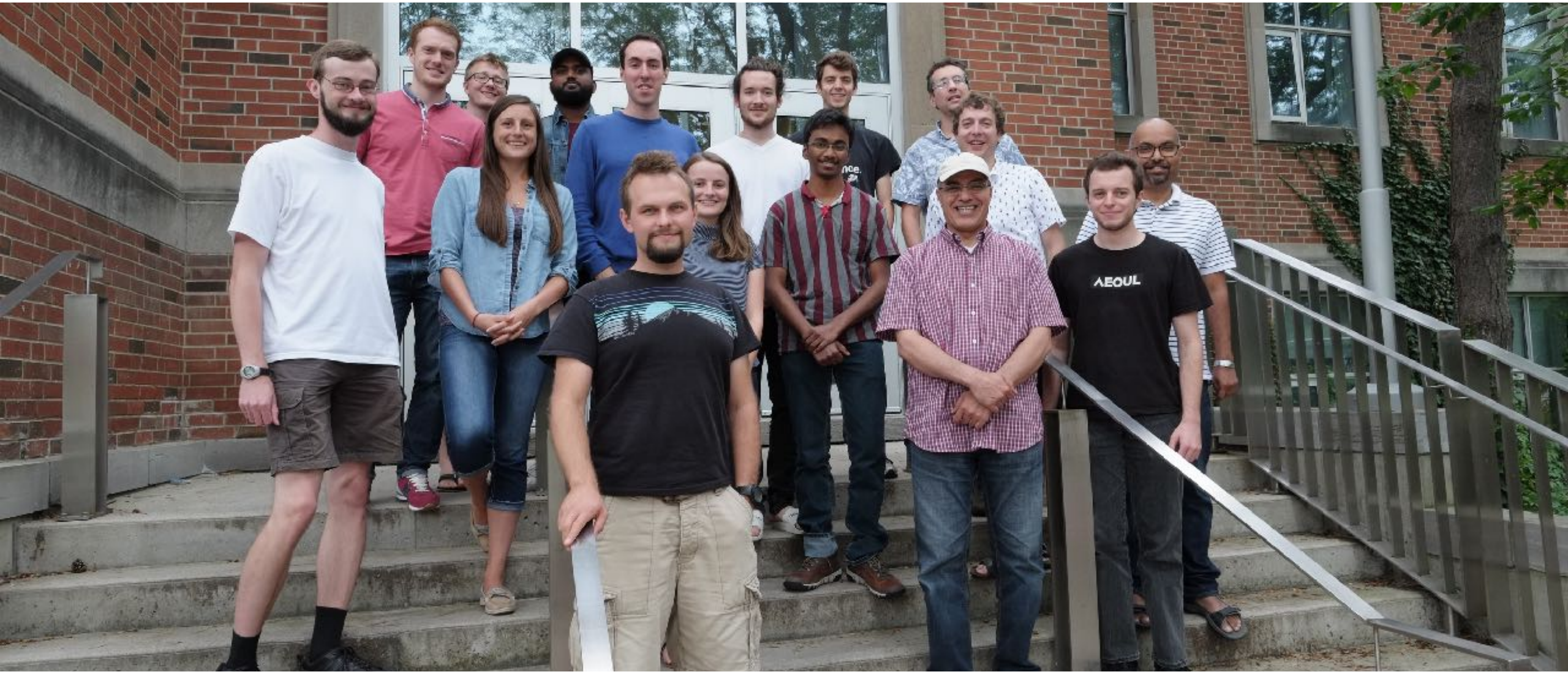
From Design to Search in High-Dimensional Spaces

Graham Taylor

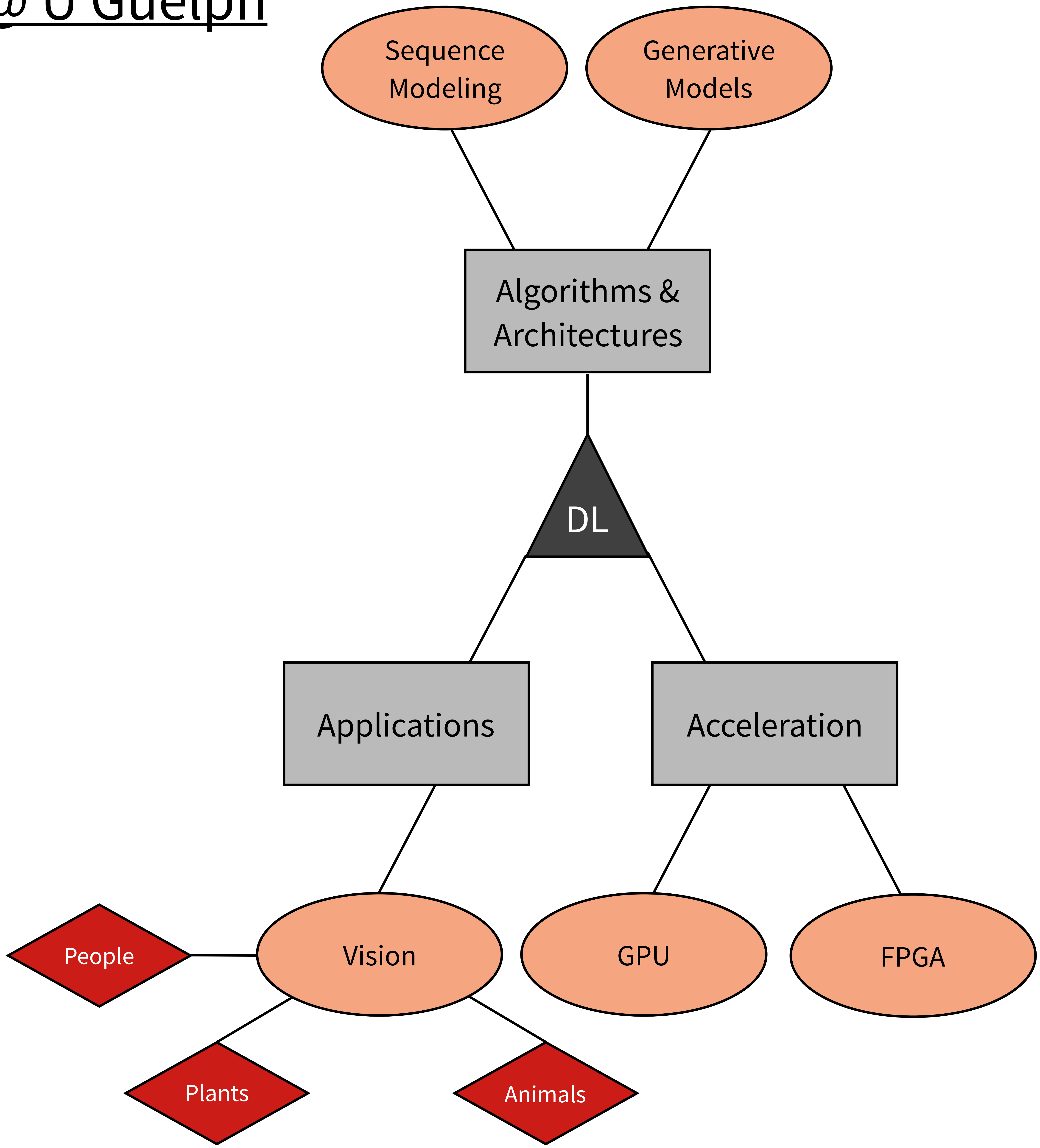
University of Guelph, Vector Institute for Artificial Intelligence and Canadian Institute for Advanced Research

Letter Field by Judson Rosebush, 1978

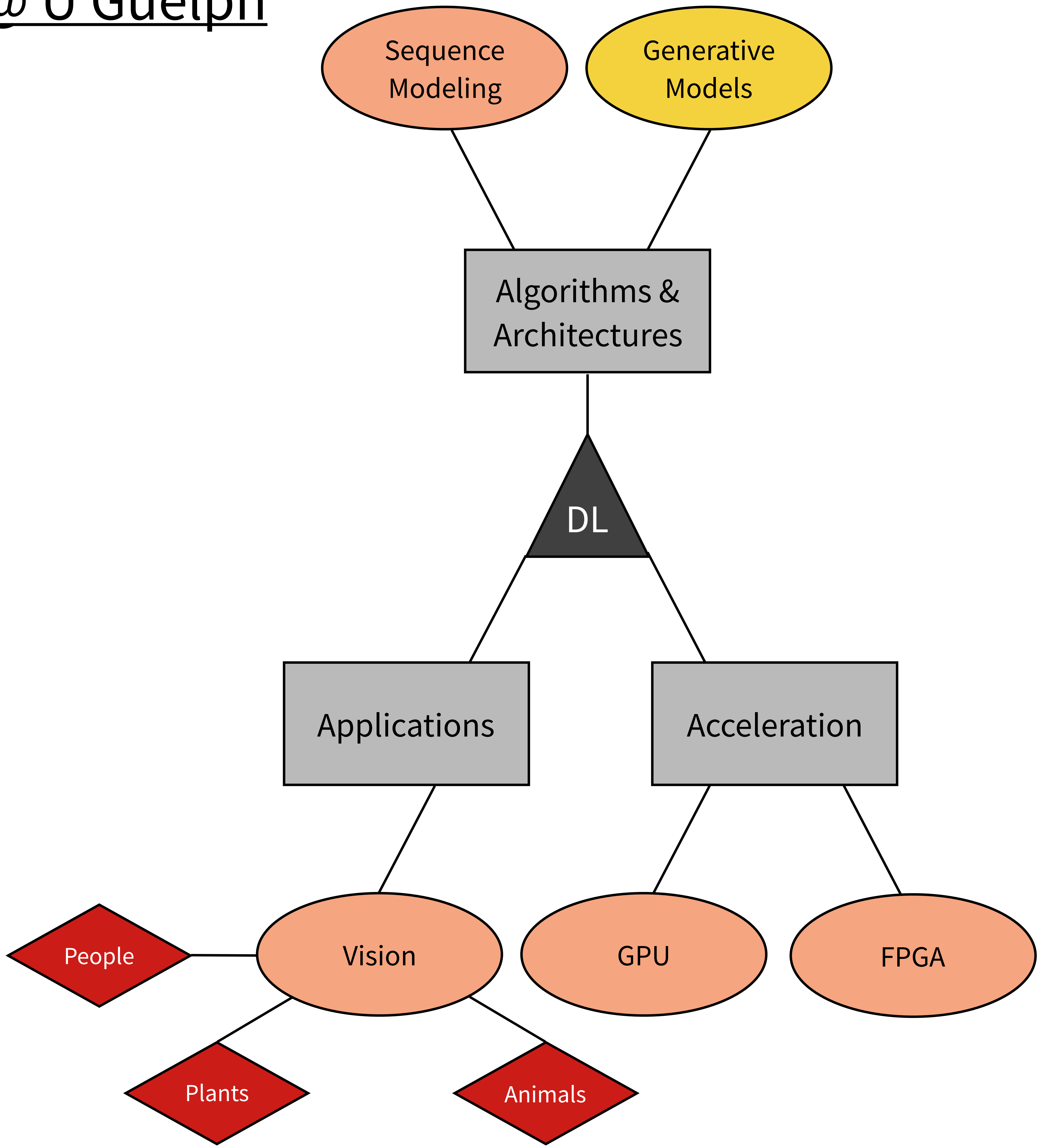
Research group: MLRG @ U of G



MLRG @ U Guelph



MLRG @ U Guelph



smartgeometry 2018

Machine Minds

Artificial Intelligence in Architecture

Workshops

9am - 9pm, Monday May 7 - Thursday May 10, Daniels Graduate Studio

AI strategies for space frame design	Clemens Preisinger, Zeynep Aksoz
behavioural enviro[NN]ments	Sam Joyce, Kate Jeffery, Jonathan Irawan, Verina Christie
data mining the city	Danil Nagy, Jim Stoddart, Lorenzo Villagi
fibrous timber joints	Dominga Garufi, Hans Wagner, Tobias Schwinn, Dylan Wood
fresh eyes	Kyle Steinfeld, Kat Park, Adam Menges, Samatha Walker
inside the black box	Mark Cichy, Ultan Byrne
materials as probes	Billie Faircloth, Christopher Connock, Ryan Welch, Patrick Weiss
mind ex machina	Panagiotis Michalatos, Nono Martinez Alonso, Jose Luis Garcia del Castillo
soft office	Giulio Brugnaro, Brian Ringley, Maria Yablonina
sound and signal	John Granzow, Catie Newell, Kim Harty

Conference

9am - 6pm, Friday May 11, Daniels Principal Hall

David Benjamin	The Living
Graham Taylor	Vector Institute / Next AI
Dan Price	adidas Future Team
Alex Tessier	Autodesk Research
Dimitrie Stefanescu	Speckle.Works
Raffaello D'Andrea	ETH Zurich
Kate Jeffery	Jeffery Lab UCL
Jeremy Hadall	Manufacturing Technology Centre
Workshop Presentations	SG Cluster Champions

9am - 6pm, Saturday May 12, Daniels Principal Hall

Christian Derix	SUPERSPACE Woods Bagot
Clemens Preisinger	Karamba 3D
Elissa Ross	MESH Geometry
Daniel Davis	WeWork
Christopher Connock	KieranTimberlake
Matt Jezyk	Autodesk
Philip Beesley	Philip Beesley Architect
Mark Cichy	Dialog
Alex Josephson	Partisans
Eric Burry	Eventscape
Nick Puckett	OCAD
Molly Steenson	Carnegie Mellon
Mickey McManus	MAYA Design

Exhibition Opening

6pm - 10pm, Friday May 11, Daniels Grad Studio

www.smartgeometry.org

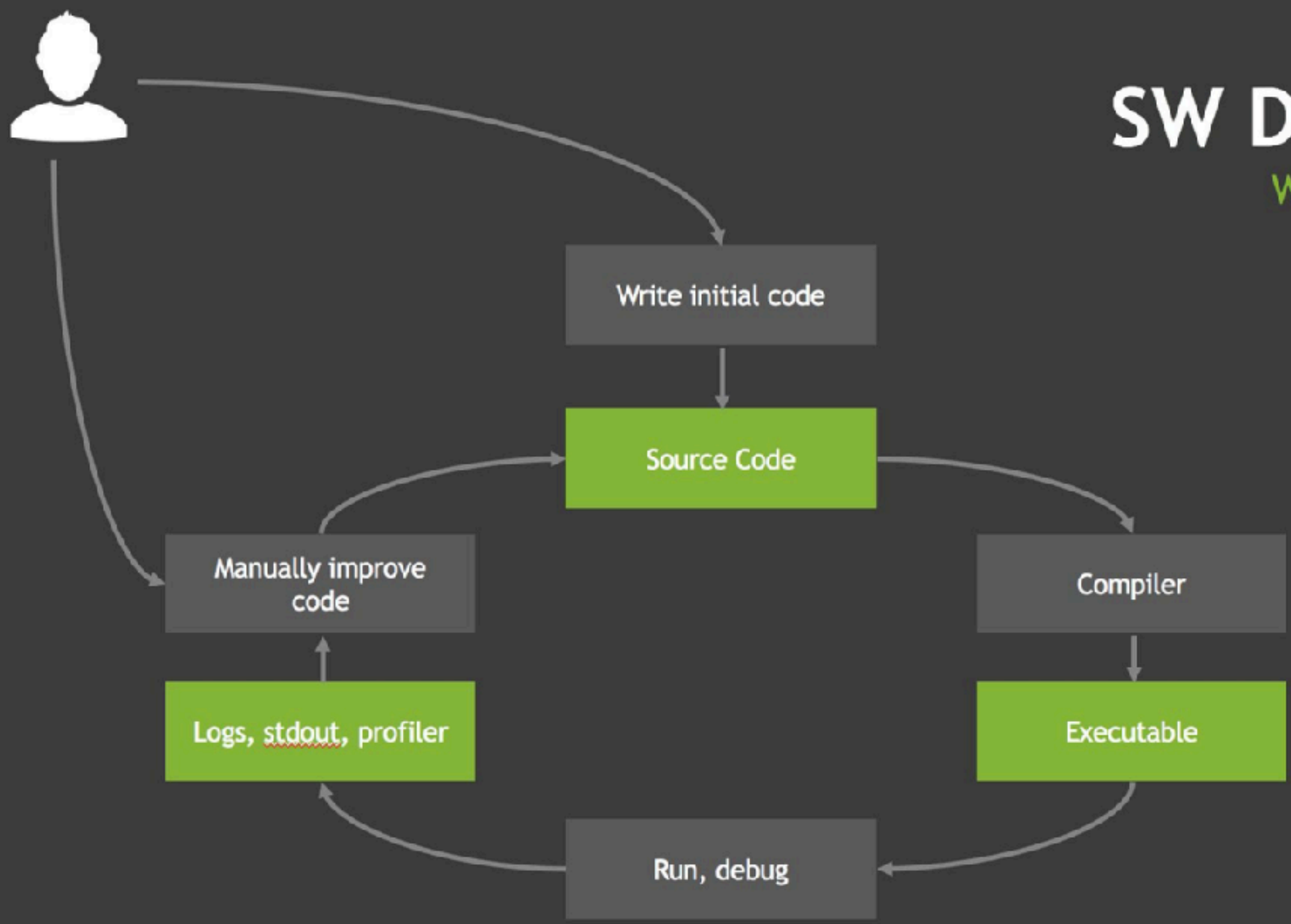
D
AN
IELS

AUTODESK.

via Google

SW DEV PROCESS

Write code, compile, test, debug, repeat...



