Deep Learning for Image Inpainting*

Biradar Nikhil *210050035*
Guramrit Singh *210050061*
Omm Agrawal *210050110*
Sabyasachi Samantaray *210050138*

***Abstract*—Recent deep learning approaches to Image Inpainting have taken over the statistical methods which solely were based on searching for identical patches in the valid regions. These methods fail to produce features which cannot be found in the rest of the image, essentially positive hallucination and understanding the context of the image cannot be accounted by statistical methods. Deep Learning Approaches have shown promising results, yet they require large datasets. In this study we replicate two such architectures proposed in literature, the GLCIC and Contextual Attention based model and give a comprehensive analytical review on a smaller dataset.**[1]

I. Introduction

The task of image inpainting is to fill in missing areas of the image, where it can't get "hints" from nearby pixels. This requires a much deeper semantic understanding of the scene, and the ability to synthesize high-level features over large spatial extents. A model needs to both understand the content of an image, as well as produce a plausible hypothesis for the missing parts. Early approaches to image inpainting relied on statistical models estimate missing pixel values. However, these methods have limitations when faced with large, diverse datasets and highly structured objects.

The emergence of deep learning revolutionized image inpainting by enabling models to automatically learn hierarchical representations of data. Deep neural networks, particularly convolutional neural networks (CNNs), demonstrated exceptional capabilities in capturing intricate features and patterns from images and Generative Adversarial Networks (GANs) in learning image generation.

Throughout this paper, we will explore the evolution of image inpainting techniques, highlighting the limitations of earlier methods and emphasizing the pivotal role of deep learning in overcoming these challenges. We experiment on the Contextual Attention based model proposed by [3] and understand its effectiveness in generative inpainting.

*Authors are arranged in the order of their roll numbers.

[1]GitHub code for the project can be found here: Deep Learning for Image Inpainting

II. Related Work

A. Context Encoder

Pathak et al., [1] presents an unsupervised visual feature learning algorithm. Analogous to auto-encoder, the context encoders are convolutional neural networks trained to generate the contents of an arbitrary image region when given as input the valid regions (the surrounding known regions). Since this is a multi modal task, the model is jointly trained to minimize both reconstruction loss and an adversarial loss. The L2 loss captures the overall structure of the missing region in relation to context, while the adversarial loss has the effect of picking a particular mode from the distribution. Adding the adversarial loss reportedly produced sharper results than just the L2 loss. The equations below summarise the losses used, $x \in \mathbb{X}$ are the ground truths and $z \in \mathbb{Z}$ are the masked images, $z = (1 - \hat{M}) \odot x$. Just like a GAN framework, the optimisation objective is a minimax expression, the discriminator tries its best to correctly identify the images generated via the generator, whereas the generator tries its best to fool the discriminator.

$$\mathbb{L}_{rec}(x) = ||\hat{M} \odot (x - F((1 - \hat{M}) \odot x))||_2^2$$

$$\mathbb{L}_{adv}(x) = max_D(\mathbb{E}_x[log(D(x))] + \mathbb{E}_z[log(1 - D(G(z)))]$$

$$\mathbb{L} = \lambda_{rec}\mathbb{L}_{rec} + \lambda_{adv}\mathbb{L}_{adv}$$

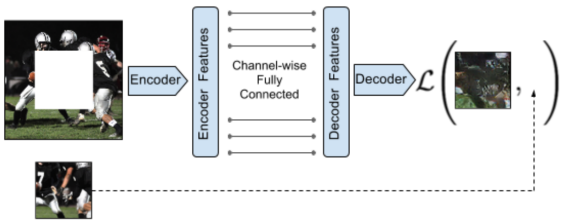

Fig. 1. Overview of Context Encoder. Image with hole is passed through the encoder which is connected to the decoder using channel-wise fully connected layer. The decoder produces the inpainted image.

