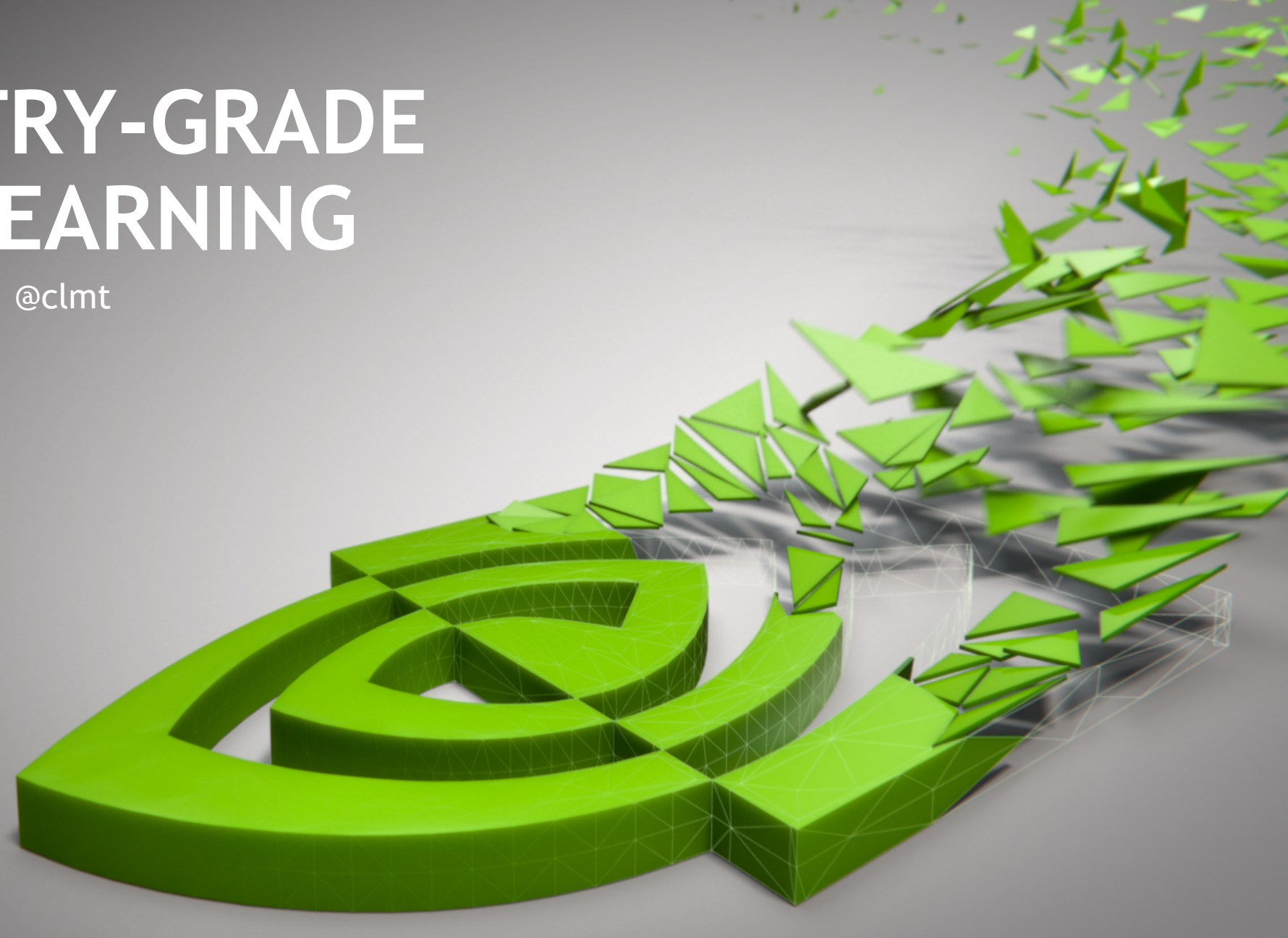
INDUSTRY-GRADE DEEP LEARNING

Clement Farabet | @clmt

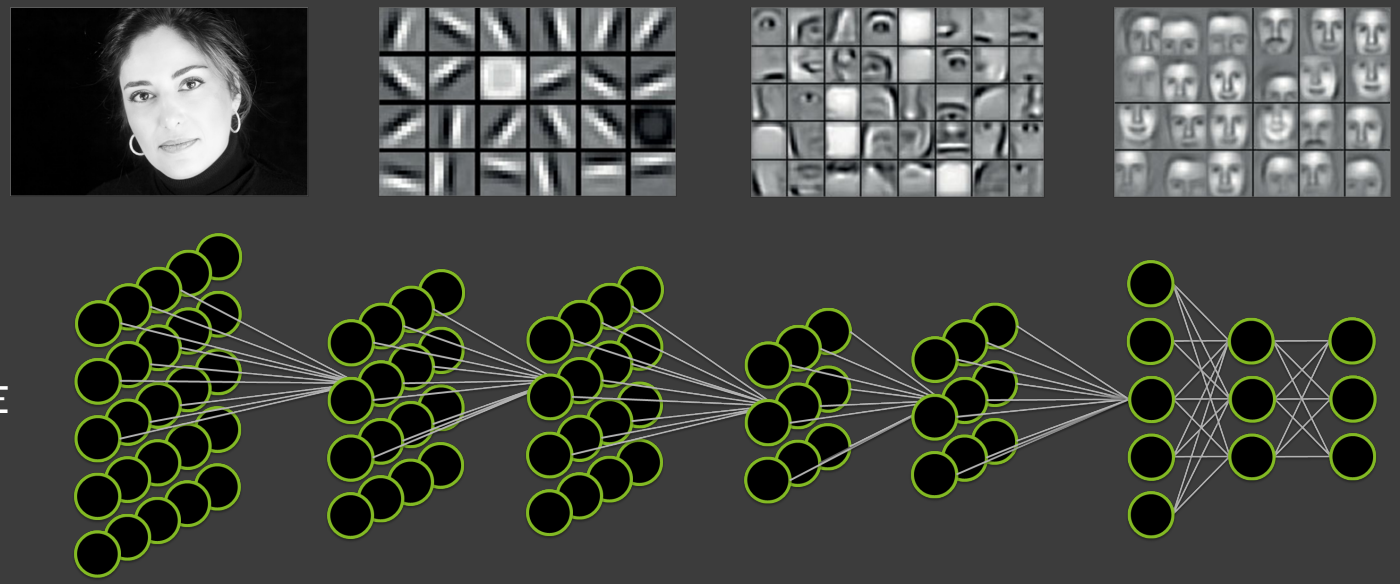
NVIDIA



Deep Learning has changed the way we think about developing software

WHERE IT ALL STARTED

The magic algorithm from the 80s

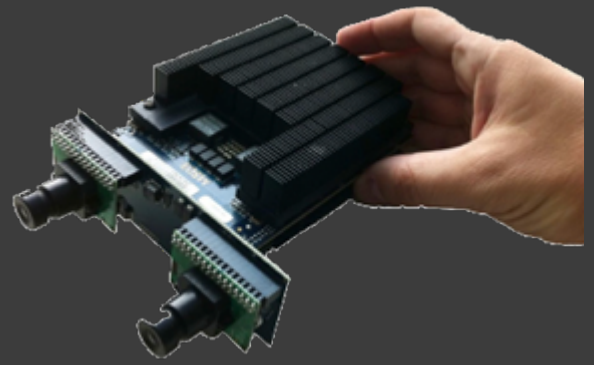


"Neural nets are universal approximators, convolutions a good prior for spatial data… there seems to be no limit to what these CNNs can learn!"

— Someone in Yann LeCun's lab, circa 2010

2010

The excitement was there... the pain as well!



