Computational Creativity Report

Bharath Radhakrishna, 210667521
ec211168@qmul.ac.uk

Introduction

The project combines several algorithms like CNN style network [11] and image captioning model into one complex generative system that is able to do more than the individual parts like generate stylized images and associated text. This system can be seen as an innovation in generative technologies.

Each part of the system used feature extractors from the pre-trained models. CNN style transfer used VGG and the image captioning used ResNet. Both models are pre-trained on ImageNet https://image-net.org/ It is easier to accelerate the convergence of the pre-trained models than for the randomly initialized models.

The idea behind this project is to stylize the original image using CNN style network. Then that stylized image along with the original image is described in text by the image captioning model. Several experiments were carried to build some intuition behind the hyper-parameters of both the models. In the end, the image caption model was able to identify the artifacts of the stylized image through some amusing text description.

Finally the systems needed to be evaluated to formalize the benchmark to improve upon or have been improved upon. However, we cannot rely on accuracy, recall, precision, and other various quantitative evaluation measures that are typically used for classification tasks. This system is a generative system and evaluation of this type of system is particularly challenging because it relies more on how creative the system is. For that the qualitative evaluation measures are more appropriate. There are many schools of thoughts on how to really evaluate the creative systems. In this project, I'll delve into few of them to evaluate the system designed to see how creative it is.

System Description

- *Background on Style Transfer using CNNs technique implemented*

Convolution Neural Networks (CNNs) is the standard technique when working with images. CNNs use filters that perform convolution operations as it is scanning the input with respect to its dimensions. The resulting output is called a feature map. CNNs preserve the spatial information of the image feature and use less number parameters than fully connected layers.

In the CNN style network, there are two terms in the loss function. The first term makes sure to have the generated image the same as the content image focusing on the higher level feature

details instead of low level feature details. The layer at the end of the model is heavily used for that. This will be referred to as a content loss function in which the mean square error (MSE) is calculated for the generated image at the given layer and the content image.

The second term of the loss function is used to transfer the similar lower level details of the style image into the generated image. For that, the earlier layers of the model are used more. This will be referred to as a style loss function in which the MSE is calculated at the given layer between the entries of the Gram matrices from the content image and the generated image.

$$\mathcal{L}_{style}(\vec{a}, \vec{x}) = \sum_{l=0} w_l E_l$$

At the above equation, *w* is the weighting of each of the model layer's contributions to the style loss and *E* at layer *I* is;

$$E_l = \frac{1}{4N_l^2 M_l^2} \sum_{i,j} \left(G_{ij}^l - A_{ij}^l\right)^2$$

G is the Gram matrix for *X*, generated image and *A* is the Gram matrix for content image. *N* and *M* are the gram matrix sizes.

The Gram Matrix for an image matrix *F*, is calculated using inner products as follows:

$$G_{ij}^l = \sum_k F_{ik}^l F_{jk}^l$$

So, for the entry at position *(i, j)* of the Gram matrix, multiply the entries at position *(i, k)* with the entries at *(j, k)* for all the *k*'s and add them together

- *Literature review on stylizing the images*

One of the related works is GANs [5] (generative adversarial networks) in which there are two networks, discriminator and generator, competing with each other in min-max game framework. The discriminator is a typical classifier which has to classify if the image is real or generated by the generator. The generator, on the other hand, has to generate an image from the input of a random noise vector with the objective to fool the discriminator to classify it as a real image. Over the training procedure, the generator learns from the discriminator to be able to produce the realistic images and gets better at it over time.
GANs are widely used as a style transfer technique. CycleGANs [4] is a popular GAN technique to style the original image. Two GANs are used in CycleGANs. First GAN works on the unpaired images in which lets say the horse image is transformed into the zebra image. The second GAN then transforms that fake zebra image back into the horse image. As a result that horse image has effectively been stylized by the zebra image generated by the first GAN.

The common theme between the style transfer through GANs and CNNs style transfer is that both techniques make use of the CNNs to learn the image features. GAN is an advanced technique which stylizes the main object (like horse) with the artefacts of the style image (like with zebra stripes). CNN style network stylizes the content image's background. GAN is a better technique for style transfer but it is very difficult to train and has architectural constraints to perform well. [6] points out that after extensive model exploration, a family of architecture was identified to work well on the range of images. [6] give out specific details of the hyper-parameter values that worked. GANs are still at early stages of the research which require specific tricks and tips to make it work well. This can be seen as a problem requiring more research endeavor to find the systematic and robust design of the GAN architecture.

