

Topics:

- Wrap-up:
 - Open directions in Deep Learning

CS 4644-DL / 7643-A
ZSOLT KIRA

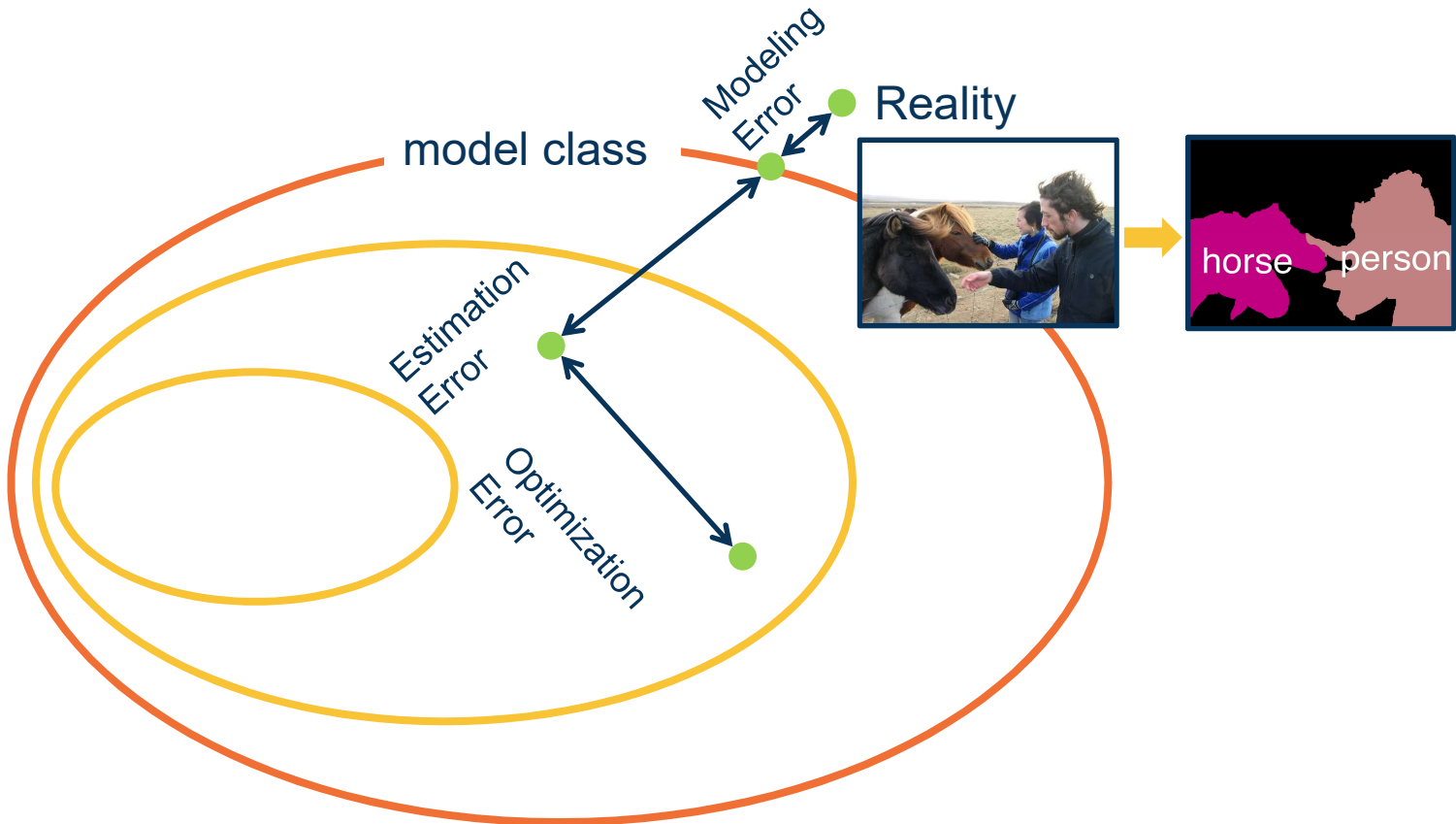
- **Projects!**
 - Due May 1st (May 3rd with grace period)
 - Cannot extend due to grade deadlines!
- **CIOS**
 - Please make sure to fill out! Let us know about things you liked and didn't like in comments so that we can keep or improve!
 - <http://b.gatech.edu/cios>

Some existing works not covered...

- Current / Recent Past
 - Meta-learning
 - AutoML
 - 3D perception
 - Beyond supervised learning: Semi-supervised, domain adaptation, zero/one/few-shot learning
 - Memory (Neural Turing Machines, etc.)
 - Visual question answering, embodied question answering
 - Adversarial Examples
 - Continual/lifelong learning without forgetting
 - World modeling, learning intuitive/physics models
 - Visual dialogue, agents, chatbots
 - Neural Radiance Fields
 - Very Large-Scale Models and what they can do

Few-Shot Learning

VGG19



From: slides by Fei-Fei Li, Justin Johnson, Serena Yeung, CS 231n

Generalization



Supervised Learning

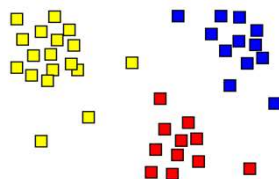
- Train Input: $\{X, Y\}$
- Learning output: $f : X \rightarrow Y, P(y|x)$
- e.g. classification



Sheep
Dog
Cat
Lion
Giraffe

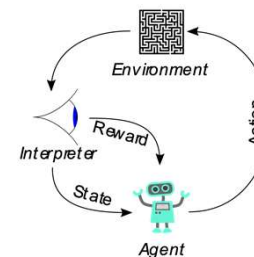
Unsupervised Learning

- Input: $\{X\}$
- Learning output: $P(x)$
- Example: Clustering, density estimation, etc.



Reinforcement Learning

- Evaluative feedback in the form of **reward**
- No supervision on the right action



