Laplacian Regularized Few Shot Learning (LaplacianShot)

Imtiaz Masud Ziko, Jose Dolz, Eric Granger and Ismail Ben Ayed




ETS Montreal

Overview

Few-Shot Learning

- *What and Why ?*
- *Brief discussion on existing approaches.*

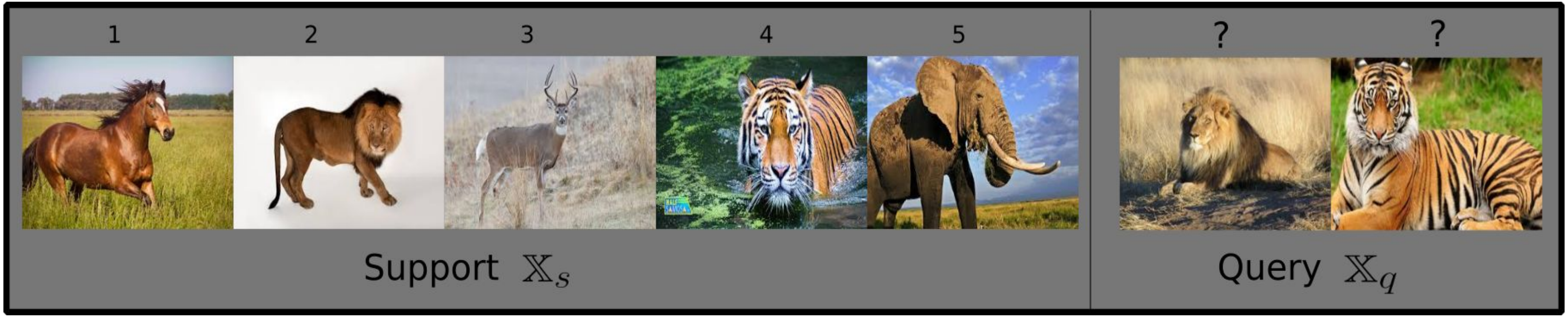
Proposed **LaplacianShot**

- *The context*
- *Proposed formulation*
- *Optimization*
- *Proposed Algorithm*

Experiments

- *Experimental Setup*
- *SOTA results on 5 different few-shot benchmarks.*

Few-Shot Learning (An example)



Few-Shot Learning (An example)

- Given *C* = ***5*** classes
- Each class *c* having **1** examples.

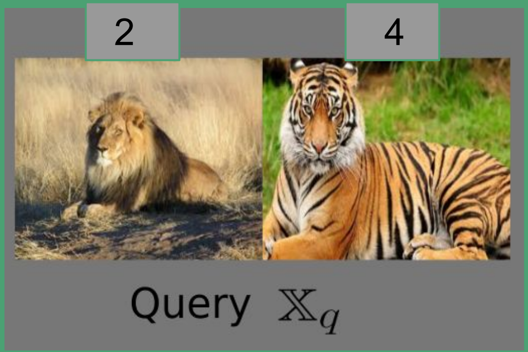
(5-way 1-shot)

From these

Learn a Model

To classify this

Few-Shot Learning (An example)



- Given C = **5** classes
- Each class c having **1** examples.

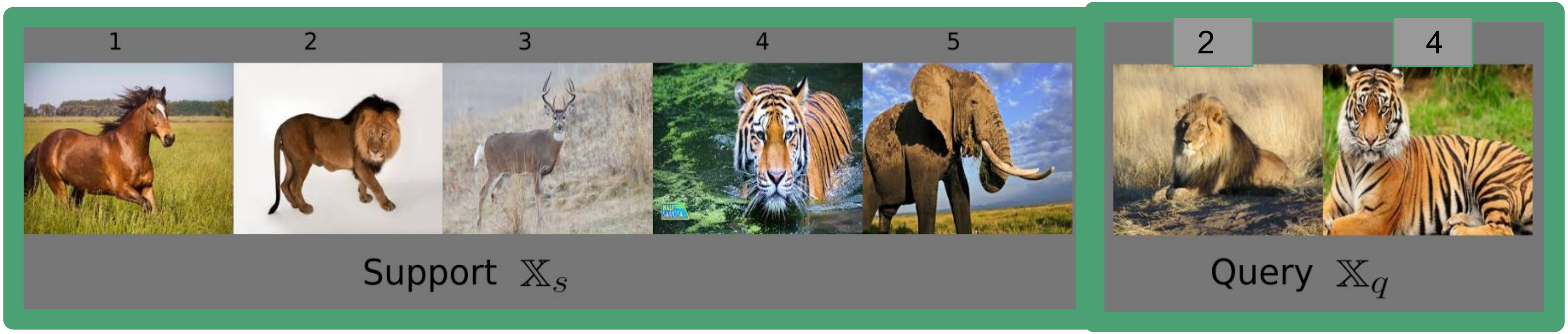
(5-way 1-shot)