via Clement Farabet

2.0

Source code → Data

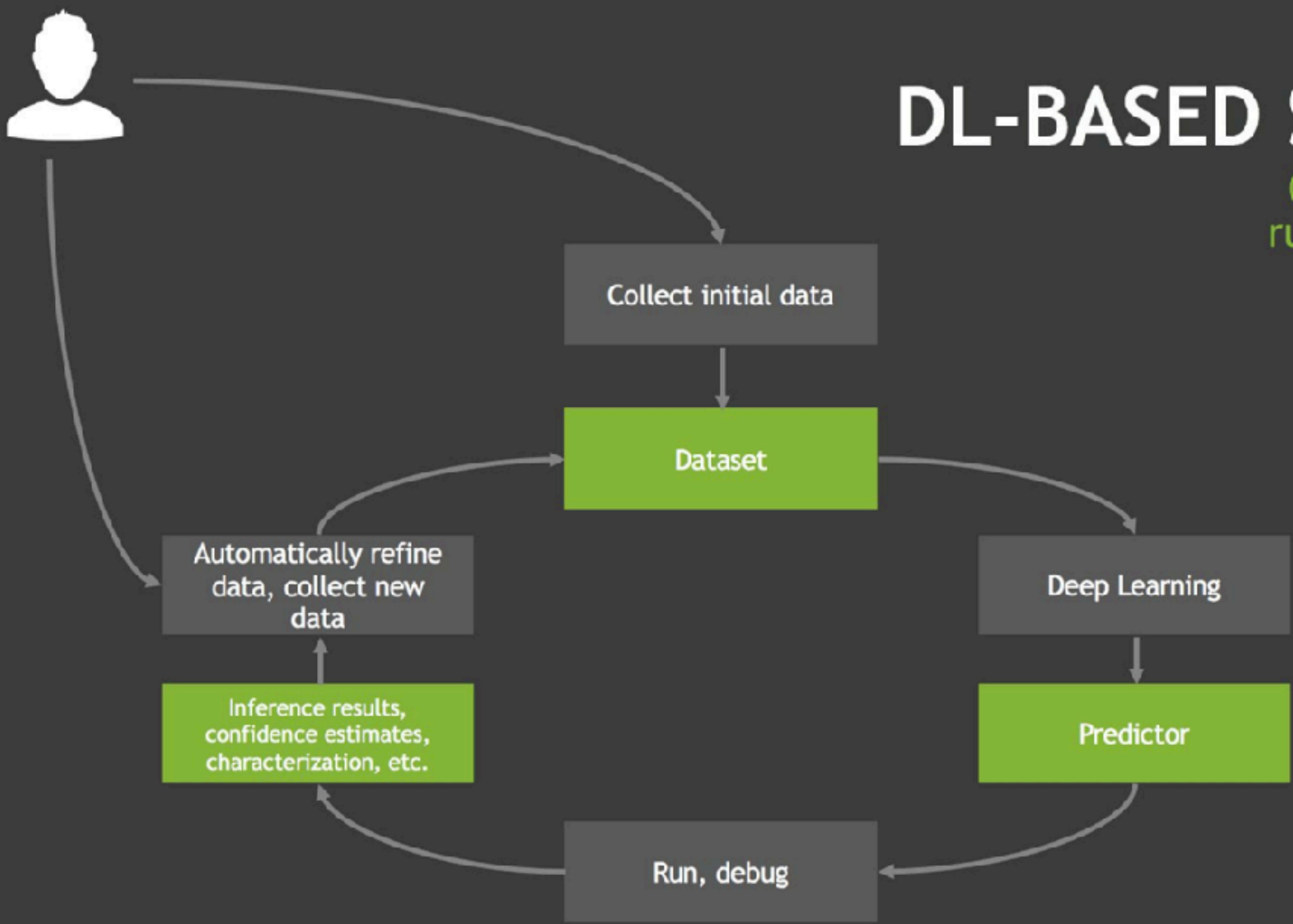
Compiler → Deep Learning

Executable → Predictor

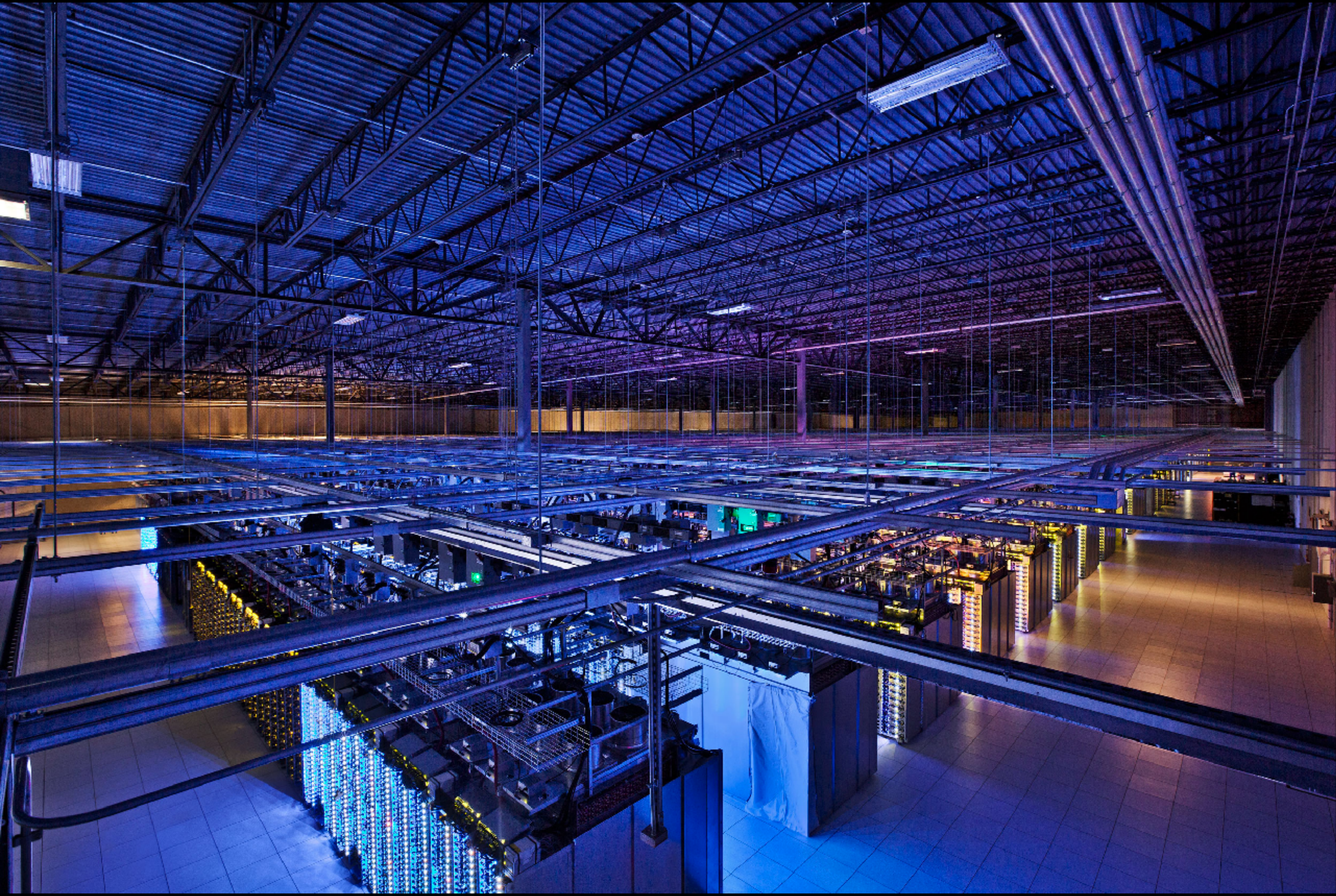
via Clement Farabet

DL-BASED SW PROCESS

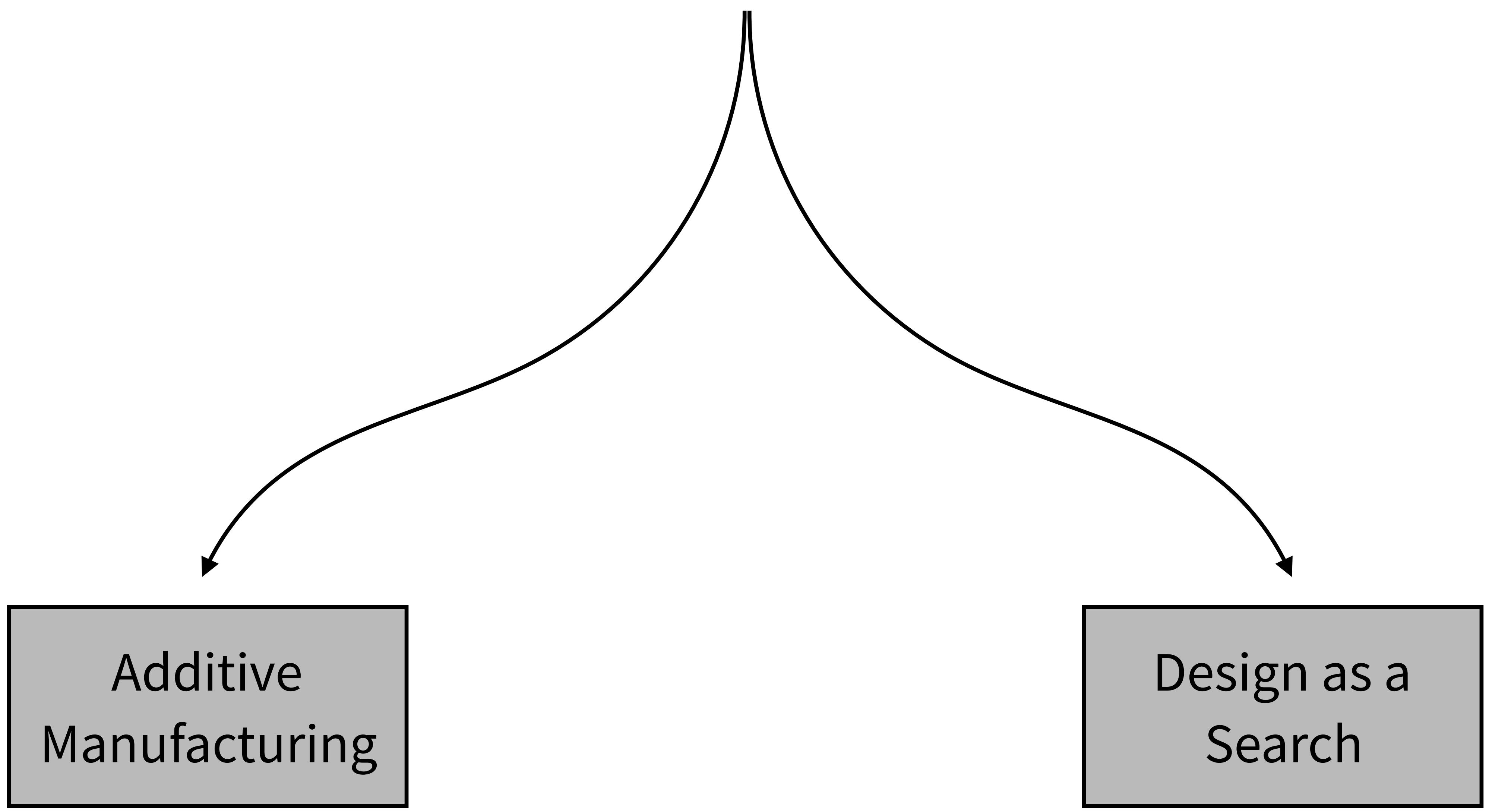
Collect initial data, train,
run/debug, mine new data

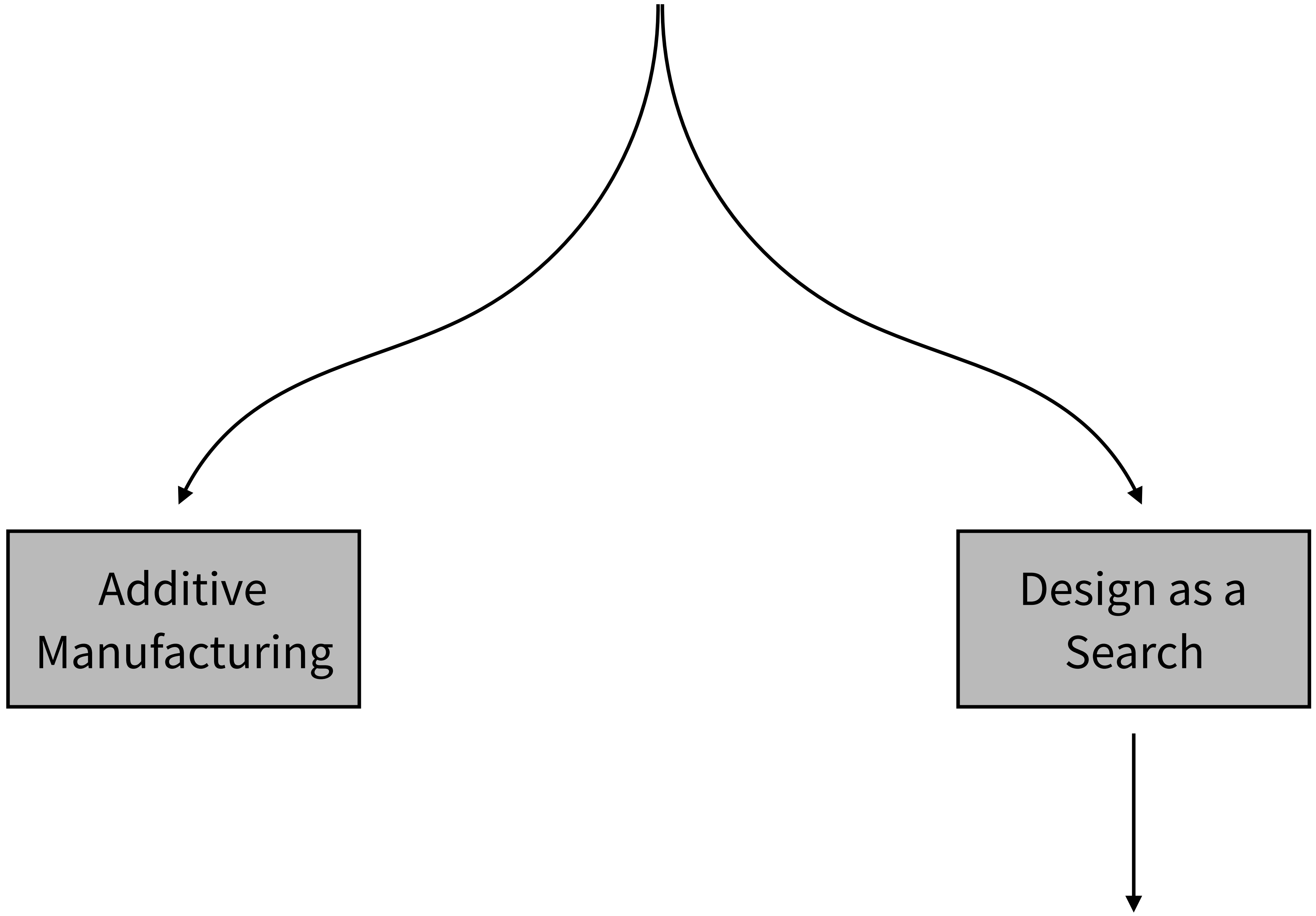


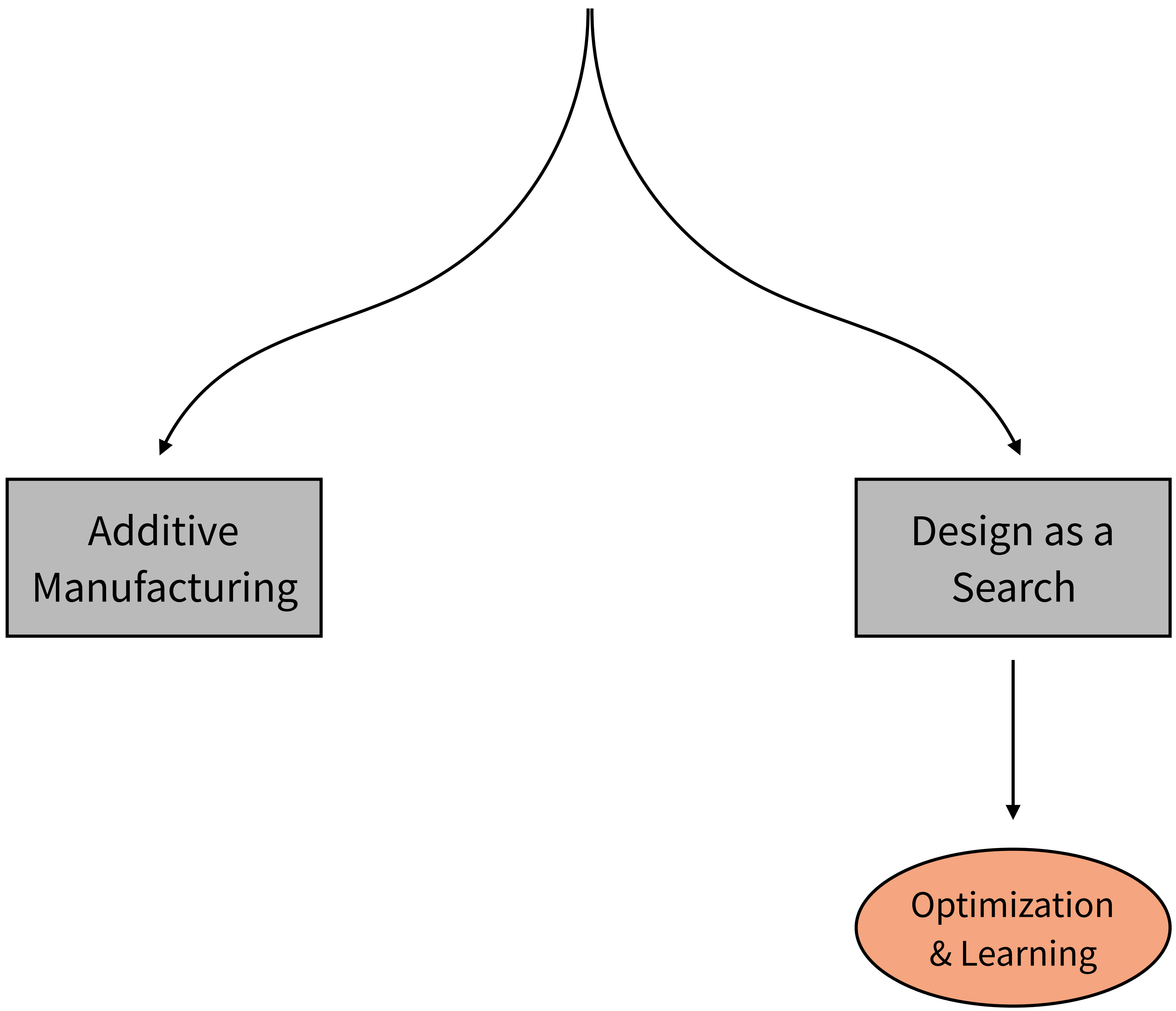
via Clement Farabet



via wired.com


Additive Manufacturing
Design as a Search


Additive Manufacturing
Design as a Search



Additive Manufacturing
Design as a Search
Optimization & Learning

Medium

Andrej Karpathy Follow
Director of AI at Tesla. Previously Research Scientist at OpenAI and PhD student at Stanford. I like to train deep neural nets on large datasets.

Nov 11, 2017 · 8 min read

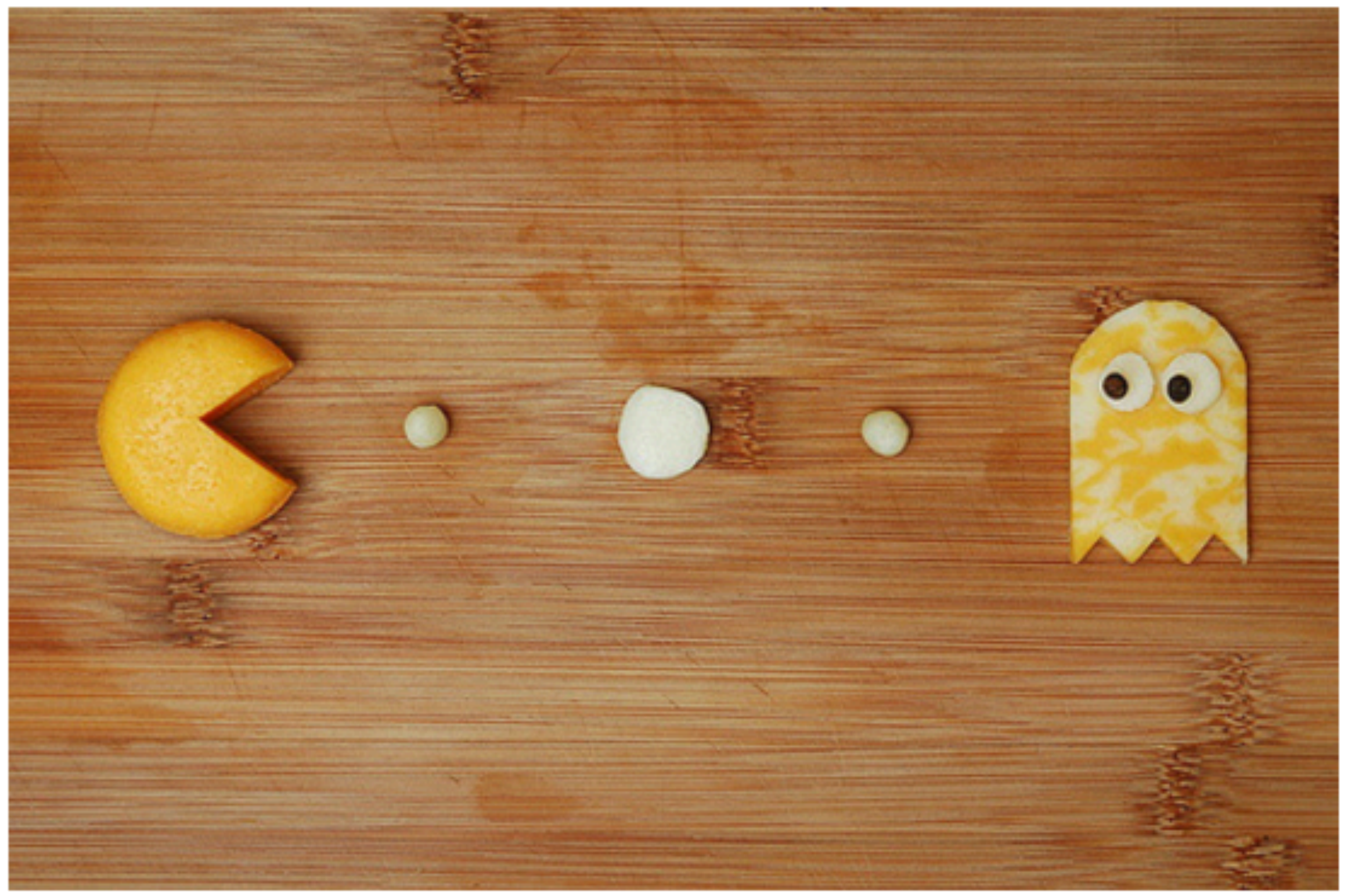
Software 2.0

I sometimes see people refer to neural networks as just "another tool in your machine learning toolbox". They have some pros and cons, they work here or there, and sometimes you can use them to win Kaggle competitions. Unfortunately, this interpretation completely misses the forest for the trees. Neural networks are not just another classifier, they represent the beginning of a fundamental shift in how we write software. They are Software 2.0.