There is a large number of different low-labeled settings in DL research

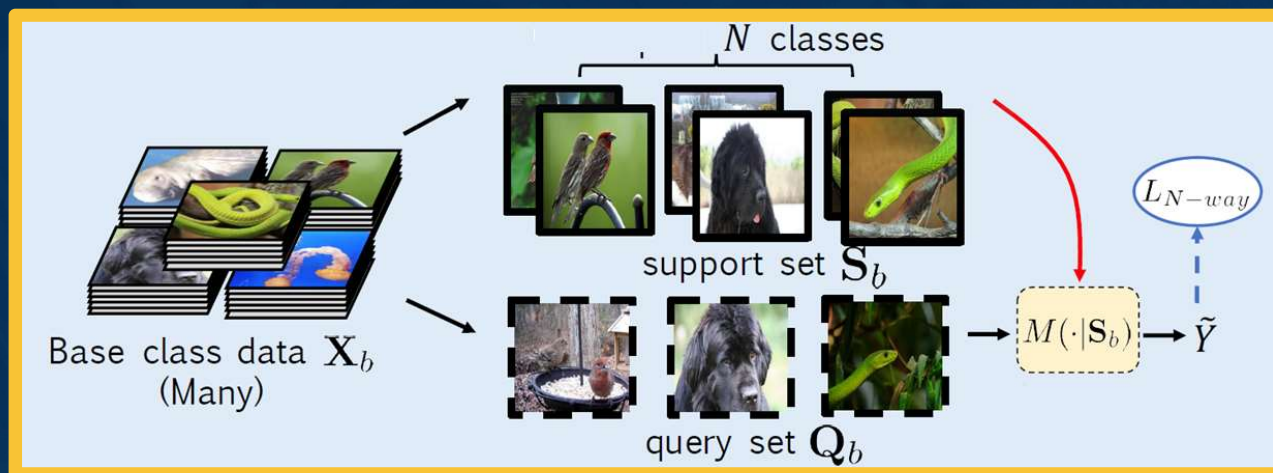
Setting	Source	Target	Shift Type
Semi-supervised	Single labeled	Single unlabeled	None
Domain Adaptation	Single labeled	Single unlabeled	Non-semantic
Domain Generalization	Multiple labeled	Unknown	Non-semantic
Cross-Category Transfer	Single labeled	Single unlabeled	Semantic
Few-Shot Learning	Single labeled	Single few-labeled	Semantic
Un/Self-Supervised	Single unlabeled	Many labeled	Both/Task

Non-Semantic Shift



Semantic Shift





Lots of Labels (Base categories)



Very Few Labels (New categories)

Chen et al., A Closer Look at Few-Shot Learning

Few-Shot Learning



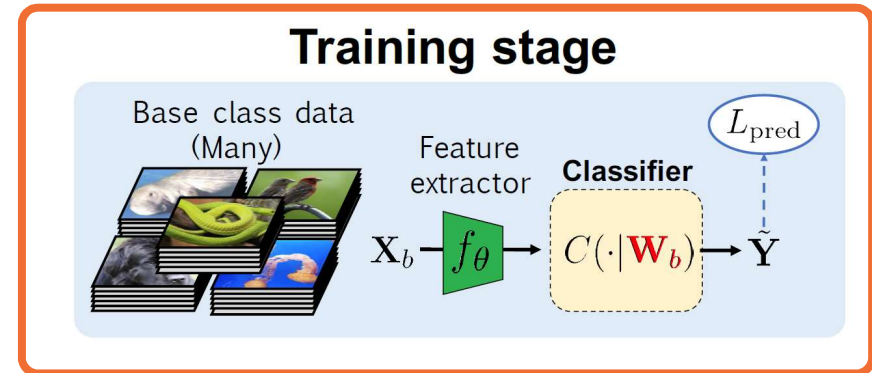
◆ Do what we always do: **Fine-tuning**

◆ Train classifier on base classes

◆ Optionally freeze feature extractor

◆ Learn classifier weights for new classes using few amounts of labeled data (during “query” time)

◆ Surprisingly effective compared to more sophisticated approaches (Chen et al., Dhillon et al., Tian et al.)



Chen et al., A Closer Look at Few-Shot Learning

Dhillon et al., A Baseline for Few-Shot Image Classification

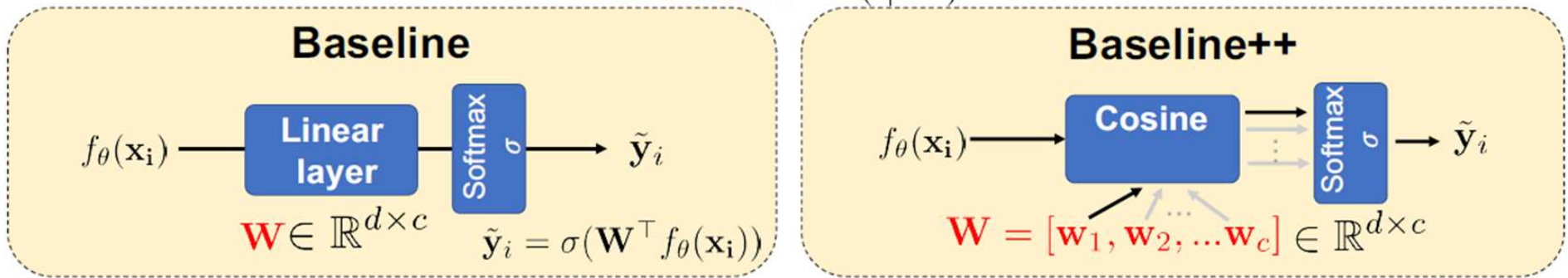
Tian et al., Rethinking Few-Shot Image Classification: a Good Embedding Is All You Need?

Finetuning Baseline



- ◆ We can use a **cosine (similarity-based)** classifier rather than fully connected linear layer

Classifier $C(\cdot | \mathbf{W})$



$$\text{similarity} = \cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\| \|\mathbf{B}\|} = \frac{\sum_{i=1}^n A_i B_i}{\sqrt{\sum_{i=1}^n A_i^2} \sqrt{\sum_{i=1}^n B_i^2}},$$

Chen et al., *A Closer Look at Few-Shot Learning*
https://en.wikipedia.org/wiki/Cosine_similarity

Cons of Normal Approach

- ◆ The training we do on the base classes **does not factor the task into account**
- ◆ No notion that we will be **performing a bunch of N-way tests**
- ◆ **Idea:** simulate what we will see during test time



Set up a set of **smaller tasks** during training which **simulates** what we will be doing during **testing: N-Way K-Shot Tasks**

