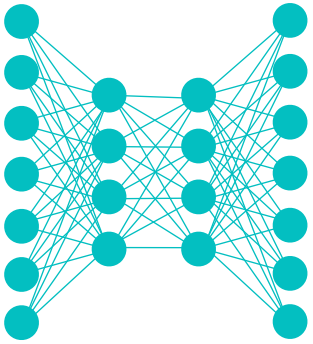
Lecture Notes for

Neural Networks and Machine Learning

Transfer Learning



Logistics and Agenda

- Logistics
  - Style Transfer Lab Due Soon
- Agenda
  - Transfer Learning Overview
  - Transfer Learning in Active Learning
- Next Time:
  - Paper Presentation: Lottery Ticket Hypothesis

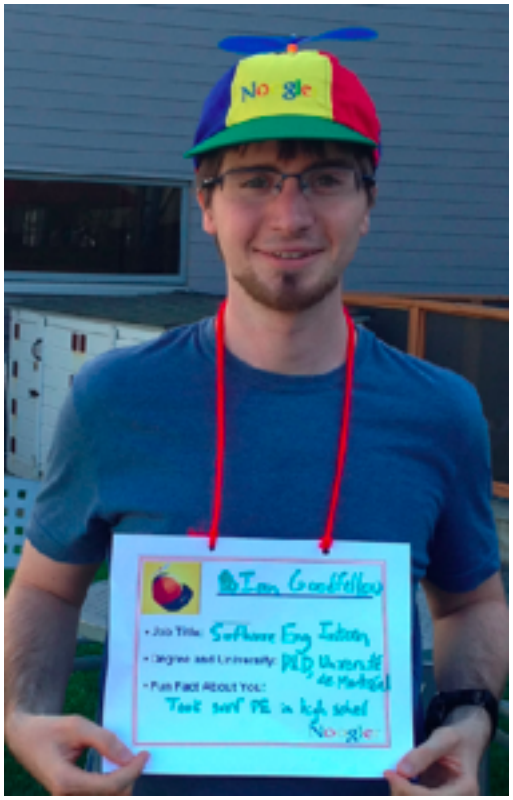
Transfer Learning Overview

Transfer Learning

- Transfer knowledge from a source prediction task to a target prediction task
  - without any regard for performing well on source task
- **Original**: Neural Information Processing 1995 (NeuRiPs)
  - Workshop on Learning to Learn
  - How to effectively retain and reuse previously learned knowledge
  - Originally used in markov chain and Bayesian networks (keeping n-grams, *etc*.)
  - **Key idea**: Humans can generalize what they learn to almost domain, can we mimic this behavior with ML?

Ian Goodfellow's Definition:

"Transfer learning refers to any situation where what has been learned in one setting is exploited to improve generalization in another setting."



Transfer Learning: Large Umbrella

- Appears under a variety of names in the literature:
  - Learning to learn / Life-long learning
  - Knowledge transfer / Inductive transfer
  - Multi-task learning
  - Knowledge consolidation
  - Context-sensitive learning
  - Knowledge-based inductive bias
  - Meta learning
  - Incremental / Cumulative learning

Precise Definition of Transfer Learning

$$X = x_1, x_2, \ldots x_N \in \mathcal{X} \qquad Y = y_1, y_2, \ldots y_N \in \mathcal{Y}$$

$$\mathcal{D} = \{\mathcal{X}, p(X)\} \qquad \mathcal{T} = \{\mathcal{Y}, p(Y|X)\}$$

Domain — **Feature Space** — **Probability Observation**

Task — **Label Space** — **Learned Probability**

- $\mathcal{D}$ Domain defines the features used and probability
- $\mathcal{X}$ is the space of all possible features
- $p(X)$ is probability of observing specific instances in $\mathcal{X}$
  - Typically **intractable** to calculate (generative)
- $\mathcal{T}$ Task is within a domain, defining labels and model
- $\mathcal{Y}$ is space of all possible labels
- $p(Y|X)$ probability of observing specific label given the specific feature:
  - **Not** intractable (discriminative)

Definition with Examples

$$X = x_1, x_2, \ldots x_N \in \mathcal{X}$$

$$Y = y_1, y_2, \ldots y_N \in \mathcal{Y}$$

$$\mathcal{D} = \{\mathcal{X}, p(X)\}$$

Domain — **Feature Space** — **Probability Observation**

$$\mathcal{T} = \{\mathcal{Y}, p(Y|X)\}$$

Task — **Label Space** — **Learned Probability**

- Image Pixels
- Sensor Readings
- Natural Language
- *Almost anything that we can represent as a feature*

- Object Classification
- Dolphin/Shark Classification
- Sentiment Analysis
- *Any labeled task for which we might be able to build a classifier*

Transfer Learning

$$X = x_1, x_2, \ldots x_N \in \mathcal{X} \qquad Y = y_1, y_2, \ldots y_N \in \mathcal{Y}$$

$$\mathcal{D} = \{\mathcal{X}, p(X)\} \qquad \mathcal{T} = \{\mathcal{Y}, p(Y|X)\}$$

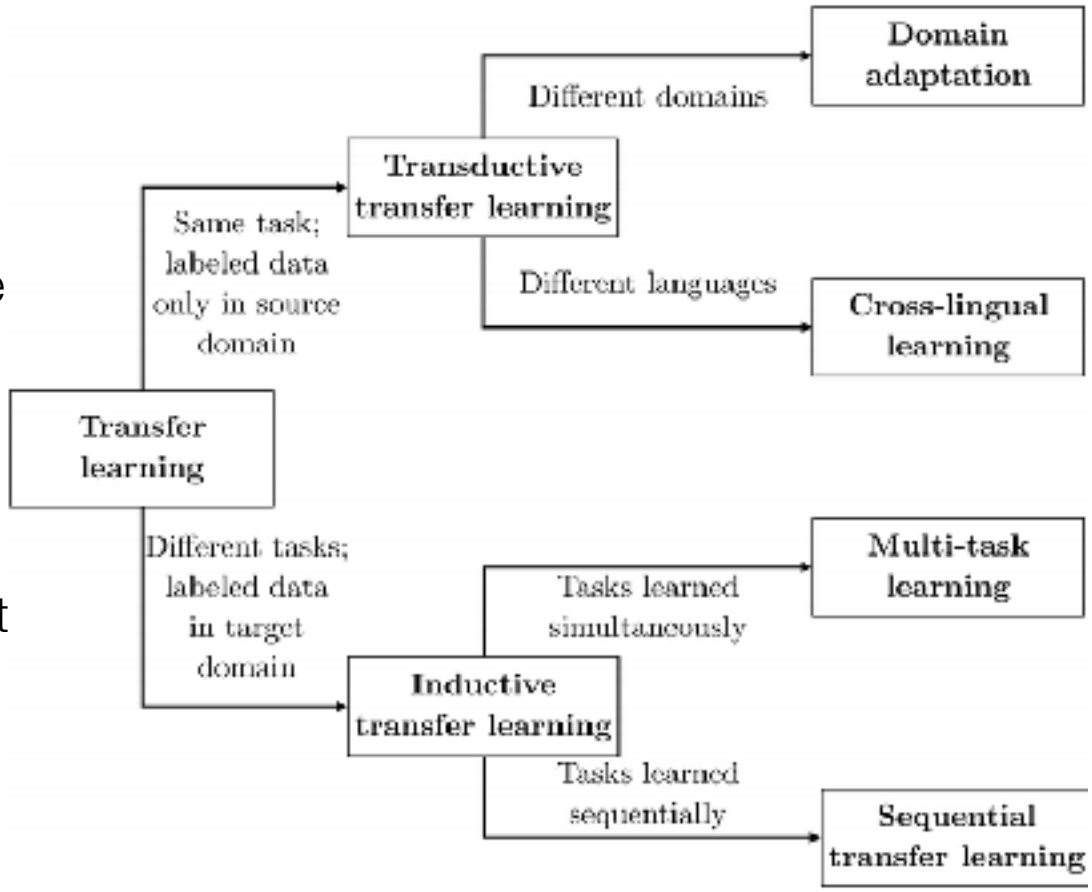
Domain — **Feature Space** — **Probability Observation**

Task — **Label Space** — **Learned Probability**

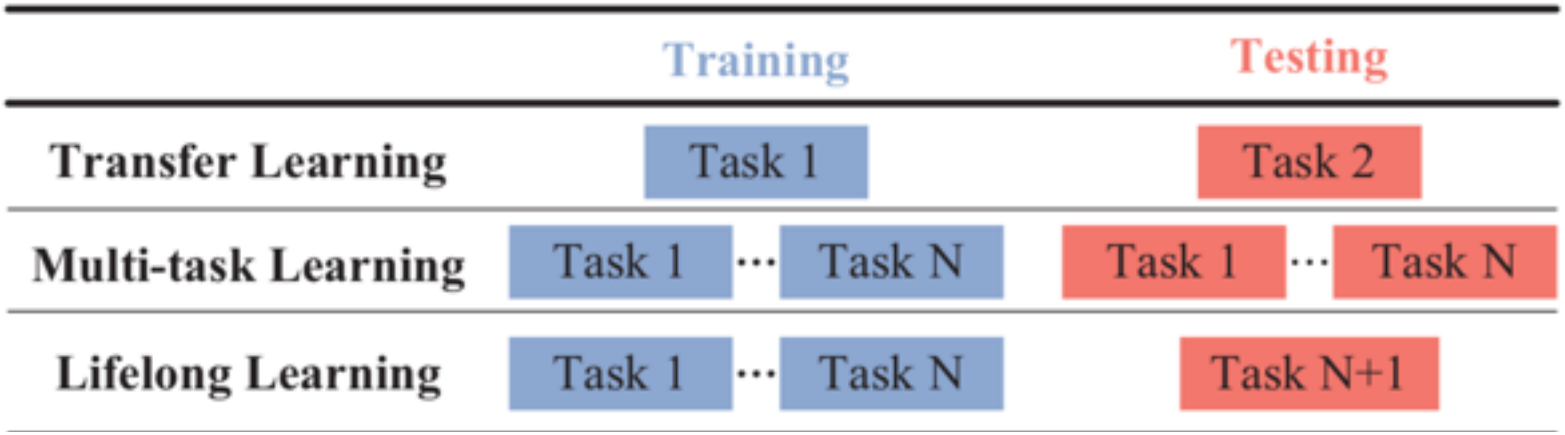
- Need to translate document Source to Target $\mathcal{T}_S \rightarrow \mathcal{T}_T$
- Variety of differences might be present. For example, in the context of document classification:
  - **Feature space**: different languages $\mathcal{X}_S \neq \mathcal{X}_T$
  - **Marginals**: same language, same label space, but differing topics $p(X_S) \neq p(X_T)$
  - **Conditional**: different label distributions or possibly different labels $p(Y_s|X_S) \neq p(Y_T|X_T)$

Categories of Transfer Learning

- **Inductive**: Same Domain, Different Task
  - Using pre-trained VGG as basis for classifying dolphins versus sharks, Style Transfer, sentiment analysis from Glove
- **Transductive**: Different (but related) Domains, Same Task
  - Place identification from RGB Images or LIDAR
- **Unsupervised Transfer**: Different Domains, Different Tasks
  - Learning to paint art and learning to be a surgeon
  - Not yet a field with much repeatable traction



Aside: Other categorizations



Humans can learn to ride a bike and use that to understand better about driving a car. Machine Learning in its current form is far from this capability. How can we move our siloed version of artificial intelligence closer to the process of human based learning? How can we accumulate knowledge from model to model?

Does biology of human learning hold any clues to success? How does a human learn to crawl? To talk? To ride a bike? What is a human's motivation to learn?

Transfer Learning with Neural Networks

Found in a recent paper:

6 Unrelated Work

This paper is not related to [8, 23, 48, 13, 35] in any way, but we think everyone should read these papers because: (1) they're real good, (2) my friends also need those citations.

7 Related Work

Deep Transfer Learning

- Almost always **Inductive Transfer**
  - (new task , same domain)
- Almost always **Feature Representation Transfer**
  - like image pre-training
- All other topics are open research topics that maybe one of you will solve!

Approaches with Deep Learning

- **Feature Extraction Transfer**
  - Most well known: use learned parameters from one task in another task in same domain
  - Most useful when labels for target domain are sparse

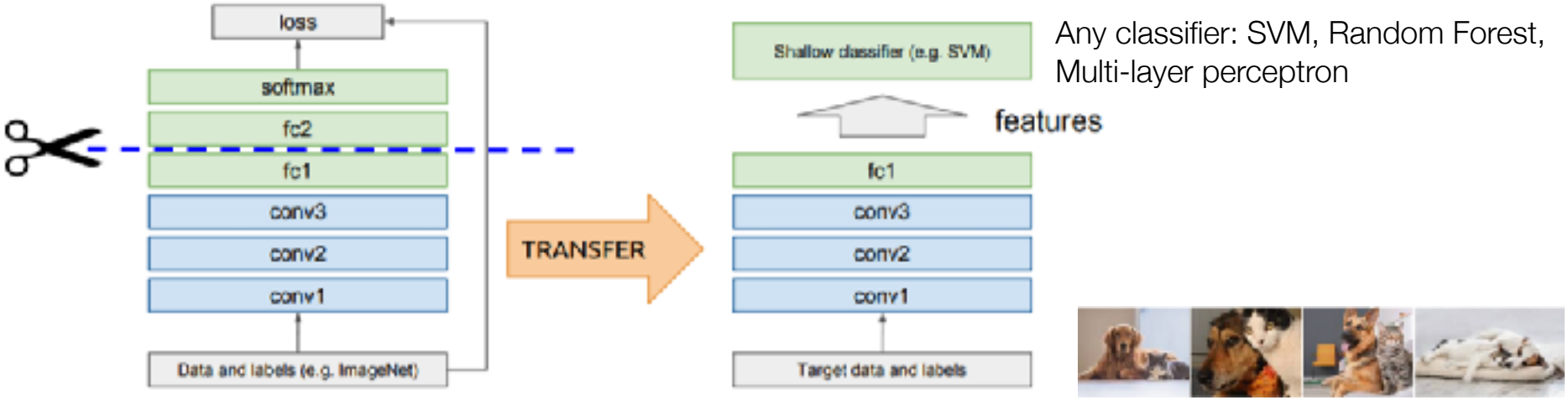


Image Net

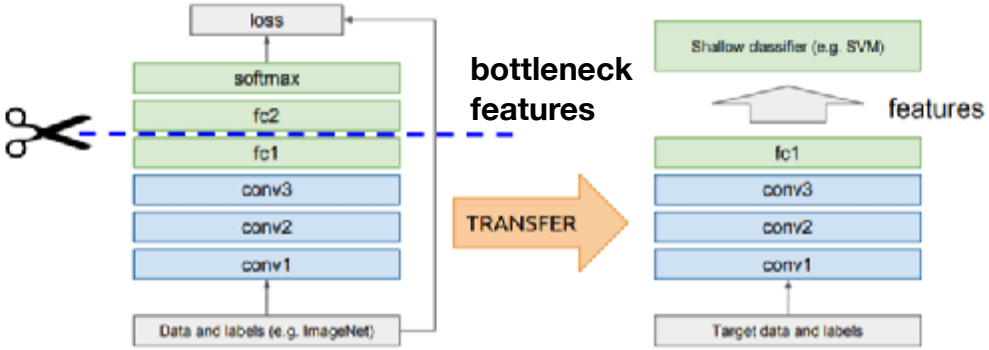
New domain: Dogs versus Cats

New domain: Gaze Classification

https://towardsdatascience.com/a-comprehensive-hands-on-guide-to-transfer-learning-with-real-world-applications-in-deep-learning-212bf3b2f27a

Defining the Bottleneck

- Frozen training layers before bottleneck:
  - Why waste computations?
  - Computing more than one forward pass on the same data—just save them out
  - *Unless* using augmentation
- In Keras, build multiple models with different entry points
  - **Input** to **Bottleneck**
  - **Bottleneck** to **Output**
  - **Input** to **Output**




```
inputs = Input(shape=(IMG_SIZE, IMG_SIZE, 3))
model_base = VGG(include_top=False,
                 input_tensor=inputs, weights="imagenet")
```

