Image Compression Using a Convolutional Autoencoder

Mariam Bouzidi

Abstract

This paper explores image compression using a convolutional autoencoder, a deep learning technique that encodes data into a compressed representation and reconstructs it with minimal loss. The focus is on reducing the dimensionality of facial images while preserving key features. The autoencoder was trained on grayscale images, demonstrating strong reconstruction capability. Applications of this model can extend to various fields, including image compression, denoising, and feature extraction.

Introduction

Autoencoders are an unsupervised learning approach used for dimensionality reduction. In contrast to traditional methods like PCA, autoencoders can capture complex, nonlinear relationships within the data. Convolutional autoencoders are specifically designed for image data, leveraging convolutional layers to preserve spatial relationships. These models are widely applied in tasks such as image compression, noise reduction, and anomaly detection. In this study, we utilize a convolutional autoencoder to compress facial images and evaluate the effectiveness of this approach in reconstructing the original images.

Methodology

Data Preprocessing

The dataset used for this project contains images of faces, resized to 128x128 pixels and converted to grayscale. Each image is normalized by scaling pixel values between 0 and 1. This preprocessing step is essential for ensuring that the model converges efficiently during training. The dataset is split into training and testing sets, with 90% of the data used for training.

Convolutional Autoencoder

Autoencoders consist of two main components: the encoder and the decoder. The encoder compresses the input into a lower-dimensional representation (latent space), while the decoder reconstructs the original data from this compressed form. The architecture of our convolutional autoencoder is composed of:

- **Encoder:** Two convolutional layers with ReLU activation, each followed by a down-sampling operation (strides=2), reducing the image dimensions by half.

- **Latent Space:** A bottleneck that represents the compressed image, containing the most important features in a low-dimensional format.
- **Decoder:** Two transpose convolutional layers that up-sample the compressed image back to its original dimensions, followed by a sigmoid activation to output pixel values between 0 and 1.

Training

The model is trained using the Mean Squared Error (MSE) loss function, which measures the difference between the original and reconstructed images. We use the Adam optimizer with a learning rate of 0.001. The model is trained over 5 epochs with a batch size of 32, sufficient to achieve satisfactory reconstruction performance.

Results

The convolutional autoencoder successfully reconstructed the input images with minimal loss of detail. The reconstructed images retained important facial features, even though they were compressed into a lower-dimensional space. Qualitatively, the output images were nearly indistinguishable from the originals. The Mean Squared Error between the original and reconstructed images was consistently low, indicating that the model effectively learned to compress and decompress the images without significant distortion.

Discussion

The results demonstrate the efficacy of convolutional autoencoders for image compression. One of the advantages of this approach is that it captures spatial hierarchies in images due to the convolutional layers, making it particularly well-suited for visual data. In future work, this model could be improved by experimenting with deeper architectures or applying transfer learning to more complex datasets. Additionally, the application of this model for tasks such as noise reduction and feature extraction could provide further insights into its potential.

Conclusion

This study showcases the use of convolutional autoencoders for image compression, achieving significant reduction in dimensionality while preserving key image features. The approach has potential applications in data storage, communication, and real-time image processing tasks. Further exploration into optimizing model architectures and expanding its use cases could yield promising results for the broader machine learning and computer vision fields.
