Neural Image Captioning with a Recurrent-CNN Network Architecture

Arik Arikhan, Julio Davila, Norman Di Palo,
Jose Jamarillo, Kyriakos Lite

Sapienza University of Rome
Deparment of Control, Computer and Management Engineering
Rome, Italy

June 12, 2018

Abstract

Neural Image Captioning is one of the growing field in computer vision. The task, nowadays, is tackled using particular neural network architectures, that use recurrent layers on top of deep convolutional neural networks, in order to create a sentence based on the visual features extracted from the image. In this work, we show an implementation of a recurrent-CNN network, based on Inception V3 and GRUs, trained on the Microsoft COCO dataset and able to create realistic captions for a wide variety of images. We encode images using the pretrained Inception model, and we encode words using the widely used GloVe embedding. We evaluate the performance of the network using BLEU, ROUGE, CIDER and METEOR.

1 Introduction

Image Captioning is the task of creating a descriptive sentence for an image, automatically. The task is considered quite hard, since it needs a semantic understanding of the image, and also a capability of transfering this understanding in fluent text. In the recent years, this task, as many other in the fields of computer vision and natural language processing, has been tackled using deep learning techniques. In particular, the architecture that is most succesful in this kind of application is a combination of convolutional neural networks and recurrent neural networks. This allows to both extract visual features from the image, and also create text in a dynamic way thanks to the recurrent layers.
More specifically, the architecture is inspired by the encoder-decoder structure that has worked very well for tasks like machine translation or image encoding with autoencoders. A CNN transforms an image into an embedded space of

features, using its many convolutional layers. After that, a GRU layer uses this encoded features as input, generating words for each time step. The caption ends when the GRU predicts a stop token.
In the following sections, we will describe the architecture used and also the training procedure, and then show some empirical results on different widely used metrics, based on the training done on the COCO dataset.

2 Architecture and Training

As previously described, our network can be seen as an extension of the widely used encoder-decoder paradigm in deep neural networks. We used, as the CNN that extracts visual features, Inception V3, pretrained on ImageNet, and we kept it frozen during all the training. The features extracted from the layer before the classification are then used as input to a GRU layer, that performs different steps of dynamic computation to generate a caption, until if predicts a stop token.
The task can be seen as a classification task, in which the recurrent layer predicts, at each time step, the correct token to add to the sentence, between the available vocabulary. Thus, we try to optimize the probability of outputting a correct sentence given the image and the weights of the network.

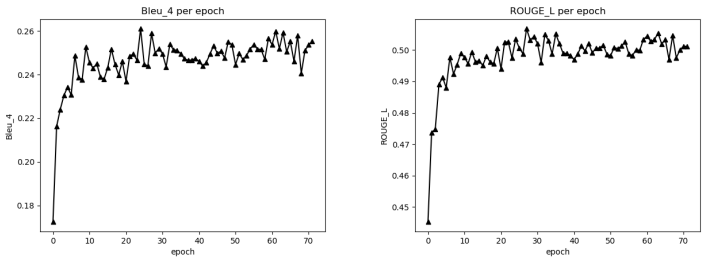
$$\theta^* = \arg\max_\theta \sum_I \log p(S^I|I;\theta)$$

where S is a sentence, I is an image and θ are the weights of the network. Thus, we want to find the set of weights that maximize the log-likelihood of predicting the right sentences for every image in our training set. The sentence are not fixed in lenght, so they all start with a particular start token and end with a stop token, that are also predicted by the network. The joint probability of predicting the correct sentence, word by word, is thus

$$\log p(S^I|I;\theta) = \sum_t \log p(S_t^I|S_{(t-1)}^I, ..., S_0^I;\theta)$$

where we indicate each word as S_t^I, highlighting the sequential structure of the sentence as a series of time steps. Our goal is thus to optimize the weights of the GRU layer in order to correctly model this conditional probability.
The input of the GRU is the results of one of the last layers of the Inception-v3 model applied to each image. In particular, we extract a vector $x \subset \mathbb{R}^{8\times8\times2048}$, that encodes the 64 spatial regions as vectors of 2048 features. We then compute from this matric a 2048×1 vector by averaging the 64 blocks of the matrix. After that, we project this resulting vector into a lower dimension vector using a linear projection followed by a sigmoid transformation $x_0 = \sigma(Wx_{avg} + b)$. This vector is the first input of the GRU model. All the sequent inputs are GloVe embeddings of words, i.e. the previous word of the caption that the network is trying to predict. During training time, we ignore the word that the network has

Figure 1: Per-epoch plots of BLEU, ROUGE.



predicted and give as input the embeddings of the right word at that particular time step, while during test the word given as input is the embedding of the last word outputted by the network.
In order to train the weights of the network, we perform, as usual, stochastic gradient descent on a loss function, that is defined as the negative log-likelihood of a sentence being generated given an image.

$$L(I, S^I) = -\sum_t \log p(S_t^I | I, \theta)$$

We used the Microsoft COCO dataset for this task, using 123000 total images, of which 5000 were used as validation and 5000 were used as test. The metric that we used to test the performance of the model, other than its decreasing loss, are ROUGE, BLEU, METEOR and CIDEr. All these metric can be easily computed using pre-configured toolkits.

3 Results

After several hours of training on a GPU, the model loss decreased significantly and the observed metrics obtained high values as we expected. Also, empirical tests of image captioning showed the capability of the model of predicting convincing descriptions of the images. In Figure 1, 2 and 3 we show all the plots showing the evolution of loss and metrics during different epochs.