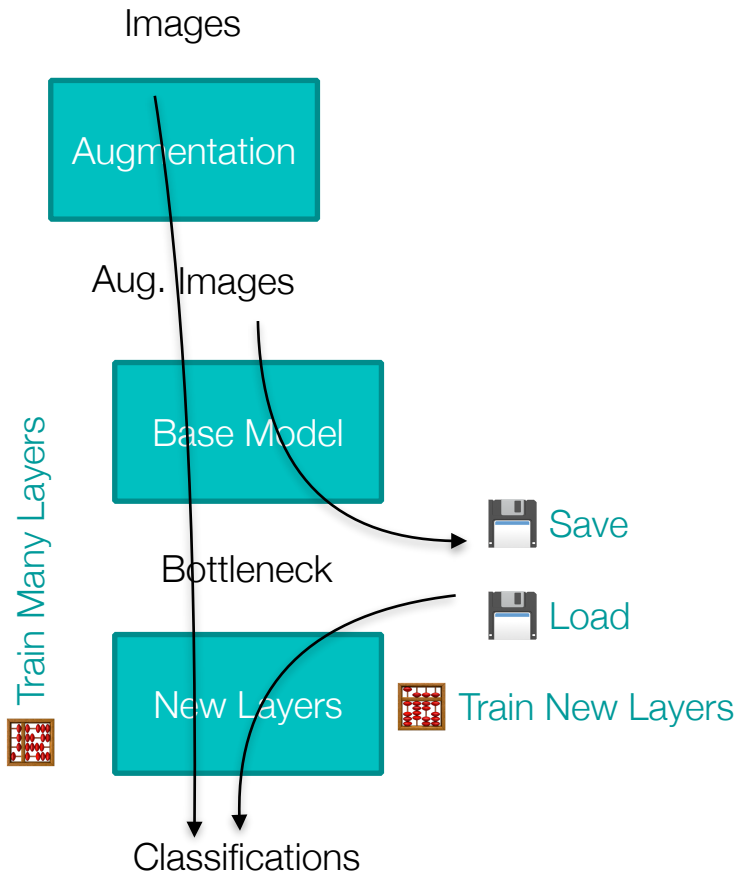
```
bottleneck = Input(shape=model_base.output.shape)
outputs = Dense(NUM_CLASSES)(bottleneck)
model_top = Model(bottleneck, outputs)
```

```
model_total = Model(inputs, outputs)
```

https://keras.io/examples/vision/image_classification_efficientnet_fine_tuning/

Freezing and Fine-tuning

- Step 1, Freeze base model:
  - No update during back-propagation
  - Only update layers after the bottleneck
  - Optional: Augment a set of training data
  - Send training dataset through base model
    - Save out bottleneck features
  - Train bottleneck features in new task
    - Typically 5-10 epochs is sufficient, easy to overfit
    - Larger training step size is okay
- Step 2, Fine-tune, unfreeze a few layers in base model:
  - Setup images to use some type of augmentation
  - Attach newly trained model to pre-trained model
  - Train to your hearts content, use smaller training step size



Bottlenecking on Maneframe

Dolphins versus Sharks

Justin Ledford

Follow Along:https://github.com/8000net/Transfer-Learning-Dolphins-and-Sharks
Or in the Master Repo:
04 Transfer Learning.ipynb

Another Great Example:
https://keras.io/examples/vision/image_classification_efficientnet_fine_tuning/

Popular Transfer Learning Models

- **Vision**:
  - ImageNet Architectures:
    - VGG, Inception, ResNet, Xception, EfficientNet
- **Audio**:
  - WaveNet, almost always WaveNet
- **Text**:
  - Word Embedding
    - Glove, Word2Vec, ConceptNet
  - Sentence Embedding
    - Universal Sentence Encoders (Google)
    - BERT (Google)
    - Skip-thought Vectors

- **From the Research my Students have done:**
  - VGG for transferring to gaze classification
  - VGG for swapped face detection
  - Emotion recognition for prosody classification
  - YOLO/DarkNet for surgical instrument detection
  - GLOVE for similar instructions in a maintenance manual

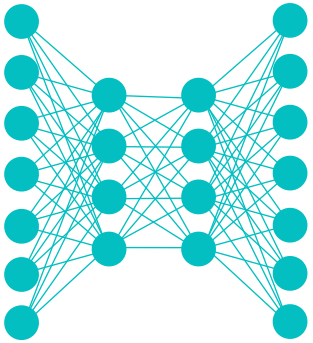
Lecture Notes for

Neural Networks and Machine Learning

Transfer Learning

Next Time:
Multi-Modal and Multi-Task

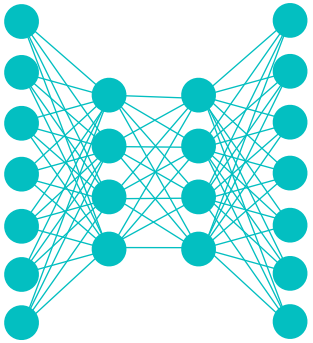
Reading: Keras F-API



Lecture Notes for

Neural Networks and Machine Learning

Adaptive, Self-supervised, Multi-modal, & Multi-task Learning



Logistics and Agenda

- Logistics
  - Lab three uses multi-task and multi-modal learning
- Agenda
  - Adaptive Learning
  - Self-Supervised Learning
  - Paper Presentation
  - Multi-modal/task Learning
    - Techniques
    - Applications and domains
- Next Time:
  - Paper Presentation: Speaker Verification with X-Vectors and SincNet

Paper Presentation: The Lottery Hypothesis

Published as a conference paper at ICLR 2019

THE LOTTERY TICKET HYPOTHESIS: FINDING SPARSE, TRAINABLE NEURAL NETWORKS

Jonathan Frankle
MIT CSAIL
jfrankle@csail.mit.edu

Michael Carbin
MIT CSAIL
mcarbin@csail.mit.edu

ABSTRACT

Neural network pruning techniques can reduce the parameter counts of trained networks by over 90%, decreasing storage requirements and improving computational performance of inference without compromising accuracy. However, contemporary experience is that the sparse architectures produced by pruning are difficult to train from the start, which would similarly improve training performance.

We find that a standard pruning technique naturally uncovers subnetworks whose initializations made them capable of training effectively. Based on these results, we articulate the *lottery ticket hypothesis*: dense, randomly-initialized, feed-forward networks contain subnetworks (*winning tickets*) that—when trained in isolation—reach test accuracy comparable to the original network in a similar number of iterations. The winning tickets we find have won the initialization lottery: their connections have initial weights that make training particularly effective.

We present an algorithm to identify winning tickets and a series of experiments that support the lottery ticket hypothesis and the importance of these fortuitous initializations. We consistently find winning tickets that are less than 10-20% of the size of several fully-connected and convolutional feed-forward architectures for MNIST and CIFAR10. Above this size, the winning tickets that we find learn faster than the original network and reach higher test accuracy.

Last Time

$X = x_1, x_2, \ldots x_N \in \mathcal{X}$ $\quad Y = y_1, y_2, \ldots y_N \in \mathcal{Y}$

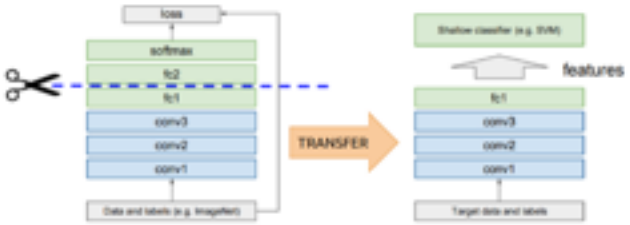
$\mathcal{D} = \{\mathcal{X}, p(X)\}$ $\quad \mathcal{T} = \{\mathcal{Y}, p(Y|X)\}$

Domain: Feature Space, Probability Observation

Task: Label Space, Learned Probability

- Domain defines the features used
- Marginal Distribution of observing instances in the feature space
  - Typically intractable to calculate (generative)
- Task is within a domain
- Label space is typically one specific classification or regression task
- Probability of observing label given the feature space:
  - Not intractable (discriminative)

- Feature Extraction Transfer
  - Most well known: use learned parameters from one task in another task in same domain
  - Most useful when labels for target domain are sparse



	Training	Testing
Transfer Learning	Task 1	Task 2
Multi-task Learning	Task 1 ··· Task N	Task 1 ··· Task N
Lifelong Learning	Task 1 ··· Task N	Task N+1

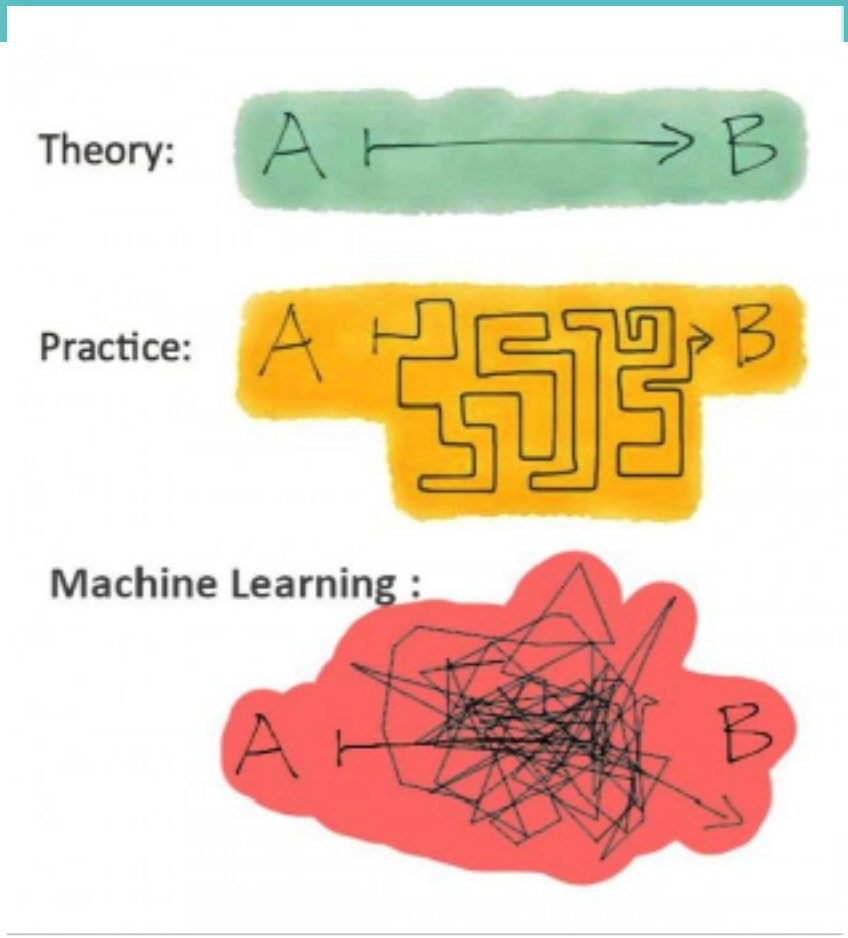
Humans can learn to ride a bike and use that to understand better about driving a car. Machine Learning in its current form is far from this capability. How can we move our siloed version of artificial intelligence closer to the process of human based learning? How can we accumulate knowledge from model to model?

Does biology of human learning hold any clues to success? How does a human learn to crawl? To talk? To ride a bike? What is a human's motivation to learn?

Ian Goodfellow's Definition:

"Transfer learning refers to any situation where what has been learned in one setting is exploited to improve generalization in another setting."

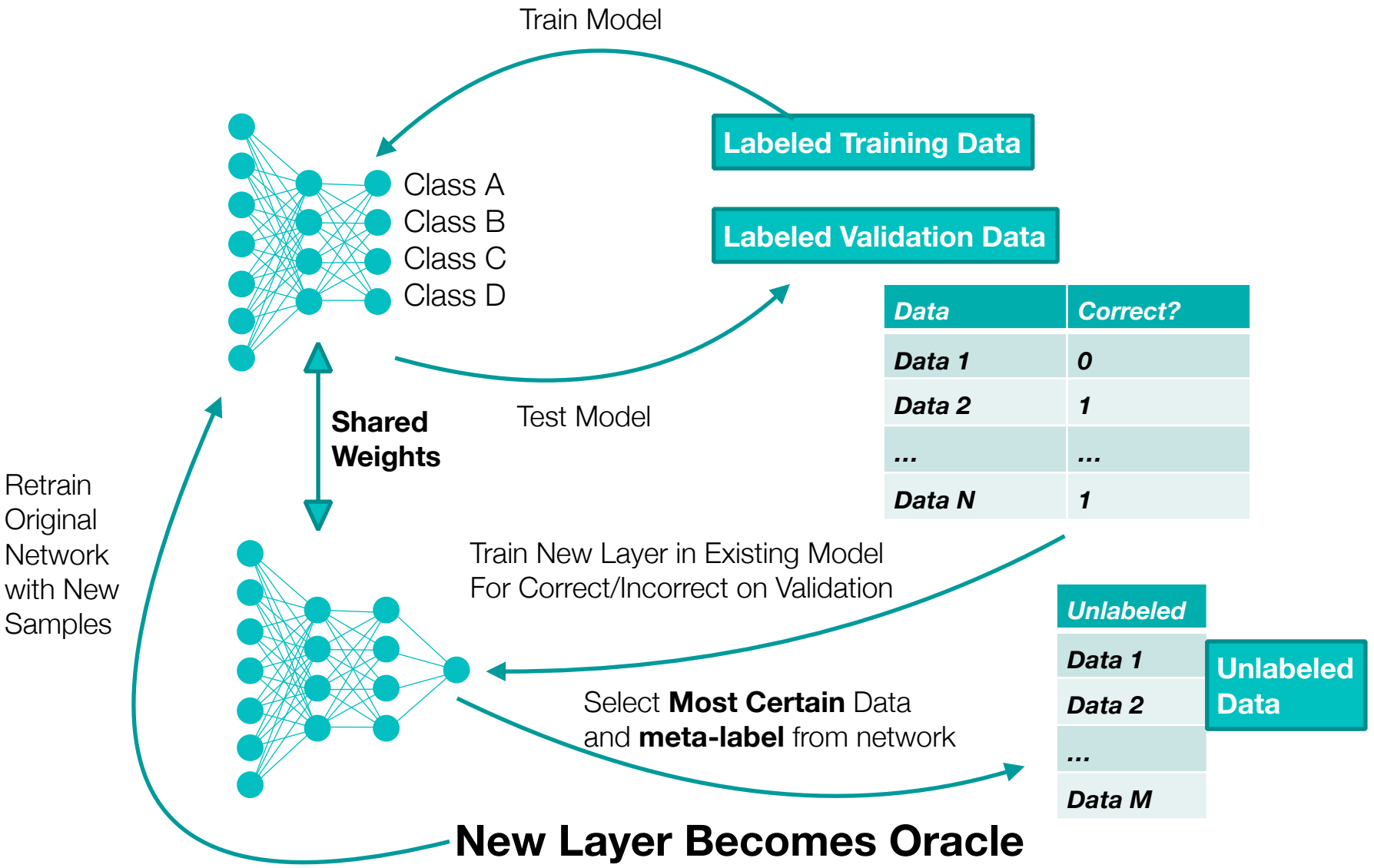
Active Transfer Learning



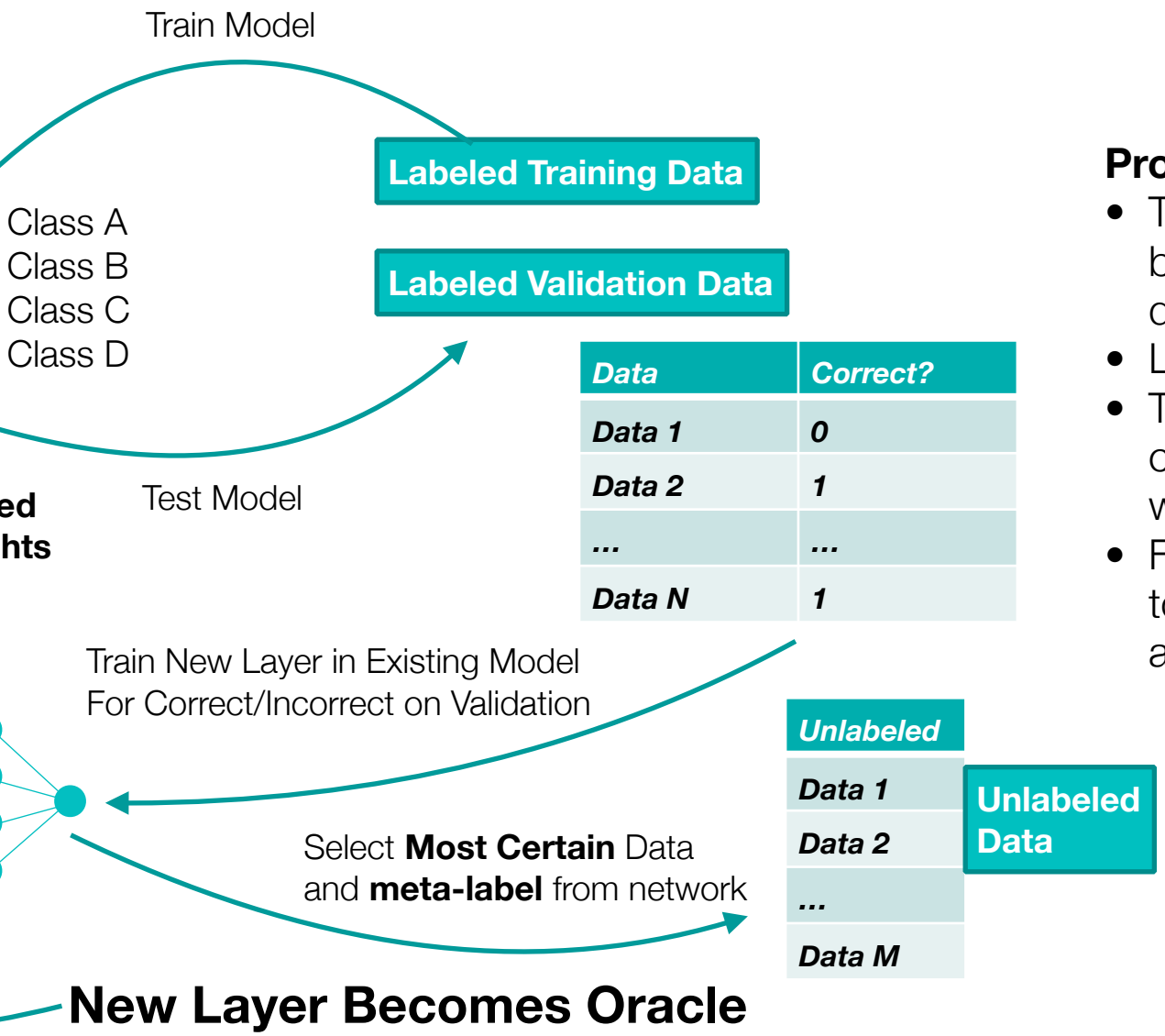
Active Learning Overview

- **Basic Idea**: Use a trained model to sample from an oracle that can magically give you a new label
  - We are asking:
    What labels should we ask the oracle about?
- Uncertainty Sampling
  - Choose instances where the model is most uncertain or most certain
  - Various ways to measure certainty
- Diversity Sampling
  - Choose instances that are similar or different from training distribution

