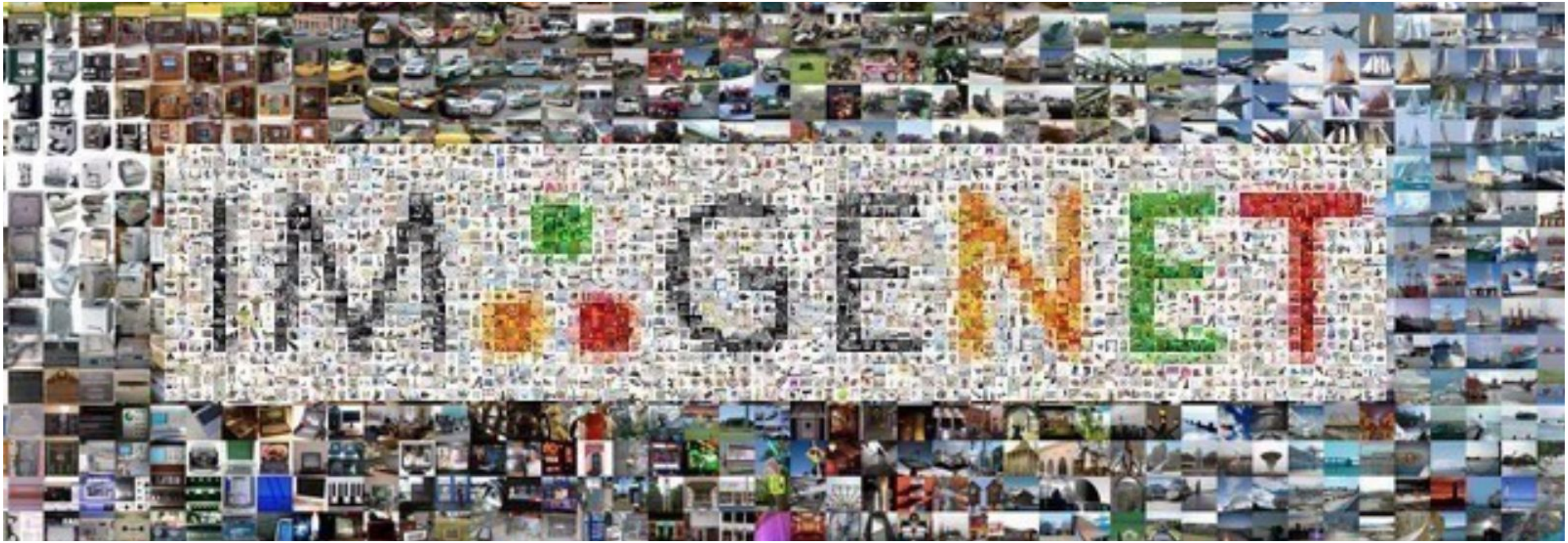
Meta-Learning for Semi-Supervised Few-Shot Classification

Mengye Ren[†⋈], **Eleni Triantafillou**[*†⋈], **Sachin Ravi**[*§], **Jake Snell**[†⋈], **Kevin Swersky**[¶],
Joshua B. Tenenbaum[♮], **Hugo Larochelle**[¶‡] **& Richard S. Zemel**[†‡⋈]
[†]University of Toronto, [§]Princeton University, [¶]Google Brain, [♮]MIT, [‡]CIFAR, [⋈]Vector Institute
{mren,eleni}@cs.toronto.edu, sachinr@cs.princeton.edu,
jsnell@cs.toronto.edu, kswersky@google.com,
jbt@mit.edu, hugolarochelle@google.com, zemel@cs.toronto.edu