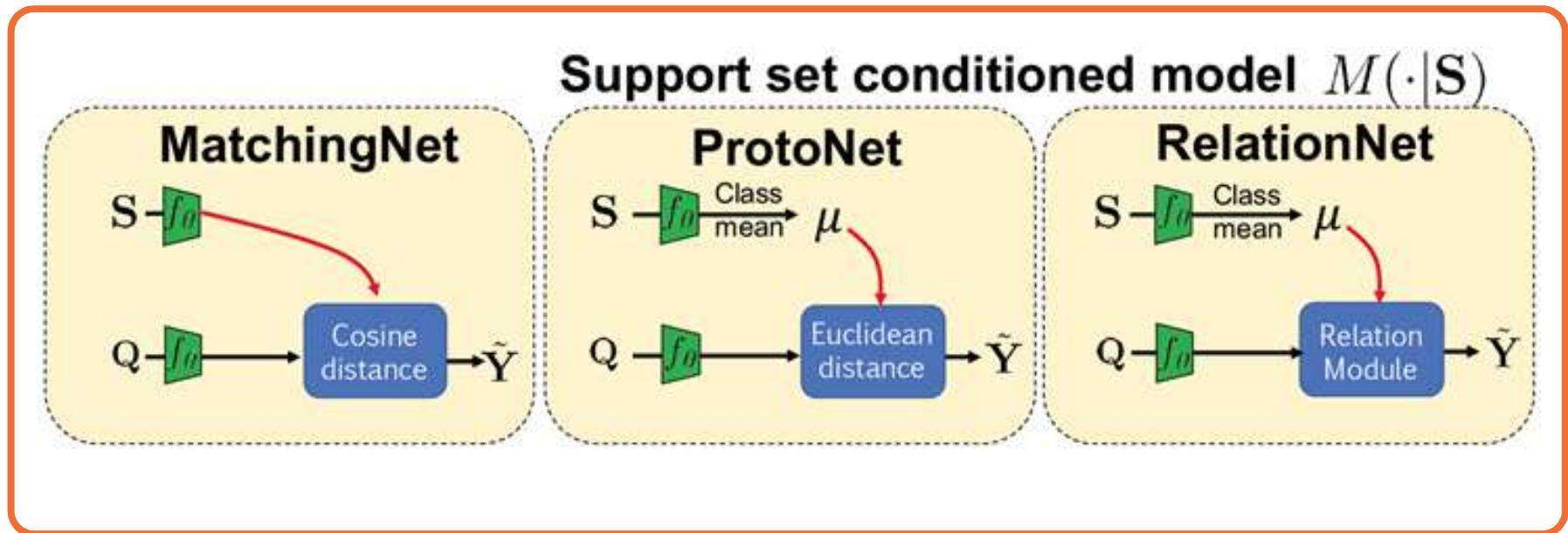


◆ Can optionally pre-train features on held-out base classes

Testing stage is now the same, but with new classes

Meta-Training

Learning a model conditioned on support set $M(\cdot|S)$



Chen et al., A Closer Look at Few-Shot Learning

Approaches using Meta-Training

How to parametrize learning algorithms?

Two approaches to defining a meta-learner:

- **Take inspiration from a known learning algorithm**
 - kNN/kernel machine: Matching networks (Vinyals et al. 2016)
 - Gaussian classifier: Prototypical Networks (Snell et al. 2017)
 - Gradient Descent: Meta-Learner LSTM (Ravi & Larochelle, 2017) , Model-Agnostic Meta-Learning MAML (Finn et al. 2017)
- **Derive it from a black box neural network**
 - MANN (Santoro et al. 2016)
 - SNAIL (Mishra et al. 2018)

Slide Credit: Hugo Larochelle

Learn gradient descent:

- Parameter initialization and update rules
- **Output:**
 - Parameter initialization
 - Meta-learner that decides *how* to update parameters

Learn just an initialization and use normal gradient descent (MAML)

- **Output:**
 - Just parameter initialization!
 - We are using SGD

Slide Credit: Hugo Larochelle

More Sophisticated Meta-Learning Approaches



- Training a “**gradient descent procedure**” applied on some learner M
 - gradient descent starts from some initial parameters θ_0 and then performs the following updates:

$$\theta_t = \theta_{t-1} - \alpha_t \nabla_{\theta_{t-1}} \mathcal{L}_t$$

- Optimization as a Model for Few-Shot Learning (2017)
Sachin Ravi and Hugo Larochelle

Slide Credit: Hugo Larochelle

- Training a “**gradient descent procedure**” applied on some learner M
 - gradient descent starts from some initial parameters θ_0 and then performs the following updates:

$$\theta_t = \theta_{t-1} - \alpha_t \nabla_{\theta_{t-1}} \mathcal{L}_t$$

- this is quite similar to LSTM cell state updates:

$$c_t = f_t \odot c_{t-1} + i_t \odot \tilde{c}_t$$

- Optimization as a Model for Few-Shot Learning (2017)
Sachin Ravi and Hugo Larochelle

Slide Credit: Hugo Larochelle

- Training a “**gradient descent procedure**” applied on some learner M
 - gradient descent starts from some initial parameters θ_0 and then performs the following updates:

$$\theta_t = \theta_{t-1} - \alpha_t \nabla_{\theta_{t-1}} \mathcal{L}_t$$

- this is quite similar to LSTM cell state updates:

$$c_t = f_t \odot c_{t-1} + i_t \odot \tilde{c}_t$$

- state c_t is model M 's parameter space θ_t ← c_0 becomes a *learned initialization*

- Optimization as a Model for Few-Shot Learning (2017)
Sachin Ravi and Hugo Larochelle

Slide Credit: Hugo Larochelle

- Training a “**gradient descent procedure**” applied on some learner M
 - gradient descent starts from some initial parameters θ_0 and then performs the following updates:

$$\theta_t = \theta_{t-1} - \alpha_t \nabla_{\theta_{t-1}} \mathcal{L}_t$$

- this is quite similar to LSTM cell state updates:

$$c_t = f_t \odot c_{t-1} + i_t \odot \tilde{c}_t$$

- state c_t is model M 's parameter space θ_t ← c_0 becomes a *learned initialization*
- state update \tilde{c}_t is the negative gradient $-\nabla_{\theta_{t-1}} \mathcal{L}_t$

- Optimization as a Model for Few-Shot Learning (2017)
Sachin Ravi and Hugo Larochelle

Slide Credit: Hugo Larochelle

- Training a “**gradient descent procedure**” applied on some learner M
 - gradient descent starts from some initial parameters θ_0 and then performs the following updates:

$$\theta_t = \theta_{t-1} - \alpha_t \nabla_{\theta_{t-1}} \mathcal{L}_t$$

- this is quite similar to LSTM cell state updates:

