best

ONNX Model Analysis Report

Generated by ONNX Autodoc v0.1.0 on 2025-12-02T22:55:03.620054Z

Executive Summary

TL;DR: The model "best.onnx" is a convolutional neural network (CNN) with approximately 43.6 million parameters and a model size of about 174.4 MB, primarily designed for image processing tasks. Its architecture includes a significant number of convolutional layers and non-standard residual connections, making it suitable for complex image classification or object detection applications.

The analyzed ONNX model, "best.onnx," is a convolutional neural network (CNN) characterized by a complex architecture comprising 392 nodes and 43607379 parameters, primarily driven by 104 convolutional layers. With a total of approximately 165 billion floating-point operations (FLOPs) and a model size of around 174 MB, it exhibits significant computational complexity, particularly in convolution operations. Key architectural patterns include a high prevalence of non-standard residual connections, such as gated and concatenation-based skip connections, which may necessitate careful handling during deployment due to their custom nature. The model is suitable for deployment on an NVIDIA GeForce RTX 4050 Laptop GPU, fitting comfortably within the available VRAM, though it may face compute bottlenecks given its high FLOP count and theoretical latency of approximately 16.5 ms. Notably, the absence of normalization layers could impact training stability, and the unconventional residual patterns warrant attention to ensure optimal performance.

Generated by gpt-4o-mini

43.61M

Parameters

165.09B

FLOPs

174.43 MB

Model Size

CNN

Architecture

KV Cache (Transformer Inference)

Metric	Value
Per Token	6.14 KB
Full Context (seq=640)	3.93 MB
Layers	1
Hidden Dim	768

