Published by :
http://www.ijert.org

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 8 Issue 10, October-2019

A Neural Approach for Image Styling

P. Malathi[1]
[1]Master of Computer Applications Department,
Dr. Ambedkar Institute of Technology,
Visvesaraya Technological University, Bengaluru, India

***Abstract*— Painters are able to create delightful visual experiences using their imaginations and combining content and style of different images. It is more of a laborious process, even using the available artistic digital tools. So far the algorithmic basis of this process was not dealt and there exists no artificial system with similar capabilities. Over the years, in the related fields of visual perception such as face and object recognition near human performance was demonstrated by the class of biologically inspired vision models called Deep Neural Networks under Artificial Intelligence [1,2]. Here we introduce an artificial system based on a Deep Neural Network that creates artistic images of high perceptual quality. The system uses neural representations to separate and recombine the style and content of arbitrary images, providing a neural algorithm for developing artistic images effectively and efficiently.**

Keywords—Deep Neural Network, CNN, VGG-19, feature, style, loss, variation, average pooling.

I. INTRODUCTION

The class of Deep Neural Networks that are most powerful and efficient in image processing tasks are called Convolutional Neural Networks. Also, they use less preprocessing compared to other algorithms. They consist of layers of small computational units that process visual information hierarchically in a feed-forward manner(Fig 1). Each layer of units can be understood as a collection of image filters, each of which extracts a certain feature from the input image. Thus, the output of a given layer consists of the so-called feature maps: differently filtered versions of the input image.

II. RELATED WORK

Earlier work on separating content from style was carried on input images of much lesser complexity such as images of faces or small figure in different poses or characters of different handwritings. The problem was approached in the branch of computer vision called non photorealistic rendering. Mainly methods used were similar to the texture transfer to achieve style transfer. These approaches mainly rely on non-parametric techniques to directly manipulate the pixel representation of an image. Thus the algorithms were heavily dependent on input image cases.

After that work was carried out for style recognition in order to classify artworks according to the period [8]. There the features from Deep Neural Networks trained on object recognition was used and the classifiers were trained on top of the raw network activations, which we call content representations.

Approaching Deep Neural Networks in a better way, we carry out manipulations in feature spaces that effectively represent high level content of image irrespective of input cases.

III. PROPOSED METHOD

CNN, when trained on object recognition, develop a representation of the image that makes the object information increasingly explicit along the processing hierarchy[3] The input image is thus transformed into representations that increasingly care about the actual content of the image compared to its detailed pixel values. We can directly visualise the information each layer contains about the input image by reconstructing the image only from the feature maps in that layer[4]. Higher layers of the network extract the high-level content in terms of objects and their arrangement in the input image but do not constrain the exact pixel values of the reconstruction. In contrast, reconstructions from the lower layers simply reproduce the exact pixel values of the original image. Hence we refer to the feature responses in higher layers of the network as the content representation.

To obtain a representation of the style of an input image, we use a feature space originally designed to capture texture information[3]. This feature space is built on top of the filter responses in each layer of the network. It consists of the correlations between the different filter responses over the spatial extent of the feature maps. By including the feature correlations of multiple layers, we obtain a stationary, multi-scale representation of the input image, which captures its texture information but not the global arrangement.

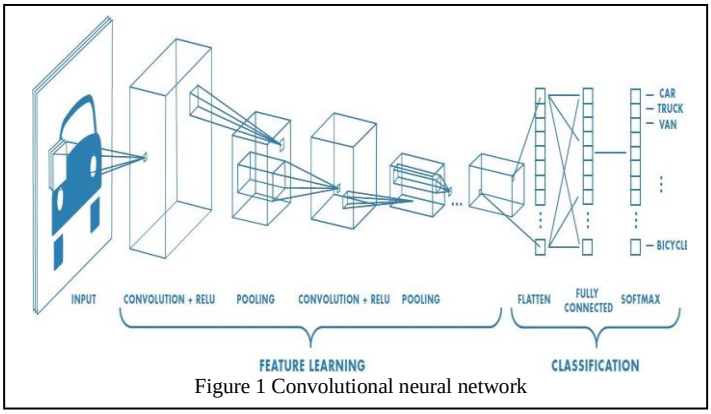

Figure 1 Convolutional neural network

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 8 Issue 10, October-2019

Both representations can be manipulated independently to produce new, perceptually meaningful images. Of course, the image content and style cannot be completely disentangled as there usually does not exist an image that perfectly matches both constraints at the same time. However, the loss function we minimise during image synthesis contains two terms for content and style respectively, that are well separated . We can therefore smoothly regulate the emphasis on either reconstructing the content or the style. On case basis also, for a particular pair of source images one can adjust the trade-off between content and style to create visually appealing images.

