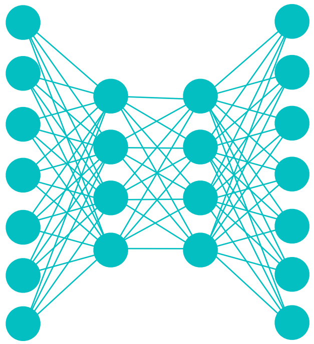
Lecture Notes for

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

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

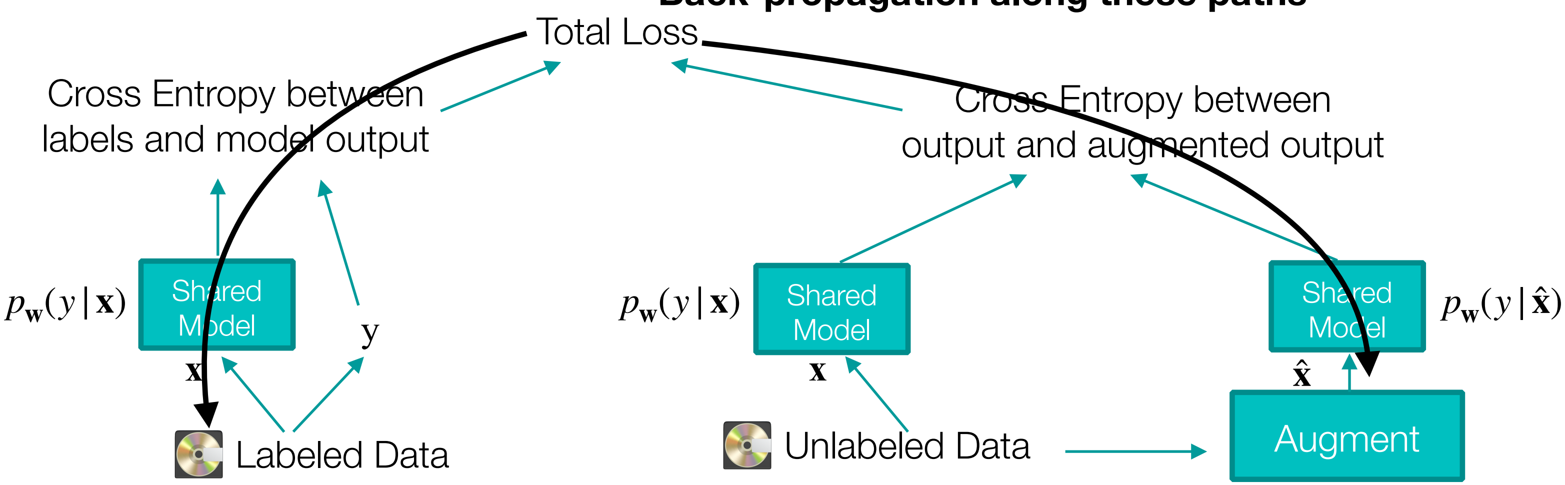
no back prop **yes** back prop