CS330 Paper Presentation: October 16th, 2019

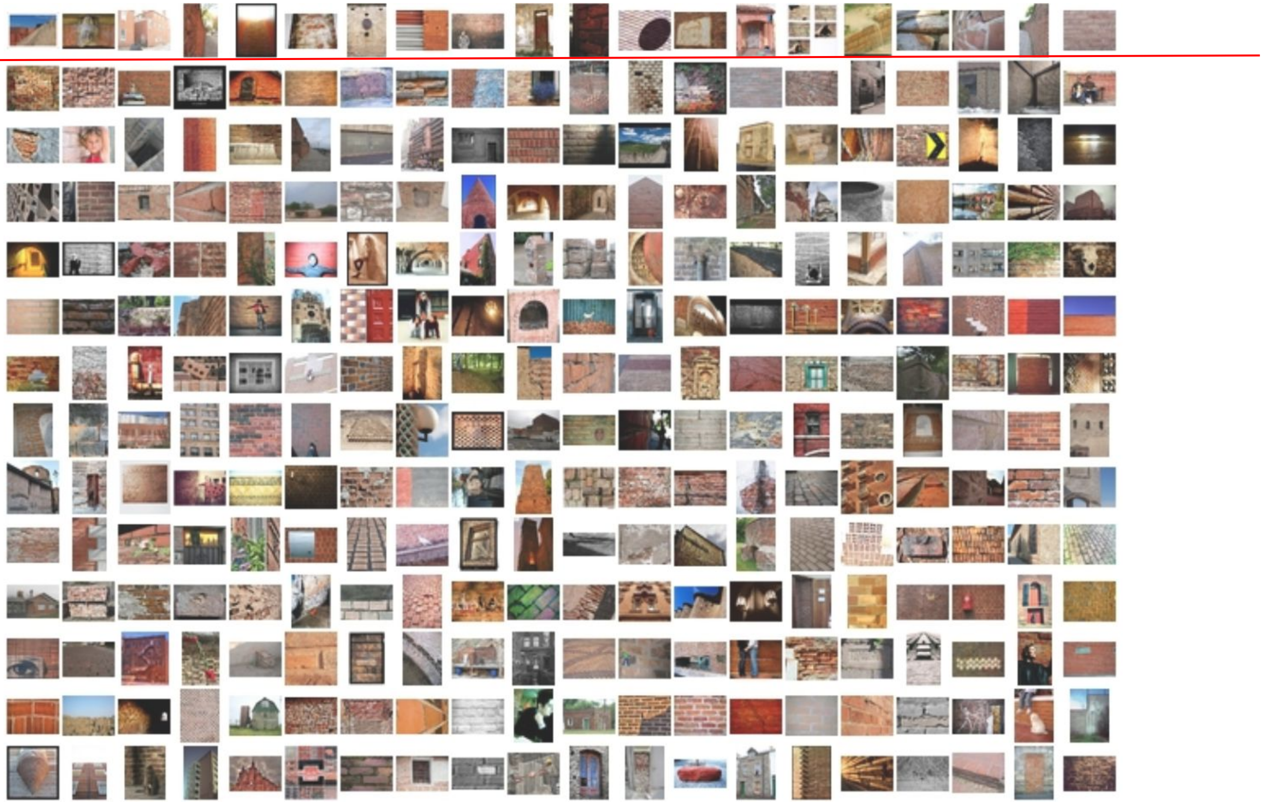
Supervised Classification



Semi-Supervised Classification: More realistic dataset

Labelled

Unlabelled



Semi-Supervised Classification

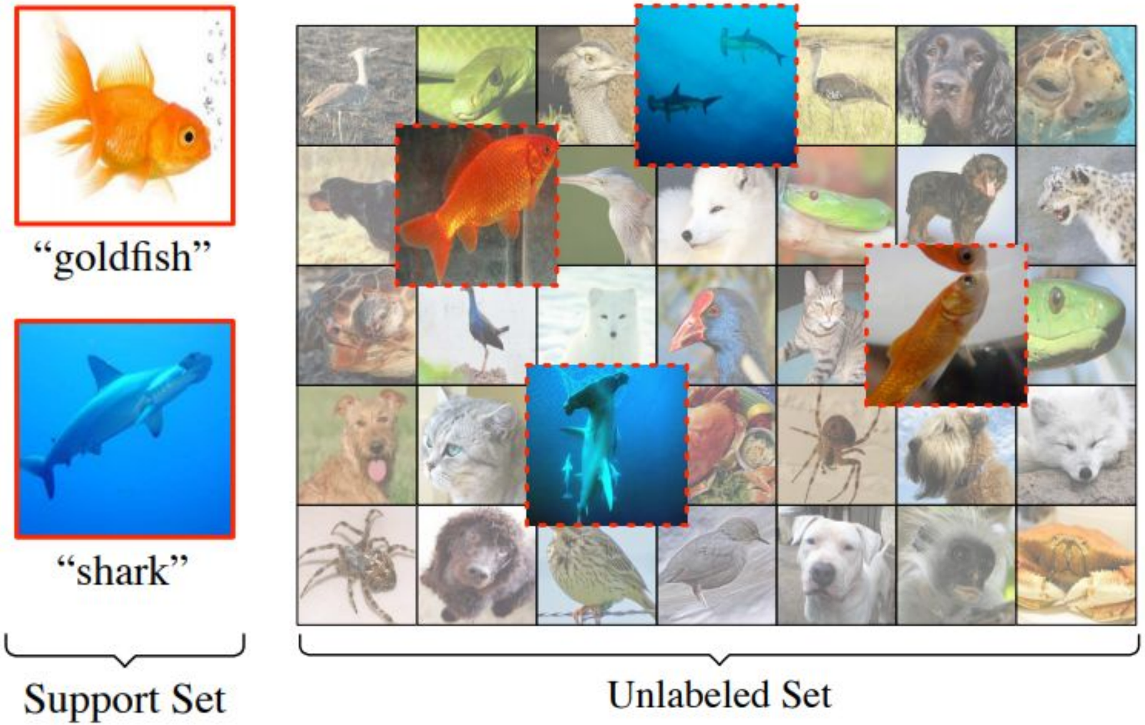
Most “biologically plausible” learning regime

A familiar problem:

?

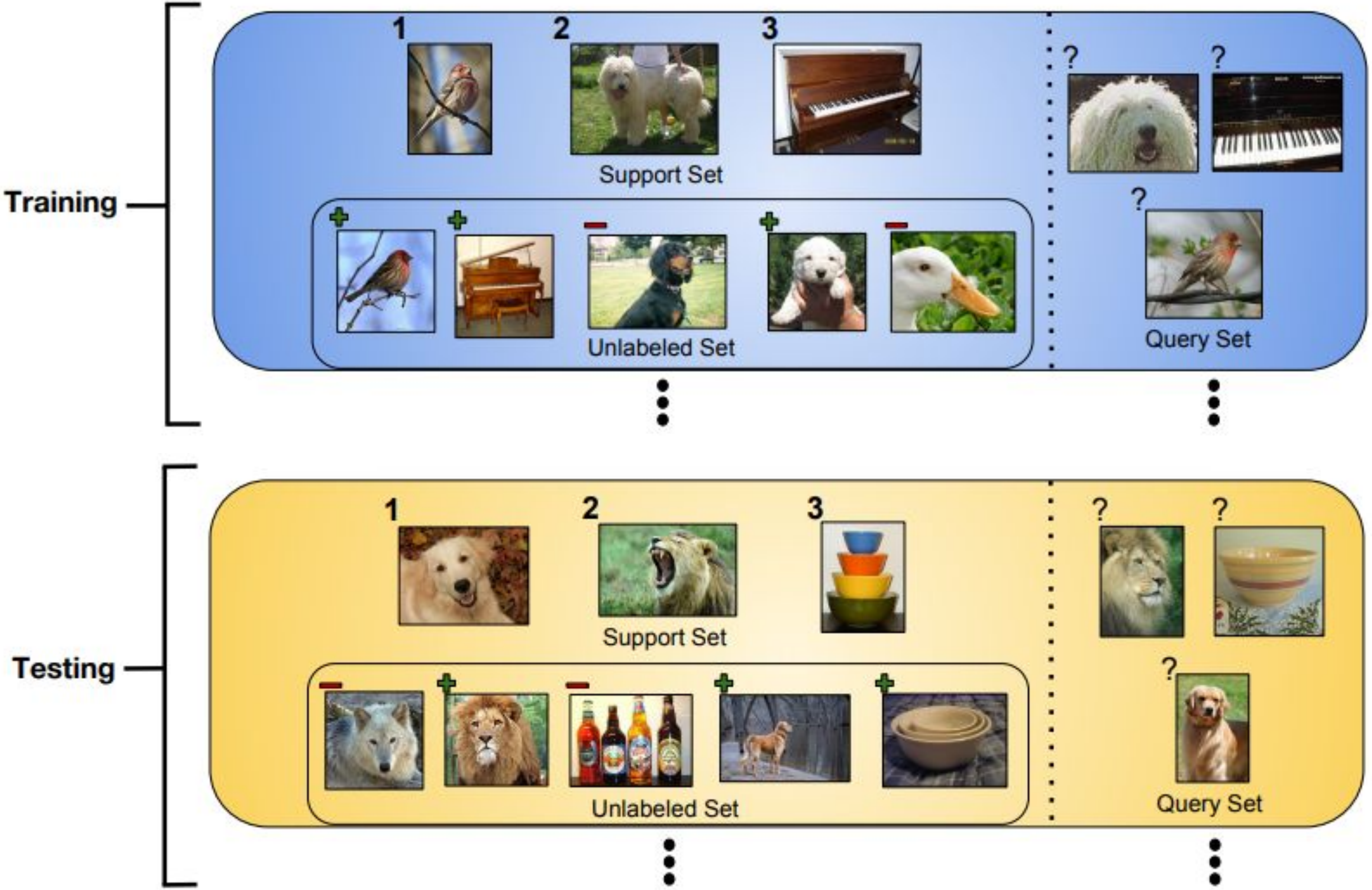
Few-shot, multi-task learning:
Generalize to unseen classes

A new twist on a familiar problem:



?

How can we leverage unlabelled data for few-shot classification?