From these

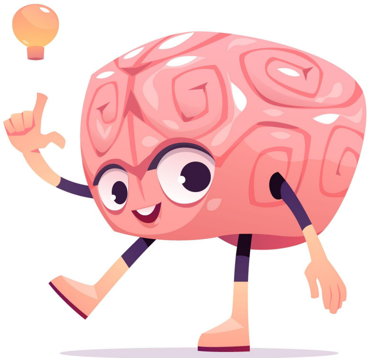
Learn a Model

To classify this

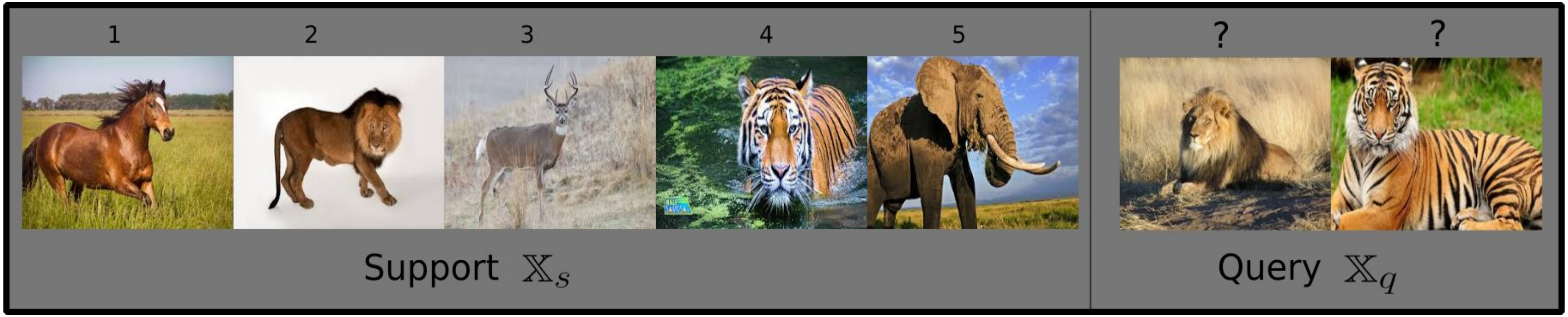
Few-Shot Learning



Humans recognize perfectly
with few examples



Few-Shot Learning



- ❏ Modern ML methods **generalize poorly**

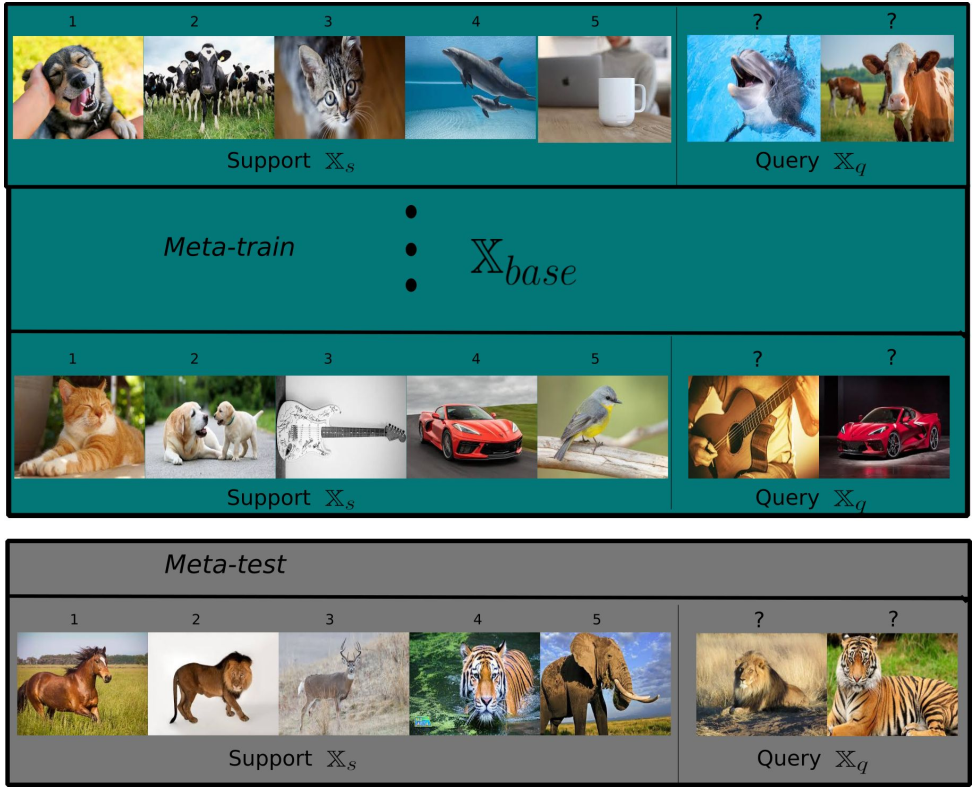
- ❏ Need a better way.

Few-shot learning

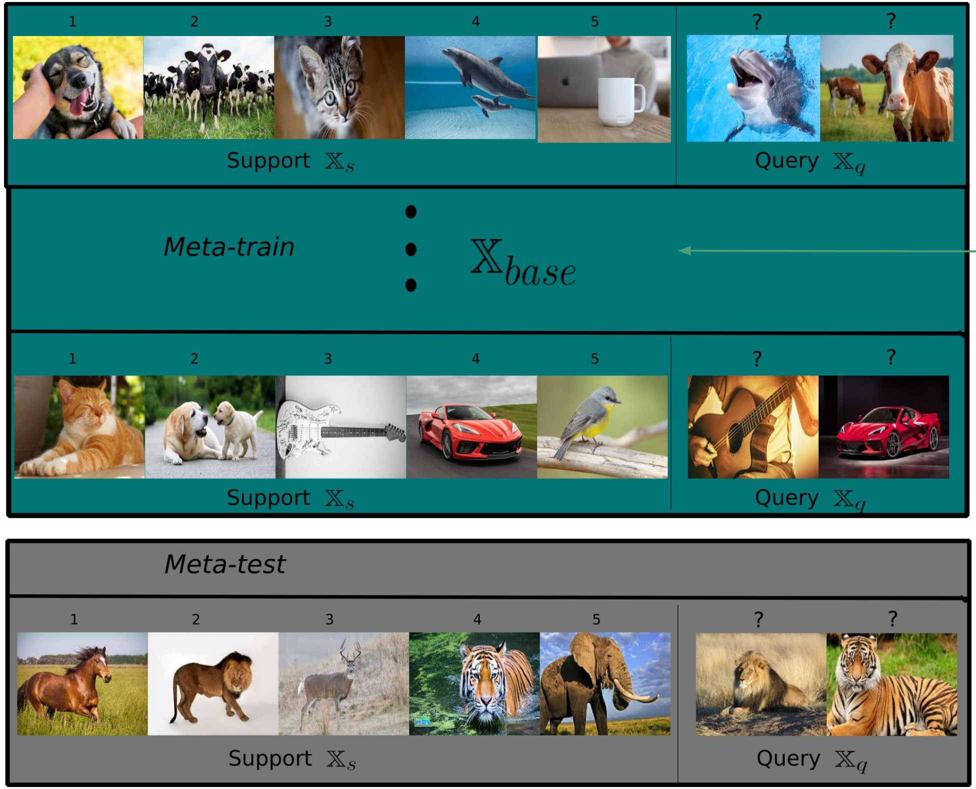
A very large body of recent works, mostly based on:

