Lecture 10:

Raising the level of abstraction for ML

Parallel Computing

Stanford CS348K, Spring 2021

Note

- Most of this class involved in-class discussion of the Ludwig and Overton papers

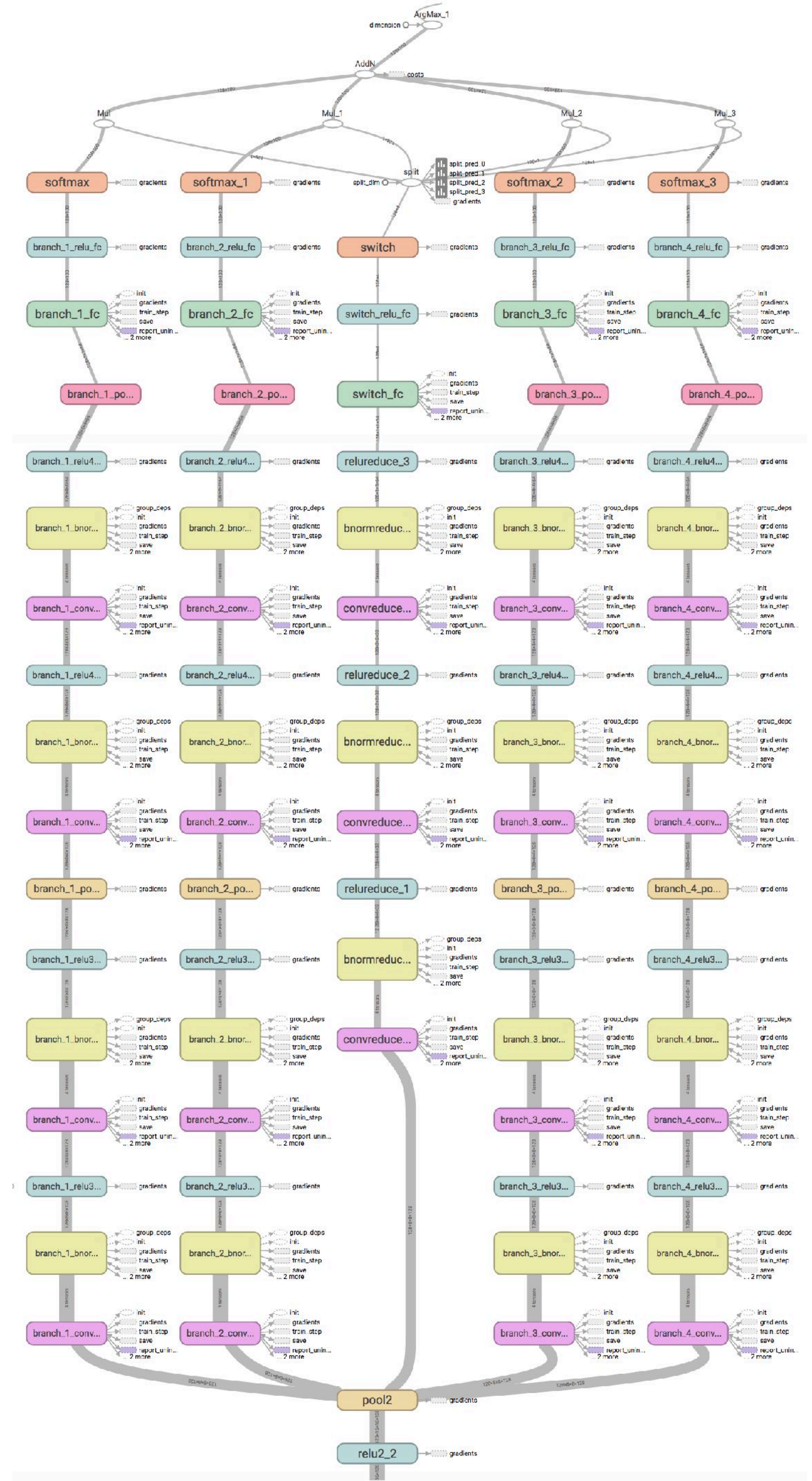
- I am posting these slides as some were used during parts of the discussion

Services provided by ML "frameworks"

