CS230

Generating Descriptions of Artworks with a CNN-RNN Encoder-Decoder Model

Carolyn Akua Asante Dartey
casanted@stanford.edu

Aleksandra Sokolova
algae09@stanford.edu

Adhara Martellini
adhara@stanford.edu

Abstract

Identifying conceptual themes in the art work and describing them with words has always been considered a unique human ability. However, as we shift into the era of implementing AI with more human-like decision making, we need an algorithm that engages with art works in the same way humans do. In addition, such classification can be useful for art conservation through digital archives and analysis of the arts' history. In this paper, we aim to build an algorithm capable of describing the content of a variety of different artworks, hereby generating image captions for the artworks. We used the pretrained CNN model Xception and fused it with a caption-generating RNN LSTM model to give us our resulting model. We trained this on the ArtEmis database based on WikiArt paintings. We discuss different hyperparameter adjustments and decisions that gave us the best results, and perform some quantitative analyses on our generated descriptions.

1 Introduction

The recognition of objects in an image and correctly identifying each element, as well as the overall context in the image, is something humans do on a day to day basis, especially artworks, that are often more ambiguous than photographs. Whether the works are literal or abstract, the human mind is able to generate descriptive analysis of them. Creating an algorithm that does the same would be very useful: not only would it give us insight in the human processing of the artworks, but it would allow AI to have a more complex understanding of the image they classify and describe them on multiple conceptual levels. In addition, the algorithm can be used to classify art in digital archives and to study art history's tendencies in a deeper way.

Recently, a lot of work has been done in the deep learning field in regards to getting artificial intelligence closer and closer to human-like intelligence in relation to visual recognition and natural language processing. There has been work on image classification and training neural networks to correctly identify images in particular classes. This work has also been extended in the art world to get neural networks to identify the emotion a certain artpiece wishes to convey. Closest to our work in this project, there have been studies that have explored image captioning and describing the content and context of photographed images. These networks are trained on datasets such as the Flickr8k, Flickr30k and the Microsoft COCO datasets to generate captions for images.

Our paper introduces a similar infrastructure for performing this task, but on artworks. It is a much more difficult task to describe the contents of pieces of art, as art is almost always subjective. Training artificial intelligent to do this, then begins to introduce some of the complexities of the human mind in decision making that allows us to be able to perform this task with ease. To train our model, we use artworks from the ArtEmis Dataset, which is a database designed to help with the task of emotion-from-art classification and text-utterances-to-emotion classification tasks. This database is based on about 81,000 WikiArt paintings, accompanied by 440,000 written responses from over

6,500 humans indicating the description of each painting. The original tasks the ArtEmis group set out to accomplish were mainly mono-modal classification tasks. Our projects seeks to combine both the visual and the language information available to create a model that is able to take in both inputs and learn to describe art pieces on its own. We do this by processing each image in a Convolutional Neural Network known as Xception, and then tokenizing the captions from annotators and train by passing these into a Recurrent Neural Network LSTM Recurrent Neural Network model. We combine these models to output a caption describing the art work's content.

2 Related work

A lot of prior work done in the field of visual recognition involves classic image classification tasks, as shown achieved through datasets such as CIFAR-10, MNIST, and ImageNet. In [1], we see how datasets like these are used to train a model to give a single word classification of an image. though this is a great feat, it would be even more progress in the world of artificial intelligence if these algorithms and models could string together words to form sentences that say more about the task it has been delegated to perform.

We found several papers exploring similar issues but with a slightly different implementation. One of them called "ArtEmis: Affective Language for Visual Art" [2] by Achlitopas et al. explored the image captioning of artworks with emotional descriptions of the visual stimuli, emulating human emotional responses to the artwork and classifying the paintings into emotional categories. Differently from us they posed two problems, instead of one: emotional classification and captioning. When it comes to captioning, our implementations are similar. Like us, the researchers use cross-entropy-based optimization applied to an LSTM text classifier trained from scratch, but they also fine tuned it to this task using a pretrained BERT model. The difference between our captioning model and theirs is that whereas their model predicts an emotion based off of a text caption, we aim to predict a caption based off of the original image. Another difference between our work and the work done in this paper is that the researchers used ResNet-32 encoder pretrained on ImageNet minimizing the KL-divergence between its output and the empirical user distributions of ArtEmis in their emotion classification task, while we use Xception as or CNN.