Meta-learning framework

Meta-Learning Framework



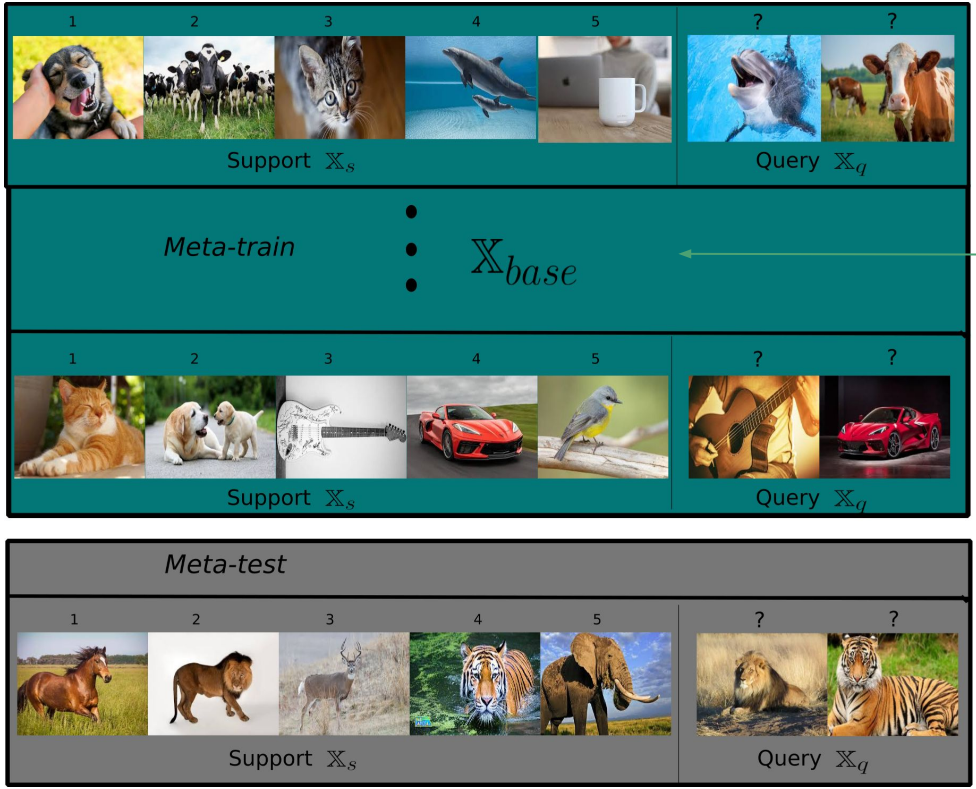
Meta-Learning Framework



Training set with enough labeled data

(base classes ***different from the*** test classes)

