Image captioning with region attention. Fine-tuning with Reinforcement Learning

Guillaume Petit
g.petit98@gmail.com

Thomas Fauré
thomasfaure87@gmail.com

January 22, 2020

Abstract

The objective of this report is to show the different techniques of image captioning, from the most basic to the most advanced. Our work and results will be carried out using the MSCOCO data set and we will try to evaluate the performance of each of these methods.

1 Introduction

Computer vision has been undergoing a permanent and impressive evolution for years, as with labelling or classification. Another field of study is that of image captioning, the objective of which is to describe a photo by a computer.
Recent improvements in recurrent neural networks (RNN), combined with advanced computer vision techniques, have allowed for the efficient development of image captioning. Image captioning has for input an image and for output a caption, a sequence of words. The CNN will process the image to detect areas of interest, and the RNN will aim to describe the picture from the areas of interest. An improvement has been made by using an RNN Faster, allowing better results. The last idea was to introduce a Reinforcement Learning mechanism in the learning phase, giving a better legend.

2 Methods

The three different models have similarities in the way they work. First, there is the encoder, which takes an input image and outputs a set of features from it. Then there is the decoder, which takes the features of the encoder as input and goes with an attention system (area of focus) to generate the next word. See fig. 3 at the end of the report for a schematic.

2.1 Top-Down Attention

In this model, the image is encoded using a convolutional neural network to extract features. Each feature corresponds to a region of interest in the image.

To generate the sentence we use a recurrent neural network. The challenge here is to remember the words generated at the time $t-2$, $t-3$, ... and not only the last word generated. Let's take for example a sentence where the subject is not directly followed by the verb. To conjugate the verb to the right person, we must remember the subject which, in this case, does not directly precede the verb. We can also use this reasoning to remember indices about when the action take place that may have been generated at the beginning of the sentence to know which tense to conjugate the verb in. The architecture of the LSTM is designed to solve this problem. In addition to reusing its last output as an input for the next iteration, the LSTM will incorporate a cell that keeps the information we have accumulated since the beginning. In each iteration, the network chooses which information to keep and updates the cell with the new information given as input so that the network memorizes in several iterations.

To help the model create the sentence we will define a context vector to tell the algorithm where to focus on the image. The contect vector weights the regions of interest

in the image to help the algorithm generate the next word. Here we choose to use the Attention soft to calculate it.

2.2 Bottom-up

Here we take inspiration from the human visual system to modify the first model. Man is able to voluntarily scan the image from top to bottom in search of a precise element in his field of vision (Top-Down) and to raise his gaze following a new or unexpected stimulus (Bottom-Up).

Here the Bottom-Up is a Faster R-CNN that will find interesting objects in the image. The Top-down will define where to focus its attention using the objects found by the Faster R-CNN rather than regions of interest. See Figure 4.

2.3 Fine-tuning with RL

We change our vision and consider the RNN as an agent who must choose the next word (set of actions) to maximize his reward which is the score of the metric used for example. The environment is the set of words and features of the image and the policy is the proba to choose the next word (which depends on a θ parameter).

Before, we used to optimize according to the cross-entropy loss :

$$L(\theta) = -\sum_{t=1}^{T} log\ p_\theta(y_t^*|y_{1:t-1}^*)$$

where y_t^* is the word generated at time t by the bottom up model.

With the new model, the loss function to be optimized becomes :

$$L(\theta) = -\mathbf{E}_{w^s \sim p_\theta}[r(w^s)]$$

where w^s a legend and r the reward.

Two changes should be noted. The first one is that the reward that comes directly into the optimization process depends on the metrics . In Bottom-up, the optimization was based on cross-entropy since it was not possible to optimize directly with respect to the metrics of interest (BLUE, RED...) since they are discrete, therefore not differentiable. Secondly, there is a change of scale in the optimization. Indeed, we stop optimizing in relation to the probability of generating the real word at time t knowing the old ones to optimize in relation to the score given by the whole sentence. This avoids a mismatch between the training and test part where an accumulation of errors results because during the test part, the computer has never been confronted with its own predictions.

3 Results

Our results are trained on the MSCOCO database. This database contains 328,000 images representing 91 different object categories, 82 of which are labelled with more than 5,000 words. In total we have more than 2,500,000 labels on our images. COCOCO has fewer categories than ImageNet but in return each category has more labels. This makes it easier to learn how to accurately detect objects in 2D.

Moreover each image is describe by 5 captions.

3.1 Quantitatif results

There are several metrics in NLP to evaluate the accuracy of a sentence. Unfortunately, not all of these metrics are equally reliable for classifying images. We will focus on BLEU and CIDEr. On one hand the BLEU-ngrams metric looks at the proportion of ngrams that match between the desired caption and the one generated by the captioning image model. On the other hand CIDEr is based on consensus. The model developed is supposed to allow the evaluation of a sentence based on its "resemblance to a human sentence". The generated sentence is compared with what a human could have said to describe this metric. The closer the result is, the higher the score. Here it will be compare to the 5 captions used to describe each image in COCO but lots of authors are wondering if 5 captions are enough compute the CIDEr score.