The "classical stack" of **Software 1.0** is what we're all familiar with—it is written in languages such as Python, C++, etc. It consists of explicit instructions to the computer written by a programmer. By writing each line of code, the programmer is identifying a specific point in program space with some desirable behavior.

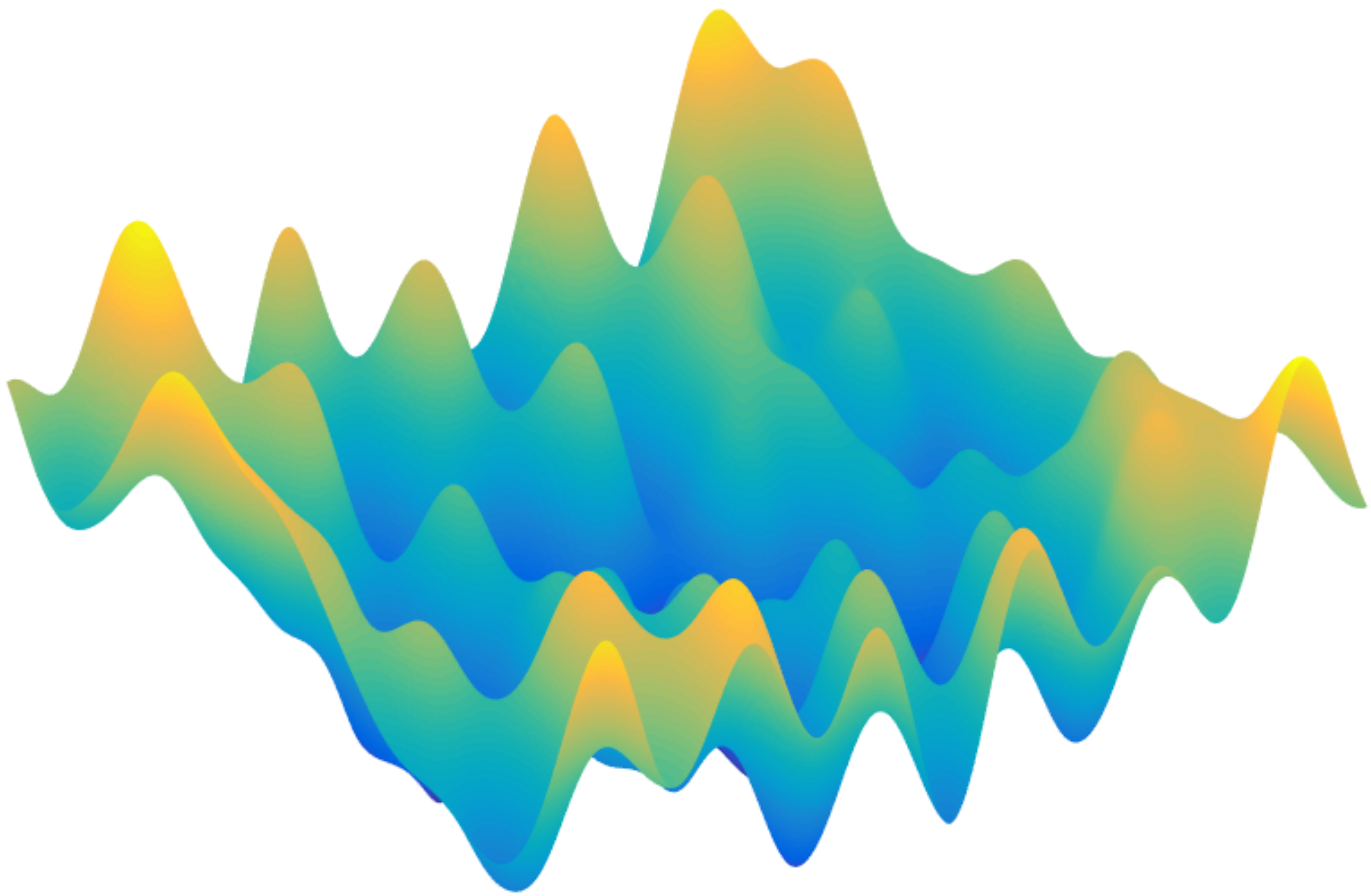
Deep Learning is Eating Software

NOVEMBER 13, 2017
By Pete Warden
in UNCATEGORIZED
22 COMMENTS



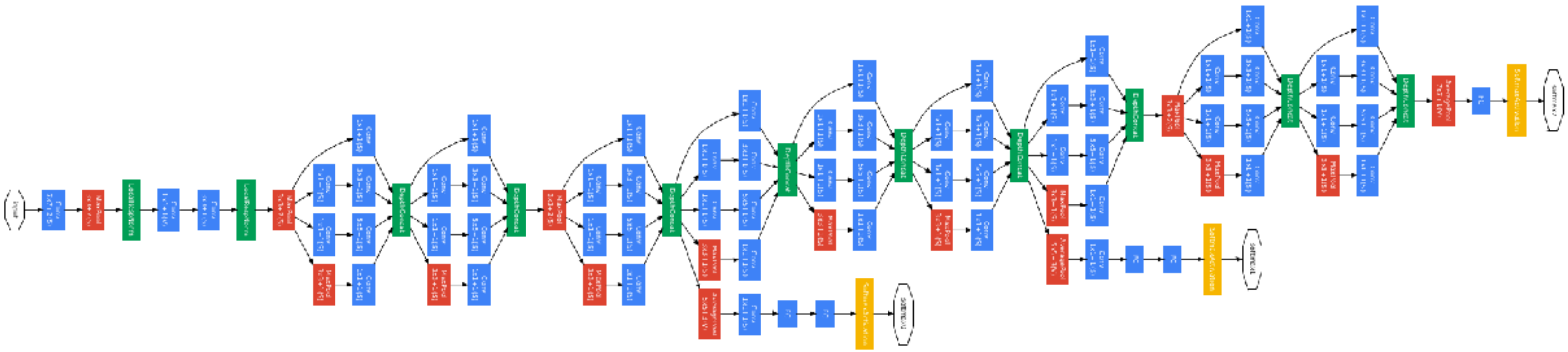
via Bloomberg

Optimization vs. Learning



via Yuejie Chi

Optimization in Machine Learning



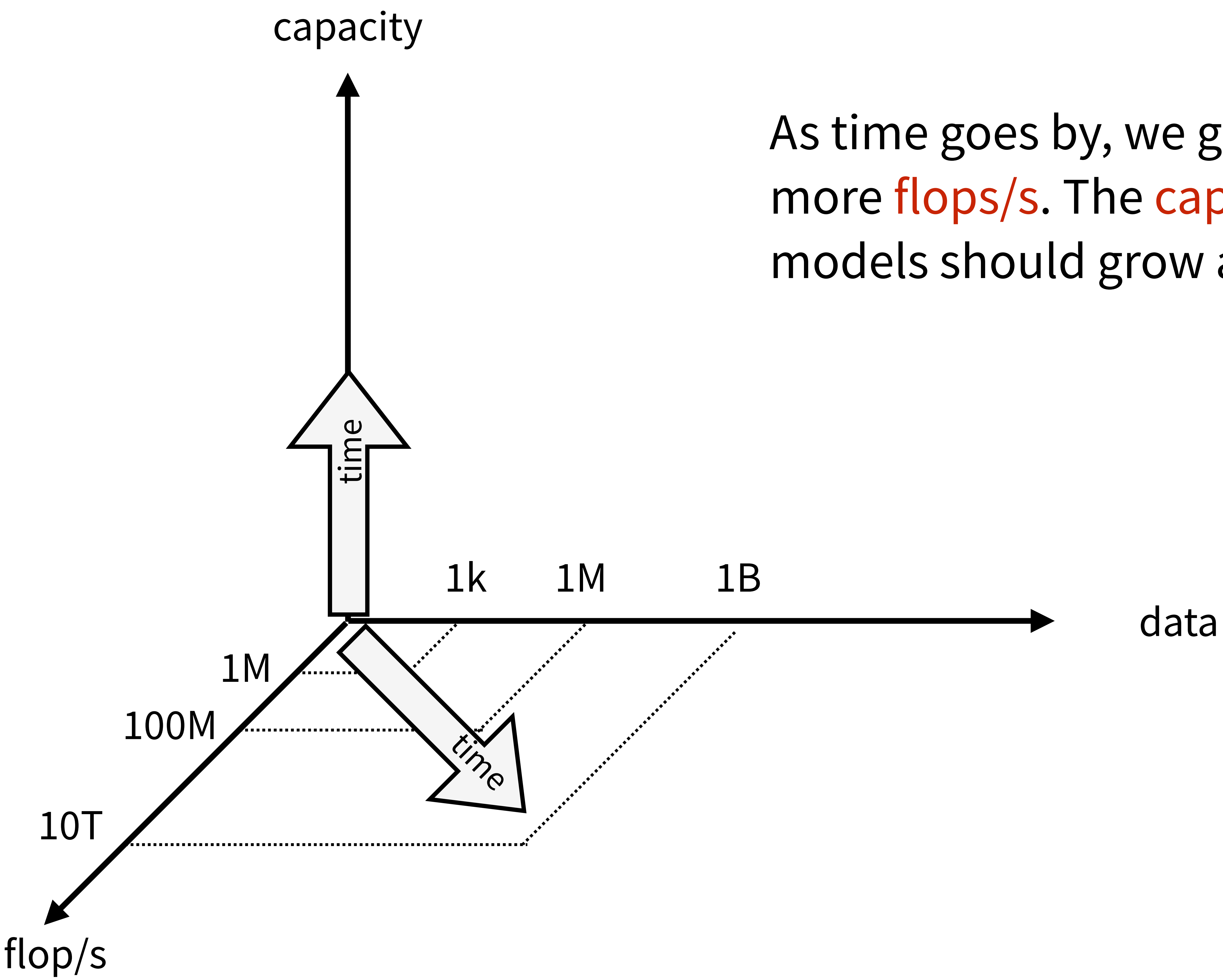
<u>Parameter search:</u>

- Knobs inside the boxes
- 100M - 1B parameters
- Core of ML

<u>Model search:</u>

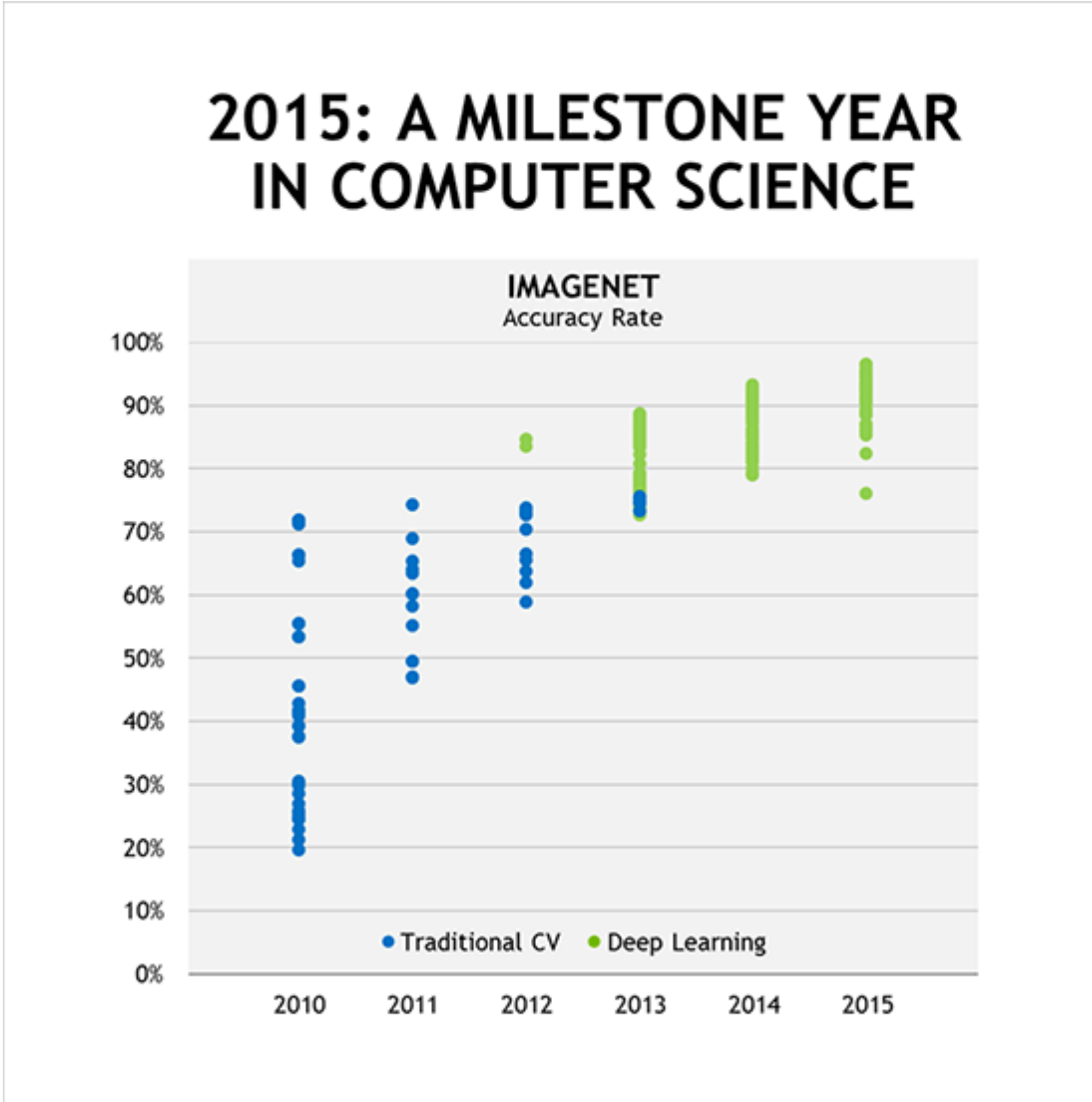
- Typically fixed architecture
- 10 - 100* *hyperparameters*
- Traditionally a human task

GoogLeNet (circa 2014)



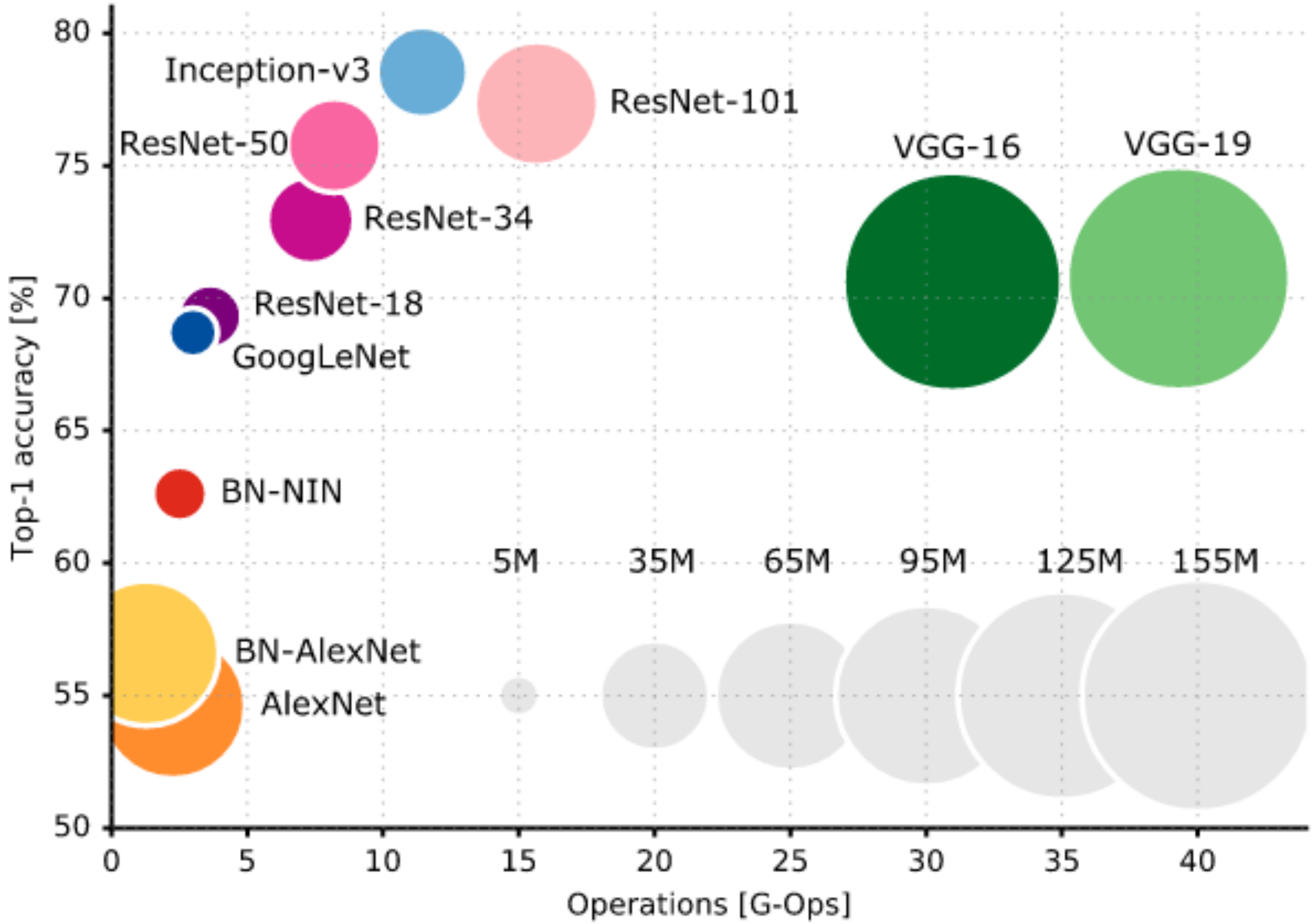
As time goes by, we get more data and more flops/s. The capacity of ML models should grow accordingly.

inspired by Marc'Aurelio Ranzato


2015: A MILESTONE YEAR IN COMPUTER SCIENCE
IMAGENET
Accuracy Rate
100%
90%
80%
70%
60%
50%
40%
30%
20%
10%
0%
Traditional CV
Deep Learning
2010
2011
2012
2013
2014
2015

via NVIDIA

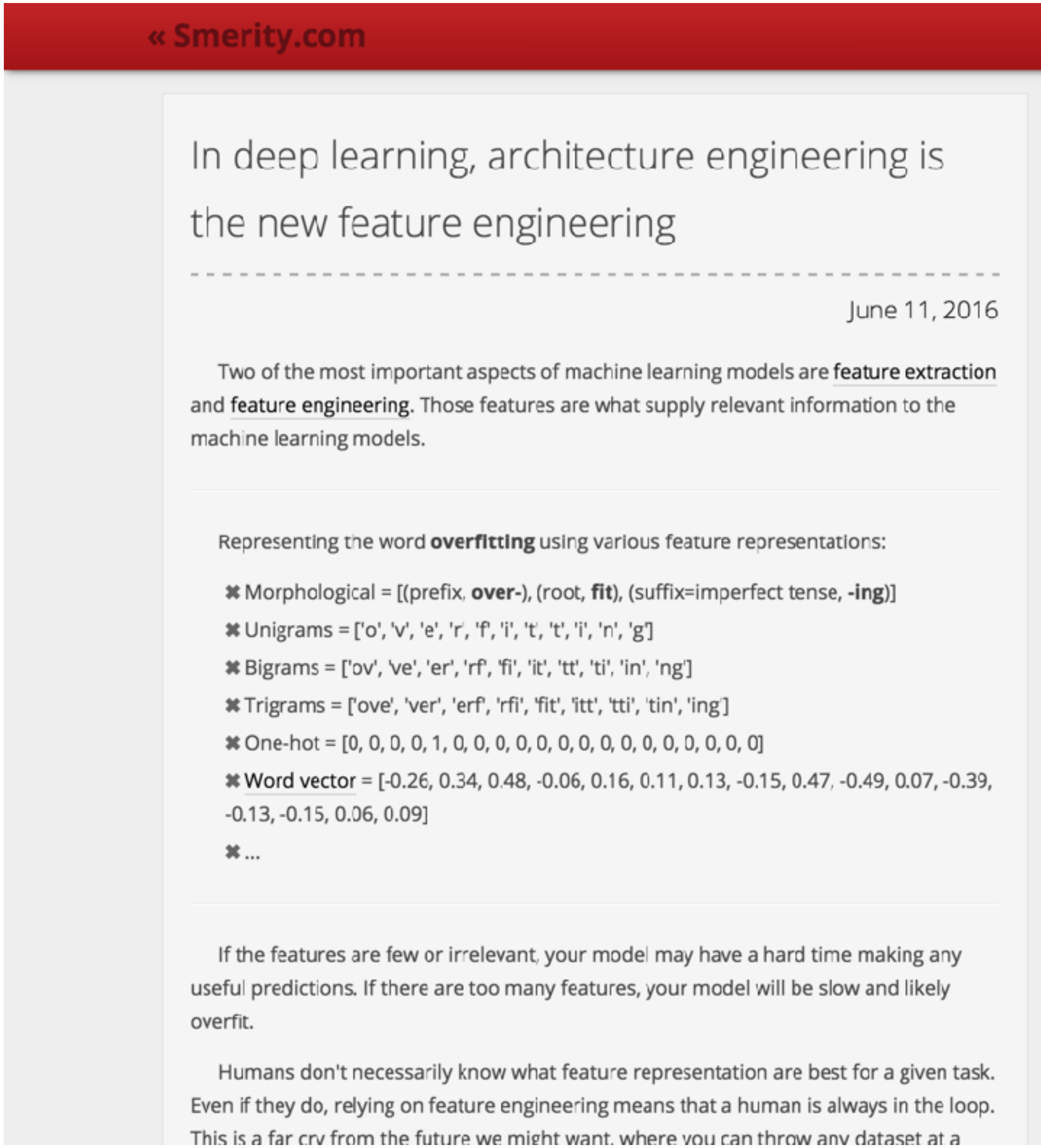
Saturation?