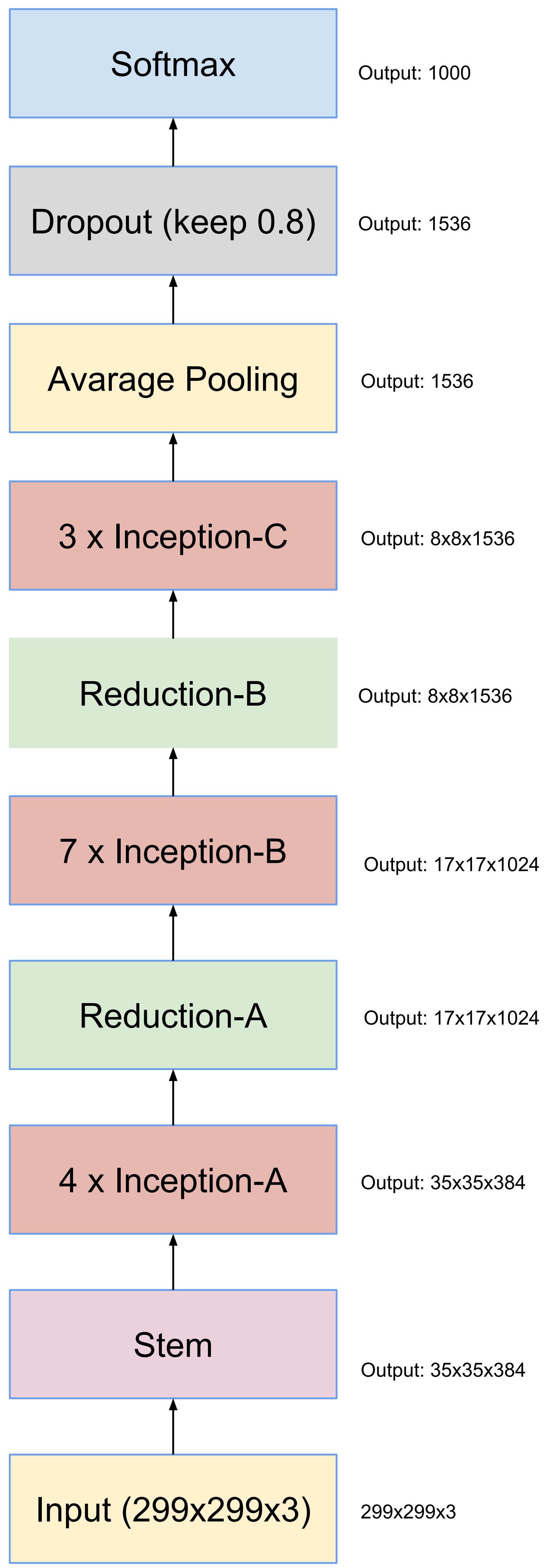
- Functionality:
  - Implementations of wide range of useful operators
    - Conv, dilated conv, relu, softmax, pooling, separable conv, etc.
    - Implementations of various optimizers:
      - Basic SGD, with momentum, Adagrad, etc.
  - Ability to compose operators into large graphs to create models
  - Carry out back-propagation
- Performance:
  - High performance implementation of operators (layer types)
  - Scheduling onto multiple GPUs, parallel CPUs (and sometimes multiple machines)
  - Automatic sparsification and pruning
- Meta-optimization:
  - Hyper-parameter search
  - More recently: neural architecture search

TensorFlow/MX.Net data-flow graphs

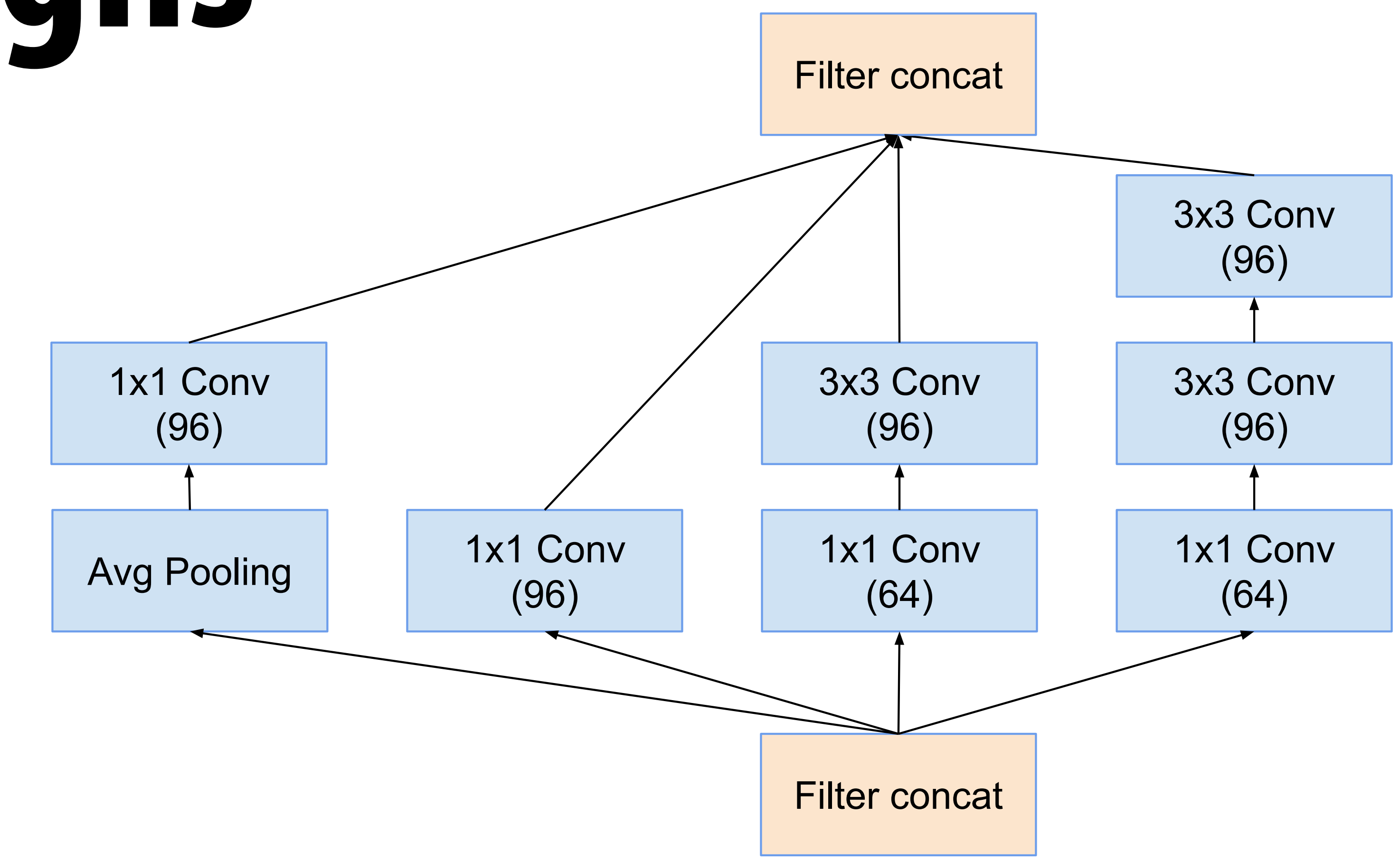
- **Key abstraction: a program is a DAG of (large granularity) operations that consume and product N-D tensors**



