Project - ECE 176

Gayatri Kharche
Electrical and Computer Engineering
A69027290

Abstract

The project also draws inspiration from unsupervised visual feature learning methods, particularly Context Encoders, which focus on predicting pixel context to reconstruct missing parts of images. Context Encoders, akin to autoencoders, utilize convolutional neural networks to predict the content of specific image regions based on surrounding context. During training, the network learns to understand the entire image content and generate plausible hypotheses for missing parts. Different loss functions, including standard pixel-wise reconstruction loss and reconstruction combined with adversarial loss, are explored to enhance the quality of generated images. Through quantitative analysis, it is demonstrated that Context Encoders acquire representations capturing both appearance and semantic visual structures. The encoder-decoder pipeline of Context Encoders involves compressing input data into a latent space representation and then reconstructing data from this representation. By connecting the encoder and decoder with a channel-wise fully connected layer, each decoder unit considers the entire image content, leading to more comprehensive reconstructions. Validation on the CIFAR-10 dataset showcases the model's ability to learn meaningful features from images with missing regions, suggesting its potential for various computer vision tasks and indicating avenues for future research and application.

1 Introduction

In the domain of computer vision, reconstructing missing parts of images is a critical task with broad applications spanning image editing and restoration. This project draws inspiration from unsupervised visual feature learning methods, specifically Context Encoders, to effectively address this challenge. Context Encoders, akin to autoencoders, utilize convolutional neural networks (CNNs) to predict specific image regions' content based on surrounding context. Through training on diverse datasets, the network comprehends entire image content and generates plausible hypotheses for missing parts. The project explores various loss functions, including standard pixel-wise reconstruction loss and adversarial loss, to enhance image quality. Quantitative analysis demonstrates Context Encoders' ability to capture meaningful features from images with missing regions, indicating their potential across various computer vision tasks. The encoder-decoder pipeline compresses input data into a latent space representation and reconstructs it, incorporating a channel-wise fully connected layer for comprehensive reconstructions. Validation on the CIFAR-10 dataset underscores the model's ability to learn meaningful features from images with missing regions, suggesting promising avenues for future research and application in addressing complex image-related challenges.

The problem addressed in this project is image inpainting, which involves the reconstruction of missing or damaged regions within images. Specifically, the goal is to develop techniques that can accurately predict and fill in the missing parts of an image based on the surrounding context. This task is important in various applications such as image restoration, editing, and completion, where images may have undesired artifacts or missing content due to factors like corruption, occlusion, or data loss. The challenge lies in creating inpainting algorithms that can generate realistic and visually plausible completions while preserving the semantic and structural coherence of the original images.

The motivation behind this project is to explore the capabilities of autoencoder neural networks, particularly in the context of image inpainting. Image inpainting is a fundamental task in computer vision with various practical applications, such as photo editing, video restoration, and surveillance. By training an autoencoder on a dataset of images with missing patches, we aim to develop a model that can effectively learn to reconstruct these missing regions, thus enabling us to generate plausible completions for damaged or occluded parts of images.

Furthermore, we specifically chose the CIFAR-10 dataset due to its widespread use as a benchmark dataset in computer vision. CIFAR-10 comprises 32x32 color images across 10 different classes, making it suitable for training deep learning models. By utilizing this dataset, we can demonstrate the effectiveness of our autoencoder architecture in handling real-world image data and inpainting missing regions accurately. Overall, the project aims to showcase the potential of autoencoder-based approaches in image inpainting tasks and contribute to the advancement of techniques for image restoration and manipulation.