$$c_t = f_t \odot c_{t-1} + i_t \odot \tilde{c}_t$$

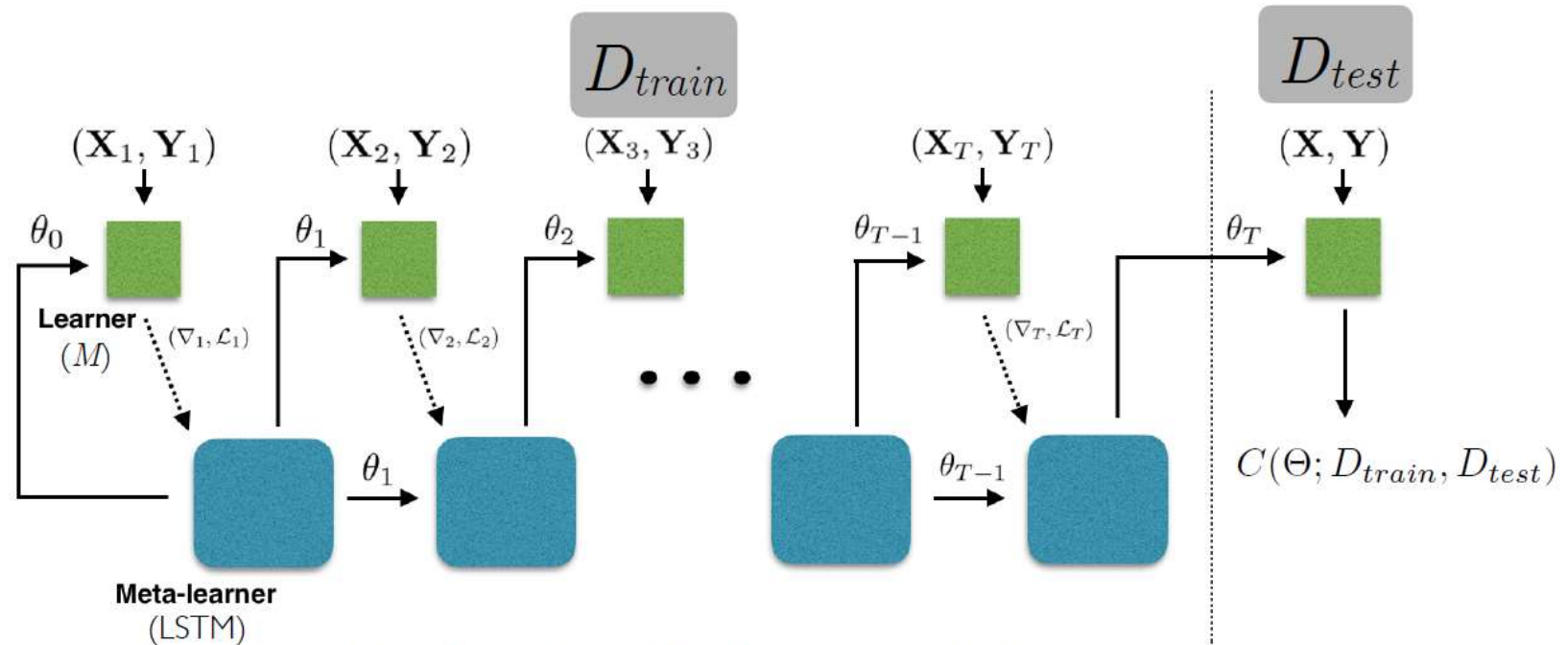
- state c_t is model M 's parameter space θ_t ← c_0 becomes a learned initialization
- state update \tilde{c}_t is the negative gradient $-\nabla_{\theta_{t-1}} \mathcal{L}_t$
- f_t and i_t are LSTM gates:
 - $i_t = \sigma(\mathbf{W}_I \cdot [\nabla_{\theta_{t-1}} \mathcal{L}_t, \mathcal{L}_t, \theta_{t-1}, i_{t-1}] + \mathbf{b}_I)$ ← adaptive learning rate
 - $f_t = \sigma(\mathbf{W}_F \cdot [\nabla_{\theta_{t-1}} \mathcal{L}_t, \mathcal{L}_t, \theta_{t-1}, f_{t-1}] + \mathbf{b}_F)$ ← adaptive weight decay

- Optimization as a Model for Few-Shot Learning (2017)

Sachin Ravi and Hugo Larochelle

Slide Credit: Hugo Larochelle

- Training a “**gradient descent procedure**” applied on some learner M



- Optimization as a Model for Few-Shot Learning (2017)

Sachin Ravi and Hugo Larochelle

Slide Credit: Hugo Larochelle

Meta-Learner LSTM

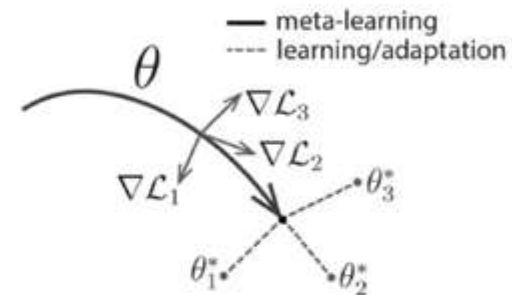
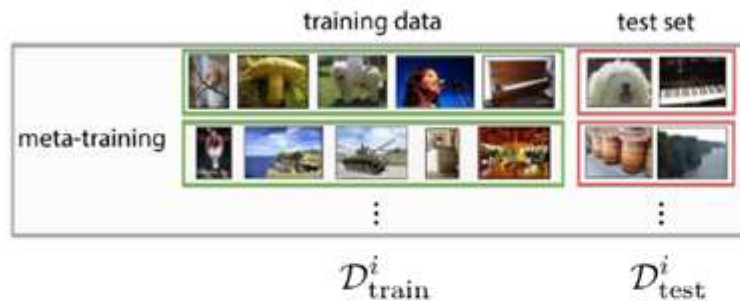
- Training a “**gradient descent procedure**” applied on some learner M
 - MAML proposes not to bother with training an LSTM for the gradient descent updates and constant step-size updates

- Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks (2017)
Chelsea Finn, Pieter Abbeel and Sergey Levine

Slide Credit: Hugo Larochelle

Model-Agnostic Meta-Learning (MAML)

a general recipe:



$$\theta \leftarrow \theta - \beta \sum_i \nabla_{\theta} \underbrace{\mathcal{L}(\theta - \alpha \nabla_{\theta} \mathcal{L}(\theta, \mathcal{D}_{\text{train}}^i), \mathcal{D}_{\text{test}}^i)}_{\text{"meta-loss" for task } i}$$

* in general, can take more than one gradient step here

** we often use 4 – 10 steps

Finn et al., "Model-Agnostic Meta-Learning"

Slide Credit: Hugo Larochelle

Model-Agnostic Meta-Learning (MAML)

supervised learning: $f(x) \rightarrow y$

supervised meta-learning: $f(\mathcal{D}_{\text{train}}, x) \rightarrow y$

model-agnostic meta-learning: $f_{\text{MAML}}(\mathcal{D}_{\text{train}}, x) \rightarrow y$

$$f_{\text{MAML}}(\mathcal{D}_{\text{train}}, x) = f_{\theta'}(x)$$

$$\theta' = \theta - \alpha \sum_{(x,y) \in \mathcal{D}_{\text{train}}} \nabla_{\theta} \mathcal{L}(f_{\theta}(x), y)$$

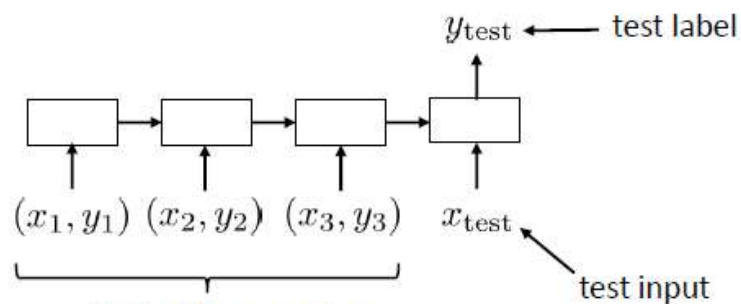
Just another computation graph...

Can implement with any autodiff package (e.g., TensorFlow)

Slide Credit: Hugo Larochelle

Model-Agnostic Meta-Learning (MAML)

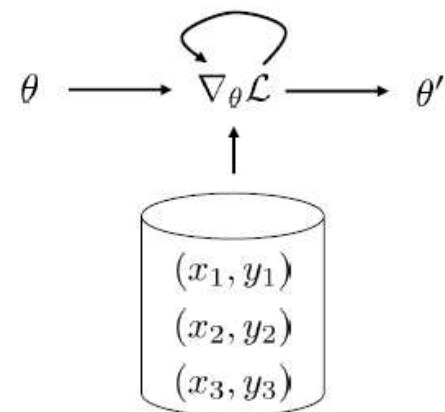
RNN-based meta-learning



this implements the
“learned learning algorithm”

- Does it converge?
 - Kind of?
- What does it converge to?
 - Who knows...
- What to do if it's not good enough?
 - Nothing...