We also looked into CNN papers to find the best algorithms for extracting features from our visual dataset. There are a lot of papers investigating best implementations of image classification and recognition using CNN. Examples of these include ResNet, Inception, VGG-16 and Xception. [13] and Dhar et al. [8] collect descriptive attributes such as interestingness, symmetry, light exposure, and depth of field, using convolutional neural network. While most of these papers focus on photography or web imagery, our work focuses on art works, which are much more ambiguous in visual representation.

Finally, we investigated a group of papers dedicated to the efficiency of the specific methods that we were planning to use in our analysis. For example, in [2] we see that bidirectional LSTM models achieve highly competitive performance to the state-of-the-art results on caption generation even without integrating additional mechanism (e.g. object detection, attention model etc.). [10] also discusses the advantage of using Long Short-Term Memory neural networks as a result of their feedback connection nature. This allows for predictions for a sentence to be made based off of the previous words in the sequence, thus making it suitable for the task of description generation. As it is significantly outperforming any recent methods on retrieval task, LSTM became our top choice for the image captioning algorithm.

3 Dataset and Features

We use the ArtEmis Dataset as mentioned in [1], a large processed dataset consisting of 454,684 descriptive annotations on 80,031 unique paintings from WikiArt obtained from over 6,500 humans, indicating how the painting makes them feel. By retrieving the ArtEmis Dataset, we found paths into the WikiArt Dataset for each painting and the emotional histogram associated with it. We downloaded the WikiArt dataset and indexed into it to retrieve the artworks for the ArtEmis Dataset. Then we downsized them to an input size of $128 \times 128 \times 3$ for the consistent feature retrieval purposes. We retrieved the first 5 captions or description annotations for each image in the dataset. For each caption, we only use the first sentence of the of the annotation because we want to train our model to predict

Figure 1: robert-campin_werl-altarpiece-st-barbara-1438: *'The woman is getting the chance to sit indoors and work on her sewing'*

similarly. We trained our model on 14,668 artworks from WikiArt, with 5 captions per each image from the ArtEmis Dataset. We set aside 200 images for testing purposes and trained on the rest of the images and captions.

We obtain features from the images using the Xception CNN. Then we connect Xception to LSTM network, tokenize our captions, and train on our image features and tokenized sequences to generate captions. We test our algorithm on our test set and analyze our results using BLEU scores. An example of a preprocessed image and its respective sample caption from ArtEmis dataset we used for training can be found above.

4 Methods

We made use of transfer learning in training our network. We first obtained feature vectors for our training images using the Xception CNN model with weights from its pretraining with the ImageNet dataset. We excluded the top layer of the prediction to give us the (2048,) shaped feature vector for each image, which was our first input into our combined model. We then created a vocabulary of all unique words in the decriptions for all our training data, and then tokenized each word to number sequenceces since that is what the computer understands. Then for each input caption, we map them to the hashed sequences and then put these sequences into our into image captioning LSTM model, together with the feature for the image, to train the model to predict the correct sequences based off of the order in which they appear and the feature vector. The output predicted sequence is mapped back into the vocabulary to give the final sentence prediction.

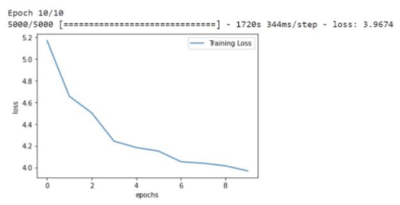
We changed the first layer of the CNN to take our preprocessed $128 \times 128 \times 3$ images, and extract their features. We used average pooling in our CNN as opposed to max pooling because it gave better results on we tried it. We use the Keras implementation of Xception, which has 71 layers. The 2048×1 feature vector would then be passed through a Dropout layer 0.5, followed by a Dense ReLU activation layer of size 256.

