Ukrainian Catholic University

Faculty of Applied Sciences

Business Analytics Program

Image Inpainting

MMML Final Project Report

Authors:
Roman Bernikov
Bohdan Pelekh
Nazar Kononenko

21 May 2024

Abstract

In our project, we aim to explore inpainting methodologies and develop a robust and efficient inpainting model capable of producing high-quality reconstructions across various applications. We will utilize cutting-edge machine learning techniques, such as convolutional neural networks (CNNs) and generative adversarial networks (GANs), to create effective approach of the image inpainting method.

1 Introduction

Inpainting, an essential task in image processing and computer vision, refers to the process of reconstructing missing or damaged regions within an image in a visually plausible manner. The goal of inpainting algorithms is to fill in the gaps seamlessly, ensuring that the resulting image appears coherent and natural to human observers.

There are three main categories of image inpainting methods: sequential-based approaches, CNN-based approaches, and GAN-based approaches.

Sequential-based methods typically fill in missing regions of an image by iteratively completing small patches, searching for well-matching replacement patches in the undamaged part of the image. These methods often rely on texture synthesis techniques and can produce good results for stationary textures, but may struggle with complex structures or semantically meaningful regions.

CNN-based approaches utilize convolutional neural networks to learn and infer missing pixel values. These methods can capture and leverage high-level semantic information from the input image, allowing for more coherent and semantically plausible completions. However, they may still produce blurry or inconsistent results, especially for larger missing regions or highly complex scenes.

In this work, we focus on a particular class of methods known as generative adversarial networks (GANs). GANs have shown remarkable success in generating highly realistic and coherent image completions, making them a promising solution for the challenging task of image inpainting.[3]

2 Method Discussion

2.1 Approach

Our approach for image inpainting is inspired by the work presented in the paper "Globally and locally consistent image completion."[5]. We implement a deep learning model that leverages the principles of Generative Adversarial Networks (GANs) to achieve realistic and coherent image completion.

The core component of our approach is a fully convolutional completion network, following encoder-decoder structure with use of dilated convolutions, responsible for generating the completed regions of the input images. Inspired by the paper, we incorporate two additional networks: a global context discriminator and a local context discriminator. These discriminator networks play a crucial role in training the completion network to produce natural and consistent image completions.

2.2 Completion Network

The completion network, which acts as the generator in our GAN-based approach, follows an encoder-decoder architecture, a widely adopted structure for image-to-image translation tasks. The encoder part of the network is responsible for extracting high-level semantic features from the input image, while the decoder part generates the completed image by combining these features with the known regions of the input. One key aspect of our CNN architecture is the incorporation of dilated convolutions, a technique that has proven effective in various computer vision tasks, including image segmentation and inpainting.

Dilated convolution is a technique used in convolutional neural networks (CNNs) to increase the receptive field of the convolution operation without increasing the number of parameters or the amount of computation required.

In a standard convolution operation, the filter (kernel) is applied to the input feature map by sliding it over the input and computing the dot product between the filter weights and the corresponding input values. The stride determines the step size at which the filter is moved across the input. In dilated convolution, the filter is sparsely sampled, allowing it to cover a larger receptive field without increasing the number of parameters.

The dilation rate, denoted as r, determines the spacing between the filter weights. For a dilation rate of $r > 1$, the filter weights are sparsely sampled, with gaps of $r - 1$ values between each weight.

The dilated convolution operation can be expressed as:

$$y_{i,j} = \sum_{k,l} x_{i+r \cdot k, j+r \cdot l} \cdot w_{k,l}$$

where $y_{i,j}$ is the output feature map, $x_{i,j}$ is the input feature map, $w_{k,l}$ are the filter weights, and r is the dilation rate.

The receptive field of a dilated convolution operation with a dilation rate of r and a filter size of $(2k + 1) \times (2k + 1)$ is $(2rk + 1) \times (2rk + 1)$. This allows the filter to capture long-range spatial information while using fewer parameters and computations compared to increasing the filter size in a standard convolution operation.

2.3 GAN

GAN - Generative Adversarial Network. The main idea is 2 neural networks that compete with each other.

The generator network takes random noise or a random vector as input and tries to generate data that resembles real data.