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

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

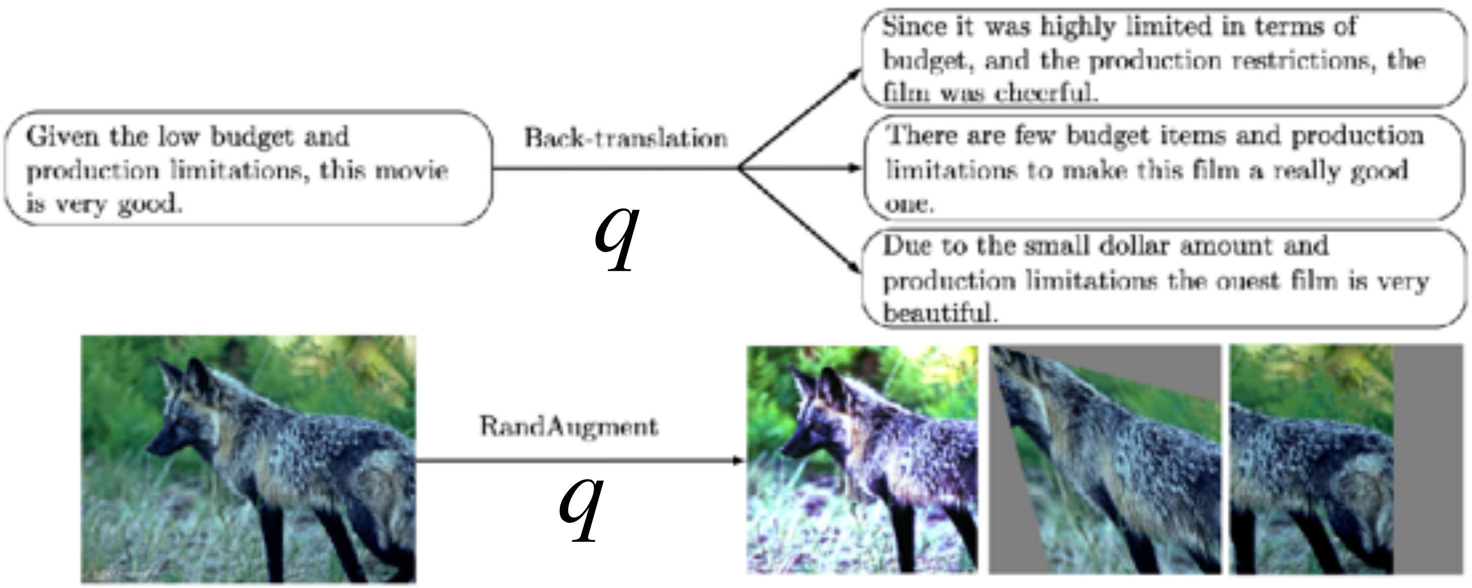


Figure 2: Augmented examples using back-translation and RandAugment.