MAML



- Does it converge?
 - Yes (it's gradient descent...)
- What does it converge to?
 - A local optimum (it's gradient descent...)
- What to do if it's not good enough?
 - Keep taking gradient steps (it's gradient descent...)

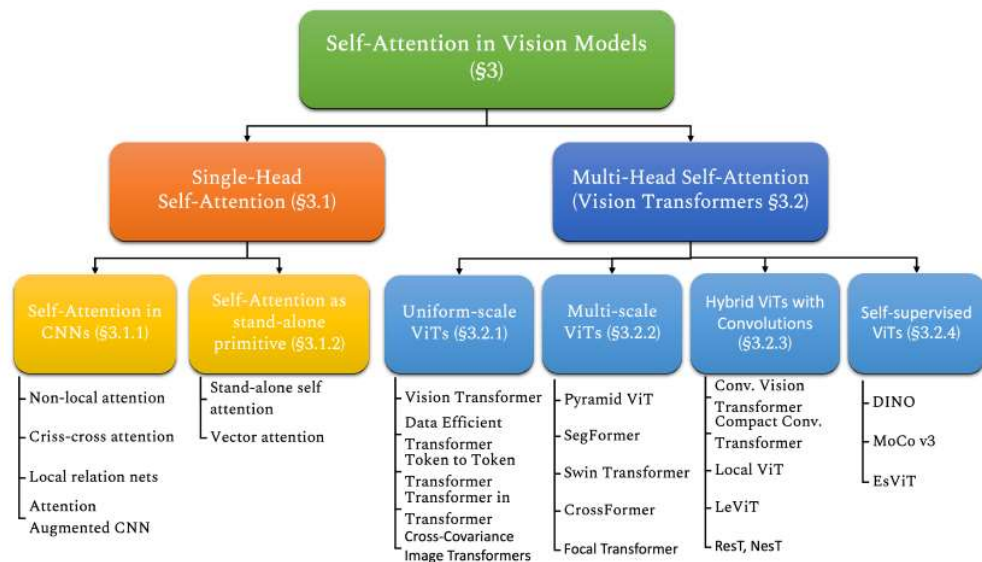
Slide Credit: Hugo Larochelle

Comparison

Wrap-up

Some current/upcoming topics

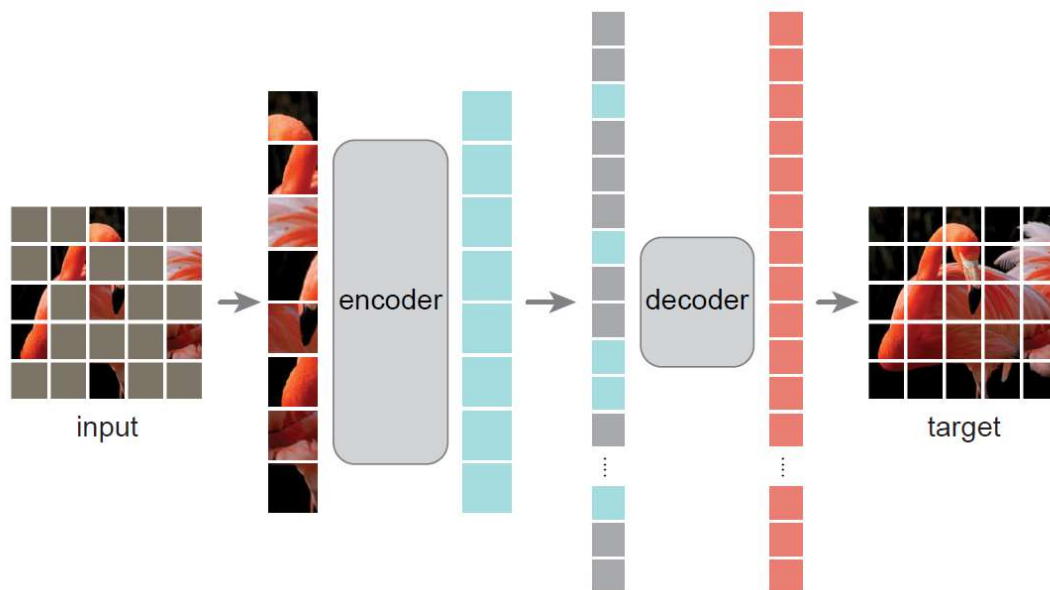
- More recent
 - Transformers for vision, audio, etc.
 - Fixing reinforcement learning
 - First you have to admit you have a problem
 - Simulation frameworks, joint perception, planning, and action
 - Navigation, mapping
 - Use of large-scale multi-modal models (CLIP)
 - Neural radiance fields (NeRF)
 - Uncertainty quantification, robustness
 - Deep Learning and logic!
 - Just scaling everything up and watch the magic!
 - Especially multi-modal, multi-task problems
 - Bias, fairness



- Transformers are extremely flexible
- Vision transformers, multi-modal transformers, etc.

Khan et al., Transformers in Vision: A Survey

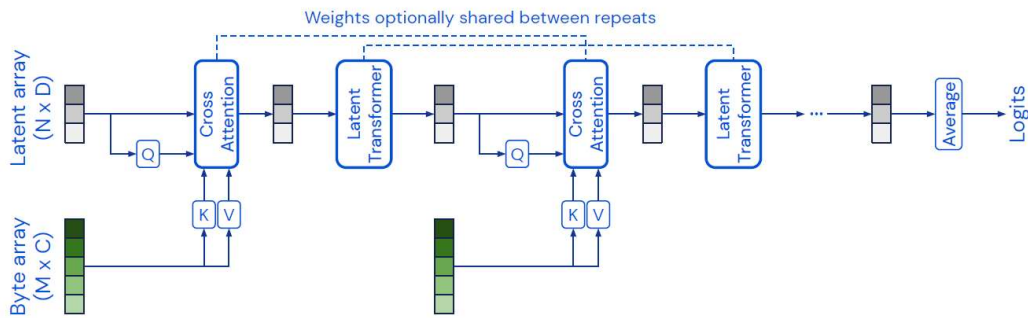
Transformers



- Transformers are extremely flexible
- Masking is a powerful concept for self-supervised learning

He et al., Masked Autoencoders Are Scalable Vision Learners

Self-Supervised Transformers



- Unified models for all Modalities

Figure 1. The Perceiver is an architecture based on attentional principles that scales to high-dimensional inputs such as images, videos, audio, point-clouds, and multimodal combinations without making domain-specific assumptions. The Perceiver uses a cross-attention module to project an high-dimensional input byte array to a fixed-dimensional latent bottleneck (the number of input indices M is much larger than the number of latent indices N) before processing it using a deep stack of Transformer-style self-attention blocks in the latent space. The Perceiver iteratively attends to the input byte array by alternating cross-attention and latent self-attention blocks.