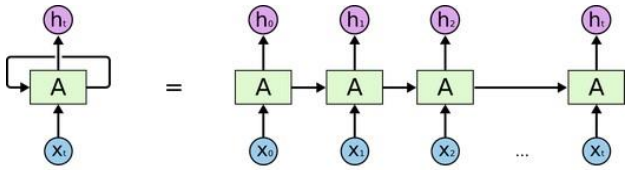
Figure is the output of the Cyclegan taken from [4] where G is the function of first GAN and F is the function of the second GAN.

- *Background on Image Captioning technique implemented*

In this project, the encoder and decoder framework has been implemented for the image caption generator in which the encoder is the pre-trained ResNet model and the decoder is the LSTM producing the sequence of text description.

ResNet is used as a feature extractor in which the image(s) is passed and the second last layer is used to extract the features of that image. The classifier layer of the ResNet has been cut off as we are interested in feature extraction and not in the classification. There's a trend in the area of neural networks which is "the Deeper the Better" but then the deep models encounter the problem of vanishing gradient descent. Residual networks (ResNets) [3] adds the skip connections to form residual blocks in which the layer's activation function is added element-wise with the some previous layer's input identity function. This enables the network to have a large number of layers without encountering the problem of vanishing gradient descent.

LSTM has been used as the decoder to learn and produce the text captions.The LSTMs are a sort of recurrent neural network (RNNs). In RNNs, the weight parameters are distributed by the hidden nodes. That is, at 't' time, the info transmitted to the hidden state comes from both the input unit and the hidden units of prior timestampsAs a result, the input nodes and previous hidden phases are fused before the weights are multiplied. As an outcome, back-propagation is carried out over time., taking into account the inputs from prior timestamps.



An unrolled recurrent neural network.

In reality the LSTM network is implemented as given below at the left side of the diagram i.e. as the rolled network. The right-side of the diagram is just for the people to understand all the math (it's like we are "imagining" how the network will look like by unrolling).

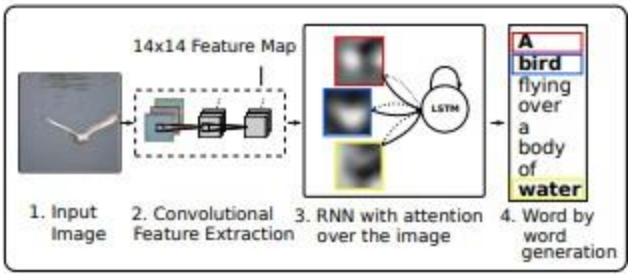
The image feature representations and the text embedding (explained later) are then concatenated together and then stacked with the fully connected layer with ReLU activation and then softmax classifier is used to predict the image captions.

- *Literature review on captioning the images*

Sequence-to-sequence models [7] are deep learning models that have achieved a lot of success in tasks like machine translation, text summarization, and image captioning.
A sequence-to-sequence model is based on an encoder and decoder framework. The encoder processes each item in the input sequence, it compiles the information it captures into a context vector. The decoder receives the context over and begins creating the output sequence item by item. For image captioning, encoder can be CNN with image features as input and the decoder can be LSTM/RNN that gives out the sequence of text description.

The information bottleneck problem is a major challenge for the sequence-to-sequence concept. The context vector provided to the decoder does not take into account the relative relevance of particular elements to the output in order. The attention mechanism takes care of this. The decoder can access all the encoder outcomes and look at various areas of encoder outcomes at various times using the attention technique. The simple-model, on the other hand, just has access to encoder's end stages.
This attention mechanism makes the image captioning by seq2seq better than this project implementation.

Figure 1. Our model learns a words/image alignment. The visualized attentional maps (3) are explained in section 3.1 & 5.4



This figure is taken from [8] which essentially has same theme as the image caption build in this project

- Higher level issues in Computational Creativity

[2] proposes three 'essential properties' that have come to be known as Ritchie's Criteria for evaluation that I've based my system on for evaluation.

1. *Novelty: To what extent is the produced item dissimilar to existing examples of its genre?*

The CNN style transfer and image captioning are not novel by themselves. But combining these two models as a system can be seen as somewhat novel or innovative.

2. *Quality: To what extent is the produced item a high quality example of its genre?*

The quality of the output is not high. The system does not pass the turing tests. Because the captions generated are not human-like responses.

3. *Typicality: To what extent is the produced item an example of the artefact class in question?*

The generated image is rendered by the styling effects and then it's artefacts are described in the text.

This AI system is also creative as per one of Margerat Boden's ways of creativity which is combinatorial creativity [1] described as 'making unfamiliar combinations of familiar ideas'. This is more like remixing things in new ways - but it requires appreciation and understanding to identify good mixes. Here, we are mixing the style network's output with the image caption generation which coincides with Margerat's thoughts.

