Practice #3

Compression

April 2, 2024

1 Introduction

Compression algorithms are widely used to reduce the size of data, making it easier and quicker to store and transmit. During the last decades the compression algorithms development has had a huge success.

1.1 Categorization

Data compression algorithms can be categorized based on various criteria, including their approach, purpose, and underlying techniques. Here's a broad categorization of data compression algorithms:

- Based on Approach
  - Lossless Compression: Algorithms that reduce the size of data without any loss of information. Examples include Huffman coding, Lempel-Ziv (LZ) algorithms, and Run-Length Encoding (RLE).
  - Lossy Compression: Techniques that sacrifice some data fidelity to achieve higher compression ratios, commonly used for multimedia data like images, audio, and video. Examples include JPEG (images), MP3 (audio), and MPEG (video).
- Based on Purpose
  - General-Purpose Compression: Algorithms designed to compress a wide range of data types, commonly used for file compression and general data storage.
  - Specialized Compression: Targeted compression methods optimized for specific types of data, such as text, images, audio, and video.
- Based on Techniques
  - Dictionary-Based Compression: Algorithms using a dictionary or codebook to represent data more efficiently, such as LZW (Lempel-Ziv-Welch) and LZ77/LZ78.

  - Statistical Compression: Techniques utilizing statistical properties of the data to achieve compression, including Huffman coding and Arithmetic coding.
  - Transform-Based Compression: Methods involving transformation of the data into a different domain for more effective compression, popular examples being JPEG (DCT) and MP3 (MDCT).
  - Entropy Coding: Encodes the data based on its entropy or information content, often used in combination with other compression techniques (e.g., Huffman coding, Shannon-Fano coding).
  - Adaptive Compression: Algorithms that adapt to the specific properties of the data, adjusting compression parameters based on the data's characteristics and content.

- Based on Domain
  - Image Compression Algorithms: Techniques specifically tailored for compressing digital images, including JPEG, PNG, and GIF.
  - Audio Compression Algorithms: Methods designed to compress audio data, such as MP3, AAC, and Ogg Vorbis.
  - Video Compression Algorithms: Optimized compression methods for video data, such as MPEG-2, H.264 (AVC), and HEVC (H.265).

- Based on New Approaches. As discussed previously, new approaches to compression can include hybrid techniques, content-adaptive methods, learned compression, perceptual compression, and more, focusing on advanced, data-specific strategies and adaptability.

This broad categorization illustrates the diverse landscape of data compression algorithms, showcasing how different techniques and methods are utilized to achieve efficient data storage and transmission in specific domains and use cases.

2 Machine learning for image compression

Machine learning techniques have garnered increasing attention in the domain of image compression, aiming to address the ever-growing demands for efficient data representation and transmission. These techniques offer novel approaches to compressing image data, leveraging neural network architectures to learn compact yet expressive representations of visual content. One compelling avenue in this pursuit is the utilization of autoencoders, a class of neural networks renowned for their ability to learn succinct and meaningful representations of complex data. In this task, we embark on a journey that aims to harness the power of autoencoders to revolutionize the landscape of image compression.

The task at hand centers on exploring the transformative potential of autoencoders in the domain of image compression. Autoencoders, with their inherent capability to autonomously learn and encode essential features from input data, promise an exceptionally intriguing alternative to conventional image compression methods. By training these neural networks to encode images into compact, yet informative representations, our objective is to redefine the paradigm of image compression, transcending the limitations of traditional compression techniques.

Our mission involves delving into the intricacies of autoencoder architectures, fine-tuning them to encapsulate the salient visual characteristics present within diverse images. Through this endeavor, we aim to equip autoencoders with the prowess to condense image data into compressed representations that conserve vital details, thereby minimizing data redundancy and optimizing storage and transmission without compromising the fidelity of the visuals.

As we embark on this task, we seek to unravel the potential of autoencoders in reshaping image compression paradigms. By harnessing their latent capacity to distill the essence of visual information into compact encodings, we endeavor to chart new frontiers in efficient image storage and transmission. Through our efforts, we aspire to forge a path toward image compression methodologies that leverage the robust foundation of autoencoders to yield superior compression performance and uphold the integrity of visual data.

2.1 Autoencoders

Autoencoders are a type of neural network architecture used in unsupervised learning tasks to learn efficient representations of input data. They consist of an encoder and a decoder, with the encoder transforming the input data into a compact representation (latent space) and the decoder reconstructing the original input from this representation. By using autoencoders one can obtain both lossless and lossy data compression.