The discriminator network, on the other hand, tries to distinguish between real data and the data generated by the generator. It takes both real data from the dataset and fake data from the generator as input and classifies them as real or fake.

Generator network G that learns distribution p_g from training data via the application of a prior distribution on input noise variables, represented as $p_z(z)$. G operates by mapping the noise variable z to the data space through a parametric function $G(z; \theta_g)$. Minimizing the difference between the generated distribution and the actual distribution of the training data involves iteratively optimizing the parameter θ_g. The discriminator network $D(x; \theta_d)$ returns a value that represents the probability of x belonging to the training data.[4]

As a result, we have a min-max optimization problem where the Generator and Discriminator networks at each iteration jointly update their parameters.

$$\min_G \max_D V(D,G) = Ex \sim p_{data}(x)[\log D(x)] + E_{z \sim p_z(z)}[\log(1 - D(G(z)))]$$

The discriminator tries to maximize the probability of correctly distinguishing real data from fake data generated by the generator, while the generator tries to minimize the ability of the discriminator to differentiate its generated fake data from real data.

One of the main issues of GAN is the instability during learning, which has led to numerous research on the topic. The training of Generative Adversarial Networks faces several challenges, including mode collapse, where the model struggles to generate diverse samples even when trained on multi-modal data. Oscillation and instability during training can occur, preventing convergence to a fixed point and leading to vanishing gradients, which hinder effective learning. Additionally, gradient estimation becomes difficult, especially when the discriminator easily distinguishes between real and fake samples, resulting in small gradients and halting updates to the generator.[2]

Training Generative Adversarial Networks also could be described as finding a Nash equilibrium in a non-convex game between the generator and the discriminator. Finding this Nash equilibrium presents a significant challenge due to the following reasons:

- Simultaneous Optimization: The generator and the discriminator are optimized simultaneously in a min-max game, where the generator aims to minimize the discriminator's ability to distinguish between real and fake data, while the discriminator tries to maximize its ability to correctly classify real and fake data. In the training process Applying gradient descent simultaneously to minimize $J(G)$ and $J(D)$ can fail to converge, as minimizing one player's cost may increase the other's cost, leading to stable orbits instead of the desired equilibrium. For instance, when one player minimizes xy w.r.t. x and the other minimizes $-xy$ w.r.t. y, gradient descent fails to converge to $x = y = 0$. [7]

- Non-Convex Optimization: The objective functions of both the generator and the discriminator are non-convex, which means that there can be multiple local minima or maxima in the optimization landscape. This makes it difficult to guarantee convergence to the global optimum, which corresponds to the Nash equilibrium.

To address problems with learning instability, the authors propose an improvement [5] to the GAN architecture. The proposed architecture incorporates two discriminators: a global context discriminator and a local context discriminator. The inclusion of these two discriminators is essential for achieving both globally and locally consistent image completion results.

Global Context Discriminator:

- The global context discriminator takes the entire completed image as input.

- Its purpose is to assess the overall consistency and coherence of the completed image with respect to the original scene.

- By considering the full image, the global discriminator ensures that the completed region seamlessly blends with the surrounding context and maintains the global structure and semantics of the image.

Local Context Discriminator:

- The local context discriminator focuses on a small region surrounding the completed area.
- It evaluates the quality and appearance of the completed region at a more detailed level.
- The local discriminator helps in capturing fine-grained details and ensuring that the completed area exhibits natural and realistic textures and patterns that are consistent with the local neighborhood.

The combination of global and local context discriminators in the GAN framework enables the model to capture both high-level semantics and low-level details. The global discriminator ensures that the completed image maintains overall consistency with the original scene, while the local discriminator focuses on the finer aspects of the completed region. This dual-discriminator approach helps in generating globally coherent and locally realistic image completions.

2.4 Data preprocessing

Together, our dataset consists of 10000 examples. There are 1000 different classes of images. To create our dataset for training the model, we perform the following steps on each image:

- Randomly generate a mask of the shape 50x50 pixels.
- Fill each pixel of the mask with a **0.5** value (authors take mean value through the dataset to fill their mask with, however, our approach is less time-consuming and receives the same result).
- Apply the mask to our image.
- Finally, we stack our masked image with the mask, as our model expects the mask as additional information.

Also, our model supports training with many masks for one image.