Memory Breakdown by Op Type

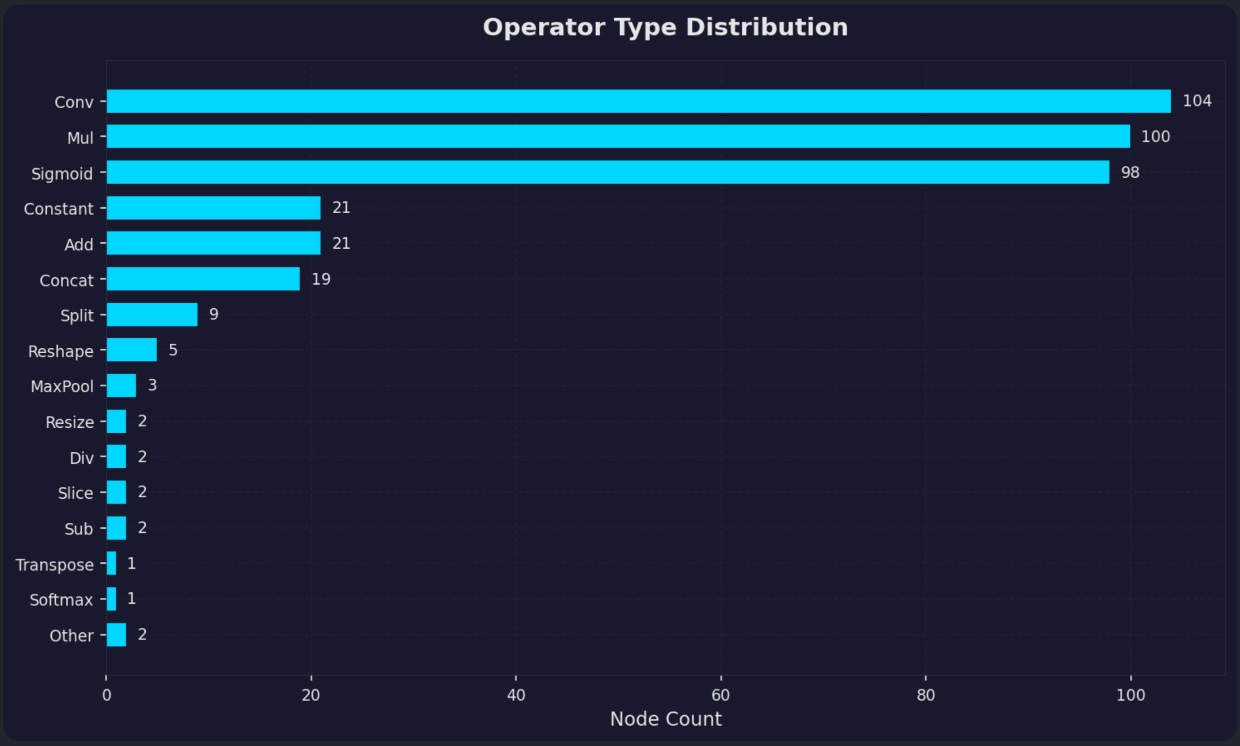
Component	Size
Conv	174.43 MB

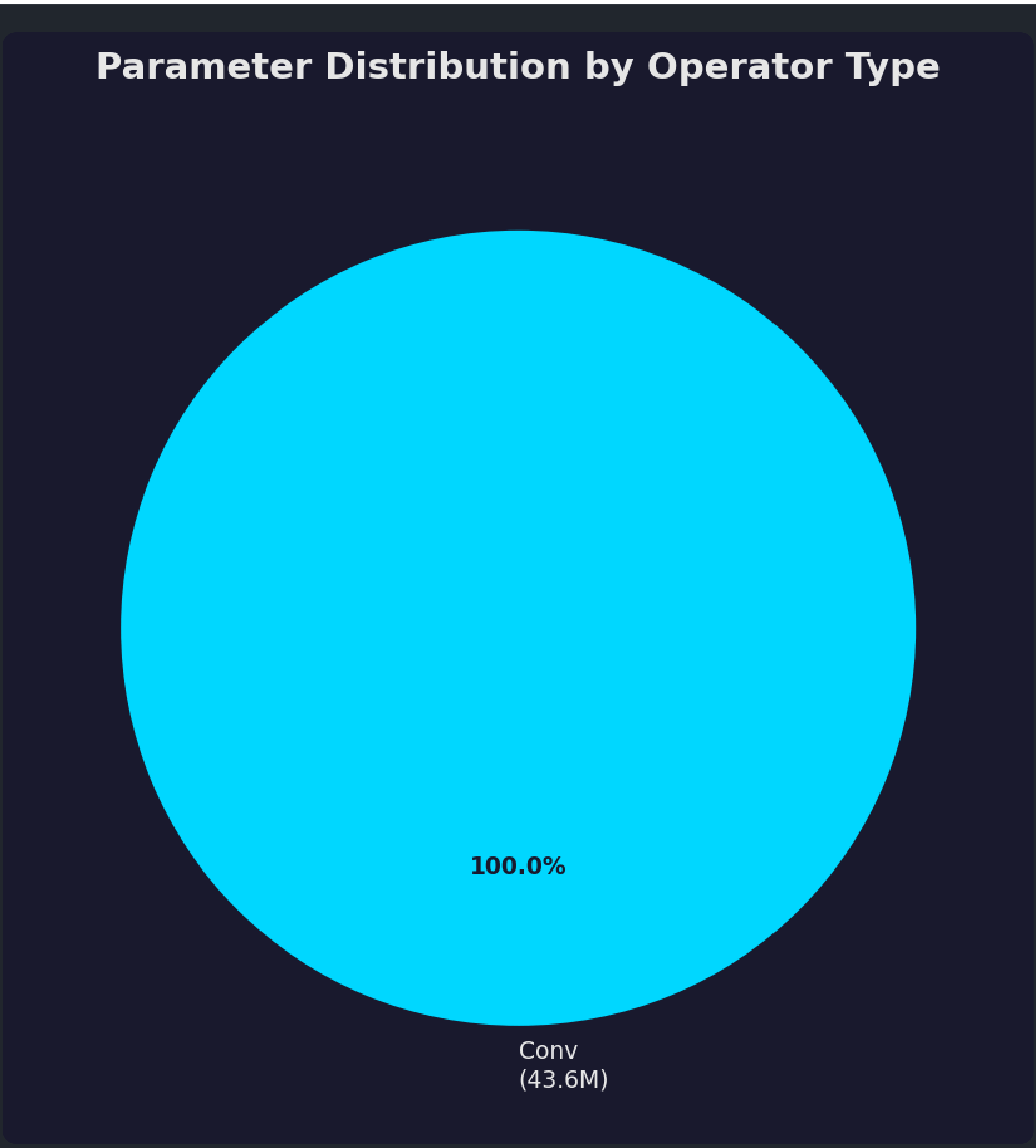
Parameter Details

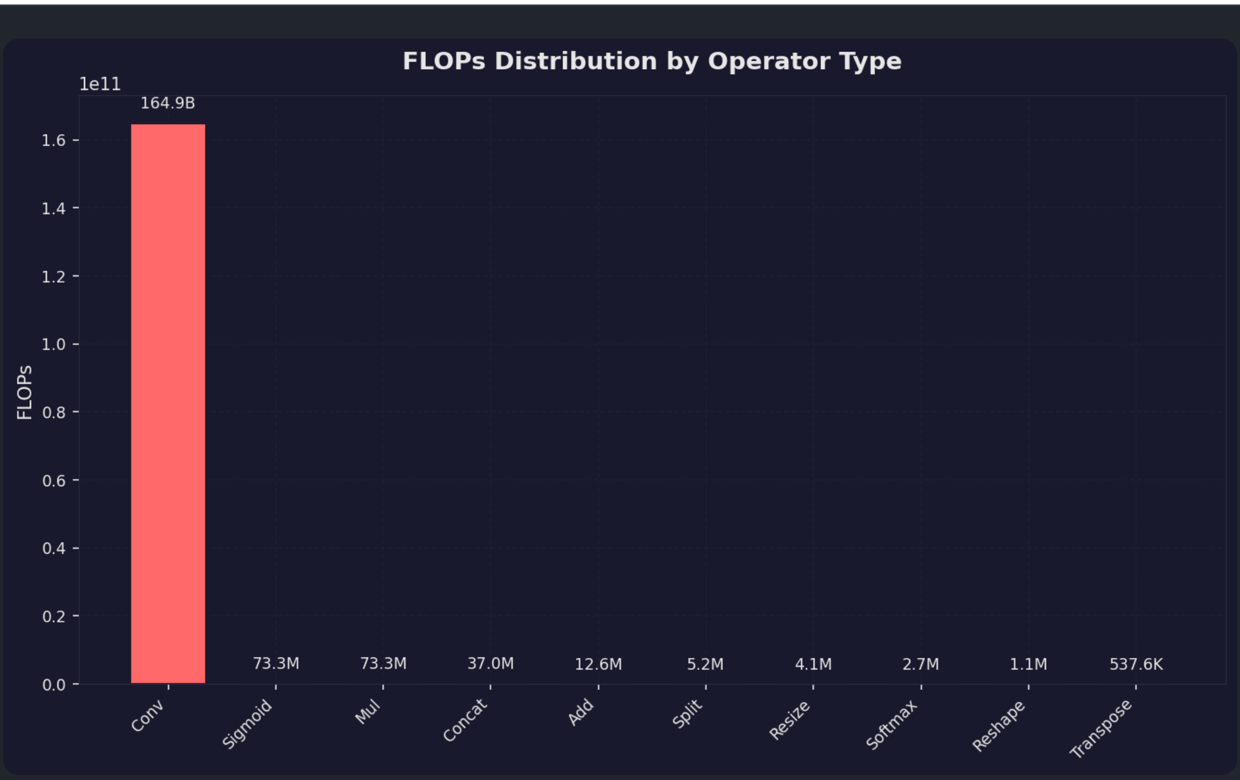
Precision Breakdown

Data Type	Parameters
fp32	43.61M

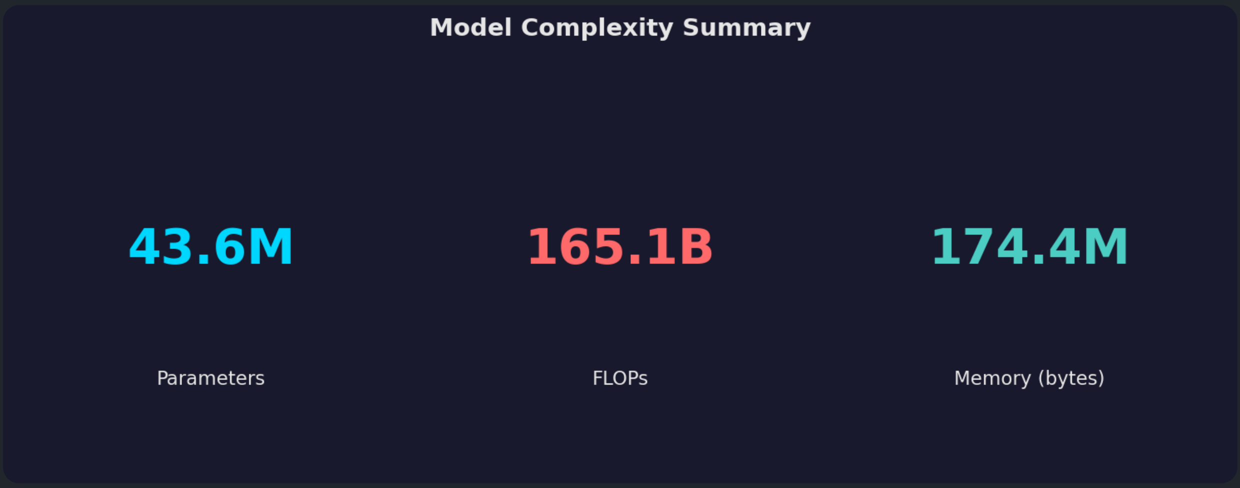
Visualizations




Parameter Distribution by Operator Type
100.0%
Conv
(43.6M)



FLOPs Distribution by Operator Type



Model Complexity Summary

43.6M
Parameters

165.1B
FLOPs

174.4M
Memory (bytes)

Model Details

Metadata

Property	Value
IR Version	8
Producer	pytorch 2.5.1
Opsets	ai.onnx:17

Graph Summary

Metric	Value
Nodes	392
Inputs	1
Outputs	1
Initializers	207

Inputs

- `images` : [1, 3, 640, 640]

Outputs

- `output0` : [1, 5, 8400]

Operator Distribution

Operator	Count
Conv	104
Mul	100
Sigmoid	98
Constant	21
Add	21
Concat	19
Split	9
Reshape	5
MaxPool	3
Resize	2
Div	2
