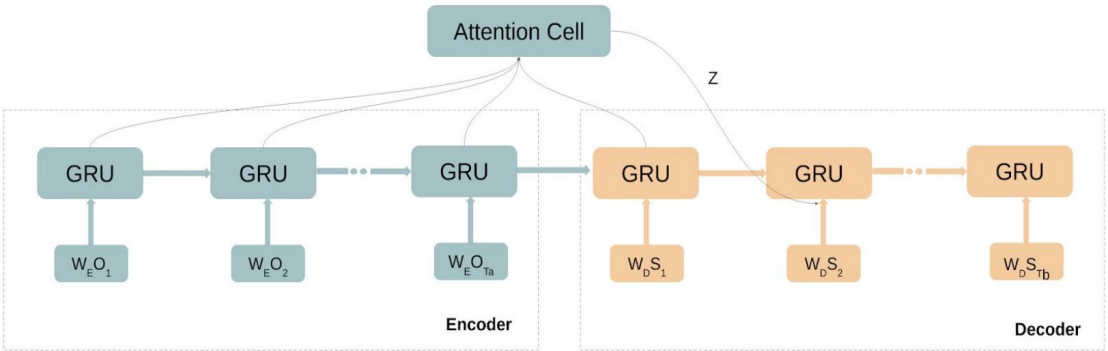
Image C

<start> Children are playing baseball on a muddy baseball diamond <end> (Caption A1 for Image A)
<start> A purple bus and a man dressed as a nun on a tall bicycle <end> (Caption B1 for Image B)
<start> A child waits for a pitch during a game of baseball <end> (Caption A2 for Image A)
<start> a wine glass sits in front of some plates of food <end> (Caption C1 for Image C)

The <start> and <end> token in the example above were added as part of the pre-processing steps. All the caption sequences were padded so that they have the same length. For the images, the preprocessing steps included resizing them to 299 pixel by 299 pixel and then normalizing them so that they contain values between -1 to +1.

4. Model:

In our project, we follow a similar idea of sequence based captioning, and added an attention layer between two RNN networks to increase accuracy.

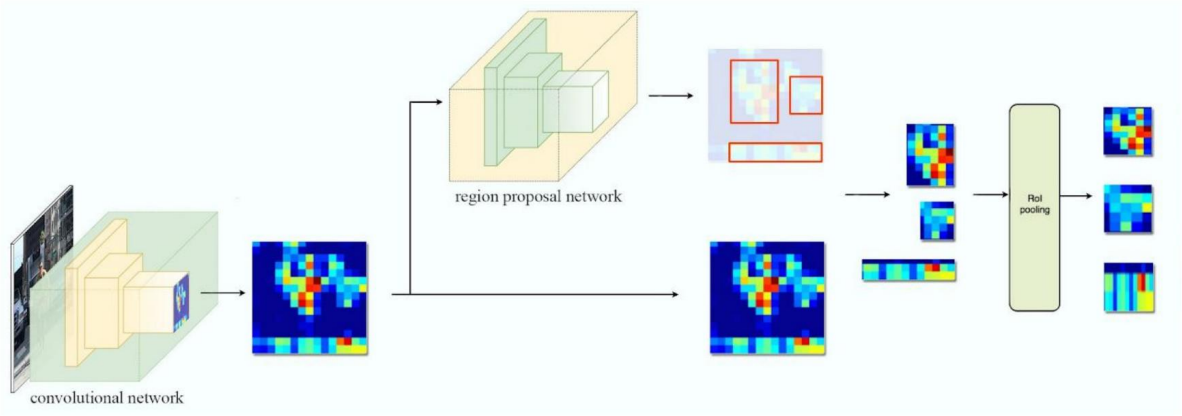


(Figure 2: Encoder/Decoder of our model)

Sequence representation of images

In order to get a sequence representation of the input image, we run object detection on it. The object detection model we chose is Mask-RCNN. It is a region based object detector. The model first uses a CNN to extract CNN features from the input image, then it uses an RPN(region proposal network) to determine the region of interest(ROIs). These ROIs are then combined with the corresponding CNN feature map to form patches for object detection. Using ROI pooling these patches are converted to have the same dimension. The patches are then passed onto fully

connected layers to obtain their object category and confidence level. We keep all the patches with confidence level higher than 0.7 and form a sorted list based on the confidence level. We also add the CNN feature of the whole input image at the end of the list. The list of these patches of dimension 7*7*256 forms the sequence representation of one input image. We limit the number of detected objects to be 40.