Real-time (10fps) semantic segmentation

On a custom FPGA circuit (neuflow)

Today, we have > 100x that compute power, in 1 chip

bit.ly/clmt-research

2011, 2012, 2013…

Dreams became reality

Compute power finally enough to train DNNs at scale

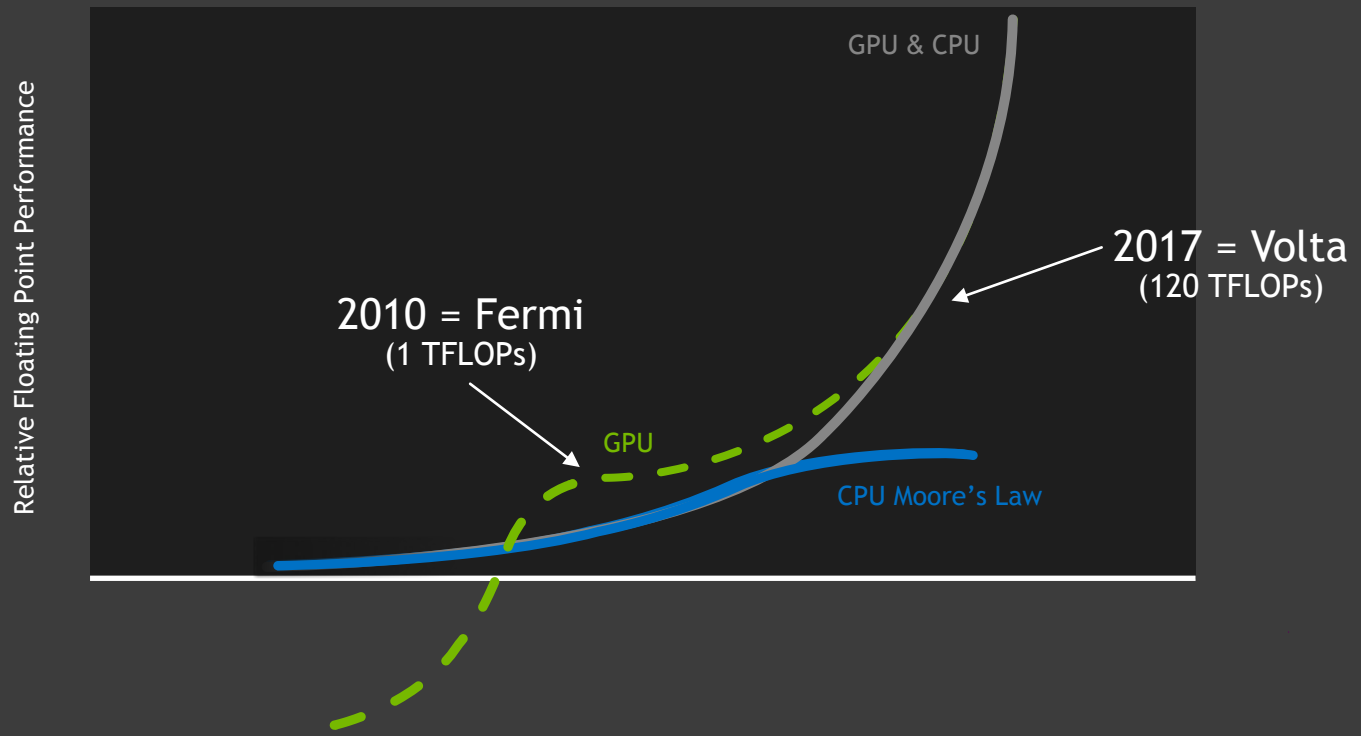
Simplified DNN architectures (ReLU) and smart regularization tricks (Dropout)

