TOPAZ LABS

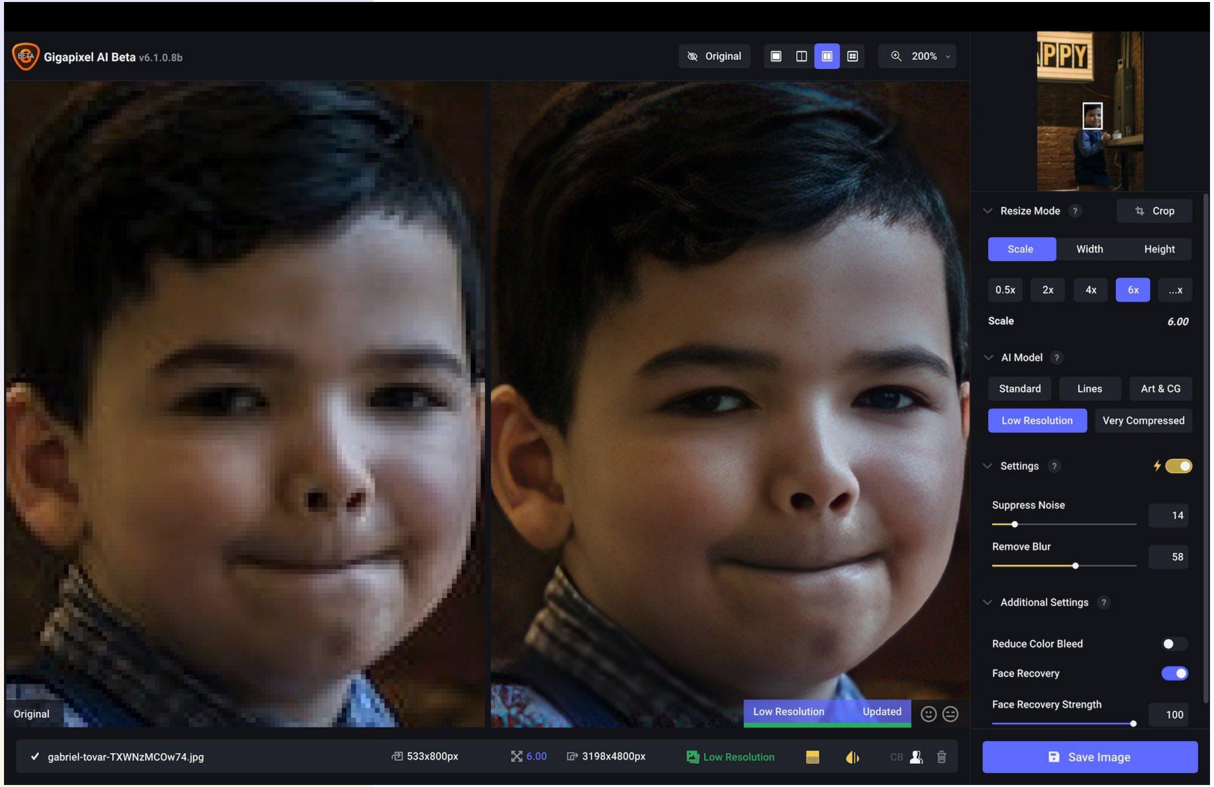
Deploying on Desktop with ONNX

Alexander Zhang, Topaz Labs

alexander.zhang@topazlabs.com

TOPAZ LABS

Gigapixel AI



- Photo upscaling software for Windows and MacOS
- Plugins for popular photo editing applications
- Automatically maximizes processing speed on a variety of desktop hardware
- >100 new or greatly improved models across all apps since 2018

Desktop deployment

- Integrates into existing photography and videography workflows
- Avoids many concerns about internet bandwidth, privacy, etc.
- Little control over system configuration
- May need to support old operating systems and constrained hardware
- Should scale with latest GPUs and drivers
- Usually running alongside other applications

Training

TensorFlow

PyTorch

ONNX

Inference

OpenVINO™

ONNX RUNTIME

TensorRT

ML

Model Conversion Challenges

- Must be aware of feature support in each inference library
- Libraries may desire different input dimensions, channel order, etc.
- Operators may be interpreted differently
- Output may change slightly but unpredictably after conversion
- Variety of hardware may be required for conversion and testing converted model

