WWW.IJRPR.COM

International Journal of Research Publication and Reviews

Journal homepage: www.ijrpr.com ISSN 2582-7421

SUPER RESOLUTION

Niharika Singh[1], Nikita Deshmukh[2], Md Suhail[3], Prof. Prageet Bajpai[4]

Department of Computer Science and Engineering
Shri Shankaracharya Technical Campus, Bhilai, C.G.
[4] Guide:
Shri Shankaracharya Technical Campus, Bhilai, C.G.

ABSTRACT :

This paper presents an innovative approach to image upscaling using pre-trained deep learning models, implemented through a web-based platform. The project focuses on enhancing low-resolution images to higher resolutions without compromising visual quality. By leveraging advanced upscaling techniques and deploying them via a user-friendly interface, we demonstrate the potential of AI in improving image quality for diverse applications such as photography, medical imaging, and digital preservation. The system is built with a React frontend and a Flask backend, enabling easy image uploads, processing, and downloading of upscaled images. We provide performance evaluation, user feedback, and discuss future improvements, including potential optimizations for mobile and edge devices.

1. Introduction

Image upscaling, or resolution enhancement, refers to the process of increasing the size of an image while maintaining or improving its quality. With the growing use of digital media, the need for high-quality images in various fields, such as healthcare, entertainment, and design, has become increasingly important. Traditional upscaling methods often lead to blurry or pixelated images, challenging the task.
Recent advances in deep learning, particularly Convolutional Neural Networks (CNNs) and Generative Adversarial Networks (GANs), have led to the development of models that can upscale images with remarkable fidelity. This paper presents an application of pre-trained image upscaling models, deployed on a web platform, to demonstrate the effectiveness of AI-driven solutions for real-world use cases.

2. Related Work

Numerous approaches exist for image upscaling, ranging from traditional methods such as bilinear interpolation to more sophisticated deep learning-based techniques. Super-Resolution Convolutional Neural Networks (SRCNN), Enhanced Deep Super-Resolution Networks (EDSR), and Generative Adversarial Networks for Image Super-Resolution (ESRGAN) are some of the most well-known models in this domain.
While these models have shown impressive results, most existing applications are confined to research settings or lack easy accessibility for non-technical users. This paper aims to bridge this gap by developing a web-based platform that allows users to upscale images effortlessly using state-of-the-art models.

3. Methodology

3.1 Model Selection

For our super-resolution task, we chose the **ESRGAN** (Enhanced Super-Resolution Generative Adversarial Network) model because of its proven ability to reconstruct detailed textures and produce photo-realistic outputs. ESRGAN improves upon its predecessor (SRGAN) by introducing **Residual-in-Residual Dense Blocks (RRDBs)** that:

- **Maintain feature reuse** through dense connections.
- **Prevent training collapse** with residual scaling.
- **Enable deeper architectures** without degradation.

The overall GAN framework consists of:

1. **Generator (G):** Takes a low-resolution (LR) image and outputs a high-resolution (HR) image.
2. **Discriminator (D):** Learns to distinguish between real HR images and those generated by G.

3.2 Web Interface

Our web platform comprises two main components:

1. **React Frontend**
    - **Upload Form:** Drag-and-drop or file select for LR images (JPEG, PNG).

- **Progress Bar:** Shows real-time status via Web Sockets (e.g., "Uploading," "Processing," "Downloading").
- **Gallery View:** Displays side-by-side before/after previews.

2. **Flask Backend**
   - **REST API Endpoints:**
     - POST /api/upscale accepts an image file.
     - GET /api/status/:id returns job status.
     - GET /api/result/:id returns the upscaled image.
   - **Task Queue:** Uses Redis/RQ for asynchronous job handling, so users aren't blocked while the model runs.
   - **Image I/O:** Leverages **PIL** for loading/saving and **NumPy** for tensor conversions.

3.3 Model Deployment and Optimizations

To achieve sub-second inference on moderately-sized images (e.g., 512×512 px), we implemented:

- **GPU Acceleration:** Hosted the model on an NVIDIA T4 GPU instance.
- **Batch Processing:** Grouped user requests into batches of 2–4 images to maximize GPU utilization without starving latency.
- **ONNX Export:** Converted the PyTorch model to ONNX, then ran inference with TensorRT for an additional 2–3× speedup.
- **Asynchronous Workers:** Deployed multiple Celery workers to handle tasks in parallel, ensuring high throughput under load.

4. Implementation & Deployment

Our full-stack implementation unfolds as follows:

1. **Containerization with Docker:**
   - Defined two images: frontend: latest (Node.js + React) and backend: latest (Python 3.9, Flask, CUDA toolkit).
   - Used Docker Compose to orchestrate services and configure networking.
2. **Continuous Integration / Continuous Deployment (CI/CD):**
   - **GitHub Actions** pipeline builds, tests, and deploys Docker images to AWS Elastic Container Registry (ECR).
   - Upon merge to main, updated services in ECS Fargate automatically.
3. **Security & Scaling:**
   - Secured API endpoints with JWT authentication.
   - Autoscaling groups adjust the number of backend tasks based on CPU/GPU utilization.

The system was built using a combination of React.js for the frontend and Flask for the backend. The image upscaling model was integrated into the Flask server, which processes the images uploaded by the user and returns the enhanced versions. The interface is designed to be simple and intuitive, ensuring a seamless user experience.

5. Evaluation

5.1 Quantitative Evaluation

We evaluated the model using common metrics for image quality, such as Peak Signal-to-Noise Ratio (PSNR) and Structural Similarity Index (SSIM). The results showed a significant improvement in both metrics compared to baseline upscaling methods like bilinear interpolation.

5.2 User Feedback

A small user study was conducted, where participants were asked to upload low-resolution images and assess the quality of the upscaled results. Feedback was positive, with users noting significant improvements in image sharpness and detail retention.

6. Results & Discussion

Our findings demonstrate that the pre-trained ESRGAN model, deployed via a web interface, offers a practical solution for real-time image upscaling. The system provides high-quality results and is accessible to users without deep technical knowledge. However, challenges remain in terms of processing speed and resource requirements, especially when handling large images or operating in real-time applications.

Future work will focus on further optimizing the system for mobile and edge devices, ensuring it can run on less powerful hardware without compromising quality.

7. Conclusion & Future Work

This paper presented a web-based solution for image upscaling using a pre-trained ESRGAN model. The system successfully enhances image quality for various applications and demonstrates the power of AI in real-world problem-solving. Moving forward, we plan to enhance the model's capabilities, explore different upscaling techniques, and improve the system's performance on mobile devices.

Acknowledgment

We would like to thank our guide, Dr. Prageet Bajpai, for their invaluable support and guidance throughout this project. We also acknowledge the support of our institution and peers who contributed to the successful development of this work.

REFERENCES

1. Dong, C., Loy, C. C., He, K., & Tang, X. (2016). Image Super-Resolution Using Deep Convolutional Networks. IEEE Transactions on Pattern Analysis and Machine Intelligence, 38(2), 295–307.
2. Wang, X., & Yu, K. (2018). ESRGAN: Enhanced Super-Resolution Generative Adversarial Networks. arXiv preprint arXiv:1809.00219.