The system built in this project also coincides with all three points of the Colton's tripod evaluation.

1. *Skill refers to the technical skill required to create something.*

The system is based on deep learning which requires technical skills.

2. *Appreciation refers to the ability to evaluate its own work*

The image captioning tries to describe the stylized image produced by the CNN style network. If the image captioning generates the text about the artefacts produced by the style network then this can be seen as the system having the ability to evaluate its own work.

3. *Imagination refers to the ability to go outside of existing knowledge*

Combining a stylizing technique with the captioning task can be seen as an innovative approach which requires some imagination.

Experiment and Results

- *Dataset and the pre-processing techniques used in the CNN style network*

I've used my own picture as a content image to be stylized by the CNN network. My stylized picture along with the original image was then given to the image caption model to generate text.

Images were pre-processed so that it can be fed to the VGG in a required size (width x height), adding batch size as another dimension, the pixel values were normalized to zero-centered with regards to the ImageNet dataset and in BGR color channel format.

The style image needs to be contrastive to the content image for better stylizing effects. For example, the content image will not be so much contrastive to the blue ocean or sky probably because there's a blue jacket in it.

- Dataset and the pre-processing techniques used in the image caption model

The LSTM decoder was trained on flickr8k dataset that contains the images with their 5 corresponding text. This is what the authors of the Flickr8k has written about the dataset;

"The images were chosen from six different Flickr groups, and tend not to contain any well-known people or locations, but were manually selected to depict a variety of scenes and situations."

A glove model is utilised to learn the representation of the words. Embedding Word [10] is a method for compressing a high dimensional word depiction into a small, dense vector space. When using a neural network to train, this technique captures semantic relationships between words. Because 'king' and 'queen' are semantically comparable because The learnt vectors for both words will be closer in distance if they appear in the same context.

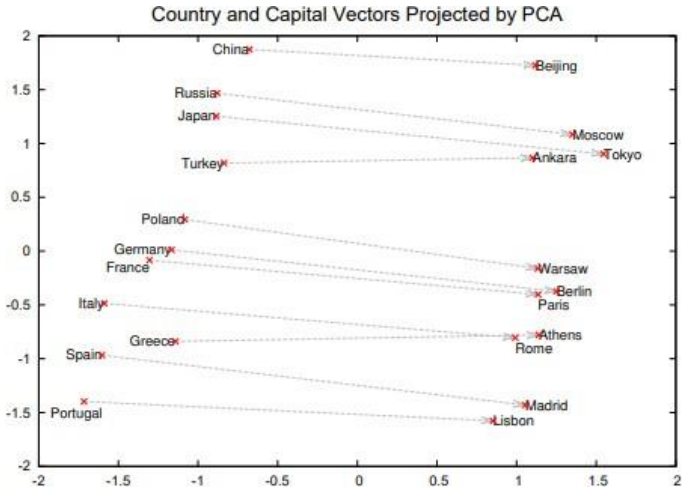


Figure of word embedding mapped to 2D is taken from [10]

For text processing, the padding has been used with a certain maximum length that enables a fixed size of the word sequence. We need a fixed matrix to be used by the neural network.

- *Experimental Study*

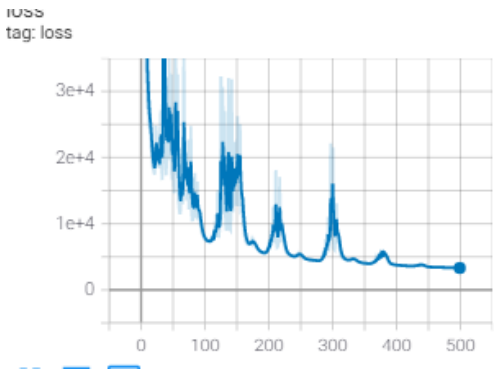
For the CNN style network, the experiments of 500 iterations each were carried out to try out various hyper parameters like initial-learning-rate, weights of content & style image, and the size of VGG network.

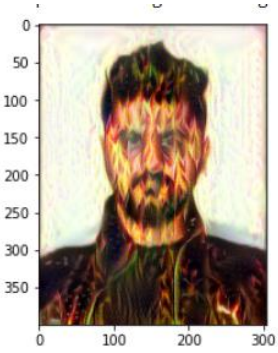
<u>1st Configuration:</u>

VGG19
Learning rate= 100
Content Weights = 2.5e-8
Style Weights = 1.5e-6

This is the default setting of the network from which further experiments are carried out to fine tune the hyper-parameters values.

