Image captioning using Convolutional Neural Networks and moderns Natural Language models

Luiz Eduardo Pita M Almeida and Letícia Rittner

***Abstract*—On the last decade, deep learning techniques archived many state of art results in computer research areas such as Computer Vision. Nowadays, we are experimenting a revolution in Natural Language Processing field. One task that join Computer Vision and NLP is the image captioning task. This task , using a deep learning approach, consists of extract deep representations of images using a Convolutional Neural Network model, associated them with a embedding representation of caption words in a language generator model. This paper purposes an algorithm based on a encoder-decoder architecture to predict new captions for an image. This model uses a ResNet101 CNN to extract features from image, a BERT model to generate word embedding, and a LSTM layer as text generator. Also, we included a Soft Attention mechanism to calculate the relationship between each predicted word and a portion of the respective image. To measure our results we used the BLEU metric, a common metric used for the chosen dataset, Coco Captions dataset. This paper source code can be find in GitHub[1].**

***Index Terms*—Image captioning. Convolutional Neural Networks. Bert. LSTM. Encoder. Decoder. Soft Attention Mechanism.**

I. Introduction

With the emergence of deep learning, specially the advent of Convolutional Neural Networks (CNN) and Recurrent Neural Networks (RNN), the image captioning area, like many other subjects of Computer Vision, achieved new state of art results. The image captioning is an automatic way to describe an image, generating a caption for the scene. It can help in many applications, such as image retrieval. The image captioning incorporate many areas of Computer Vision, such as object detection, object recognition, scene understanding, object properties and their interactions. All of these are necessary to make a machine understand an image, but it also necessary to make a machine learn how to generate a sentence [3].

Another research area that got an upgrade with the advances on deep learning is the Natural Language Processing (NLP). This area gotten better results with the use of neural models, followed by the use of RNN, such as LSTM (Long Short Term Memory) and GRU (Gated Recurrent Unit). Nowadays, the state of arts results are focus on the use of attention mechanisms, mainly with the advent of Transform models [10]. Pre-trained Transform models like BERT (Bidirectional Encoder Representations from Transformers) [1] and T5 [8] are the most modern state of art methods in NLP.

In general, image caption models follow a encoder-decoder architecture, where the encoder is an image feature extractor, mostly a CNN. This image encoder can be associated to a language model encoder to generate a jointly embedded representation of words and images. This association forms the multi-modal language models. On the other hand, the decoder is a text generator model, mainly using LSTM language models, that translate the embedding from encoders into a sentence. This approach is mainly trained using a supervised method. Others approaches include the usage of reinforcement learning, unsupervised learning, attention mechanisms, semantic concepts, additional code blocks that check the quality of the text generated, generative adversarial networks, and others [3].

This paper focus in implement a image caption generator demonstration that follows a encoder-decoder architecture. The encoder is a CNN model only to extract images features, so it is not a multi-modal encoder. The decoder uses a modern NLP pre-trained model, the BERT, to generate word embedding, and a LSTM network to generate captions.

[1]https://github.com/LuizPitaAlmeida/image_caption_generator

II. Related Works

The first famous paper using Deep Learning in image captioning was the Show and Tell (2015) paper [11]. The authors of this paper proposed an encoder-decoder approach for the task basing in the advances in machine translation algorithms. Instead of use a RNN encoder, they propose to use an CNN to extract deep features from the images. In the time of the paper, CNN approach were hype algorithms for image processing tasks. The final model proposed had a CNN encoder and RNN decoder, as shown in Fig. 1.

