Chapter 11
Image GPT with Super Resolution

Bhumika Shah, Ankita Sinha, and Prashant Saxena

Abstract A Generative Pre-trained Transformer (GPT) model which can generate text by looking at previous text was trained to generate image pixels sequentially by making a correlation between the image classification accuracy and the image quality. This model uses the generative model for generating images. The Image Generative Pre-trained Transformer (IGPT) works on a low-resolution image which in turn produces a low-resolution output. In this paper, we have attempted to eliminate this limitation by enhancing the resolution of the output image produced by IGPT. The primary focus during this research work is to check different models and choose the simplest model for improving quality of the image generated because there are several models that support deep neural networks that have been successful in upscaling the image quality with great accuracy for achieving super resolution for a single image. The output image of low resolution is upscaled to high-resolution space employing a single filter and bicubic interpolation. We have also considered peak signal-to-noise ratio (PSNR) score and structural similarity (SSIM) value to analyze the standard of the image produced by the algorithm. The proposed approach has been evaluated using images from publicly available datasets. We have used leaky ReLU instead of ReLU as the activation function which produces better PSNR score and SSIM value, improving the overall result. Combining efficient sub-pixel convolutional neural network (ESPCNN) algorithm with IGPT, we have managed to get better output compared to the output generated by IGPT solely.

11.1 Introduction

The OpenAI Team trained GPT-2 [1] on images by dividing them into long sequences of pixels, which they called IGPT [2]. OpenAI found that the model appears to recognize the characteristics of a 2-D image, like object appearance and category [2]. This can be proven by a range of coherent image samples it generates with no

B. Shah (✉) · A. Sinha · P. Saxena
Department of Computer Science, Gujarat University, Ahmedabad, India
e-mail: bhumikashah@gujaratuniversity.ac.in

S. C. Satapathy et al. (eds.), *Intelligent Data Engineering and Analytics*,
Smart Innovation, Systems and Technologies 266,
https://doi.org/10.1007/978-981-16-6624-7_11

help of labeled data. But the quality of the image generated by the OpenAI model was very low, and our efforts were to upscale the resolution of the image produced using a CNN-based model [3, 7] named efficient sub-pixel convolutional neural network (ESPCNN). The recovery of a high-resolution image from its low-resolution counterpart may be a topic of great interest in digital image processing.

Many popular methods work on the assumption that there are multiple low-resolution images of a scene having different perspectives that are available. These methods are classified as multi-image super resolution methods. These methods exploit the redundancy of the multiple images by repeatedly applying the same information during the sampling process. However, these methods usually require computationally demanding image registration and fusion stages for images, the accuracy of which directly impacts the standard of the result. Another family of methods is single image super-resolution (SISR) techniques. These techniques seek to find out implicit redundancy that is present in natural data to recover missing high-resolution information from one low-resolution instance which is completed through local spatial correlations for images.

In this research, we have also tried to experiment with different activation functions like tanh, ReLU, leaky ReLU in order that we are able to improve the ultimate results of our output.

11.2 Background and Related Work

11.2.1 IGPT Algorithm

Introduction. OpenAI trained IGPT-S [2] (small), IGPT-M [2] (medium), and IGPT-L [2] (large) transformers containing 76 M, 455 M, and 1.4 B parameters, respectively, on ImageNet [2]. They have also trained IGPT-XL [2] (extra large), a transformer that takes 6.8 billion parameters, on a combination of ImageNet and pictures from the net. There is a high computational cost for modeling long sequences with dense attention layers, which is the reason for training this model at low resolutions of 32×32, 48×48, and 64×64 pixels. The work will be done at an even lower resolution to further reduce the computational cost, but prior work has demonstrated that human performance on image classification begins to drop rapidly when the images are below these sizes. Instead, motivated by early color display palettes, OpenAI created their own 9-bit color palette [2] to represent pixels which help to provide an input sequence of a length that is three times shorter than the quality palette.

Architecture. The transformer model's decoder [8] takes an input sequence of discrete values and produces a d-dimensional embedding for every position. The mixing of the sequential elements of the image takes place only once, i.e., within the attention operation, and in order to make sure proper conditioning is applied when training the auto-regressive model [8]. After that, the ordinary upper triangular BERT mask is applied to an $n \times n$ matrix of attention logits [9]. When using the BERT objective [9], there is no requirement of attention logit masking. After applying content embeddings to the input sequence, the positions in M are zeroed out.