Figure 1 shows the model architecture, the intermediate latent feature representation is required as there

is no other way for information to directly propagate from one corner of feature map to another. This is because convolutional layers connect all feature maps together, but never directly connect all locations within a specific feature map. However, fully connecting the encoder with decoder would result in an explosion of number of parameters, and the paper alleviates this by using per channel fully connected layers.

B. Globally and Locally Consistent Context Encoder

Iizuka et al., [2] improves over the Context Encoder in two aspects. (1) Usin Dilation in CNNs instead of fully connected layers, to reduce the number of parameters while maintaining the spread of the receptive area. (2) Multi-Scale discriminators, for global and local consistency. Model architecture is shown in Figure 2. During training, there is always one single missing region. During testing, there could be multiple missing regions in an image. The optimisation objective is as below.

$$min_C max_D \mathbb{E}[L(x, M_c) + \alpha log(D(x, M_d))$$

$$+\alpha log(1 - D(C(x, M_c), M_c))]$$

where the $L(x, M_c)$ is the MSE loss, $C(x, M_c)$ is the completion network. M_d is a random mask, M_c is the input mask, α is a hyperparameter.

$$L(x, M_c) = ||M_c \odot (C(x, M_c) - x)||^2$$

III. Models

A. Baselines

1) Statistical Methods: As baseline comparison models, we use the following two statistical models. Both these methods are available in the OpenCV library.

- NS (Navier Stokes) - This algorithm is directly inspired from the Navier Stokes' partial differential equation for fluid dynamics. Starting from the edges (known regions) towards the unknown regions, it propagates isophote lines (lines that join same-intensity points) [4].
- FMM(Fast Marching Method) - This is based on Level Set method. It iteratively propagates information from known parts of image to unknown parts minimizing the difference [5].

2) Autoencoders: We implemented standard Autoencoder model to learn the inpainting of the masked image. The masked image is given as training input and the latent space is made to learn the compact feature representation of the image. Through this feature representation, the decoder is made to produce filled image by training it on reconstruction (L2) loss against ground truth image.

The encoder uses 8 convolutional blocks and 4 pool blocks while the decoder uses 4 transpose convolution blocks.

B. Stable Variant of GLCIC

Jiahui et al., [3] propose a modification over GLCIC. They replace the completion network with two subnetworks to further enlarge the receptive fields and stabilise learning - Coarse Network and Refinement Network (inspired from residual learning), which outperforms GLCIC in terms of both results and time (8x speedup). The coarse network is trained on L1 Reconstruction Loss to make an initial coarse prediction, and the second network is trained with L1 as well as GAN losses to predict refined image (Refer Figure 3). There are a bunch of subtle yet important modifications over GLCIC, these are listed below.

- Removes batch normalisation, which was reported to deteriorate color coherence.
- Uses ELU instead of ReLU as activation function, and clipping the output filter values instead of using tanh or sigmoid function.
- Uses WGANs instead of DCGANs, as the losses coming from WGANs measure pixel-wise L1 distances, the combined loss with L1 reconstruction loss is easier to train and makes the optimisation process stabler.
- Separates global and local feature representations for GAN training, reported that this works better than feature concatenation as in GLCIC.
- L1 loss is spatially discounted, that is the weight of each pixel in the mask is calculated as γ^l where l is the distance of missing pixel to the nearest valid pixel. Intuitively, missing pixels near the hole boundaries have much lesser ambiguity than those closer to the hole, and in such cases it induces stability to do a discounted loss, it also helps to not regress the holes so strongly with the ground truth image, as there may be multiple correct ways to inpaint the missing region.

