Faster model inference with Tensorflow - TensorRT integration

Rajesh Shreedhar Bhat - Sr. Data Scientist, Walmart Global Tech

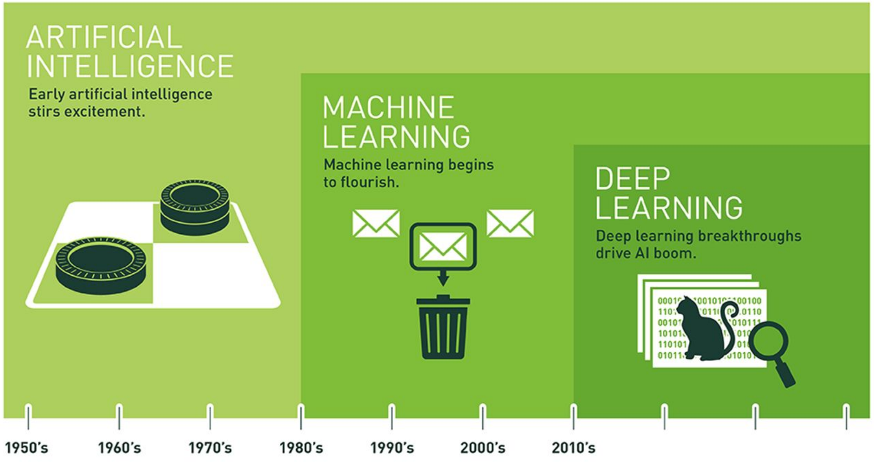
Experts
2022
Machine Learning

Agenda

- Deep learning
  - Overview
  - Deep learning in multiple domains
  - Model training vs inference
  - Challenges
- Nvidia-TensorRT
  - Overview
  - Optimizations for faster model inference.
- Tensorflow - TensorRT code example.
- Key takeaways
- Q&A.

Deep Learning

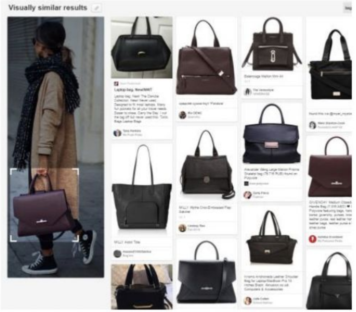
Deep learning is part of a broader family of machine learning methods based on artificial neural networks with representation learning.

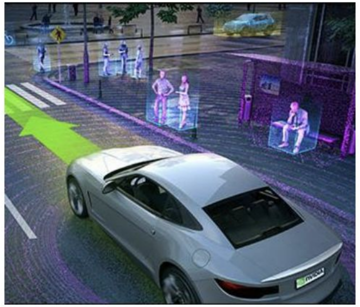


Since an early flush of optimism in the 1950s, smaller subsets of artificial intelligence – first machine learning, then deep learning, a subset of machine learning – have created ever larger disruptions.

Deep learning applications

- Speech Recognition
- Recommender Systems
- Autonomous Driving
- Real-time object recognition.
- Language Translation
- Many more ..