Pre-Training. For the pre-training purpose, an unlabeled dataset (X) containing data as $x = (\times 1, \times 2, \times 3, \ldots, \times n)$ is given, then different permutations are computed as π of the set [1, n] and the density is modelled auto-regressively. The formula for the same is given as [2]:

$$p(x) = \prod_{i=1}^{n} p(x_{\pi_i} | x_{\pi_1}, \ldots, x_{\pi_{i-1}}, \theta) \tag{11.1}$$

We have to pick up the permutation $=i$ for $1 \leq i \leq n$. Here, the main aim is to minimize the negative log-likelihood of this cost function can be given as [2]:

$$L_{\text{AR}} = \sum_{x \sim X} \left[-\log \log p(x) \right] \tag{11.2}$$

The BERT model [9] samples a sub-sequence $M[1, n]$ such that each index i has the probability of 0.15 of appearing within the set M where M is the BERT mask. Here, the aim is to attenuate the negative log-likelihood of the masked elements $\times M$ conditioned on the unmasked ones $x[1, n] \backslash M$. This function is given as [2]:

$$L_{\text{BERT}} = \sum_{x \sim X} \sum_{M} \sum_{i \in M} \left[-\log \log p(x_i | x_{\frac{[1,n]}{M}}) \right] \tag{11.3}$$

During the pre-training, AR [8] or Bert [9] can be used to minimize the loss over the pre-trained dataset.

Fine-Tuning. During the fine-tuning, IGPT averages the n^L pool across the sequence dimension to obtain one d-dimensional vector of features for each sample [2]:

$$f^L = \langle n_i^L \rangle_i \tag{11.4}$$

Projection from f^L to logits that utilizes minimization of cross entropy loss LCLF. During the fine-tuning of LCLF, it attains acceptable performance. Then, the empirical joint objective is found which can be given as below [10]:

$$L_{\text{GEN}} + L_{\text{CLF}} \tag{11.5}$$

Limitation of IGPT. One of the limitations of Image GPT is that it works on only low-resolution images while there are many other algorithms that work on high resolution, but they are based on supervised learning. Since the Image GPT works on low-resolution input images, the final output is also of low resolution. So when the pixelated image is generated through the standard interpolation or enlargement, a low-quality image is generated.

11.2.2 ESPCNN Algorithm

ESPCNN Structure. Suppose there are L layers in an SR network. For first L−1 layers, $f_1 \times f_1$ convolution is performed on the low-resolution input image to obtain n_{l-1} feature maps. An efficient sub-pixel convolution is performed at the final layer to induce the HR image as the output. We can conclude that it is a shallow network as the number of layers is 3. The parameters for each layer are: (f_1, n_1) which is (5, 64), (f_2, n_2) which is (3, 32) and $f_3 = 3$.

Layer No. 1: There are 64 filters with the filter of size of 5×5.
Layer No. 2: There are 32 filters with the filter of size of 3×3.
Layer No. 3: A single filter with a filter of size of 3×3.

For a YUV image [3], the Y parameter is taken into consideration as human eyes are more sensitive to the brightness, i.e., luminance than the color, i.e., chrominance of the image (Fig. 11.1).

Leaky ReLU as Activation Function. ReLUs are not without drawbacks. Generally, ReLUs are non-zero centered and are non-differentiable at zero, but differentiable elsewhere. Another problem is the dying ReLU problem, where some ReLU neurons essentially die for all inputs and remain inactive for all the other input which is supplied; here, we have no gradient flow and if a sizable number of dead neurons are there in a large network, the performance is affected which could be corrected by making use of Leaky ReLU in which the slope is modified left of $x = 0$, and thus, it causes a leak which extends the range of ReLU.