Training
1
2
3
Support Set
Unlabeled Set
?
Query Set
Testing
1
2
3
Support Set
Unlabeled Set
?
Query Set

Unlabelled data may come from the support set or not (distractors)

Strategy:

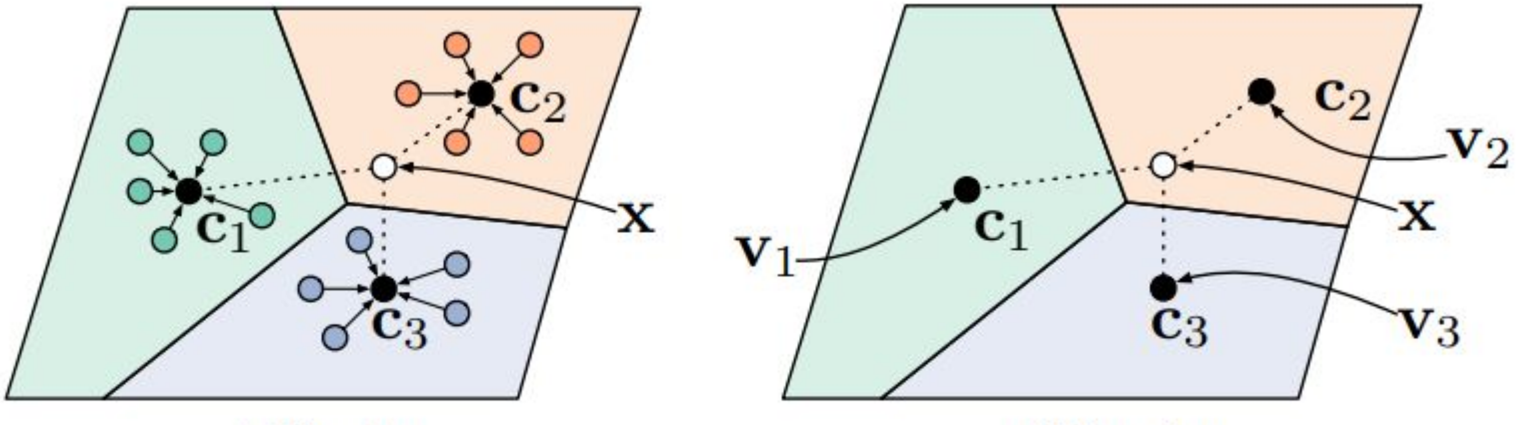
As we can now appreciate, there are a number of possible ways to approach the original problem. To name a few:

- Siamese Networks (Koch et al, 2015)
- Matching Networks (Vinyals et al., 2016)
- Prototypical Networks (Snell et al., 2017)
- Weight initialization / Update step learning (Ravi et al., 2017, Finn et al., 2017)
- MANN (Santoro et al., 2016)
- Temporal convolutions (Mishra et al., 2017)

All are reasonable starting points for semi-supervised few-shot classification problem!

Prototypical Networks (Snell et al., 2017)

Very simple inductive bias!

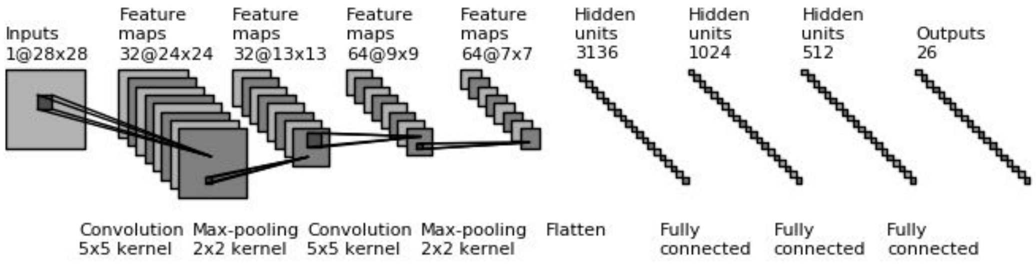


(a) Few-shot (b) Zero-shot

Prototypical Networks (Snell et al., 2017)

$$\mathbf{c}_k = \frac{1}{|S_k|} \sum_{(\mathbf{x}_i, y_i) \in S_k} f_\phi(\mathbf{x}_i)$$

For each class, compute **prototype**



Embedding is generated via a simple convnet:
Pixels - 64 [3x3] Filters - Batchnorm - ReLU - [2x2] MaxPool = **64D Vector**

https://jasonyzhang.com/convnet/

Prototypical Networks (Snell et al., 2017)

$$\mathbf{c}_k = \frac{1}{|S_k|} \sum_{(\mathbf{x}_i, y_i) \in S_k} f_{\boldsymbol{\phi}}(\mathbf{x}_i)$$

For each class, compute **prototype**

$$p_{\boldsymbol{\phi}}(y = k \mid \mathbf{x}) = \frac{\exp(-d(f_{\boldsymbol{\phi}}(\mathbf{x}), \mathbf{c}_k))}{\sum_{k'} \exp(-d(f_{\boldsymbol{\phi}}(\mathbf{x}), \mathbf{c}_{k'}))}$$

Softmax distribution of distances to **prototypes** for new image

$$J(\boldsymbol{\phi}) = -\log p_{\boldsymbol{\phi}}(y = k \mid \mathbf{x})$$

Compute loss

Prototypical Networks (Snell et al., 2017)

$$\mathbf{c}_k = \frac{1}{|S_k|} \sum_{(\mathbf{x}_i, y_i) \in S_k} f_\phi(\mathbf{x}_i)$$

For each class, compute **prototype**

$$p_\phi(y = k \,|\, \mathbf{x}) = \frac{\exp(-d(f_\phi(\mathbf{x}), \mathbf{c}_k))}{\sum_{k'} \exp(-d(f_\phi(\mathbf{x}), \mathbf{c}_{k'}))}$$

Softmax distribution of distances to **prototypes** for new image

$$J(\phi) = -\log p_\phi(y = k \,|\, \mathbf{x})$$

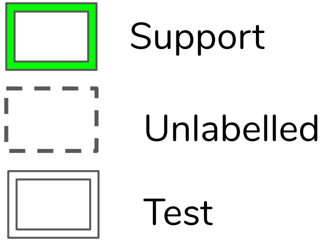
Compute loss

Very simple inductive bias:

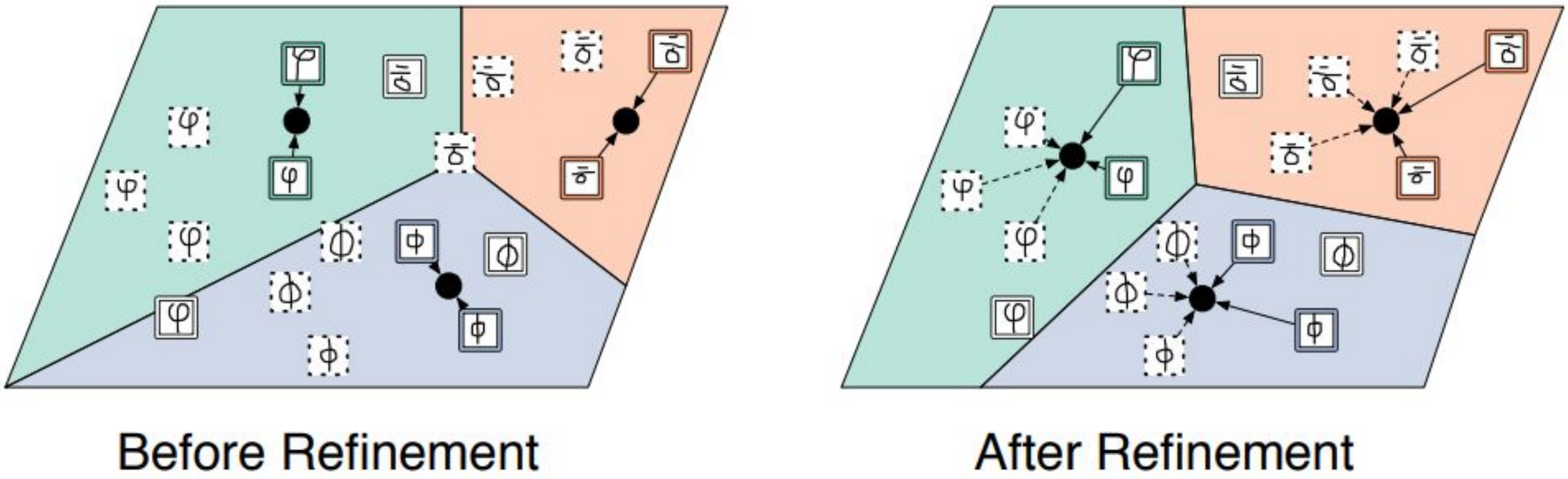
Reduces to a linear model with Euclidean distance $d(\mathbf{z}, \mathbf{z}') = \|\mathbf{z} - \mathbf{z}'\|^2$