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

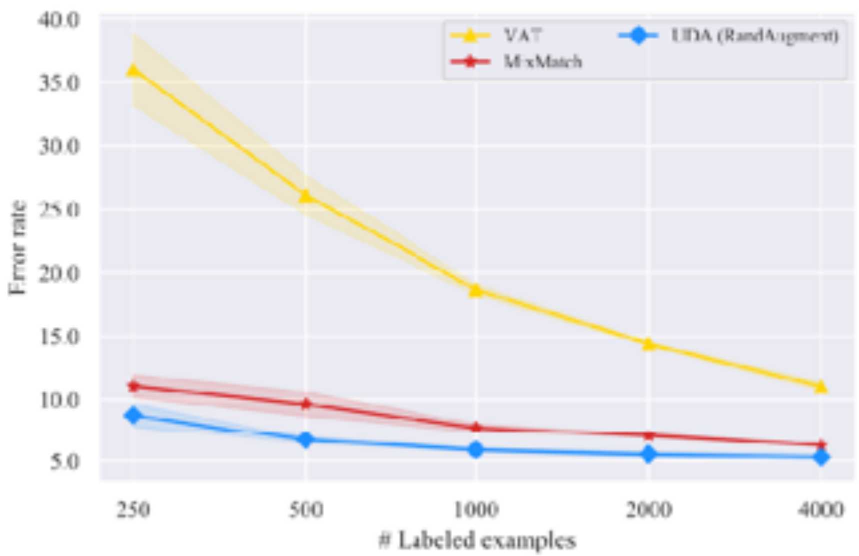
Unsupervised Consistency Loss

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

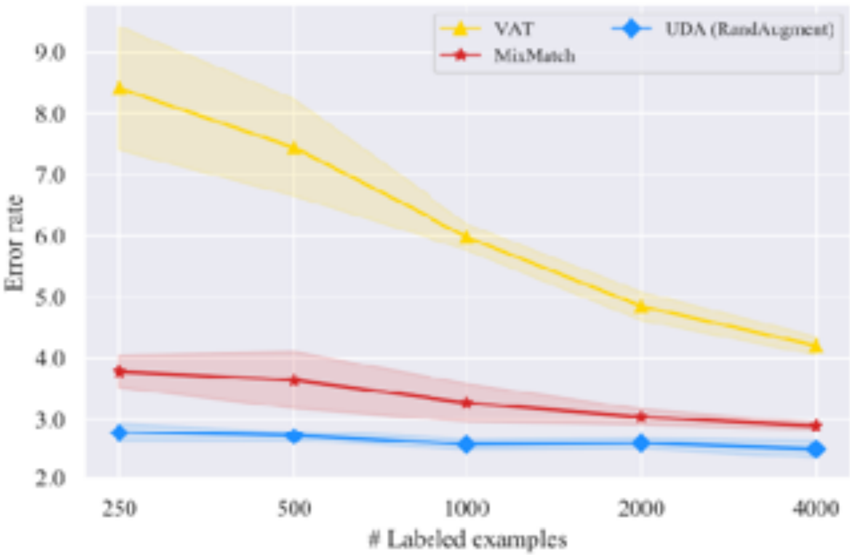
Table 1: Error rates on CIFAR-10.

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., NeurIps 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

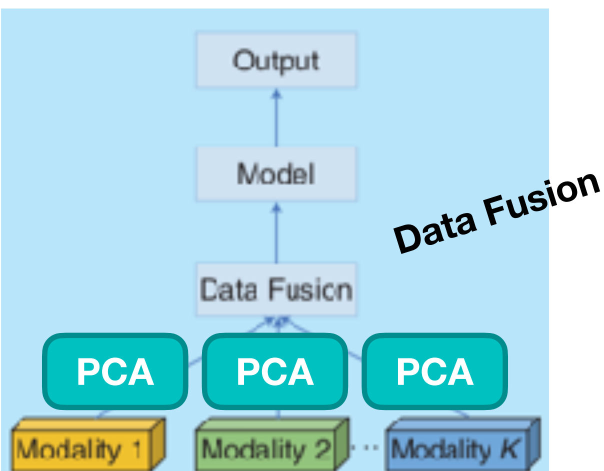
Multi-modal Review

Multi-modal == Multiple Data Sources

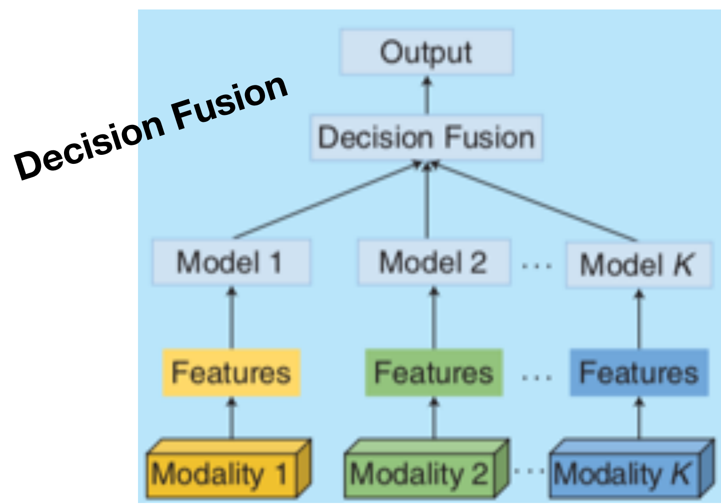
- **Modal** comes from the “sensor fusion” definition from Lahat, Adali, and Jutten (2015) for deep learning
- Using the Keras functional API, this is extremely easy to implement
  - … and we have used it since CS7324!
- But now let’s take a deeper dive and ask:
  - What are the different types of modalities 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