(Figure 3: Mask-RCNN model
source :https://medium.com/@jonathan_hui/image-segmentation-with-mask-r-cnn-ebe6d793272)

The image sequence can be represented as the following sequence.

$$W_e O_t, t \in \left(1, 2, \ldots, T_A\right)$$

Here W_e is the embedding matrix, O_t is the CNN feature of t-th object given by the Mask-RCNN model. T_A is the maximum length of the input sequence.

Sequence representation of the captions

We tokenize all captions by providing each unique word an index including the <start> and <end> which were mentioned in the Dataset preprocessing step. The target caption sentences are presented as follows:

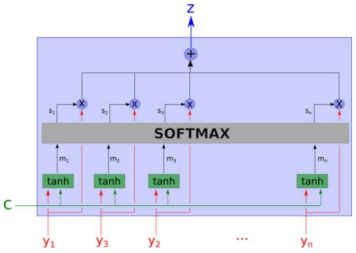
$$W_d S_t,\ t \in \left(1, 2, \ldots, T_B\right)$$

Here W_d is the embedding matrix, S_t is each word in the caption represented by a one-hot vector with dimension of total vocabulary size. T_B is max length of all captions.

Encoder and decoder with attention layer

We use two GRUs as the encoder and decoder for our model to translate input sequence to output sequence. GRU was chosen over LSTM because it was simpler and faster to train.

The attention mechanism we use is bahdanau attention[4]



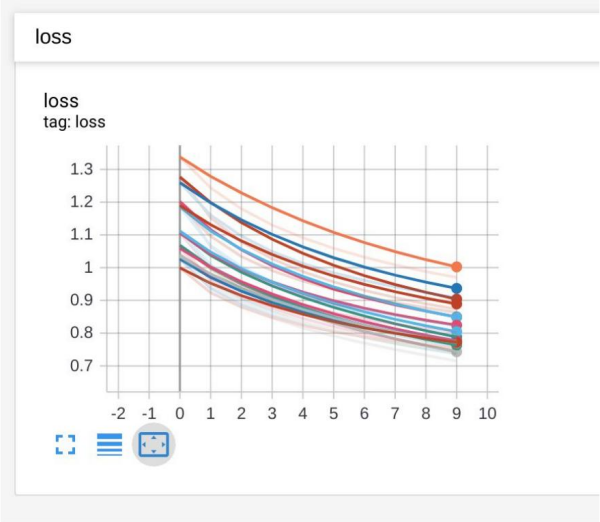
(Figure4: Bahdanau Attention
source: *https://blog.heuritech.com/2016/01/20/attention-mechanism*)

Pseudocode is the following:

score = FC(tanh(FC(encoder_output) + FC(hidden_state)))
attention weights = softmax(score, axis = 1)
context vector = sum(attention weights * encoder_output, axis = 1)

5. Hyperparameter tuning:
Following hyperparameters were tuned: mini batch size, learning rate and hidden unit size. We use a grid search to try all possible combination between these hyperparameters. We tried all 36 parameter combinations with below values:
Mini-batch size: 16, 32, 64, Learning rate: 0.1, 0.01, 0.001, 0.0001, Hidden unit: 256, 512, 1024

Plot of all 36 combinations using Tensorboard (y-axis: Loss, x-axis: #of Epoch):