Uncertainty Sampling with a Neural Network



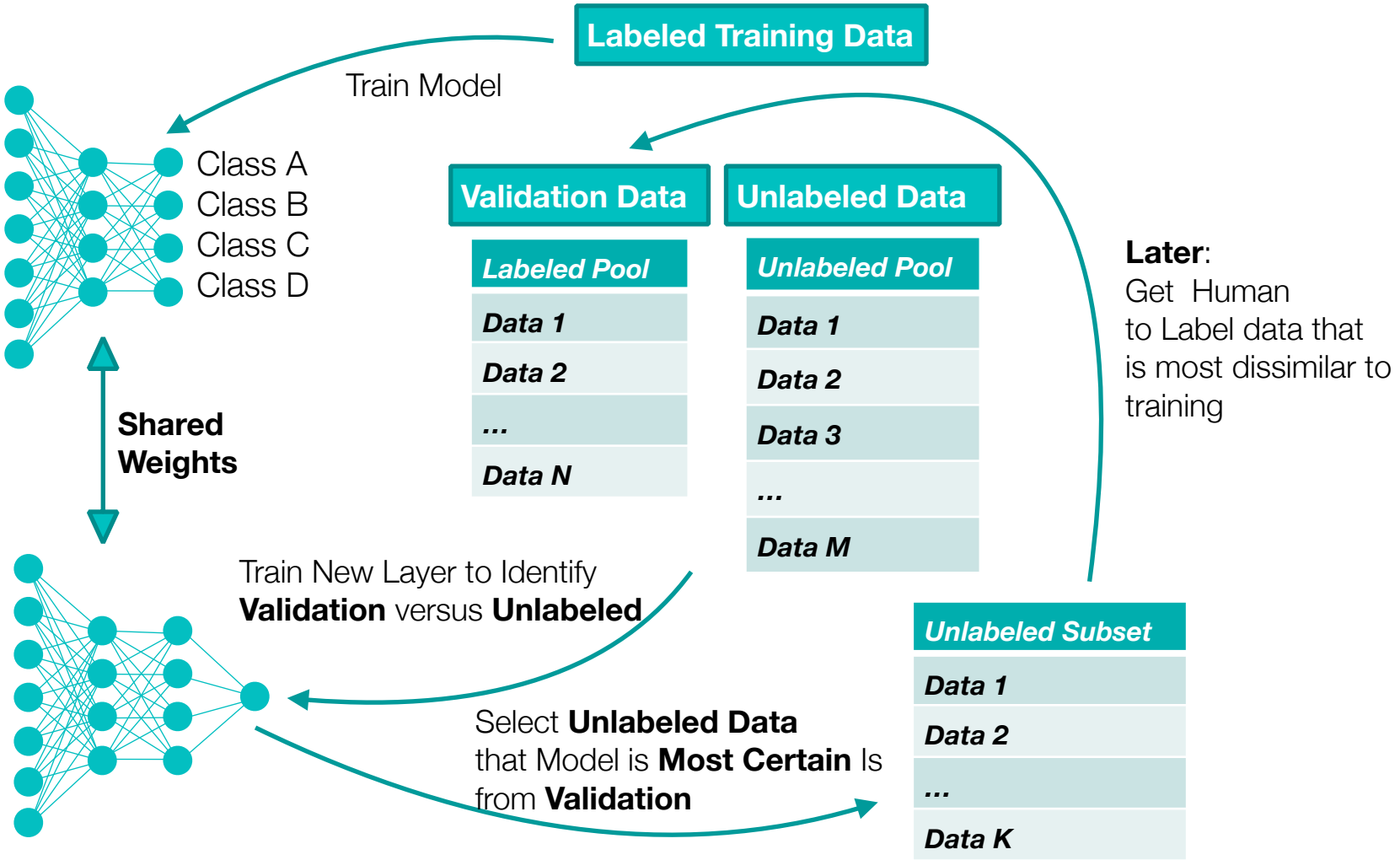
Uncertainty Sampling with a Neural Network



Problems:

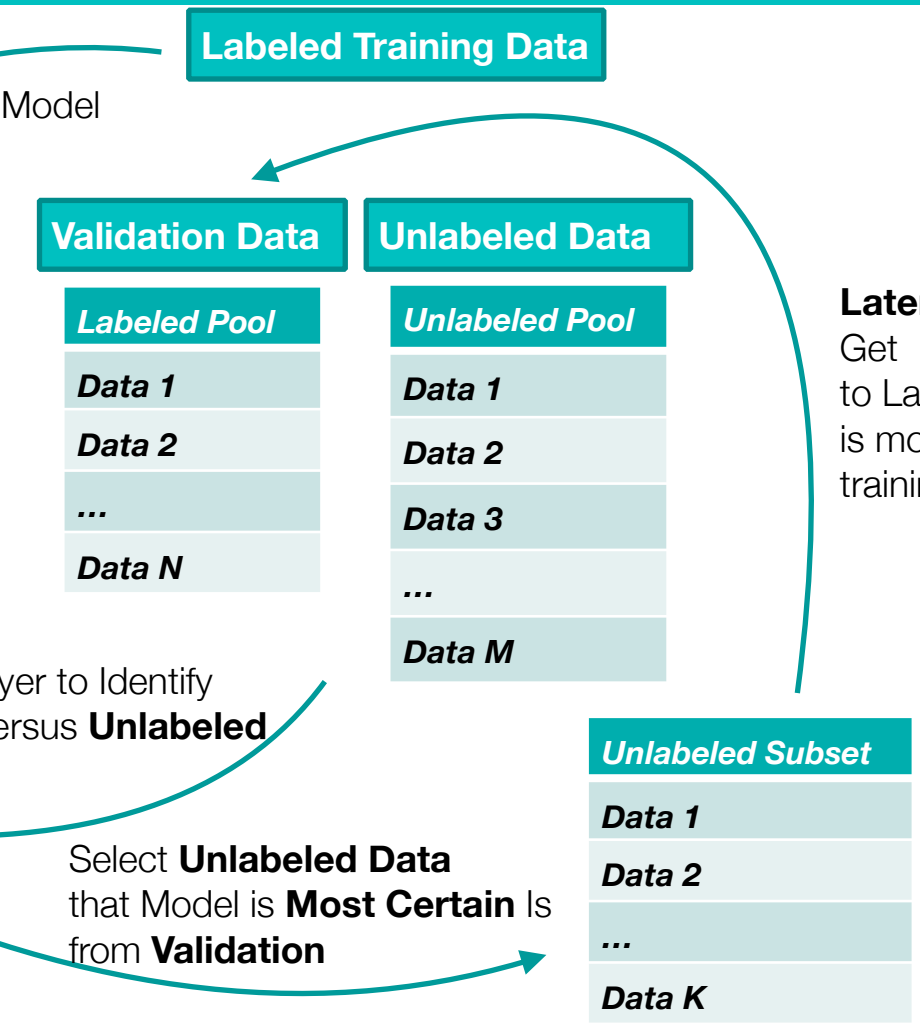
- Training pool is represented by classes the model already does well predicting
- Limited diversity of Samples
- Training pool can become contaminated easily from a few wrong predictions
- For Oracle: we might be asking to get labels that the model is already good at classifying

Diversity Sampling with a Neural Network



New Layer Decides if Samples are Added to Validation Data

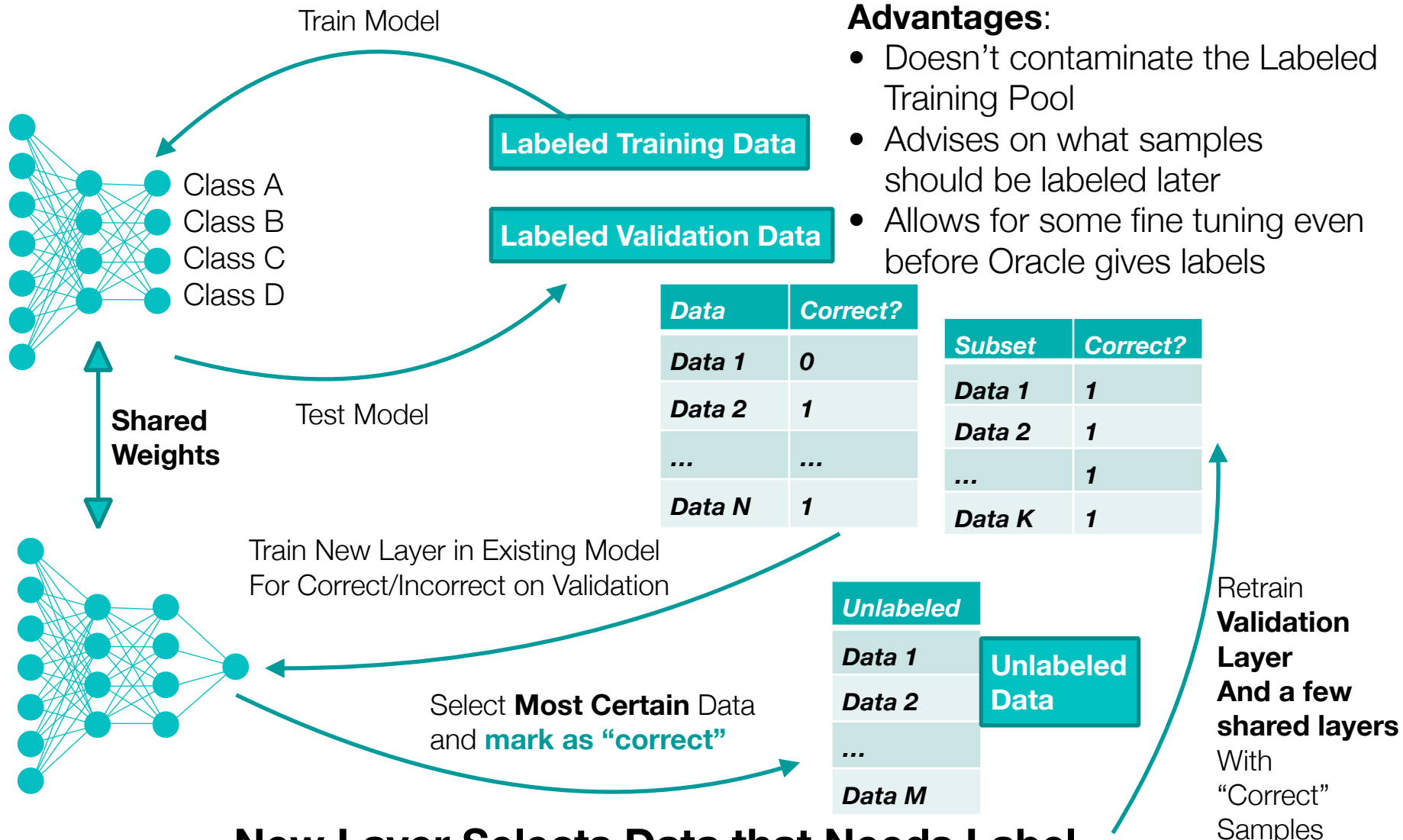
Diversity Sampling with a Neural Network



Discussion:

- Training pool is not contaminated
- Expands validation data in well mannered way, not adding too "far away" samples
- Validation versus Unlabeled might not be the best comparison, because it ignores confusions in the training data
- For Oracle: we can get labels to inputs that the model is likely to be unsure about
- But… this only helps us when we have an Oracle to give us labels

ATLAS: Active Transfer Learning for Adaptive Sampling



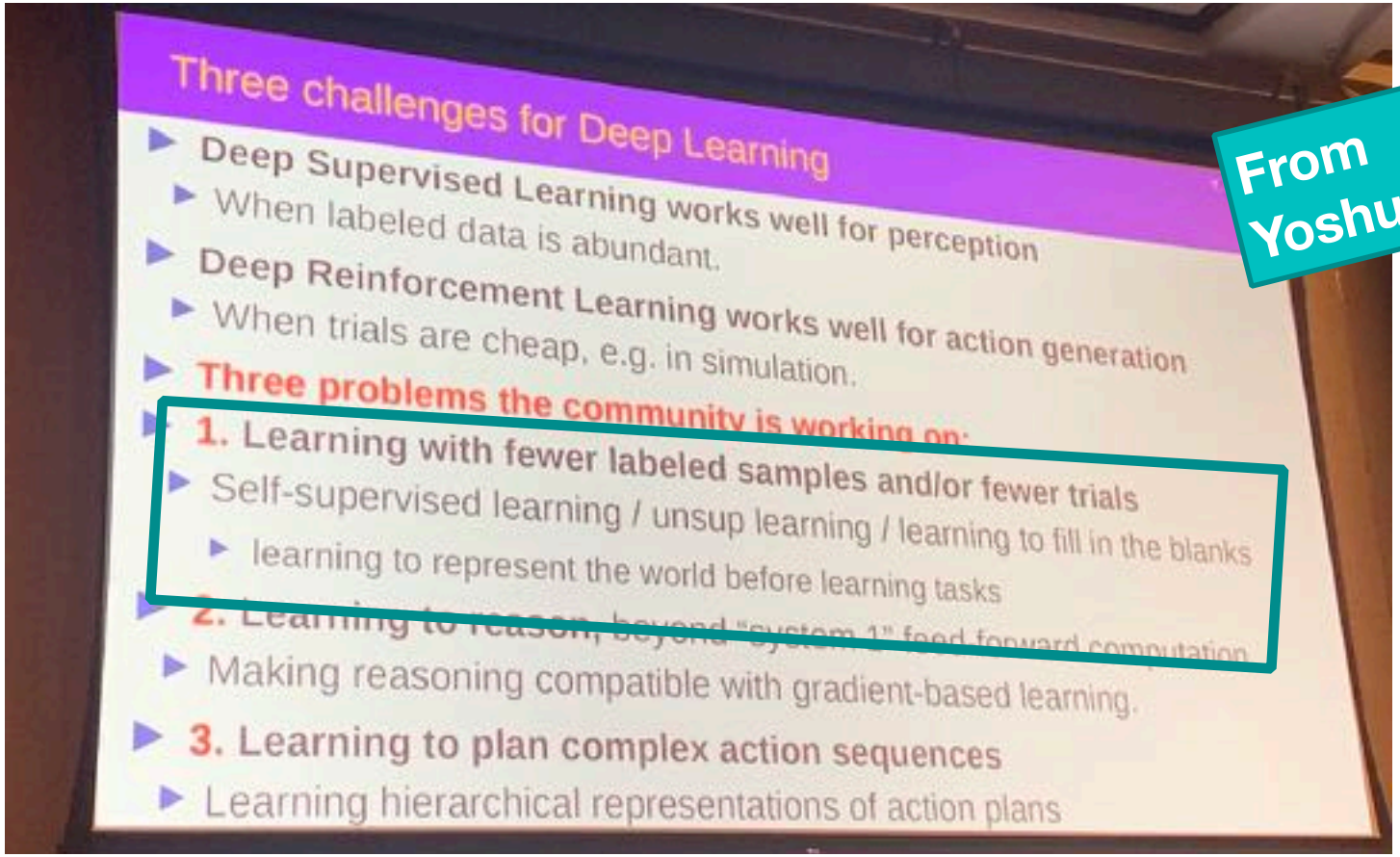
Advantages:

- Doesn't contaminate the Labeled Training Pool
- Advises on what samples should be labeled later
- Allows for some fine tuning even before Oracle gives labels