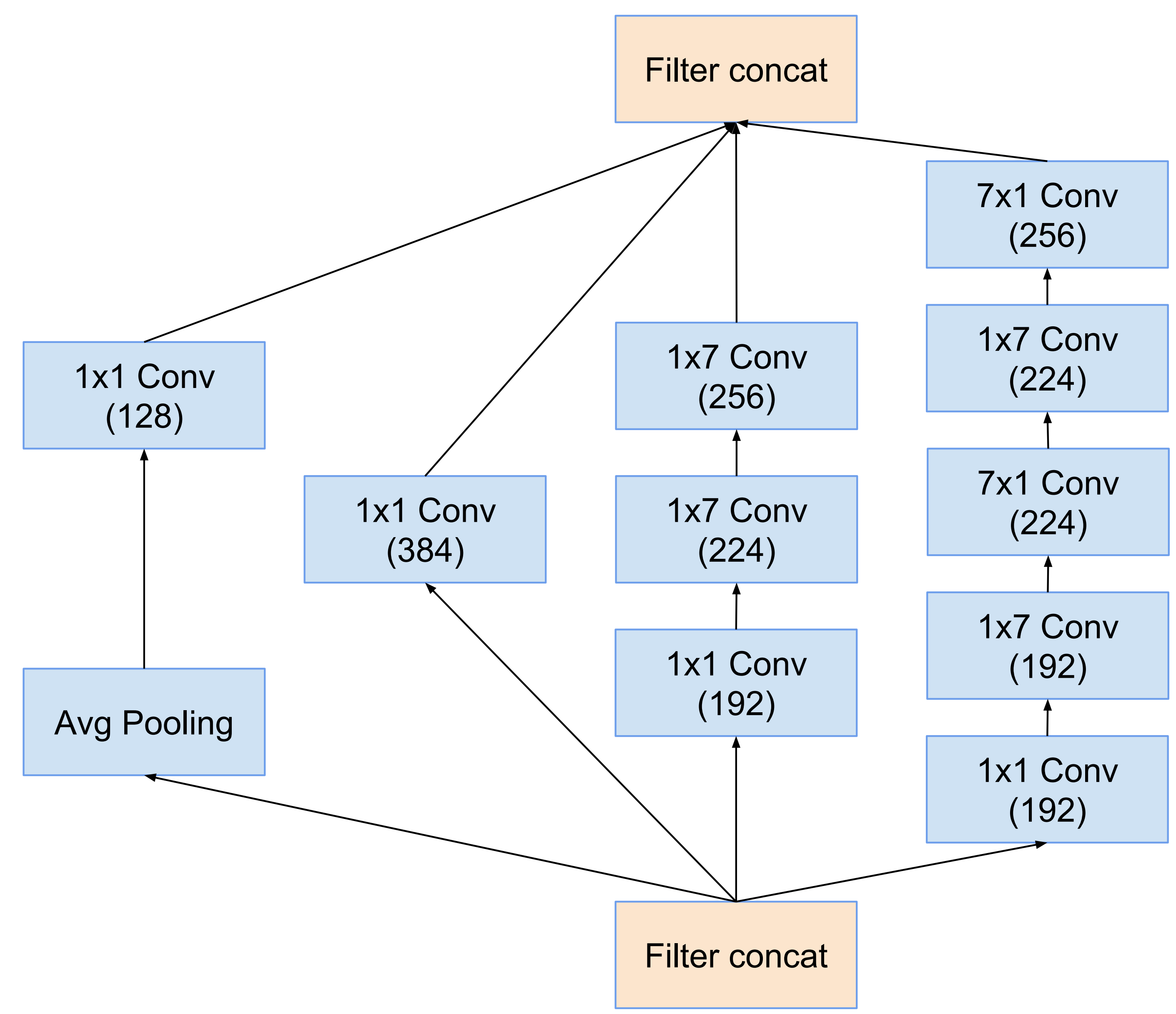
Modular network designs



Inception v4

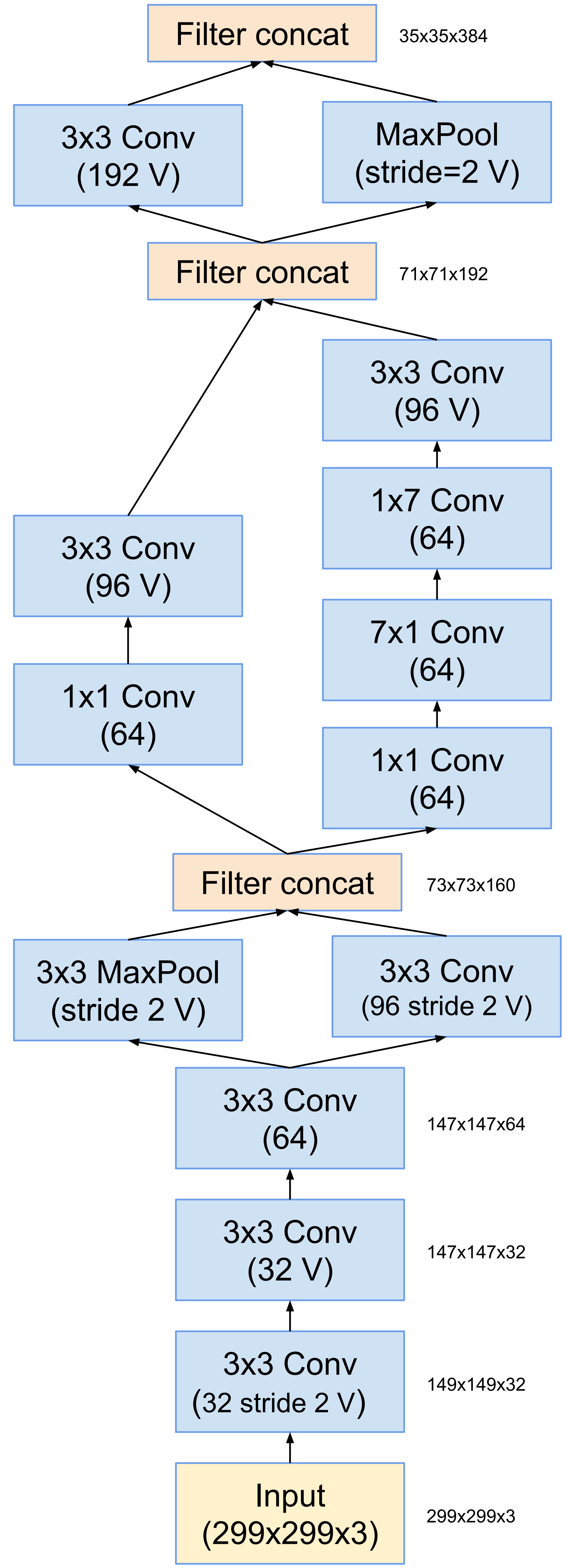


A block

