DeepCompress Instruction Manual

Table of Contents

---

1. Introduction

DeepCompress is a toolkit for compressing and decompressing images using a Fully Convolutional Autoencoder (FCN). It provides both a user-friendly web interface and a command-line tool for batch processing. The project is suitable for research, education, and practical image compression tasks.

2. Features

- Compress images to a compact latent representation using deep learning.
- Decompress latent representations back to images with minimal loss.

- Web app (Streamlit) for interactive use.
- Command-line interface for automation and scripting.
- Support for custom training on your own datasets.
- Pretrained model available (via Git LFS).

3. System Requirements

- Python 3.7 or higher
- pip (Python package manager)
- Git and Git LFS (for pretrained model)
- Supported OS: Windows, macOS, Linux
- Recommended: CUDA-compatible GPU for training (optional for inference)

4. Installation

4.1. Clone the Repository

```
git clone <your-repo-url>
cd deepcompress
```

4.2. Download Pretrained Model (if available)

Install Git LFS if not already installed:

- Git LFS Installation Guide

Then run:

```
git lfs pull
```

4.3. Install Python Dependencies

It is recommended to use a virtual environment:

```
python -m venv venv
source venv/bin/activate  # On Windows: venv\Scripts\activate
pip install -r requirements.txt
```

5. Model Training

You can use the provided scripts to train the autoencoder model.

5.1. Training on CIFAR-10

Downloads and trains on the CIFAR-10 dataset:

```
python train_cifar10.py
```

The trained model will be saved as `fcn_compression_ae.pth`.

5.2. Training on a Custom Dataset

Place your training images in `./data/train` and run:

```
python train.py
```

This will also produce a `.pth` model file.

6. Using the Web App (Streamlit)

1. Ensure the model file (`fcn_compression_ae.pth`) is present in the project directory.
2. Start the web app:

   ```
   streamlit run app.py
   ```

3. Open the provided local URL in your browser.
4. Upload an image (JPG/PNG), set the output quality, and click "Compress & Decompress".
5. Download the reconstructed image.

7. Using the Command Line Interface (CLI)

The CLI tool is provided in `compress.py` and supports both compression and decompression.

7.1. Compress an Image

```
python compress.py compress path/to/input.jpg -l path/to/output_latent.npy
```

- `path/to/input.jpg`: Path to your input image.

- `-l path/to/output_latent.npy`: (Optional) Output file for the latent representation (default: `fcn_compressed_latent.npy`).

7.2. Decompress an Image

```
python compress.py decompress path/to/output_latent.npy -o path/to/reconstructed.jpg -q 85
```

- `path/to/output_latent.npy`: Path to the latent file saved during compression.
- `-o path/to/reconstructed.jpg`: (Optional) Output image path (default: `fcn_reconstructed.jpg`).
- `-q 85`: (Optional) JPEG quality (default: 85).

8. File Descriptions

- `app.py`: Streamlit web application.
- `compress.py`: CLI tool for compression and decompression.
- `dataset.py`: Dataset loader for training.
- `model.py`: Fully Convolutional Autoencoder model definition.
- `train.py`: Training script for custom datasets.
- `train_cifar10.py`: Training script for CIFAR-10.
- `requirements.txt`: Python dependencies.
- `fcn_compression_ae.pth`: Pretrained model weights (downloaded via Git LFS).
- `README.md`: Quick start and usage guide.

9. Troubleshooting

- **Model file not found:**
  - Ensure you ran `git lfs pull` after cloning.
  - Check that `fcn_compression_ae.pth` is in the project directory.

- **CUDA errors:**
  - If you do not have a GPU, the code will automatically use CPU.
- **Streamlit not found:**
  - Run `pip install -r requirements.txt` to install all dependencies.
- **Image size errors:**
  - The model expects images with dimensions divisible by 8. Padding is handled automatically.

10. FAQ

Q: Can I use my own images for training? A: Yes. Place your images in `./data/train` and run `python train.py`.

Q: How do I change the model architecture? A: Edit `model.py` and retrain the model.

Q: Can I use this for grayscale images? A: The current model is for RGB images. You can adapt it for grayscale by changing the input/output channels in `model.py`.

Q: Is a GPU required? A: No, but training is much faster with a CUDA-compatible GPU.

11. Contact

For questions, bug reports, or contributions, please contact:

- Aviv Elbaz (aviv000)
- Ron Butbul (RonButbul626)

Or open an issue on the project repository.