Time Period	Protocol	Expected Feedback
First Week	Homeowner provides 8-20 examples over the first week: • 1-2 Shower usages • 1 run of the dishwasher • 1 run of the laundry machine • 2 examples of each toilet • 1 example of hot and cold water use for each dual handle faucet • 1 example of hot, cold, and mixed water use for each single handle faucet (2 examples if in kitchen)	HydroSense relies on the rule based classifier for the first week. Pressure waves are saved in order to create a sparse codebook of features. Results are displayed at the fixture category for dishwashers, showers, and washing machines.
Start of Second Week	Homeowner provides 2-4 labels every other day when the system messages them on their mobile device	Results are displayed at the full fixture category level from the CoDBN-VE algorithm. Expected accuracy: • 85% at fixture category level
End of Second Week	Homeowner has supplied 9-12 examples that were flagged by active learning.	HydroSense now displays results at the Lumped Fixture level. Expected accuracy: • 82% at fixture level • 87% at fixture category level
End of Third Week	Homeowner continues to supply sparsely selected examples every other day. About 9-12 additional examples provided.	Valve level accuracy now provided. Expected accuracy: • 80% at valve level • 87% at fixture level • 92% at fixture category level
Fourth Week	Homeowner can optionally continue to provide examples to the system for increased accuracy.	Expected accuracy: • 81% at valve level • 89% at fixture level • 93% at fixture category level

Table 8-2. Expected feedback and calibration protocol for semi-supervised HydroSense system

Self-Supervised Learning

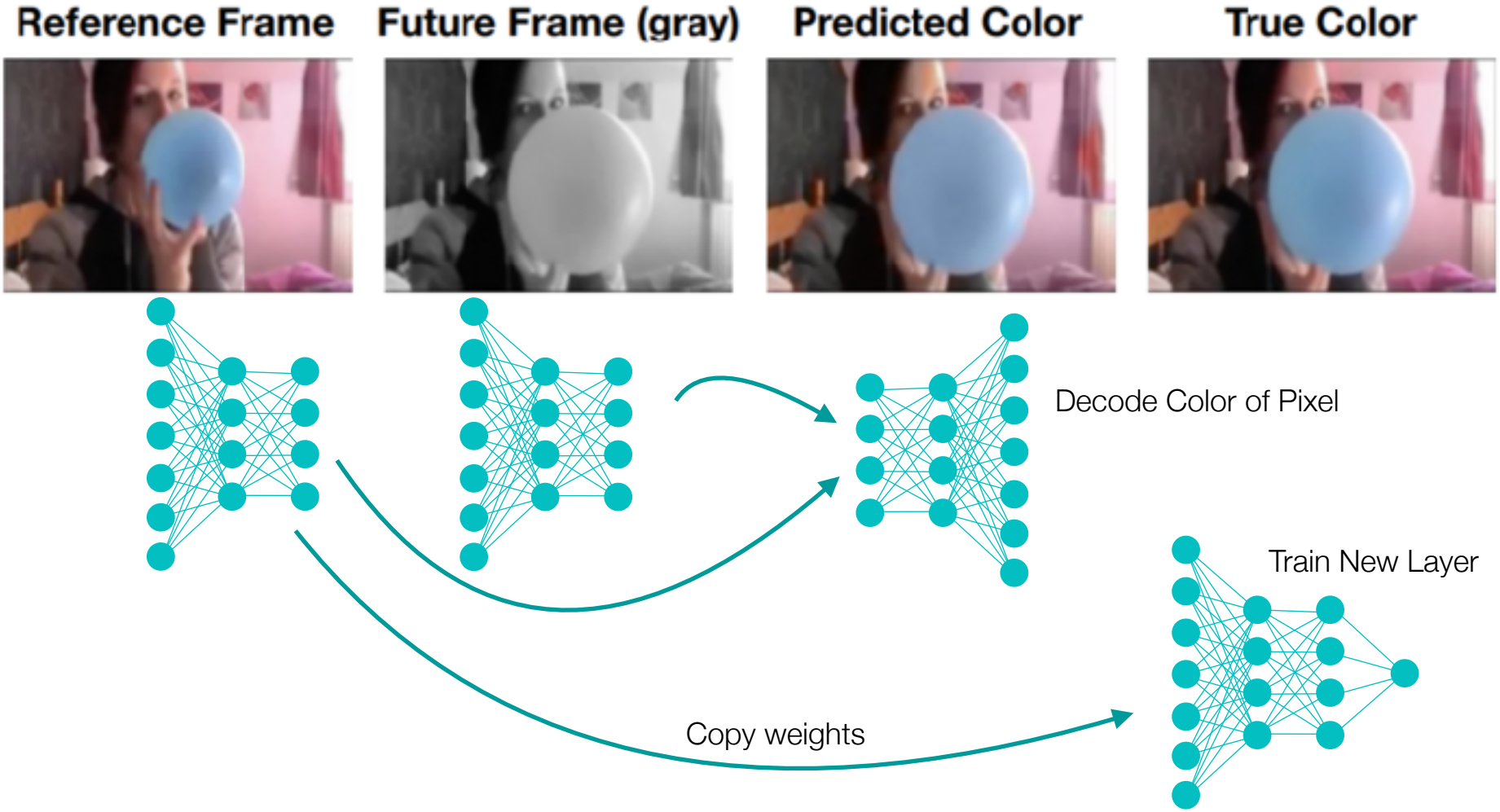


From Yoshua Bengio

Self-supervised Learning

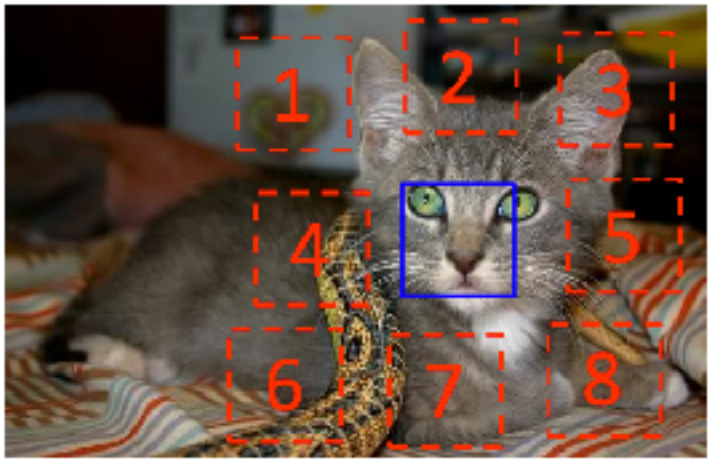
- **Problem**: deep learning is not sample efficient
- **Idea**: learn about the world before learning the task
- **New Problem**: how do we learn about the world?
- **Solution**: transfer learning on toy problem
  - 1. train on auxiliary task that is easy to label
  - 2. throw away anything specific to auxiliary task
  - 3. train new network with task of interest, transferring knowledge (downstream task)
  - 4. profit

Examples of Self Supervised Learning

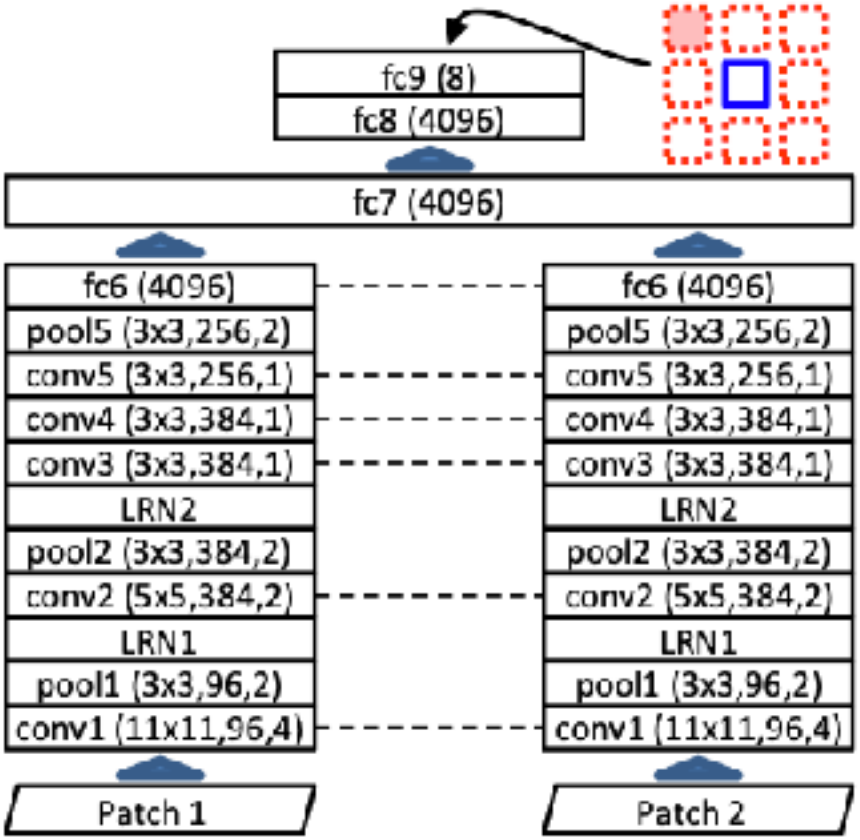


https://www.fast.ai/2020/01/13/self_supervised/

Examples of Self Supervised Learning



X = (,); Y = 3



Unsupervised Visual Representation Learning by Context Prediction

Carl Doersch[1,2] Abhinav Gupta[1] Alexei A. Efros[2]
[1] School of Computer Science Carnegie Mellon University
[2] Dept. of Electrical Engineering and Computer Science University of California, Berkeley

https://www.fast.ai/2020/01/13/self_supervised/

Examples of SSL

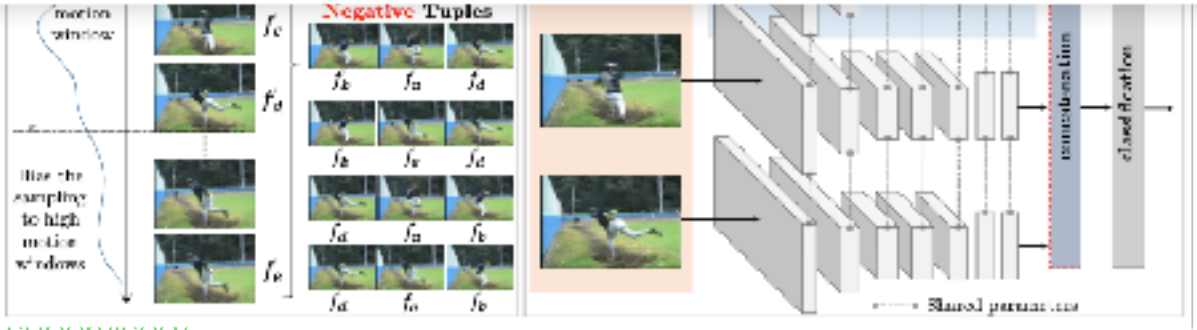
Shuffle and Learn: Unsupervised Learning using Temporal Order Verification

Ishan Misra[1] C. Lawrence Zitnick[2] Martial Hebert[1]

[1] The Robotics Institute, Carnegie Mellon University
[2] Facebook AI Research

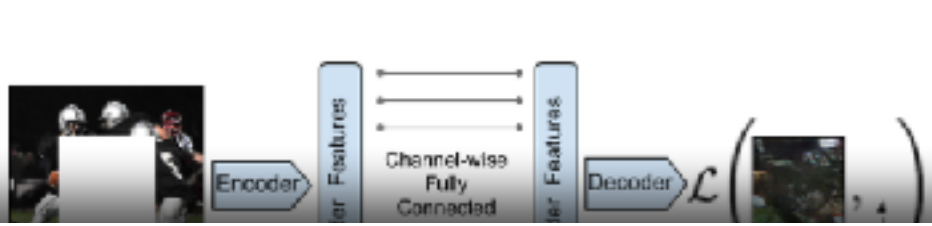
Table 2: Mean classification accuracies over the 3 splits of UCF101 and HMDB51 datasets. We compare different initializations and finetune them for action recognition.

Dataset	Initialization	Mean Accuracy
UCF101	Random	38.6
	(Ours) Tuple verification	**50.2**
HMDB51	Random	13.3
	UCF Supervised	15.2
	(Ours) Tuple verification	**18.1**

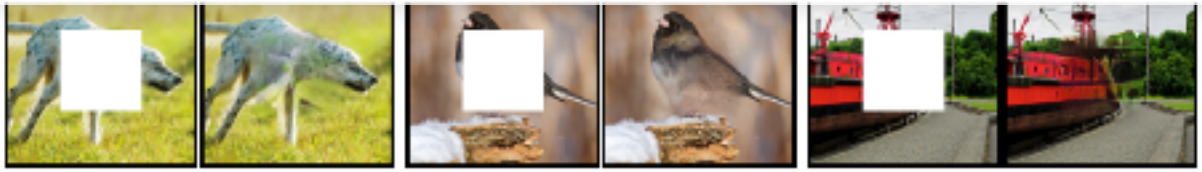


https://www.fast.ai/2020/01/13/self_supervised/

Examples of Self Supervised Learning



Pretraining Method	Supervision	Pretraining time	Classification	Detection	Segmentation
ImageNet [26]	1000 class labels	3 days	78.2%	56.8%	48.0%
Random Gaussian	initialization	< 1 minute	53.3%	43.4%	19.8%
Autoencoder	-	14 hours	53.8%	41.9%	25.2%
Agrawal *et al.* [1]	egomotion	10 hours	52.9%	41.8%	-
Wang *et al.* [39]	motion	1 week	58.7%	47.4%	-
Doersch *et al.* [7]	relative context	4 weeks	55.3%	46.6%	-
Ours	context	14 hours	56.5%	44.5%	30.0%



Context Encoders: Feature Learning by Inpainting

Deepak Pathak Philipp Krähenbühl Jeff Donahue Trevor Darrell Alexei A. Efros
University of California, Berkeley

https://www.fast.ai/2020/01/13/self_supervised/

Unsupervised Consistency Loss

$$\min_{\mathbf{w}} \overbrace{\mathbf{E}_{\mathbf{x},y\in L}[-\log p_{\mathbf{w}}(y\,|\,\mathbf{x})]}^{\text{cross entropy}} + \lambda \overbrace{\mathbf{E}_{\mathbf{x}\in U}\mathbf{E}_{\hat{\mathbf{x}}\leftarrow q(\hat{\mathbf{x}}|\mathbf{x})}\left[\mathscr{D}_{KL}\left(p_{\mathbf{w}}(y\,|\,\mathbf{x})\,||\,p_{\mathbf{w}}(y\,|\,\hat{\mathbf{x}})\right)\right]}^{\text{consistency in augmentation}}$$

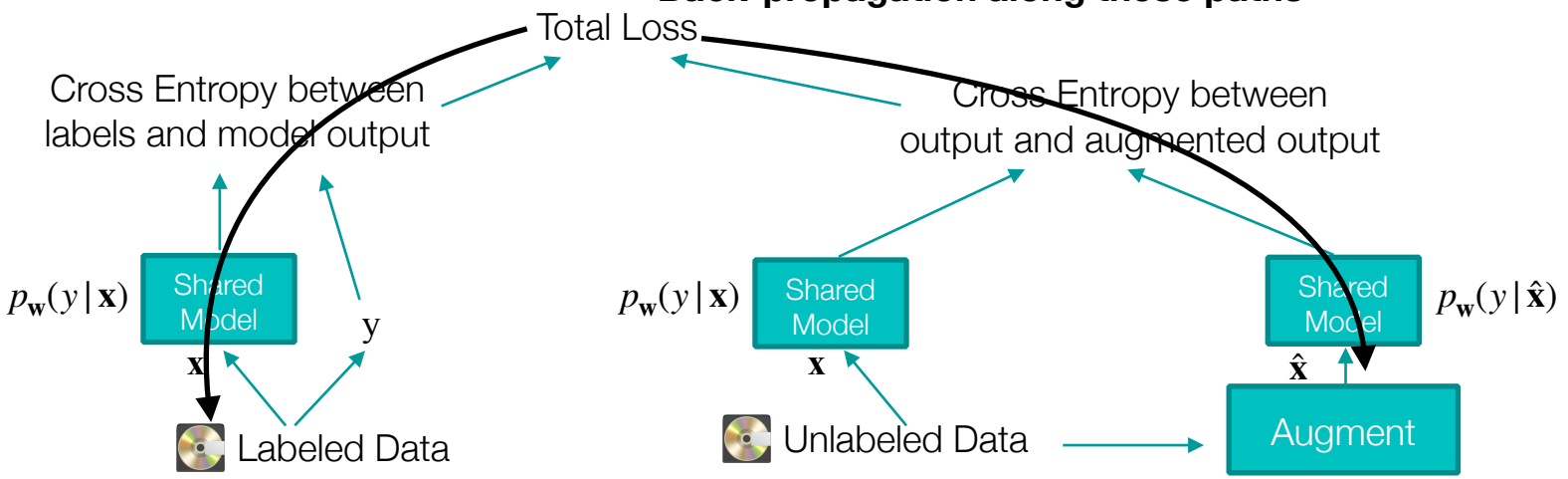
no back prop **yes** back prop

Neural Network approximates $p(y|\mathbf{x})$ by $\mathbf{w}$
Use labeled data to minimize network