Final loss = 3304.5



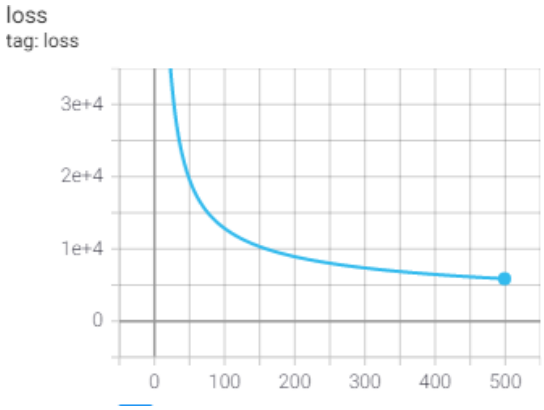


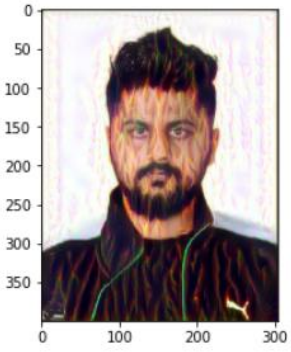
2nd Configuration

VGG19
Learning rate= 25
Content Weights = 2.5e-8
Style Weights = 1.5e-6

Final loss = 5865.9126

As a result of decreasing the initial learning rate, the model's loss function decreased stably but slowly with the higher final loss. There's also less style transfer with lower learning rate value.



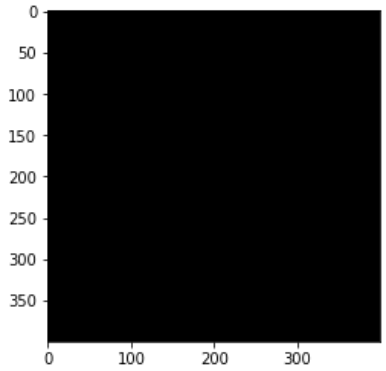


3rd Configuration

VGG19
Learning rate= 500
Content Weights = 2.5e-8
Style Weights = 1.5e-6

Final loss = 211574

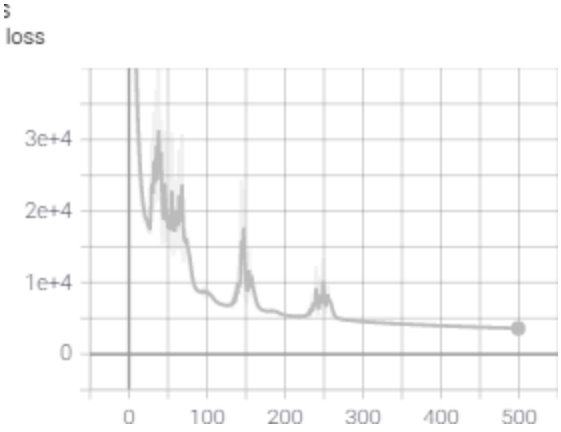
Having a large initial learning rate value caused instability in the learning of the network's parameters. Therefore, the stylized image failed to be generated.

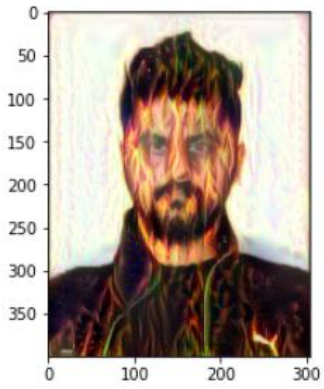


4th Configuration

VGG19
Learning rate= 75
Content Weights = 2.5e-8
Style Weights = 1.5e-6

Final Loss = 3890.1





In conclusion, the learning rate of 0.75 has been selected for further experiments as it provides stable training.

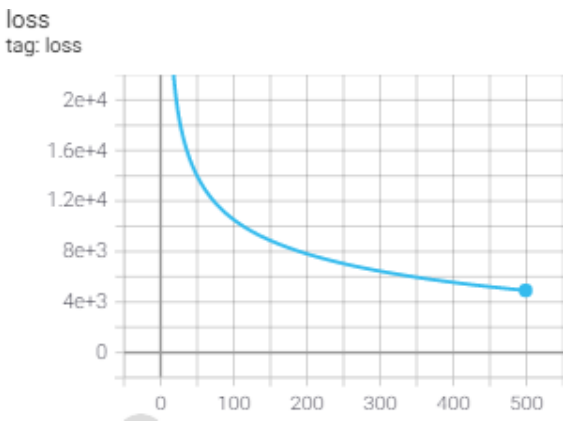
5th Configuration	6th Configuration
VGG19 **Learning rate= 75** **Content Weights = 2.5e-3 (increase)** **Style Weights = 1.5e-8 (decrease)**	***VGG19*** ***Learning rate= 75*** ***Content Weights = 2.5e-10 (decrease)*** ***Style Weights = 1.5e-3 (increase)***