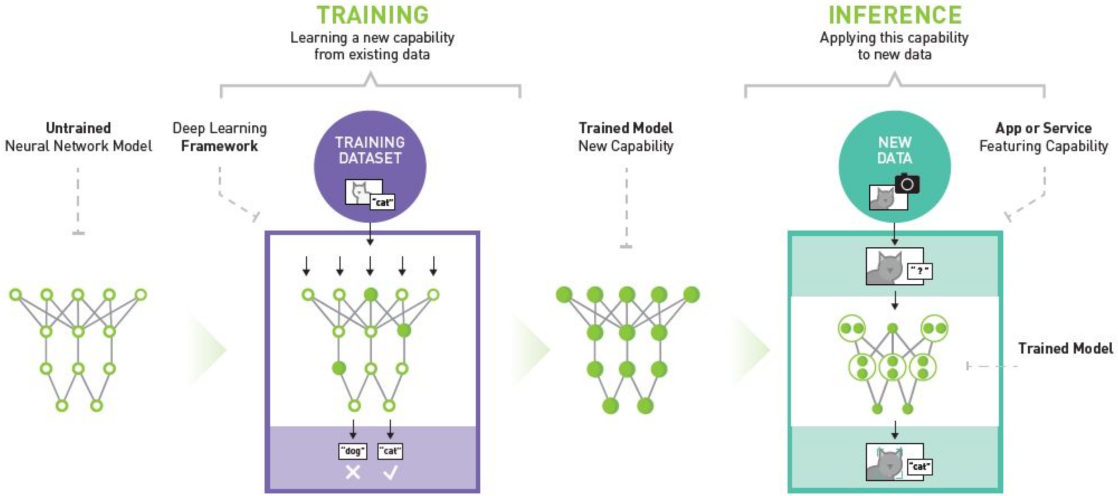
Deep learning model training vs inference

Training

- Iterative
- Computationally intensive
- Training time - several hours to days on GPU's !

Inference (Prod Environment)

- Real-time
- Batch jobs
- Cloud vs Edge



Challenges with model inference

Requirement of :

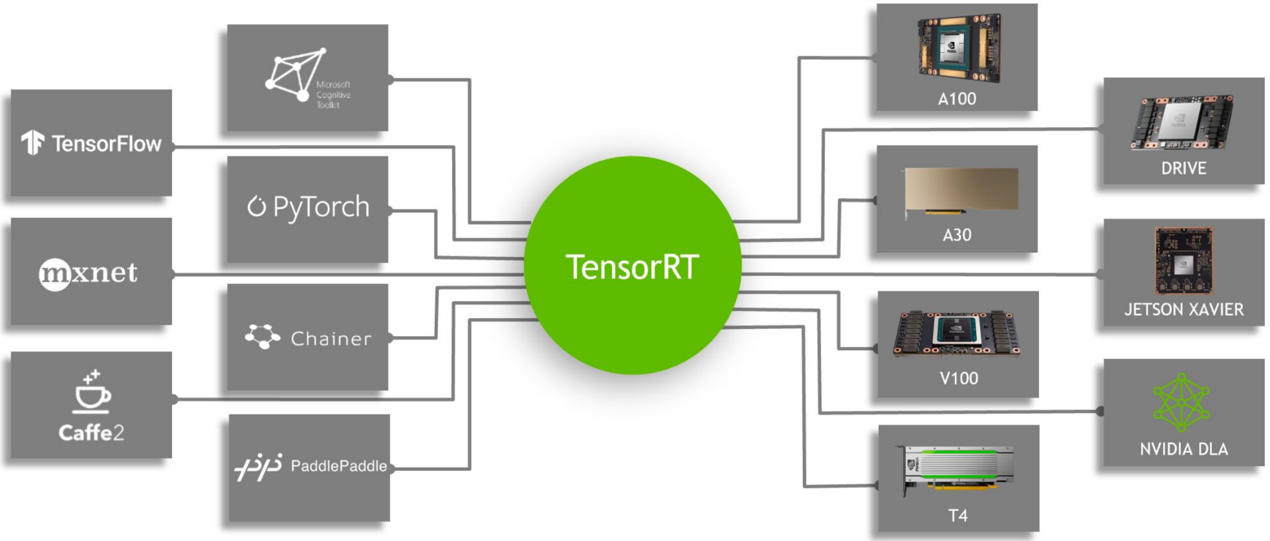
- High Throughput
  - Challenge: handling high volume, high velocity data
  - Impact : increased processing time resulting in higher compute costs.
- Low latency
  - Challenge: Delivering real-time results.
  - Impact : poor user experience.
- Power and memory efficiency
  - Challenge: in-efficient applications
  - Impact : Increased costs (scaling and cooling)

How do we overcome these challenges ?