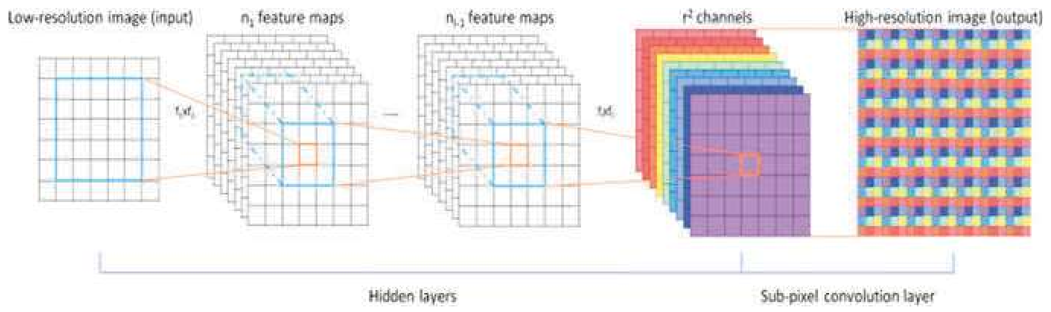

Low-resolution image (input)
n₁ feature maps
nₗ₋₁ feature maps
r² channels
High-resolution image (output)
f₁×f₁
fₗ×fₗ
Hidden layers
Sub-pixel convolution layer

Fig. 11.1 The efficient sub-pixel convolutional neural network (ESPCNN) has two convolution layers for the extraction of feature maps, and there is a single sub-pixel convolution layer which aggregates the feature that build the super resolution image employing a single step by mapping from low-resolution space [3]

PSNR (Peak Signal-to-Noise Ratio) Score [11]**.** The model will have a high PSNR score if the mean squared error (MSE) is low. This approach works well, but even with a high PSNR scores, the images might not look good to the human eye, and this can be a major downside of this approach. The way humans perceive image quality does not perfectly correlate to the PSNR score. Trying to minimize the MSE produces images which will look similar to the original but will not always look pleasing to the human eye.

$$\mathrm{PSNR} = 10 \cdot \frac{\mathrm{MAX}_J^2}{\mathrm{MSE}} \tag{11.6}$$

SSIM (Structural SIMilarity) Index [12]**.** The similarity of two images is compared using the SSIM index. The SSIM index value is often considered as a top quality measure of an image that is being compared; here, it is assumed that the referenced image is of high quality.

$$\mathrm{SSIM}(x, y) = \frac{\left(2\mu_x\mu_y + c_1\right)\left(2\sigma_{xy} + c_2\right)}{\left(\mu_x^2 + \mu_y^2 + c_1\right)\left(\sigma_x^2 + \sigma_y^2 + c_2\right)} \tag{11.7}$$

All distorted images have approximately the same mean squared error values when compared to the original image, but they have different quality. SSIM gives a better indication of image quality [12] (Fig. 11.2).

Comparison of ESPCNN with other models. ESPCNN uses residual blocks [3] rather than normal convolution layers. The success of architectures like ESPCNN popularized the idea that residual blocks are more powerful than simple convolutional layers because they allow using more layers without overfitting.

In ESPCNN, the upsampling step is done at center of the neural network. Generally, in an SR architecture, a step of upsampling is always involved. If we are using bicubic interpolation inside the network to upsample, then we can either use it at the

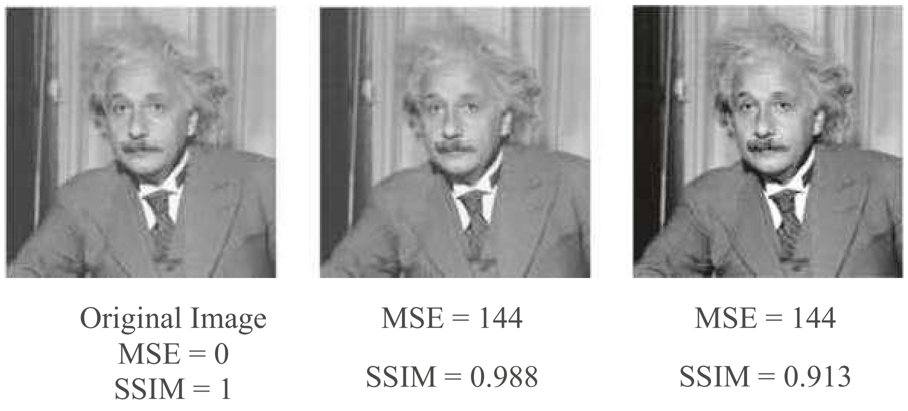


Fig. 11.2 Comparison between images with same MSE but different SSIM values [12]