Research results exploded, beating several tough perception challenge benchmarks

Timescales started to change…

NVIDIA GPUS POWERED DL SINCE 2009

Turns out… DL, much like games, relies on a small set of ops



bit.ly/nv-moore

2017

Deep Learning is now (almost) a commodity

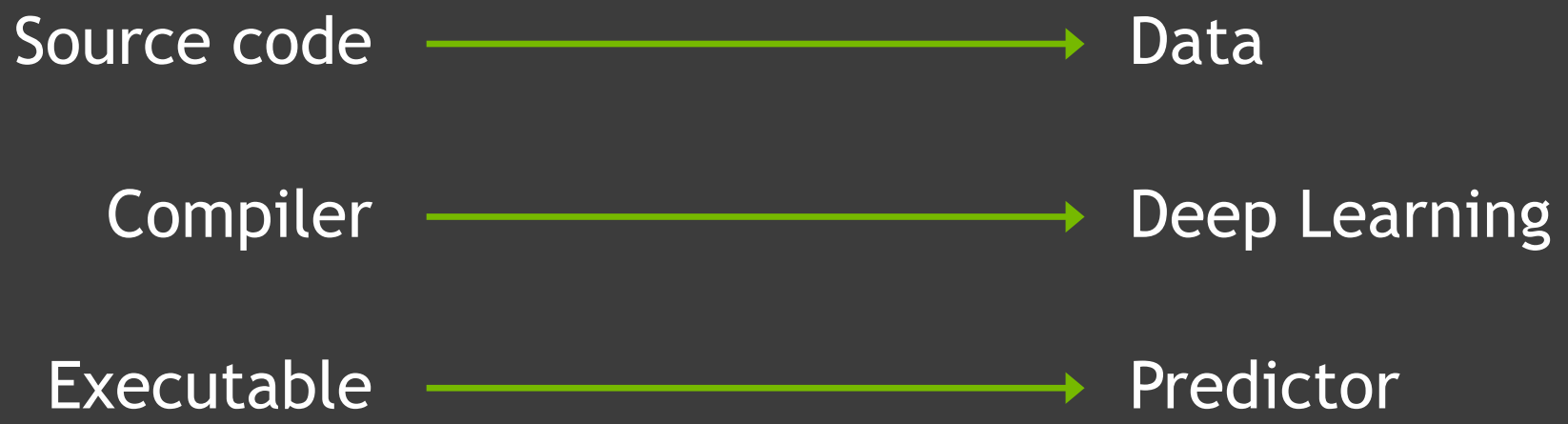
Used as a building block to improve many baseline systems, for ads, content recommendation, image/video search, speech recognition, autonomous vehicles…

Results quickly transitioning from research to production

Many self-taught engineers are now using Deep Learning frameworks and methods to add value to their products