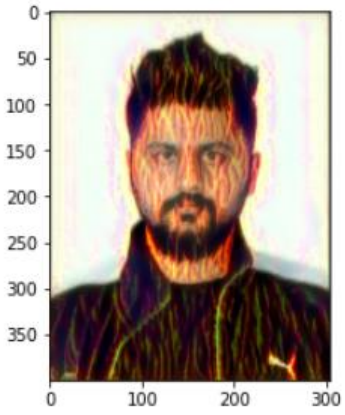
I found that both the above configurations were highly unstable in training and the stylized image failed to be generated. The above experiments did not worked.

7th Configuration

VGG16
Learning rate= 75
Content Weights = 2.5e-8
Style Weights = 1.5e-6

Final Loss = 4908.8

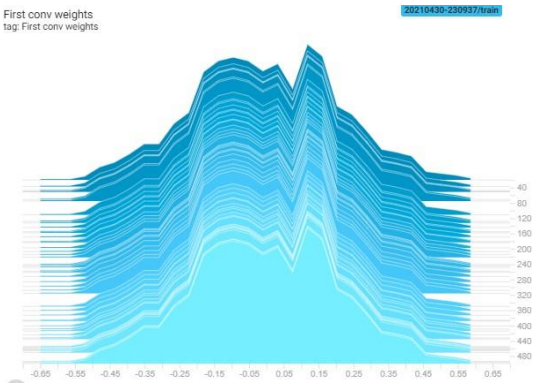




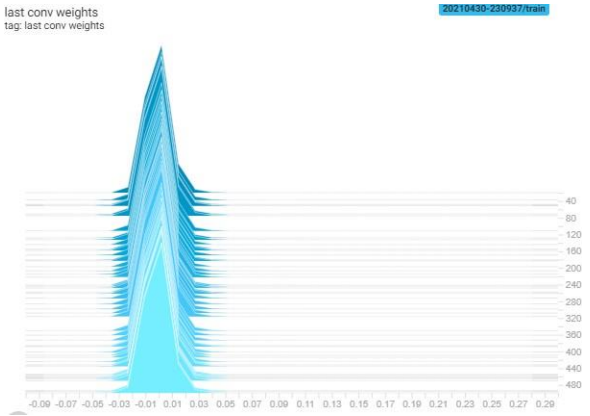
In this configuration, a smaller VGG model is used that has a lesser number of layers. As a result, training time decreased slightly from 4.5 mins to 4.25 mins.

Throughout the experiments, the weights of the final convolution layers are less distributed than the weight values of the first convolution layers. It can be because the initial layers tend to capture the lower level features of the image while the final last few layers learn the higher level image feature representation.

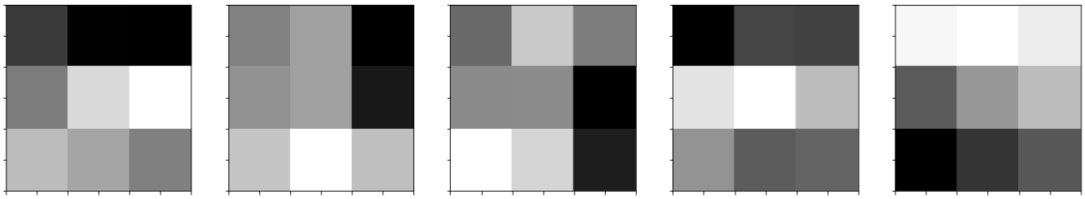
Histogram of first Convolution weight distribution display



Histogram of Last convolution weight distribution display



Below is the visualization of the some weight filters at the last VGG convolutional layers