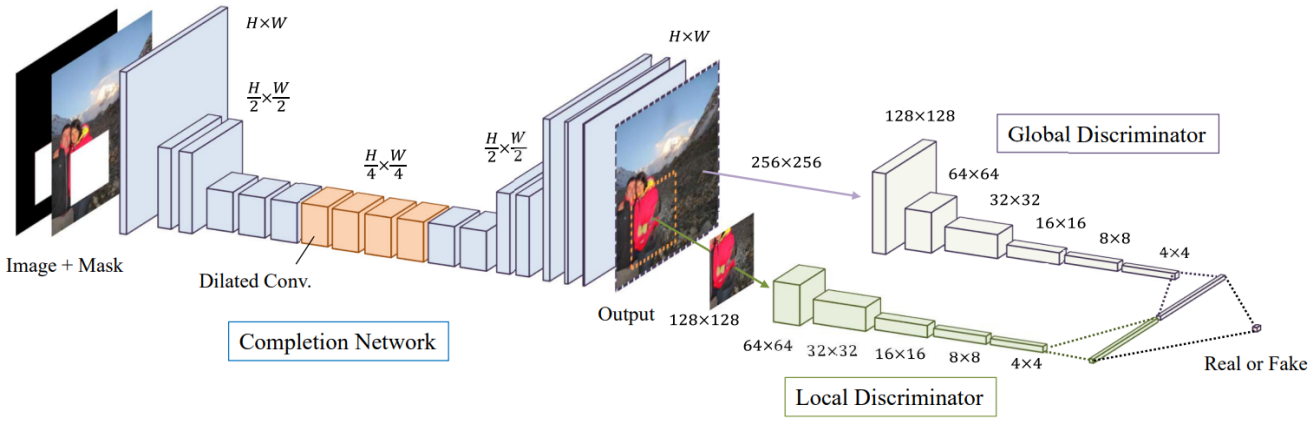

Fig. 2. Overview of GLCIC Model Architecture. The completion network generates the inpainted complete image, which is subjected to two auxiliary context discriminator networks.

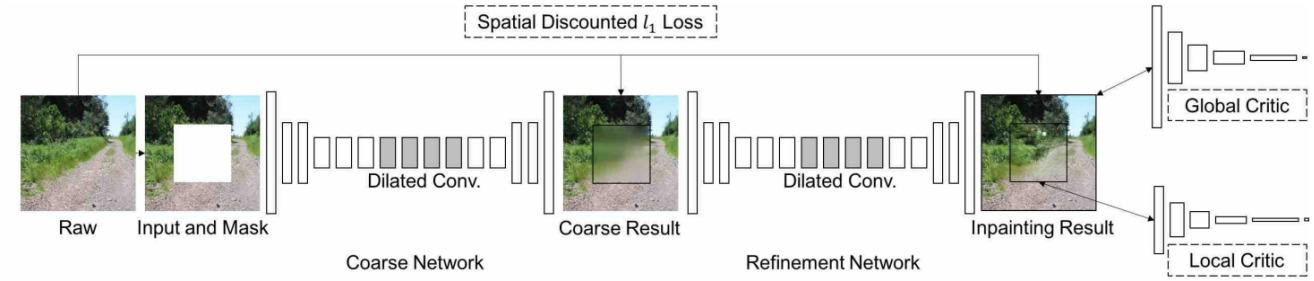

Fig. 3. Overview of the Improved Stable Variant of GLCIC, as proposed by Jiahui et al., [3]

C. Contextual Attention (CA)

In addition to the modifications proposed in the previous model, Jiahui et al., [3] wanted to do the best of two worlds - Synthetic methods and Deep Learning based methods - searching for similar patches in valid regions and hallucination of new content. They propose a contextual attention layer (Figure 4) which learns where to borrow or copy feature information from the background valid patches to generate missing patches. The core idea of CA is to use features of valid patches as convolutional filters to process the generated patches. Channel-wise softmax is used to weight relevant patches and deconvolution to reconstruct the generated patches with contextual patches. There is another spatial propagation layer to encourage spatial coherency of attention.

In order to also allow the network to hallucinate novel content, CA layer is added in a parallel pathway of Refinement network as shown in Figure 5. The CA is implemented in two steps.