$$\mathbf{w}_k^\top f_\phi(\mathbf{x}) + b_k, \text{ where } \mathbf{w}_k = 2\mathbf{c}_k \text{ and } b_k = -\mathbf{c}_k^\top \mathbf{c}_k$$

Strategy for semi-supervised:



Refine *Prototypes centers* with unlabelled data.



Strategy for semi-supervised:

1. Start with labelled prototypes
2. Give each unlabelled input a **partial assignment** to each cluster
3. Incorporate unlabelled examples into original prototype

Prototypical networks with Soft *k*-means

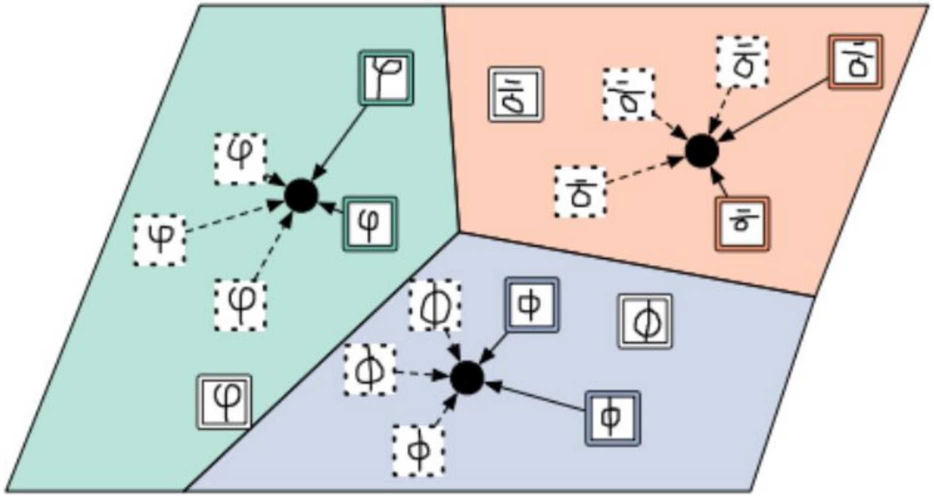
$$\mathcal{R} = \{\tilde{\boldsymbol{x}}_1, \tilde{\boldsymbol{x}}_2, \ldots, \tilde{\boldsymbol{x}}_M\}$$

Unlabelled support set

$$\tilde{\boldsymbol{p}}_c = \frac{\sum_i h(\boldsymbol{x}_i) z_{i,c} + \sum_j h(\tilde{\boldsymbol{x}}_j)\tilde{z}_{j,c}}{\sum_i z_{i,c} + \sum_j \tilde{z}_{j,c}}, \quad \text{where} \quad \tilde{z}_{j,c} = \frac{\exp\left(-||h(\tilde{\boldsymbol{x}}_j) - \boldsymbol{p}_c||_2^2\right)}{\sum_{c'} \exp\left(-||h(\tilde{\boldsymbol{x}}_j) - \boldsymbol{p}_{c'}||_2^2\right)}$$

Partial Assignment

Prototypical networks with Soft *k*-means



After Refinement

What about distractor classes?

Prototypical networks with Soft *k*-means w/ Distractor Cluster

Add a ***buffering prototype*** at the origin to “capture the distractors”

$$\boldsymbol{p}_c = \begin{cases} \frac{\sum_i h(\boldsymbol{x}_i) z_{i,c}}{\sum_i z_{i,c}} & \text{for } c = 1...N \\ \mathbf{0} & \text{for } c = N+1 \end{cases}$$

Prototypical networks with Soft *k*-means w/ Distractor Cluster

Add a ***buffering prototype*** at the origin to “capture the distractors”

$$\boldsymbol{p}_c = \begin{cases} \frac{\sum_i h(\boldsymbol{x}_i) z_{i,c}}{\sum_i z_{i,c}} & \text{for } c = 1...N \\ \boldsymbol{0} & \text{for } c = N+1 \end{cases}$$

Assumption: Distractors all come from one class!

Soft k-means + Masking Network

1. Distance

$$\tilde{d}_{j,c} = \frac{d_{j,c}}{\frac{1}{M}\sum_j d_{j,c}}, \text{ where } d_{j,c} = ||h(\tilde{\boldsymbol{x}}_j) - \boldsymbol{p}_c||_2^2$$

2. Compute mask with small network

$$[\beta_c, \gamma_c] = \text{MLP}\left(\left[\min_j(\tilde{d}_{j,c}), \max_j(\tilde{d}_{j,c}), \operatorname{var}_j(\tilde{d}_{j,c}), \operatorname{skew}_j(\tilde{d}_{j,c}), \operatorname{kurt}_j(\tilde{d}_{j,c})\right]\right)$$

$$\tilde{\boldsymbol{p}}_c = \frac{\sum_i h(\boldsymbol{x}_i) z_{i,c} + \sum_j h(\tilde{\boldsymbol{x}}_j)\tilde{z}_{j,c} m_{j,c}}{\sum_i z_{i,c} + \sum_j \tilde{z}_{j,c} m_{j,c}}, \text{ where } m_{j,c} = \sigma\left(-\gamma_c\left(\tilde{d}_{j,c} - \beta_c\right)\right)$$

Soft k-means + Masking Network

$$\tilde{d}_{j,c} = \frac{d_{j,c}}{\frac{1}{M}\sum_j d_{j,c}}, \text{ where } d_{j,c} = ||h(\tilde{\boldsymbol{x}}_j) - \boldsymbol{p}_c||_2^2$$

$$[\beta_c, \gamma_c] = \text{MLP}\left(\left[\min_j(\tilde{d}_{j,c}), \max_j(\tilde{d}_{j,c}), \text{var}_j(\tilde{d}_{j,c}), \text{skew}_j(\tilde{d}_{j,c}), \text{kurt}_j(\tilde{d}_{j,c})\right]\right)$$