Using a Faster R-CNN to detect objects of interest, the algorithm knows directly which objects to consider rather than a region that is not labelled as in Top-Dpwn. It is therefore not surprising that the use of a Faster R-CNN significantly increases the results. Our results are not as good as those of the research paper. This is mainly due to the fact that we trained on less epochs.

	BLEU-1	BLEU-2	BLEU-3	BLEU-4
our results	67.2	47.2	32.1	20.8
paper results	70.7	49.2	34.4	24.3

Figure 1: Top-Down

	BLEU-1	BLEU-2	BLEU-3	BLEU-4	CIDEr	Meteor
our results	72.4	54.6	43.1	28.23	105.7	26
paper results	77.2	-	-	36.2	113.5	27

Figure 2: Bottomp-Up

3.2 Qualitatif results

We see that the areas generated by the Top-down Attention are not necessarily all used to generate the output phrase. Similarly we can reasonably assume that the use of the Faster R-CNN does not imply the use of every word found by the network, since this model generates sentences with two or three visual words (i.e. names found in the image) whereas the image contains more elements identifiable by the neural network.

4 Implementation

We implemented the Top-Down model using the github provided in the project brief. We encode the image using a CNN. We use a pre-trained Resnet101 (Encoder). Then an LSTM is used to generate each word one by one knowing the previous ones (Decoder). We train the model on about 50 epochs with the GPU integrated on google colab. The CNN weights are not frozen so we also adapt them to each epoch.

To implement [2], we had difficulties to train our model on COCO, each epoch requiring a lot of time to be computed. We tried to allocate several GPUs with google cloud but, although we tried to run the codes through the terminal, we were not able to run them simultaneously. So we used a google cloud GPU. However we ran out of time and had to reduce the number of epochs to train the model. This explains the significant difference in our performance compared to the research papers. A good idea would have been to use a smaller database to save time and try to go further. However, a performance evaluation on a database other than COCO did not allow us to compare our results with the articles.

5 Disclaimer

We said during the presentation that we used Flickr30k to train the Top-Down model. That was true in a first approach. We did indeed use this database (as well as Flickr8k). We then wanted to compare the result of our two implementations and we used COCO in a second step on Top-Down in order to compare the results on two similar bases

6 Conclusion

In this project, we have studied image captioning through different techniques, each one presenting an improvement compared to the previous ones. It should be noted, however, that the performance of the captions generated by the algorithm will strongly depend on the database used for training. Indeed, we have used COCO to train the model, so it will be more efficient on images ”close” to those in this database. For example, trying to give medical images would be completely inefficient.

7 References

[1] Show, Attend and Tell: Neural Image Caption Generation with Visual Attention, Kelvin Xu

[2] Bottom-Up and Top-Down Attention for Image Captioning and Visual Question Answering, Peter Anderson, Xiaodong He, Chris Buehler

[3] Self-critical Sequence Training for Image Captioning, Steven J. Rennie, Etienne Marcheret, Youssef Mroueh

[4] SEQUENCE LEVEL TRAINING WITH RECURRENT NEURAL NETWORKS, Marc'Aurelio Ranzato, Sumit Chopra, Michael Auli, Wojciech Zaremba

[5] CIDEr: Consensus-based Image Description Evaluation, Ramakrishna Vedantam, C. Lawrence Zitnick, Devi Parikh

[6] Microsoft COCO: Common Objects in Context, Tsung-Yi Lin, Michael Maire, Serge Belongie, Lubomir Bourdev, Ross Girshick, James Hays, Pietro Perona, Deva Ramanan, C. Lawrence Zitnick, Piotr Dollar

[7] "Faster r-cnn: Towards real-time object detection with region proposal networks." Ren, Shaoqing, et al.