For the second part of our model, our input sequences (which would be of length 94, as that was the maximum length of any caption) wouls be passed through an Embedding layer followed by a Dropout layer of 0.5. We used the Keras implementation of the LSTM RNN and ran the sequences through the gates of this model, before joining its output with that of the feature vectors. Our combined inputs are then passed through a Dense ReLU activation layer, and finally output through a softmax layer.

loss function used was categorical cross-entropy loss function defined as:

$$Loss = -\sum_{i=1}^{outputsize} y_i log(\hat{y_i})$$

where $\hat{y_i}$ is the i-th scalar value in the model output, y_i is the corresponding target value, and output size is the number of scalar values in the model output. We chose cross-entropy loss function as it's a great optimization tool for the performance of the classification tasks, and a very good measure of how distinguishable two discrete probability distributions are from each other.



```
Epoch 10/10
5000/5000 [==============================] - 1720s 344ms/step - loss: 3.9674
```

Figure 2: The loss function for 5000 steps and 10 epochs

We used Adam Optimizer used to accelerate the gradient descent algorithm by taking into consideration the 'exponentially weighted average' of the gradients. Using averages makes the algorithm converge towards the minima in a faster pace.

The LSTM consists of three parts (or gates), where each gate performs an individual function. The first Forget gate chooses whether the information coming from the previous timestamp is to be remembered or is irrelevant and can be forgotten. In the second part, the cell tries to learn new information from the input to this cell. At last, in the third part, the cell passes the updated information from the current timestamp to the next timestamp.

In our experiment, we trained over our training data set for 10 epochs, and changed the number of steps we take each epoch (or the batch size) to see the effect of this on our prediction, as well as see the opportunity cost for this in terms of train time. Our results are explained further in the next section.

5 Experiments Discussion and Results

We chose to vary number of steps per each epoch to analyze the effect of minibatch sizes on our training time, quality of our prediction, as well as the BLEU scores for our testing data. Our learning rate was kept at a constant of 0.001, and for each experiment, we ran 10 epochs as a control of our experiment.

At first, we've tried to train the model using the entire training batch as the batch size, but this just made training time very long. This is why we experimented with different number of steps to determine which one would give us the best results. We've performed a standard grid choice over the minibatch size and the number of epochs to determine the optimal hyperparameters. We also created a grid of values, including number of epochs to train and the size of the steps. After trying training on 5, 6, 10, 15 and 50 epochs, we concluded that 10 is an optimal number of epochs for successful and fast training. We also tweaked hyperparameters of the number of steps and tried 50, 100, 1000, 2000, 5000, 7000, and 10000 steps of training. Our investigation led to conclusion that 10000 number of steps produced the highest accuracy of prediction. The Figure [2] illustrates the loss function graph for the run with 5000 steps and 10 epochs, with loss value ending at 3.9674 after training and .

For our primary metrics we have chosen accuracy and implemented it using the algorithm of BLEU, a metric for evaluating a generated sentence (the n-gram translation) to an n-gram of reference sentence, with a perfect match being 1.0 and the 0.0 being a perfect mismatch. We chose this algorithm as it's quick and inexpensive to calculate, easy to understand and correlates highly with human evaluation.

In the end, we were able to train our algorithm for 10 number of epochs with 10000 number of steps and achieved a pretty high accuracy in terms of predictions, as well as a final value of 3.84 on the loss function. We calculated BLEU scored for the all of the first 20 images in our test set, cumulative over each of the 5 captions in the label for each image. However, for the purposes of analysis in the next section, BLEU score will refore to the score received by the test-image titled `camille-corot_girl-reading-1850`.

Here is a table with with values for analysis:

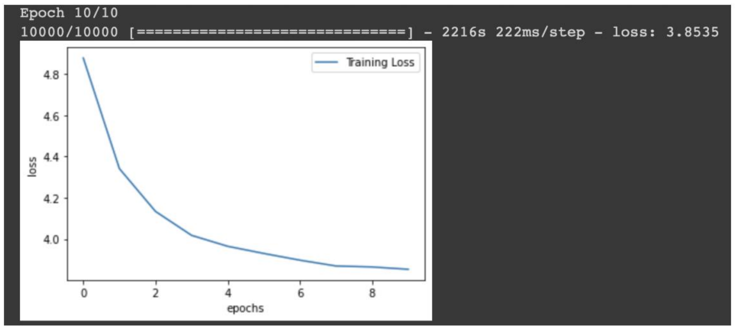

```
Epoch 10/10
10000/10000 [==============================] - 2216s 222ms/step - loss: 3.8535
```