Eugenio Culurciello's blog:
https://culurciello.github.io/tech/2016/06/04/nets.html

Learning architectures

- Romanticized notion of DL - end of feature engineering
- Feature engineering has decreased
- Architectures have become more complex



« Smerity.com

In deep learning, architecture engineering is the new feature engineering

June 11, 2016

Two of the most important aspects of machine learning models are feature extraction and feature engineering. Those features are what supply relevant information to the machine learning models.

Representing the word **overfitting** using various feature representations:

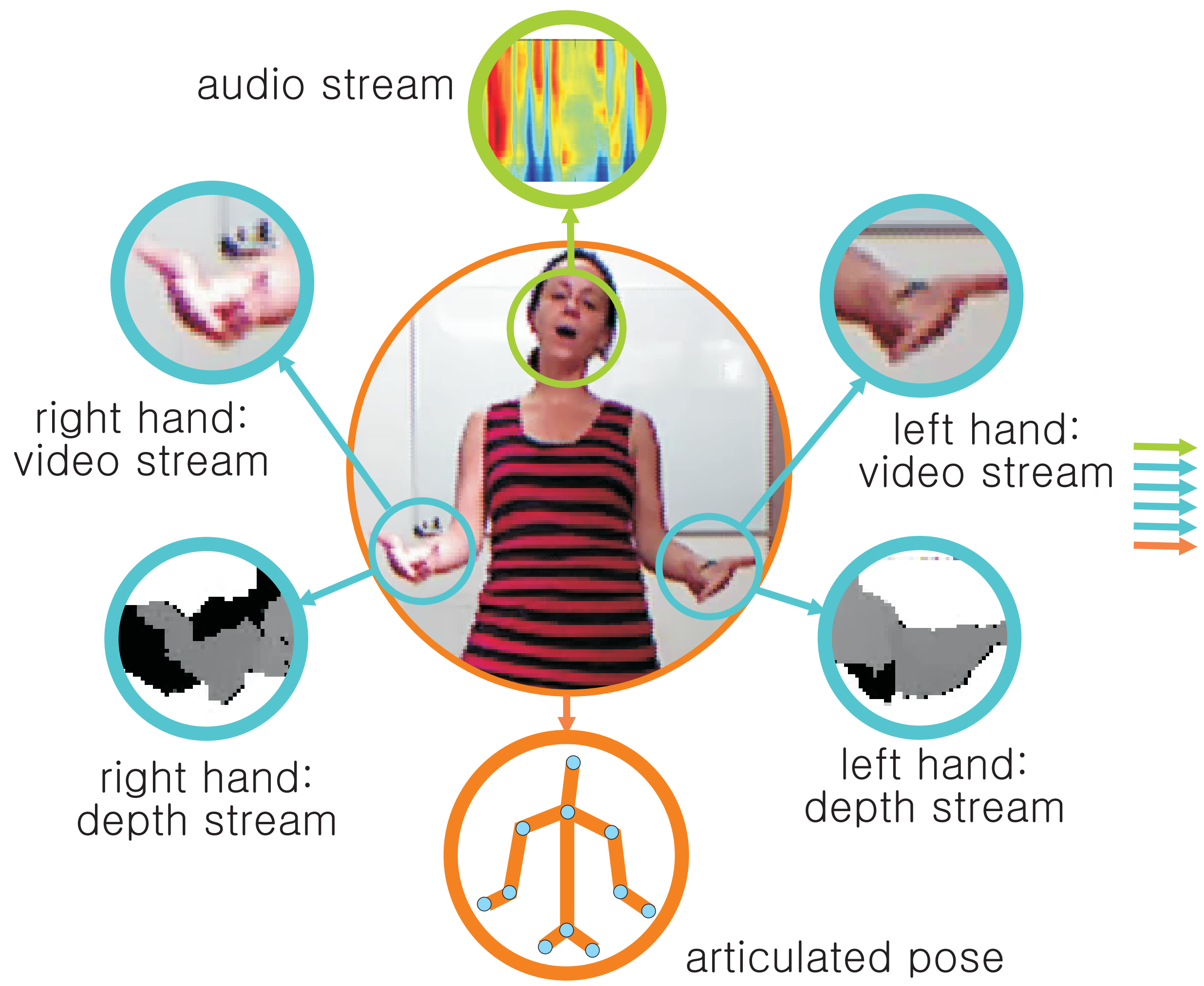
- Morphological = [(prefix, **over-**), (root, **fit**), (suffix=imperfect tense, **-ing**)]
- Unigrams = ['o', 'v', 'e', 'r', 'f', 'i', 't', 't', 'i', 'n', 'g']
- Bigrams = ['ov', 've', 'er', 'rf', 'fi', 'it', 'tt', 'ti', 'in', 'ng']
- Trigrams = ['ove', 'ver', 'erf', 'rfi', 'fit', 'itt', 'tti', 'tin', 'ing']
- One-hot = [0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
- Word vector = [-0.26, 0.34, 0.48, -0.06, 0.16, 0.11, 0.13, -0.15, 0.47, -0.49, 0.07, -0.39, -0.13, -0.15, 0.06, 0.09]
- ...

If the features are few or irrelevant, your model may have a hard time making any useful predictions. If there are too many features, your model will be slow and likely overfit.

Humans don't necessarily know what feature representation are best for a given task. Even if they do, relying on feature engineering means that a human is always in the loop. This is a far cry from the future we might want, where you can throw any dataset at a

http://smerity.com/articles/2016/architectures_are_the_new_feature_engineering.html

Example: Learning Multimodal Fusion for Gesture Recognition

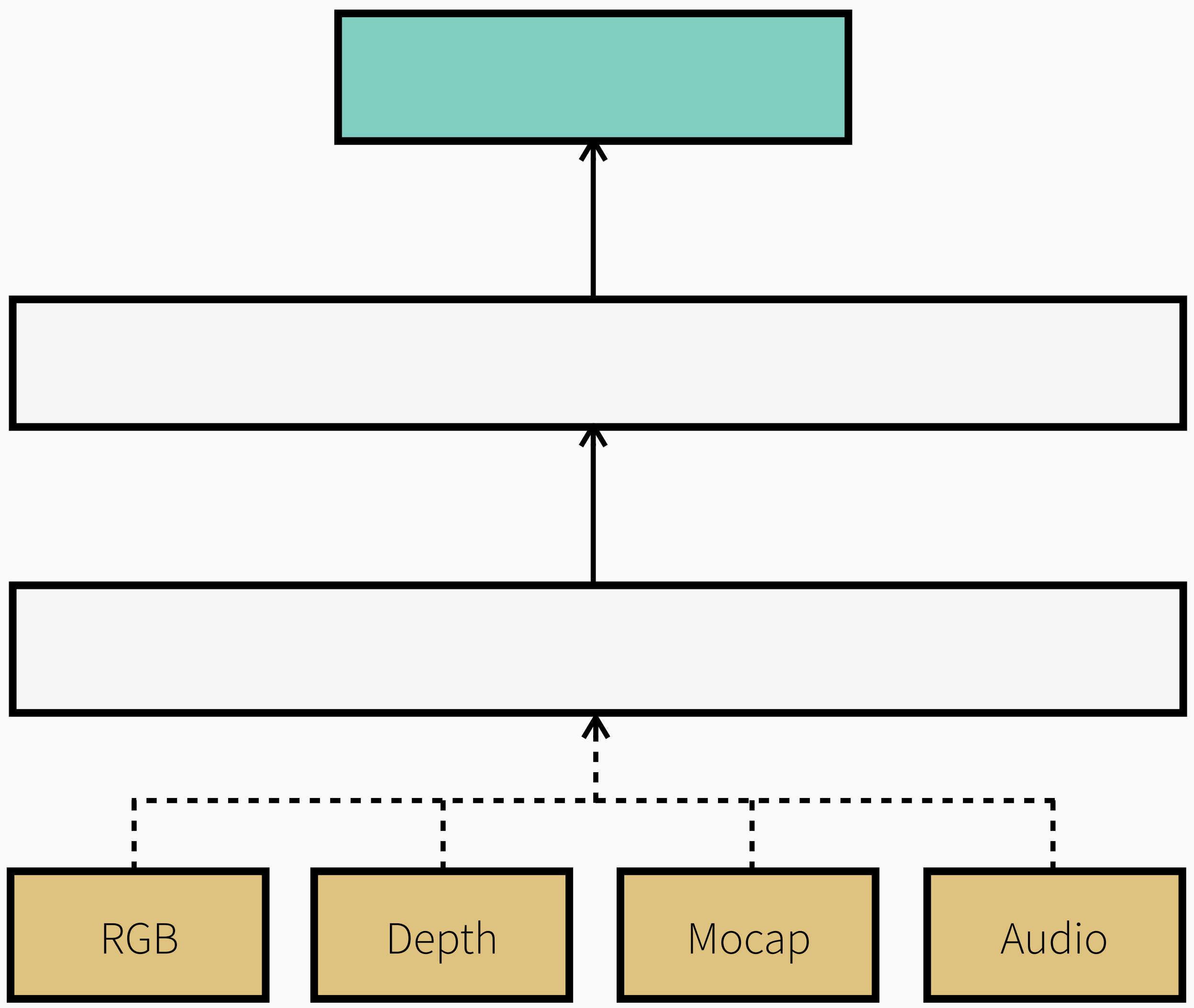


Neverova, Wolf, Taylor (2016)

Early vs. late fusion

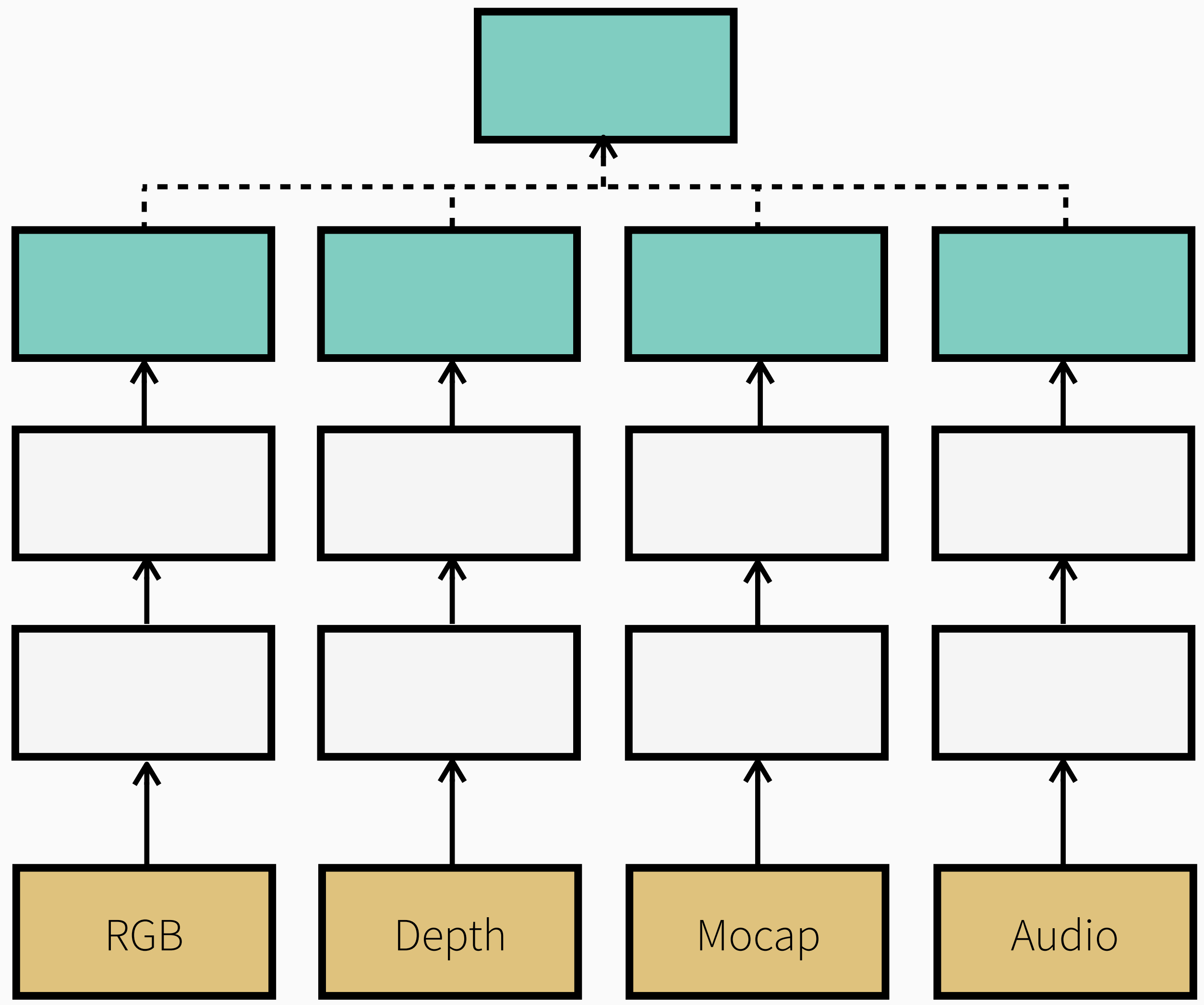
Early Fusion

Fuse modalities at input
(or preprocessed feature) level

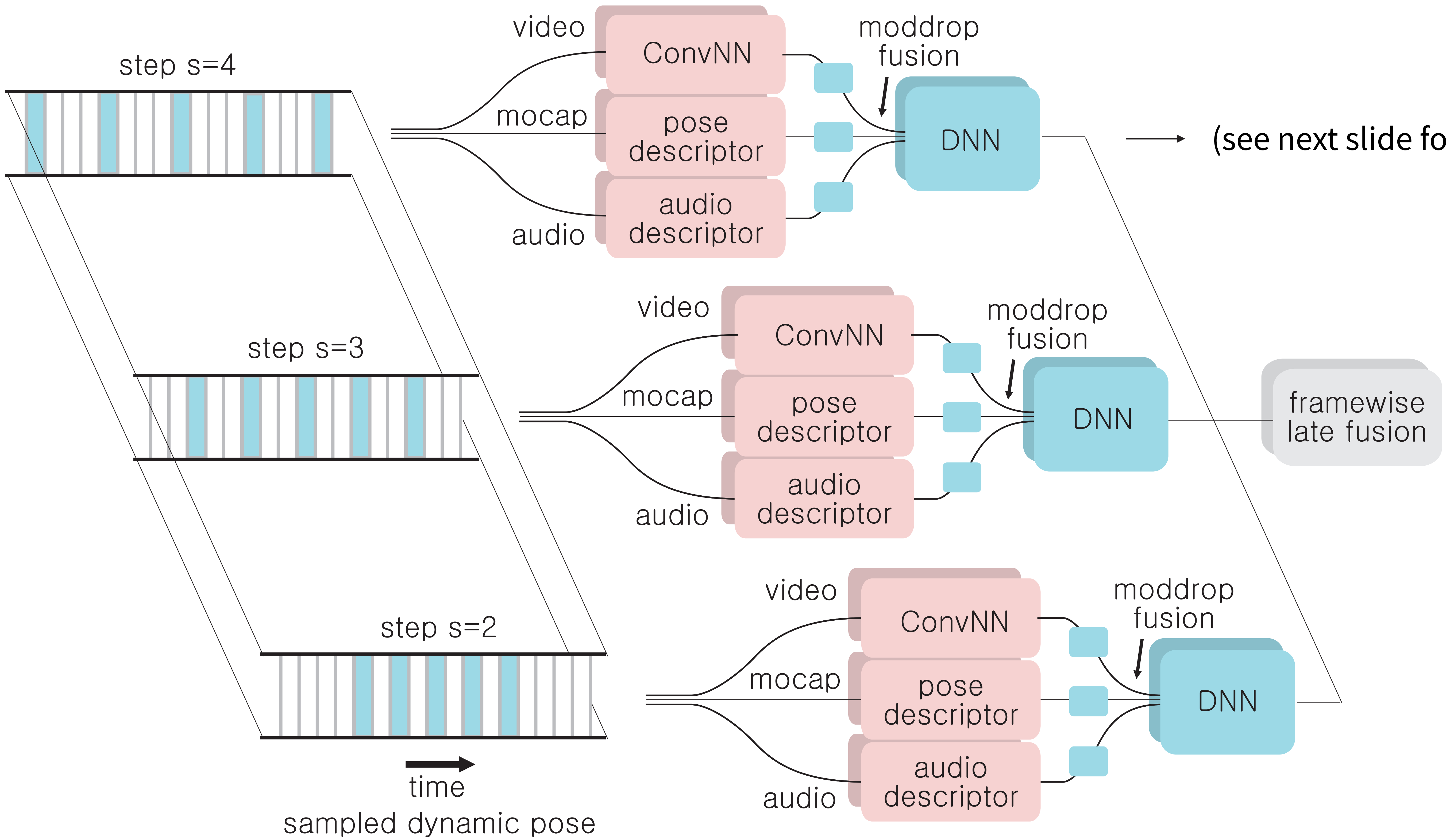


Late Fusion

Fuse modalities at output level



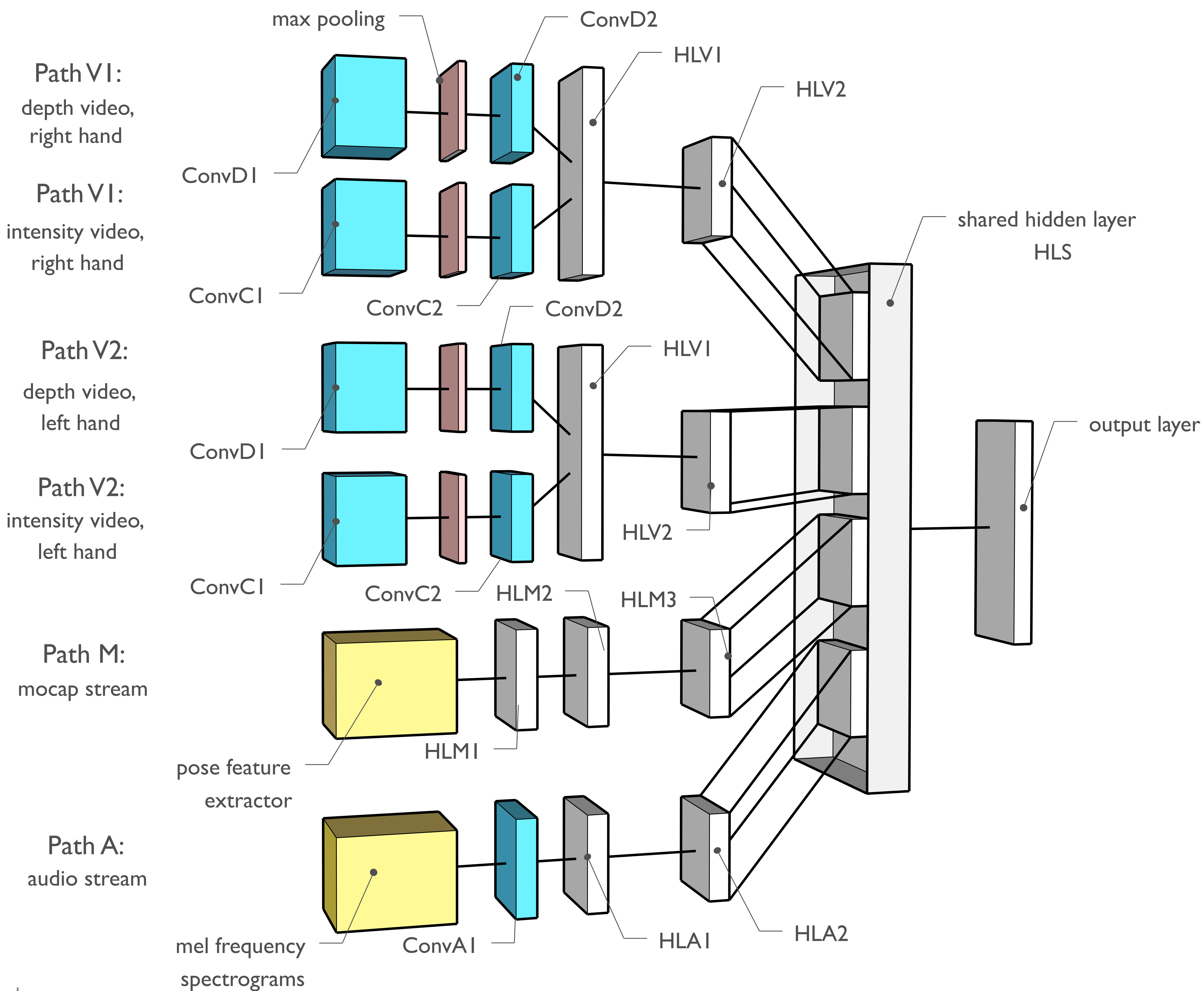
A multi-scale architecture



(see next slide for detail)