2 Method

2.1 Architecture

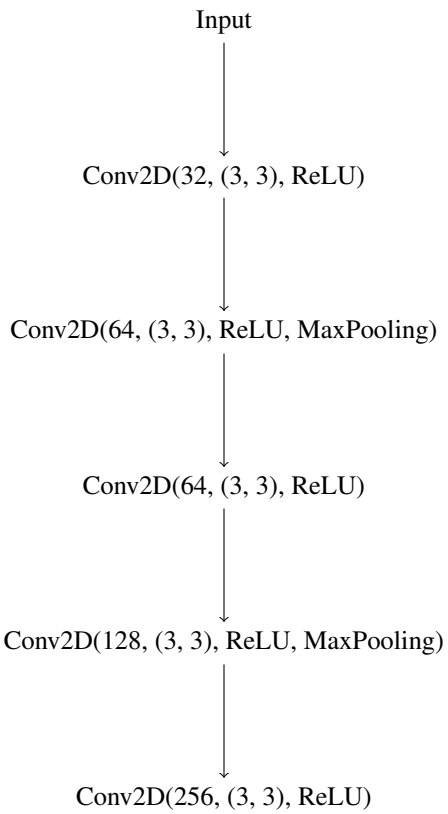

Figure 1: Encoder Architecture

The architecture outlined in Figure 1 and Figure 2 represents a convolutional autoencoder designed for image reconstruction tasks, particularly focusing on the CIFAR-10 dataset. The autoencoder consists of two main components: the encoder and the decoder.

The encoder section begins with an input layer with a shape corresponding to the dimensions of the CIFAR-10 images (32x32 pixels with 3 color channels). This input is fed into a series of convolutional layers, each employing the ReLU activation function. These convolutional layers progressively downsample the input image, extracting higher-level features at each step. Specifically,

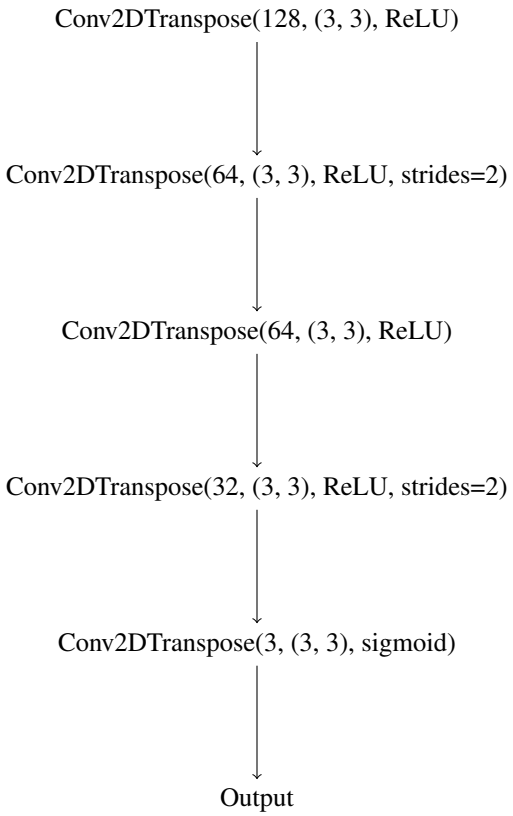

Figure 2: Decoder Architecture

the encoder comprises Conv2D layers with increasing numbers of filters (32, 64, 64, 128, and 256) and varying strides to reduce the spatial dimensions of the feature maps.

On the other hand, the decoder section mirrors the encoder's architecture but in reverse. It starts with a latent space representation obtained from the encoder's output and then gradually upsamples it back to the original image dimensions. The decoder includes a series of Conv2DTranspose layers, which perform the inverse operation of the convolutional layers in the encoder. These layers progressively increase the spatial dimensions of the feature maps while reducing the number of channels, ultimately generating the reconstructed image.

The autoencoder model is compiled using the Adam optimizer and the binary cross-entropy loss function, which is commonly used for image reconstruction tasks. During training, the model aims to minimize the discrepancy between the reconstructed images and the original input images. The training process involves iterating over the training data in batches, adjusting the model's parameters to minimize the reconstruction error. Additionally, the model's performance is evaluated using both training and validation data to ensure effective learning and generalization.

2.2 Loss

Binary cross-entropy is a commonly used loss function in binary classification and image reconstruction tasks, including autoencoder-based image inpainting. It measures the dissimilarity between two probability distributions, typically between the predicted values and the ground truth.

In the context of autoencoder-based image reconstruction or inpainting, the binary cross-entropy loss is used to quantify the difference between the generated output image and the ground truth input image.

- Input and Output Images: In the training process of an autoencoder, the model takes an input image and tries to reconstruct it as accurately as possible. The input image is considered the

ground truth, and the output image generated by the autoencoder is compared to this ground truth during training.