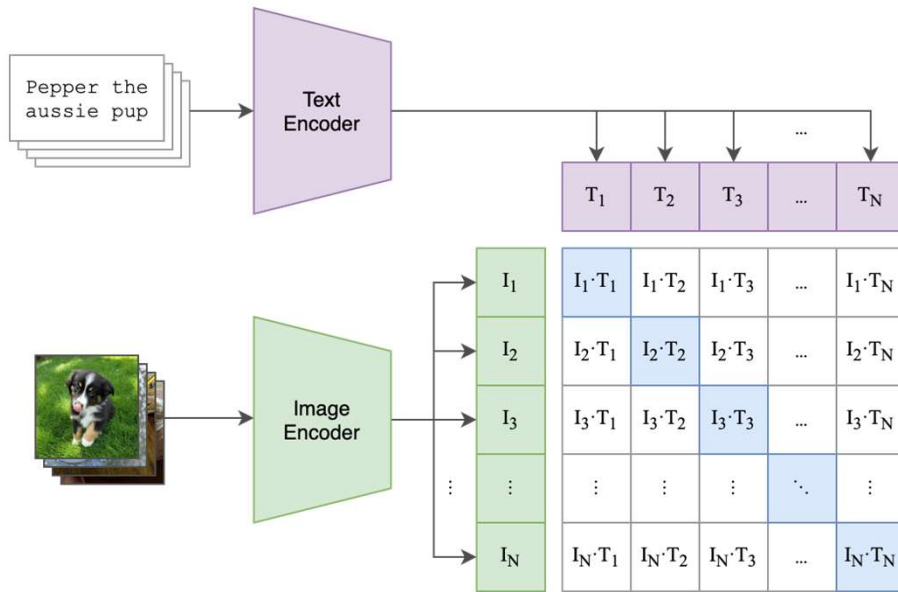
Figure 2. We train the Perceiver architecture on images from ImageNet (Deng et al., 2009) (left), video and audio from AudioSet (Gemmeke et al., 2017) (considered both multi- and uni-modally) (center), and 3D point clouds from ModelNet40 (Wu et al., 2015) (right). Essentially no architectural changes are required to use the model on a diverse range of input data.

Jaegle et al., Perceiver: General Perception with Iterative Attention

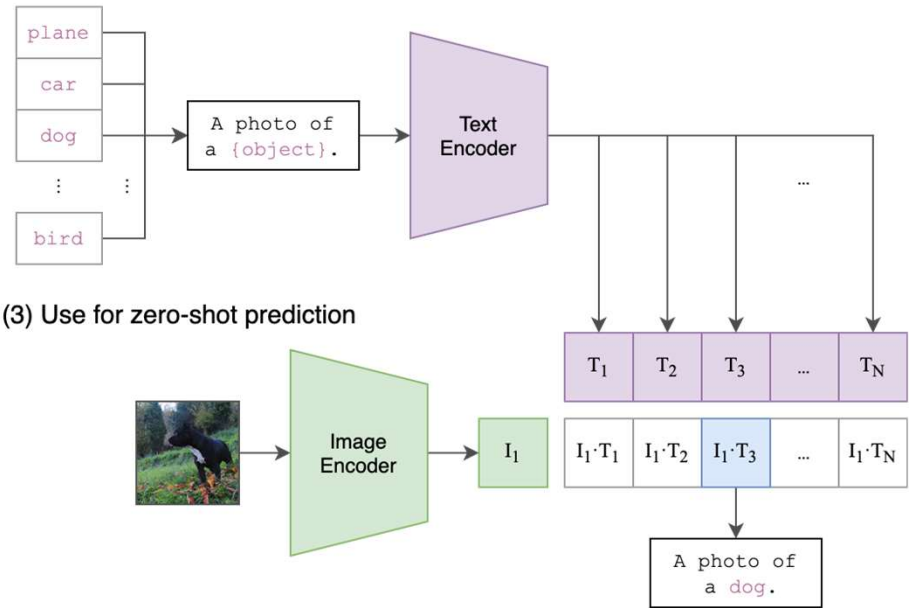
Unified Transformer Models



(1) Contrastive pre-training



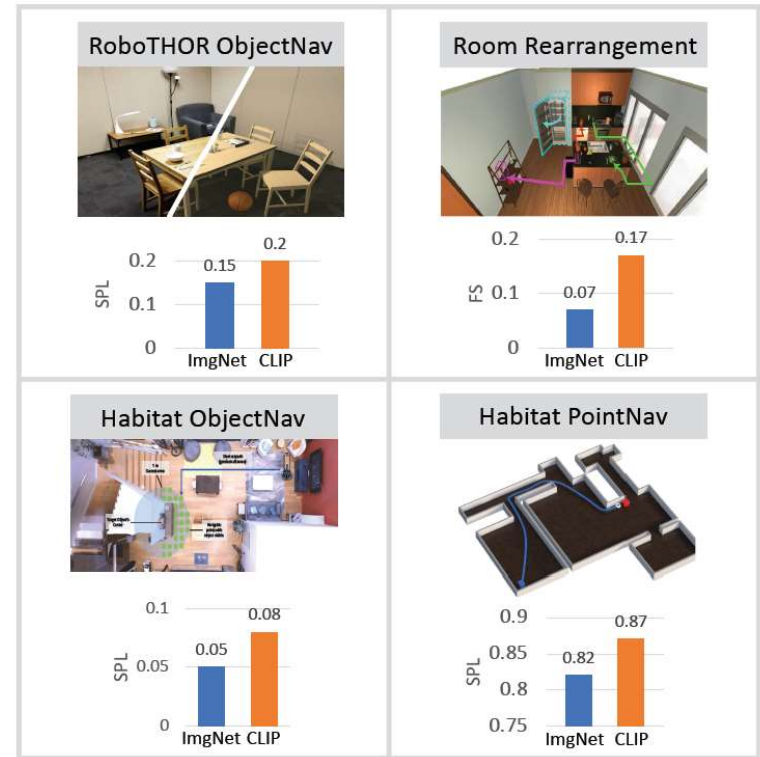
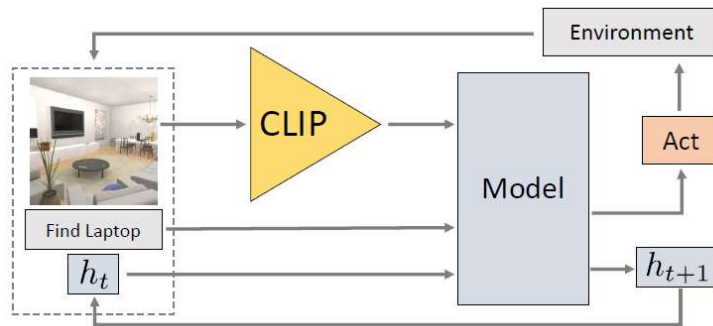
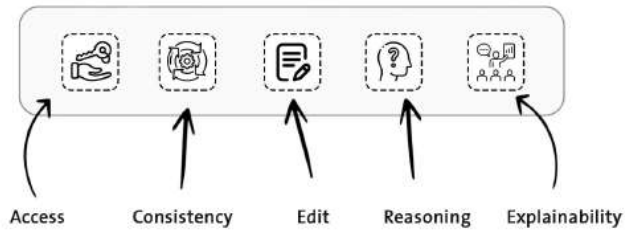
(2) Create dataset classifier from label text