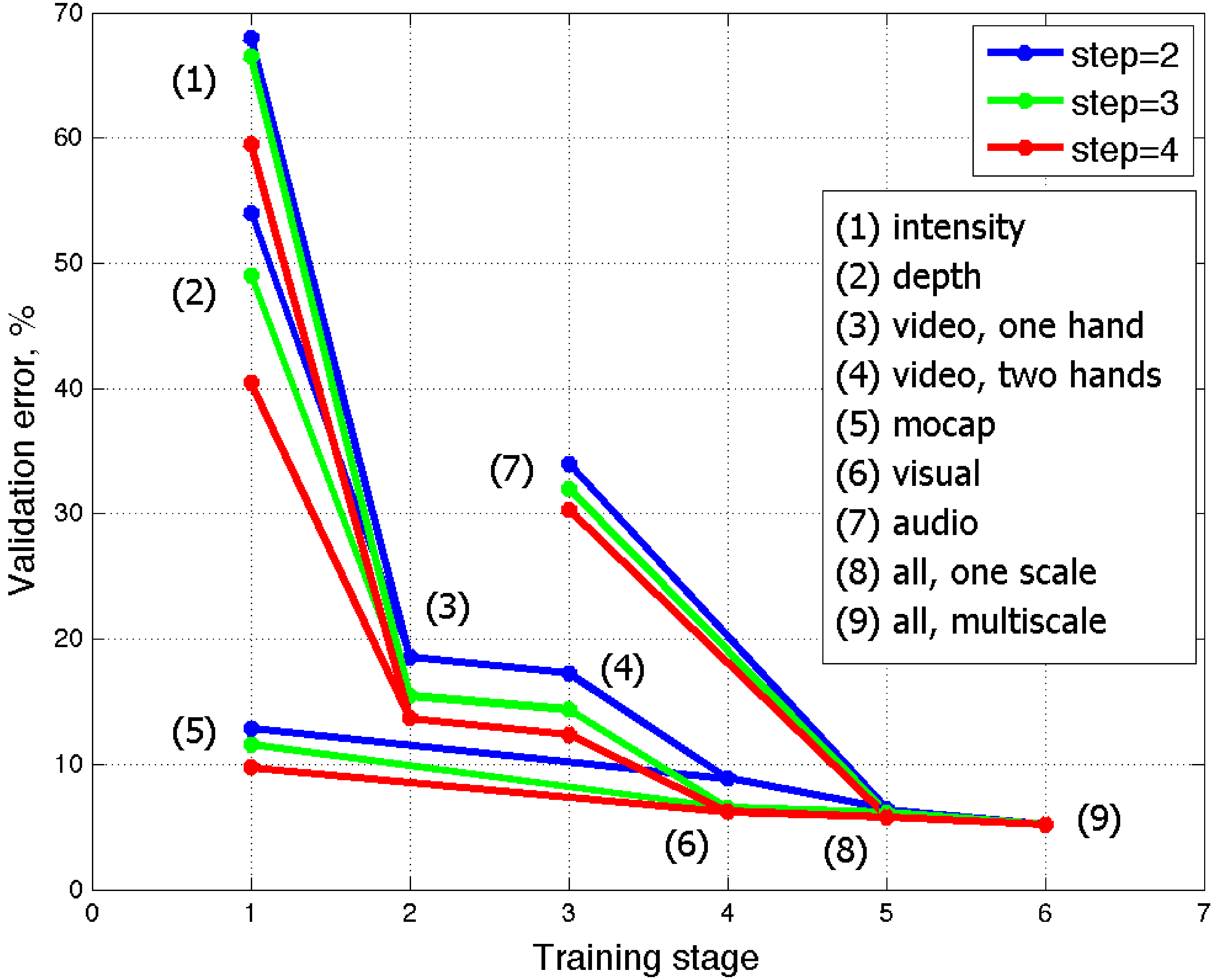
Operates at 3 temporal scales
corresponding to dynamic poses of 3 different durations

Single-scale deep architecture



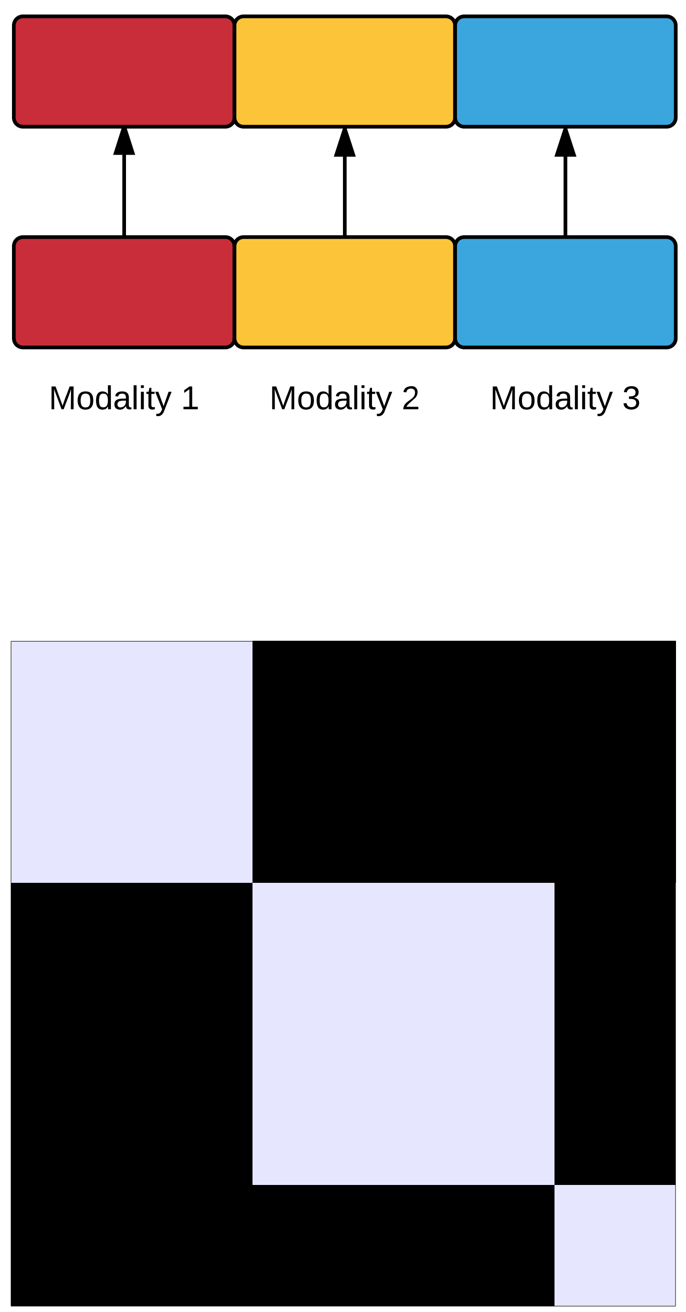
Neverova, Wolf, Taylor (2016)

Error evolution during iterative training

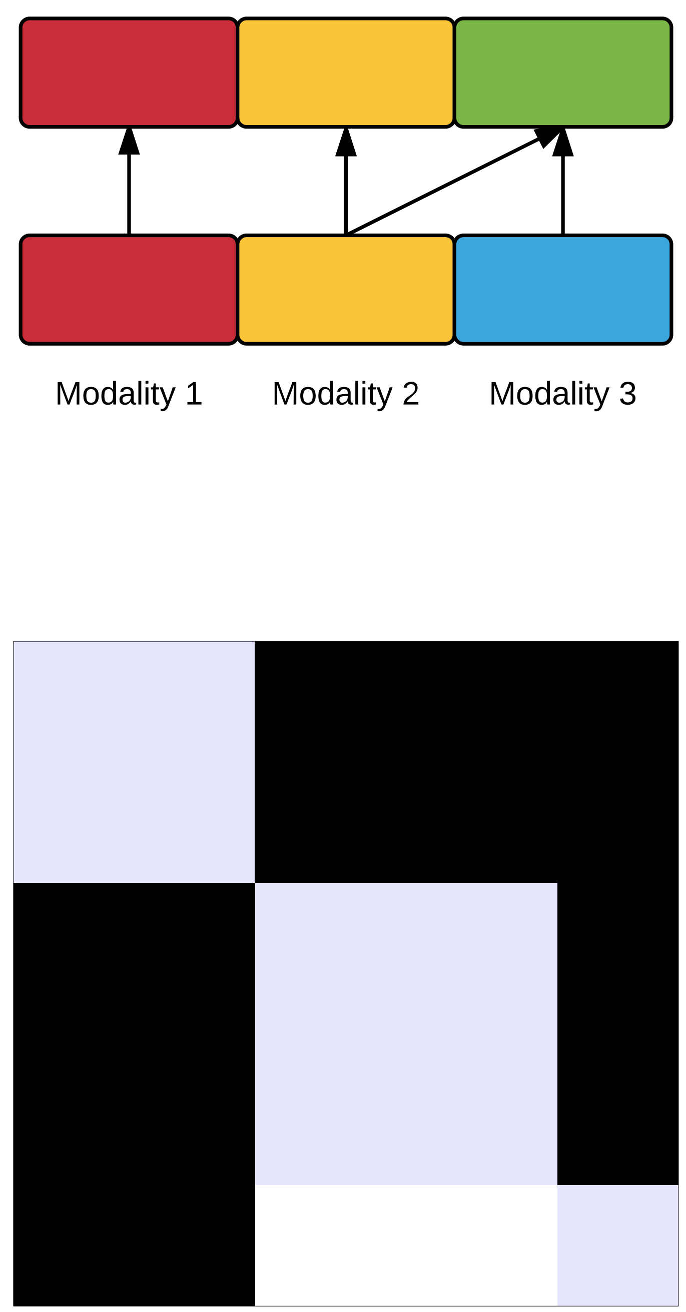


Learning Fusion Architectures (1)

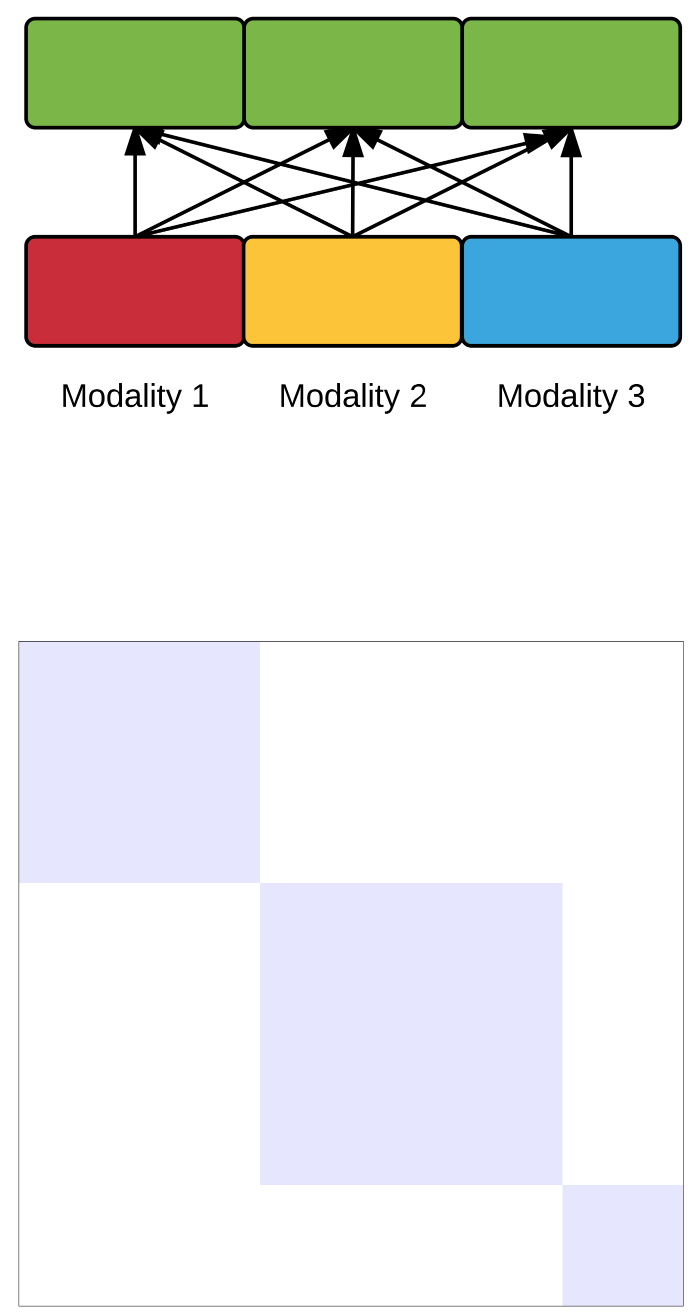
Three typical fusion architectures achievable by *Modout*, with corresponding weight masks:



Independent



Fused

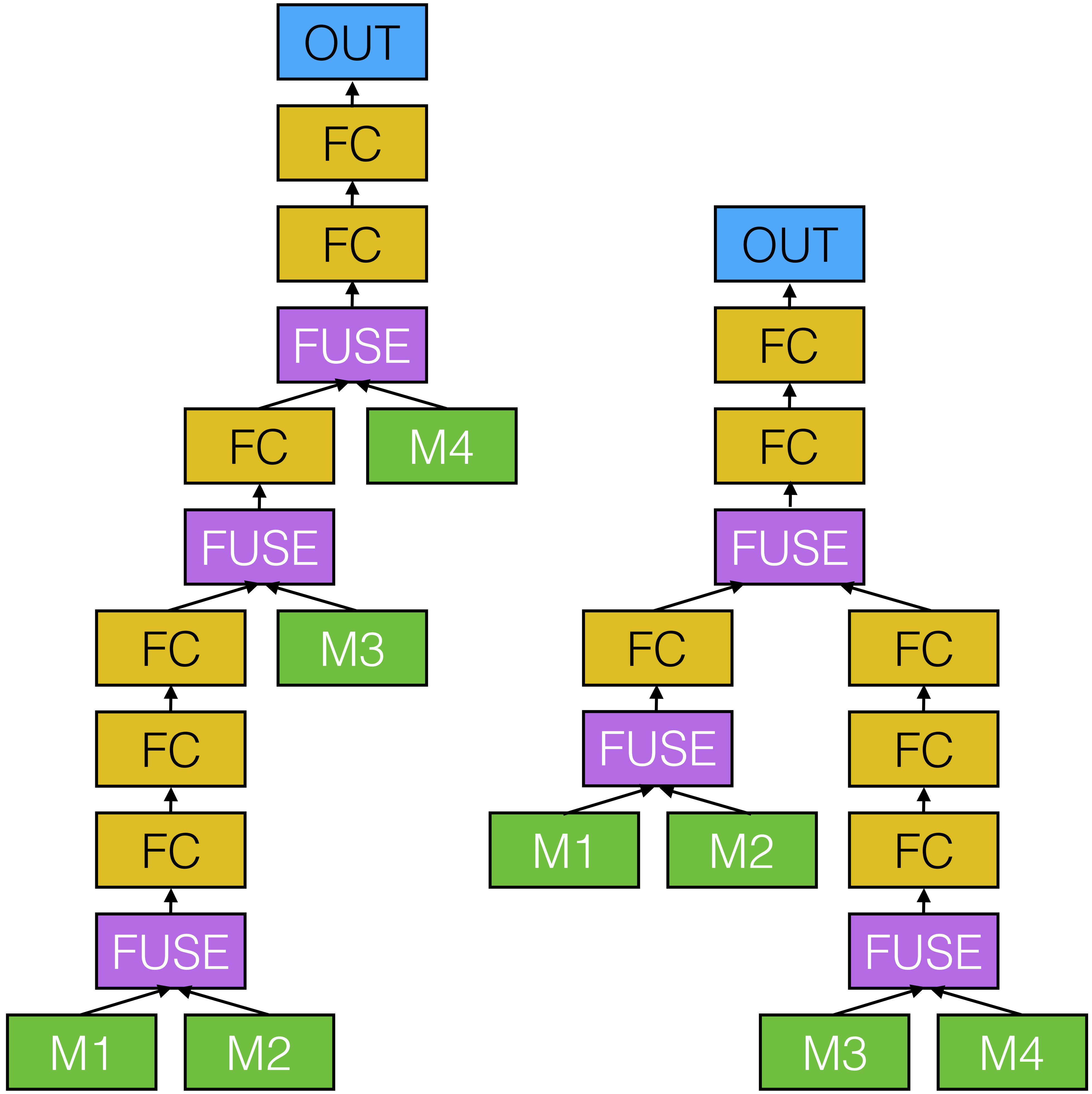


Fully-connected

Li, Neverova, Wolf, Taylor (2017)

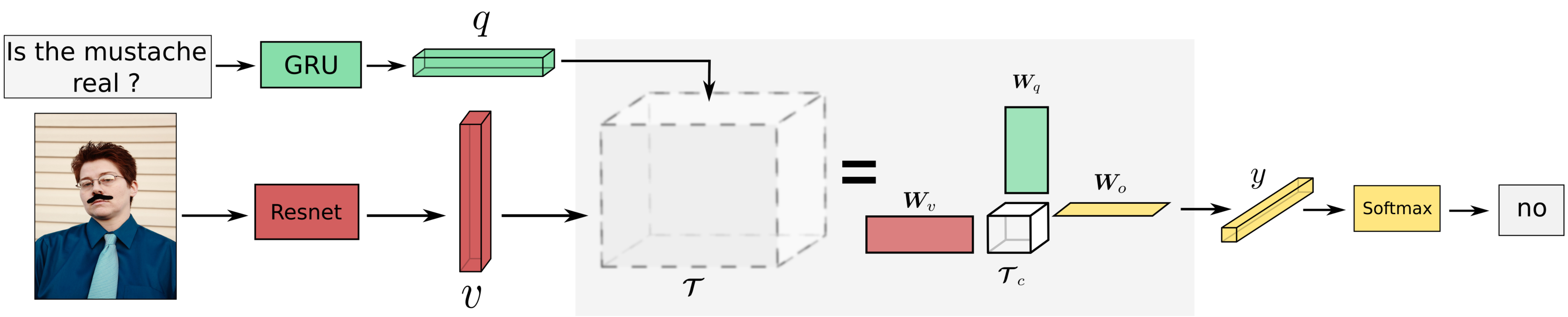
Learning Fusion Architectures (2)