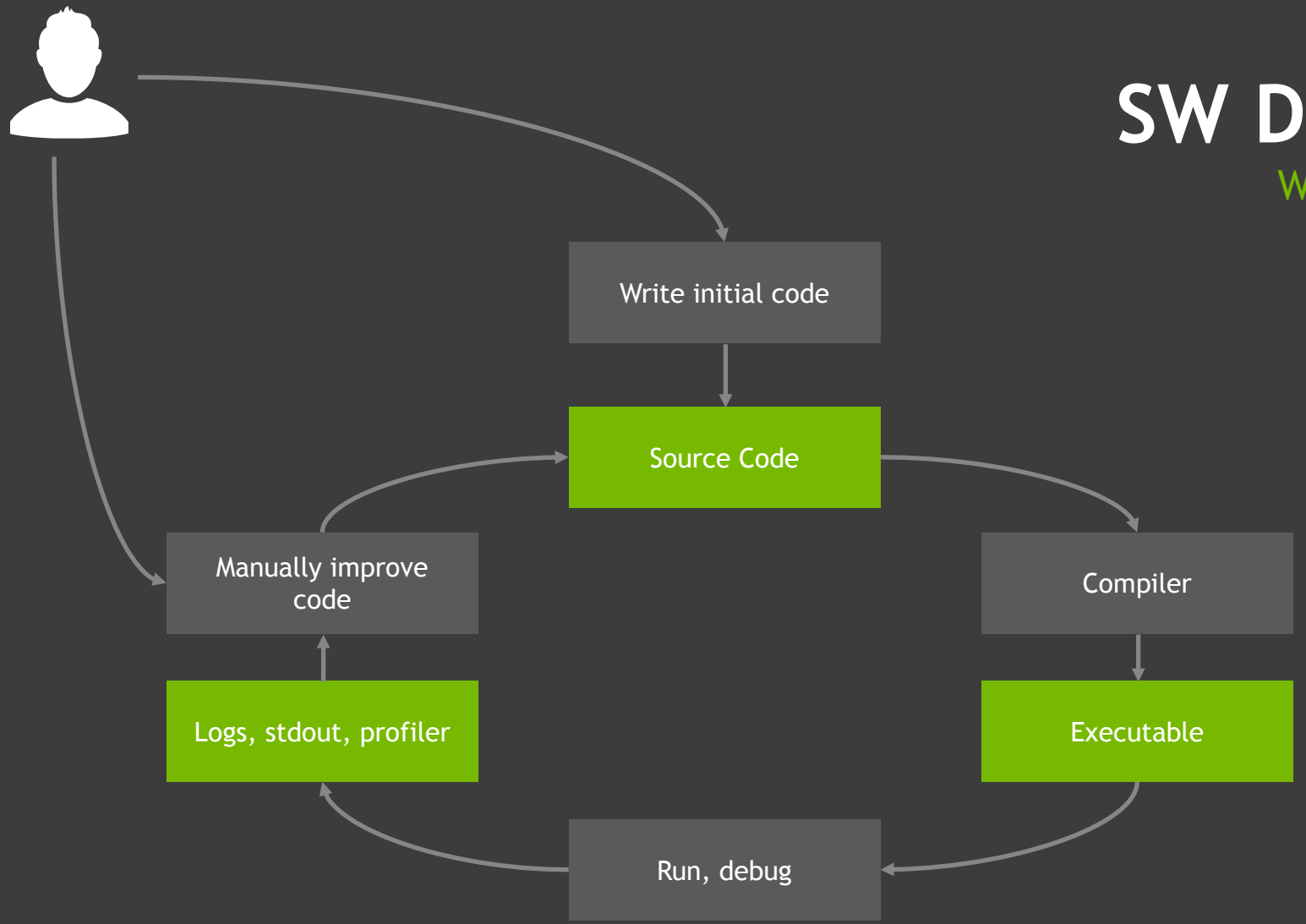
Deep Learning has changed the way we think about developing software…

… we need to *actually* change the way we develop software!