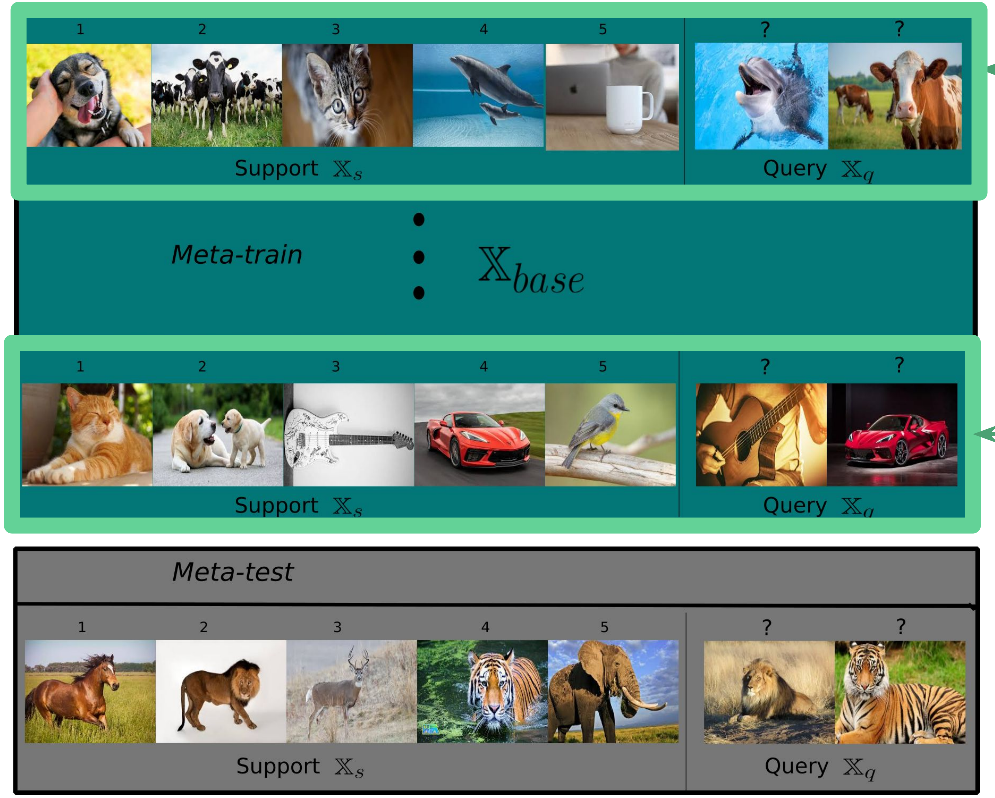
Meta-Learning Framework



Training set with enough labeled data **to learn initial model**

$$f_\theta$$

Meta-Learning Framework



Create episodes and do ***episodic training*** to learn ***meta-learner***

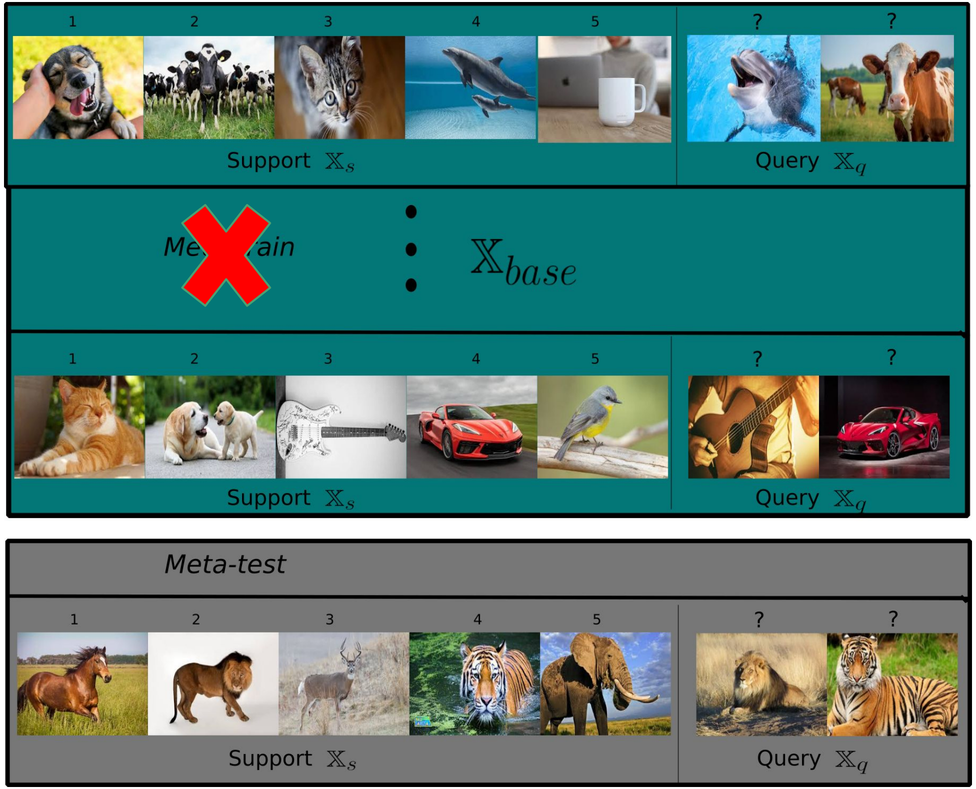
Vinyal et al, (Neurips '16),
Snell et al, (Neurips '17),
Sung et al, (CVPR ' 18),
Finn et al, (ICML' 17),
Ravi et al, (ICLR' 17),
Lee et al, (CVPR' 19),
Hu et al, (ICLR '20),
Ye et al, (CVPR '20), . . .

Taking a few steps backward . .

Recently [Chen et al., ICLR’19, Wang et al., ’19, Dhillon et al., ICLR’20] :

Simple baselines **outperform** the overly convoluted **meta-learning** based approaches.

Baseline Framework



No need to

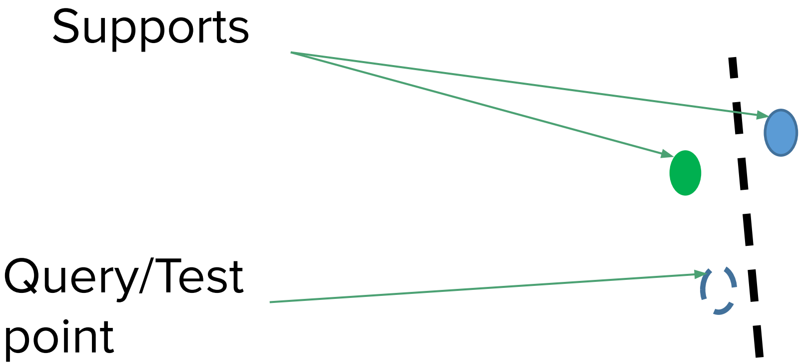
meta-train

Baseline Framework

Simple conventional cross-entropy training

The approaches mostly differ **during inference**

Inductive vs **Transductive** inference

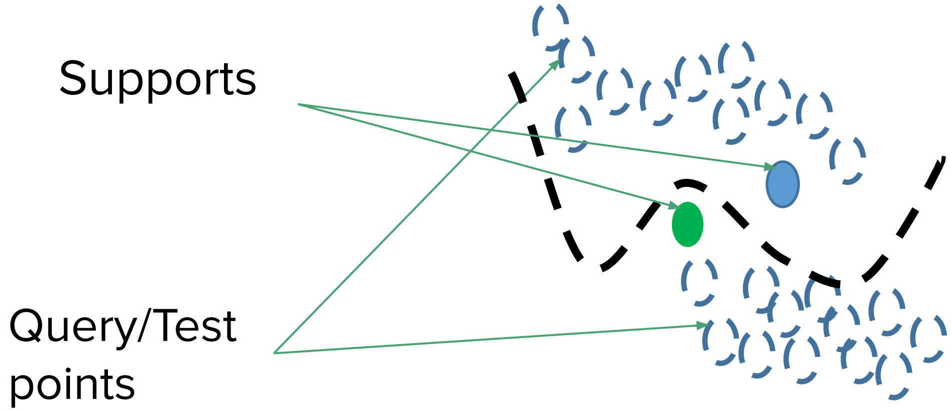


Examples

Vinayls et al., NEURIPS' 16
(Attention mechanism)

Snell et al., NEURIPS' 17
(Nearest Prototype)

Inductive vs **Transductive** inference



Transductive: Predict for all test points, instead of one at a time

Examples

Liu et. al., ICLR'19 (Label propagation)

Dhillon, ICLR'20 (Transductive fine-tuning)

Proposed LaplacianShot

Laplacian-regularized objective:

$$\boxed{\mathcal{E}(\mathbf{Y}) = \mathcal{N}(\mathbf{Y}) + \frac{\lambda}{2}\mathcal{L}(\mathbf{Y})}$$