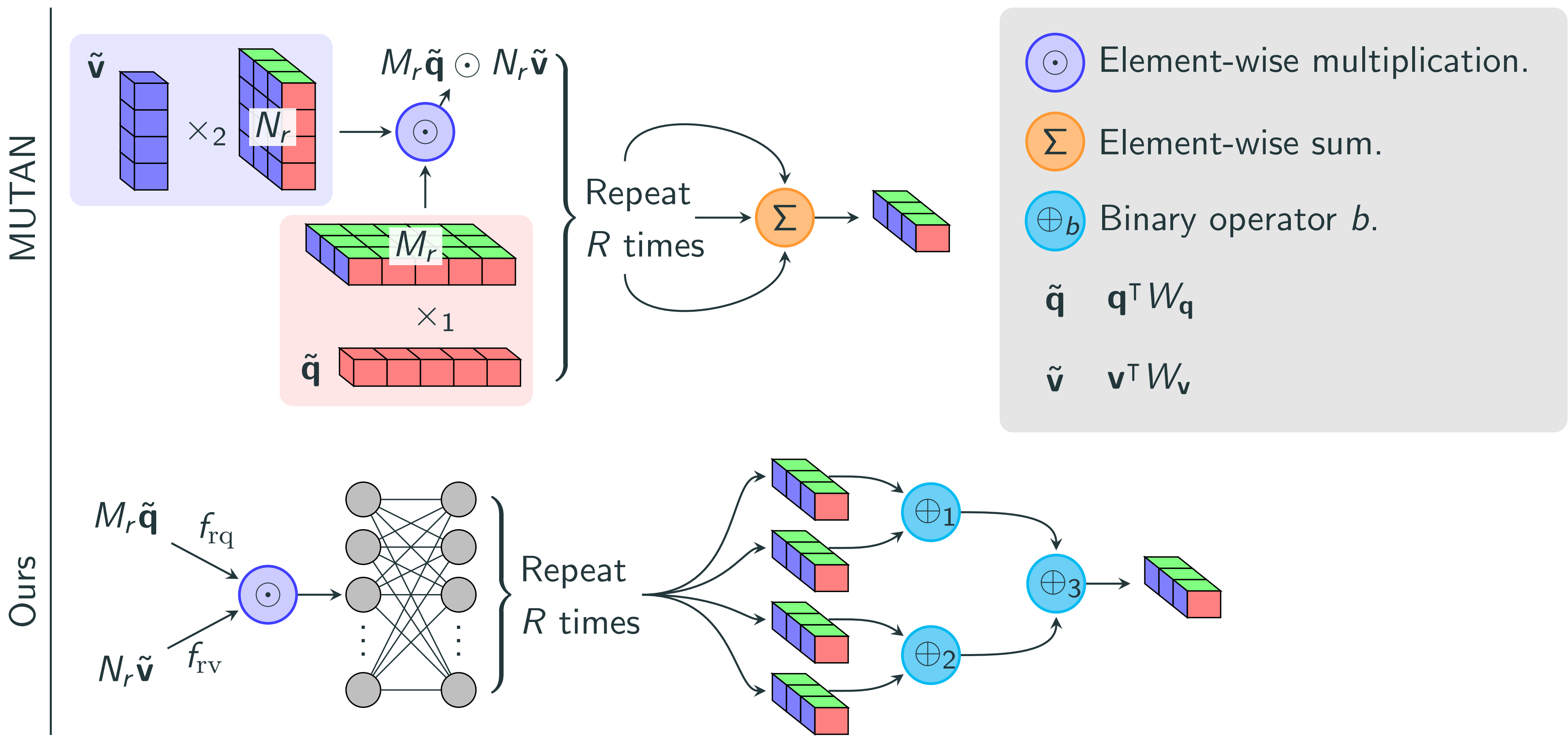
Applying *Bayesian Optimization* to Search Space Over Graphs



Ramachandram, Lisicki, Shields, Amer, Taylor (2018)

Learning Fusion Operators for VQA





Duke, Taylor (2018)

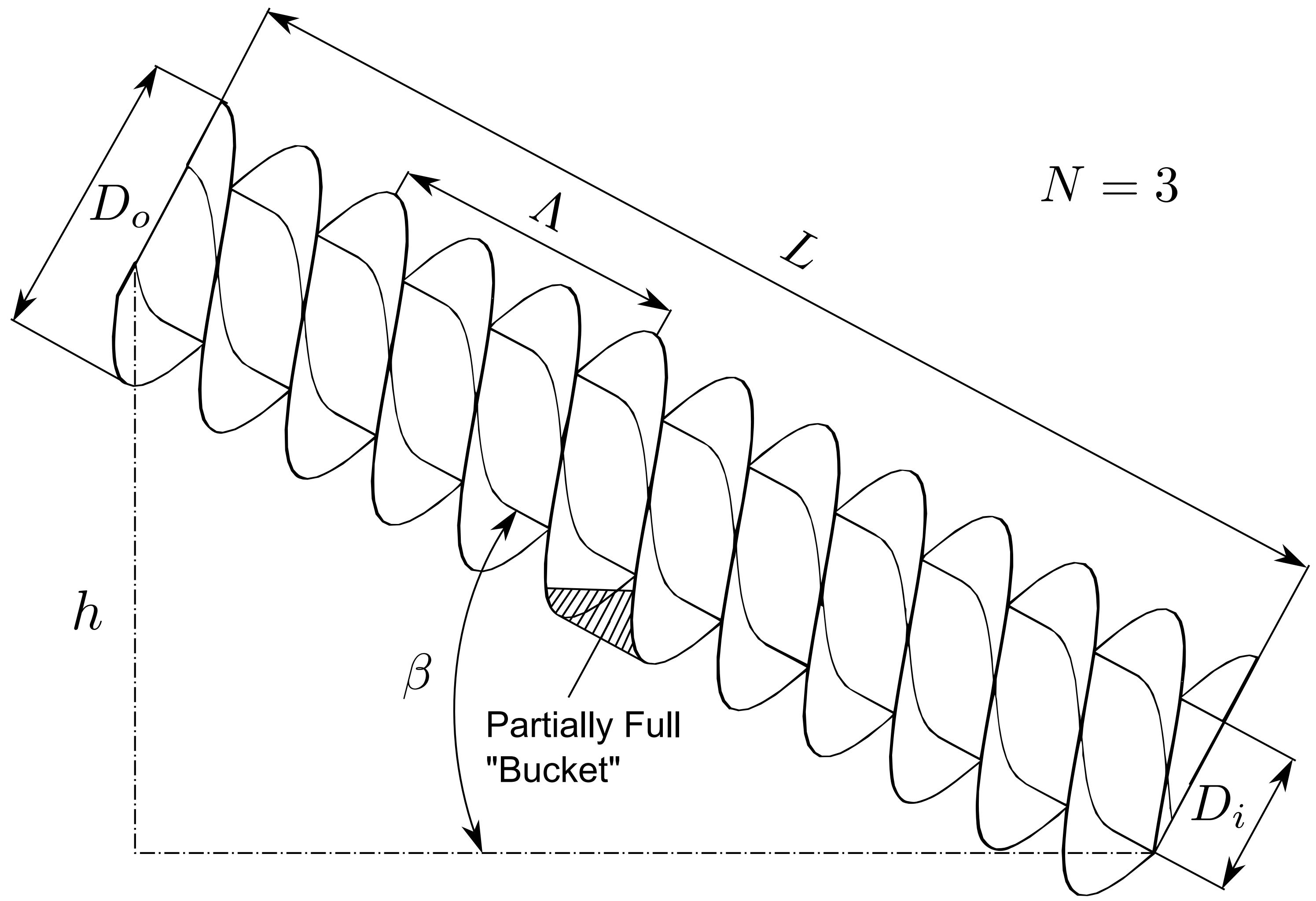
via Canadian Geographic

Bayesian Optimization of Archimedes Screw Turbine

Design parameters:

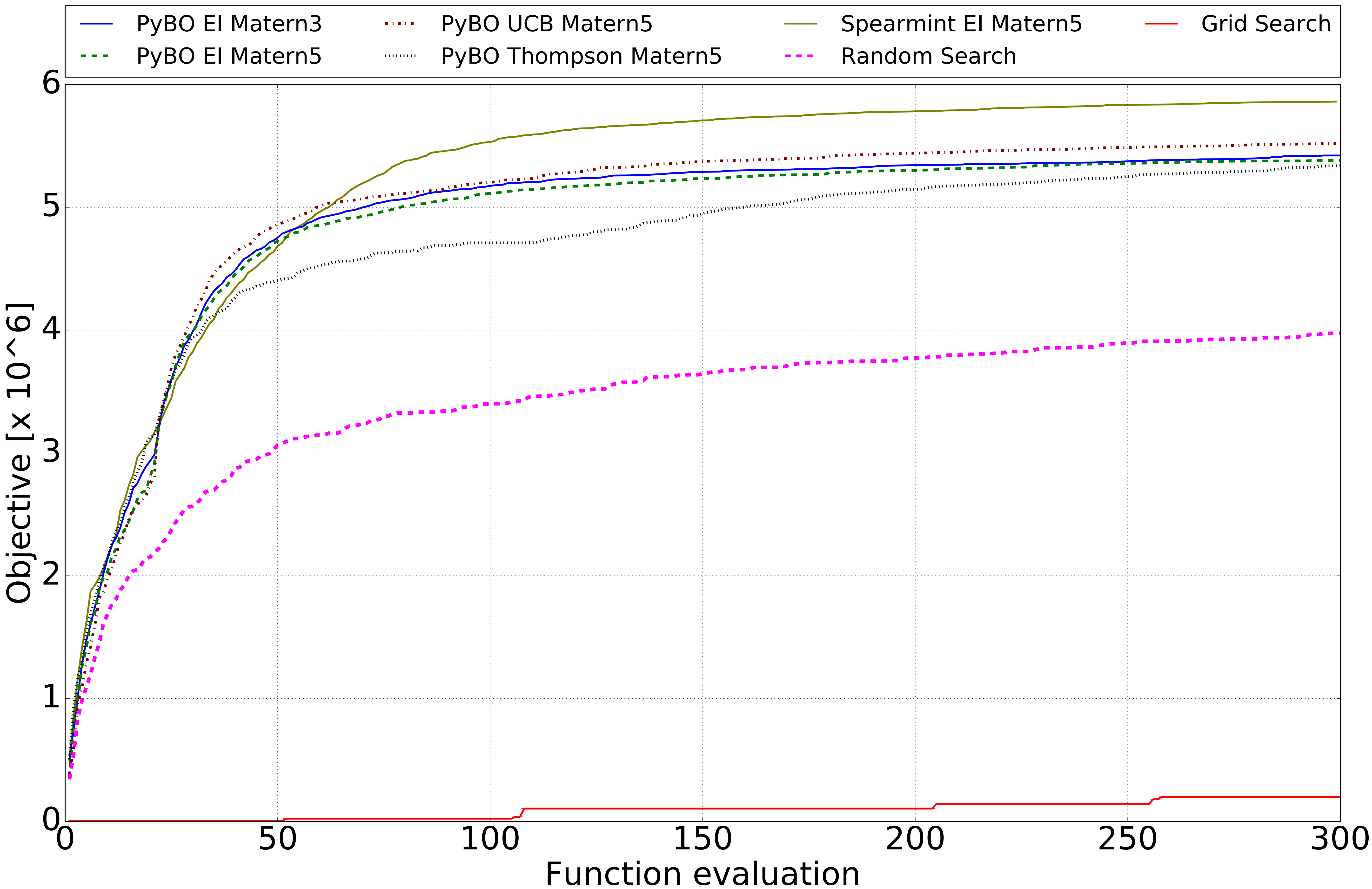
- D_i inner cylinder diameter
- D_o outer cylinder diameter
- h working head
- N number of flights
- β tilt angle of the screw
- Λ pitch
- $f_{\max}$ maximum fill of the bucket

Optimized for power output / mass of steel



GreenBug Energy Inc.

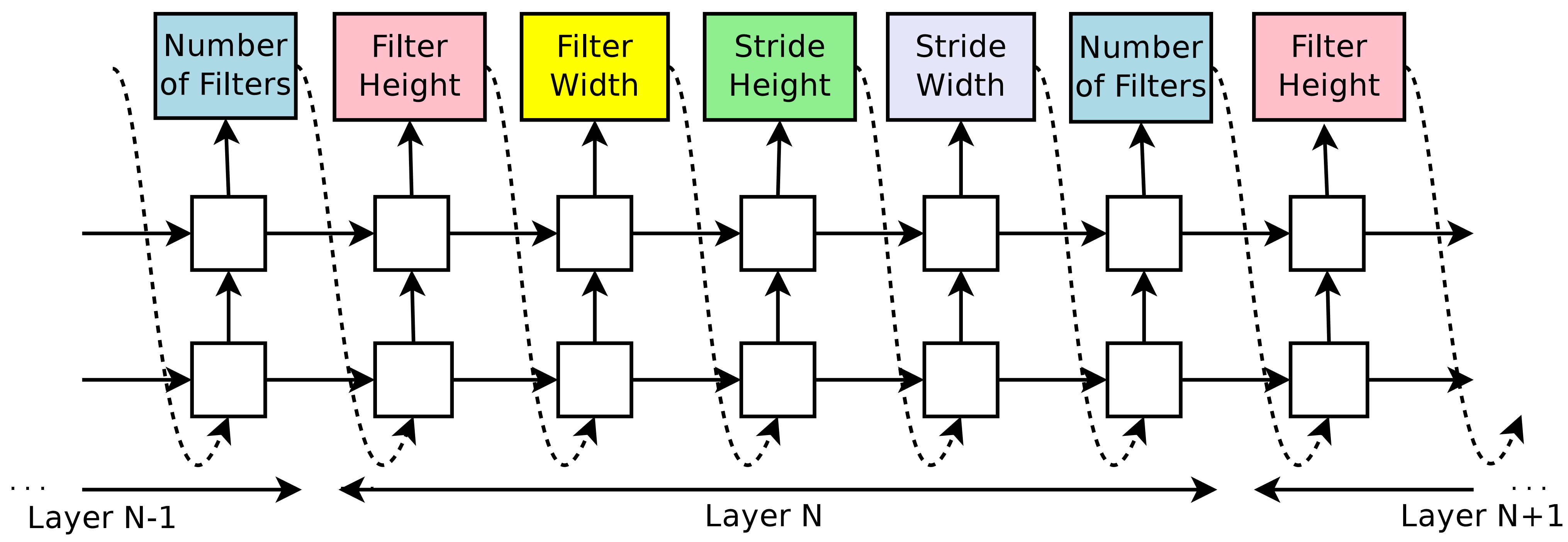
Lisicki, Lubitz, Taylor (2016)



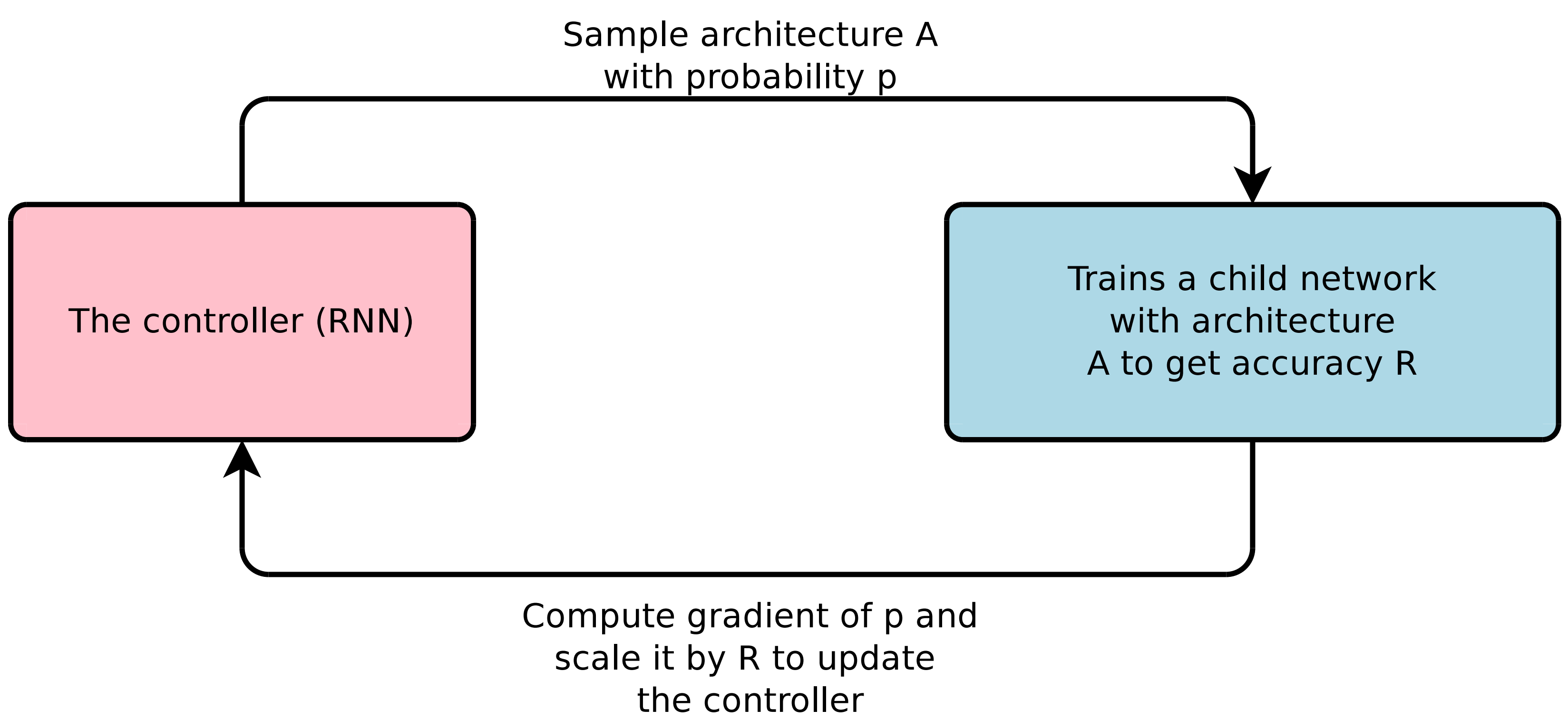
Lisicki, Lubitz, Taylor (2016)

Meta-learning: Architecture

Recurrent neural network (RNN) controller outputs architecture as string



RNN controller trained with reinforcement learning

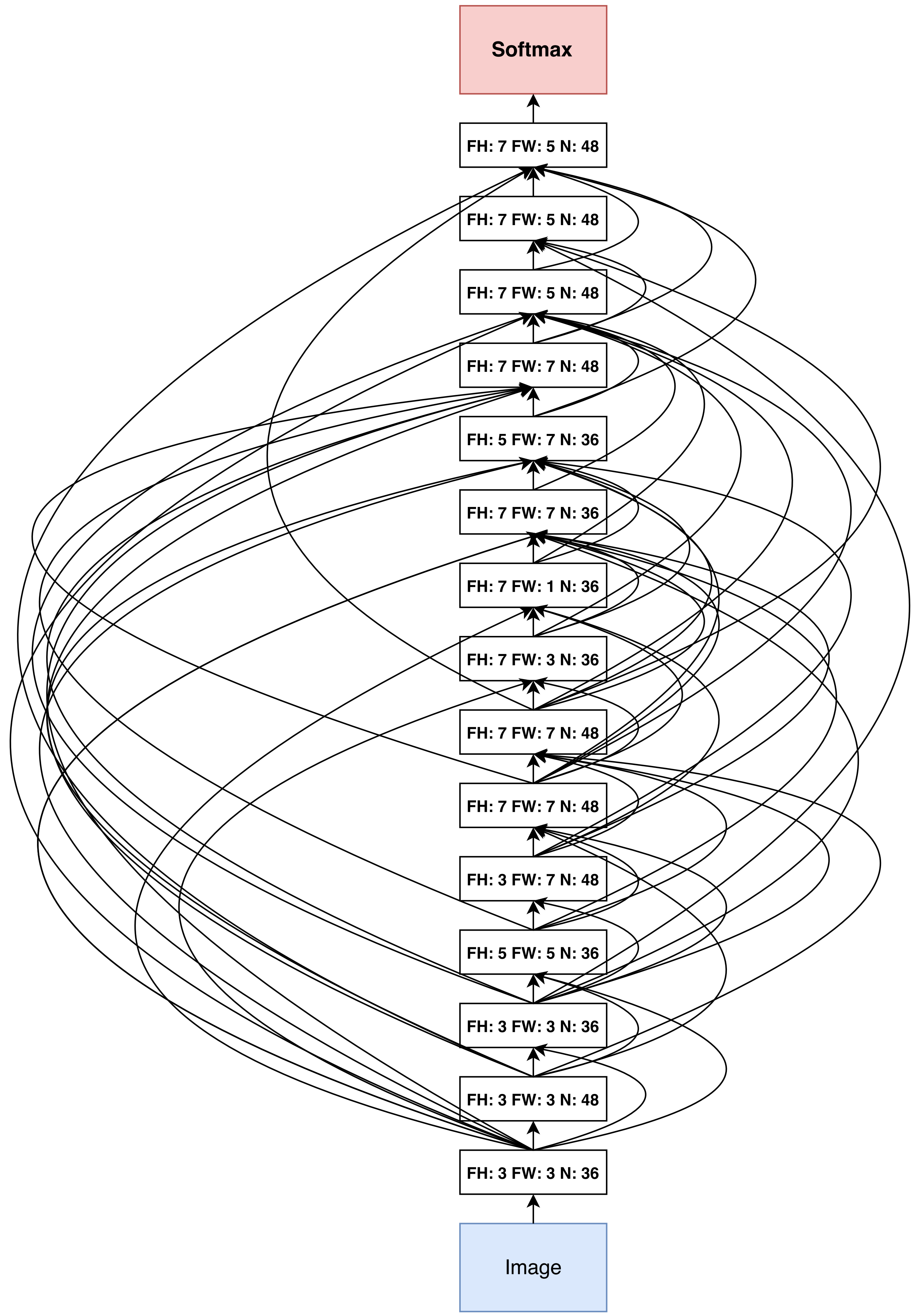


via Zoph and Le (2017)