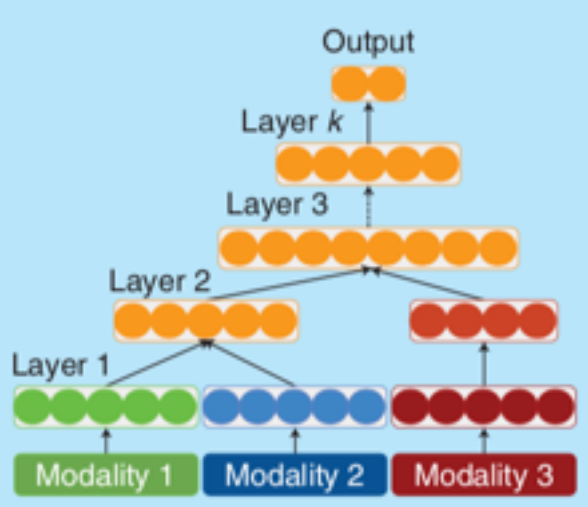


Ramamchandran and Taylor, 2017

Intermediate Fusion

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

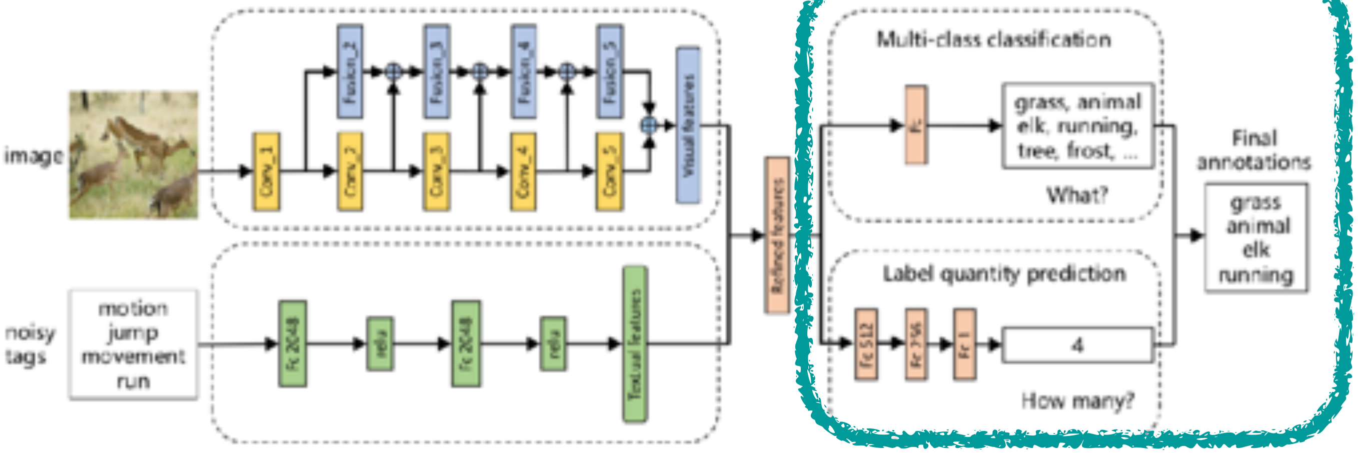
1. Stacked Auto-Encoders
   [Ding and Tao, 2015]
2. Early fuse layers that are correlated
   [Neverova *et al.* 2016]
3. Fully train each modality merge based on criterion of similarity in activations
   [Lu and Xu 2018]



Ramamchandran and Taylor, 2017

Multi-modal Merging

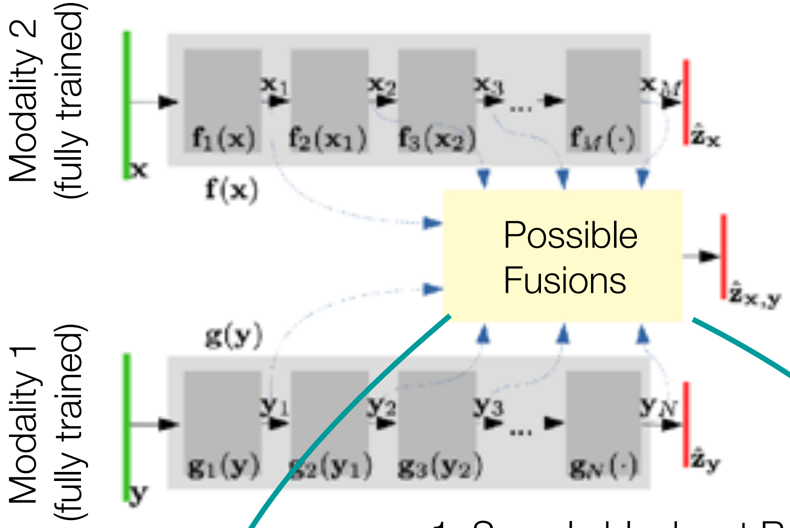
- **Still an open research problem**
- How to develop merging techniques that
  - Can handle exponentially many pairs of modalities
  - Automatically merge meaningful modes
  - Discard poor pairings
  - Selectively merge early or late (or dynamically)