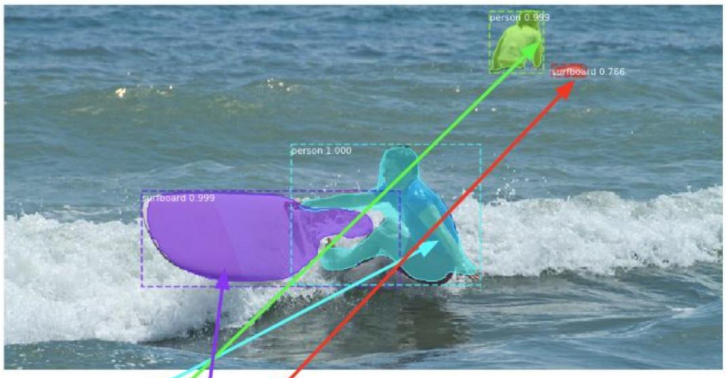


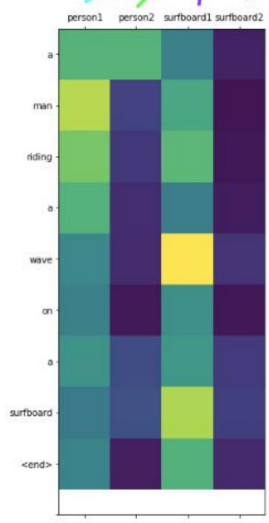
The combination of (Mini-batch = 64 Learning rate = 0.001 hidden unit = 512) gave the lowest loss after training over 10 epoch.

To avoid overfitting, we also tested number of epoch from 0 to 20. After each epoch we calculated the bleu score of the output sentences using validation set. After epoch 18, the average bleu score of validation set starts to decrease, so we stop training at epoch 18.

6. Results and Analysis:
We calculated the average BLEU score of the generated caption compared to reference captions across all the test sets for our model and the baseline model. As seen in the table below, our model outperformed the baseline model.

Model	**Bleu_1**	**Bleu_2**	**Bleu_3**	**Blue_4**
Baseline model with CNN encoder	0.495	0.337	0.221	0.145
Our model	0.683	0.514	0.371	0.264





Attention plot

Predicted sentence:

Greedy search:
a man riding a wave on a surfboard

Beam search: k = 3
a man riding a surfboard in the ocean

Beam search: k = 8
a man on a surfboard riding a wave

Reference:

1. A young man riding a wave on a boogie board.
2. A man riding on top of a wave with a surfboard.
3. A young man having fun riding a wave to the shore.
4. A man surfs on his surfboard in the water.
5. A boy is falling off of a surfboard in the water.

Also shown in the left is the output of Mask-RCNN model. We print out the attention matrix at the time our model was generating captions for this image. The output sentence is “a man riding a wave on a surfboard”. From the attention matrix, we can see when the model is generating word “man”, it pays more attention to the first detected object “person1”, and when it is generating the words “wave” and “surfboard”, it pays more attention to the third detected object “surfboard 1”. The model doesn’t pay attention to the person and surfboard in the background. This is a desired behavior, since in all five reference captions none of them mention the other person or the surfboard in the background.

7. Future works:

In this project, we are inspired by Chang’s paper to change the traditional CNN encoder to an object detector encoder. In this way, the input of the model is not the input image itself, but a sequence of objects detected from the image. We also add a attention layer between encoder and decoder to make sure the decoder is using the right object to generate the right word. Although our model has better BLEU score than the baseline model, it is not as good as other models on COCO leaderboard. In the future we can improve our model in the following ways:

1. The Mask-RCNN model we use can only detected 80 kinds of objects. We can re-train the Mask-RCNN to detect more objects.
2. Number of objects vary from image to image. We can group images having similar number of objects and train in buckets instead of padding all input sequences to have same length.

8. Contributions:

Each group member contributed to the project equally.

Github repository: https://github.com/spacetaro/image_caption

Reference:

1. [1] Vinyals et al. (2014). Show and Tell: A Neural Image Caption Generator. [Online] arXiv: 1411.4555
2. [2] Xu et al. (2015). Show, Attend and Tell: Neural Image Caption Generation with Visual Attention. [Online] arXiv: 1502.03044

3. [3] Liu et al. (2017). MAT: A Multimodal Attentive Translator for Image Captioning. [Online] arXiv: 1702.05658
4. [4] Bahdanau et al. (2014). Neural Machine Translation by Jointly Learning to Align and Translate. Online] arXiv: 1409.0473