Discovered CNN Architecture

Error rate on CIFAR

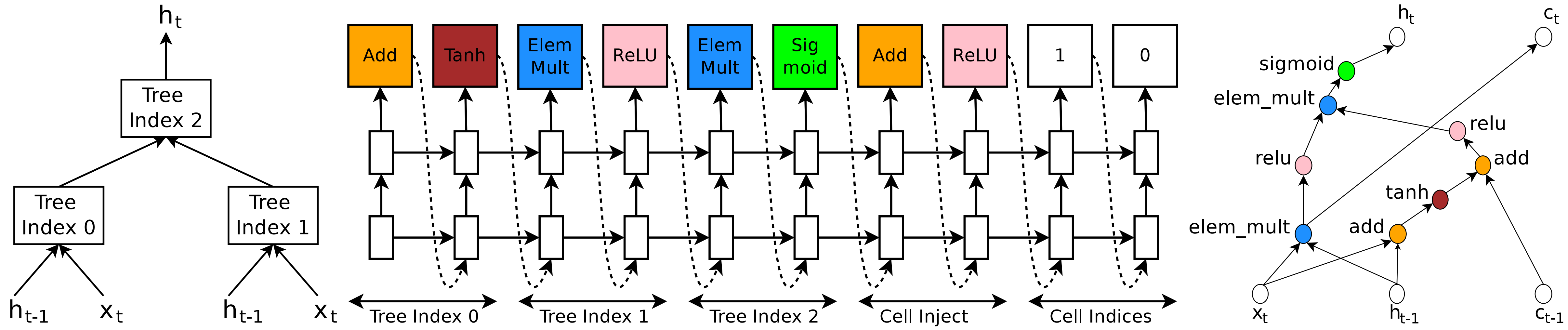
Model	Depth	Parameters	Error rate (%)
Network in Network (Lin et al., 2013)	-	-	8.81
All-CNN (Springenberg et al., 2014)	-	-	7.25
Deeply Supervised Net (Lee et al., 2015)	-	-	7.97
Highway Network (Srivastava et al., 2015)	-	-	7.72
Scalable Bayesian Optimization (Snoek et al., 2015)	-	-	6.37
FractalNet (Larsson et al., 2016)	21	38.6M	5.22
with Dropout/Drop-path	21	38.6M	4.60
ResNet (He et al., 2016a)	110	1.7M	6.61
ResNet (reported by Huang et al. (2016c))	110	1.7M	6.41
ResNet with Stochastic Depth (Huang et al., 2016c)	110	1.7M	5.23
	1202	10.2M	4.91
Wide ResNet (Zagoruyko & Komodakis, 2016)	16	11.0M	4.81
	28	36.5M	4.17
ResNet (pre-activation) (He et al., 2016b)	164	1.7M	5.46
	1001	10.2M	4.62
DenseNet $(L = 40, k = 12)$ Huang et al. (2016a)	40	1.0M	5.24
DenseNet$(L = 100, k = 12)$ Huang et al. (2016a)	100	7.0M	4.10
DenseNet $(L = 100, k = 24)$ Huang et al. (2016a)	100	27.2M	3.74
DenseNet-BC $(L = 100, k = 40)$ Huang et al. (2016b)	190	25.6M	3.46
Neural Architecture Search v1 no stride or pooling	15	4.2M	5.50
Neural Architecture Search v2 predicting strides	20	2.5M	6.01
Neural Architecture Search v3 max pooling	39	7.1M	4.47
Neural Architecture Search v3 max pooling + more filters	39	37.4M	3.65



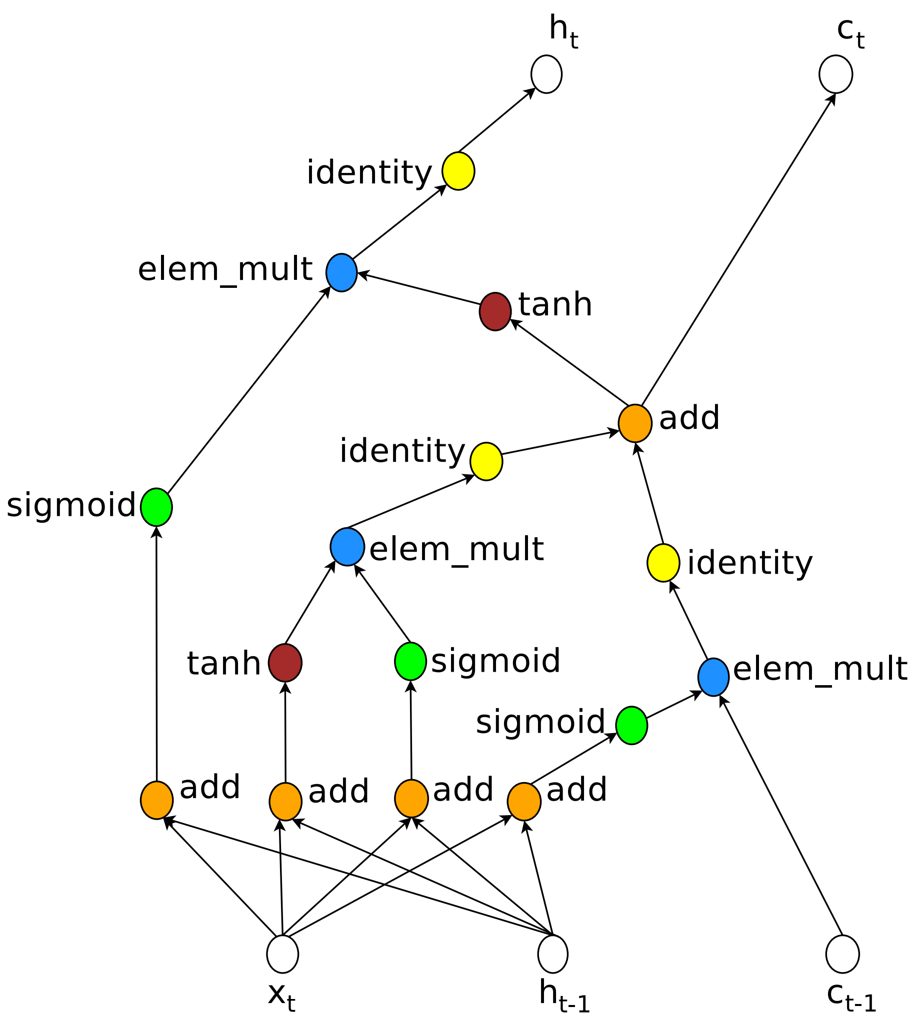
via Zoph and Le (2017)

Learns RNN Cells

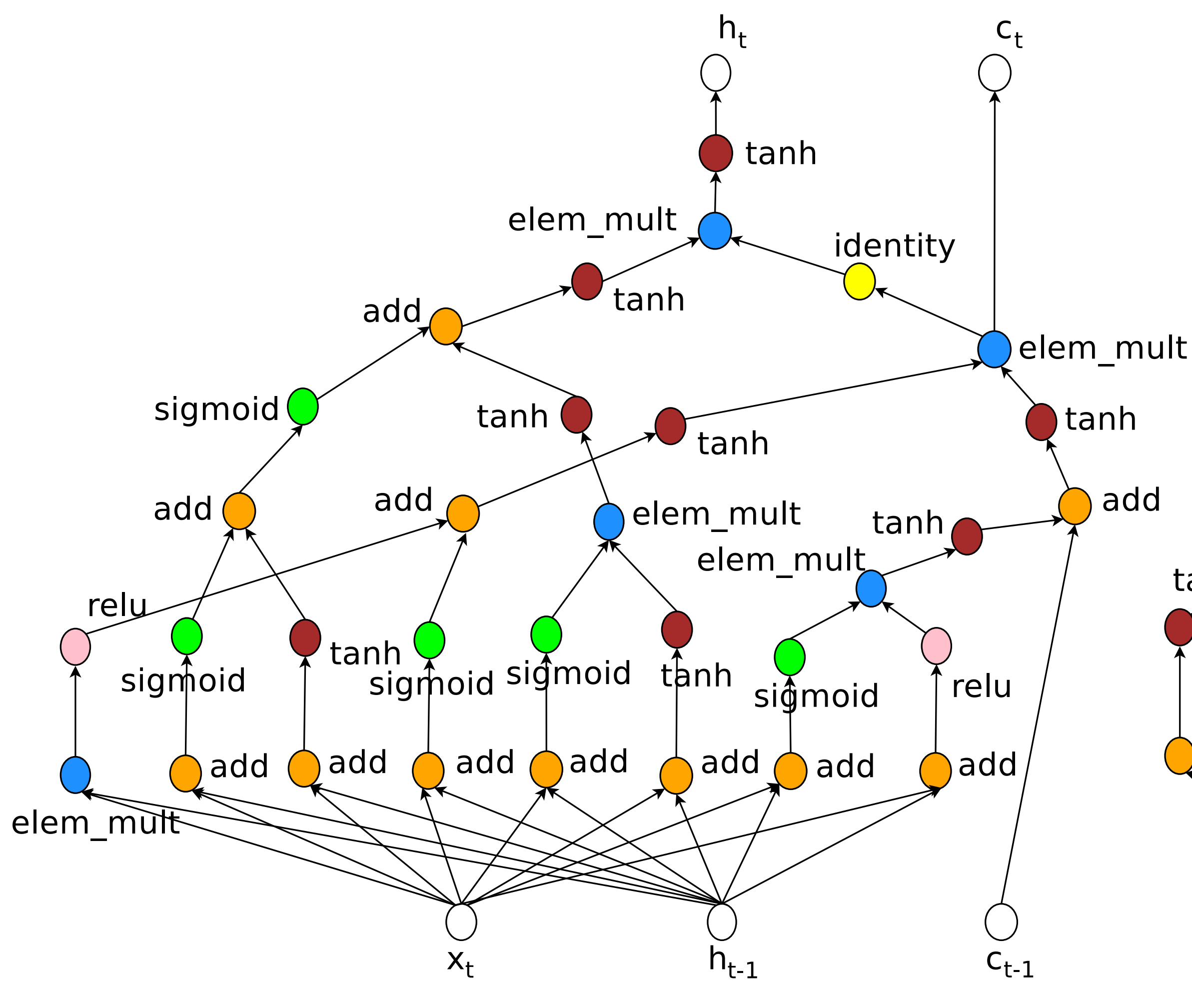
Writing RNN cells as strings



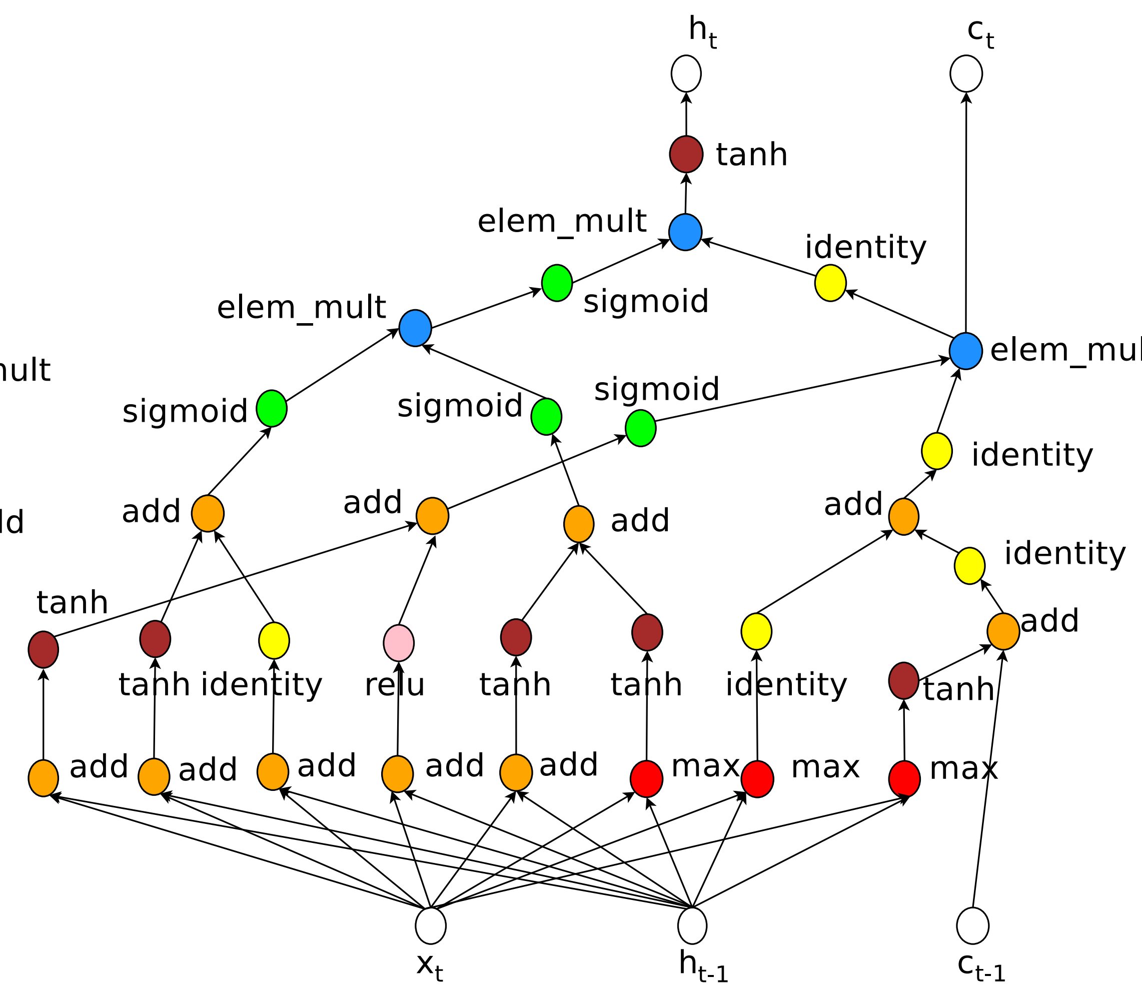
LSTM cell



Found cell (no max or sin)

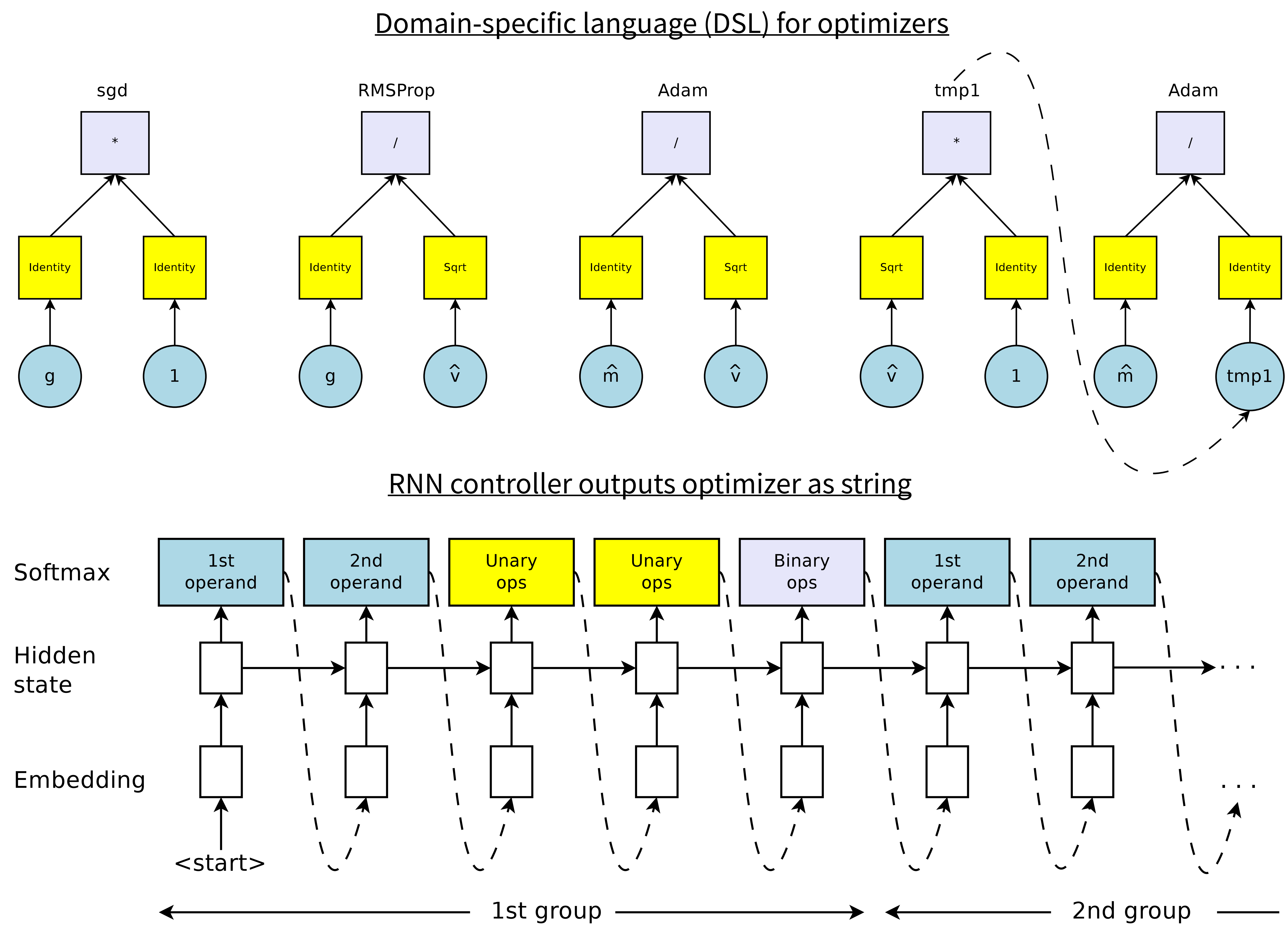


Found cell (max, sin included)



via Zoph and Le (2017)

Meta-learning: Optimizer Design



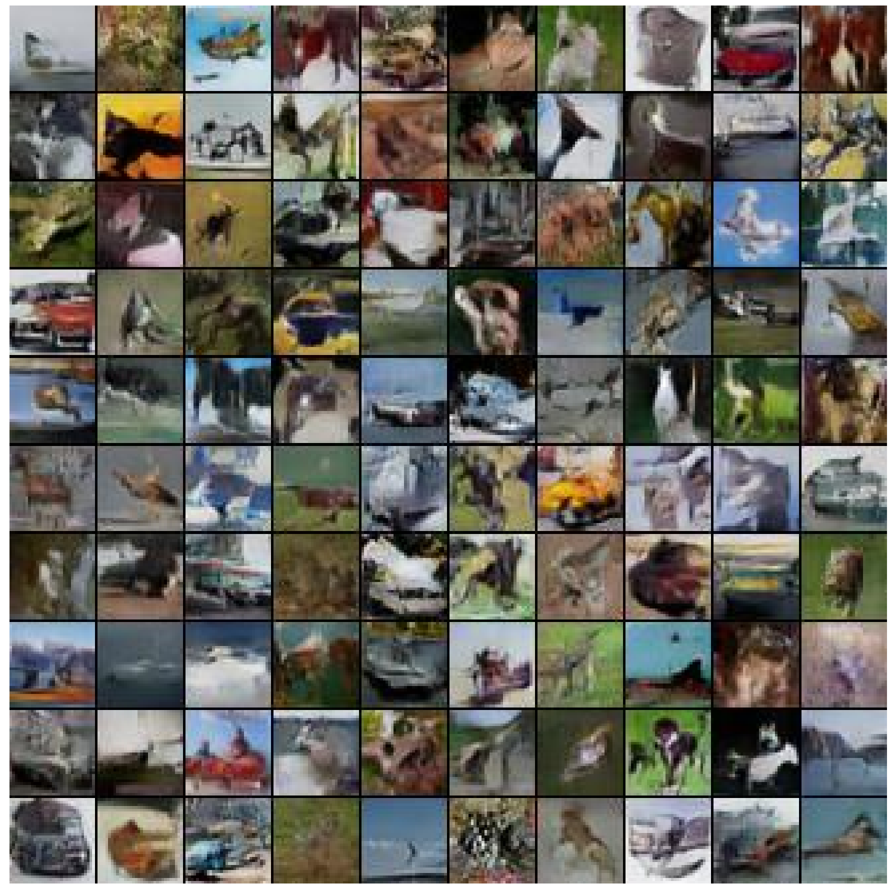
via Zoph and Le (2017)

via medium.com

via NVIDIA

Evaluation?