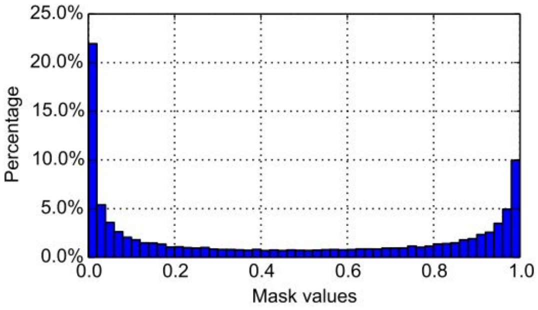
differentiable

$$\tilde{\boldsymbol{p}}_c = \frac{\sum_i h(\boldsymbol{x}_i) z_{i,c} + \sum_j h(\tilde{\boldsymbol{x}}_j)\tilde{z}_{j,c} m_{j,c}}{\sum_i z_{i,c} + \sum_j \tilde{z}_{j,c} m_{j,c}}, \text{ where } m_{j,c} = \sigma\left(-\gamma_c\left(\tilde{d}_{j,c} - \beta_c\right)\right)$$

Soft k-means + Masking

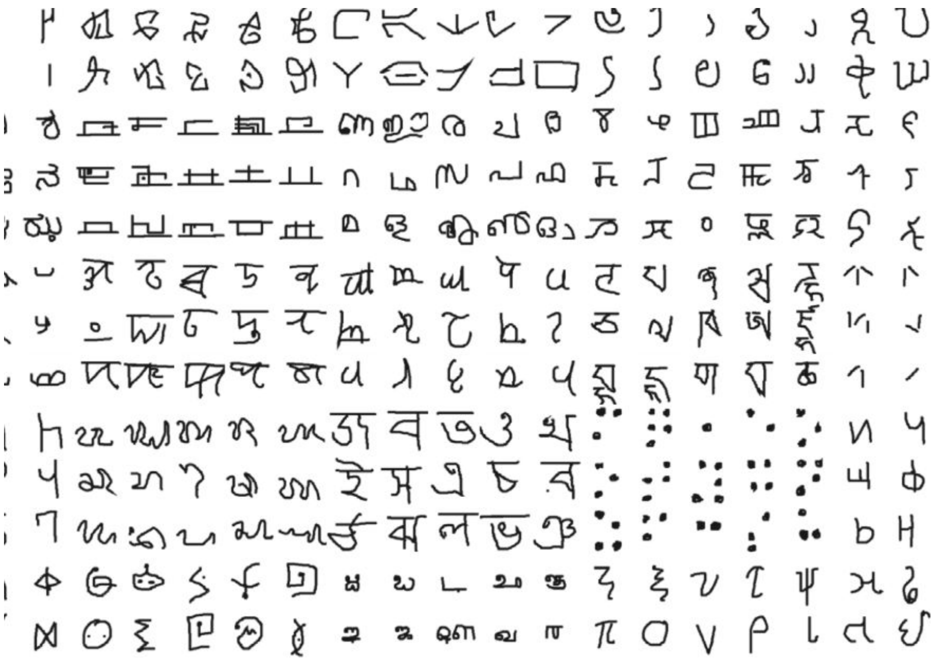
$$[\beta_c, \gamma_c] = \mathrm{MLP}\left(\left[\min_j(\tilde{d}_{j,c}), \max_j(\tilde{d}_{j,c}), \mathrm{var}_j(\tilde{d}_{j,c}), \mathrm{skew}_j(\tilde{d}_{j,c}), \mathrm{kurt}_j(\tilde{d}_{j,c})\right]\right)$$

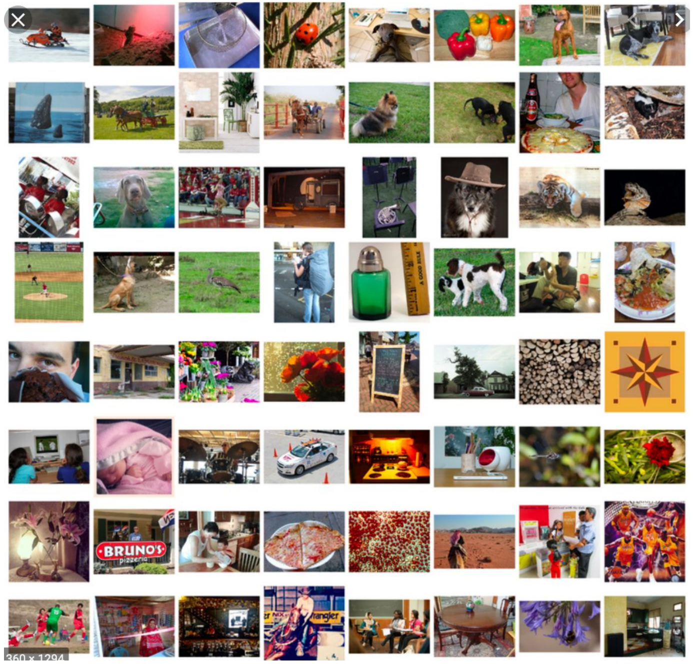
In practice, MLP is a dense layer with 20 hidden units (tanh nonlinearity)



Datasets

- Omniglot
- *mini*ImageNet (600 images from 100 classes)





Hierarchical Datasets

Omniglot

Train Alphabets: Alphabet_of_the_Magi, Angelic, Anglo-Saxon_Futhorc, Arcadian, Asomtavruli_(Georgian), Atemayar_Qelisayer, Atlantean, Aurek-Besh, Avesta, Balinese, Blackfoot_(Canadian_Aboriginal_Syllabics), Braille, Burmese_(Myanmar), Cyrillic, Futurama, Ge_ez, Glagolitic, Grantha, Greek, Gujarati, Gurmukhi (character 01-41), Inuktitut_(Canadian_Aboriginal_Syllabics), Japanese_(hiragana), Japanese_(katakana), Korean, Latin, Malay_(Jawi_-_Arabic), N_Ko, Ojibwe_(Canadian_Aboriginal_Syllabics), Sanskrit, Syriac_(Estrangelo), Tagalog, Tifinagh

Validation Alphabets: Armenian, Bengali, Early_Aramaic, Hebrew, Mkhedruli_(Geogian)

Test Alphabets: Gurmukhi (character 42-45), Kannada, Keble, Malayalam, Manipuri, Mongolian, Old_Church_Slavonic_(Cyrillic), Oriya, Sylheti, Syriac_(Serto), Tengwar, Tibetan, ULOG