Most current methods are still ad-hoc

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

Neural Architecture Search for Mode Fusion

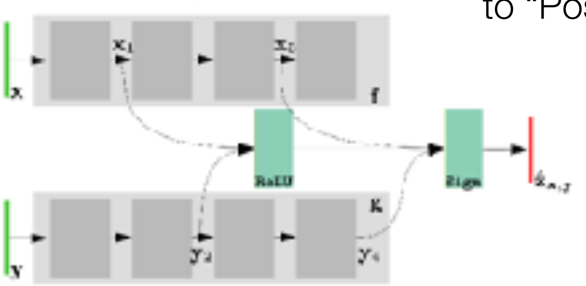


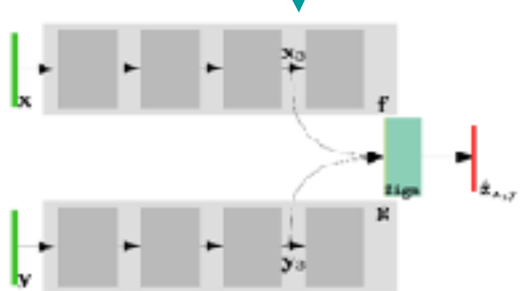
Genetic Algorithm

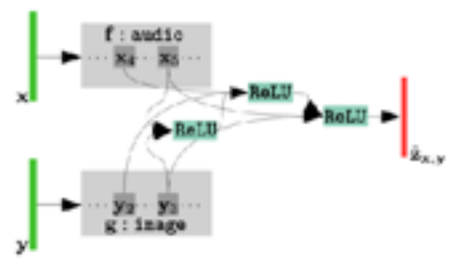
1. Sample new candidates
2. Evaluate fitnesses
3. Mutate and Crossover
4. Keep the best solutions
5. Repeat

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"