- Latent Assignment matrix for N query samples:

$$\mathbf{Y} = [\mathbf{y}_q] \in \{0,1\}^{N\times C}$$

- Label assignment for each query:

$$\mathbf{y}_q = [y_{q,1}, \ldots, y_{q,C}]^t \in \{0,1\}^C$$

- And Simplex Constraints:

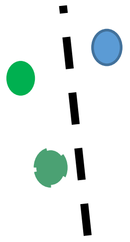
$$\mathbf{1}^t\mathbf{y}_q = 1$$

Proposed LaplacianShot

Laplacian-regularized objective:

$$\mathcal{E}(\mathbf{Y}) = \boxed{\mathcal{N}(\mathbf{Y})} + \frac{\lambda}{2}\boxed{\mathcal{L}(\mathbf{Y})}$$

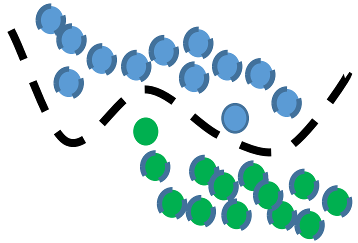
Nearest Prototype classification



When $\lambda = 0$

Similar to ProtoNet (Snell ’17) or SimpleShot (Wang ’19)

Laplacian Regularization



Well known in Graph Laplacian:
Spectral clustering (Shi I‘00, Von ‘07) , SLK (Ziko ’18)
SSL (Weston ‘12, Belkin ‘06)

LaplacianShot Takeaways

- ✓ **SOTA results** without bell and whistles.
- ✓ Simple **constrained graph clustering** works very well.
- ✓ **No** network **fine-tuning**, neither **meta-learning**
- ✓ **Model Agnostic**
- ✓ **Fast** transductive inference: almost inductive time

LapLacianShot

More Details

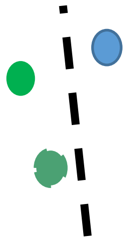
Proposed LaplacianShot

Laplacian-regularized objective:

$$\mathcal{E}(\mathbf{Y}) = \boxed{\mathcal{N}(\mathbf{Y})} + \frac{\lambda}{2}\mathcal{L}(\mathbf{Y})$$

$$\mathcal{N}(\mathbf{Y}) = \sum_{q=1}^{N}\sum_{c=1}^{C} y_{q,c}\, d(\mathbf{x}_q - \mathbf{m}_c)$$

Nearest Prototype classification



When $\lambda = 0$

Labeling according to nearest support prototypes

- Feature embedding: $\mathbf{x}_q = f_\theta(x_q)$
- Prototype $\mathbf{m}_c$ can be :
    - **The support example** in 1-shot or
    - **Simple mean** from support examples or
    - **Weighted mean** from both support and initially predicted query samples

Proposed LaplacianShot

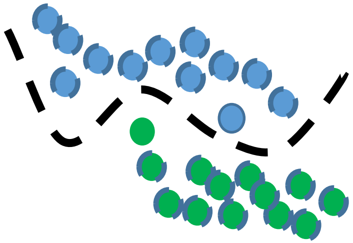
Pairwise similarity

Laplacian-regularized objective:

$$\mathcal{E}(\mathbf{Y}) = \mathcal{N}(\mathbf{Y}) + \frac{\lambda}{2}\boxed{\mathcal{L}(\mathbf{Y})}$$

$$\boxed{\mathcal{L}(\mathbf{Y}) = \sum_{q,p} w(\mathbf{x}_q, \mathbf{x}_p)\|\mathbf{y}_q - \mathbf{y}_p\|^2}$$

Laplacian Regularization



Well known in Graph Laplacian:

Encourages nearby points to have similar assignments

Proposed Optimization

Laplacian-regularized objective:

$$\mathcal{E}(\mathbf{Y}) = \mathcal{N}(\mathbf{Y}) + \frac{\lambda}{2}\mathcal{L}(\mathbf{Y})$$

$$\boxed{\mathcal{N}(\mathbf{Y}) = \sum_{q=1}^{N}\sum_{c=1}^{C} y_{q,c} d(\mathbf{x}_q - \mathbf{m}_c)}$$

$$\boxed{\mathcal{L}(\mathbf{Y}) = \sum_{q,p} w(\mathbf{x}_q, \mathbf{x}_p)\|\mathbf{y}_q - \mathbf{y}_p\|^2}$$

Tricky to optimize due to:

Proposed Optimization

Laplacian-regularized objective:

$$\mathcal{E}(\mathbf{Y}) = \mathcal{N}(\mathbf{Y}) + \frac{\lambda}{2}\mathcal{L}(\mathbf{Y})$$

$$\boxed{\mathcal{N}(\mathbf{Y}) = \sum_{q=1}^{N}\sum_{c=1}^{C} y_{q,c} d(\mathbf{x}_q - \mathbf{m}_c)}$$

$$\boxed{\mathcal{L}(\mathbf{Y}) = \sum_{q,p} w(\mathbf{x}_q, \mathbf{x}_p)\|\mathbf{y}_q - \mathbf{y}_p\|^2}$$

Tricky to optimize due to:

❌ Simplex/Integer Constraints.

$$\mathbf{Y} = [\mathbf{y}_q] \in \{0, 1\}^{N \times C}$$

$$\mathbf{1}^t \mathbf{y}_q = 1$$

Proposed Optimization

Laplacian-regularized objective:

$$\mathcal{E}(\mathbf{Y}) = \mathcal{N}(\mathbf{Y}) + \frac{\lambda}{2}\mathcal{L}(\mathbf{Y})$$

$$\boxed{\mathcal{N}(\mathbf{Y}) = \sum_{q=1}^{N}\sum_{c=1}^{C} y_{q,c} d(\mathbf{x}_q - \mathbf{m}_c)}$$

$$\boxed{\mathcal{L}(\mathbf{Y}) = \sum_{q,p} w(\mathbf{x}_q, \mathbf{x}_p)\|\mathbf{y}_q - \mathbf{y}_p\|^2}$$

Tricky to optimize due to:

❌ Laplacian over discrete variables.