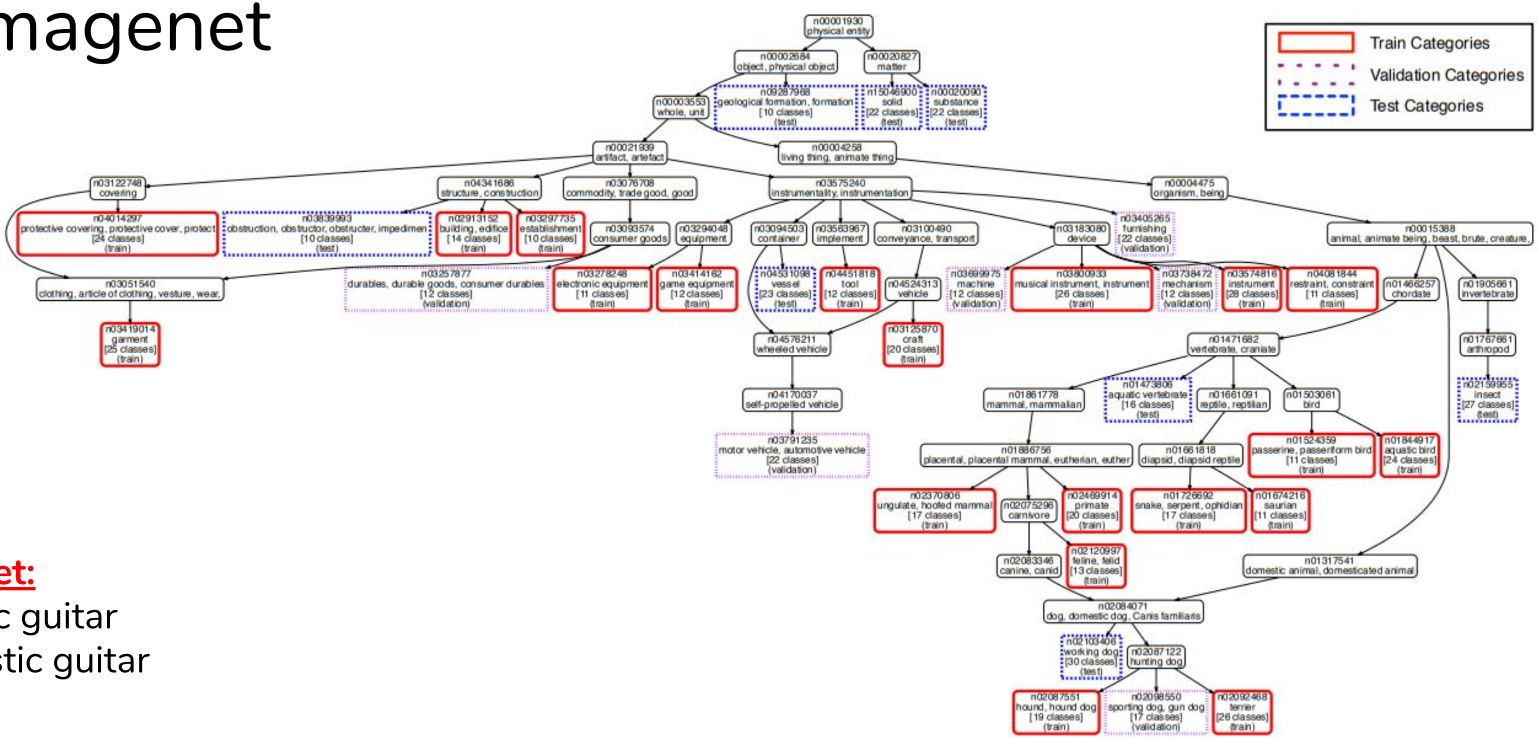
*tiered*ImageNet

Train Categories: n02087551 (hound, hound dog), n02092468 (terrier), n02120997 (feline, felid), n02370806 (ungulate, hoofed mammal), n02469914 (primate), n01726692 (snake, serpent, ophidian), n01674216 (saurian), n01524359 (passerine, passeriform bird), n01844917 (aquatic bird), n04081844 (restraint, constraint), n03574816 (instrument), n03800933 (musical instrument, instrument), n03125870 (craft), n04451818 (tool), n03414162 (game equipment), n03278248 (electronic equipment), n03419014 (garment), n03297735 (establishment), n02913152 (building, edifice), n04014297 (protective covering, protective cover, protection).

Validation Categories: n02098550 (sporting dog, gun dog), n03257877 (durables, durable goods, consumer durables), n03405265 (furnishing), n03699975 (machine), n03738472 (mechanism), n03791235 (motor vehicle, automotive vehicle),

Test Categories: n02103406 (working dog), n01473806 (aquatic vertebrate), n02159955 (insect), n04531098 (vessel), n03839993 (obstruction, obstructor, obstructer, impediment, impedimenta), n09287968 (geological formation, formation), n00020090 (substance), n15046900 (solid).

tieredImagenet



miniImageNet:
Test - electric guitar
Train - acoustic guitar

tierediImageNet:
Test - musical instruments
Train - farming equipment

Datasets

- Omniglot
- *mini*ImageNet (600 images from 100 classes)
- *tiered*ImageNet (34 broad categories, each containing 10 to 30 classes)

10% goes to labeled splits
90% goes to unlabelled classes and distractors*

*40/60 for miniImageNet

Datasets

- Omniglot
- *mini*ImageNet (600 images from 100 classes)
- *tiered*ImageNet (34 broad categories, each containing 10 to 30 classes)

10% goes to labeled splits

Much less labelled data than standard few-shot approaches!!!

90% goes to unlabelled classes and distractors*

*40/60 for miniImageNet

Datasets

N: Classes
K: Labelled samples from each class
M: Unlabelled samples from N classes
H: Distractors (Unlabelled sample from classes other than N)

H=**N**=**5**
M=5 for training & **M=20** for testing

Baseline Models

	Models	Acc.	Acc. w/ D
1.	Supervised	94.62 ± 0.09	94.62 ± 0.09
	Semi-Supervised Inference	97.45 ± 0.05	95.08 ± 0.09

1. Vanilla Protonet

Baseline Models

	Models	Acc.	Acc. w/ D
1.	Supervised	94.62 ± 0.09	94.62 ± 0.09
2.	Semi-Supervised Inference	97.45 ± 0.05	95.08 ± 0.09

1. Vanilla Protonet
2. Vanilla Protonet + one step of Soft k-means refinement at **test only** (supervised embedding)

Results: Omniglot

Models	Acc.	Acc. w/ D
Supervised	94.62 ± 0.09	94.62 ± 0.09
Semi-Supervised Inference	97.45 ± 0.05	95.08 ± 0.09
Soft k-Means	97.25 ± 0.10	95.01 ± 0.09
Soft k-Means+Cluster	$\mathbf{97.68 \pm 0.07}$	97.17 ± 0.04
Masked Soft k-Means	97.52 ± 0.07	$\mathbf{97.30 \pm 0.08}$

Results: miniImageNet

Models	1-shot Acc.	5-shot Acc.	1-shot Acc w/ D	5-shot Acc. w/ D
Supervised	43.61 ± 0.27	59.08 ± 0.22	43.61 ± 0.27	59.08 ± 0.22
Semi-Supervised Inference	48.98 ± 0.34	63.77 ± 0.20	47.42 ± 0.33	62.62 ± 0.24
Soft k-Means	$\mathbf{50.09 \pm 0.45}$	$\mathbf{64.59 \pm 0.28}$	$\mathbf{48.70 \pm 0.32}$	$\mathbf{63.55 \pm 0.28}$
Soft k-Means+Cluster	49.03 ± 0.24	63.08 ± 0.18	$\mathbf{48.86 \pm 0.32}$	61.27 ± 0.24
Masked Soft k-Means	$\mathbf{50.41 \pm 0.31}$	$\mathbf{64.39 \pm 0.24}$	$\mathbf{49.04 \pm 0.31}$	62.96 ± 0.14

Table 2: *mini*ImageNet 1/5-shot classification results.