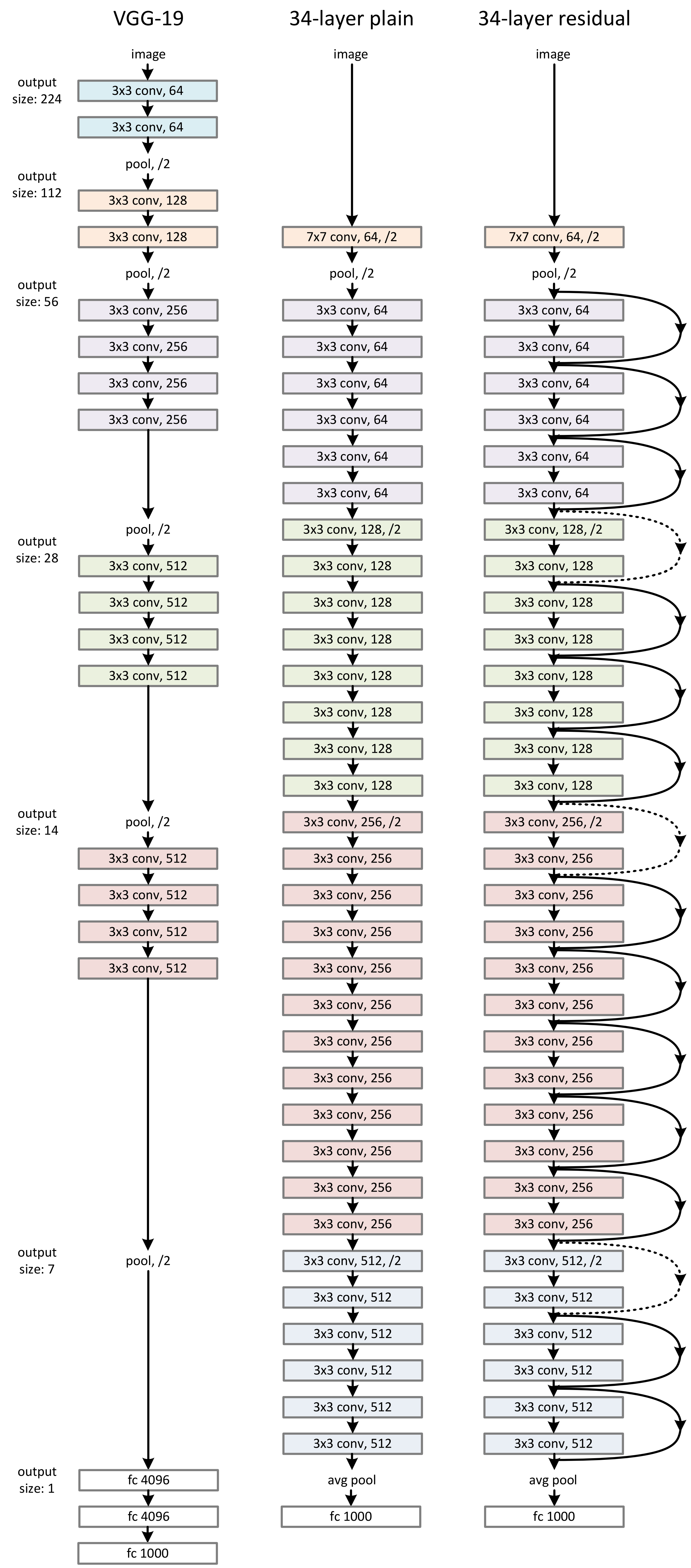


B block

Inception stem



ResNet



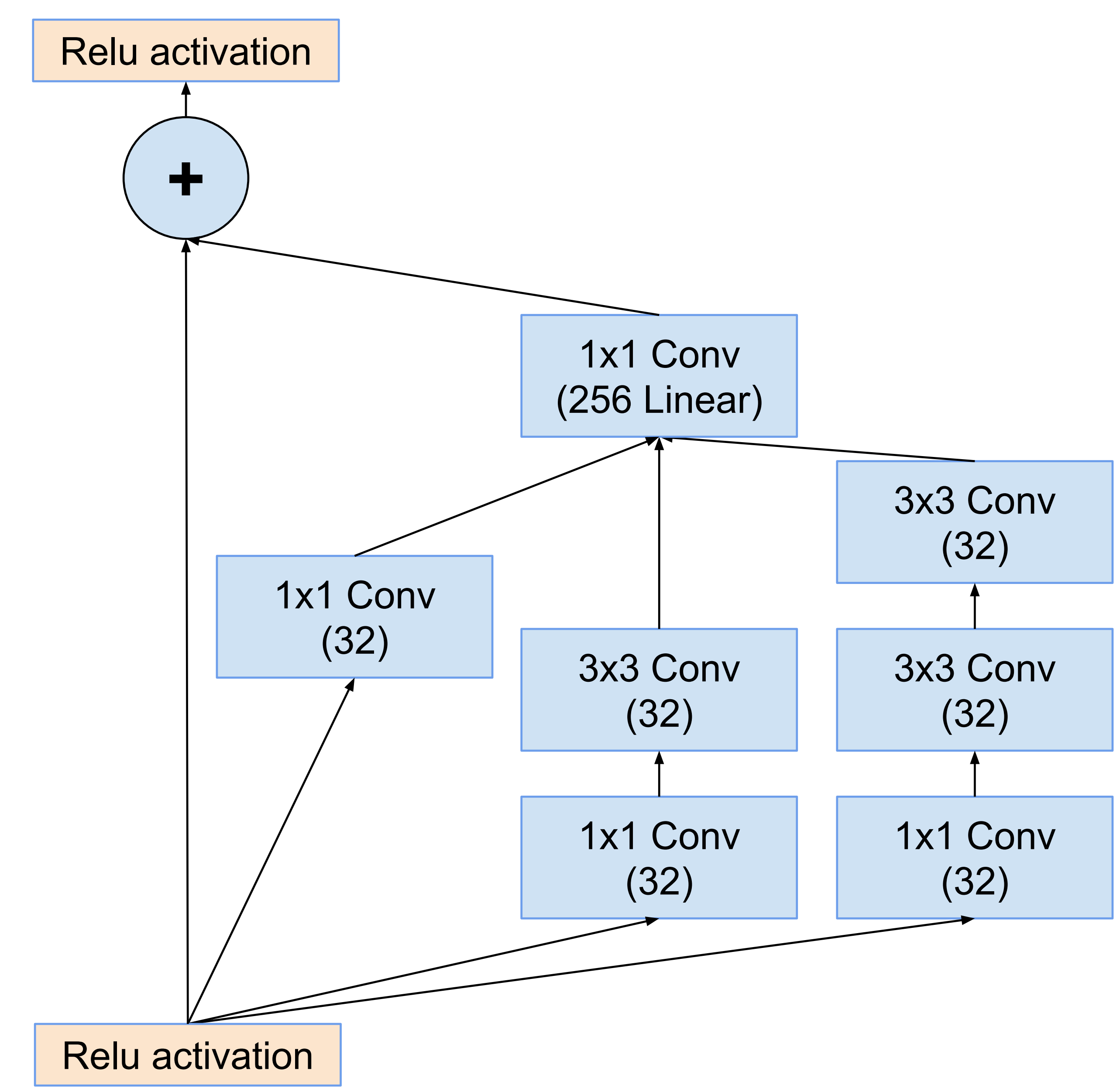


Figure 10. The schema for 35 × 35 grid (Inception-ResNet-A) module of Inception-ResNet-v1 network.

How to improve system support for ML?