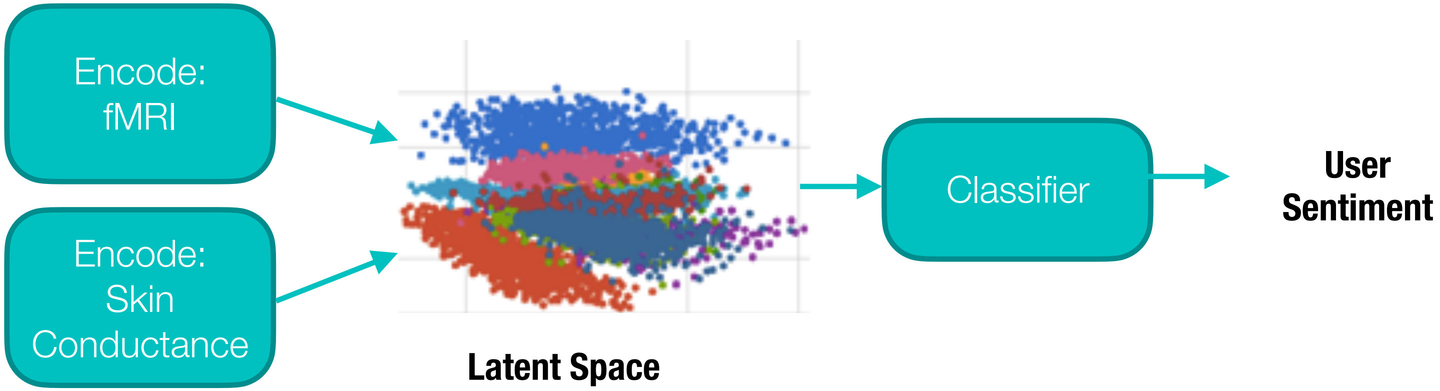


Found solution for AV-MNIST

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

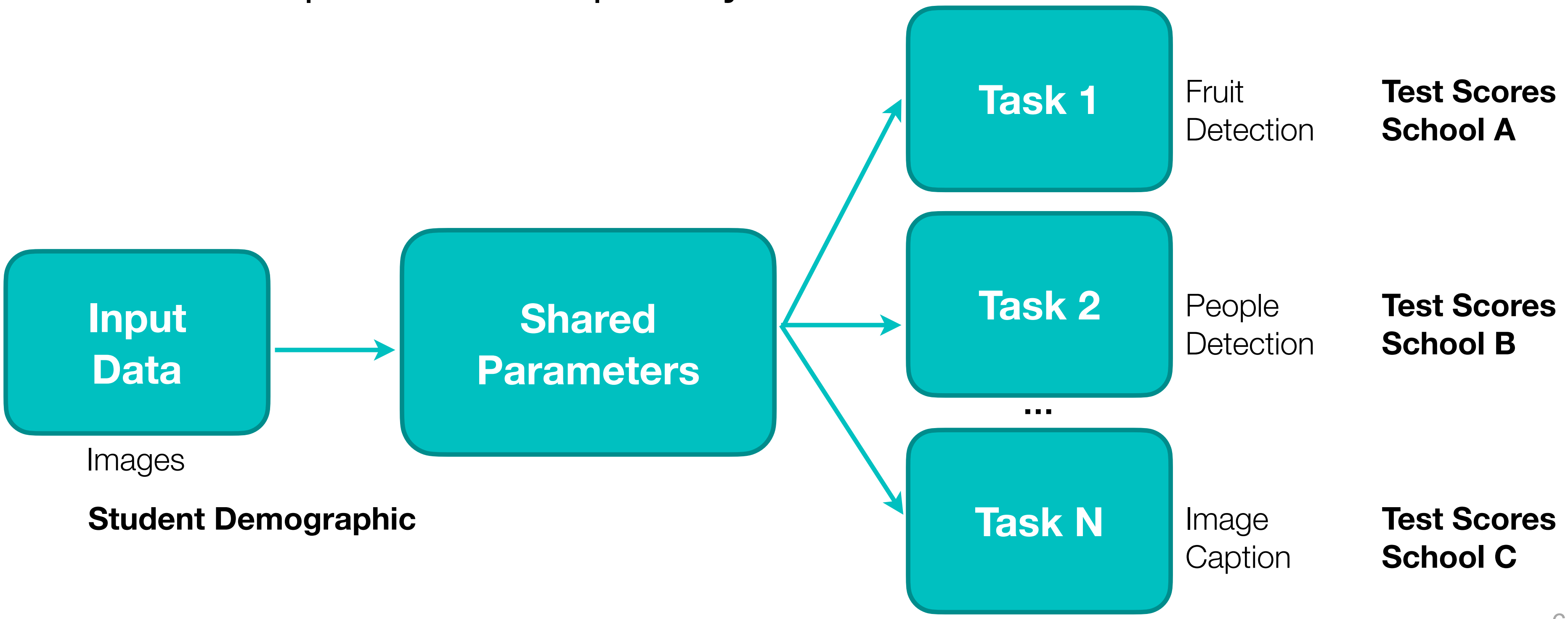
- Latent Space Transfer (universality)
  - From another domain, map to a similar latent space for the same task
  - Useful for unifying data based upon a new input mode when old mode is well understood
    - for example, biometric data
    - **I have never seen a research paper on this…**