Results: tieredImageNet

Models	1-shot Acc.	5-shot Acc.	1-shot Acc. w/ D	5-shot Acc. w/ D
Supervised	46.52 ± 0.52	66.15 ± 0.22	46.52 ± 0.52	66.15 ± 0.22
Semi-Supervised Inference	50.74 ± 0.75	69.37 ± 0.26	48.67 ± 0.60	67.46 ± 0.24
Soft k-Means	51.52 ± 0.36	$\mathbf{70.25 \pm 0.31}$	49.88 ± 0.52	68.32 ± 0.22
Soft k-Means+Cluster	$\mathbf{51.85 \pm 0.25}$	69.42 ± 0.17	$\mathbf{51.36 \pm 0.31}$	67.56 ± 0.10
Masked Soft k-Means	$\mathbf{52.39 \pm 0.44}$	$\mathbf{69.88 \pm 0.20}$	$\mathbf{51.38 \pm 0.38}$	$\mathbf{69.08 \pm 0.25}$

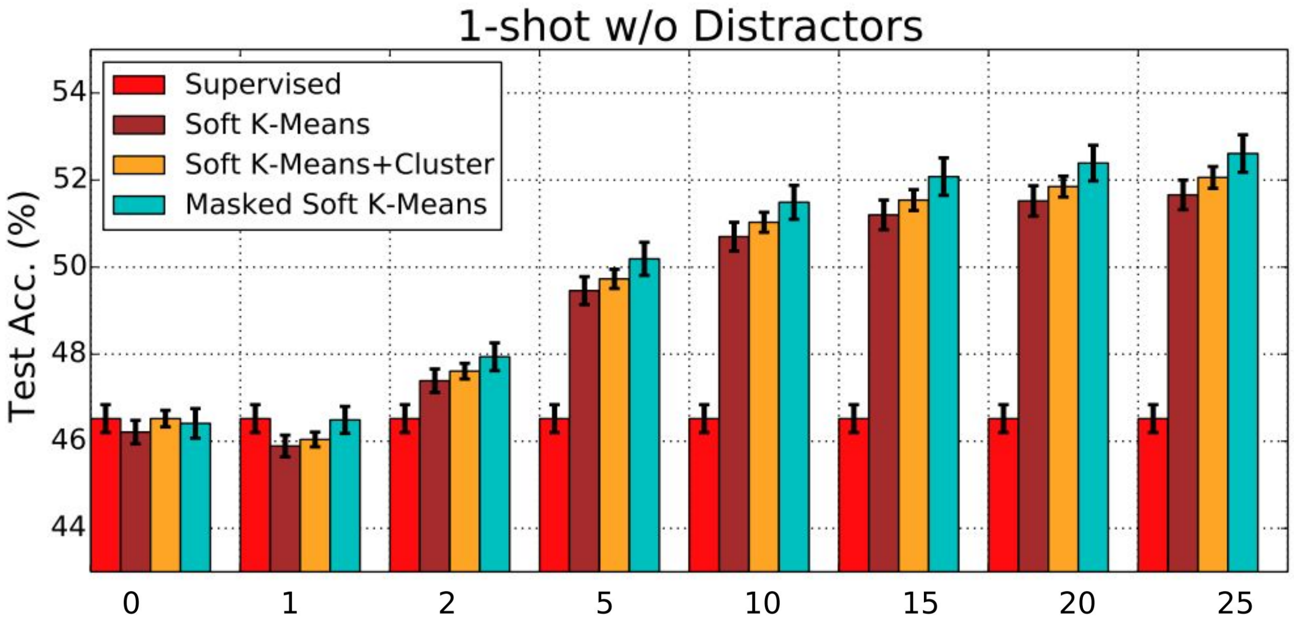
Table 3: *tiered*ImageNet 1/5-shot classification results.

Results: Other Baselines

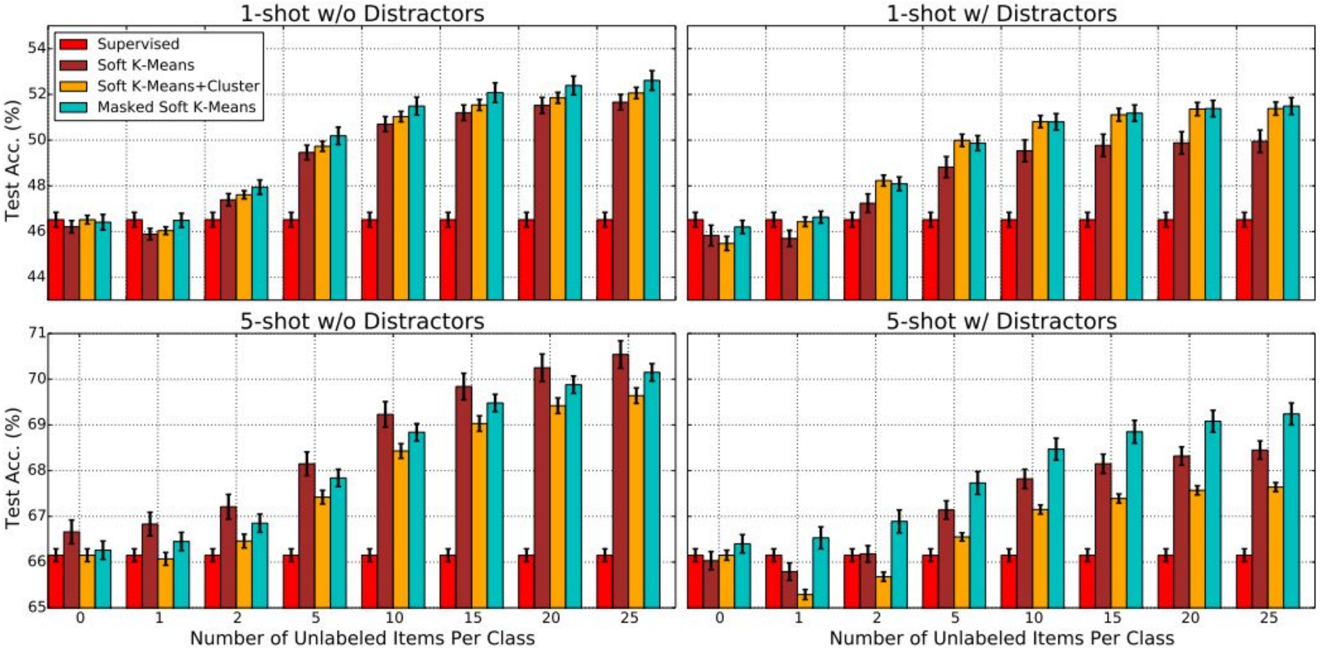
Models	Omniglot	*mini*ImageNet		*tiered*ImageNet	
	1-shot	1-shot	5-shot	1-shot	5-shot
1-NN Pixel	40.39 ± 0.36	26.74 ± 0.48	31.43 ± 0.51	26.55 ± 0.50	30.79 ± 0.53
1-NN CNN rnd	59.55 ± 0.46	24.03 ± 0.38	27.54 ± 0.42	25.49 ± 0.45	30.01 ± 0.47
1-NN CNN pre	52.53 ± 0.51	32.90 ± 0.58	40.79 ± 0.76	32.76 ± 0.66	40.26 ± 0.67
LR Pixel	49.15 ± 0.39	24.50 ± 0.41	33.33 ± 0.68	25.70 ± 0.46	36.30 ± 0.62
LR CNN rnd	57.80 ± 0.45	24.10 ± 0.50	28.40 ± 0.42	26.55 ± 0.48	32.51 ± 0.52
LR CNN pre	48.49 ± 0.47	30.28 ± 0.54	40.27 ± 0.59	34.52 ± 0.68	43.58 ± 0.72
ProtoNet	$\mathbf{94.62 \pm 0.09}$	$\mathbf{43.61 \pm 0.27}$	$\mathbf{59.08 \pm 0.22}$	$\mathbf{46.52 \pm 0.32}$	$\mathbf{66.15 \pm 0.34}$