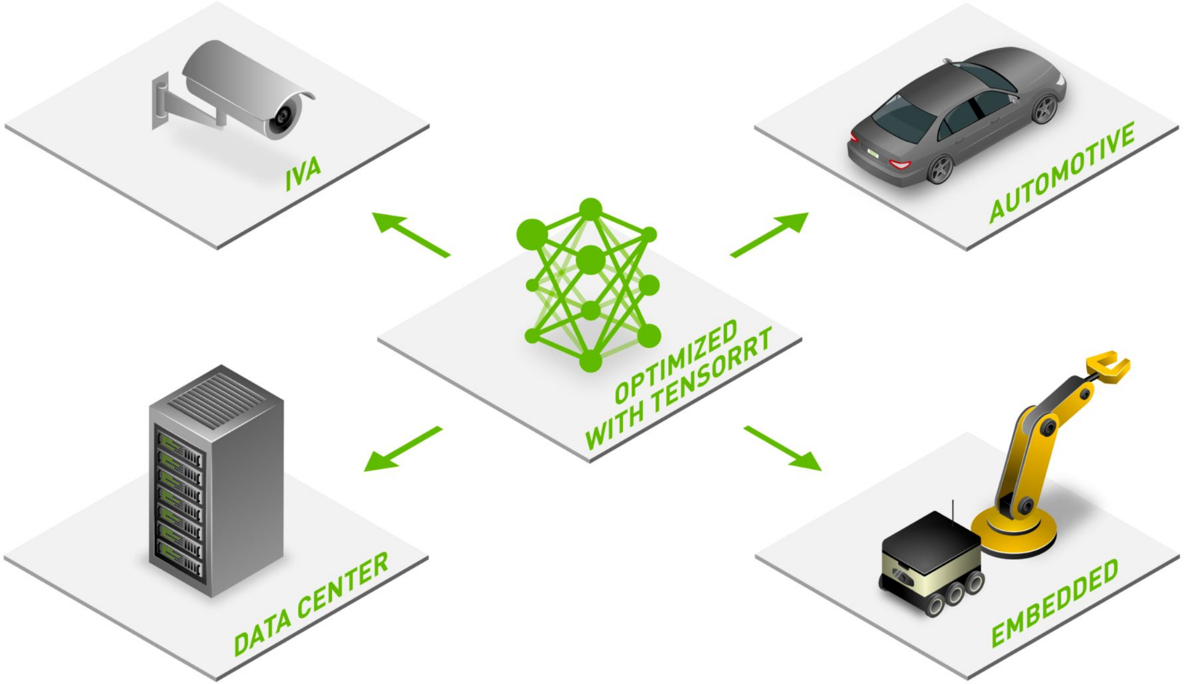
Nvidia-TensorRT to rescue !!

Nvidia-TensorRT

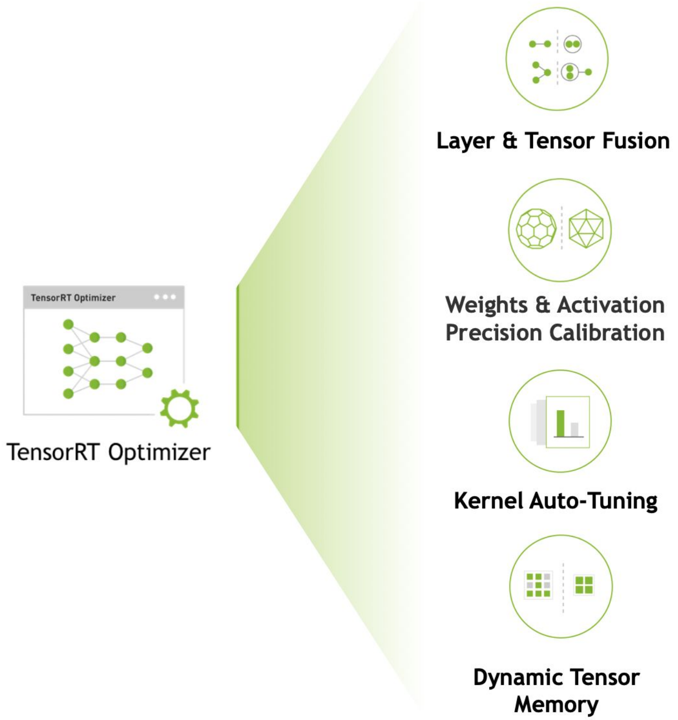
SDK for high-performance deep learning inference, includes a deep learning inference optimizer and runtime that delivers low latency and high throughput for inference applications