Slice	2
Sub	2
Transpose	1
Softmax	1
...	(2 more)

Architecture

Detected Type: CNN

Detected Blocks

Block Type	Count
ConvSigmoid	97
ResidualGate	97
ResidualAdd	21
ResidualConcat	5
ResidualSub	2

Non-Standard Skip Connections

This model uses 104 non-standard skip connection(s):

- ▶ Concat-based (DenseNet-style) (5)
- ▶ Gated skip (Highway/attention) (97)
- ▶ Subtraction-based (2)

Dataset Info

Task: detect

Number of Classes: 1

▶ Class Names (1 classes) - click to expand

Source: ultralytics

Hardware Estimates

NVIDIA GeForce RTX 4050 Laptop GPU (detected)

Precision: fp32 | Batch Size: 1

Metric	Value
VRAM Required	311.55 MB
Fits in VRAM	Yes
Theoretical Latency	16.5093 ms
Bottleneck	compute
Compute Utilization	70%
GPU Saturation	1.65e-02 (1.6509%)

Device Specifications

- **VRAM:** 6.44 GB
- **FP32 Peak:** 10.0 TFLOPS
- **FP16 Peak:** 20.0 TFLOPS

Risk Signals

INFO **missing_normalization**

Model has 104 trainable layers but no normalization layers detected. This may affect training stability.

Recommendation: If this model will be fine-tuned, consider adding normalization layers. For inference-only, this is typically not a concern.

INFO **nonstandard_residuals**

Model uses 104 non-standard skip connection(s): 5 concat-based (DenseNet-style), 97 gated (Highway/attention gate), 2 subtraction-based. Model also has 21 standard Add-based residuals. These patterns may indicate custom architectures requiring special attention.

Recommendation: Non-standard skip connections are valid but may need special handling: Concat-based patterns increase tensor sizes through the network. Gated patterns add compute overhead but enable selective information flow. Ensure your deployment target and optimization tools support these patterns.