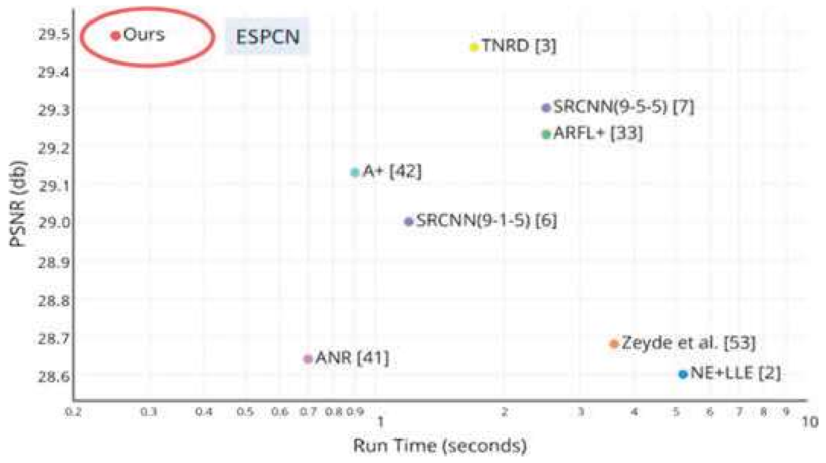


Fig. 11.3 Plot of the comparison between PSNR score and speed in seconds for various models when performing resolution upscaling with the scale of three. The average PSNR and run-time run on one CPU core clocked at 2.0 GHz over the photographs from Set14 are displayed in these results [3]

beginning or at the top. We cannot use it at the center because it is a set operation that cannot be learned. The way ESPCNN got around was that it used sub-pixel convolutions to upscale. This method of upscaling is a learnable operation, and it led to an improvement in the results. Hence, an optimized PSNR score was achieved [3] (Fig. 11.3).

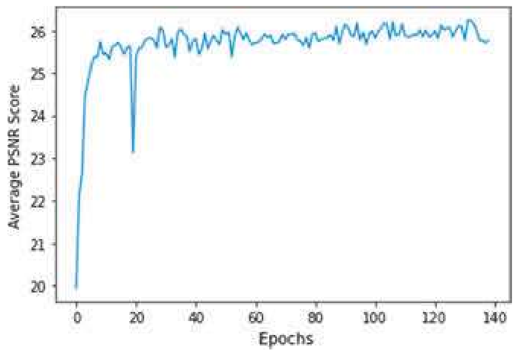
Limitation of other Super-Resolution Models. Convolutional neural network (CNN) approaches like super-resolution convolutional neural network, fast super-resolution convolutional neural network, and very deep super resolution.

- Upscaling or upsampling of the low-resolution image.
- Performing convolution in order to generate a high-resolution image.
- The convolutions are based on the low-resolution image on which the upsampling has been done, because the low-resolution image is upsampled at the initial stage.

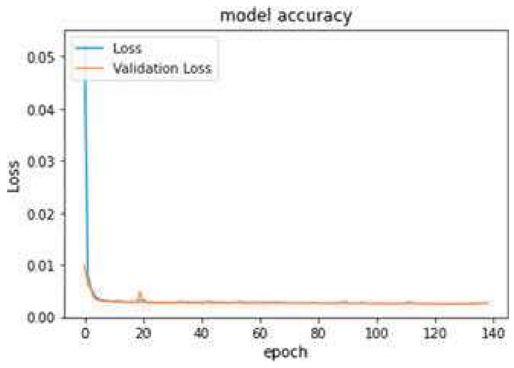
Thereby, the number of computations is increased.

11.3 Proposed Algorithm

As discussed in section II.A.5), we are focusing on how to generate a better output from the Image GPT. To do that, we generated output using IGPT-S on various datasets like MNIST, Fashion MNIST, Cifar-10, etc., and then forwarded the output to ESPCNN that improved the quality of the output generated previously. Here, we have validated the result using PSNR score and SSIM index value.



(a) Epochs VS Average PSNR Score



(b) Epochs Vs Validation Loss

Fig. 11.4 Showing prediction plots at different epochs

11.4 Our Results on IGPT-S and ESPCNN Super Resolution

The graph above shows that the average PSNR score is continuously increasing and validation loss is continuously decreasing, and we get better results as we increase the number of epochs. For this experiment, we kept considering 140 epochs, and we saw a good result based on the ESPCNN algorithm.

Table 11.1a shows the images with the distorted output when the learning rate is set to 0.006, and Table 11.1c shows a better generated image when the learning rate is reduced to 0.003.