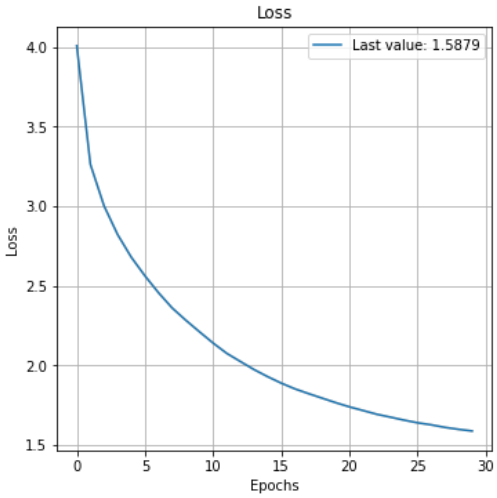


Below is the visualization the CNN feature maps of 8th VGG convolutional layer

Below is the visualization the CNN feature maps of 1st VGG convolution layer

- *Image Captioning Experiments*

1st Configuration: Without Dropout and with ReLU

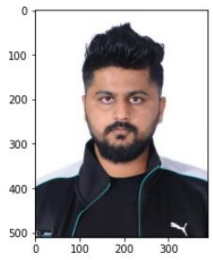


As expected the loss has decreased faster because there's no dropout used. Dropout makes the training and the convergence slower.

```
BLEU Score for image id : 3601978895_9fec23ce0c.jpgis : 0.35831291876413535
prediction: two men lean off the city street while one is carrying toy walk is
(500, 374, 3)
```

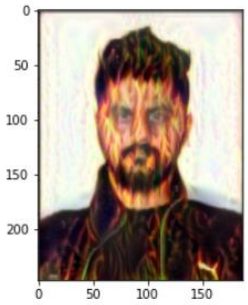
```
/opt/conda/lib/python3.7/site-packages/nltk/translate/bleu_score.py:490: UserWarn
Corpus/Sentence contains 0 counts of 3-gram overlaps.
BLEU scores might be undesirable; use SmoothingFunction().
  warnings.warn(_msg)
```

Without the use of dropout, the model predicts the train images really well. The Bleu is also low here.



```
photo_enc = encode_image('/content/Resources/pictures/me.jpg').reshape((1,2048))
predict(photo_enc,model)
```

```
'man with cloth hat and writing'
```

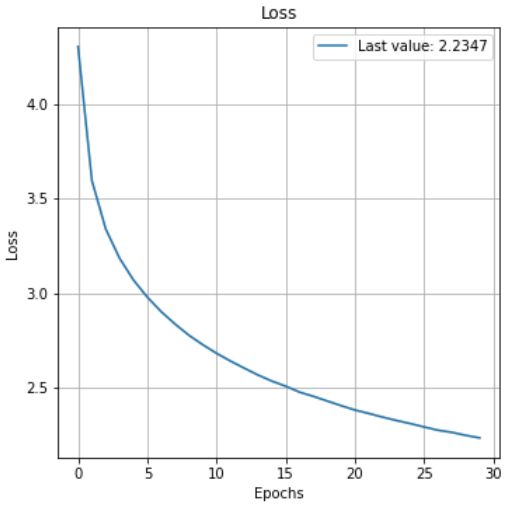


```
photo_enc = encode_image('/content/Resources/style_pictures/me_fire.png').reshape((1,2048))
predict(photo_enc,model)
```

```
'young boy is sitting in the street with flowers and paint drawing happily around him'
```

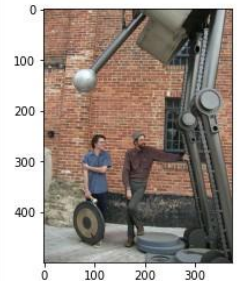
Here, the model has not generated as good captions on the test images as on to the training images. It can be because without dropout, the model can overfit and fail to generalize well on to the unseen data. The model still has managed to capture style of fire. For example, the model has generated ‘drawing’ and ‘paint’ in the caption for the stylized image.

2nd Configuration With Dropout and sigmoid

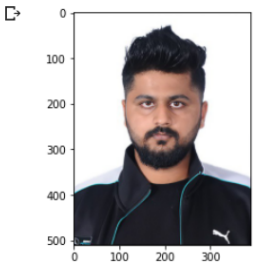



```
BLEU Score for image id : 3601978895_9fec23ce0c.jpgis : 0.7071067811865476
prediction: man in red shirt is standing on his head and is sitting on the ground next to another person in front of some soda
(500, 374, 3)
```

```
/opt/conda/lib/python3.7/site-packages/nltk/translate/bleu_score.py:490: UserWarning:
Corpus/Sentence contains 0 counts of 2-gram overlaps.
BLEU scores might be undesirable; use SmoothingFunction().
  warnings.warn(_msg)
```



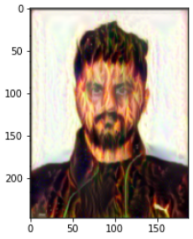
The BLEU score is highest in this configuration. BLEU has been used to evaluate the image caption model's predictions quantitativelyBLEU calculates the n-gram precision of the model's created text in comparison to the example texts. The number tokens utilised in generated text contained in the references, for example, will be computed using the unigram precision. The word sequence is examined further by the n-grams precision. By default, the geometric mean of the 4-gram precisions is used by BLEU in the NLTK package to calculate the score.
However, when measuring the comparison between the generated text and the text reference, BLEU can be deceiving.




```
[ ] photo_enc = encode_image('/content/Resources/pictures/me.jpg').reshape((1,2048))
    predict(photo_enc,model)
```

```
'the boy is wearing black shirt and standing next to the boy wearing black jacket'
```

```
style_img = plt.imread('/content/Resources/style_pictures/me_fire.png')
plt.imshow(style_img)
plt.show()
```