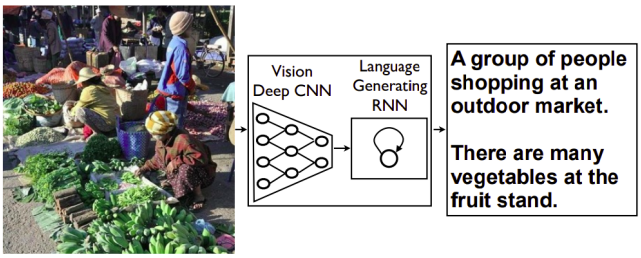

Figure 1. Show and Tell model. [11]

A second paper that based our implementation was the Show, Attend and Tell (2015) paper [12]. This paper was published near the publication of Show and Tell (2015) paper. It uses the same idea of Show and Tell, but change the RNN decoder for a LSTM decoder, and add the concept of Attention Mechanism for images. The main contribution of the paper were the two attention mechanisms proposed. The first one, used in our paper, was the Soft Attention that uses linear layers to determine in a deterministic way what region of the image

embedding the decoder needs to look to predict a correlated word. The second one is the Hard Attention, that combines supervised approach with a reinforcement learning approach creating an stochastic attention mechanism for images. Fig.2 and Fig.3 illustrate the proposed model and how the attention mechanism acts on images.

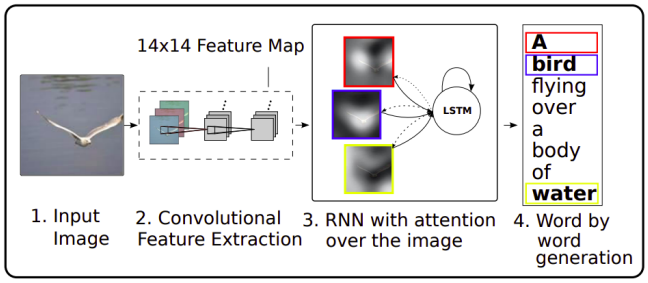

Figure 2. Show, Attend, and Tell attention mechanism effect. [12]

Figure 3. Show, Attend, and Tell attention mechanism effect. [12]

Also, we proposed to add BERT transformer model to Show, Attend and Tell implementation. BERT is recent paper published by Google AI team in 2019 that, in the last year, got state-of-art results in a wide range of NLP tasks. It is designed using only attention mechanism, i.e., a Transformer module [10], and considering a bidirectional context.

Transformers models are multi-head attention models formed by an attention encoder and an attention decoder. An attention encoder layer for NLP tasks does linear combinations of the input tokens embedding predicting a probability for each one of those combinations. The BERT is a Transformer encoder that does a bidirectional context combination trying to use both past and future context to predict a word [1].

Bert was trained to be a language model that understand better the language context and flow. Researches notice that BERT could be distribute as a pre-trained model such as pre-trained CNNs, so common in Computer Vision area. In our approach we used BERT only to generate embedding for the LSTM language model.

The modern approaches for image captioning consists of create a single model that can handle with image and text features as a multi-modal feature extractor to predict text outputs. In some case the input are only images. These novel approaches are named as Visual-Linguistics representation models. Most of them combines transformers architectures with CNN for feature extraction, classification task and object detection [2], [4], [5], [7], [9].

III. Materials and Methods

In this section we describe the materials such used programming language and data.

A. Programming Language

For this project we are using the Python 3 language with the Pytorch 1.3 library. Pytorch is an open source machine learning framework designed to accelerate research prototyping. In the last years together with TensorFlow is the most used framework for deep learning.

B. MS COCO database

The Microsoft Common Objects in Context [6] is a large and well knowing database used for benchmark many Computer Vision applications. It has a dataset for image captioning containing more than 330000 images with five different descriptions for each one. We used the COCO Captions challenger split of data. Since annotation of test dataset is not available, we will show our results over the validation dataset. The validation dataset is composed of 40504 images with five captions for each one. Fig. 4 shows a data sample example.

Figure 4. Coco Caption Dataset Sample.

The above image is a demonstration of the dataset. In other to use the dataset in our model we need to pre-process our images, generate a vocabulary of the words, tokenize this words, add special tokens ("<start>" and "<end>" tokens), and generate minibatches. In Torch these minibatches are generated by dataloader class. In this paper we used a 9956 words vocab and a batch size of 64. Also we limited training batches number and validation batches number due to problems with full memory.

IV. Proposed Model

In this section we present our encoder-decoder model. The encoder is formed by the feature extraction of ResNet-101, while the decoder used the BERT to generate contextualized word vectors that are after detokenized and predicted captions are formed. These captions uses BERT embedding and images features together due to the attention mechanism module. This module is based in Abdulrahman Jamjoom[2], Yunjey Choi[3], and Sagar Vinodababu[4] implementations. Fig. 5 shows an summarizing schema of our model.