Radford et al., Learning Transferable Visual Models From Natural Language Supervision

Multi-Modal Large-Scale Models

LMs-as-KBs

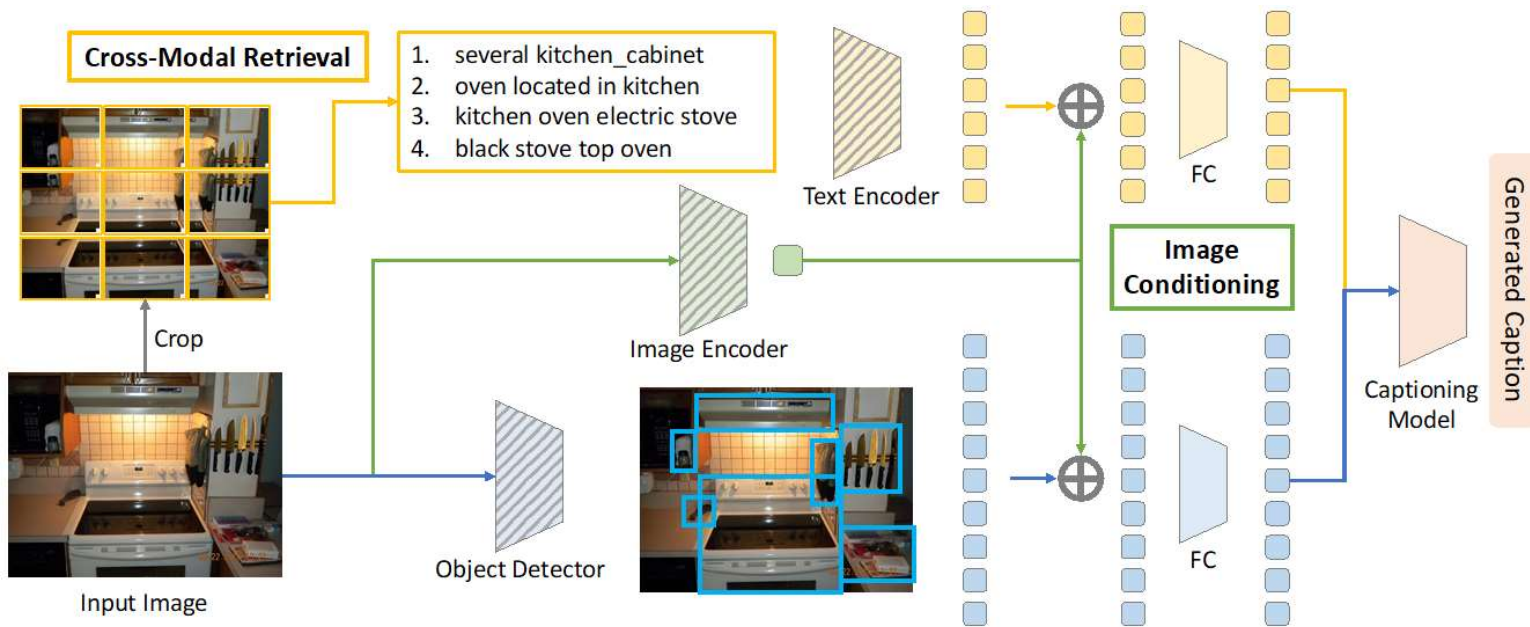


AlKhamissi et al., A Review on Language Models as Knowledge Bases
 Khandelwal et al., Simple but Effective: CLIP Embeddings for Embodied AI

Extracting Knowledge from Large-Scale Models



- Q: The frozen, pre-trained object detector may not encode all necessary info of X.
- A: Use CLIP to retrieve a set of text descriptions to provide complementary info.
- Q: Conditional relationship $P(O|X)$ is not jointly optimized with the target VL task.
- A: Add a simple, trainable MLP between f_o and f_x to model **the conditional probability of $P(O|X)$ based on CLIP.**



Kuo & Kira, Beyond a Pre-Trained Object Detector: Cross-Modal Textual and Visual Context for Image Captioning, CVPR 2022

Extracting Knowledge from Large-Scale Models

DALL-E 2 can create original, realistic images and art from a text description. It can combine concepts, attributes, and styles.

TEXT DESCRIPTION

An astronaut Teddy bears A bowl of soup

riding a horse lounging in a tropical resort
in space playing basketball with cats in
space

in a photorealistic style in the style of Andy
Warhol as a pencil drawing



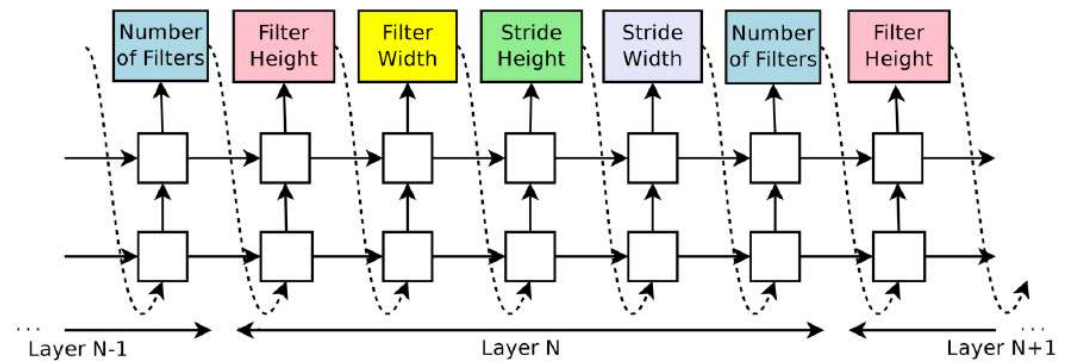
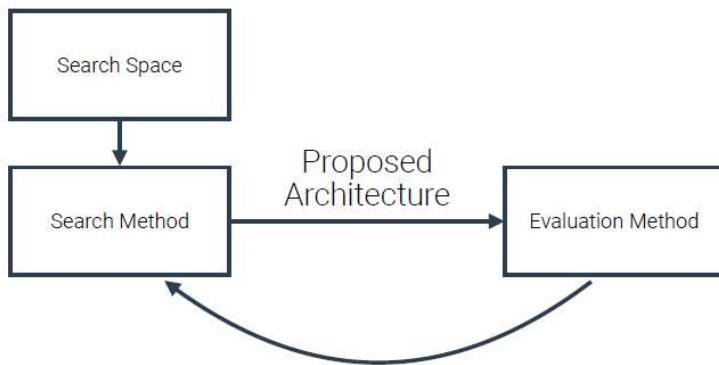
DALL-E 2