Source code
Data
Compiler
Deep Learning
Executable
Predictor

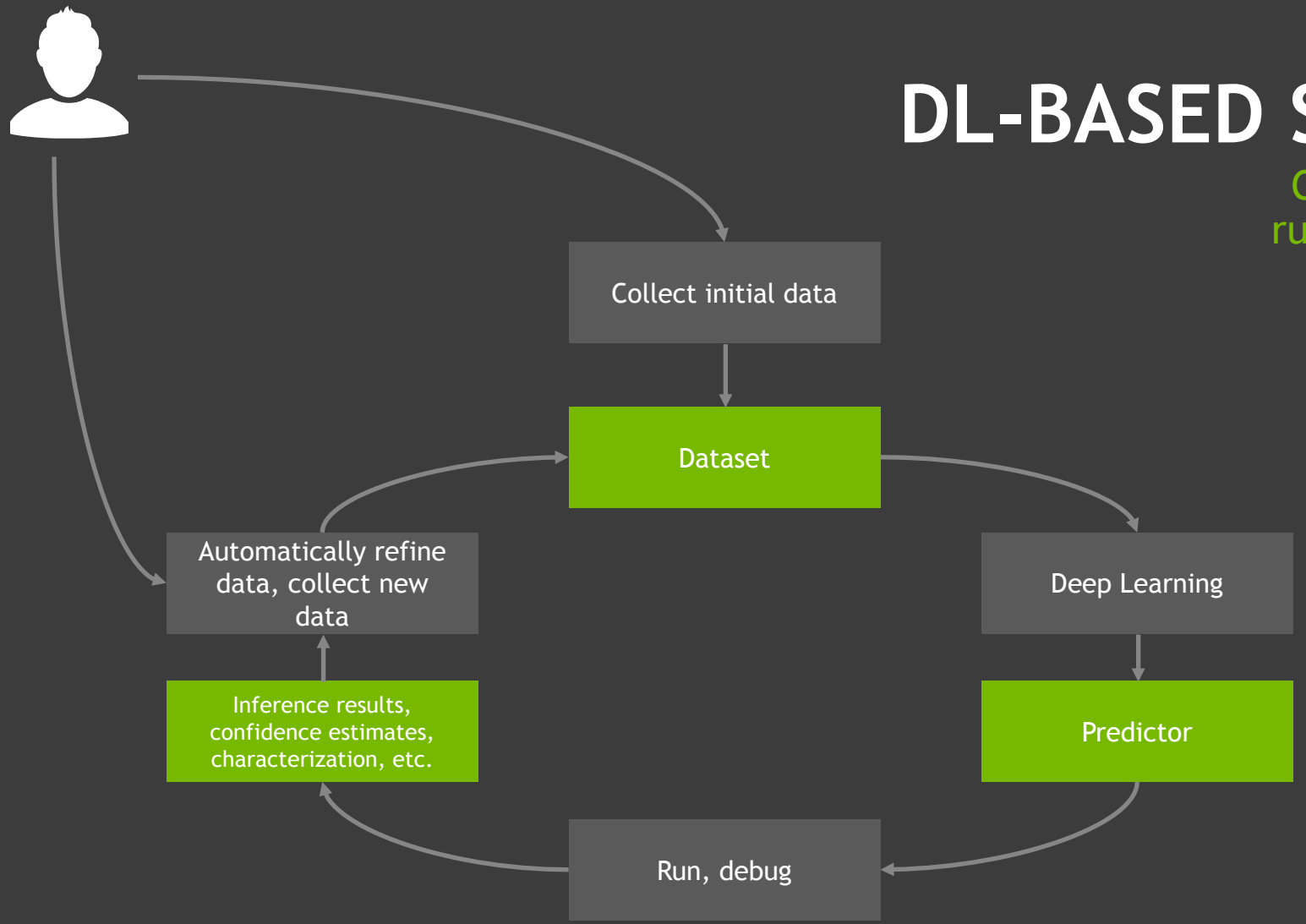
SW DEV PROCESS

Write code, compile, test, debug, repeat…



DL-BASED SW PROCESS

Collect initial data, train, run/debug, mine new data



DL-BASED SOFTWARE PROCESS

Building industry-grade predictors involves 2 phases

Repeat:

1. Given a fixed dataset, find the best predictor
2. Given a fixed predictor, find the next best data

   … until predictor's desired performance is met
3. Deploy!

DL-BASED SOFTWARE PROCESS

Methods & tools required

1. Given a fixed dataset, find the best predictor
    a) Explore best architectures/models (meta-optimize) — large-scale map jobs
    b) Fit one model give meta-params — multi-GPU/node, high-bandwidth job
2. Given a fixed predictor, find the next best data
    a) Mine/analyze raw, unlabeled data — large-scale inference jobs
    b) Store datasets, models, experiment data — asset/version management system
3. Deploy
    a) Convert end-to-end inference pipeline — exporter/converter/compiler to target env

DL FOR CARS

Data Scale, Training scale, etc.

Safety is critical, reproducibility, interpretability

Data scale is … ridiculous, dealing with raw data, 10+ sensors per car, geo-diversity

Many orthogonal loss functions (DNNs) to detect all sorts of things

Training time massive

End-to-end workflow [collect new data, label, re-train, re-validate, deploy] needs to be sped up as much as possible

DL FOR CARS

Assumptions, scale

Data Collection fleet == 100 cars

2000h of data collected per car, per year

Assuming 5 2MP cameras per car, radar data, etc. => 1 TB / h / car

Grand total of 200 PB collected per year!