- Lossless Compression
  - In the context of autoencoders, a lossless compression scheme aims to preserve all information of the input data in the compressed representation. This means that the original data can be perfectly reconstructed from the compressed form, without any loss of information. Autoencoders used for lossless compression typically employ a bottleneck architecture, where the dimension of the latent space is chosen to be equal to or greater than the input dimension to ensure exact reconstruction.
  - Training a lossless autoencoder involves minimizing a loss function that measures the difference between the input data and the reconstructed output. Common loss functions for this purpose include mean squared error (MSE) or binary cross-entropy, depending on the nature of the input data.
- Lossy Compression

  - Conversely, in lossy compression, the objective is to reduce the size of the input data by discarding certain information deemed less critical, with the understanding that some loss of fidelity in the reconstructed output is acceptable. In the context of autoencoders, the latent space dimension is typically smaller than the input dimension, forcing the network to learn a compressed representation of the input data by discarding non-critical information.
  - Training a lossy autoencoder involves optimizing a loss function that balances reconstruction fidelity with the extent of compression. This can be achieved using a combination of reconstruction loss (e.g., MSE or other suitable losses) and a regularization term that encourages sparsity or other constraints on the latent space representation.

Applications:

- Lossless autoencoders are beneficial in scenarios where exact preservation of data is crucial, such as in archival storage, signal processing, or applications where loss of information is not tolerable.
- On the other hand, lossy autoencoders are often employed in scenarios where the tradeoff between compression and fidelity is acceptable, such as image or audio compression in multimedia applications, where reducing data size without perceptible loss is desirable.
- Both lossless and lossy autoencoders can be adapted for various types of data, including images, audio, text, and more, and their performance is influenced by factors such as the choice of architecture, latent space dimension, and the nature of the input data.

3 Task

In this lab task, you will explore the application of autoencoders for image compression. You will use a dataset of images and design an autoencoder architecture to achieve both lossless and lossy compression for the images. Through this task, you will gain practical experience with neural network design, training, and evaluation using autoencoders for image data.

Task steps:

- **Dataset Selection**: Select a dataset of images (for example, the MNIST dataset or any other dataset of interest) to use for training and testing your image compression autoencoder.
- **Architecture Design**: Design an autoencoder architecture suitable for image compression. This should include an encoder and a decoder component. Consider the choice of convolutional and pooling layers for the encoder and upsampling layers for the decoder to effectively capture and reconstruct image features.

- **Lossless Compression Training**: Train the designed autoencoder for lossless image compression. Configure the latent space dimension to be equal to or greater than the input dimension to ensure exact reconstruction. Use appropriate loss function such as mean squared error (MSE) to measure the difference between the input images and the reconstructed outputs.
- **Lossy Compression Training**: Modify the autoencoder architecture to achieve lossy image compression. Configure the latent space dimension to be smaller than the input dimension to encourage information loss during compression. Utilize suitable regularization and reconstruction loss functions to balance compression with fidelity in reconstruction.
- **Evaluation and Comparison**: Evaluate the performance of the trained autoencoder for both lossless and lossy compression. Measure the compression ratio achieved by the autoencoder for lossy compression and the fidelity of reconstructed images in both lossless and lossy settings.
- **Comparative Analysis**: Compare the reconstructed images from the lossless and lossy compression settings to assess the tradeoff between compression ratio and reconstruction fidelity. Discuss the advantages and limitations of lossless and lossy image compression using autoencoders.
- **Visualization**: Visualize the latent space representations of the images before and after compression to gain insights into the learned compressed representations and the level of data preservation in the encoded space.

Deliverables:

1. A report detailing the entire process, including the dataset selection, architecture design, training process, evaluation results, and comparative analysis.
2. Visualizations and explanations of the latent space representations.
3. Code implementation and results from the lossless and lossy compression experiments (.py or .ipynb).

By completing this lab task, you will gain hands-on experience in using autoencoders for image compression and develop insights into the tradeoffs involved in achieving efficient compression with varying degrees of fidelity.

Helpful tutorials

- https://www.tensorflow.org/tutorials/generative/autoencoder
- https://theailearner.com/2019/01/01/compression-of-data-using-autoencoders/
- https://blog.paperspace.com/autoencoder-image-compression-keras/
- https://www.edureka.co/blog/autoencoders-tutorial/
- https://github.com/gr-b/autoencoder-latent-space-visualization