Image Outpainting on Crop Dataset

Course: Deep Learning

Instructor: Anupam Sobti

Teaching Fellows:
Moti Rattan Gupta, Varchita Lalwani, Yogendra Kumar

Team Members:
Harsh Mishra (U20220039)
Anshika Singh (U20220015)
Rohit Singh (U20220074)
Govind (U20220037)

Plaksha University

December 14, 2024

Contents

1 Problem Statement

In agricultural datasets, crop images are frequently incomplete, with missing or obscured regions that limit their usefulness for essential tasks such as disease prediction, weed detection, and accurate visual analysis. These gaps pose a significant challenge, as they hinder the ability to make reliable assessments and predictions. While existing techniques, such as image inpainting, can fill in missing areas within an image, they are not designed to extend beyond the boundaries of the image itself.

This limitation is particularly pronounced in domain-specific contexts like agriculture, where accurate extrapolation of image content is crucial for tasks such as creating complete visual representations of crops or analyzing growth patterns. Thus, there is a need for methods that not only fill in missing parts but also generate coherent and realistic content beyond the image's existing boundaries.

2 Objective

- **Develop a deep learning model for image outpainting:** The primary objective of this project is to create a deep learning model that performs image outpainting specifically on a rice crop dataset. This model will focus on generating realistic extensions of crop images beyond their original boundaries.
- **Utilize advanced techniques like hybrid residual skip connections:** To improve the model's performance, advanced architectural techniques such as hybrid residual skip connections will be incorporated. These techniques are designed to enhance the flow of information through the network, enabling more accurate and detailed image extrapolation.
- **Generate realistic and contextually coherent image extensions:** The model will be trained to generate extensions of crop images that are not only visually realistic but also contextually coherent with the existing image content. This will allow the model to produce seamless and natural-looking extrapolations, even when data is limited or incomplete.

3 Literature Review

Image Outpainting and Harmonization using Generative Adversarial Networks by Basile Van Hoorick, Columbia University (15th Feb, 2020) (Link)

The researchers trained their models on two datasets:

- **MIT CSAIL Places365-Standard:** A large dataset of natural photos containing landscapes, buildings, and everyday scenarios.
- **WikiArt:** A dataset of visual artwork belonging to many different categories.

Images from both datasets were resized to 192x192 pixels, and the generator was tasked with expanding a central 128x128 crop back to the original size. This means over half of the output image's pixels were hallucinated.

The researchers experimented with three models:

1. Trained on 200,000 images from Places365, using only L1 reconstruction loss.
2. Trained on 200,000 images from Places365, using both reconstruction and adversarial loss.
3. Trained on 50,000 images from WikiArt, using both reconstruction and adversarial loss.

3.1 Approach

The research focused on two outpainting methods:

- **Context Encoder:** Inspired by inpainting techniques.
- **Post-Processing Step:** Incorporates a single-image generative model for harmonization and enhanced resolution.

The architecture of the outpainting GAN consists of:

- **Generator Network (G):**
  - Based on inpainting architecture.
  - Uses a context encoder with 6 convolutional layers to downsample the input.
  - Decoder upsamples using deconvolutional layers to restore the original image size.
- **Discriminator Network (D):**
  - Estimates the probability of an image being real or generated.
  - Operates on the entire outpainted image to avoid inconsistencies.
  - Outputs a 24x24 grid of probabilities, with errors averaged during training.

A challenge during the training process was balancing the reconstruction and adversarial loss functions. Initially, the GAN struggled to incorporate the adversarial loss effectively, with the discriminator consistently outperforming the generator. To address this, the researchers implemented a dynamic weighting scheme for the adversarial loss, gradually increasing its influence as training progressed. This led to sharper and more realistic outputs.

3.2 Results

- **Mean Squared Error (MSE):** The models trained with adversarial loss exhibited higher MSE values compared to those relying solely on reconstruction loss. This was expected, as adversarial loss encourages the generation of realistic details, even at the cost of pixel-wise accuracy. The model trained on WikiArt achieved an average MSE of 0.0227.
- **Discriminator Output (Realism):** Including adversarial loss led to higher discriminator output values, indicating that the generated images were more likely to be perceived as real. The WikiArt model achieved the highest realism score, fooling the discriminator with a probability of 0.2371.