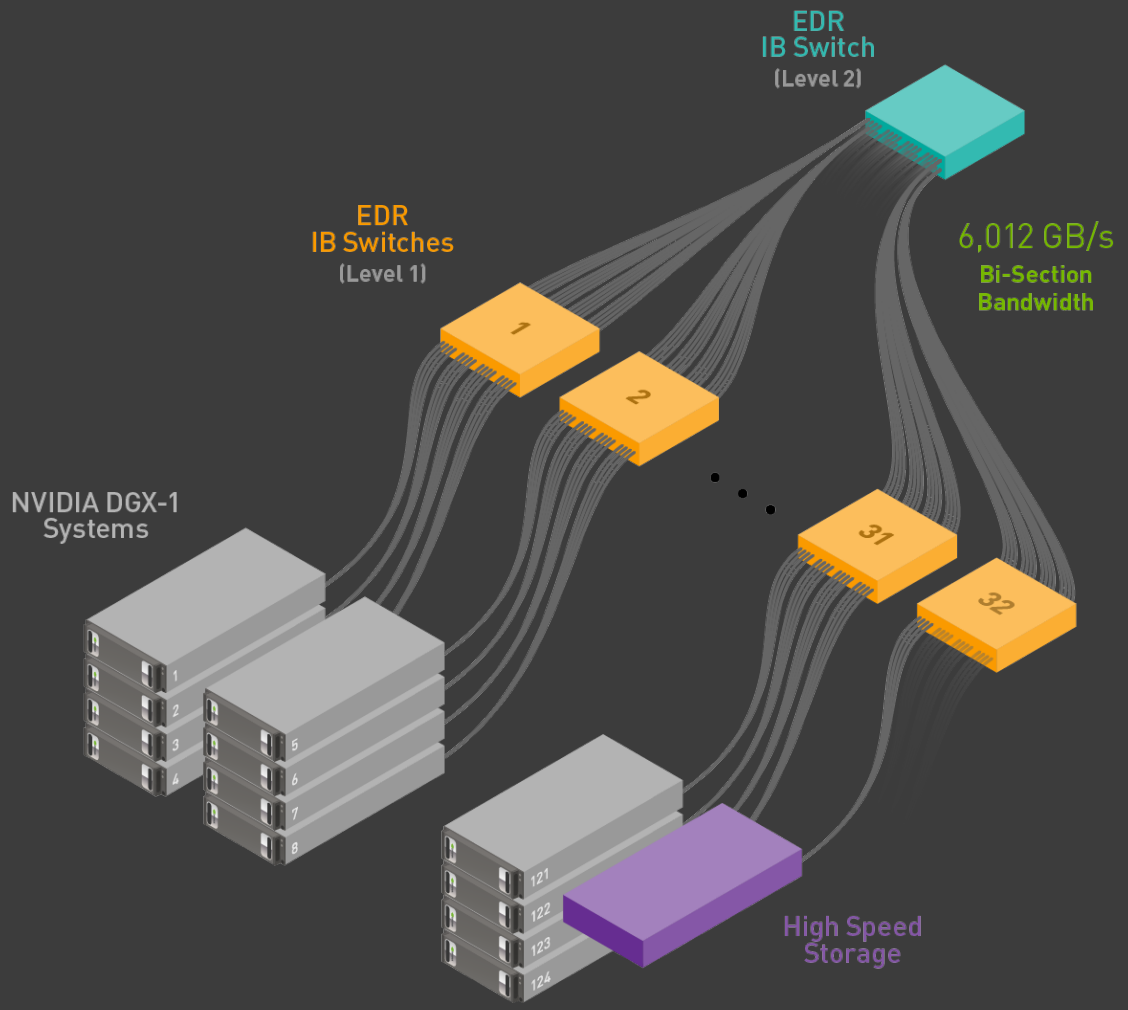
Only 1/1000 likely to be used for training (curated, labeled data)

12.1 years training a ResNet50-like network on Pascal, 1.5 years on DGX1 w/ Volta

Today, with 8 DGX1s, and 1/10th of that training data, can train in 1 week

DL FOR CARS

Infrastructure



960 TFLOPs per DGX1 (FP16)

7TB SSD per DGX1

High-speed external storage (multi-PB)