In next subsections we will discuss about the two parts of the model: encoder and decoder.

[2]https:github.comajamjoomImage-Captions
[3]https:github.comyunjeypytorchtutorial
[4]https:github.comsgrvinoda-PyTorch-Tutorial-to-Image-Captioning

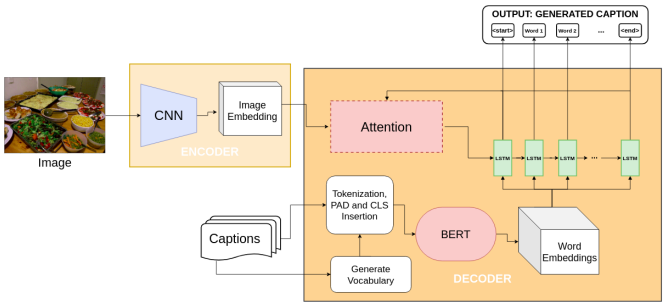

Figure 5. Proposed Encoder-Decoder Model.

A. Encoder: CNN Feature Extractor

We decided to use a common used CNN for image feature extraction, so we chosen the ResNet-101. This CNN receives as input a batch of images with dimension [3, 244, 244]. The first dimension is the number of channels in image, while the second and third ones are the number of rows and cols respectively. To only do a feature extraction, not a classification, we removed from ResNet101 its two final layers (classification layers). Also, we add at the end of the network an adaptative average pooling layer to make possible to chose the output image size. Since by default we chosen to use a output image size of 14x14, the average pooling does an upper-sampling over image size (ResNet101 last convolutional layer is 7x7). So the encoder output will have the shape [batch_size, 14, 14, 2048] where 2048 is the embedding size of the last layer.

B. Decoder: Association between Bert, LSTM, and Soft Attention

The decoder is formed by a LSTM layer to do captions predictions. The LSTM layer is feed by some image features and a set of word embedding provided by a BERT model acting over train dataset captions. Each predicted word feed a Soft Attention layers that applies weights for each pixel in image, saying to the model where to look. This weighted image embedding is the above-mentioned image feature that is inputted back into LSTM for next word prediction.

Notice that in our model both image and text pass by an attention mechanism process. The word deep representations are obtained from BERT while images deep representations are obtained from the encoder associated with an attention mechanism.

Decoder input is composed of the encoder output, the tokenized captions, and a list with the original length of each caption. We started generating word representations using Bert. For this, first we fix the captions' size using the length of the largest caption and padding the others until they have the same size as the largest. Then this captions are detokenized only to be tokenized by Bert tokenizer. This new tokens feed Bert model generating a embedding tensor. This tensor is transformed into a list with same size of the caption tokens, each element of this list is a portion of Bert embedding correspondent to each token of caption. The word embedding tensor is generated from this list.

After this, the second step is initialize LSTM layer, hidden states and cells with the mean value of encoder embedding. With this initial value, we can do the first pass over attention layer. The attention layer receives the encoder output and the hidden states of LSTM, and returns the alpha values, kind of probability or weight for each pixel in image, and the encoder output product with these alpha values. This product is concatenated with the Bert embedding feeding back the LSTM, that will update your hidden layers. This new hidden layers go to the attention mechanism, keeping this circle until the end of the caption. The complete output of LSTM pass throw a linear classification model, that predict the most probably words in the vocab to form the final caption. The decoder output is composed of the new captions predictions, the lists of captions tokens id in original tokenization, a list of length of each predicted caption and the alphas values used in image attention.

V. Results

In the output of encoder we had a set of score predictions, one for each word in vocabulary. To compute a readable output of these predictions it is necessary to pass these predictions in a softmax layer and get the words ids with highest probability to form a new caption.

Also we evaluate our results using BLEU metric. This is a common metric used in machine translation problems, that compares a sentence generated by a machine (hypotheses) with humans annotations for that sentence (references). The BLEU metric is the metric used for the COCO Captions Challenger where we got our database. Also this metric has some weighting variations that consider the appearance of different n-grams in common with the reference (BLEU-1, BLEU-2, BLEU-3, and BLEU-4).