4 Project Methodology

4.1 Base Model

The first model used in our work is a Generative Adversarial Network (GAN), inspired by the research of Basile Van Hoorick. This serves as the base model, which will be used as a comparison metric for our architecture. The workflow of the base model is as follows:

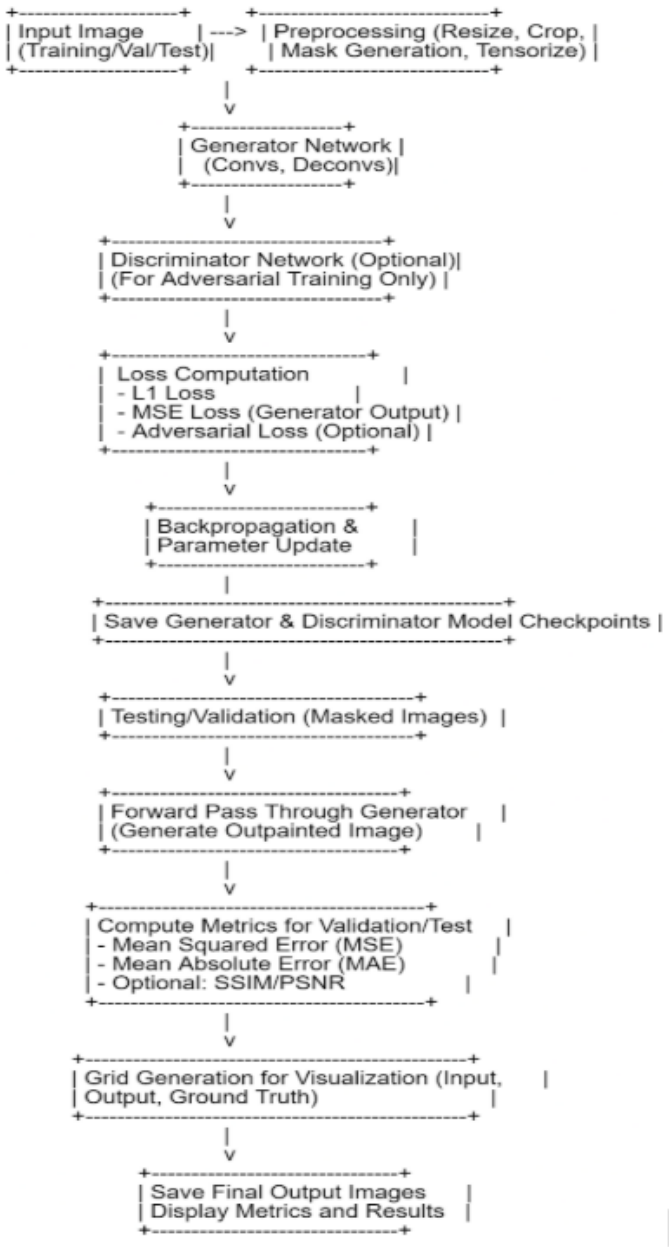


Figure 1: Workflow of the Base Model Architecture

- **Input Image:** The dataset used for this task includes training, validation, and test images. These images are essential for training the model and evaluating its performance on the outpainting task.
- **Preprocessing:** Prior to training, images undergo several preprocessing steps:

  - Resize, crop, and mask generation: Input images are resized to a consistent size, cropped to focus on the relevant content, and masked to simulate missing regions. These masks provide the model with the context it needs to learn how to predict and extend the missing parts.
  - Tensor conversion: The images are then converted into tensor format, which ensures compatibility with the deep learning framework and facilitates efficient model training.

- **Generator Network:** The generator network is responsible for producing realistic outpainted images based on the input. It consists of:
  - Encoder: The encoder downscales the input image through six convolutional layers, capturing the necessary contextual information for image extension.
  - Bottleneck: The encoder output is reduced to 4000 channels, providing a compact feature representation.
  - Decoder: The decoder upsamples the feature map using transposed convolutions, progressively reconstructing the image and generating realistic outpainted content.

- **Discriminator Network:** The discriminator's role is to evaluate the generated images and distinguish between real and fake ones. It consists of:
  - Evaluation: The discriminator uses four convolutional layers to downsample the image, learning high-level features that help it distinguish between real and generated images.
  - Output: The discriminator outputs a scalar value indicating whether the image is real or fake, providing feedback to the generator through adversarial loss.

- **Loss Computation:** The loss functions guide the optimization of the model during training. These losses include:
  - L1 Loss: Measures the pixel-level differences between the generated and ground truth images.
  - Mean Squared Error (MSE) Loss: Compares the pixel-wise differences between the generator's output and the ground truth.
  - Adversarial Loss: Used when adversarial training is applied, this loss encourages the generator to produce more realistic images.