[8] github : sgrvinod, A pytorch tutorial for image captioning

[9] github : poojahira, Image captioning top-down bottom-up

[10] github : ruotianluo, Self critical

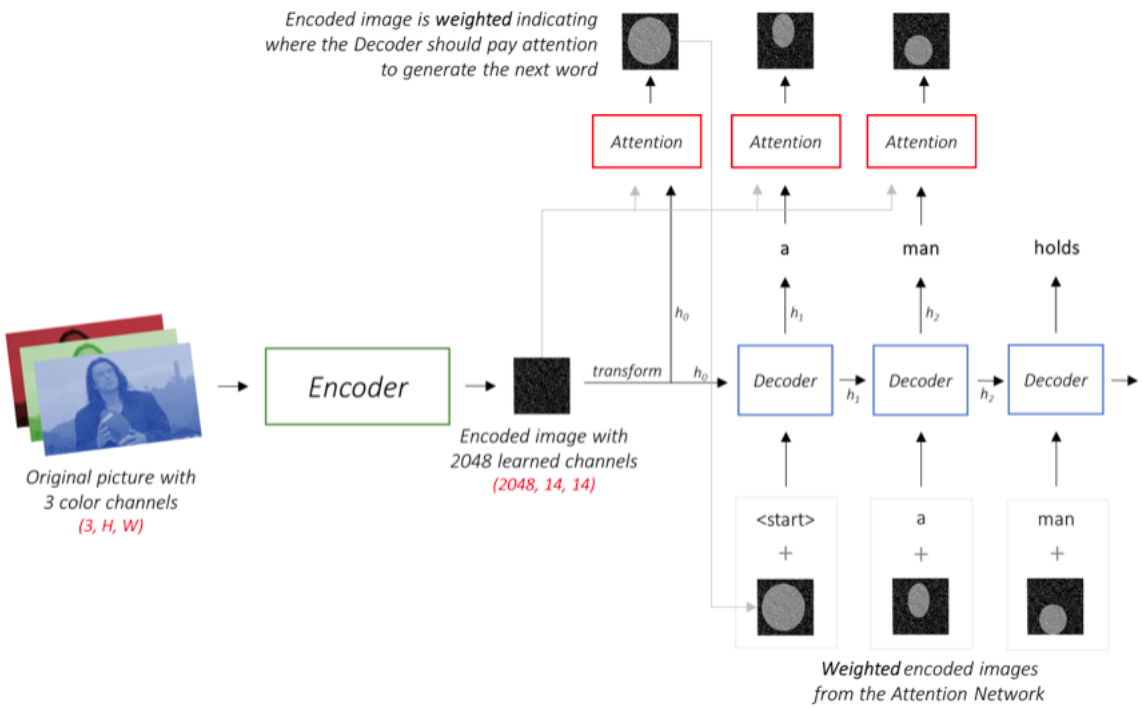

Figure 3: Top Down schema.

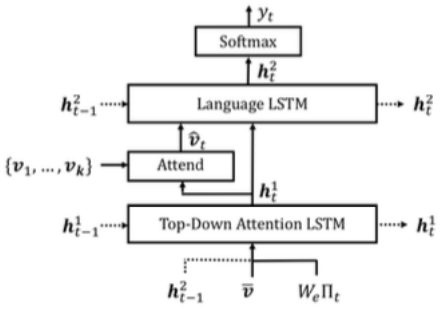

Figure 4: Bottom-Up Architecture.

Figure 5: Difference between Bottom-up model and Top-Down model.

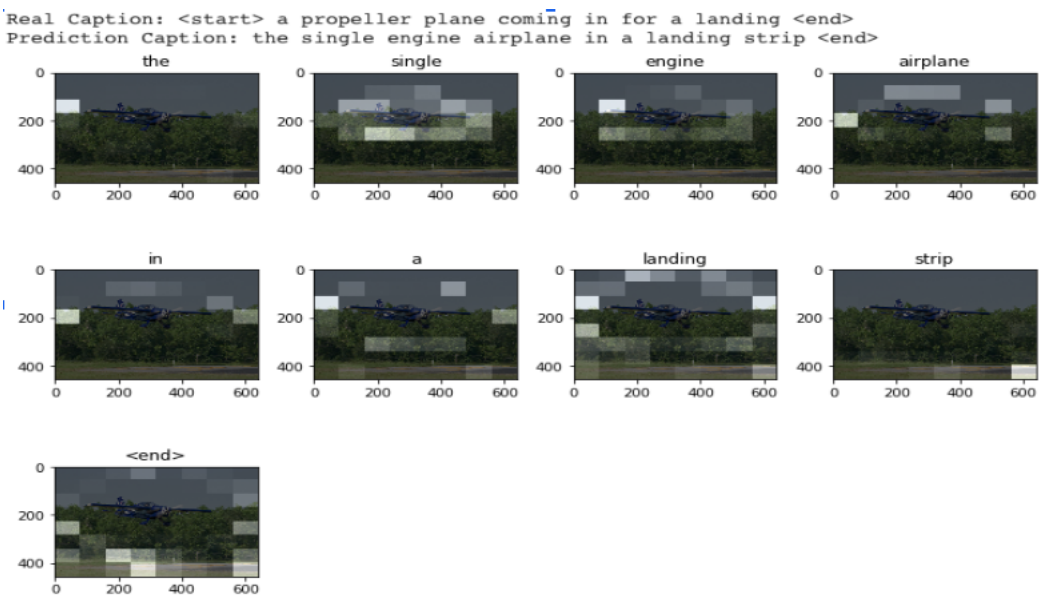

Figure 6: Result with Top-down Attention.