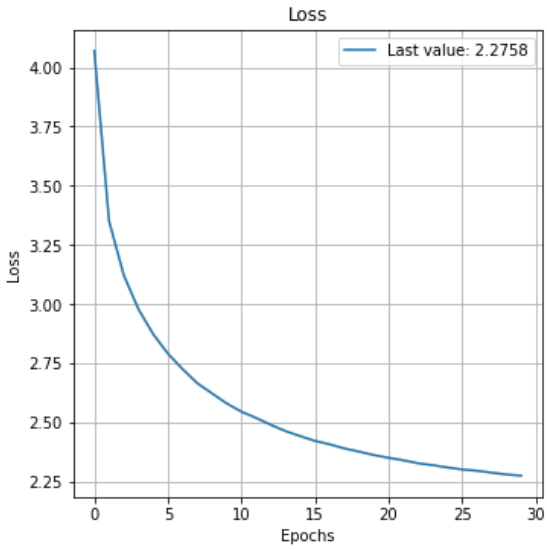



```
photo_enc = encode_image('/content/Resources/style_pictures/me_fire.png').reshape((1,2048))
predict(photo_enc,model)
```

```
'man in black hat is standing next to two video'
```

The captions generated are not right. The sigmoid activation function is not ideal configuration.

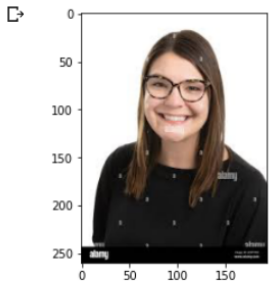
3rd Configuration Dropout and ReLU




```
BLEU Score for image id : 3601978895_9fec23ce0c.jpgis : 0.668740304976422
prediction: man lays on bench with his hand up to him
(500, 374, 3)
```

```
/opt/conda/lib/python3.7/site-packages/nltk/translate/bleu_score.py:490: Us
Corpus/Sentence contains 0 counts of 2-gram overlaps.
BLEU scores might be undesirable; use SmoothingFunction().
  warnings.warn(_msg)
```

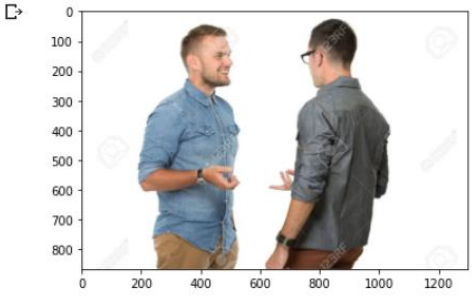
```
img2 = plt.imread('/content/Resources/pictures/pic.jpg')
plt.imshow(img2)
plt.show()
```



```
[ ] photo_enc = encode_image('/content/Resources/pictures/pic.jpg').reshape((1,2048))
    predict(photo_enc,model)
```

```
'man and woman are smiling'
```

```
style_img = plt.imread('/content/Resources/style_pictures/nn.webp')
plt.imshow(style_img)
plt.show()
```



```
[ ] photo_enc = encode_image('/content/Resources/style_pictures/nn.webp').reshape((1,2048))
    predict(photo_enc,model)
```

```
'two men are standing next to each other'
```

The model in this configuration has captured some stylized representation reflected onto the image.

In the end, I chose the usage of the Dropout and ReLU because these configurations seem to get better text generated than the rest of the configurations.

Discussion

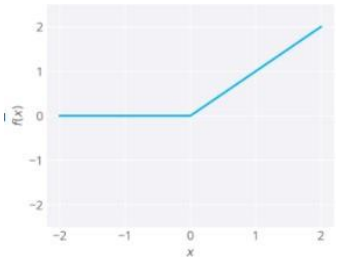
The creative system evaluation is a critical topic in the field of computational creativity. The system built in this project resolves issues like the system can itself evaluate its own generated output. In the experiments, it can be seen that the image generator was able to identify artefacts of the style image with amusing related words like ‘flower’, ‘paint’, ‘lighting’.

In the CNN style transfer, VGG19 is chosen because it stylizes the image more. The learning rate should not be high that can cause instability in learning but also not low that slows down the convergence. Therefore, the initial learning rate value of 75 seems to be reasonable. It seems that the network is highly sensitive to the content and style weights in the loss function. It is quite difficult to find their right values and so needs to be referenced from the related research literature. The values of the content and style weights from default configuration are used as final configuration values.