A. Qualitative Results

Let's first analyse the qualitative results. The Fig.6 shows a predicted new caption hypotheses in comparison with a reference caption and its image.

Figure 6. Single sample predicted result.

We notice that the result obtained in hypothesis is very similar to the reference sentence. However, our prediction has less details and repeated some words. This is caused because we interrupted out training in epoch 39. Also, we limited our

train dataset, because lack of memory issues. We can also look to alphas variable and try to plot the attention mechanism effect, as shown in Fig.7.

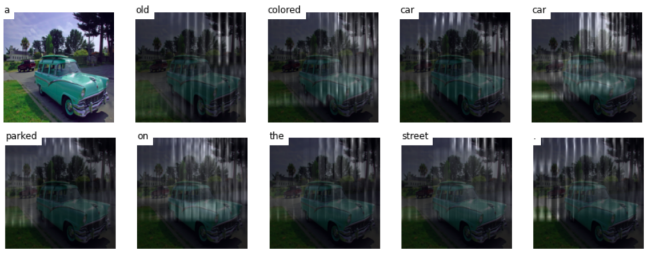

Figure 7. Effect of Soft Attention over a sample result.

We can notice that the attention alpha values are starting to split relevant portions of the image. Pay attention in "parked" word, and notice how the attention mask excludes the car to generate this word. With more training this results starts to become more visible.

B. Quantitative Results

We calculated BLEU metrics over all validation dataset. The results are presented in Table I.

Model	Val (loss)	BLEU-1	BLEU-2	BLEU-3	BLEU-4
JAMJOOM	**1.901**	**78.27**	**59.53**	**46.22**	**36.53**
OUR	2.460	66.94	42.37	27.74	18.73

Table I
MODEL VALIDATION LOSS AND BLEU SCORES ON THE VALIDATION DATASET

From the Table, we can see that all of our results are below JamJoom implementation. This confirm the fact that we stopped in the middle of the train.

VI. CONCLUSION

Build a model for image captioning is not easy. Many modules are necessary: image encoder, word embedding generator, attention mechanism, text generator. Connect this model and fine-tuning each model together is a challenger.

Our results showed that we still have space to improve. For this is necessary to continue the training until archive the best results.

In future works, i will develop an inference model to make new images predictions. And try to developed approach that does not need a reference caption as input. Another desire goal is to remove the LSTM layer and use only a Transformer layer.

REFERENCES

[1] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[2] Simao Herdade, Armin Kappeler, Kofi Boakye, and Joao Soares. Image captioning: Transforming objects into words. In *Advances in Neural Information Processing Systems*, pages 11135–11145, 2019.

[3] MD Zakir Hossain, Ferdous Sohel, Mohd Fairuz Shiratuddin, and Hamid Laga. A comprehensive survey of deep learning for image captioning. *ACM Computing Surveys (CSUR)*, 51(6):1–36, 2019.

[4] Gen Li, Nan Duan, Yuejian Fang, Daxin Jiang, and Ming Zhou. Unicoder-vl: A universal encoder for vision and language by cross-modal pre-training. *arXiv preprint arXiv:1908.06066*, 2019.

[5] Liunian Harold Li, Mark Yatskar, Da Yin, Cho-Jui Hsieh, and Kai-Wei Chang. Visualbert: A simple and performant baseline for vision and language. *arXiv preprint arXiv:1908.03557*, 2019.

[6] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014.

[7] Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. In *Advances in Neural Information Processing Systems*, pages 13–23, 2019.

[8] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683*, 2019.

[9] Weijie Su, Xizhou Zhu, Yue Cao, Bin Li, Lewei Lu, Furu Wei, and Jifeng Dai. Vl-bert: Pre-training of generic visual-linguistic representations. *arXiv preprint arXiv:1908.08530*, 2019.

[10] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.

[11] Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. Show and tell: A neural image caption generator. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3156–3164, 2015.

[12] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. Show, attend and tell: Neural image caption generation with visual attention. In *International conference on machine learning*, pages 2048–2057, 2015.