Sample new $\mathbf{x}$ from unlabeled pool with function q
function q is augmentation procedure
Minimize cross entropy of two models

Get accustomed to this notation

Update Model with Back-propagation along these paths



Unsupervised Data Augmentation (UDA) for Consistency Training, Xie et al., Neurlps 2019

Unsupervised Consistency Loss

$$\min_{\mathbf{w}} \overbrace{\mathbf{E}_{\mathbf{x},y\in L}[-\log p_{\mathbf{w}}(y\,|\,\mathbf{x})]}^{\text{cross entropy}} + \lambda \overbrace{\mathbf{E}_{\mathbf{x}\in U}\mathbf{E}_{\hat{\mathbf{x}}\leftarrow q(\hat{\mathbf{x}}|\mathbf{x})}\left[\mathscr{D}_{KL}\left(p_{\mathbf{w}}(y\,|\,\mathbf{x})\,||\,p_{\mathbf{w}}(y\,|\,\hat{\mathbf{x}})\right)\right]}^{\text{consistency in augmentation}}$$

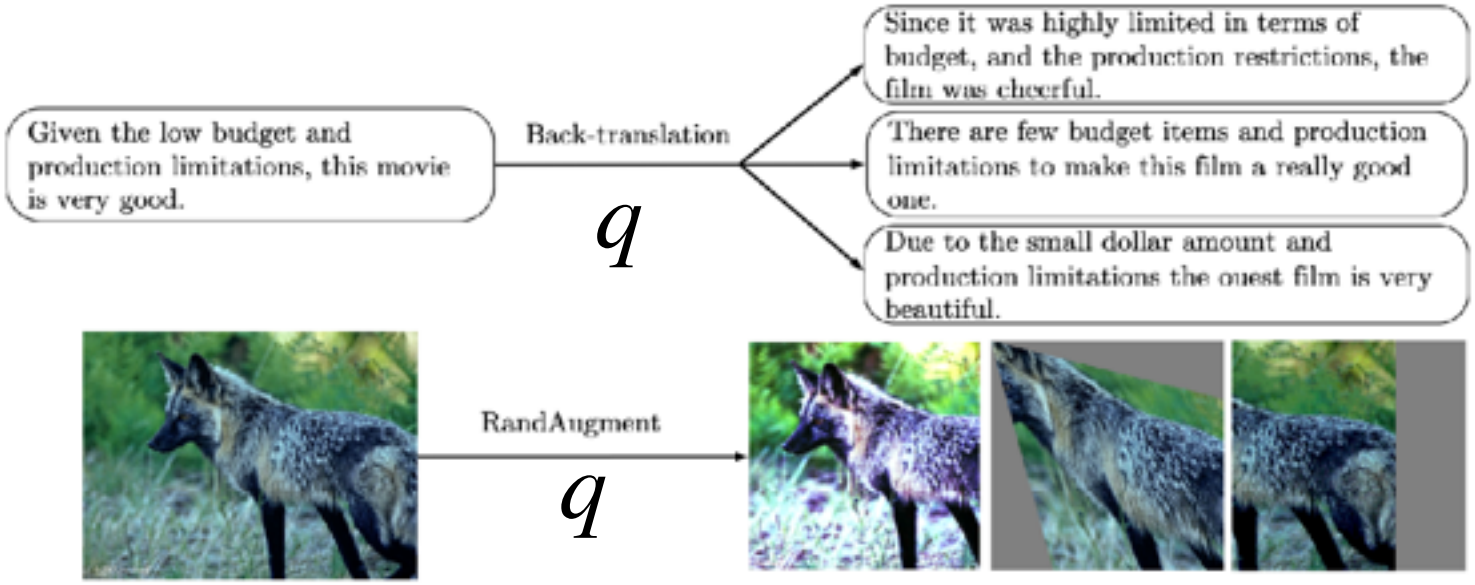


Figure 2: Augmented examples using back-translation and RandAugment.

Unsupervised Data Augmentation (UDA) for Consistency Training, Xie et al., NeurIps 2019

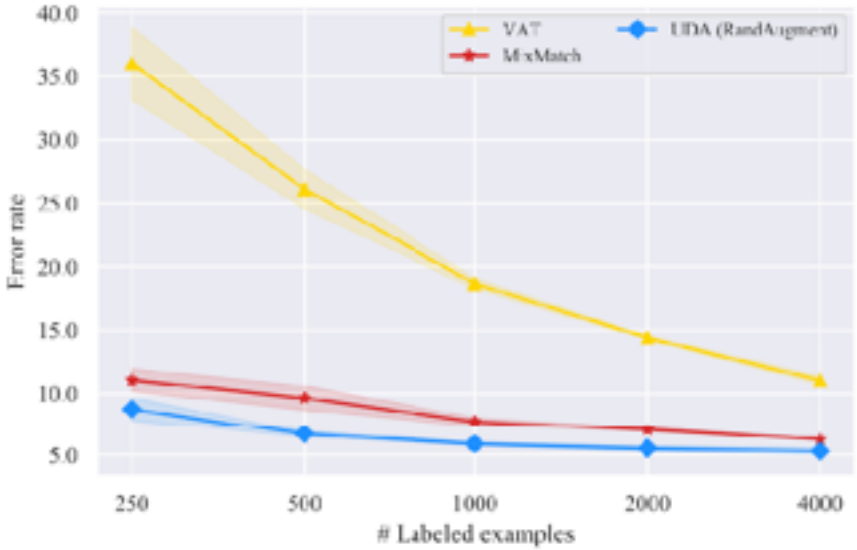
Unsupervised Consistency Loss

Augmentation (# Sup examples)	Sup (50k)	Semi-Sup (4k)
Crop & flip	5.36	16.17
Cutout	4.42	6.42
RandAugment	**4.23**	**5.29**

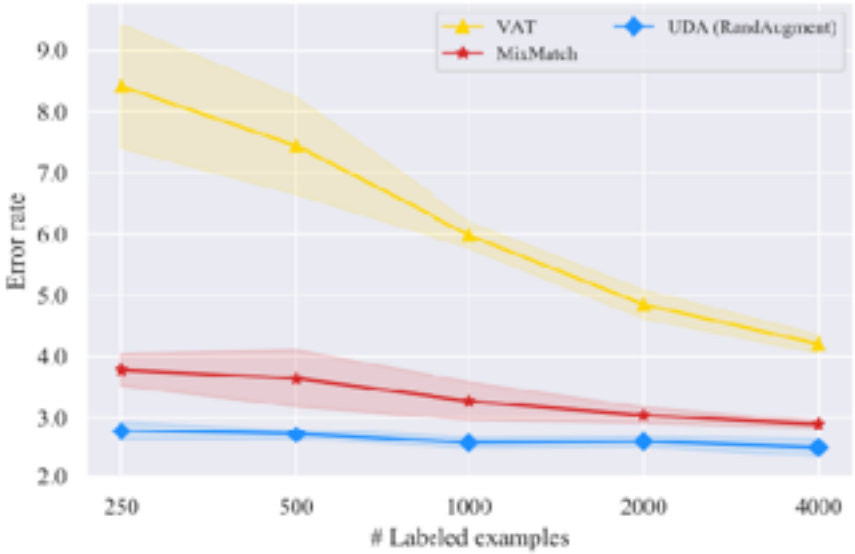
Table 1: Error rates on CIFAR-10.

Augmentation (# Sup examples)	Sup (650k)	Semi-sup (2.5k)
✗	38.36	50.80
Switchout	37.24	43.38
Back-translation	**36.71**	**41.35**

Table 2: Error rate on Yelp-5.



(a) CIFAR-10



(b) SVHN

Unsupervised Data Augmentation (UDA) for Consistency Training, Xie et al., Neurlps 2019

Unsupervised Consistency Loss

Method	Model	# Param	CIFAR-10 (4k)	SVHN (1k)
Π-Model (Laine & Aila, 2016)	Conv-Large	3.1M	12.36 ± 0.31	4.82 ± 0.17
Mean Teacher (Tarvainen & Valpola, 2017)	Conv-Large	3.1M	12.31 ± 0.28	3.95 ± 0.19
VAT + EntMin (Miyato et al., 2018)	Conv-Large	3.1M	10.55 ± 0.05	3.86 ± 0.11
SNTG (Luo et al., 2018)	Conv-Large	3.1M	10.93 ± 0.14	3.86 ± 0.27
VAdD (Park et al., 2018)	Conv-Large	3.1M	11.32 ± 0.11	4.16 ± 0.08
Fast-SWA (Athiwaratkun et al., 2018)	Conv-Large	3.1M	9.05	-
ICT (Verma et al., 2019)	Conv-Large	3.1M	7.29 ± 0.02	3.89 ± 0.04
Pseudo-Label (Lee, 2013)	WRN-28-2	1.5M	16.21 ± 0.11	7.62 ± 0.29
LGA + VAT (Jackson & Schulman, 2019)	WRN-28-2	1.5M	12.06 ± 0.19	6.58 ± 0.36
mixmixup (Hataya & Nakayama, 2019)	WRN-28-2	1.5M	10	-
ICT (Verma et al., 2019)	WRN-28-2	1.5M	7.66 ± 0.17	3.53 ± 0.07
MixMatch (Berthelot et al., 2019)	WRN-28-2	1.5M	6.24 ± 0.06	2.89 ± 0.06

Methods	SSL	10%	100%
ResNet-50	✗	55.09 / 77.26	77.28 / 93.73
w. RandAugment		58.84 / 80.56	78.43 / 94.37
UDA (RandAugment)	✓	**68.78 / 88.80**	**79.05 / 94.49**

Table 5: Top-1 / top-5 accuracy on ImageNet with 10% and 100% of the labeled set. We use image size 224 and 331 for the 10% and 100% experiments respectively.

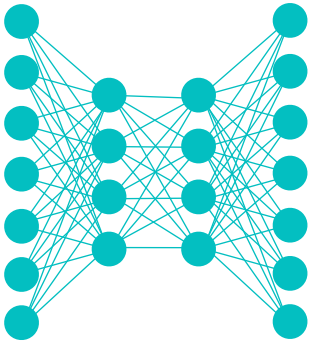
Lecture Notes for

Neural Networks and Machine Learning

Ada, SSL,

Next Time:
M-Modal/task

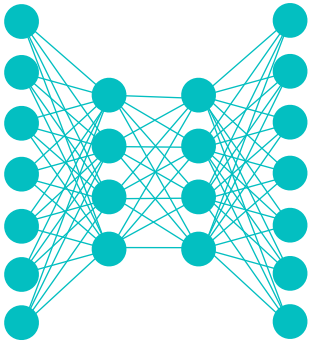
Reading: Papers



Lecture Notes for

Neural Networks and Machine Learning

Adaptive, Self-supervised, Multi-modal, & Multi-task Learning



Logistics and Agenda

- Logistics
  - Newest Lab uses multi-task / multi-modal learning

- Agenda
  - Adaptive Learning (last time)
  - Self-Supervised Learning (last time)
  - Paper Presentation: X-vectors (today)
  - Multi-modal/task Learning (today)
    - Techniques
    - Applications and domains
- Next Time:
  - Paper Presentation: Multi-task Methods in Chemistry

Consistency Loss

I'm from Canada, but live in the States now.

It took me a while to get used to writing boolean variables with an "Is" prefix, instead of the "Eh" suffix that Canadians use when programming.

For example:

```
MyObj.IsVisible

MyObj.VisibleEh
```


Unsupervised Consistency Loss

$$\min_{\mathbf{w}} \overbrace{\mathbf{E}_{\mathbf{x},y\in L}[-\log p_{\mathbf{w}}(y\,|\,\mathbf{x})]}^{\text{cross entropy}} + \lambda \overbrace{\mathbf{E}_{\mathbf{x}\in U}\mathbf{E}_{\hat{\mathbf{x}}\leftarrow q(\hat{\mathbf{x}}|\mathbf{x})}\left[\mathscr{D}_{KL}\left(p_{\mathbf{w}}(y\,|\,\mathbf{x})\,||\,p_{\mathbf{w}}(y\,|\,\hat{\mathbf{x}})\right)\right]}^{\text{consistency in augmentation}}$$

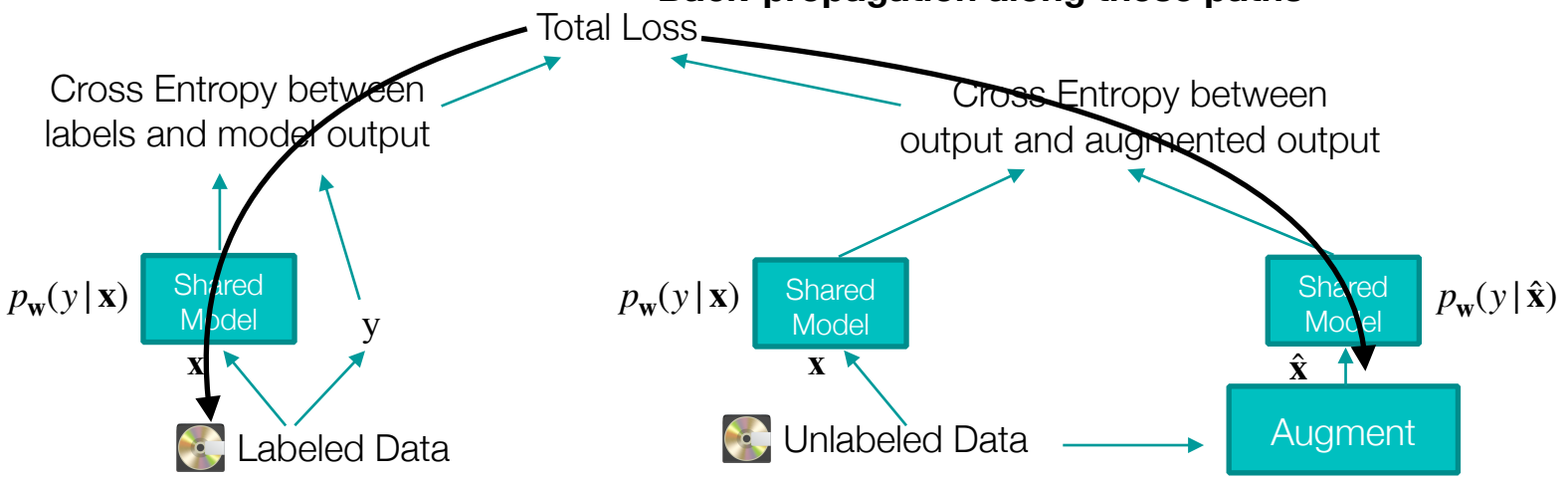
no back prop **yes** back prop

Neural Network approximates $p(y|\mathbf{x})$ by $\mathbf{w}$
Use labeled data to minimize network

Sample new $\mathbf{x}$ from unlabeled pool with function q
function q is augmentation procedure
Minimize cross entropy of two models

Get accustomed to this notation

Update Model with Back-propagation along these paths



Unsupervised Data Augmentation (UDA) for Consistency Training, Xie et al., NeurIps 2019

Unsupervised Consistency Loss

$$\min_{\mathbf{w}} \overbrace{\mathbf{E}_{\mathbf{x},y\in L}[-\log p_{\mathbf{w}}(y\,|\,\mathbf{x})]}^{\text{cross entropy}} + \lambda \overbrace{\mathbf{E}_{\mathbf{x}\in U}\mathbf{E}_{\hat{\mathbf{x}}\leftarrow q(\hat{\mathbf{x}}|\mathbf{x})}\left[\mathscr{D}_{KL}\left(p_{\mathbf{w}}(y\,|\,\mathbf{x})\,||\,p_{\mathbf{w}}(y\,|\,\hat{\mathbf{x}})\right)\right]}^{\text{consistency in augmentation}}$$

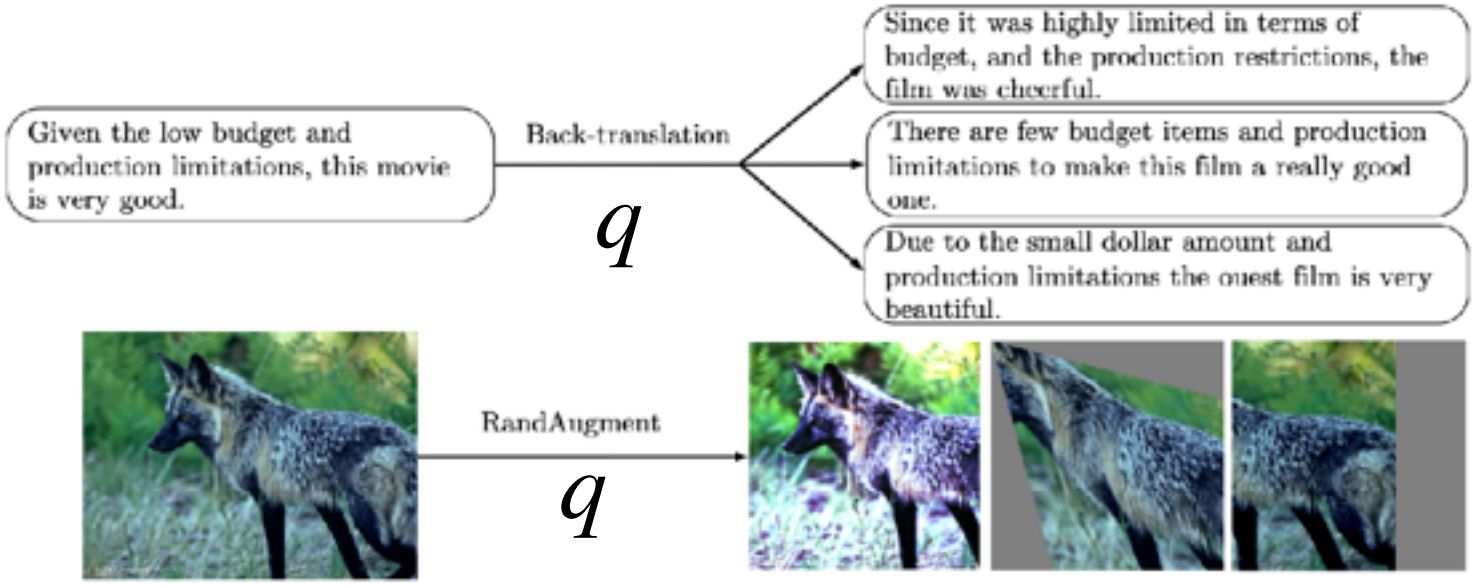


Figure 2: Augmented examples using back-translation and RandAugment.