Inception Score?



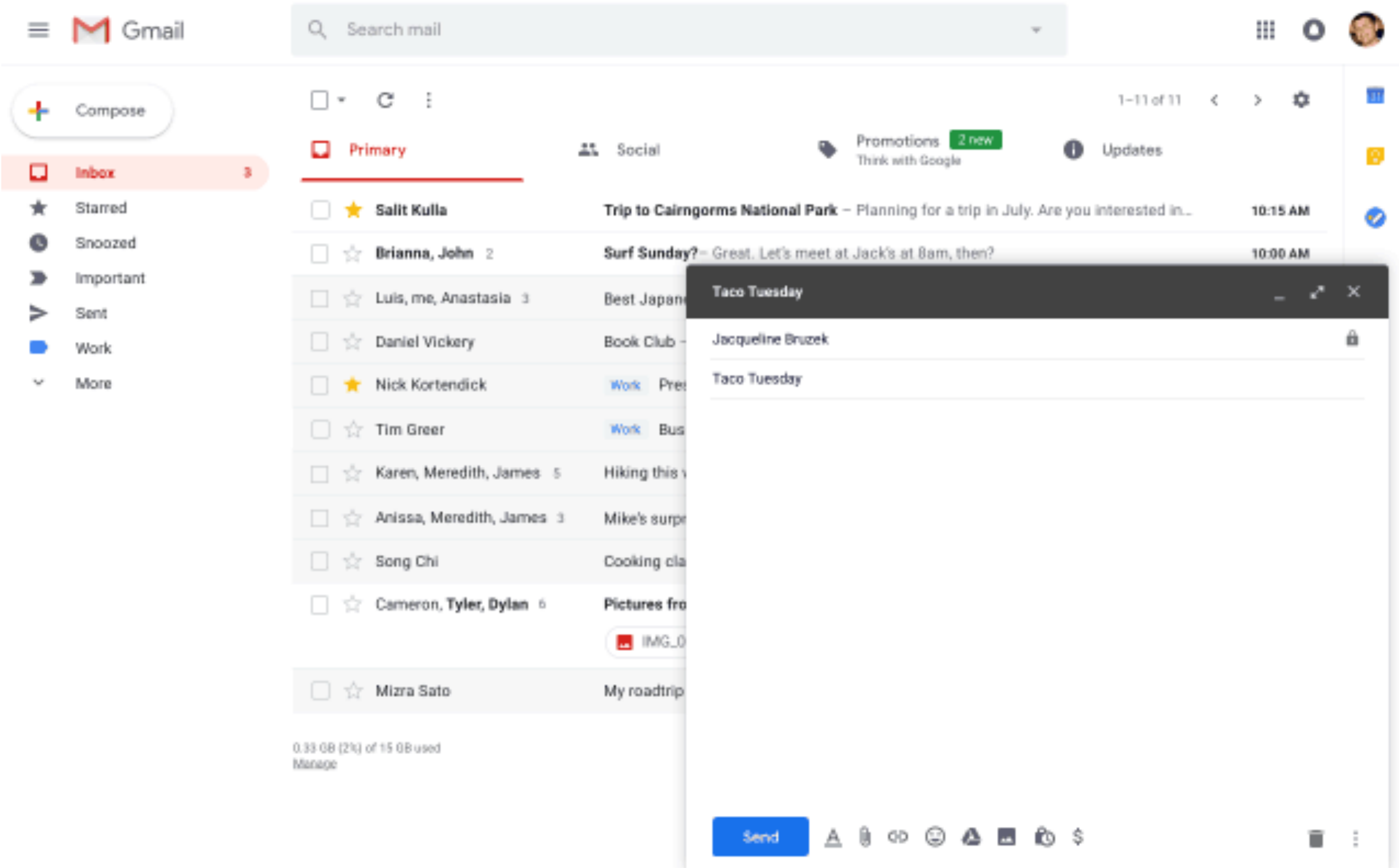
IS = 6.45

IS = 6.31

(Higher is better)

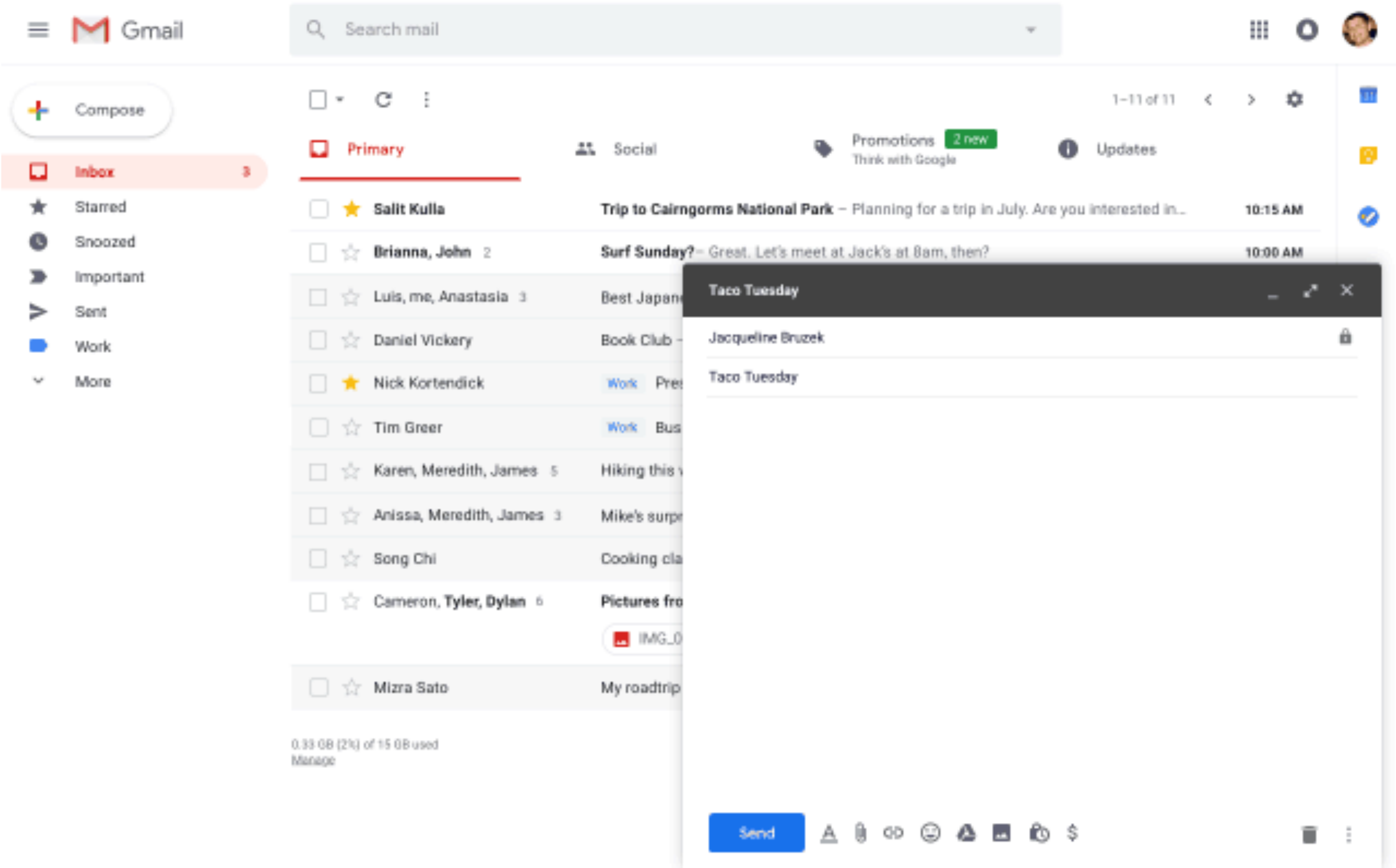
Im, Ma, Taylor, Branson (2018)

Gmail: Smart Compose

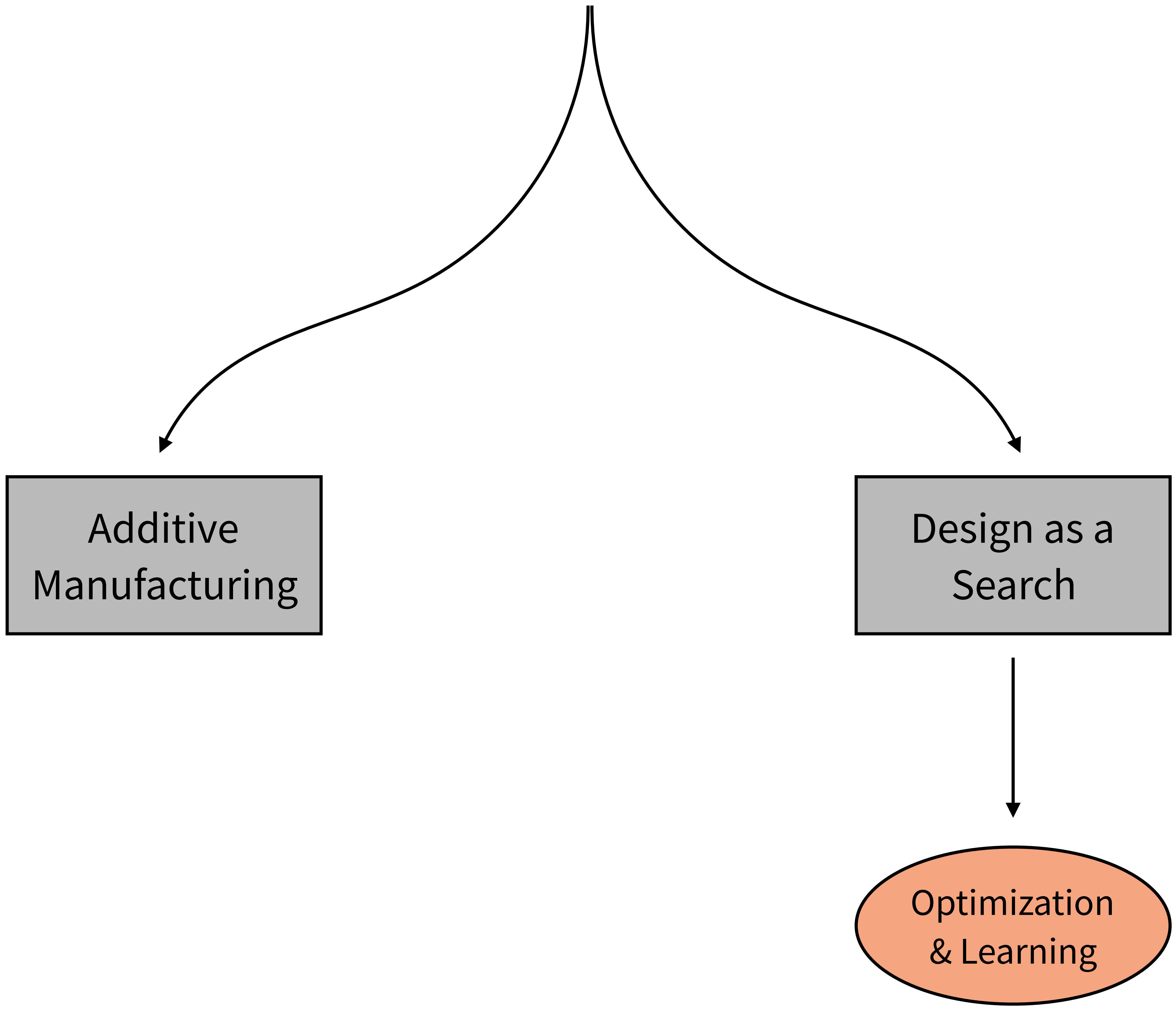


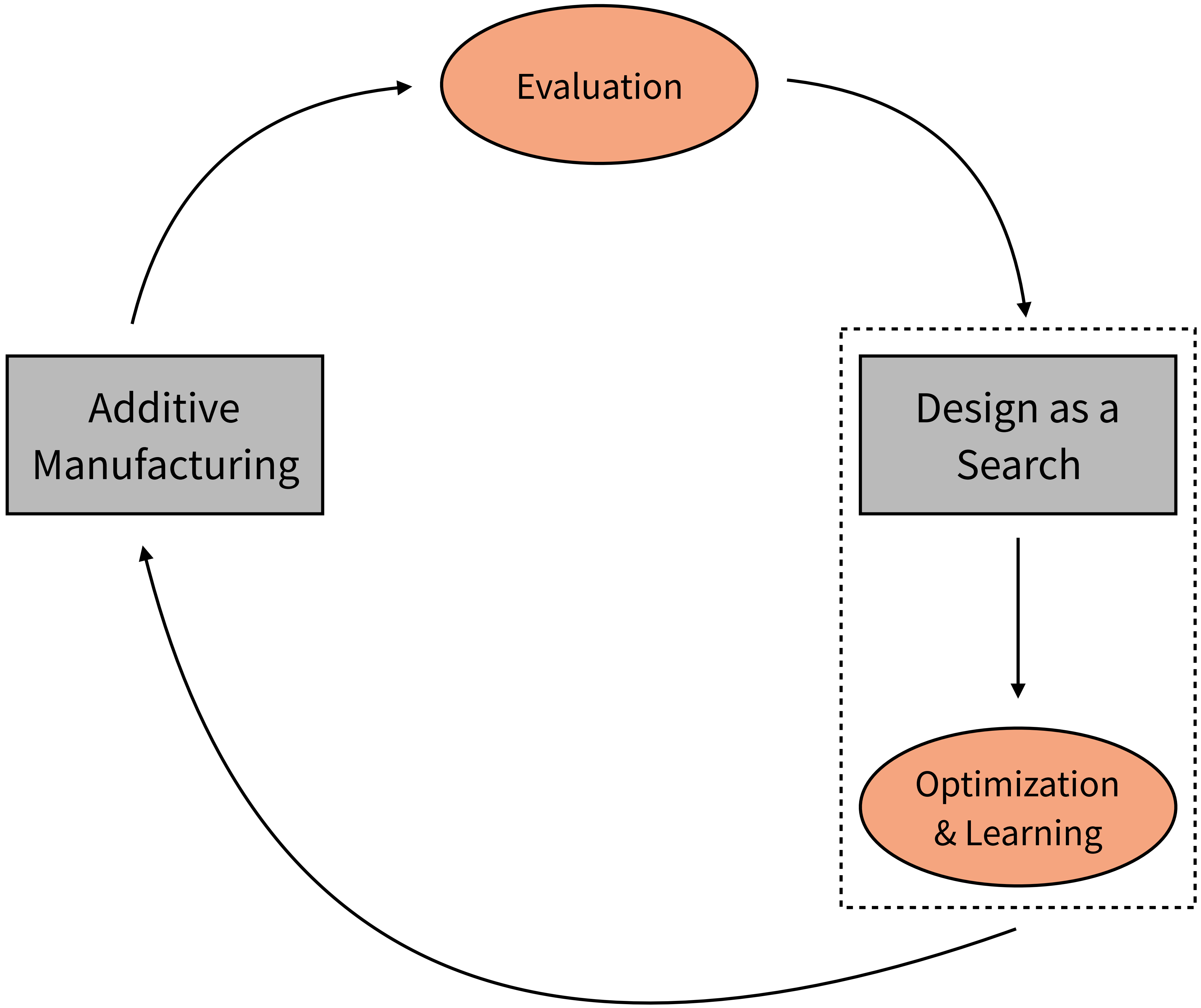
via Google

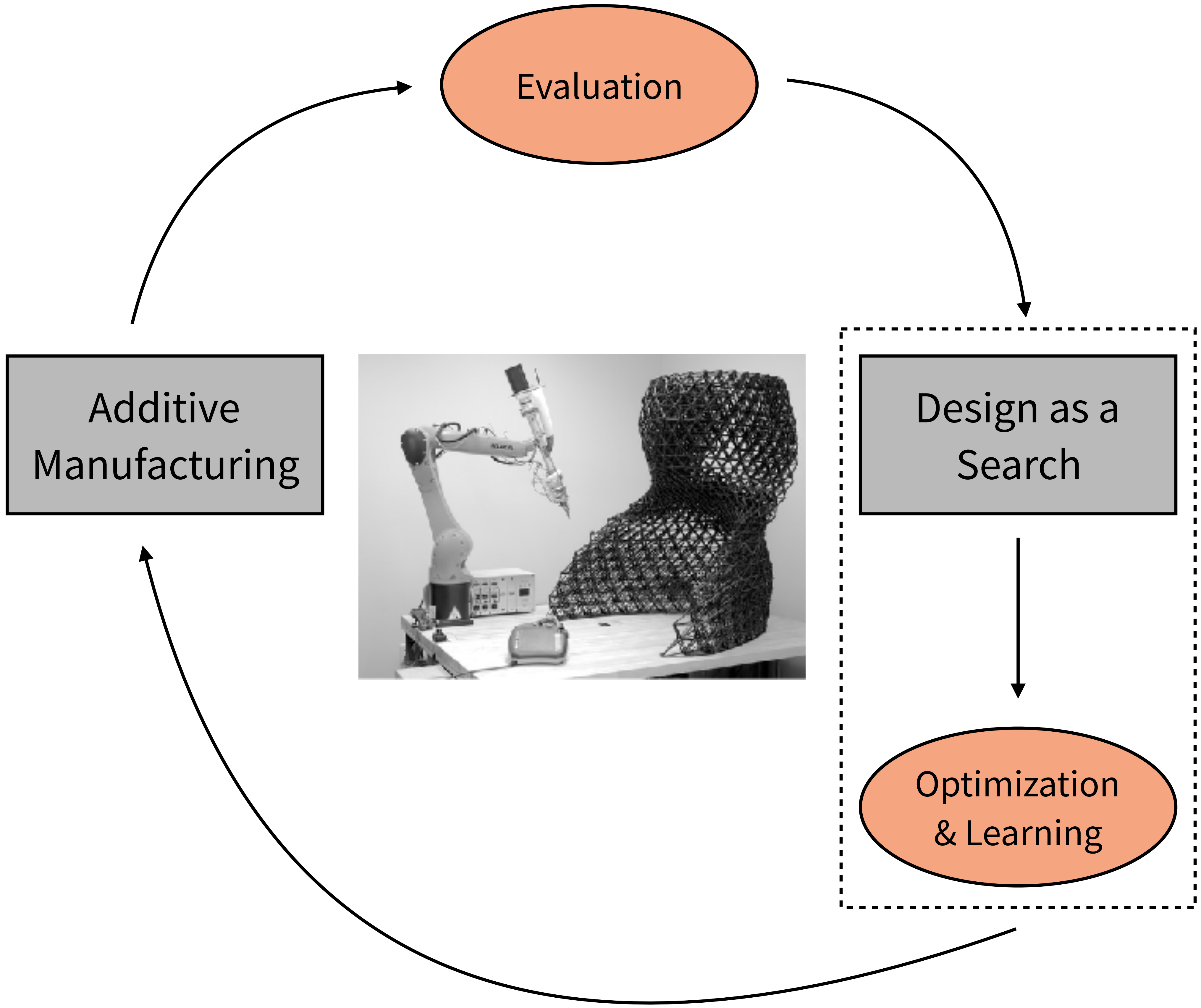
Gmail: Smart Compose



via Google


Additive Manufacturing
Design as a Search
Optimization & Learning


Evaluation
Additive Manufacturing
Design as a Search
Optimization & Learning


Evaluation
Additive Manufacturing
Design as a Search
Optimization & Learning

via nbww.com


Will Martin (U of Guelph)