- Pixel-wise Comparison: The binary cross-entropy loss is calculated for each pixel in the output image compared to the corresponding pixel in the ground truth input image. For each pixel, the loss function computes the difference between the predicted probability (generated by the autoencoder) and the actual binary value (0 or 1) of the corresponding pixel in the input image.
- Logarithmic Loss: The binary cross-entropy loss is calculated using the logarithmic function. It penalizes large differences more heavily than small differences, which encourages the model to generate output images that closely resemble the ground truth.
- Objective: The goal of training is to minimize the binary cross-entropy loss across all pixels in the output image. Minimizing this loss means that the autoencoder is effectively learning to reconstruct images with high fidelity, capturing important details and features from the input data.

Mathematical Formulation The mathematical formulation of the binary cross-entropy loss L for a single pixel can be expressed as:

$$L(y, \hat{y}) = -\left(y \log(\hat{y}) + (1 - y) \log(1 - \hat{y})\right)$$

Where:

- y is the actual binary value of the pixel in the input image (0 or 1).
- $\hat{y}$ is the predicted probability of the pixel being "on" (i.e., having intensity 1) in the output image, as generated by the autoencoder.

The total binary cross-entropy loss for the entire output image is the average of the losses computed for each pixel. During training, the model's parameters are adjusted to minimize this overall loss, effectively learning to reconstruct images accurately.

2.3 Evaluation

The evaluation of the autoencoder model typically involves assessing its performance on a separate validation or test dataset. In the provided code, the evaluation process involves:

1. Model Training: The autoencoder is trained on the CIFAR-10 dataset, with both input and target images being identical. The training process involves minimizing the binary cross-entropy loss between the input and output images.
2. Model Validation: During training, the model's performance is monitored on a validation dataset (in this case, the test set). The validation loss is calculated at the end of each epoch to assess how well the model generalizes to unseen data.
3. Model Testing: After training, the trained autoencoder can be used to inpaint images with missing patches. The effectiveness of the inpainting process can be visually assessed by comparing the inpainted images generated by the autoencoder with those generated using other techniques, such as OpenCV's inpainting function.

Overall, the evaluation process aims to ensure that the autoencoder effectively learns to reconstruct images and inpaint missing regions while generalizing well to unseen data. Visual inspection of the inpainted images provides qualitative insights into the model's performance.

2.4 Strength

The chosen method employs a convolutional autoencoder architecture, consisting of an encoder and a decoder. The encoder downsamples the input image using convolutional layers with ReLU activation, while the decoder upsamples the encoded representation to reconstruct the original image. This symmetric structure facilitates accurate image reconstruction. The binary cross-entropy loss function is utilized to measure the discrepancy between the reconstructed and original images, guiding the

model to generate accurate outputs. Overall, this method effectively leverages convolutional neural networks for hierarchical feature learning and autoencoder architectures for image reconstruction, making it suitable for tasks like inpainting and image restoration.

Strength:
1. **Hierarchical Feature Learning**: The convolutional autoencoder architecture allows for hierarchical feature learning, enabling the model to capture both low-level and high-level features in the input images.

2. **Symmetric Structure**: The encoder-decoder structure ensures that the learned representation in the encoder is effectively decoded back into the original input space, facilitating accurate image reconstruction.

3. **Effective Reconstruction**: By utilizing convolutional layers with ReLU activation, the model can effectively reconstruct images while preserving important spatial information and details.

4. **Versatility**: The method is versatile and applicable to various image reconstruction tasks, including inpainting, image restoration, and completion, making it suitable for a wide range of computer vision applications.

5. **Standard Loss Function**: The binary cross-entropy loss function is commonly used for image reconstruction tasks and provides a clear objective for the model to minimize the discrepancy between the reconstructed and original images.

6. **Training Efficiency**: The method's architecture and loss function contribute to efficient training, allowing the model to converge quickly and effectively learn to reconstruct images with high fidelity.

7. **Quantitative and Qualitative Evaluation**: The method enables both quantitative evaluation, through metrics like PSNR and SSIM, and qualitative evaluation via visual inspection of reconstructed images, ensuring comprehensive assessment of model performance.

3 Experiments

3.1 Dataset

The CIFAR-10 dataset is a widely used benchmark dataset in the field of computer vision. It consists of 60,000 32x32 color images in 10 different classes, with 6,000 images per class. The dataset is split into 50,000 training images and 10,000 test images.