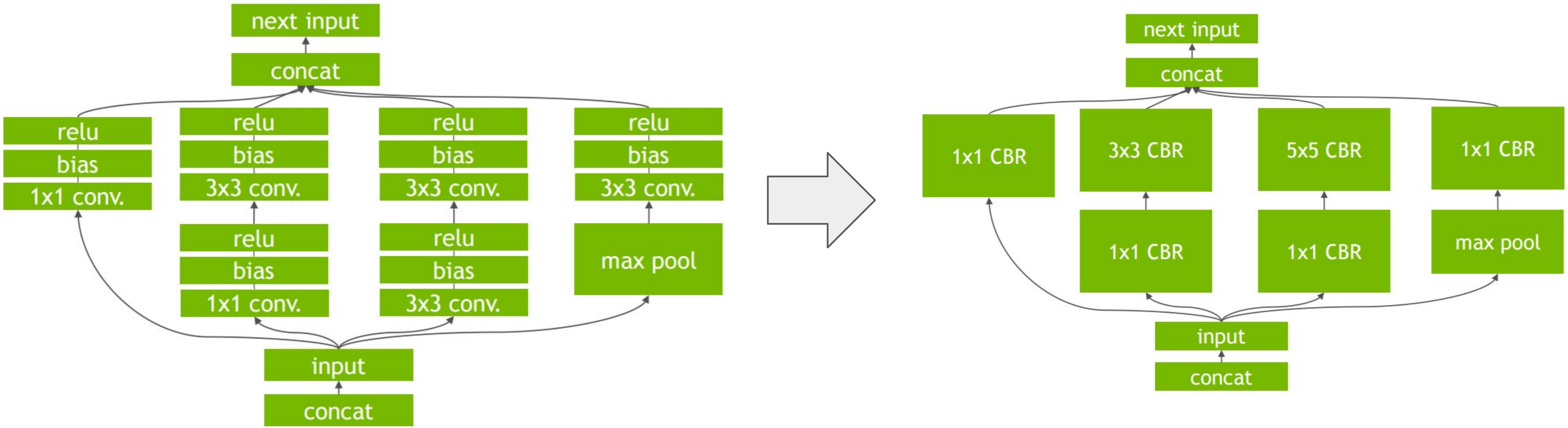
Accelerates every inference platform



TensorRT Optimizations



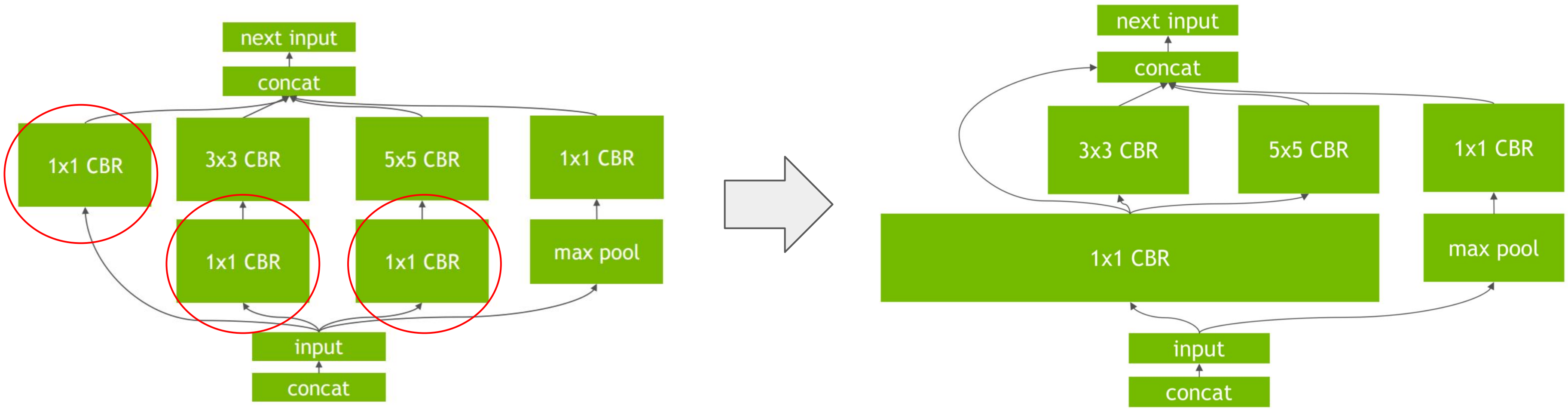
Vertical layer fusion



- Multiple function calls for each layer.
- Each operation is performed on GPU -> multiple CUDA kernel launches.
- Multiple kernel launch overhead.

- TensorRT vertically fuses layers to perform the sequential operations together.
- Layer fusion reduces kernel launches and avoids writing into and reading from memory between layers.

Horizontal Layer Fusion



- TensorRT recognizes layers that share the same input data and filter size, but have different weights.
- Instead of three separate kernels, TensorRT fuses them horizontally into a single wider kernel as shown for the 1×1 CBR layer.

Overall result : layer fusion

- Smaller, faster and more efficient graph.
- Fewer layers -> reduced kernel launches.
- Reduced inference time !

Network	Layers	Layers after fusion
VGG19	43	27
Inception V3	309	113
ResNet-152	670	159

TensorRT’s graph optimization for some common image classification networks.

Precision calibration

- Most deep learning frameworks train neural networks in full 32-bit precision (FP32).
- Inference computations can use half precision FP16 or even INT8 tensor operations (Since backpropagation is not required during inference)
- Lower precision results in
    - Smaller model size
    - Lower memory utilization
    - Lower latency
    - High throughput

Precision calibration ..