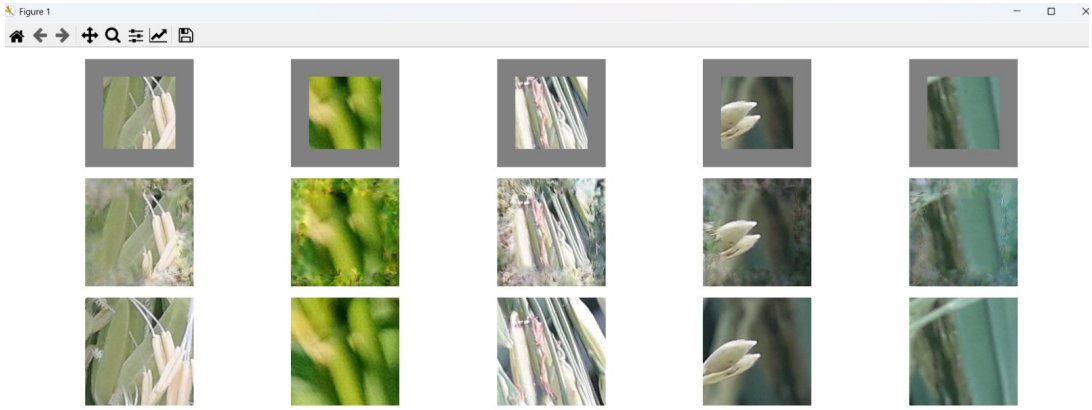
- **Backpropagation and Parameter Update:** Backpropagation is applied during training to adjust the model's weights, minimizing the computed loss and improving the model's performance over time.

- **Model Checkpoints:** To avoid losing progress during training, the model periodically saves the generator and discriminator. This ensures that the best-performing models are retained for further evaluation or use.

- **Testing/Validation:** Once the model is trained, it is evaluated using masked images to assess its ability to extrapolate and outpaint missing content. This step allows us to test how well the model generates missing parts of the image.

- **Forward Pass:** During testing, the trained generator is used to produce outpainted images. These are generated by predicting the content beyond the original image boundaries.
- **Validation/Test Metrics Computation:** The model's performance is evaluated using:
  - Mean Squared Error (MSE): Measures pixel-wise differences between generated and ground truth images.
  - Mean Absolute Error (MAE): Computes the absolute difference between predicted and true pixel values, serving as an alternative evaluation metric.
- **Grid Visualization:** To visualize and compare results, grids of input images, generated images, and ground truth images are created. This allows for an easy assessment of the quality and realism of the generated content.
- **Final Outputs and Results:** Finally, the outpainted images are saved, and performance metrics are summarized. These results help evaluate the model's success and provide insight into where further improvements can be made.

4.2 Output



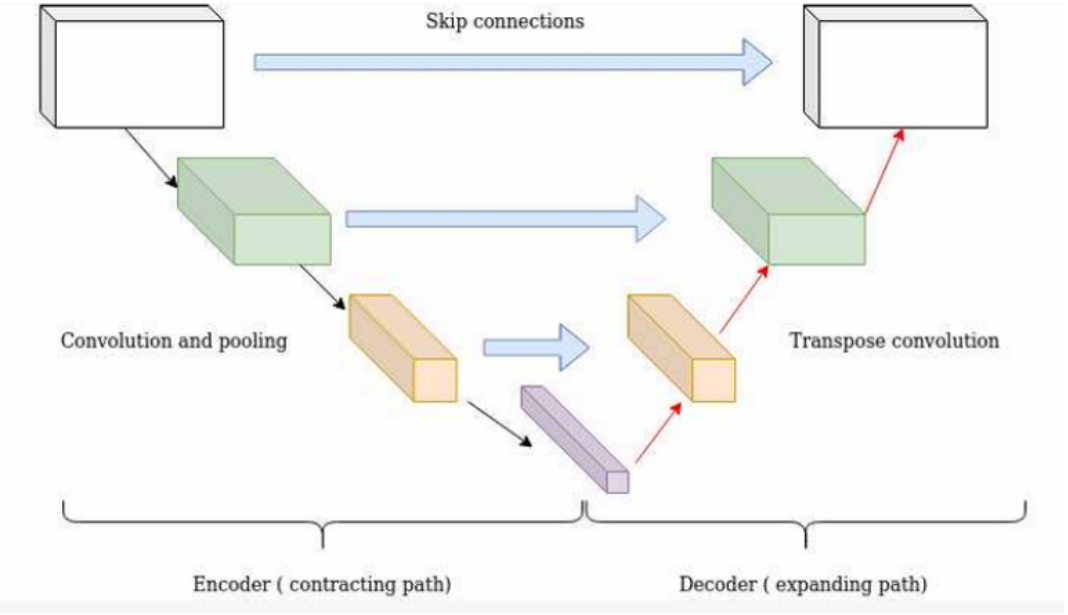
4.3 Result

```
. This limits the functions that could be executed during unpickling. Arbitrary objects
plicitly allowlisted by the user via `torch.serialization.add_safe_globals`. We recommen
t have full control of the loaded file. Please open an issue on GitHub for any issues re
  state_dict = torch.load(model_path, map_location=torch.device('cpu'))
Mean MSE: 0.0192
Mean MAE: 0.0920
```