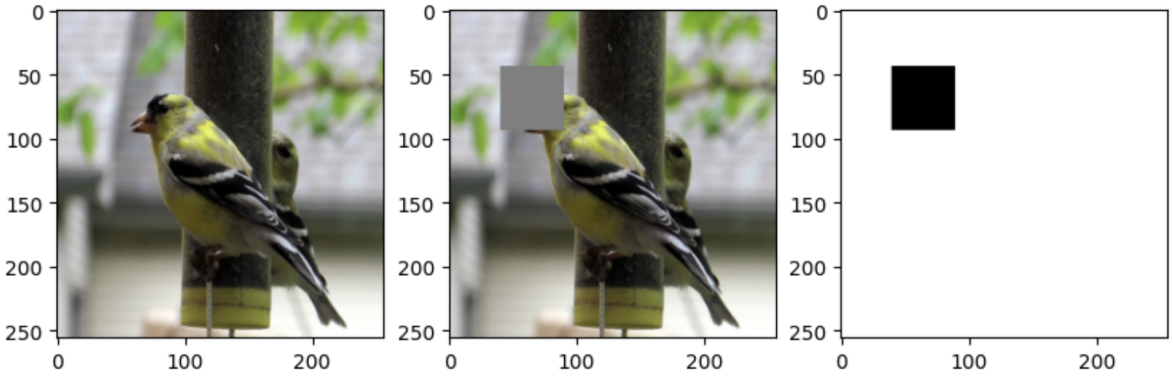

Figure 1: Example of image after preprocessing (original image, masked image, mask)

2.5 Training

To train the network for realistic image completion, two loss functions are employed jointly: a weighted Mean Squared Error (MSE) loss for training stability, and a Generative Adversarial Network (GAN) loss to enhance the realism of the results. Utilizing this mixture of loss functions allows for stable training of the high-performance network model and has been utilized for image completion, as well as concurrently for various image-to-image translation problems. The training is performed using backpropagation. For stabilizing the training process, a weighted MSE loss that accounts for the completion region mask is utilized. The MSE loss is defined as:

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

where $\odot$ denotes pixelwise multiplication, and $\|\cdot\|$ represents the Euclidean norm. The context discriminator networks also act as a type of penalty, commonly known as the GAN loss. This element turns the typical neural network optimization into a minimax optimization challenge. We divide our training process into three steps. Firstly, we train only the generator for a few epochs, after that we continue with training of the discriminator, finally, we train both of them in an adversarial manner simultaneously updating both the generator and the discriminator [5]. This approach is important as we need to ensure that the discriminator would not perceive all our images as fake, leading to a poor training process for the generator. In terms of optimization for our completion and context discriminator networks, the process transforms into:

$$\min_C \max_D E \left[\log D(x, M_d) + \log(1 - D(C(x, M_c), M_c))\right] ,$$

- M_d - random mask
- M_c - input mask
- E - the average over the training images x.

2.6 Incorporation of DAM blocks for Enhanced Inpainting

DAM-GAN paper[1] leverages an innovative approach of Dynamic Attention Maps (DAM) to significantly refine the inpainting process. This method particularly targets the reduction of fake textures - a common issue in generated images that results in pixel artifacts or color inconsistencies.

2.6.1 Understanding Dynamic Attention Maps (DAM)

Dynamic Attention Maps (DAM) serve as a sophisticated mechanism designed to identify and mitigate fake textures within inpainted images. These maps are dynamically generated during the image reconstruction phase and operate by highlighting regions within the feature maps that are likely to contribute to unrealistic image textures. The DAM's core function is to direct the generator's focus towards these problematic areas, enabling a more targeted and effective correction.