Results

Models trained with M=5
During meta-test: vary amount of unlabelled examples



Results



1-shot w/o Distractors
1-shot w/ Distractors
5-shot w/o Distractors
5-shot w/ Distractors
Supervised
Soft K-Means
Soft K-Means+Cluster
Masked Soft K-Means
Test Acc. (%)
Number of Unlabeled Items Per Class
0
1
2
5
10
15
20
25

Conclusions:

1. Achieve state of the art performance over logical baselines on 3 datasets

Conclusions:

1. Achieve state of the art performance over logical baselines on 3 datasets
2. K-means Masked models perform best with distractors

Conclusions:

1. Achieve state of the art performance over logical baselines on 3 datasets
2. K-means Masked models perform best with distractors
3. Novel: models extrapolate to increases in amount of labelled data

Conclusions:

1. Achieve state of the art performance over logical baselines on 3 datasets
2. K-means Masked models perform best with distractors
3. Novel: models extrapolate to increases in amount of labelled data
4. New dataset: *tiered*ImageNet

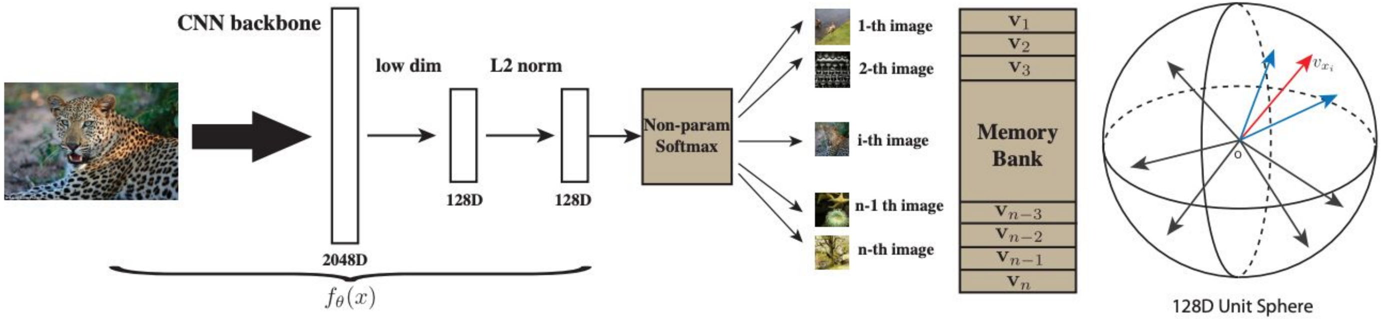
Critiques:

1. Results are convincing, but the work is actually a relatively straightforward application of (a) **Protonets** and (b) **k-means clustering**
2. **Model Choice:** protonets are very simple. It’s not clear what they gained by the simple inductive bias
3. Presented approach does not generalize well beyond classification problems

Future directions: extension to unsupervised learning

I would be really interested in withholding labels alltogether
Can the model learn how many classes there are?

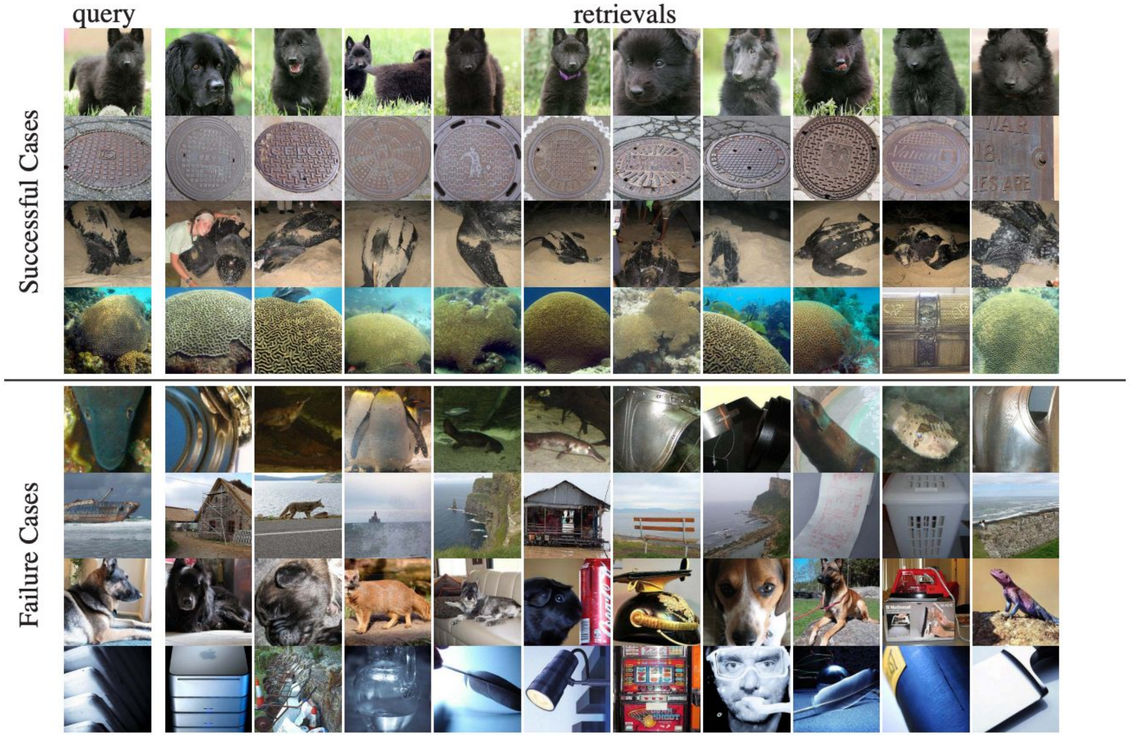
... and correctly classify them?



Unsupervised Feature Learning via Non-Parametric Instance Discrimination

Zhirong Wu[*†] Yuanjun Xiong[†‡] Stella X. Yu[*] Dahua Lin[†]
[*]UC Berkeley / ICSI [†]Chinese University of Hong Kong [‡]Amazon Rekognition

Future directions: extension to unsupervised learning



Unsupervised Feature Learning via Non-Parametric Instance Discrimination

Zhirong Wu*† Yuanjun Xiong†‡ Stella X. Yu* Dahua Lin†
*UC Berkeley / ICSI †Chinese University of Hong Kong ‡Amazon Rekognition

Thank you!

Supplemental:
Accounting for Intra-Cluster Distance

$$\tilde{z}_{j,c} = \frac{\exp\left(-\frac{1}{r_c^2}||\tilde{\boldsymbol{x}}_j - \boldsymbol{p}_c||_2^2 - A(r_c)\right)}{\sum_{c'} \exp\left(-\frac{1}{r_c^2}||\tilde{\boldsymbol{x}}_j - \boldsymbol{p}_{c'}||_2^2 - A(r_{c'})\right)}, \quad \text{where} \quad A(r) = \frac{1}{2}\log(2\pi) + \log(r) \tag{6}$$