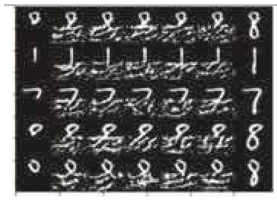
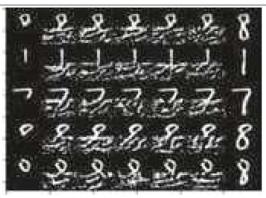
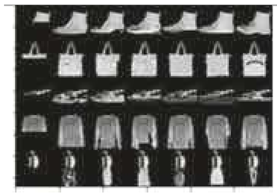
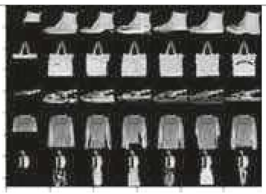
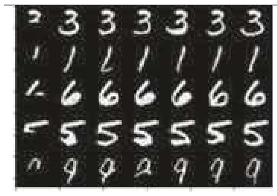
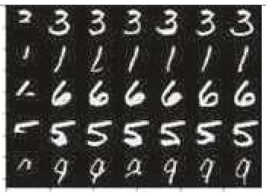
11.5 Conclusion

An auto-regressive model can do significantly well in image generation and classification as well. The Generative Pre-trained Transformer (GPT) is known for NLP, but with this research, it is clear that we can generate and classify images by using the same technique. Better generative models learn better representations. As the size of the model is increased, the accuracy improves too which implies that the quality of the model is dependent on the size of the training dataset even for such a huge amount of data.

We found that on a larger learning rate (0.006), the model generated distorted output and produced a better image on a smaller learning rate (0.003).

Our proposed model works on a large number of parameters and an enormous dataset as compared to other models. The sequence transformer used in IGPT can compute based on different features in multiple domains like text and images because of its simplicity and generality. We have demonstrated that a non-adaptive upscaling at the primary layer provides worse results compared to an adaptive upscaling for SISR and due to which it requires more computational complexity.

Table 11.1 Final results generated by the IGPT and ESPCNN

	
(a) Comparison between low-resolution and predicted image = > MSE: 872.26, SSIM: 0.48PSNR of low-resolution image and predicted image is 23.4956	
	
(b) Comparison between low-resolution and predicted image => MSE: 477.84, SSIM: 0.41PSNR of low-resolution image and predicted image is 26.1092	
	
(c) Comparison between low-resolution and predicted image => MSE: 415.55, SSIM: 0.24PSNR of low-resolution image and predicted image is 26.7158	

To address the matter, we have proposed to perform the feature extraction stages within the HR space rather than LR space. To meet this requirement, we have implemented a different sub-pixel convolution layer which can resolve the given LR data into HR space with a little to almost negligible computational cost compared to a deconvolutional layer at the time of training. Evaluation performed on an extended benchmark dataset with upscaling factor of 3 shows that the speed is significantly good approximately ten times better and performance boost compared to the previous CNN approach in which there are more parameters considered.

References

1. Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I.: Language models are unsupervised multitask learners. Accessed: 30 December 2020. [Online]. Available https://github.com/codelucas/newspaper

2. Chen, M., et al.: Image GPT. ICML (2020)
3. Shi, W., et al.: Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. In: Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition, vol. 2016-Decem, pp. 1874–1883 (2016). https://doi.org/10.1109/CVPR.2016.207
4. Donahue, J., Krähenbühl, P., Darrell, T.: Adversarial Feature Learning, pp. 1–18 (2016). [Online]. Available http://arxiv.org/abs/1605.09782
5. Huang, Y., et al.: GPipe: efficient training of giant neural networks using pipeline parallelism
6. Isola, P.: Contrastive multiview coding
7. Kolesnikov, A., Zhai, X., Beyer, L.: Revisiting self-supervised visual representation learning. arXiv (2019)
8. Vaswani, A., et al.: Attention is all you need
9. Devlin, J., Chang, M.-W., Lee, K., Google, K.T., Language, A.I.: BERT: pre-training of deep bidirectional transformers for language understanding. [Online]. Available https://github.com/tensorflow/tensor2tensor
10. Kornblith, S., Shlens, J., Le, Q.V.: Do better ImageNet models transfer better ? pp. 2661–2671
11. SR with OpenCV Homepage. https://bleedai.com/super-resolution-with-opencv/
12. NYU Center for Neural Science Homepage. https://arxiv.org/abs/2006.13846