Infiniband as interconnect

NCCL 2.0

Data+model management

DL FOR CARS

Workflow

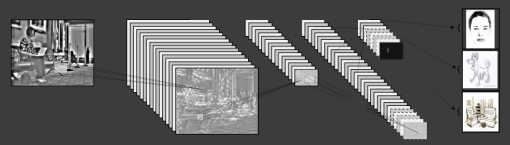
	Build the Model	Continuous Integration	Train the model on real data (hyperparameter tuning)	Optimize and Validate the Model	Deploy the Model
Goal	Build a promising model	Make sure that the code base remains bug free	Make the model work with real data and optimise it	Prepare the model for serving and validate it	Provide functionality using the model
Iteration Time	Hours	Hours	Days - Weeks	Hours - Weeks	Milliseconds
# of Machines	1	10s	10s – 100s	10s	Hundreds (test fleet) Millions (live fleet)
GPU	2-4 TitanX / Tesla P/V 100	4-8 Tesla P/V100	4-8 Tesla P/V100	4-8 Tesla P/V100	Xavier

DL FOR CARS

PBs of data, large-scale labeling, large-scale training, etc.

POST /datasets/{id}

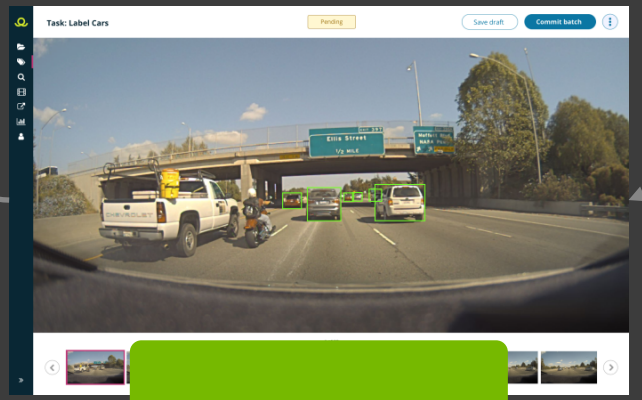
Inference optimized DNN (TensorRT)



Datasets

Labels

Manually selected data



Labeling

Deep Learning

Train/test data

Metrics

Simulation, verification results

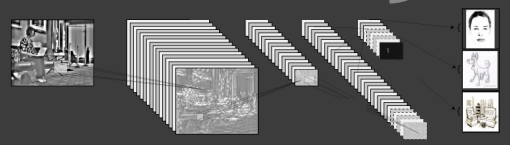
DL FOR CARS

Active learning strategies to meet business needs

POST /datasets/{id}

Trained Models

Inference optimized DNN (TensorRT)



Deep Learning

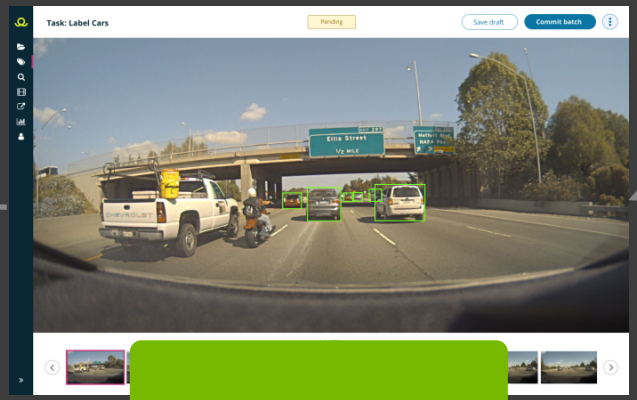
Train/test data

Datasets

Labels

Mine highly confused / most informative data

Intelligently selected data



Labeling


Total AD: 0.00 miles (Lat), 0.00 miles (Long)
MAD: 0.00 miles (Lat), 0.00 miles (Long)
Comfort: 0.00 % (Lat), 0.00 % (Long)
FPS: 1.56, Latency: 640.18 ms (avg), 969.00 ms (max)
Acc: on
LKS: off
268
28.8 m
-1.61 s
1.0 deg

THANK YOU

twitter.com/clmt