2.6.2 Mechanism of the DAM Block

The DAM block is integrated into the decoding layers of the generator, functioning through several key steps:

- **Feature Map Enhancement:** Initially, the DAM block receives concatenated feature maps from previous layers. These feature maps undergo a transformation via a 1x1 convolutional filter, producing an enhanced feature map, F_i, tailored for fakeness detection.
- **Fakeness Map Generation:** Utilizing F_i, the block applies another set of 1x1 convolutional filters followed by a sigmoid activation to generate a fakeness map, M_i. This map effectively signifies the probability of each pixel being artificially generated or fake.
- **Attention Application:** The generated fakeness map, M_i, is then used to modulate the original feature map through element-wise multiplication, resulting in an adjusted feature map that emphasizes areas of potential fake texture.
- **Feature Map Merging:** Finally, the original and adjusted feature maps are combined through element-wise summation to produce the final output for the layer, which is then forwarded to the next layer in the network.

$$M_i = \sigma(\text{Conv}_{1\times1}(F_i)),$$

$$F_i' = M_i \odot F_i,$$

$$T_{i-1} = F_i \oplus F_i',$$

where σ represents the sigmoid function, signaling the transition of the convolutional output into a probability distribution indicative of fakeness within the feature map. The operations $\odot$ and $\oplus$ denote element-wise multiplication and addition, respectively.

2.6.3 Enhancement through DAM

By integrating the DAM blocks, we can make our inpainting model attain a notable improvement in addressing the challenge of fake textures in inpainted images. Through this dynamic attention mechanism, the model ensures that the reconstructed regions are not only visually plausible but also maintain consistency with the surrounding textures, thereby significantly enhancing the quality and realism of the inpainted output.

3 Application Domains

3.1 U-Net

In our work, we also discovered another architecture approach - U-Net. U-Net is a convolutional neural network primarily used for biomedical image segmentation, designed with a U-shaped architecture that includes an encoder-decoder structure. The encoder path consists of convolutional layers to capture context, while the decoder path uses transposed convolutions to enable precise localization, allowing the model to generate high-resolution segmentations from lower-resolution input images. This architecture is particularly effective for tasks where precise boundary detection and spatial information are crucial.[6]

We tried the U-Net architecture on the completion network and compared the results with those of the architecture described in our target paper. You can see the comparison

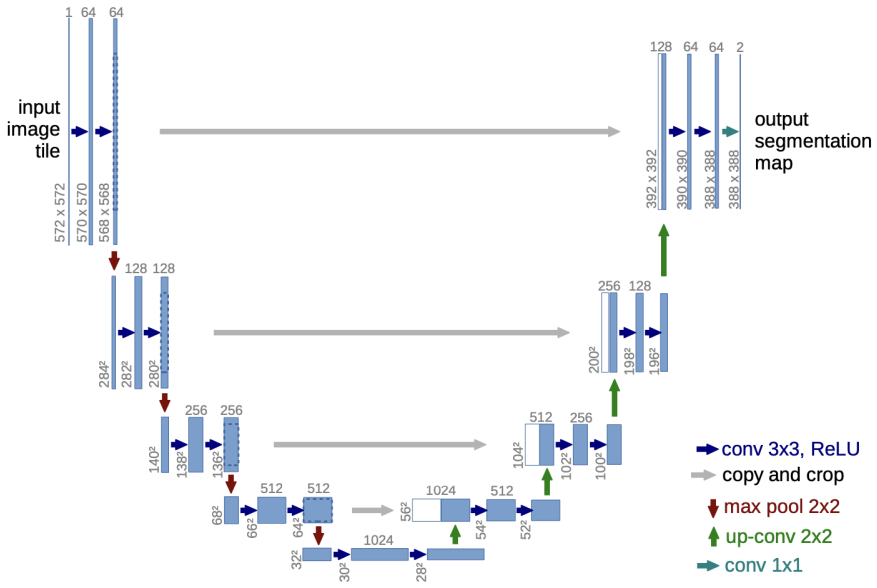

Figure 2: U-net architecture (example for 32x32 pixels in the lowest resolution)

in the corresponding section. Briefly, we can say that the model learned to replicate the unmasked region of the input image almost 1-to-1, outperforming the base architecture's results. However, when it comes to filling the mask, the performance is essentially inferior to that of the proposed architecture.

3.2 DAM

Enhancing generation by integrating DAM blocks into the model pipeline could have made our results more realistic and improved the consistency of reconstructed regions with the surroundings. However, due to the lack of information on the implementation of DAM blocks and their configuration from the authors, we were not able to integrate them into our pipeline to test their performance.

4 Results

4.1 Masked MSE vs global MSE

As we described in the Training subsection, authors use the mean squared error (MSE) loss over the masked region. This approach makes sense because we are interested only in completing the masked region. The model's capability of preserving information of the whole image is not valuable for our task since we can take the existing parts from the input image.