1) **Match and Attend** - similarity ($s_{x,y,x',y'}$) between a foreground path $f_{x,y}$ with a background patch centred at $b_{x',y'}$ is measured as normalized inner product (cosine). To get the attention scores, we take softmax along (x', y') dimension. To reconstruct foreground, extracted patches are used as deconvolution filters.

$$s_{x,y,x',y'} = \langle \frac{f_{x,y}}{||f_{x,y}||}, \frac{b_{x',y'}}{||b_{x',y'}||} \rangle$$

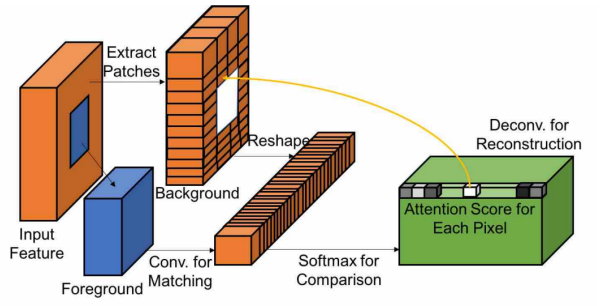

Fig. 4. Improved Generative inpainting framework proposed by Jiahui et al.,

$$s^*_{x,y,x',y'} = softmax_{x',y'}(\lambda s_{x,y,x',y'})$$

2) **Attention Propagation** - To ensure attention coherency, a left-right propagation followed by a top-down propagation with kernel size of k is performed to get the new attention score. This enriches gradients in training.

$$s'_{x,y,x',y'} = \sum_{i\in\{-k,\cdots,k\}} s^*_{x+i,y,x'+i,y'}$$

$$\hat{s}_{x,y,x',y'} = \sum_{j\in\{-k,\cdots,k\}} s'_{x,y+j,x',y'+j}$$

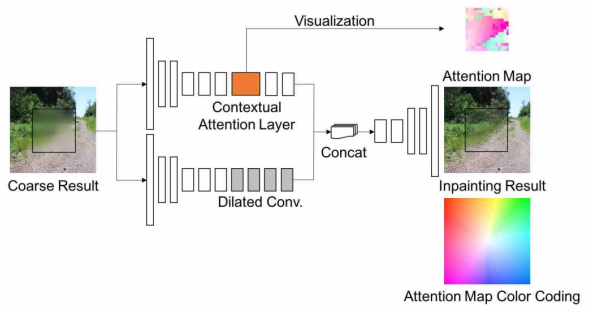

Fig. 5. Improved Generative inpainting framework using Contextual Attention block in a parallel pathway in refinement network. To visualise attention map, color indicates relative location of the most attended background patch for every foreground pixel.

IV. Experiments

A. Datasets

1) Animals Dataset: This dataset [6] consists of 90 different animals, with 60 images in each class. We perform a split of 1:11 (test : training) for each class to keep the test set unbiased. We resize each image to (256x256x3) before feeding it into the training model.

2) ImageNet: We use a subset of 50K diverse images from a pool of 1.3M training images publicly available at [7].

B. Model training

We train the following models:

- Autoencoder - Trained on the Animals dataset with Hole mask type for 100K iterations
- Stable GLCIC - Trained on Animals dataset with Hole mask type for 50K iterations
- Contextual Attention - We perform ablation study on this model
  - On Animals dataset with Hole mask type for 150K iterations
  - On ImageNet dataset with Hole mask type for 300K iterations
  - Above model further finetuned on Animal dataset for 70K iterations
  - On Animals dataset with Mosaic type mask for 50K iterations

C. Baselines

We set three baselines, two statistical techniques of NS (Navier-Stokes) and TELEA (Fast Marching Method). We also build a simple Autoencoder to generate inpainted images. We input images of size (256x256x3) with a central squared mask of size (128x128).