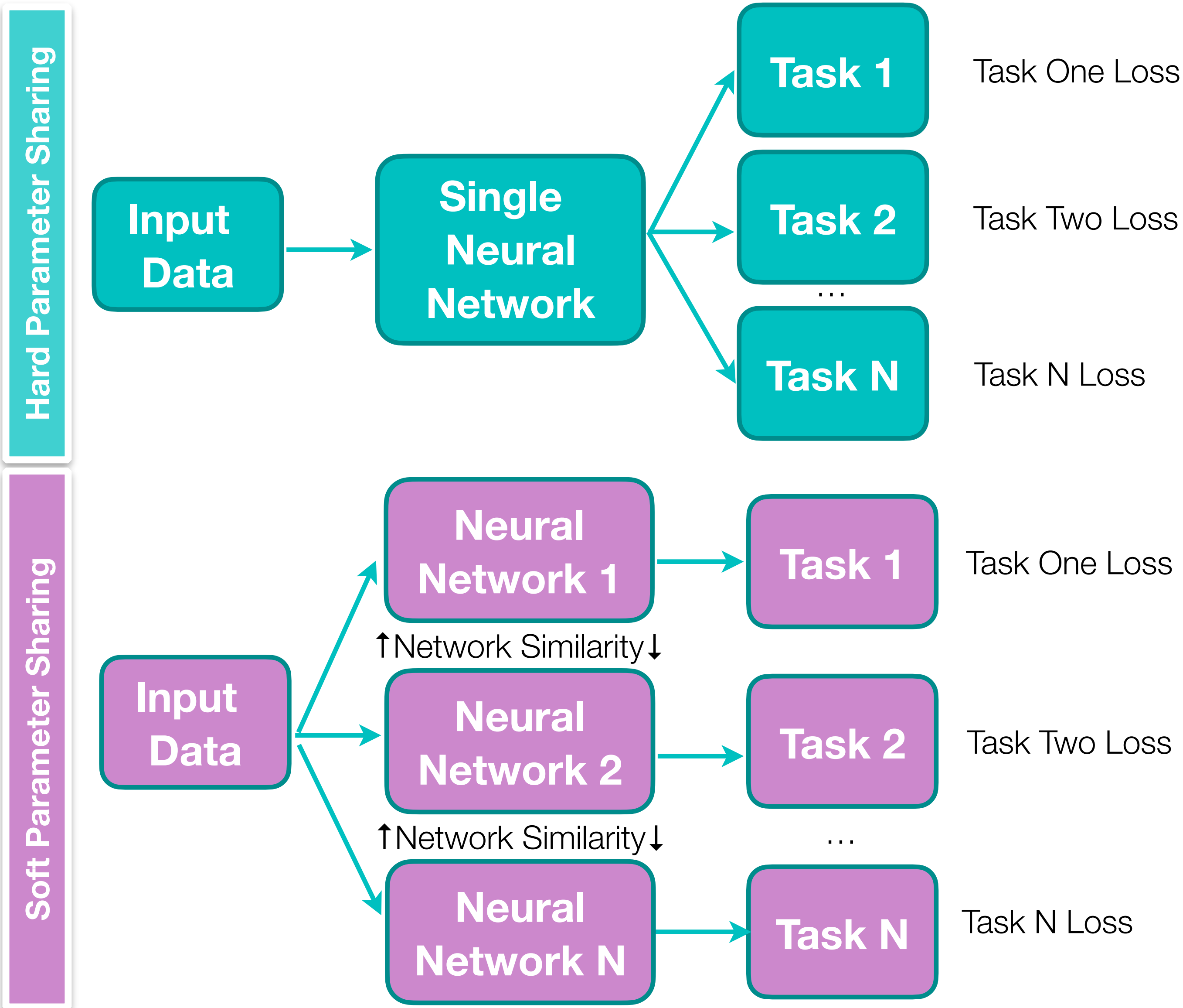
Multi-Task Models

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

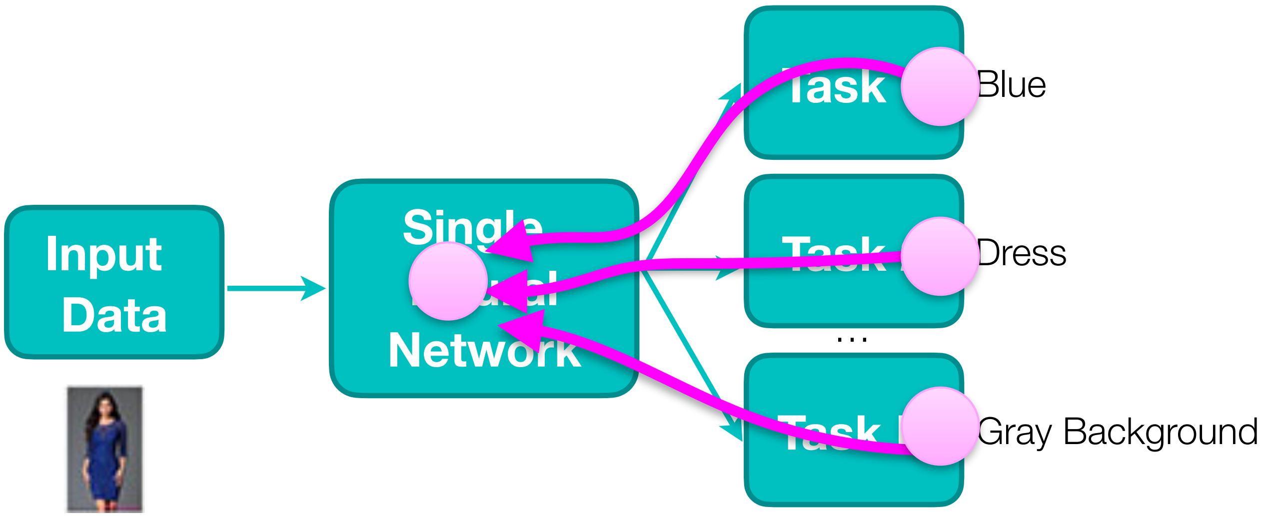


Pool Losses
Over Multiple Batches
From Multiple Tasks,
Update via BackProp

Pool Losses
Over Multiple Batches