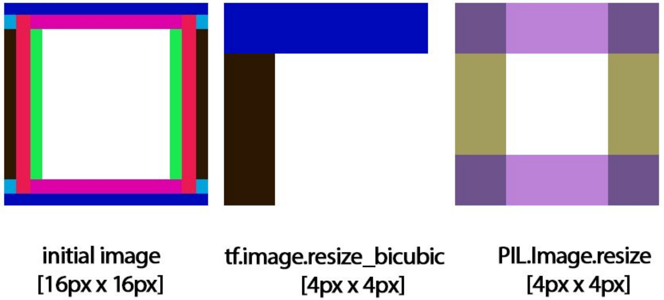


Image from Oleksandr Savsunenko

Model Conversion Process

- Verify conversions are possible before training
- Create different intermediate ONNX models for each inference library
- Make ONNX as unambiguous as possible, potentially avoid troublesome operators
- Compare before and after conversion model outputs both numerically and visually
- Setup automated tools for running pipeline on different machines

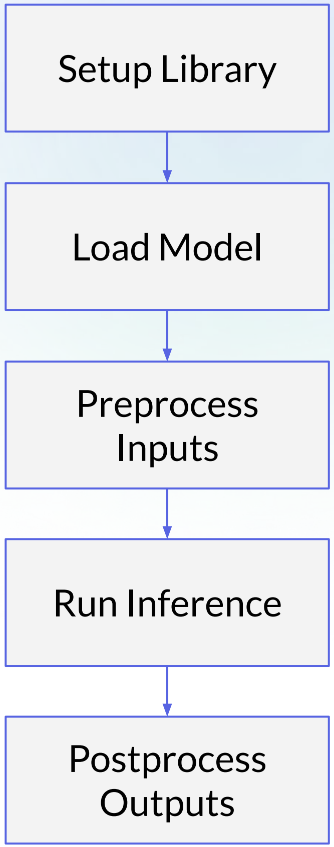
TOPAZ LABS

Why use your own inference wrapper?

- Have tried using same OpenGL or ONNX Runtime code on all hardware in the past
- Hardware specific tuning and optimization can make a big difference in performance
- Avoid doing conversion or compilation work at load time
- More quickly understand and incorporate your specific fixes and workarounds
- Better control over performance characteristics and feature priorities

Inference Pipeline Tasks

- Should determine library compatibility before loading to minimize required downloads and setup time
- Select fastest of multiple inference libraries and make use of different available GPUs, but fallback to different libraries or devices on error
- Efficiently process images of arbitrary size both as previews and for batch output
- Smooth over differences between libraries in image format, memory allocation, device selection, etc.



Inference Pipeline

- Describe system requirements, input names and dimensions, file download URL etc. for each inference library in a JSON blob
- Prefer device with most RAM, then compatible inference library most specific to that device; reselect after library-specific errors
- Split images into blocks that can be batched or dispatched to different devices in parallel, allow models to customize per-block processing

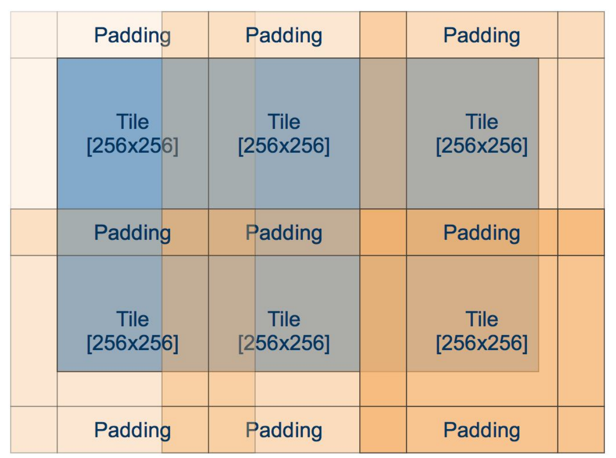


Image from Ievgen Khvedchenia

Future Work

- More quickly add support for new operators and architectures
- Reduce time spent on converting models and testing on different hardware
- Manage proliferation of model files for different libraries, hardware, block sizes, etc.
- Perform more pre/post processing pipeline on GPU without extra PCIe transfers.
- Further reduce loading time when chaining multiple large models per image

TOPAZ LABS

Thank you!

Alexander Zhang, Topaz Labs

alexander.zhang@topazlabs.com