D. Deep Learning Architectures

We experiment on both the GLCIC and CA models and perform an ablation study with different experimental settings. Input image is generated by masking the original image, we implement this in 3 ways, hole — mosaic — hole_ns. Hole puts a random squared mask of size roughly (128x128), with minor variations, mosaic enforces a mosaic effect on an image using a downsample-upsample approach with bilinear interpolation (with a scale factor depending inversely on mosaic unit size) and a mask to control the blending of the images, hole_ns takes the input image with mask, inpaints it with the NS method and gives that as input to the generator. We use batch size of 64, one iteration refers to processing one batch. The values of hyperparameters used are listed in Table I.

TABLE I
HYPERPARAMETERS USED FOR THE MODELS

spatial discounting factor(γ)	0.9	mosaic_unit_size	12
learning rate	0.0001	num_critics	5
β_1	0.5	β_2	0.9
coarse_l1_alpha	1.2	l1_loss_alpha	1.2
ae_loss_alpha	1.2	global_wgan_loss_alpha	1
gan_loss_alpha	0.001	wgan_gp_lambda	10
fuse_k	3		

V. Results

A. Metrics

Image inpainting lacks good quantitative evaluation metrics, as reported by the paper [3]. Reconstruction errors are not perfect as there may be multiple correct solutions for the missing regions, neither are inception score based metrics which are used for GAN tasks, as main purpose of image inpainting was to fill background

or missing regions to the best context and quality and not just mere generation of classes of objects. Nevertheless we use MSE, L1 loss and PSNR (Peak Signal to Noise Ratio) to measure reconstruction error. Additionally we use Learned Perceptual Image Patch Similarity (LPIPS) and Total Variation (TV) Loss. LPIPS is a measure of perceptual similarity based on learned features from patches of the image. Lower scores indicate more perceptual similarity. The TV loss encourages spatial smoothness in generated images.

B. Comparison of Model's Performance

The non decreasing validation loss despite a decreasing training loss is observed for the autoencoder in Figure 6, coupled with color artifacts in the generated inpainted results, suggests potential issues with overfitting to the training data and an inability to generalize. The fixed central (128x128) hole in the mask and the MSE loss might be causing the model to memorize the training set rather than learning meaningful features for inpainting.

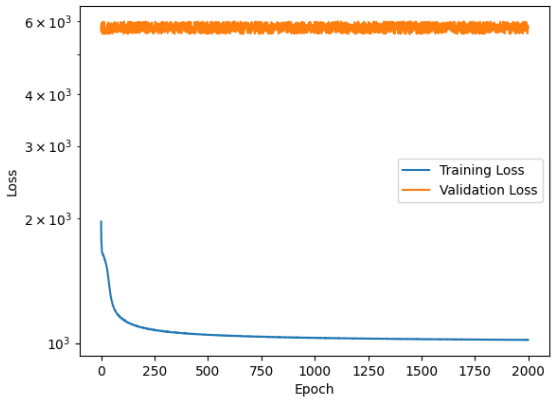

Fig. 6. Variation of Training and Validation log-scaled MSE Losses for the simple AutoEncoder architecture

In Figure 7, we observe the training progress of four models: Stable GLCIC, CA with hole, CA with mosaic and CA with hole_ns. As anticipated, the mosaic model outperforms the others. This outcome aligns with expectations since the mosaic approach introduces a subtle blurry effect while preserving essential ultra coarse details of the masked region. Although strictly speaking, this method may not fall precisely within the realm of image inpainting, it can be categorized as image deblurring. We also observe CA performs better than GLCIC, which can be attributed to the addition of attention layer.