Proposed Optimization

Laplacian-regularized objective:

$$\mathcal{E}(\mathbf{Y}) = \mathcal{N}(\mathbf{Y}) + \frac{\lambda}{2}\mathcal{L}(\mathbf{Y})$$

$$\boxed{\mathcal{N}(\mathbf{Y}) = \sum_{q=1}^{N}\sum_{c=1}^{C} y_{q,c}\, d(\mathbf{x}_q - \mathbf{m}_c)}$$

$$\boxed{\mathcal{L}(\mathbf{Y}) = \sum_{q,p} w(\mathbf{x}_q, \mathbf{x}_p)\|\mathbf{y}_q - \mathbf{y}_p\|^2}$$

Relax integer constraints:

➢ Convex quadratic problem

❌ Require solving for the N×C variables all together

❌ Extra projection steps for the simplex constraints

Proposed Optimization

Laplacian-regularized objective:

$$\mathcal{E}(\mathbf{Y}) = \mathcal{N}(\mathbf{Y}) + \frac{\lambda}{2}\mathcal{L}(\mathbf{Y})$$

$$\boxed{\mathcal{N}(\mathbf{Y}) = \sum_{q=1}^{N}\sum_{c=1}^{C} y_{q,c} d(\mathbf{x}_q - \mathbf{m}_c)}$$

$$\boxed{\mathcal{L}(\mathbf{Y}) = \sum_{q,p} w(\mathbf{x}_q, \mathbf{x}_p)\|\mathbf{y}_q - \mathbf{y}_p\|^2}$$

We do:

- ✓ Concave relaxation
- ✓ Independent and closed-form updates for each assignment variable
- ✓ Efficient bound optimization

Concave Laplacian

$$\boxed{\mathcal{L}(\mathbf{Y}) = \sum_{q,p} w(\mathbf{x}_q, \mathbf{x}_p)\|\mathbf{y}_q - \mathbf{y}_p\|^2}$$

Concave Laplacian

When $\mathbf{y}_q \in \{0,1\}^C$

$$\mathcal{L}(\mathbf{Y}) = \boxed{2\left(\sum_q \mathbf{y}_q^t \mathbf{y}_q D_q - \sum_{q,p} w(\mathbf{x}_q, \mathbf{x}_p) \mathbf{y}_q^t \mathbf{y}_p\right)}$$

$$= \boxed{2\left(\sum_q D_q - \sum_{q,p} w(\mathbf{x}_q, \mathbf{x}_p) \mathbf{y}_q^t \mathbf{y}_q\right)}$$

Equal

Concave Laplacian

When $\mathbf{y}_q \in [0,1]^C$

$$\mathcal{L}(\mathbf{Y}) = \boxed{2\left(\sum_q \mathbf{y}_q^t \mathbf{y}_q D_q - \sum_{q,p} w(\mathbf{x}_q, \mathbf{x}_p)\mathbf{y}_q^t \mathbf{y}_p\right)}$$

$$\neq \boxed{2\left(\sum_q D_q - \sum_{q,p} w(\mathbf{x}_q, \mathbf{x}_p)\mathbf{y}_q^t \mathbf{y}_q\right)}$$

Not Equal

Concave Laplacian

When $\mathbf{y}_q \in [0,1]^C$

$$\mathcal{L}(\mathbf{Y}) = \boxed{2\left(\sum_q \mathbf{y}_q^t \mathbf{y}_q D_q - \sum_{q,p} w(\mathbf{x}_q, \mathbf{x}_p)\mathbf{y}_q^t \mathbf{y}_p\right)}$$

$$\neq \boxed{2\left(\sum_q D_q - \sum_{q,p} w(\mathbf{x}_q, \mathbf{x}_p)\mathbf{y}_q^t \mathbf{y}_q\right)}$$

Not Equal

Degree $D_q = \sum_p w(\mathbf{x}_q, \mathbf{x}_p)$

Concave Laplacian

Remove constant terms

$$\mathcal{L}(\mathbf{Y}) = 2\left(\sum_q D_q - \sum_{q,p} w(\mathbf{x}_q, \mathbf{x}_p)\mathbf{y}_q^t \mathbf{y}_q\right)$$

Concave Laplacian

$$\boxed{\mathcal{L}(\mathbf{Y}) = -2 \sum_{q,p} w(\mathbf{x}_q, \mathbf{x}_p)\mathbf{y}_q^t \mathbf{y}_p}$$

Concave for PSD matrix $\mathbf{W} = [w(\mathbf{x}_q, \mathbf{x}_p)]$

Concave-Convex relaxation

Putting it altogether

$$\mathcal{R}(\mathbf{Y}) = \boxed{\mathbf{Y}^t \log \mathbf{Y}} + \mathcal{N}(\mathbf{Y}) + \lambda \mathcal{L}(\mathbf{Y})$$

Convex barrier function:

- Avoids extra dual variables for $\mathbf{y}_q \geq 0$
- Closed- form update for the simplex constraint duel

Bound optimization

First-order approximation of concave term

$$\mathcal{R}(\mathbf{Y}) = \mathbf{Y}^t \log \mathbf{Y} + \boxed{\mathcal{N}(\mathbf{Y})} + \boxed{\lambda \mathcal{L}(\mathbf{Y})}$$

Fixed unary

Bound optimization

We get **Iterative tight upper bound:**

Iteratively optimize:

$$\mathbf{Y}^{i+1} = \arg\min_{\mathbf{Y}} \mathcal{B}_i(\mathbf{Y})$$

$$B_i(\mathbf{Y}) = \sum_{q=1}^{N} \mathbf{y}_q^t(\log(\mathbf{y}_q) + \mathbf{a}_q - \lambda \mathbf{b}_q^i)$$

Where:

$$\mathbf{a}_q \quad = [a_{q,1}, \ldots, a_{q,C}]^t; \;\; a_{q,c} = d(\mathbf{x}_q, \mathbf{m}_c)$$

$$\mathbf{b}_q^i = [b_{q,1}^i, \ldots, b_{q,C}^i]^t; \;\; b_{q,c}^i = \sum_p w(\mathbf{x}_q, \mathbf{x}_p) y_{p,c}^i$$