- Tensorflow-TensorRT: *precision_mode=trt.TrtPrecisionMode.< FP32 or FP16 or INT8 >*

	FP32 Top 1	INT8 Top 1	Difference
Googlenet	68.87%	68.49%	0.38%
VGG	68.56%	68.45%	0.11%
Resnet-50	73.11%	72.54%	0.57%
Resnet-152	75.18%	74.56%	0.61%

Minimal difference in Top 1 accuracy post precision calibration for some common image classification networks.

Other optimizations

- Kernel Auto-tuning
  - TensorRT picks implementation from a library of kernels that delivers best performance based on target GPU, input data size, filter size, tensor layout, batch size, etc ..
  - Ensures that the deployed model is performance tuned for the specific deployment platform and neural network.
- Dynamic Tensor Memory
  - TensorRT reduces memory footprint and improves memory reuse by designating memory for each tensor only for the duration of its usage.

Tensorflow-TensorRT Code example

SCAN ME

https://github.com/rajesh-bhat/faster_inference_tensorflow_tensorrt

Key Takeaways

- Generate optimized, deployment-ready runtime engines for low latency inference with Nvidia-TensorRT.

- TensorRT optimizations : fully automatic with very few lines of code
    - High throughput
    - Low response time
    - Memory and power efficient
    - Reduced cost !!

References

- “NVIDIA TensorRT.” NVIDIA Developer
- “TensorRT 3: Faster TensorFlow Inference and Volta Support.” *NVIDIA Technical Blog*
- “Deep learning deployment with Nvidia-TensorRT”
- “Optimize TensorFlow Models For Deployment with TensorRT.” *Coursera*

Q & A

rajeshshreedhar

rajesh_s_bhat