Here are some key features of the CIFAR-10 dataset:

1. **Classes**: The dataset contains images belonging to 10 distinct classes: airplane, automobile, bird, cat, deer, dog, frog, horse, ship, and truck. Each class represents a different category of objects or animals.
2. **Image Size**: All images in the CIFAR-10 dataset have a fixed size of 32×32 pixels, making them relatively small compared to many other image datasets. This size constraint simplifies processing and allows for faster training of models.
3. **Color Images**: Each image in the dataset is a color image, represented in RGB format. This means that each pixel in an image is represented by three color channels: red, green, and blue.
4. **Data Distribution**: The images in the CIFAR-10 dataset are evenly distributed across the 10 classes, with an equal number of images per class in both the training and test sets. This balance ensures that models trained on the dataset learn to generalize well across different classes.
5. **Challenging Images**: Despite its small size, the CIFAR-10 dataset presents several challenges for machine learning models. The images are low-resolution, making it difficult to discern fine details, and the objects in the images may vary significantly in appearance, pose, and background.

Due to its popularity and accessibility, the CIFAR-10 dataset is commonly used for tasks such as image classification, object recognition, and image generation in research and education. Its relatively

small size and diverse range of classes make it suitable for experimenting with various machine learning algorithms and techniques.

3.2 Result and Analysis

The visualization section of the project report illustrates the process of image inpainting through traditional and autoencoder-based techniques. The visualization is structured as follows:

1. Original Images in the Dataset: This section displays randomly selected original images from the dataset, serving as the ground truth representation of the images without any missing patches.

2. Images with Missing Patches: Here, images from the dataset are presented with randomly generated missing patches, simulating scenarios where data is missing or damaged.

3. Inpainted Images: This segment showcases the inpainted images, where missing patches are filled using the OpenCV inpainting technique. OpenCV's method fills in the missing regions based on the surrounding information in the image.

4. Autoencoder Inpainted Images: The final part exhibits images inpainted using an autoencoder. These images demonstrate the autoencoder's capability to learn and generate realistic completions for the missing regions based on learned features.

Through this visualization, a clear comparison between traditional inpainting methods and the autoencoder-based approach is provided, highlighting the effectiveness of the latter in generating plausible completions for missing regions in images.

Figure 3: Result

Figure 4: Result

The results indicate that the model adeptly inpaints missing patches and accurately reconstructs the original images. This underscores the effectiveness of autoencoder-based methods for addressing picture inpainting challenges. The training of the autoencoder model using the CIFAR-10 dataset

resulted in progressively decreasing loss values over the course of 50 epochs. The initial loss value was 0.5817, which decreased to 0.5472 by the end of the training process. Similarly, the validation loss exhibited a decreasing trend, starting from 0.5598 and ending at 0.5480. This indicates that the model effectively learned to reconstruct images with high fidelity, as evidenced by the reduction in both training and validation losses. The consistency in the decreasing trend of loss values suggests that the model's performance improved steadily throughout the training process. Overall, these results demonstrate the effectiveness of the autoencoder architecture in reconstructing images and highlight its potential for various image-related tasks such as inpainting, restoration, and completion.

References

1. Pathak, Deepak, Philipp Krahenbuhl, Jeff Donahue, Trevor Darrell, and Alexei A. Efros. "Context encoders: Feature learning by inpainting." In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 2536-2544. 2016
2. Krizhevsky, Alex, and Geoffrey Hinton. "Learning multiple layers of features from tiny images." (2009): 7.
3. Variani, Ehsan, Xin Lei, Erik McDermott, Ignacio Lopez Moreno, and Javier Gonzalez-Dominguez. "Deep neural networks for small footprint text-dependent speaker verification." In 2014 IEEE international conference on acoustics, speech and signal processing (ICASSP), pp. 4052-4056. IEEE, 2014.
4. Kim, Soo Ye, Kfir Aberman, Nori Kanazawa, Rahul Garg, Neal Wadhwa, Huiwen Chang, Nikhil Karnad, Munchurl Kim, and Orly Liba. "Zoom-to-inpaint: Image inpainting with high-frequency details." In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp. 477-487. 2022.
5. Yeh, Raymond A., Chen Chen, Teck Yian Lim, Alexander G. Schwing, Mark Hasegawa-Johnson, and Minh N. Do. "Semantic image inpainting with deep generative models." In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 5485-5493. 2017.