- **Mean Squared Error (MSE):** 0.0192
- **Mean Absolute Error (MAE):** 0.092

5 Second Model

5.1 Residual Skip Connection Architecture



- **Encoder:**
  - **Convolutional Layers:** Apply a series of convolutional layers (64 to 256 filters), progressively reducing spatial dimensions.
  - **Dilated Convolutions:** Some layers use dilated convolutions to capture larger receptive fields and handle missing content.
- **Residual and Skip Connections:**
  - **Residual Connection:** Passes features from intermediate encoder layers directly to the decoder, retaining fine-grained details.
  - **Skip Connections:** Connect earlier encoder layers to corresponding decoder layers to help preserve low-level features during upsampling.
- **Latent Space:** Intermediate layers of the encoder form a latent space, capturing contextual information about the image, including missing regions.
- **Decoder:**
  - **Transposed Convolutions (Upsampling):** Uses deconvolutions to progressively upsample the feature maps from the latent space back to the original image resolution.
  - **Sigmoid Activation:** Applied to the final image to scale pixel values to the range [0, 1].

Generator Completion Network

Layer	Details	Activation Function	Purpose
Conv1	Conv2D (4, 64, kernel=5, stride=1, padding=2)	ReLU	Extracts low-level features from input
Conv2	Conv2D (64, 128, kernel=3, stride=2, padding=1)	ReLU	Downsampling with increased depth
Conv3, Conv4	Conv2D (128→128, 256→256, kernel=3, stride=1/2, padding=1)	ReLU	Multi-scale feature extraction
Conv5-Conv10	Conv2D (256→256, kernel=3, dilation=2/4/8/16, padding=corresponding)	ReLU	Captures large receptive fields for context
Conv11, Conv12	Conv2D (256→256, kernel=3, stride=1, padding=1)	ReLU	Further feature refinement
Deconv13, Deconv15	ConvTranspose2D (256→128, 128→64, kernel=4, stride=2, padding=1)	ReLU	Upsampling layers to reconstruct image size
Conv14, Conv16	Conv2D (128→128, 64→32, kernel=3, stride=1, padding=1)	ReLU	Refinement of upsampled features
Conv17	Conv2D (32→3, kernel=3, stride=1, padding=1)	Sigmoid	Produces final 3-channel output

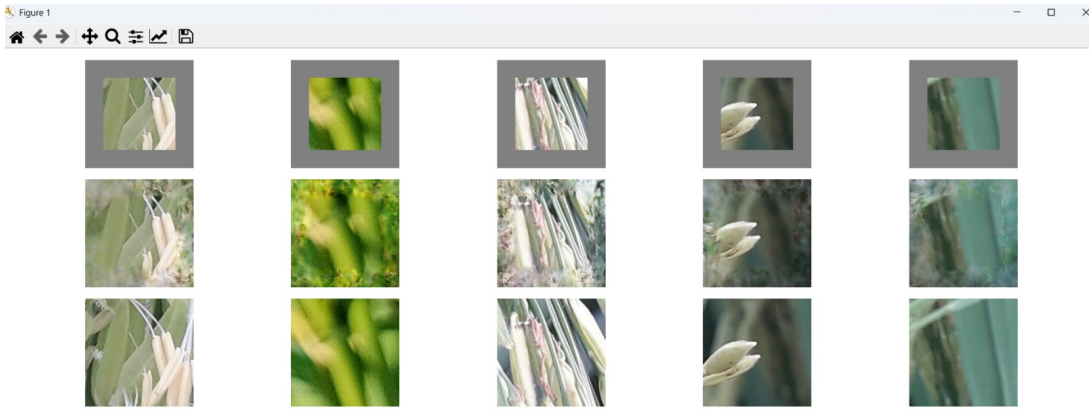
Local Discriminator

Layer	Details	Activation Function	Purpose
Conv1-Conv5	Conv2D (C→512, kernel=5, stride=2, padding=2)	ReLU + BatchNorm	Extracts multi-scale features from patches
Flatten6	Flatten	None	Flattens feature map for fully connected input
Linear6	Linear (Input→1024)	ReLU	Produces latent representation for the discriminator