VGG(Visual Geometry Group)- 19 Network[5] (Fig 3) a Convolutional Neural Network is made of many convolutional and pooling layers. The input is a fixed 224x224 RGB image. The mean RGB value is subtracted as part of preprocessing. 3x3 sized filters are used for convolution and 2x2 Max pooling layers are used. For classification tasks fully connected layers are used in the end to calculate the softmax loss.

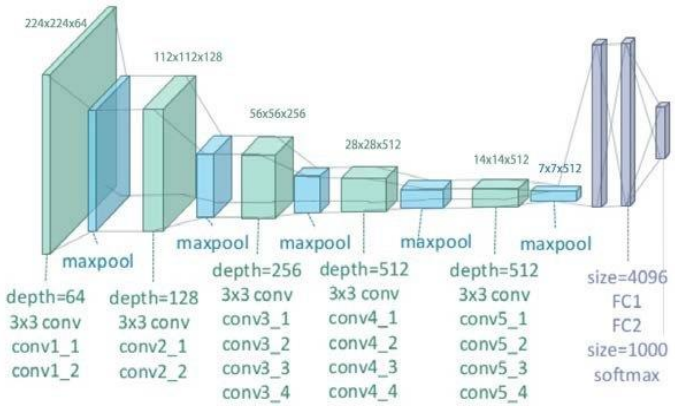

Figure 2 VGG – 19

The network is close to human performance on a common visual object recognition benchmark task[6] and was introduced and extensively described in[5] The feature space provided by the 16 convolutional and 5 pooling layers of the 19 layer VGG Network is used. We do not use any of the fully connected layers. The model is publicly available and can be explored in the caffe-framework[7] . For image synthesis it was found that replacing the max-pooling operation by average pooling improves the gradient flow and hence provide slightly more appealing results.

The VGG network constructed for style transfer does not have the final fully connected layers. The pretrained weights are loaded into this network and are made constant throughout the training. The generated image is initialized with the content image and is considered as the variable parameter (Note that the weights are variable in most of the training tasks)

The algorithm is as follows:

1. Load and preprocess the content and style image
2. Initialise the final image with the content image
3. Extract the features of all the 3 images (Content, Style and new generated image)
4. Calculate the loss which is a sum of Content loss + Style loss + Total variation loss.
5. Use Adam optimizer to reduce the loss and modify the new generated image.

Preprocessing - The content and style images are loaded and reshaped to 224x224x3 dimensions since the pretrained VGG network accepts images of this dimension. The pixels values are subtracted from the VGG mean as part of preprocessing (123. 68, 116. 779, 103 .939 for RGB). These processes are reversed while generating the artistic images.

The loss function we minimise is calculated as –

$$\text{Loss} = L_c + L_s + L_{tv} \quad (1)$$

Content loss- Style loss- L_s

Total Variation loss- L_{tv}

Content loss is used to penalize deviations from the content of the content image and the generated image. Let

Let

$F^\ell \in \mathbb{R}^{M^\ell \times C^\ell}$ be the feature map for the current image and

$P_\ell \in \mathbb{R}^{M^\ell \times C^\ell}$ be the feature map for the content source image where $M_\ell = H_\ell \times W_\ell$ is the number of elements in each feature map. Each row of F^ℓ or P^ℓ represents the vectorized activations of a particular filter, convolved over all positions of the image. Finally, let wc be the weight of the content loss term in the loss function.

$$L_c = w_c \times \sum_{i,j} (F^\ell_{ij} - P^\ell_{ij})^2 \quad (2)$$

Where i,j,l stands for ith filter at position j in layer l.

Style loss measures the deviation of the style of the generated image from the style image. The Gram matrix is an approximation to the covariance matrix and it is used to calculate the style loss. Given a feature map F_ℓ of shape $(M_{\ell,\ell})$, the Gram matrix has shape (C_ℓ, C_ℓ) and its elements are given by:

$$G^\ell_{ij} = \sum_k F^\ell_{ki} F^\ell_{kj}$$

Where G^ℓ_{ij} is the inner product between the vectorised feature map i and j in layer l.

Assuming G^ℓ is the Gram matrix from the feature map of the current image, A^ℓ is the Gram Matrix from the feature map of the source style image, and w_ℓ a scalar weight term, then the style loss for the layer ℓ is simply the weighted Euclidean distance between the two Gram matrices:

$$L^\ell_s = w_\ell \sum_{i,j} (G^\ell_{ij} - A^\ell_{ij})^2$$

In practice we usually compute the style loss at a set of layers L rather than just a single layer ℓ; then the total style loss is the sum of style losses at each layer:

$$L_s = \sum_{\ell \in L} L^\ell_s \quad (3)$$

Total variation loss is used to smoothen the image. We compute it as the sum of the squares of differences in the pixel values for all pairs of pixels that are next to each other (horizontally or vertically). Here we sum the total-variation regularization for each of the 3 input channels (RGB), and weight the total summed loss by the total variation weight, w_t:

$$L_{tv} = w_t \times \left(\sum_{c=1}^{3} \sum_{i=1}^{H-1} \sum_{j=1}^{W} (x_{i+1,j,c} - x_{i,j,c})^2 + \sum_{c=1}^{3} \sum_{i=1}^{H} \sum_{j=1}^{w-1} (x_{i+1,j,c} - x_{i,j,c})^2 \right) \quad (4)$$

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 8 Issue 10, October-2019

Here in this implementation we consider the features of 13th layer of VGG network to calculate the content loss. For the style loss we consider the feature maps generated by the layer number 1,4,7,12 and 17.