Hardware/software for…
faster inference?
faster training?

Compilers for fusing layers, performing code optimizations?

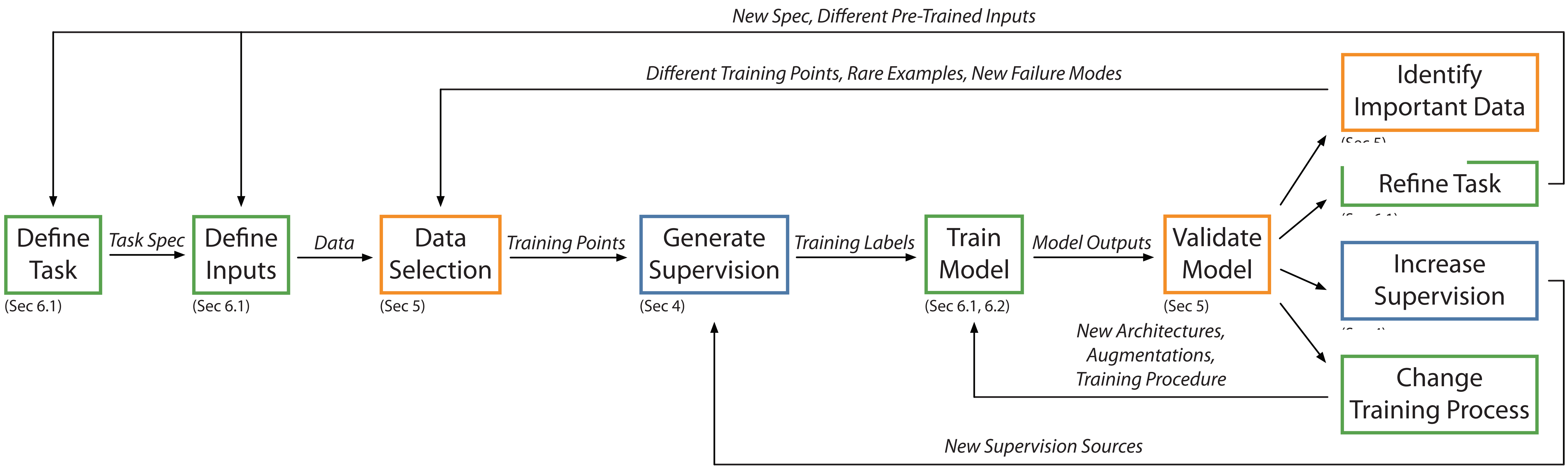
Mon Mar 02, 2020	
Time	**Ballroom A**
07:00 AM (Breaks)	
07:45 AM (Breaks)	Opening Remarks
08:00 AM (Orals)	Distributed and Parallel Learning Algorithms A System for Massively Parallel Hyperparameter Tuning
08:25 AM (Orals)	PLink: Discovering and Exploiting Locality for Accelerated Distributed Training on the public Cloud
08:50 AM (Orals)	Federated Optimization in Heterogeneous Networks
09:15 AM (Orals)	BPPSA: Scaling Back-propagation by Parallel Scan Algorithm
09:40 AM (Orals)	Distributed Hierarchical GPU Parameter Server for Massive Scale Deep Learning Ads Systems
10:30 AM (Orals)	Efficient Model Training Resource Elasticity in Distributed Deep Learning
10:55 AM (Orals)	SLIDE : In Defense of Smart Algorithms over Hardware Acceleration for Large-Scale Deep Learning Systems
11:20 AM (Orals)	FLEET: Flexible Efficient Ensemble Training for Heterogeneous Deep Neural Networks
11:45 AM (Orals)	Breaking the Memory Wall with Optimal Tensor Rematerialization
01:30 PM (Invited Talks)	Theory and Systems for Weak Supervision
02:30 PM (Orals)	Efficient Inference and Model Serving What is the State of Neural Network Pruning?
02:55 PM (Orals)	SkyNet: a Hardware-Efficient Method for Object Detection and Tracking on Embedded Systems
03:20 PM (Orals)	MNN: A Universal and Efficient Inference Engine
03:45 PM (Orals)	Willump: A Statistically-Aware End-to-end Optimizer for Machine Learning Inference
04:30 PM (Orals)	Model / Data Quality and Privacy Attention-based Learning for Missing Data Imputation in HoloClean
04:55 PM (Orals)	Privacy-Preserving Bandits
05:20 PM (Orals)	Understanding the Downstream Instability of Word Embeddings
05:45 PM (Orals)	Model Assertions for Monitoring and Improving ML Models
06:00 PM (Demonstrations)	