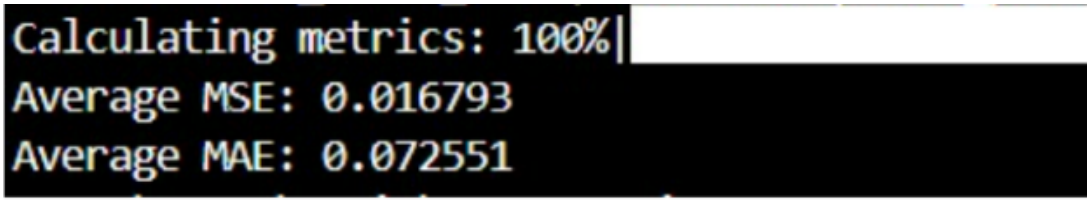
Global Discriminator

Layer	Details	Activation Function	Purpose
Conv1-Conv5	Conv2D (C→512, kernel=5, stride=2, padding=2)	ReLU + BatchNorm	Global feature extraction from entire input
Conv6	Conv2D (512→512, kernel=5, stride=2, padding=2) (for `places2`)	ReLU + BatchNorm	Additional global feature extraction
Flatten6/7	Flatten	None	Flattens feature map for fully connected input
Linear6/7	Linear (Input→1024)	ReLU	Produces latent representation for the discriminator

5.2 Output



5.3 Result



5.4 Novelty

- **Skip Connections:** Added residual connections to enhance gradient flow and feature propagation by bypassing intermediate layers.
- **Dilated Convolutions:** Expanded the receptive field to capture larger contextual information without increasing parameters.
- **Local and Global Discrimination:** Ensured fine-grained details and overall image consistency using both local and global discriminators.

Why?

- Residual connection is introduced in the Completion Network ($x = x + \text{residual}$), allowing better gradient flow, improving training stability, and model performance.
- The addition of skip connections helps retain spatial information by skipping layers, enhancing feature reuse and performance in deep architectures.

6 Key Learnings from Experimentation

1. **Importance of Skip Connections in Feature Preservation:**
   Experimenting with the base model revealed that skip connections are critical for retaining both high- and low-level features during outpainting. This learning has driven the development of a U-Net-like structure to enhance the quality of generated images.

2. **Role of Dense Residual Learning:**
   Dense residual blocks proved essential for preserving finer image details and ensuring smooth transitions in outpainted regions. The use of these blocks in our final model is expected to address the limitations of the base approach.

3. **Training Stability Challenges:**
   The base model training highlighted the instability of GANs, particularly when generating complex outpainting tasks. This reinforced the need for progressive learning techniques to improve training reliability and output quality.

4. **Insights into Model Scalability:**
   The base model provided an understanding of computational resource requirements, helping in planning for a more efficient implementation of the full model.

7 Conclusion

In this project, we tackled the challenge of image outpainting for crop datasets, a relatively underexplored field in the context of agriculture. Our GAN-based approach has demonstrated the ability to generate realistic extensions of incomplete crop images, thus enhancing the quality and usability of agricultural data.

By incorporating innovations like dense residual learning with skip connections, we were able to preserve fine details and improve the quality of outpainted regions.

8 References

1. Y. Kim, J. Lim, and C. Kim, "Painting Outside As Inside: Edge Guided Image Outpainting via Bidirectional Rearrangement with Progressive Step Learning," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, 2021, pp. 993-1002.

2. C. Cheng, C. Lu, and X. Ren, "InOut: Diverse Image Outpainting via GAN Inversion," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022, pp. 14090-14100.

3. K. Wang, J. H. Pan, C. C. Loy, and L. Lin, "Diverse Image Completion with Bidirectional GANs," *IEEE Transactions on Image Processing*, vol. 29, pp. 3497-3506, 2020.

4. M. Mirzaei and M. Abbasi, "A hybrid machine learning approach for intelligent IoT energy management systems in smart buildings," *Neural Networks*, vol. 156, pp. 225-233, 2023.

5. S. Zor, B. Baykal, and O. Koncagul, "Deep neural networks in decision support systems: Application of multi-objective optimization in power allocation problem," *Expert Systems with Applications*, vol. 213, 2022.

6. A. Liu and Y. Huang, "Generative Adversarial Networks for Image Completion," Stanford University, CS230 Project Report, Spring 2018.