Bound optimization

Independent **upper bound:**

$$B_i(\mathbf{Y}) = \sum_{q=1}^{N} \boxed{\mathbf{y}_q^t(\log(\mathbf{y}_q) + \mathbf{a}_q - \lambda \mathbf{b}_q^i)}$$

Bound optimization

Minimize Independent **upper bound:**

$$\min_{\mathbf{y}_q \in \nabla_C} \mathbf{y}_q^t(\log(\mathbf{y}_q) + \mathbf{a}_q - \lambda \mathbf{b}_q^i), \ \forall q$$

KKT conditions brings **closed form updates:**

$$\mathbf{y}_q^{i+1} = \frac{\exp(-\mathbf{a}_q^i + \lambda \mathbf{b}_q^i)}{\mathbf{1}^t \exp(-\mathbf{a}_q^i + \lambda \mathbf{b}_q^i)} \ \forall q$$

LaplacianShot Algorithm

Input: $\mathbb{X}_s$, $\mathbb{X}_q$, λ, f_θ
Output: $Labels \in \{1, .., C\}^N$ for $\mathbb{X}_q$
Get prototypes $\mathbf{m}_c$.
$\boxed{\text{Compute } \mathbf{a}_q\ \forall \mathbf{x}_q \in \mathbb{X}_q.}$
Initialize $i = 1$.
Initialize $\mathbf{y}_q^i = \frac{\exp(-\mathbf{a}_q)}{\mathbf{1}^t \exp(-\mathbf{a}_q)}$.
repeat
    $\boxed{\text{Compute } \mathbf{y}_q^{i+1}}$
    $\mathbf{y}_q^i \leftarrow \mathbf{y}_q^{i+1}$.
    $\mathbf{Y} = [\mathbf{y}_q^i];\ \forall q$.
    $i = i + 1$.
until $\mathcal{B}_i(\mathbf{Y})$ does not change
$l_q = \arg\max_c \mathbf{y}_q;\ \forall \mathbf{y}_q \in \mathbf{Y}$.
$Labels = \{l_q\}_{q=1}^N$

$$\boxed{\mathbf{a}_q = [a_{q,1}, \ldots, a_{q,C}]^t;\ a_{q,c} = d(\mathbf{x}_q, \mathbf{m}_c)}$$

$$\boxed{\mathbf{b}_q^i = [b_{q,1}^i, \ldots, b_{q,C}^i]^t;\ b_{q,c}^i = \sum_p w(\mathbf{x}_q, \mathbf{x}_p) y_{p,c}^i}$$

$$\boxed{\mathbf{y}_q^{i+1} = \frac{\exp(-\mathbf{a}_q^i + \lambda \mathbf{b}_q^i)}{\mathbf{1}^t \exp(-\mathbf{a}_q^i + \lambda \mathbf{b}_q^i)}\ \forall q}$$

Experiments

Datasets:

1. *Mini*-ImageNet
2. *Tierd-ImageNet*
3. CUB 200-2001
4. Inat

Generic Classification

***mini*ImageNet splits:** 64 base, 16 validation and 20 test classes

***tiered*ImageNet splits:** 351 base, 97 validation and 160 test classes

Fine-Grained Classification

Splits: 100 base, 50 validation and 50 test classes

Experiments

Datasets:

1. *Mini*-ImageNet
2. *Tierd-ImageNet*
3. CUB 200-2001
4. Inat

Evaluation protocol:

- **5**-way **1**-shot/**5**-shot .
- **15** query samples per class (N=75).
- **Average accuracy** over **10,000** few-shot tasks with 95% confidence interval.

Experiments

Datasets:

1. *Mini*-ImageNet
2. *Tierd-ImageNet*
3. CUB 200-2001
4. Inat

- More realistic and challenging
- Recently introduced (Wertheimer& Hariharan, 2019)
- Slight class distinction
- **Imbalanced class distribution** with variable number of supports/query per class

Experiments

Datasets:

1. *Mini*-ImageNet
2. *Tierd-ImageNet*
3. CUB 200-2001
4. Inat

Evaluation protocol:

- **227**-way **multi-shot** .
- **Top-1 accuracy** averaged over the test images **Per Class**.
- **Top-1 accuracy** averaged over **all the** test images (**Mean**)

Experiments

We do
Cross-entropy training with base classes

LaplacianShot during inference

Results (Mini-ImageNet)

Methods	Network	1-shot	5-shot
MAML [Finn et al., 2017]	ResNet-18	49.61 ± 0.92	65.72 ± 0.77
Chen [Chen et al., 2019]	ResNet-18	51.87 ± 0.77	75.68 ± 0.63
RelationNet [Sung et al., 2018]	ResNet-18	52.48 ± 0.86	69.83 ± 0.68
MatchingNet [Vinyals et al., 2016]	ResNet-18	52.91 ± 0.88	68.88 ± 0.69
ProtoNet [Snell et al., 2017]	ResNet-18	54.16 ± 0.82	73.68 ± 0.65
Gidaris [Gidaris and Komodakis, 2018]	ResNet-15	55.45 ± 0.89	70.13 ± 0.68
SNAIL[Mishra et al., 2018]	ResNet-15	55.71 ± 0.99	68.88 ± 0.92
AdaCNN [Munkhdalai et al., 2018]	ResNet-15	56.88 ± 0.62	71.94 ± 0.57
TADAM [Oreshkin et al., 2018]	ResNet-15	58.50 ± 0.30	76.70 ± 0.30
CAML [Jiang et al., 2019]	ResNet-12	59.23 ± 0.99	72.35 ± 0.71
TPN [Yanbin et al., 2019]	ResNet-12	59.46	75.64
TEAM [Qiao et al., 2019]	ResNet-18	60.07	75.90
MTL [Sun et al., 2019]	ResNet-18	61.20 ± 1.80	75.50 ± 0.80
VariationalFSL [Zhang et al., 2019]	ResNet-18	61.23 ± 0.26	77.69 ± 0.17
Transductive tuning [Dhillon et al., 2020]	ResNet-12	62.35 ± 0.66	74.53 ± 0.54
MetaoptNet[Lee et al., 2019]	ResNet-18	62.64 ± 0.61	78.63 ± 0.46
SimpleShot [Wang et al., 2019]	ResNet-18	63.10 ± 0.20	79.92 ± 0.14
CAN+T [Hou et al., 2019]	ResNet-12	67.19 ± 0.55	80.64 ± 0.35
LaplacianShot (ours)	ResNet-18	$\mathbf{72.11} \pm 0.19$	$\mathbf{82.31} \pm 0.14$