However, during our training experiments, we discovered that training with the masked MSE led to poor results. The problem is that during the stage of training both the generator and discriminator, the model's performance reduces significantly. The discriminator model consists of local and global parts, and the global part of the discriminator can see the whole image. Since we train our generator only with the masked MSE loss and we don't penalize the model for reconstructing the whole image, the gradients from the adversarial loss seriously degrade our model. We observe that most of the times we run the training, the reconstructed images turn out to be a complete mess.

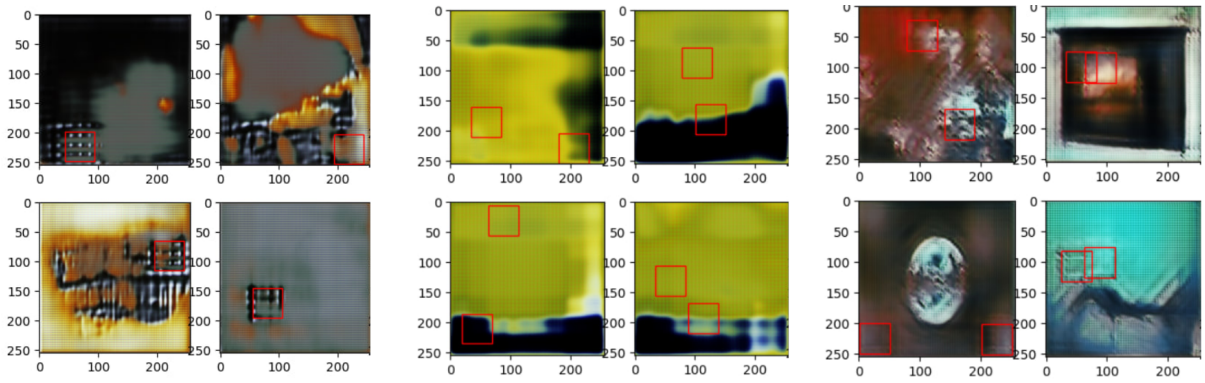

Figure 3: Results of different runs using masked MSE (4 epoch of only generator training, 1 epoch of discriminator only training, 10 epochs of combined training). Images are taken from validation dataset.

Given this problem, we replaced the masked MSE with the global MSE on the whole image. This fixed the problem with training instability.

Figure 4: Results of different runs using global MSE (4 epoch of only generator training, 1 epoch of discriminator only training, 10 epochs of combined training). Images are taken from validation dataset.

4.2 Impact of number of masks

Additionally, we studied the impact of the number of masks on model performance. We tried training the model with two and three masks additionally. As a result, we discovered that the results were similar to the setup with one mask per image. You can see examples both on Figure 4. and Figure 5.

Amount of masks	Average MSE on masked regions
1	0.000416
2	0.000424
3	0.000429

4.3 U-Net architecture results

As discussed earlier, the U-Net architecture generates an almost identical part of the unmasked image. However, when focusing on the region that interests us the most—the

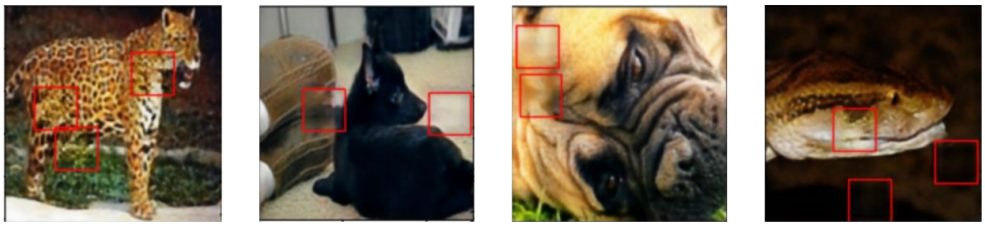

Figure 5: Results of different runs with different number of masks (4 epoch of only generator training, 1 epoch of discriminator only training, 10 epochs of combined training). Images are taken from validation dataset.

masked region—we observe a noticeable difference from the surrounding area. Specifically, the generated region is poorly reconstructed and appears blurry. Any restoration of the destroyed information is basic and very general. The limitations of this approach become especially apparent when compared with the architecture proposed by the authors.