Figure 3: The loss function for 10000 steps and 10 epochs

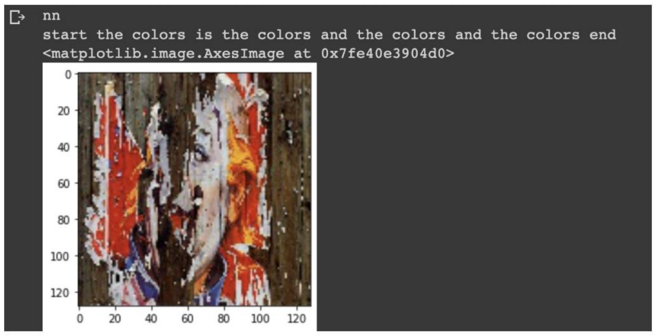
Experiment	Runtime for each epoch	Final Loss Value	BLEU score
50 steps per epoch	11s	5.8237	0.098
100 steps per epoch	38s	5.4752	0.113
1000 steps per epoch	415s	4.3923	0.178
2000 steps per epoch	447s	4.1878	0.259
5000 steps per epoch	1720s	3.9674	0.297
7000 steps per epoch	6323s	3.9346	0.286
10000 steps per epoch	2215s	3.8535	0.315

From the table, we see that our best results were obtained from our iteration using 10000 steps per epoch as this gave the highest BLEU score, and had the lowest loss function calculation. Our loss is comparable to that in other implementations of this model with regular photographs such as 3.62 in the Caption Generator by Yash Sarwaswa [16]. Our final BLEU score is not that high, and we attribute this to the subjectivity implicitly included in each caption in our dataset. For many artpieces, the annotators wrote how these artpieces made them feel or the emotion they evoked, and this is hard for the algorithm to learn to describe artistic pieces with sentiment, or identify objects that may not look the same from artist to artist.

Some of our results are as follows.

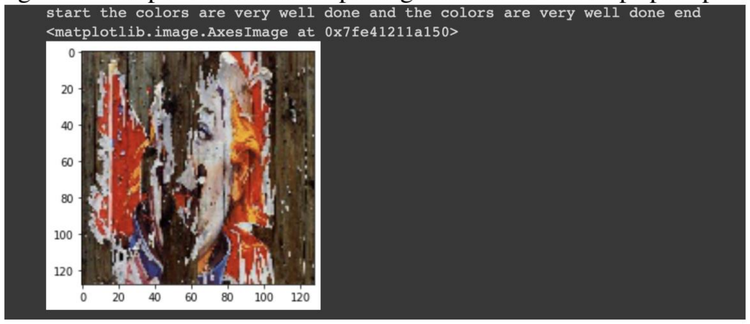
6 Conclusion/Future Work

Overall, our algorithm almost matched our expectations. Transfer learning was especially helpful in improving model performance in our architectures. There's not enough data sets dedicated to art and acquiring such data might be challenging, so transfer learning was helpful in that realm too. For the given task the chosen Xception architecture in combination with LSTM produced results various in their quality and accuracy. In the future, we would be working on improving the model so that the captions generation has higher accuracy. We also were considering expanding our algorithm so that the network not only creates captions of the artworks, but classifies them into emotional categories, which then change the way in which the descriptions of the artworks are made. That would require training the Xception model on emotional classification first, plus a separate data set connecting emotional categories and potential emotional descriptions.



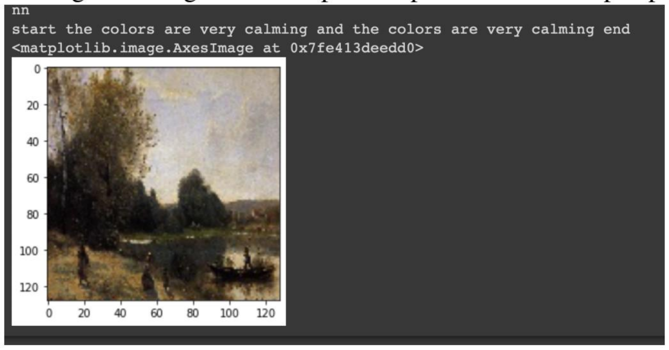
```
nn
start the colors is the colors and the colors and the colors end
<matplotlib.image.AxesImage at 0x7fe40e3904d0>
```