Tue Mar 03, 2020	
Time	**Ballroom A**
07:00 AM (Breaks)	
08:00 AM (Orals)	ML programming models and abstractions & ML applied to systems AutoPhase: Juggling HLS Phase Orderings in Random Forests with Deep Reinforcement Learning
08:25 AM (Orals)	Automatically batching control-intensive programs for modern accelerators
08:50 AM (Orals)	Predictive Precompute with Recurrent Neural Networks
09:15 AM (Orals)	Sense & Sensitivities: The Path to General-Purpose Algorithmic Differentiation
09:40 AM (Orals)	Ordering Chaos: Memory-Aware Scheduling of Irregularly Wired Neural Networks for Edge Devices
10:30 AM (Orals)	Efficient inference and model serving Fine-Grained GPU Sharing Primitives for Deep Learning Applications
10:55 AM (Orals)	Improving the Accuracy, Scalability, and Performance of Graph Neural Networks with Roc
11:20 AM (Orals)	OPTIMUS: OPTImized matrix MUltiplication Structure for Transformer neural network accelerator
11:45 AM (Orals)	PoET-BiN: Power Efficient Tiny Binary Neurons
01:30 PM (Invited Talks)	The Emerging Role of Cryptography in Trustworthy AI
02:30 PM (Orals)	Quantization of deep neural networks Memory-Driven Mixed Low Precision Quantization for Enabling Deep Network Inference on Microcontrollers
02:55 PM (Orals)	Trained Quantization Thresholds for Accurate and Efficient Fixed-Point Inference of Deep Neural Networks
03:20 PM (Orals)	Riptide: Fast End-to-End Binarized Neural Networks
03:45 PM (Orals)	Searching for Winograd-aware Quantized Networks
04:30 PM (Orals)	Efficient Model Training 2 Blink: Fast and Generic Collectives for Distributed ML
04:55 PM (Orals)	A Systematic Methodology for Analysis of Deep Learning Hardware and Software Platforms
05:20 PM (Orals)	MotherNets: Rapid Deep Ensemble Learning
05:45 PM (Orals)	MLPerf Training Benchmark

List of papers at MLSys 2020 Conference

But as a user wanting to create a model, where does most of my time *really* go?

ML model development is an iterative process

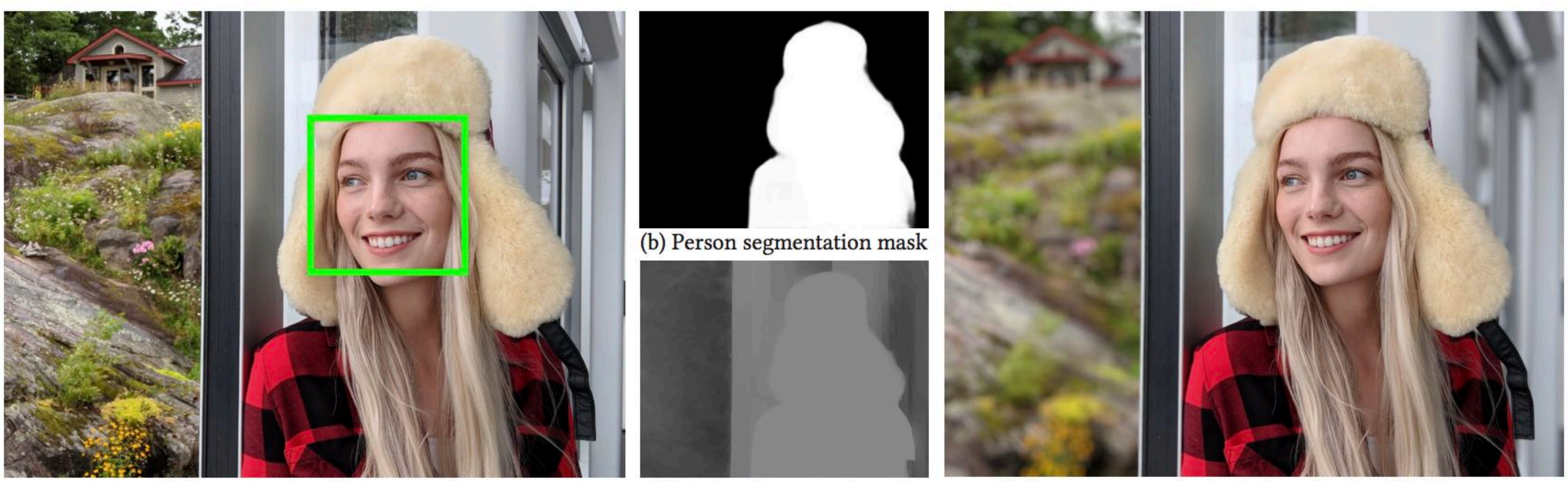


Example: does TensorFlow help with data curation?

"We cannot stress strongly enough the importance of good training data for this segmentation task: choosing a wide enough variety of poses, discarding poor training images, cleaning up inaccurate [ground truth] polygon masks, etc. With each improvement we made over a 9-month period in our training data, we observed the quality of our defocused portraits to improve commensurately."