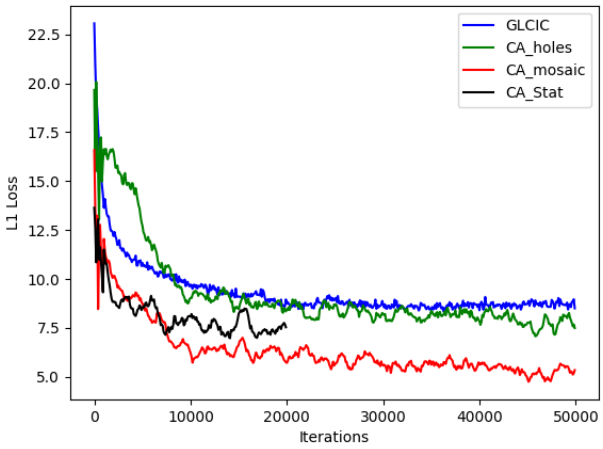

Fig. 7. Progression of Training Losses for 3 variants of models on Animals dataset. 1 iteration corresponds to processing of 1 batch = 64 images

Results of CA trained on ImageNet for 300K iterations produce promising results (Figure 8) even on Places2 dataset (which is completely unknown to the model) [8], although we still find arbitrary border and mask artifacts. We decided to fine tune this model on the animals dataset for 70K iterations. The comparative results for all models can be seen in the Figure 9.

We also witness the potential of this model to remove certain objects from the image by matching with the background. Results displayed in Figure 10. We compile the performance comparison amongs all our models in table II. Despite all this success, our model still suffers with inability to hallucinate intricately structured objects like face and body (Refer Figure 11)

VI. Conclusion

While the effectiveness of Deep Learning-based models, such as our CA and GLCIC models, in image inpainting is evident through their superior performance over traditional statistical models, it is crucial to address the significant factor of training time. Our models were trained at a rate of approximately 1 hour for every 2000 iterations on a computational setup comprising 5 GPUs. Reducing training time is a critical concern in deep learning research, prompting dedicated efforts to enhance efficiency without compromising performance. One noteworthy direction in this pursuit involves exploring alternative approaches, such as wavelet transform-based methods [9]. These techniques have shown promise in significantly lowering training times while maintaining comparable inpainting performance.

TABLE II
COMPARISON OF SCORES FOR IMAGE INPAINTING TASKS

Model Variant	Scores				
	MSE	***L1***	***PSNR***	***LPIPS***	***TV***
TELEA Baseline	821.73	9.66	19.61	0.17	7.97
NS Baseline	810.47	9.68	19.63	0.16	8.15
Autoencoder(100K)	1472	9.34	**26.25**	0.18	2.18
GLCIC(50K)	720.49	9.25	20.16	0.13	1.70
CA-Animals-Hole(150K)	631.22	8.09	20.99	0.11	1.75
CA-ImageNet-Hole(300K)	565.84	8.00	21.40	0.10	**1.43**
CA-ImageNet-FT-Animals(70K)	**540.59**	**7.65**	21.61	**0.09**	2.38
CA-Animals-Mosaic(50K)	**270.44**	**5.31**	**24.68**	**0.06**	**1.25**

Fig. 8. Results on Places2 dataset using CA trained on ImageNet. (A) shows the masked input image, (B) contains the inpainted image and (C) shows the corresponding attention map

In critique of CA based method, we notice that CA with mosaic learnt faster, it learnt at 50K iterations, what CA with holes learnt after 150K. But this is not really inpainting, in some sense it is cheating. We wish to do the best of both worlds. Given an input hole masked image, we first generate the inpainted image using NS method, and then pass it through the CA model. We trained it for 20K iterations, the training progression in comparison to other models can be seen in figure 7. We observe that there is scope of improvement if trained for higher number of iterations.

Another observation in the CA block is that the attention vectors are calculated for every missing pixel with every background patch. This calculation of attention scores can be computationally intensive. Instead of computing attention scores for every pixel, we may use striding or subsampling to reduce the number of pixels involved in the computation. This can significantly speed up the process at the cost of some loss of spatial resolution which can be regained over multiple iterations, if subsampling is done randomly.

It might be possible to use approximations or heuristics to reduce the number of computations. For example, focusing on regions of interest or dynamically adjusting the level of detail in different parts of the image. One may first segment the background patches into similar regions. We can now represent every background patch cluster with a mean feature vector (segmentation using MRF to ensure spatial smoothness alongside spectral smoothness in segmentation). So effectively, we have reduced the number of background patches to be attended on for a pixel.