Figure 4: Sample bad test description generated on 100 steps per epoch



```
start the colors are very well done and the colors are very well done end
<matplotlib.image.AxesImage at 0x7fe41211a150>
```

Figure 5: Image from Figure 4 description improves in 5000 step experiment



```
nn
start the colors are very calming and the colors are very calming end
<matplotlib.image.AxesImage at 0x7fe413deedd0>
```

Figure 6: Sample acceptable test description generated on 2000 steps per epoch

7 Contributions

The team collectively worked on the conceptual development of the project. Carolyn worked on development of the CNN model, while Aleksandra and Adhara helped on testing different hyper-parameters and obtaining different accuracy scores. Aleksandra, Carolyn, and Adhara collectively worked on obtaining papers for the project and making the write up.

References

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In Advances in neural information processing systems, pages 1097–1105, 2012. 2, 4, 5

[2] Achlioptas, Panos and Ovsjanikov, Maks and Haydarov,Kilichbek and Elhoseiny, Mohamed and Guibas, Leonidas, "ArtEmis: Affective Language for Visual Art", CoRR, abs/2101.07396, 2021

[3] Chen, X., amp; Zitnick, C. L. (n.d.). Mind's eye: A recurrent visual representation for ... - cv-foundation.org. Retrieved June 1, 2022, from https://www.cv-foundation.org/openaccess/content_cvpr_2015/papers/Chen_Minds_Eye_A_2015_CVPR_paper.pdf

[4] Cheng Wang, Haojin Yang, Christian Bartz, and Christoph Meinel. 2016. Image Captioning with Deep Bidirectional LSTMs. In Proceedings of the 24th ACM international conference on Multimedia (MM '16). Association for Computing Machinery, New York, NY, USA, 988–997. https://doi.org/10.1145/2964284.2964299

[5] E. Cetinic, T. Lipic and S. Grgic, "A Deep Learning Perspective on Beauty, Sentiment, and Remembrance of Art," in IEEE Access, vol. 7, pp. 73694-73710, 2019, doi: 10.1109/ACCESS.2019.2921101.

[6] Ghupta, S. (2022, January 10). Image caption generator using Deep Learning. Analytics Vidhya. Retrieved May 31, 2022, from https://www.analyticsvidhya.com/blog/2021/12/step-by-step-guide-to-build-image-caption-generator-using-deep-learning/

[7] Guo, J. (2022, January 1). Deep Learning Approach to text analysis for human emotion detection from Big Data. De Gruyter. Retrieved April 12, 2022, from https://www.degruyter.com/document/doi/10.1515/jisys-2022-0001/html?lang=en

[8] J. Devlin, H. Cheng, H. Fang, S. Gupta, L. Deng, X. He, G. Zweig, and M. Mitchell, "Language models for image captioning: The quirks and what works," in ACL, 2015.

[9] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu. Bleu: a method for automatic evaluation of machine translation. In ACL, 2002.

[10] M. Tanti, A. Gatt, and K. P. Camilleri, "What is the Role of Recurrent Neural Networks (RNNs) in an Image Caption Generator?," Aug. 2017.

[11] O. Russakovsky et al., "Imagenet large scale visual recognition challenge," arXiv Prepr. arXiv1409.0575, 2014.

[12] O. Vinyals, A. Toshev, S. Bengio, and D. Erhan, "Show and tell: A neural image caption generator," in CVPR, 2015.

[13] P. Obrador, M. A. Saad, P. Suryanarayan, and N. Oliver. Towards category-based aesthetic models of photographs. In Proc. MMM, 2012.

[14] Shah, P. (2020, November 6). My absolute go-to for sentiment analysis-textblob. Medium. Retrieved April 12, 2022, from https://towardsdatascience.com/my-absolute-go-to-for-sentiment-analysis-textblob-3ac3a11d5244

[15] W. Zaremba, I. Sutskever, and O. Vinyals, "Recurrent neural network regularization," in arXiv:1409.2329, 2014.

[16] https://github.com/yash-sarwaswa/Image-Caption-Generator