Unsupervised Data Augmentation (UDA) for Consistency Training, Xie et al., Neurlps 2019

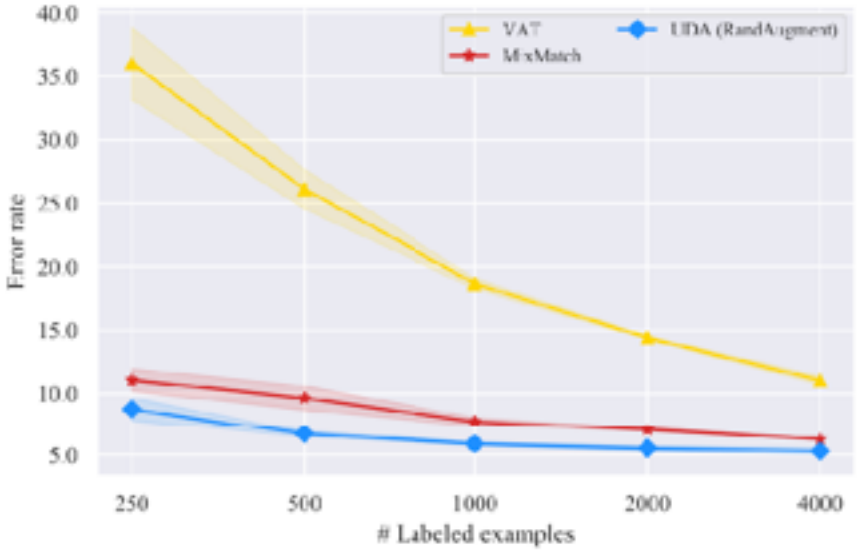
Unsupervised Consistency Loss

Augmentation (# Sup examples)	Sup (50k)	Semi-Sup (4k)
Crop & flip	5.36	16.17
Cutout	4.42	6.42
RandAugment	**4.23**	**5.29**

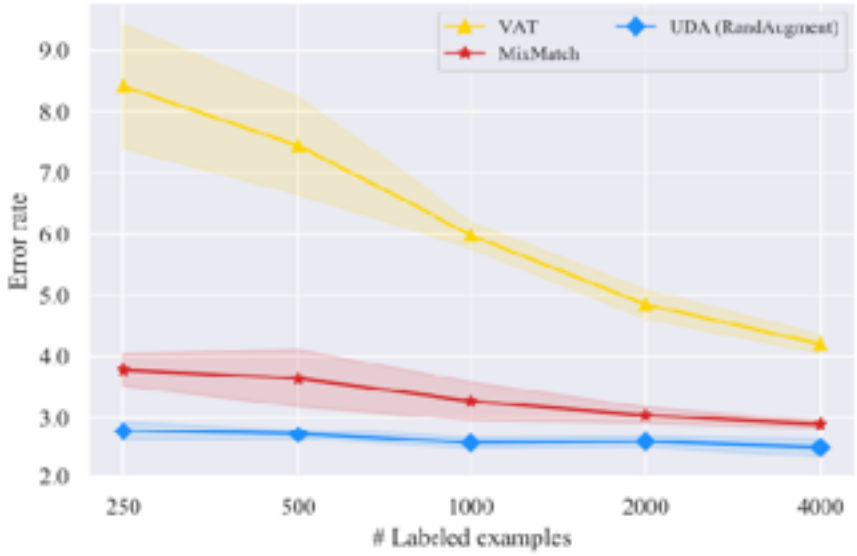
Table 1: Error rates on CIFAR-10.

Augmentation (# Sup examples)	Sup (650k)	Semi-sup (2.5k)
✗	38.36	50.80
Switchout	37.24	43.38
Back-translation	**36.71**	**41.35**

Table 2: Error rate on Yelp-5.



(a) CIFAR-10



(b) SVHN

Unsupervised Data Augmentation (UDA) for Consistency Training, Xie et al., Neurlps 2019

Unsupervised Consistency Loss

Method	Model	# Param	CIFAR-10 (4k)	SVHN (1k)
Π-Model (Laine & Aila, 2016)	Conv-Large	3.1M	12.36 ± 0.31	4.82 ± 0.17
Mean Teacher (Tarvainen & Valpola, 2017)	Conv-Large	3.1M	12.31 ± 0.28	3.95 ± 0.19
VAT + EntMin (Miyato et al., 2018)	Conv-Large	3.1M	10.55 ± 0.05	3.86 ± 0.11
SNTG (Luo et al., 2018)	Conv-Large	3.1M	10.93 ± 0.14	3.86 ± 0.27
VAdD (Park et al., 2018)	Conv-Large	3.1M	11.32 ± 0.11	4.16 ± 0.08
Fast-SWA (Athiwaratkun et al., 2018)	Conv-Large	3.1M	9.05	-
ICT (Verma et al., 2019)	Conv-Large	3.1M	7.29 ± 0.02	3.89 ± 0.04
Pseudo-Label (Lee, 2013)	WRN-28-2	1.5M	16.21 ± 0.11	7.62 ± 0.29
LGA + VAT (Jackson & Schulman, 2019)	WRN-28-2	1.5M	12.06 ± 0.19	6.58 ± 0.36
mixmixup (Hataya & Nakayama, 2019)	WRN-28-2	1.5M	10	-
ICT (Verma et al., 2019)	WRN-28-2	1.5M	7.66 ± 0.17	3.53 ± 0.07
MixMatch (Berthelot et al., 2019)	WRN-28-2	1.5M	6.24 ± 0.06	2.89 ± 0.06

Methods	SSL	10%	100%
ResNet-50	✗	55.09 / 77.26	77.28 / 93.73
w. RandAugment	✗	58.84 / 80.56	78.43 / 94.37
UDA (RandAugment)	✓	**68.78 / 88.80**	**79.05 / 94.49**

Table 5: Top-1 / top-5 accuracy on ImageNet with 10% and 100% of the labeled set. We use image size 224 and 331 for the 10% and 100% experiments respectively.

Paper Presentation: X-Vectors and SincNet Fusion

Speaker Recognition using SincNet and X-Vector Fusion

Mayank Tripathi, Divyanshu Singh, and Seba Susan (✉)[0000−0002−6709−6591]

Department of Information Technology
Delhi Technological University, Delhi 110042, India
{mayank_bt2k16,divyanshu_bt2k16}@dtu.ac.in, seba_406@yahoo.in

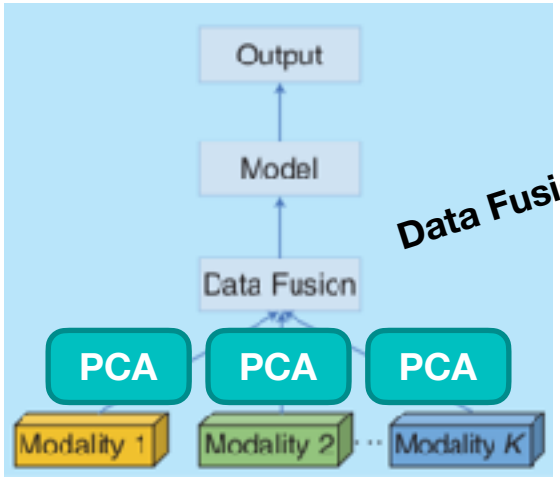
Multi-modal Review

Multi-modal == Multiple Data Sources

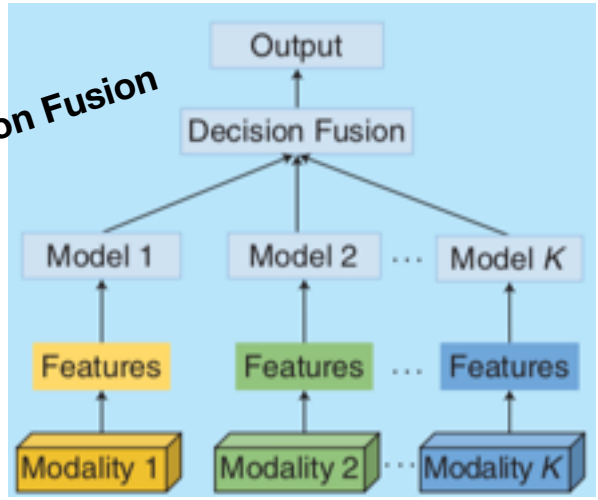
- **Modal** comes from the "sensor fusion" definition from Lahat, Adali, and Jutten (2015) for deep learning
- Using the Keras functional API, this is extremely easy to implement
  - … and we have used it since CS7324!
- But now let's take a deeper dive and ask:
  - What are the different types of modalities that we might try?
  - Is there a more optimal way to merge information?
  - When? Early, Intermediate, and late fusion

Early and Late Stage Fusion

- **Early Fusion:** Merge sensor layers early in the process
- **Assumption**: there is some data redundancy, but modes are conditionally independent
- **Problem**: architecture parameter explosion
  - Need dimensionality reduction



- **Late Fusion**: Merge sensor layers right before flattening
- Use Decision Fusion on outputs
- **Assumption**: little redundancy or conditional independence—just an ensemble architecture
- **Problem**: just separate classifiers, limited interplay

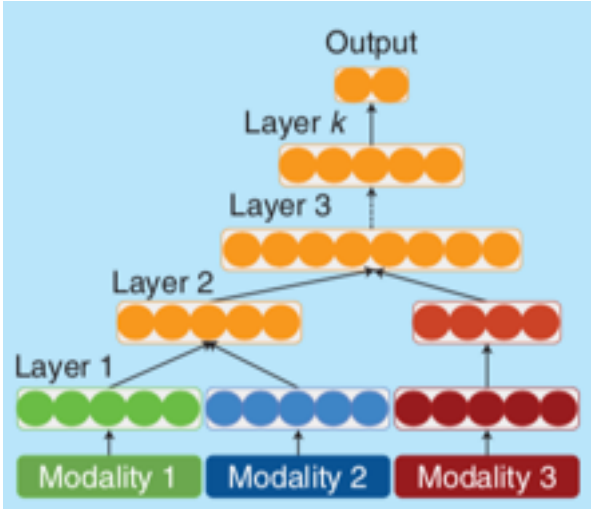


Ramamchandran and Taylor, 2017

Intermediate Fusion

- Merge sensor layers in soft way
- **Assumption**: some features interplay and others do not
- **Problem**: how to optimally tie layers together?

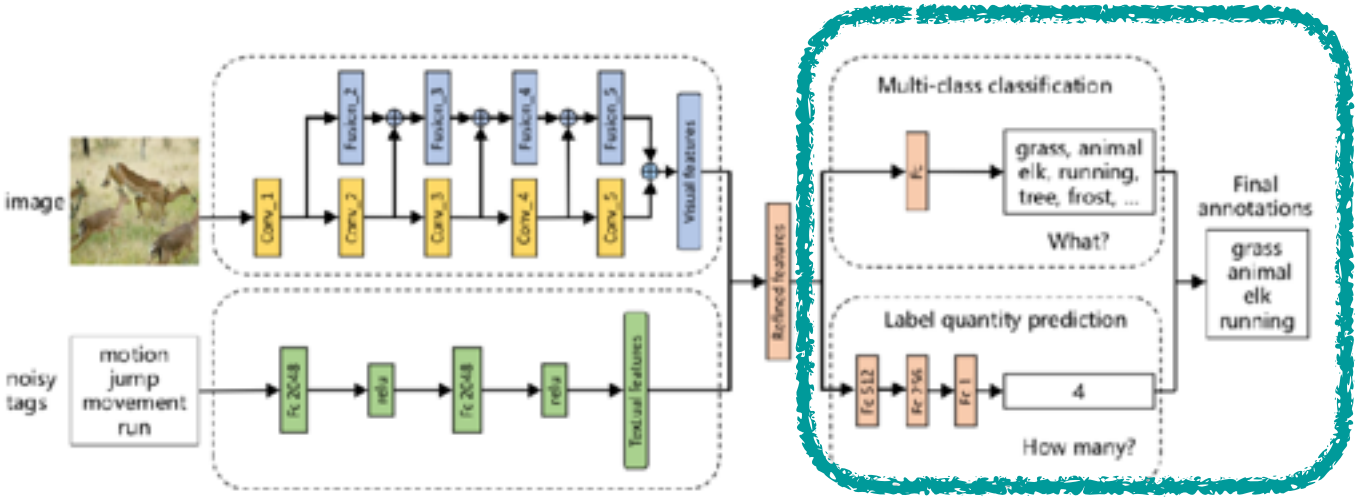
1. Stacked Auto-Encoders [Ding and Tao, 2015]
2. Early fuse layers that are correlated [Neverova *et al.* 2016]
3. Fully train each modality merge based on criterion of similarity in activations [Lu and Xu 2018]



Ramamchandran and Taylor, 2017

Multi-modal Merging

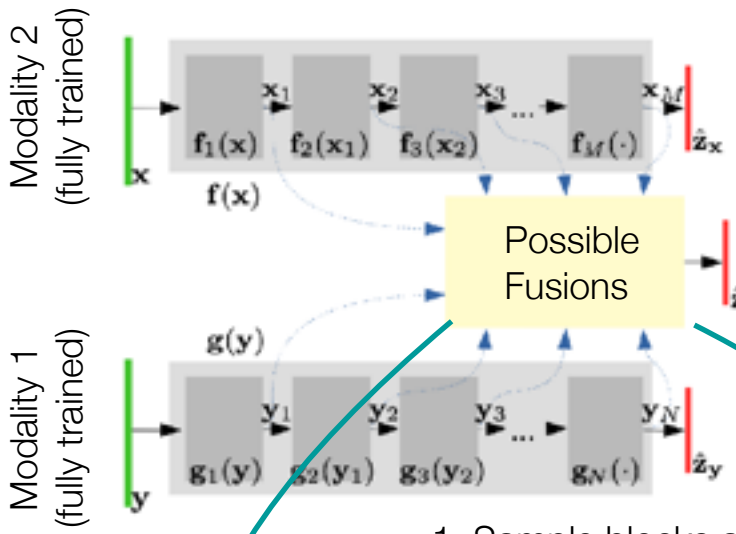
- **Still an open research problem**
- How to develop merging techniques that
  - Can handle exponentially many pairs of modalities
  - Automatically merge meaningful modes
  - Discard poor pairings
  - Selectively merge early or late (or dynamically)



https://arxiv.org/pdf/1709.01220.pdf

Most current methods are still ad-hoc

Neural Architecture Search for Mode Fusion

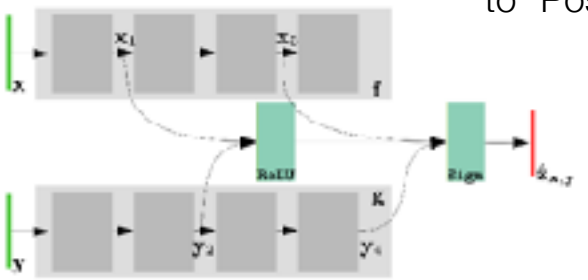


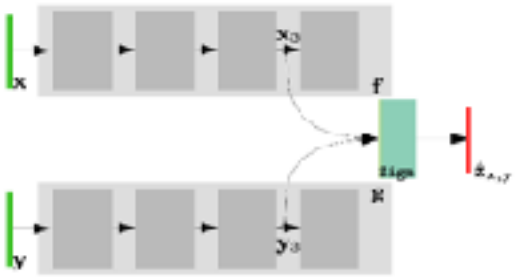
Genetic Algorithm

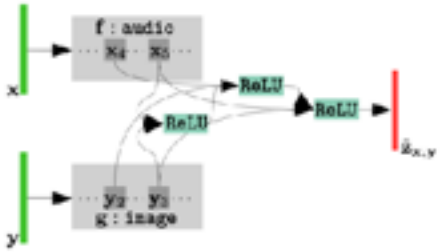
1. Sample new candidates
2. Evaluate fitnesses
3. Mutate and Crossover
4. Keep the best solutions
5. Repeat

Very computational when starting, because candidates are all untrained.
However, as more blocks start from "mostly trained" positions, training becomes faster.