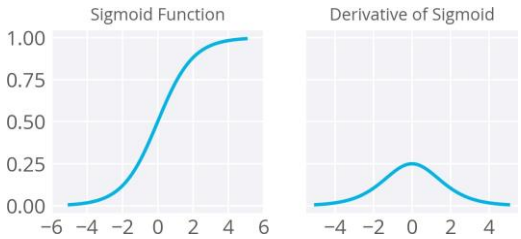
The dropout has been applied to the picture caption model, which is intended to avoid overfitting. Dropout is dropping the units at random with the given probability during the training state. This adds noise to the network's units, making it difficult for them to co-adapt with one another.[4] During the test time, the network’s weights are scaled up by the same probability value. Instead of sigmoids, most recent deep learning networks use rectified linear units (ReLUs) in the hidden layers. Mathematically, ReLU looks like;

f(x) = max(x, 0), where x is the input units or hidden units

Graphically ReLU is depicted as;



When the input is positive, the derivative is 1, so there isn't the vanishing effect when the error is back-propagated.



The derivative of the sigmoid maxes out at 0.25 (see above). This means during backpropagation with sigmoid units, the errors will be shrunk by at least 75% at every layer. For

layers close to the input layer, the weight updates will be tiny which can be seen as a vanishing gradient descent problem. Due to this, sigmoids have fallen out of favor as hidden activations function.

Conclusion and Future Work

Both approaches [1] and [2] emphasise the importance of people in the evaluation process. It's impossible to talk about the impact of creativity without referencing people somehow in the process. Therefore, the further work can be to deploy the web application that can be accessed by the entire world. In this way, more feedback can be received about the system to evaluate how creative the system is.

The system in this project does not pass the turing tests which is about building a machine that thinks or responds like a human. This is another evaluation method that cannot be addressed within this project. The further extension to improve the performance of the implemented techniques will be to build the CycleGANs instead of CNN for style transfer and then make use of the image captioning based on attention to identify the artefacts of the images with the text description which can seem human-like responses as per the turing tests.

Both GANs and attention based techniques have already made a huge impact on the computation creativity community. DeepFakes is the synthetic media application, in which a person in an existing image or video is replaced with someone else's likeness. This application uses GANs.

Attention based models like transformers have a lot of potential to make advancements in the computational creativity research and practice. Transformer is a stack of encoders and decoders. OpenAI has already made significant progress in computational creativity by creating the models like GPT which are based on transformers. GPT models generate coherent paragraphs of text, achieving state-of-the-art performance on many language modeling benchmarks.

Reference

[1] Margaret Boden. The Creative Mind: Myths and Mechanisms https://www.routledge.com/The-Creative-Mind-Myths-and-Mechanisms/Boden/p/book/9780415314534

[2] Graeme Ritchie. Some Empirical Criteria for Attributing Creativity to a Computer Program https://dl.acm.org/doi/10.1007/s11023-007-9066-2

[3] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In Proceedings of CVPR, pages 770–778, 2016. https://arxiv.org/pdf/1512.03385.pdf

[4] Jun-Yan Zhu, Taesung Park, Phillip Isola, Alexei A. Efros. Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks https://arxiv.org/abs/1703.10593

[5] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, Yoshua Bengio. Generative Adversarial Networks https://arxiv.org/abs/1406.2661

[6] Ian Goodfellow NIPS 2016 Tutorial: Generative Adversarial Networks https://arxiv.org/pdf/1701.00160.pdf

[7] Ilya Sutskever, Oriol Vinyals, Quoc V. Le. Sequence to Sequence Learning with Neural Networks https://arxiv.org/abs/1409.3215

[8] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhutdinov, Richard Zemel, Yoshua Bengio. Show, Attend and Tell: Neural Image Caption Generation with Visual Attention https://arxiv.org/pdf/1502.03044.pdf

[9] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, Illia Polosukhin. Attention Is All You Need https://arxiv.org/pdf/1706.03762.pdf

[10] Tomas Mikolov, Kai Chen, Greg Corrado, Jeffrey Dean Efficient Estimation of Word Representations in Vector Space https://arxiv.org/abs/1301.3781

[11] Leon A. Gatys, Alexander S. Ecker, Matthias Bethge. A Neural Algorithm of Artistic Style https://arxiv.org/abs/1508.06576

[12] Karen Simonyan, Andrew Zisserman. Very Deep Convolutional Networks for Large-Scale Image Recognition https://arxiv.org/abs/1409.1556