Figure 2: Per-epoch plots of METEOR, CIDEr.

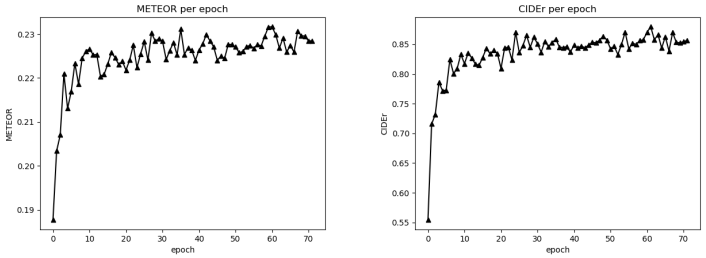
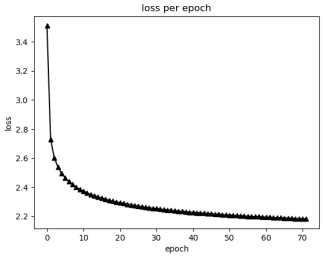


Figure 3: Per-epoch plot of loss function.



We also show some example captions created by the model, both for correct case (the majority of the cases) and also some interesting errors by the network.

Figure 4: Examples of correctly captioned images.

4 Conclusion

In this work we described the implementation and training of a neural image captioning architecture, based on an encoder-decoder paradigm. The encoder is made by a pretrained CNN, Inception-V3, that extracts high-level features from images. The decoder network is a GRU, that receives as input sequentially first

Figure 5: Examples of captioned images with small errors.

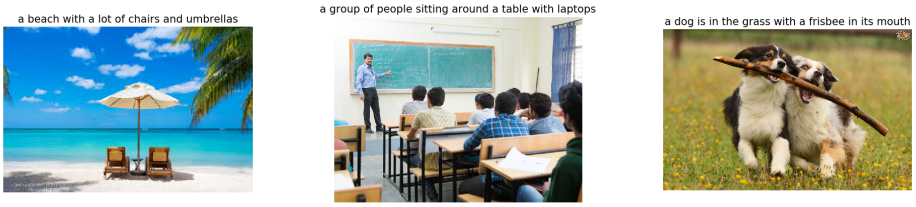
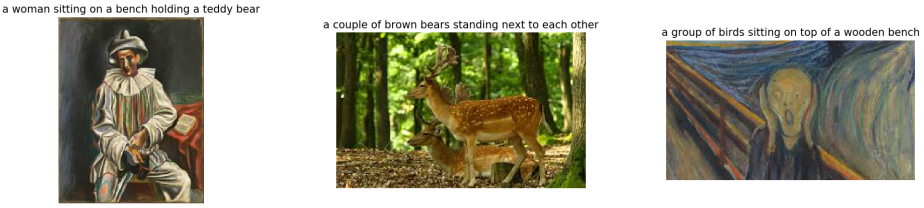


Figure 6: Examples of miscaptioned images.



an embedding of the image, and then embedding of the previous words of the caption, and its task it to predict the probability of the next word to appear. We showed how after several hours of training on a GPU the model reaches a very good level on several metrics, as well as on the empirical task of captioning correctly images.
This field is rapidly growing, since it is on the boundary of computer vision and natural language processing. Improvements on this task can arise from several parts, including better CNN architectures, as well as better word embeddings, training procedures, and also, possibly, bigger datasets.

5 References

[4] Tsung-Yi Lin et al. Microsoft COCO: Common Objects in Context. EECV, 2014
[5] Steven J. Rennie et al. Self-critical Sequence Training for Image Captioning. arXiv:1612.00563, 2016
[6] Jiasen Lu and Caiming Xiong et al. Knowing When to Look: Adaptive Attention via A Visual Sentinel for Image Captioning. CVPR, 2017
[7] Sepp Hochreiter and Jurgen Schmidhuber. Long short-term memory. Neural Computation, vol- ume: 9, issue: 8, 1997
[8] Kyunghyun Cho et al. Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. EMNLP, 2014

[9] Christian Szegedy et al. Rethinking the Inception Architecture for Computer Vision. Proceedingsof the IEEE Conference on Computer Vision and Pattern Recognition, 2016
[10] Olga Russakovsky and Jia Deng et al. ImageNet Large Scale Visual Recognition Challenge. IJCV, 2015
[11] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. GloVe: Global Vectors for Word Representation. 2014
[12] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. ICLR, 2015
[13] K. Papineni et al. BLEU: A method for automatic evaluation of machine translation. ACL, 2002
[14] Ramakrishna Vedantam, C. Lawrence Zitnick and Devi Parikh. CIDEr: Consensus based image description evaluation. arXiv:1411.5726, 2015
[15] Satanjeev Banerjee and Alon Lavie. Meteor: An automatic metric for MT evaluation with improved correlation with human judgments. Proceedings of the ACL workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization, vol. 29, 2005
[16] Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. Text summarization branches out: Proceedings of the ACL-04 workshop, vol. 8, 2004
[17] Samy Bengio et al. Scheduled sampling for sequence prediction with recurrent neural net works. Advances in Neural Information Processing Systems, NIPS, 2015
[18] Kaiming He et al. Deep residual learning for image recognition. CVPR, 2016
[19] Caiming Xiong, Stephen Merit and Richard Socher. Dynamic Memory Networks for Visual and Textual Question Answering. ICML, 2016
[20] Gustaffson Fredrik, Neural Image Captioning for Intelligent Vehicle-to-Passenger Communication, 2017, CS224 Stanford University