1. Sample blocks at Random
2. Train by fine tuning
3. Evaluate fitness
4. Add trained blocks back to "Possible Fusions"





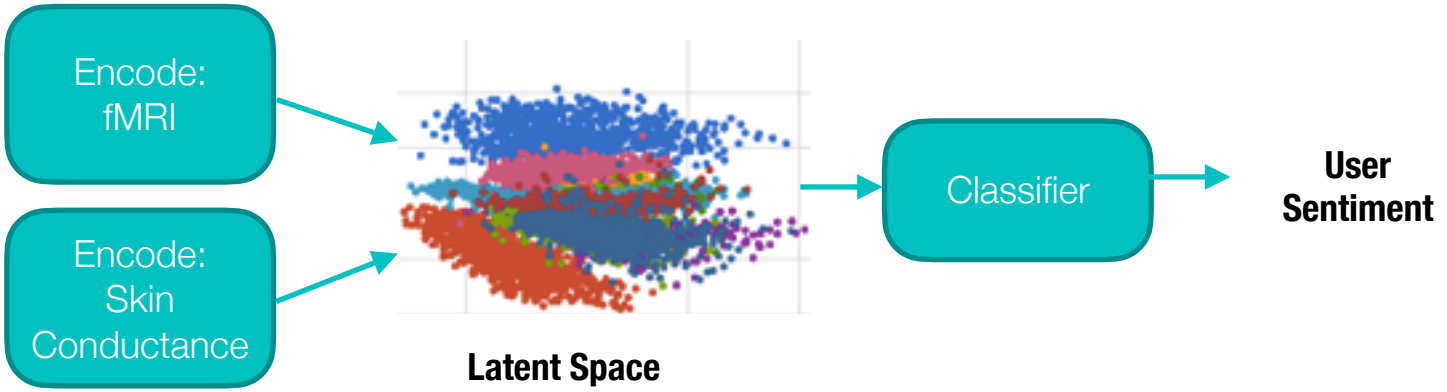


Found solution for AV-MNIST

Pérez-Rúa, Juan-Manuel, Valentin Vielzeuf, Stéphane Pateux, Moez Baccouche, and Frédéric Jurie. "Mfas: Multimodal fusion architecture search." In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 6966-6975. 2019.

Approaches with Deep Learning

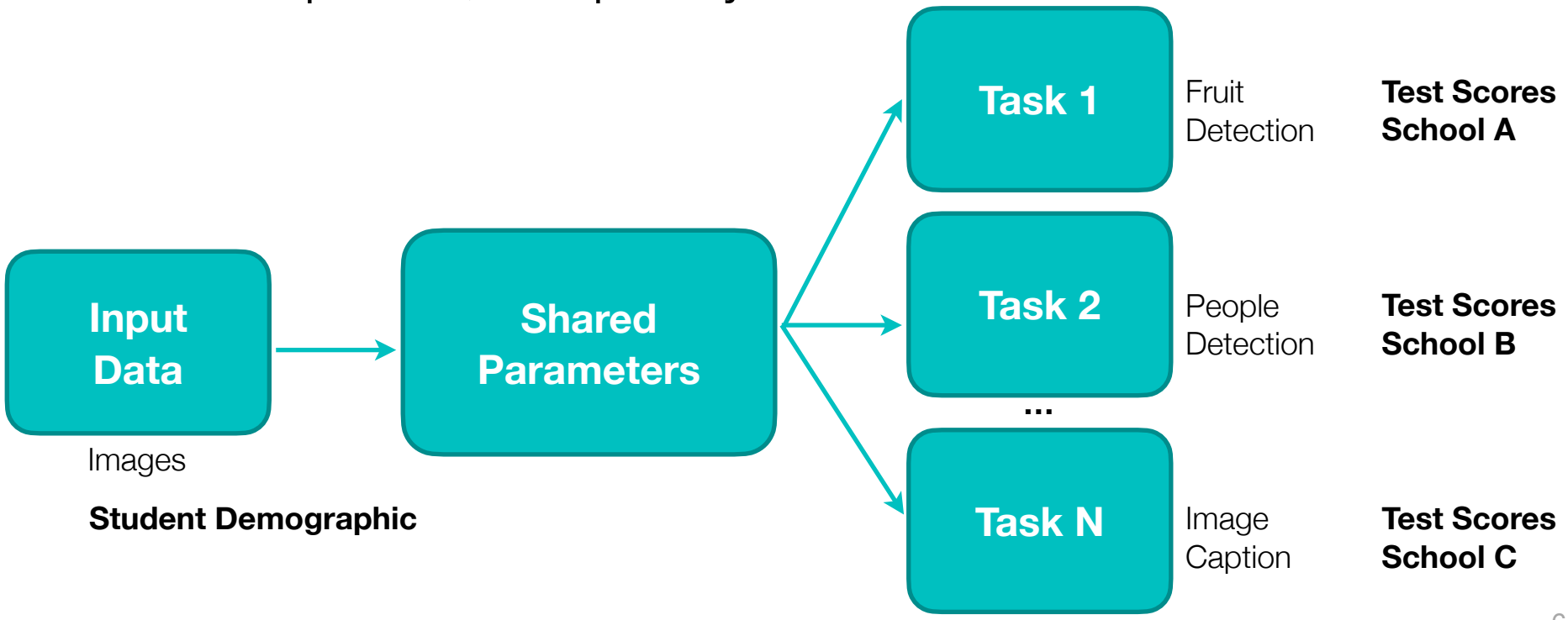
- Latent Space Transfer (universality)
  - From another domain, map to a similar latent space for the same task
  - Useful for unifying data based upon a new input mode when old mode is well understood
    - for example, biometric data
    - **I have never seen a research paper on this…**



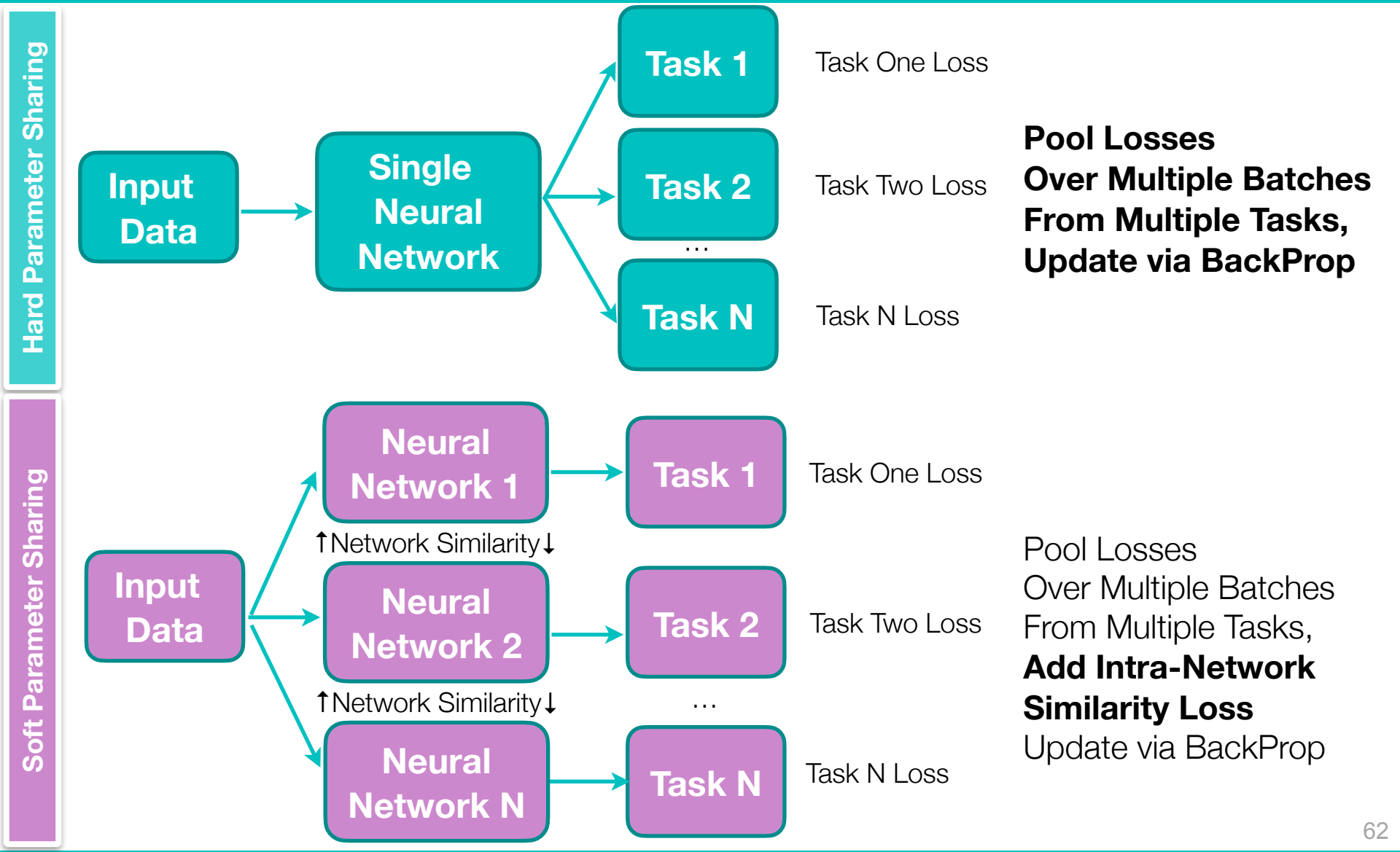
Multi-Task Models

Multi-task learning overview

- For deep networks, simple idea: share parameters in early layers
- Used shared parameters as feature extractors
- Train separate, unique layers for each task

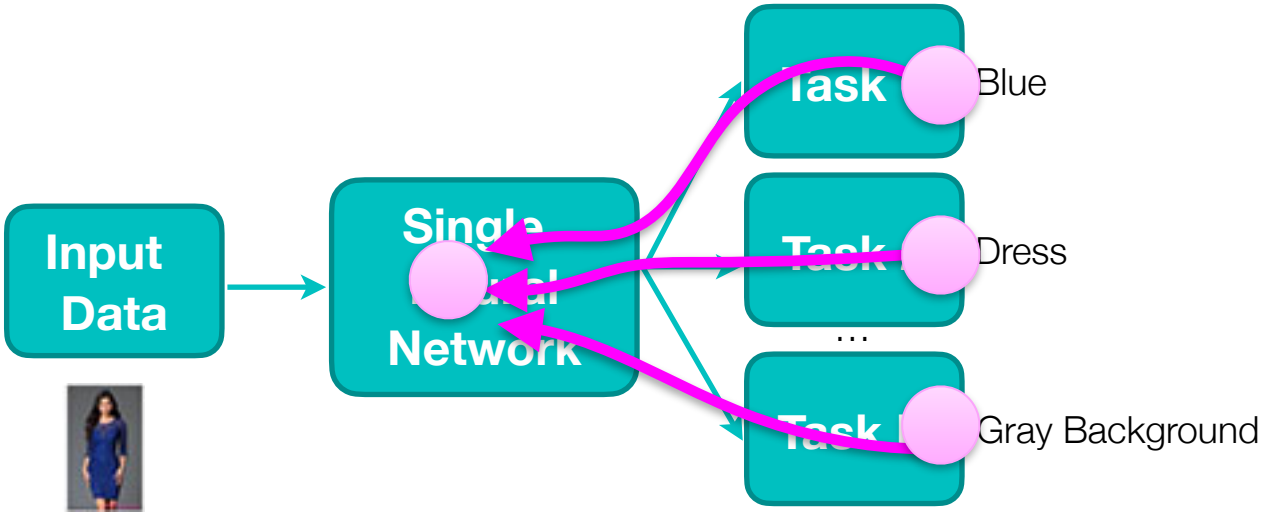


Multi-task Learning Parameter Sharing



Multi-task Optimization

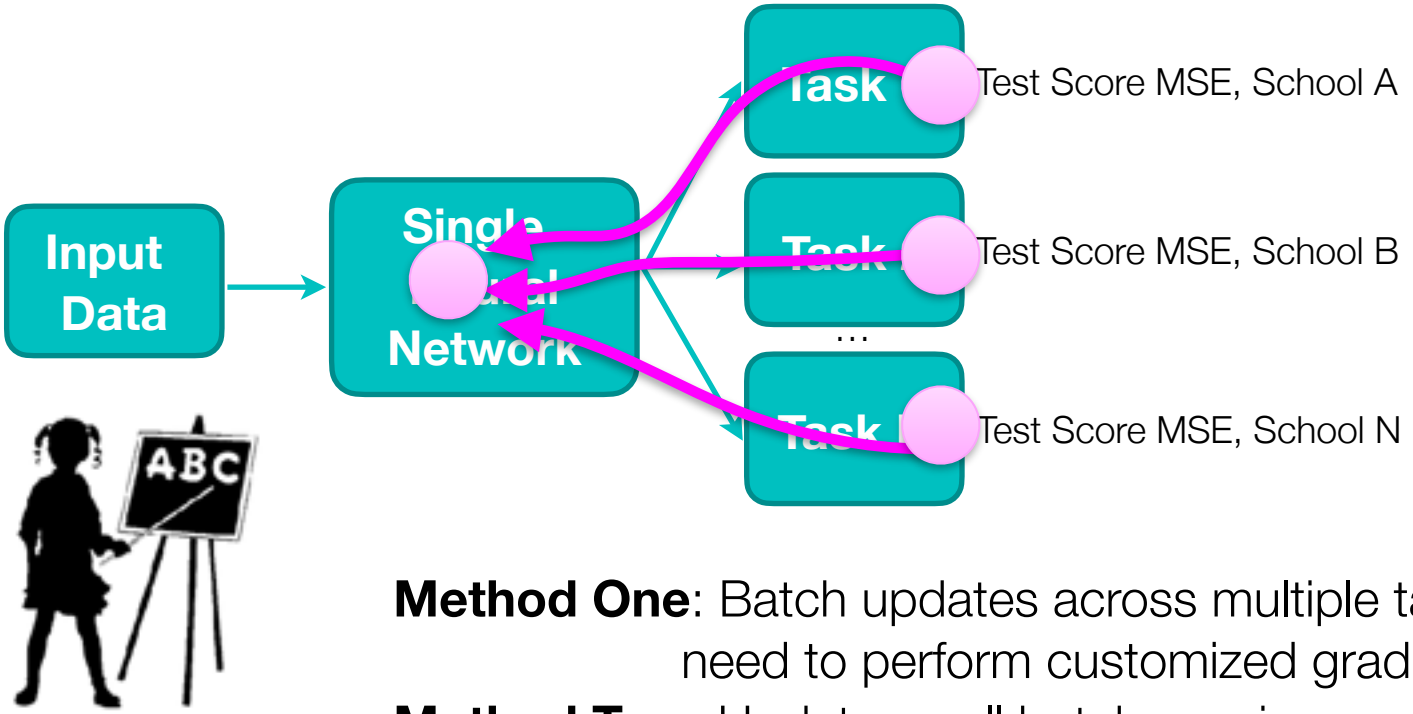
Multi-Label per Input



Measure Loss for each label simultaneously
Back propagate everything at one for a given batch

Multi-task Optimization

Single Task Label per Input



Method One: Batch updates across multiple tasks
need to perform customized gradient calculations

Method Two: Update small batches using a random task
easier, but can cause instability in training

Multi-Task Learning in Keras with Multi-Label Data

Fashion week, colors and dresses

"if time"

Follow Along: https://www.pyimagesearch.com/2018/06/04/keras-multiple-outputs-and-multiple-losses/

Multi-Task Learning

School Data, Computer Surveys

“if time”