From Multiple Tasks,
Add Intra-Network
Similarity Loss
Update via BackProp

Multi-task Optimization

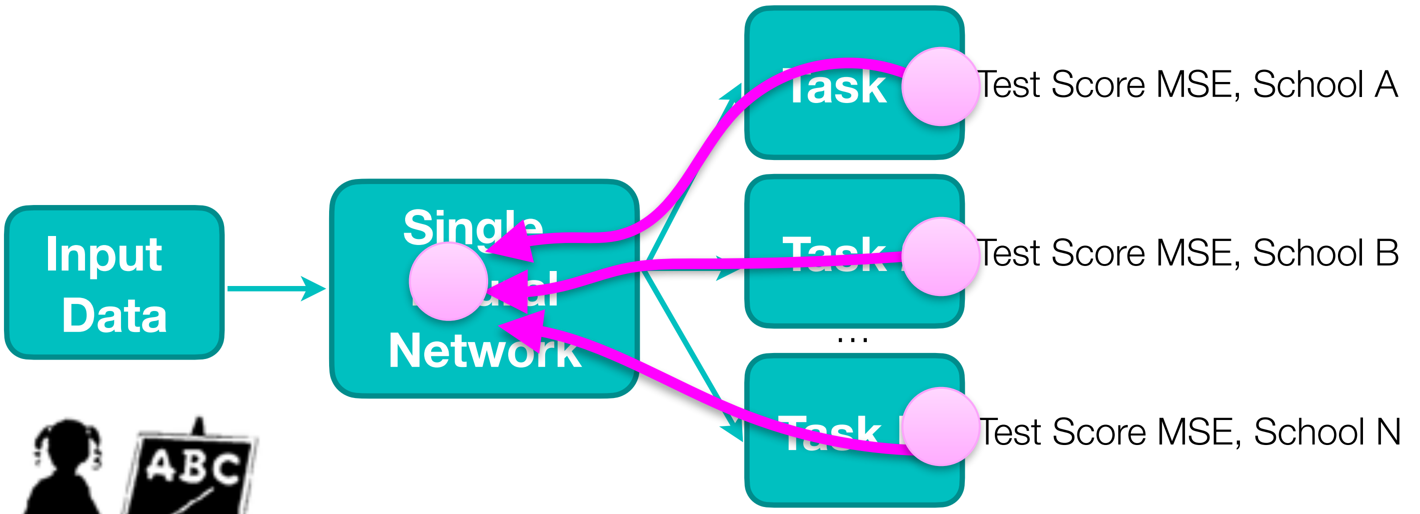
Multi-Label per Input



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

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