Finally the Adam optimization is used to reduce the loss generated. It is used instead of the classical stochastic gradient descent procedure to update network weights iterative based in training data. It's features are- Straightforward to implement, Computationally efficien Little memory requirements, Invariant to diagonal rescale the gradients. Well suited for problems that are large in ter of data and/or parameters, Appropriate for non-station objectives, Appropriate for problems with very noisy sparse gradients, Hyper-parameters have intuit interpretation and typically require little tuning.

Different relative weightings of the content and st reconstruction loss lead to different results. By giv emphasis on content (higher content weight) the photogra can be clearly identified but the style of the painting is well matched. In the same way if emphasis is given to styl texturized version of the style image is generated with l content. The local image structures captured by the st representation increase in size and complexity wh including style features from higher layers of the network.

IV. IMPLEMENTATION RESULTS

The algorithm was implemented using Anaconda framework, python language. Class libraries numpy, keras facilitated Deep Neural Network functions. The sample input and output images are shown in fig 3-5.

Figure 1 - Content, Style Images

Figure 3 - Initial Iteration Images

Figure 2 - Final Iteration Images

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 8 Issue 10, October-2019

V. CONCLUSION

All in all it is truly fascinating that a neural system, which is trained to perform one of the core computational tasks of biological vision, automatically learns image representations that allow the separation of image content from style. The explanation could be that when learning object recognition, the network has to become invariant to all image variation that preserves object identity. Representations that factorise the variation in the content of an image and the variation in its appearance would be extremely practical for this task. Thus, our ability to abstract content from style and therefore our ability to create and enjoy art might be primarily a preeminent signature of the powerful inference capabilities of our visual system.

Replacing the max-pooling operation by average pooling improves the gradient flow and one obtains slightly more appealing results. Visually appealing images can be generated using this technique of CNN architectures as they are highly suitable for such tasks involving images . By varying the content and style weights different visualizations can be obtained. Also, the layers of the feature maps considered to calculate the loss can be modified to get different results.

The loss calculation is simple and can be implemented using libraries that specialize in optimized computations (e.g. Numpy). For the CNN construction popular libraries like Tensorflow (with Keras) or Pytorch can be used. These libraries have various built in functions which help in constructing the CNN layers very easily.

REFERENCES

[1] Krizhevsky, A., Sutskever, I. & Hinton, G. E. Imagenet classification with deep convolutional neural networks. In Advances in neural information processing systems, 1097–1105 (2012). URL http://papers.nips.cc/paper/4824-imagenet.

[2] Taigman, Y., Yang, M., Ranzato, M. & Wolf, L. Deepface: Closing the gap to human-level performance in face verification. In Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on, 1701–1708 (IEEE, 2014). URL http://ieeexplore. .eee.org/xpls/abs_all.jsp? arnumber=6909616

[3] Gatys, L. A., Ecker, A. S. & Bethge, M. Texture synthesis and the controlled generation of natural stimuli using convolutional neural networks. arXiv:1505.07376 [cs, q-bio] (2015). URL http://arxiv.org/abs/1505.07376. ArXiv: 1505.07376.

[4] Mahendran, A. & Vedaldi, A. Understanding Deep Image Representations by Inverting Them. arXiv:1412.0035 [cs] (2014). URL http://arxiv.org/abs/1412.0035. ArXiv: 1412.0035.

[5] Simonyan, K.&Zisserman, A. VeryDeepConvolutional Networksfor Large-Scale Image Recognition. arXiv:1409.1556 [cs] (2014). URL http://arxiv.org/abs/1409. 1556. ArXiv: 1409.1556. Karayev, S. et al. Recognizing image style. arXiv preprint arXiv:1311.3715 (2013). URL http://arxiv.org/abs/1311.3715

[6] Russakovsky, O. et al. ImageNet Large Scale Visual Recognition Challenge. arXiv:1409.0575 [cs] (2014). URL http://arxiv.org/abs/1409.0575. ArXiv: 1409.0575.

[7] Jia, Y. et al. Caffe: Convolutional architecture for fast feature embedding. In Proceedings of the ACM International Conference on Multimedia, 675–678(ACM,2014). URLhttp: //dl.acm.org/citation.cfm?id=2654889.

[8] Karayev, S. et al. Recognizing image style. arXiv preprint arXiv:1311.3715 (2013). URL http://arxiv.org/abs/1311.3715.