One may also divide the image into hierarchical regions and process attention scores hierarchically. This can be beneficial in cases where the importance of distant pixels is lower.

Fig. 9. Comparison of quality and correctness of inpainted images for all models on randomly chosen images from ImageNet Test Set

Fig. 10. Application to Object Removal tasks. We use the CA trained on ImageNet for this purpose. The images are randomly choosen for places2 and imagenet dataset.

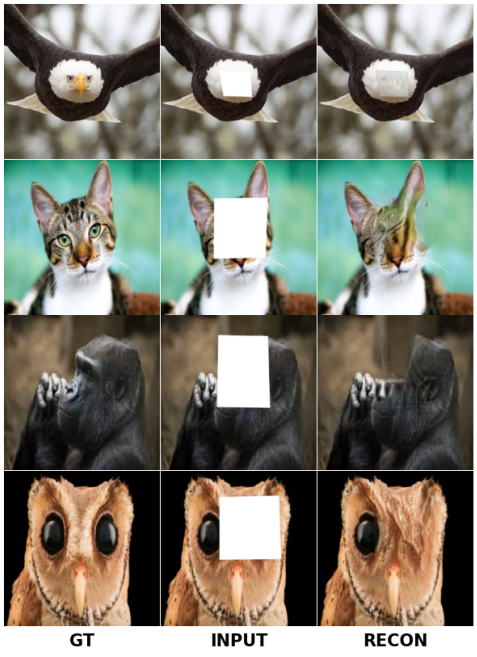

Fig. 11. Model still cannot hallucinate highly structured objects like face and body.

REFERENCES

[1] Pathak. Deepak , Krahenbuhl. Philipp , Donahue. Jeff , Darrell. Trevor and Efros. Alexei. (2016). Context Encoders: Feature Learning by Inpainting. 2536-2544. 10.1109/CVPR.2016.278.

[2] Iizuka. Satoshi , Simo-Serra, Edgar and Ishikawa. Hiroshi. (2017). Globally and locally consistent image completion. ACM Transactions on Graphics. 36. 1-14. 10.1145/3072959.3073659.

[3] Yu. Jiahui , Lin. Zhe, Yang. Jimei , Shen. Xiaohui and‘ Lu. Xin. (2018). Generative Image Inpainting with Contextual Attention. 5505-5514. 10.1109/CVPR.2018.00577.

[4] M. Bertalmio, A. L. Bertozzi and G. Sapiro, ”Navier-stokes, fluid dynamics, and image and video inpainting,” Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001, Kauai, HI, USA, 2001, pp. I-I, doi: 10.1109/CVPR.2001.990497.

[5] Telea, Alexandru. (2004). An Image Inpainting Technique Based on the Fast Marching Method. Journal of Graphics Tools. 9. 10.1080/10867651.2004.10487596.

[6] Banerjee, Sourav. (2022). ”Animal Image Dataset, 90 Different Animals, Kaggle”. https://www.kaggle.com/datasets/iamsouravbanerjee/animal-image-dataset-90-different-animals.

[7] Imagenet, Research Prediction Competition. (2018). ”ImageNet Object Localization Challenge, Kaggle”. https://www.kaggle.com/c/imagenet-object-localization-challenge/overview/description.

[8] Nikhil Joson. (2021). ”Places-2_MIT_Dataset, Kaggle”. https://www.kaggle.com/datasets/nickj26/places2-mit-dataset/.

[9] Yu. Yingchen, Zhan. Fangneng, Lu. Shijian, Pan. Jianxiong, Ma. Feiying, Xie. Xuansong and Miao. Chunyan. (2021). WaveFill: A Wavelet-based Generation Network for Image Inpainting. 14094-14103. 10.1109/ICCV48922.2021.01385.