Synthetic Depth-of-Field with a Single-Camera Mobile Phone

NEAL WADHWA, RAHUL GARG, DAVID E. JACOBS, BRYAN E. FELDMAN, NORI KANAZAWA, ROBERT CARROLL, YAIR MOVSHOVITZ-ATTIAS, JONATHAN T. BARRON, YAEL PRITCH, and MARC LEVOY, Google Research



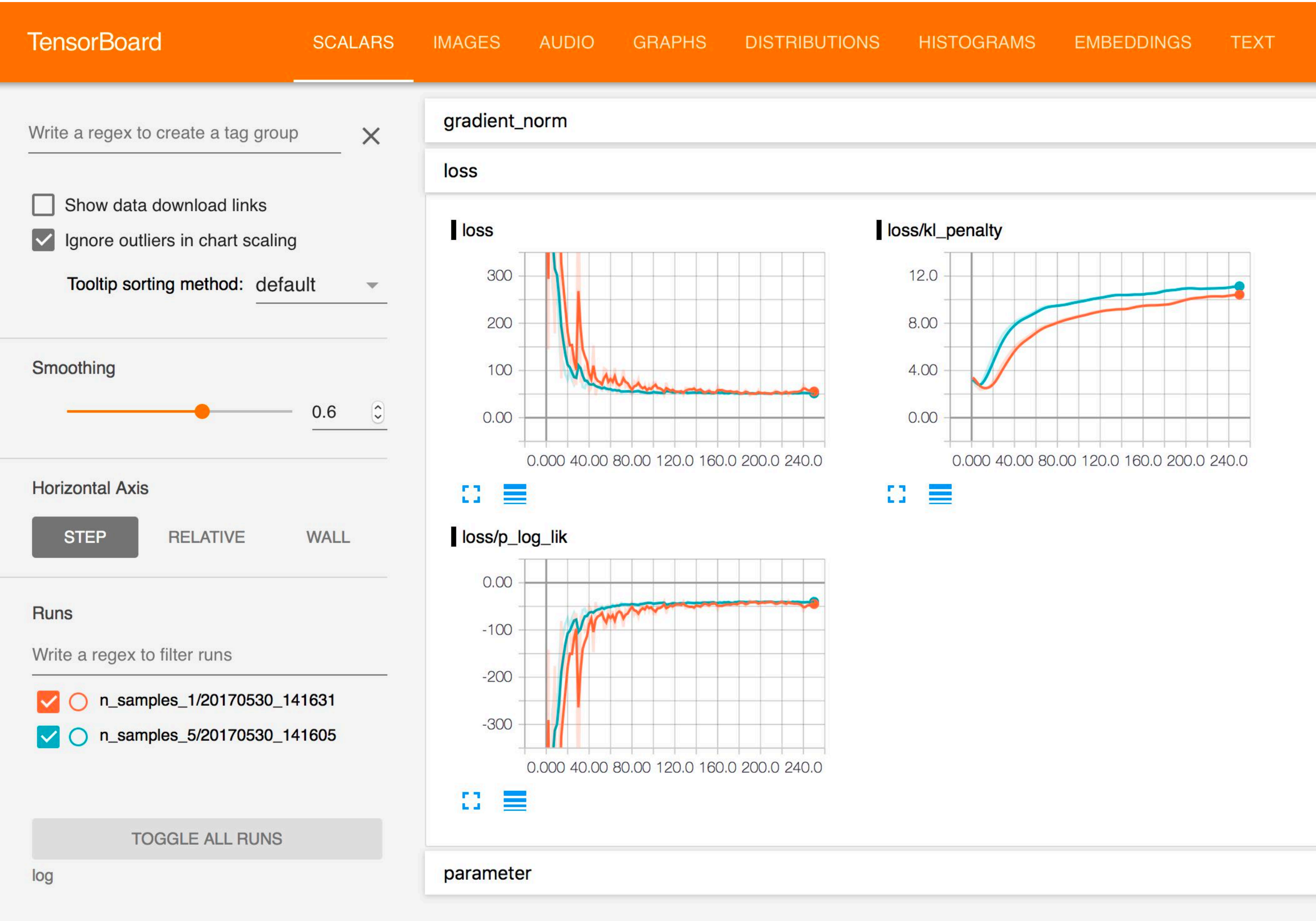
(a) Input image with detected face

(b) Person segmentation mask

(c) Mask + disparity from DP

(d) Our output synthetic shallow depth-of-field image

Thought experiment: I ask you to train a car or person detector for a specific intersection


TensorBoard
SCALARS
IMAGES
AUDIO
GRAPHS
DISTRIBUTIONS
HISTOGRAMS
EMBEDDINGS
TEXT
Write a regex to create a tag group
Show data download links
Ignore outliers in chart scaling
Tooltip sorting method: default
Smoothing
0.6
Horizontal Axis
STEP
RELATIVE
WALL
Runs
Write a regex to filter runs
n_samples_1/20170530_141631
n_samples_5/20170530_141605
TOGGLE ALL RUNS
log
gradient_norm
loss
loss
300
200
100
0.00
0.000 40.00 80.00 120.0 160.0 200.0 240.0
loss/kl_penalty
12.0
8.00
4.00
0.00
0.000 40.00 80.00 120.0 160.0 200.0 240.0
loss/p_log_lik
0.00
-100
-200
-300
0.000 40.00 80.00 120.0 160.0 200.0 240.0
parameter

- **A good system provides valuable services to the user.**
- **So in the Ludwig/Overton papers, who is the "user" (what is their goal, what is their skillset?) and what are the painful, hard, or tedious things that the systems are designed to do for the user?**

- **Let’s specifically contrast the abstractions of Ludwig with that of a lower-level ML system like TensorFlow. TensorFlow/MX.Net/PyTorch largely abstract ML model definition as a DAG of N-Tensor operations. How is Ludwig different?**

- **Then let’s compare those abstractions to Overton.**

- **Comparison to Google's AutoML?**