Traian Pop

Luke Wood

```
Follow Along: LectureNotesMaster/
05 LectureMultiTask.ipynb
```


Next Time

- Multi-task demonstrations with various datasets
- Paper Presentations

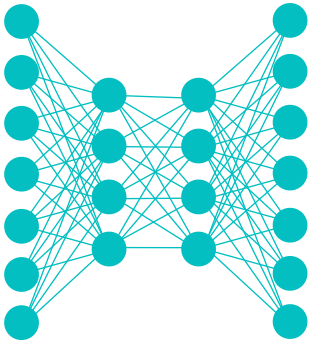
Lecture Notes for

Neural Networks and Machine Learning

Multi-Modal and Multi-Task

Next Time:
Demo

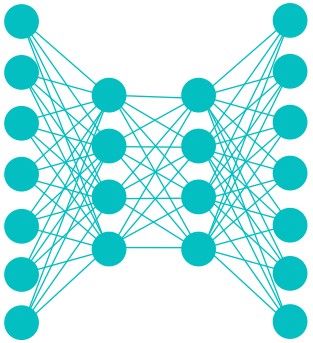
Reading: Papers



Lecture Notes for

Neural Networks and Machine Learning

Multi-Task Demo

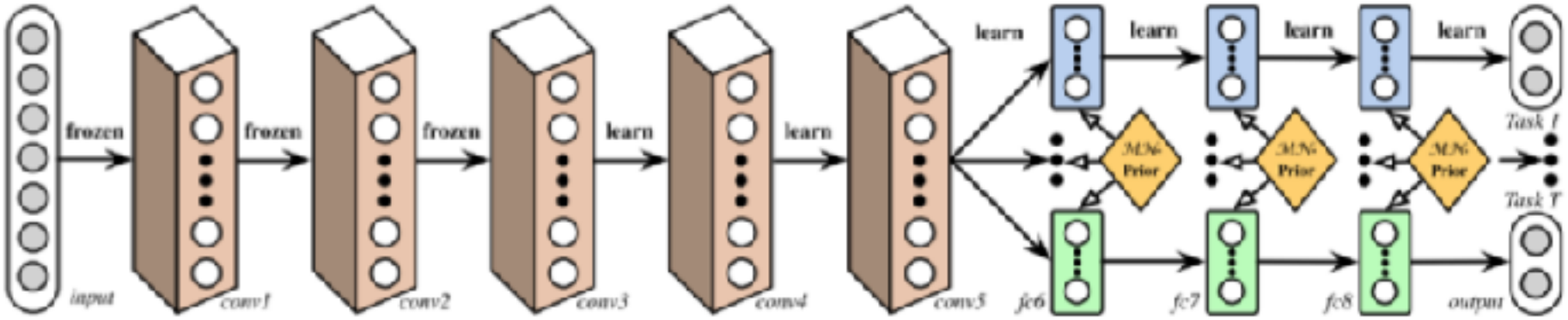


Logistics and Agenda

- Logistics
  - None!
- Agenda
  - Multi-Task Examples
  - Paper Presentation
  - Multi-Task Demos
  - Multi-Task Town Hall
- Next Time
  - Variational Auto-Encoders

Multi-Task Model Examples

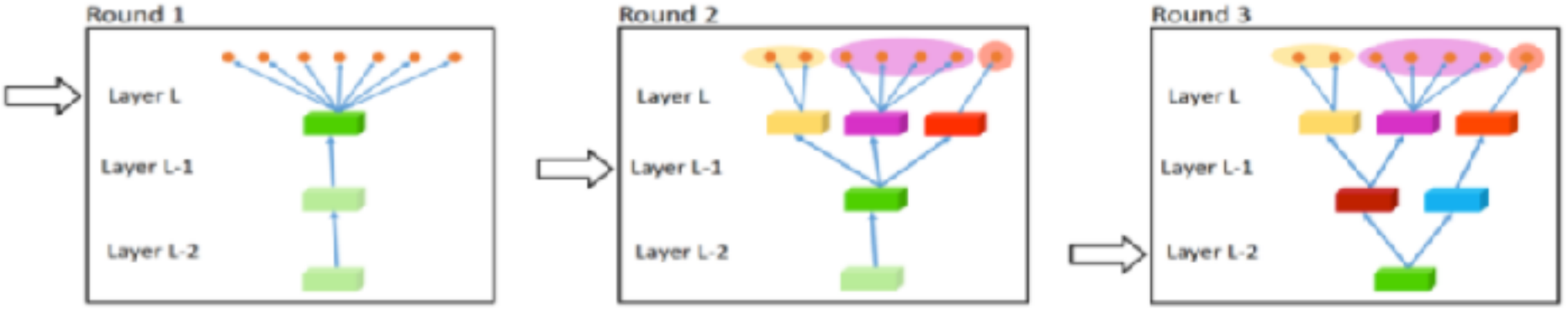
Multi-task: Deep Relationship Networks



- Start training traditionally
- Minimize Kroenecker Product between fully connected task specific layers
  - that is, make Covariance between layers close to identity
  - encourages feature maps in each task to be **less correlated** to feature maps of another task

https://arxiv.org/pdf/1506.02117.pdf

Multi-task: Adaptive Feature Sharing



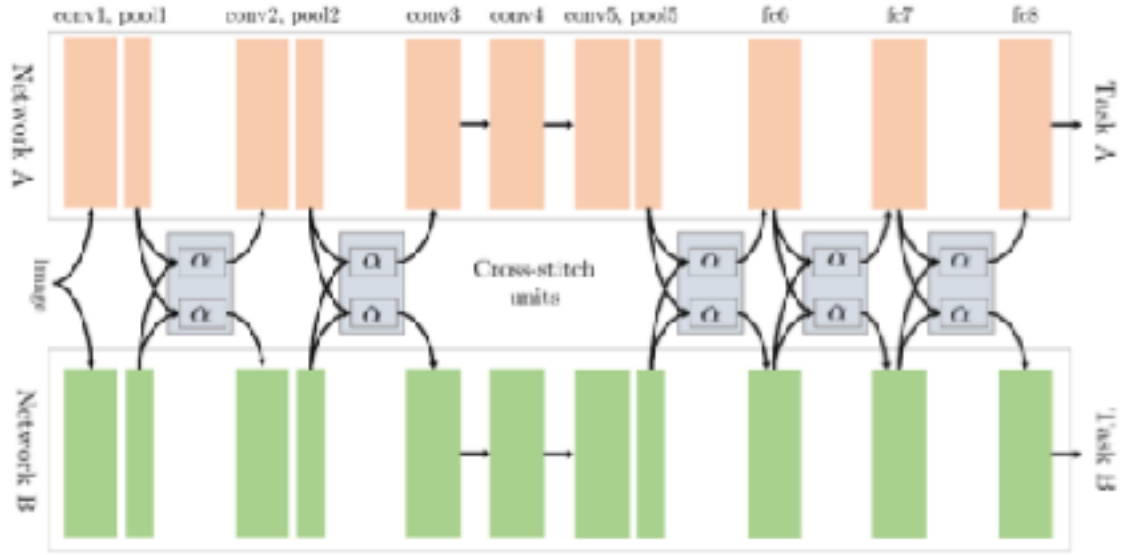
- Train
- Repe
  - D
  - Cut weights and fine tune network
  - Decrement current layer index

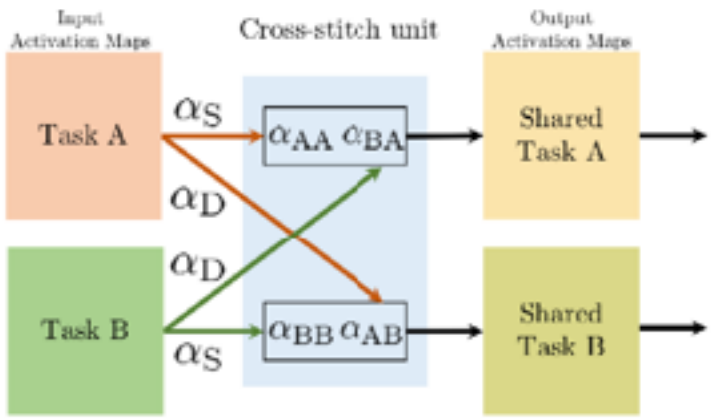
$$A^\star, \omega^\star(l) = \underset{A \in \mathbb{R}^{d \times d'}, |\omega| = d'}{\arg\min} ||W^{p,l} - A W^{p,l}_{\omega:}||_F, \qquad (2)$$

where $W^{p,l}_{\omega:}$ is a truncated weight matrix that only keeps the rows indexed by the set ω. This problem is NP-hard, however, there exist approaches based on convex relaxation

http://openaccess.thecvf.com/content_cvpr_2017/papers/Lu_Fully-Adaptive_Feature_Sharing_CVPR_2017_paper.pdf

Multi-task: Cross Stitch Networks

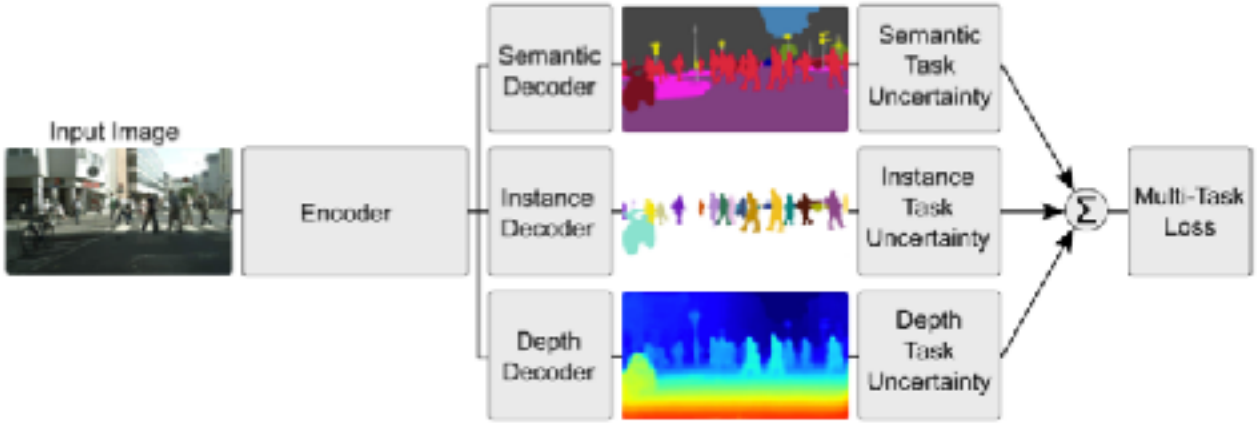




- Only works for simultaneous multi-label problems
  - like semantic segmentation and surface normal segmentation (clustering similarly facing objects)
- Take a learned weighted sum of the activations
- Works a little better than single task, but no worse

https://arxiv.org/pdf/1604.03539.pdf

Multi-task: Uncertainty Weighting



- Use variance of each loss function from each task to normalize
  - call it homoscedastic without sound reasoning because that feels better than “normalized variance”
  - talk about homoscedasticity for no reason
- Write an entire paper in a “mathy” way to make it seem like more of a contribution
- Profit because you are Oxford/Cambridge and reviewers give you a pass

https://arxiv.org/pdf/1705.07115.pdf

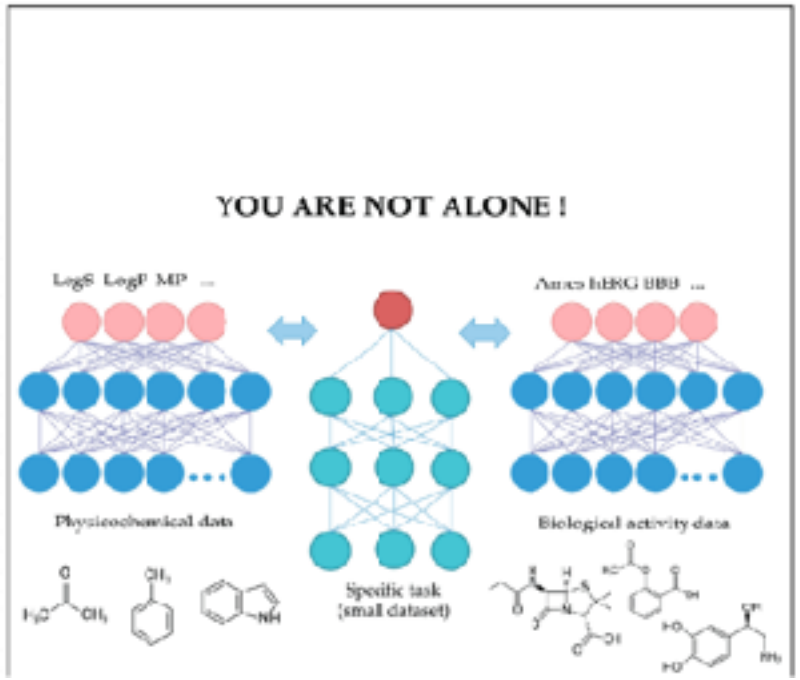
Current Multi-task Research

- Incredibly diverse sets of solutions
- Mostly not evaluated on similar datasets
- Reasoning given is mostly ad-hoc…
- Theory is wildly under developed
  - because the problem is incredibly difficult
- Neural architecture search is an option…

Paper Presentation: Multi-task with Chemical Fingerprints

A Survey of Multi-task Learning Methods in Chemoinformatics

Sergey Sosnin,[a] Mariia Vashurina,[a] Michael Withnall,[c] Pavel Karpov,[b] Maxim Fedorov,[a,d] and Igor V. Tetko*[a]



Multi-Task Learning in Keras with Multi-Label Data

Fashion week, colors and dresses

"finish demo"

Follow Along: https://www.pyimagesearch.com/2018/06/04/keras-multiple-outputs-and-multiple-losses/

Multi-Task Learning

School Data, Computer Surveys

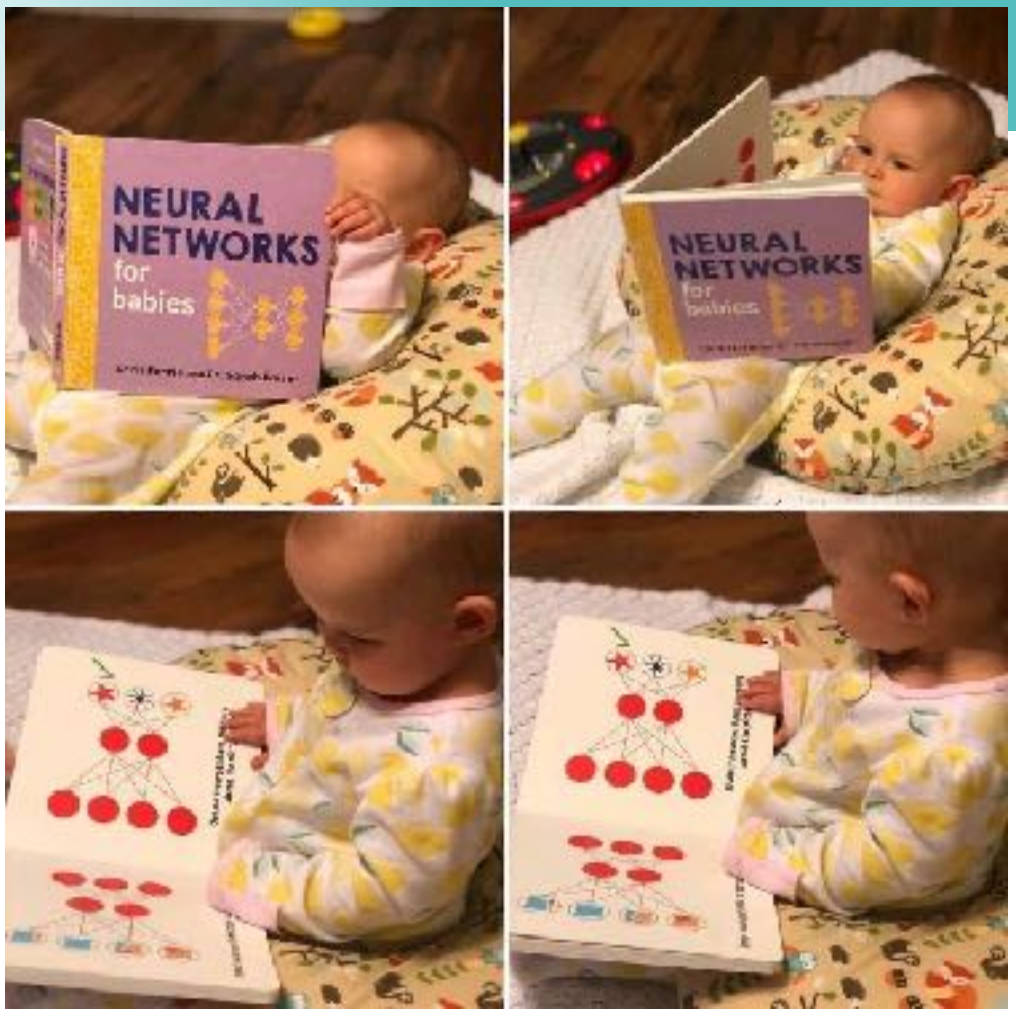
“finish demo”

Traian Pop

Luke Wood

```
Follow Along: LectureNotesMaster/
05 LectureMultiTask.ipynb
```


Lab Three Town Hall



Multi-Task Networks
Multi-Modal Networks

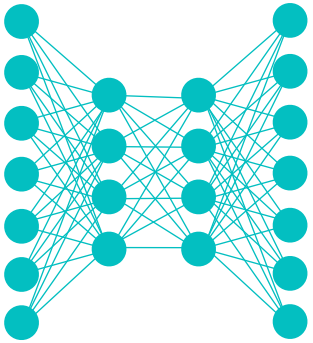
Lecture Notes for

Neural Networks and Machine Learning

Demo Multi-Task

Next Time:
GANs

Reading: Chollet 8.1-8.5



Backup slides

Title Between Topics

Example Slide

Title

Subtitle

`Follow Along: Notebook Name`