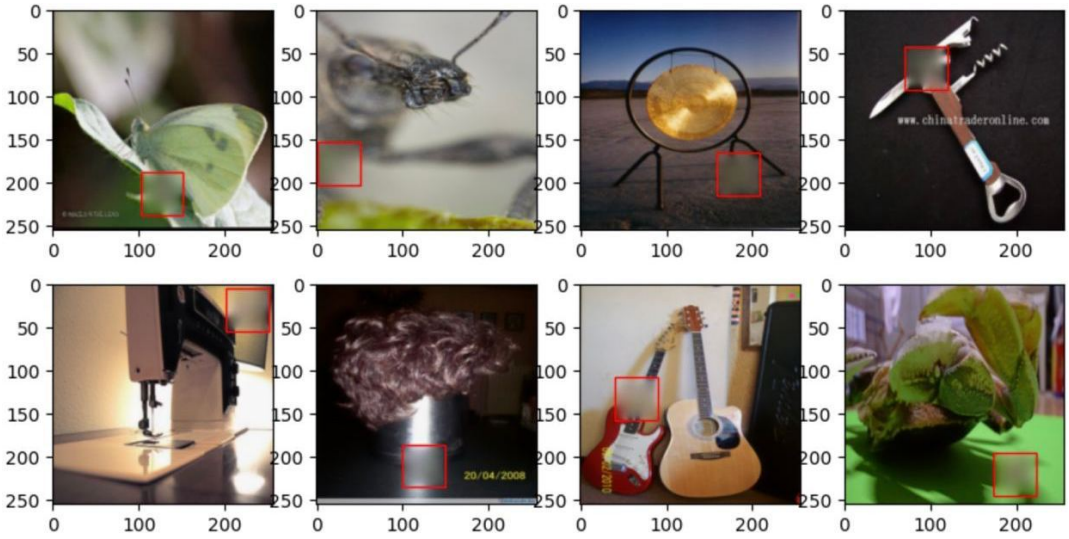

Figure 6: U-net completion result

5 Conclusions

From our results, we can see that the quality of many reconstructions wishes to be better. To improve it, we potentially can train the model on a bigger dataset or apply some augmentation methods. Another point is to include some components in the model pipeline such as DAM blocks or use more sophisticated architecture, in order to make reconstructed regions look more realistic.

For future work, it would be beneficial to experiment with different types of datasets, particularly those with varying levels of complexity, to see how well new architectures perform. Testing models in diverse scenarios will help identify the most effective methods for enhancing the quality of image inpainting using GANs

6 GitHub

Code implementation:

https://github.com/Bohdan213/mmml_image_inpainting

7 Bibliography

References

[1] Kim D. Cha D. Dam-gan : Image inpainting using dynamic attention map based on fake texture detection. 2022. https://arxiv.org/pdf/2204.09442.pdf.

[2] Jiannong C. Divya S. Generative adversarial networks (gans): Challenges, solutions, and future directions. 2020. https://arxiv.org/pdf/2005.00065.pdf.

[3] Al-Maadeed S. Akbari Y. Elharrouss O., Almaadeed N. Image inpainting: A review. 2019. https://arxiv.org/ftp/arxiv/papers/1909/1909.06399.pdf.

[4] Mirza M Xu B. Warde-Farley D. Ozair Sherjil. Courville A. Bengio Y. Goodfellow I. J., Pouget-Abadie J. Generative adversarial nets. jun 2013. https://arxiv.org/pdf/1406.2661.pdf.

[5] Simo-Serra-Edgar Ishikawa Hiroshi Iizuka, Satoshi. Globally and locally consistent image completion. *ACM Trans. Graph.*, 36(4), jul 2017. https://doi.org/10.1145/3072959.3073659.

[6] Brox Thomas Ronneberger O., Fischer P. U-net: Convolutional networks for biomedical image segmentation. 2015. https://arxiv.org/pdf/1505.04597.

[7] Zaremba W. Cheung V. Radford A. Chen X. Salimans T., Goodfellow I. Improved techniques for training gans. 2016. https://papers.nips.cc/paper_files/paper/2016/file/8a3363abe792db2d8761d6403605aeb7-Paper.pdf.