<https://openai.com/dall-e-2/>

DALL-E



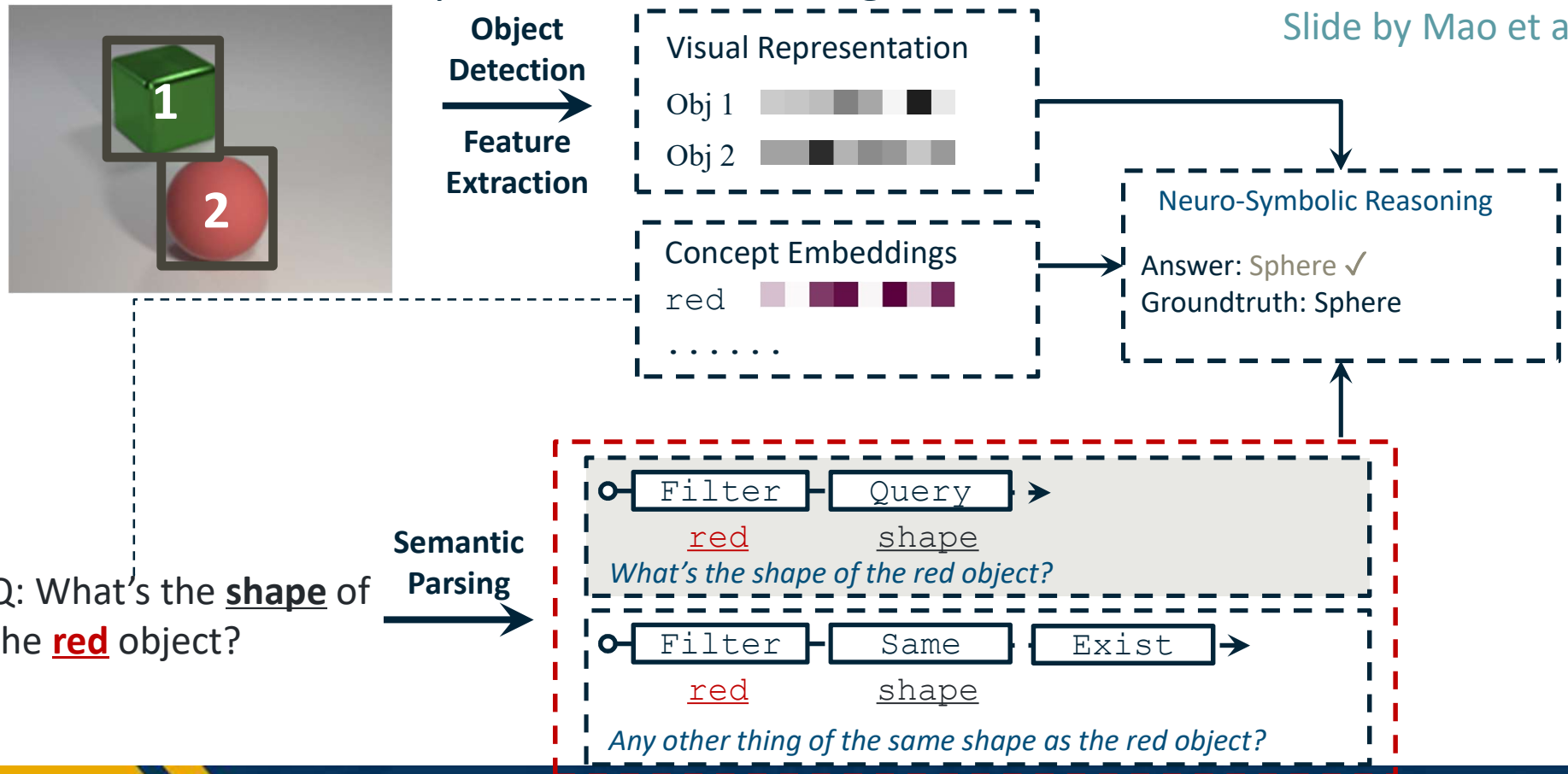


Slides by Erik Wijmans

Architecture Search

Concepts Facilitate Parsing New Sentences

Slide by Mao et al.



Q: What's the shape of the red object?

Semantic Parsing

Filter Query
red shape
What's the shape of the red object?

Filter Same Exist
red shape
Any other thing of the same shape as the red object?

Neuro-Symbolic Reasoning
Answer: Sphere ✓
Groundtruth: Sphere

Neuro-Symbolic Concept Learning

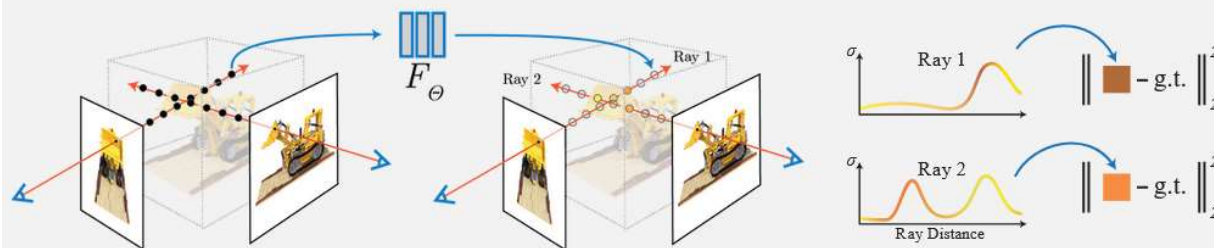
Abstract & Method

We present a method that achieves state-of-the-art results for synthesizing novel views of complex scenes by optimizing an underlying continuous volumetric scene function using a sparse set of input views.

$$(x, y, z, \theta, \phi) \rightarrow \begin{array}{c} \boxed{} \\ \boxed{} \\ \boxed{} \end{array} \rightarrow (RGB\sigma)$$

F_{Θ}

Our algorithm represents a scene using a fully-connected (non-convolutional) deep network, whose input is a single continuous 5D coordinate (spatial location (x, y, z) and viewing direction (θ, ϕ)) and whose output is the volume density and view-dependent emitted radiance at that spatial location.



We synthesize views by querying 5D coordinates along camera rays and use classic volume rendering techniques to project the output colors and densities into an image. Because volume rendering is naturally differentiable, the only input required to optimize our representation is a set of images with known camera poses. We describe how to effectively optimize neural radiance fields to render photorealistic novel views of scenes with complicated geometry and appearance, and demonstrate results that outperform prior work on neural rendering and view synthesis.

<https://www.matthewtancik.com/nerf>

Neural Radiance Fields (NeRF)




TECHNOLOGY

Deep Learning Is Hitting a Wall

What would it take for artificial intelligence to make real progress?

BY GARY MARCUS March 10, 2022

 Share



 Limitations of Deep Learning

Georgia
Tech 

Things to Watch out For

- Research is cyclical
 - SVMs, boosting, probabilistic graphical models & Bayes Nets, Structural Learning, Sparse Coding, Deep Learning
 - Deep learning is unique in its depth and breadth, but...
 - Deep learning may be improved, reinvented, combined, overtaken
- Learn fundamentals for techniques across the field:
 - Know the span of ML techniques and choose the ones that fit your problem!
 - **Be responsible** in 1) how you use it, 2) promises you make and how you convey it
- Try to understand landscape of the field
 - Look out for what is coming up next, not where we are
- Have fun!