Results (Mini-ImageNet)

Methods	Network	1-shot	5-shot
Qiao [Qiao et al., 2018]	WRN	59.60 ± 0.41	73.74 ± 0.19
LEO [Rusu et al., 2019]	WRN	61.76 ± 0.08	77.59 ± 0.12
ProtoNet [Snell et al., 2017]	WRN	62.60 ± 0.20	79.97 ± 0.14
CC+rot[Gidaris et al., 2019]	WRN	62.93 ± 0.45	79.87 ± 0.33
MatchingNet [Vinyals et al., 2016]	WRN	64.03 ± 0.20	76.32 ± 0.16
FEAT [Ye et al., 2020]	WRN	65.10 ± 0.20	81.11 ± 0.14
Transductive tuning [Dhillon et al., 2020]	WRN	65.73 ± 0.68	78.40 ± 0.52
SimpleShot [Wang et al., 2019]	WRN	65.87± 0.20	82.09 ± 0.14
SIB [Hu et al., 2020]	WRN	70.0 ± 0.6	79.2 ± 0.4
BD-CSPN [Liu et al., 2019]	WRN	70.31 ± 0.93	81.89 ± 0.60
LaplacianShot (ours)	WRN	**74.86** ± 0.19	**84.13** ± 0.14

Results (Tiered-ImageNet)

Methods	**Network**	**1-shot**	**5-shot**
MetaoptNet[Lee et al., 2019]	ResNet-18	65.99 ± 0.72	81.56 ± 0.53
SimpleShot [Wang et al., 2019]	ResNet-18	69.68 ± 0.22	84.56 ± 0.16
CAN+T [Hou et al., 2019]	ResNet-12	73.21 ± 0.58	84.93 ± 0.38
LaplacianShot (ours)	ResNet-18	$\mathbf{78.98} \pm 0.21$	$\mathbf{86.39} \pm 0.16$
Meta SGD [Li et al., 2017]	WRN	62.95 ± 0.03	79.34 ± 0.06
LEO [Rusu et al., 2019]	WRN	66.33 ± 0.05	81.44 ± 0.09
FEAT [Ye et al., 2020]	WRN	70.41 ± 0.23	84.38 ± 0.16
CC+rot[Gidaris et al., 2019]	WRN	70.53 ± 0.51	84.98 ± 0.36
SimpleShot [Wang et al., 2019]	WRN	70.90 ± 0.22	85.76 ± 0.15
Transductive tuning [Dhillon et al., 2020]	WRN	73.34 ± 0.71	85.50 ± 0.50
BD-CSPN [Liu et al., 2019]	WRN	78.74 ± 0.95	86.92 ± 0.63
LaplacianShot (ours)	WRN	$\mathbf{80.18} \pm 0.21$	$\mathbf{87.56} \pm 0.15$

Results (CUB)

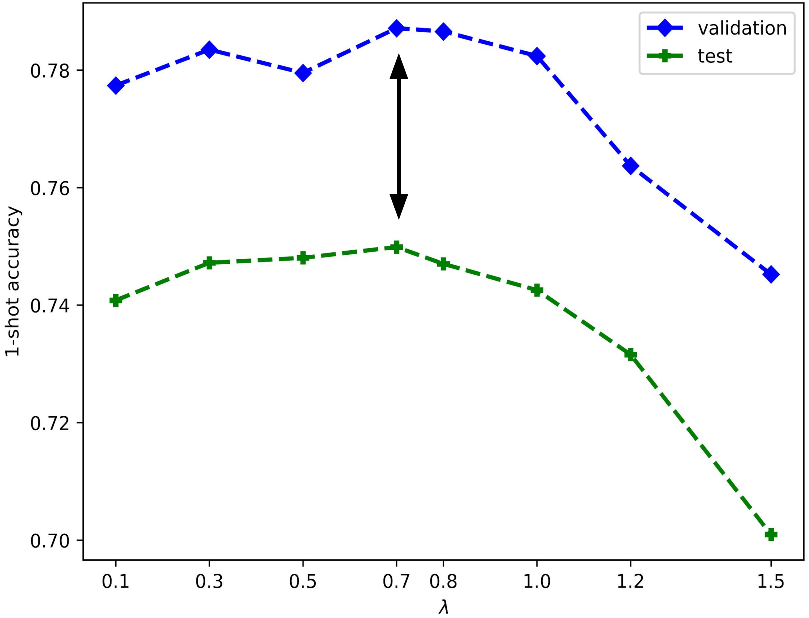
Methods	Network	CUB		*mini*Imagenet → CUB	
		1-shot	5-shot	1-shot	5-shot
MatchingNet [Vinyals et al., 2016]	ResNet-18	73.49	84.45	-	53.07
MAML [Finn et al., 2017]	ResNet-18	68.42	83.47	-	51.34
ProtoNet [Snell et al., 2017]	ResNet-18	72.99	86.64	-	62.02
RelationNet [Sung et al., 2018]	ResNet-18	68.58	84.05	-	57.71
Chen [Chen et al., 2019]	ResNet-18	67.02	83.58	-	65.57
SimpleShot [Wang et al., 2019]	ResNet-18	70.28	86.37	48.56	65.63
LaplacianShot(ours)	ResNet-18	**80.96**	**88.38**	**55.46**	**66.33**

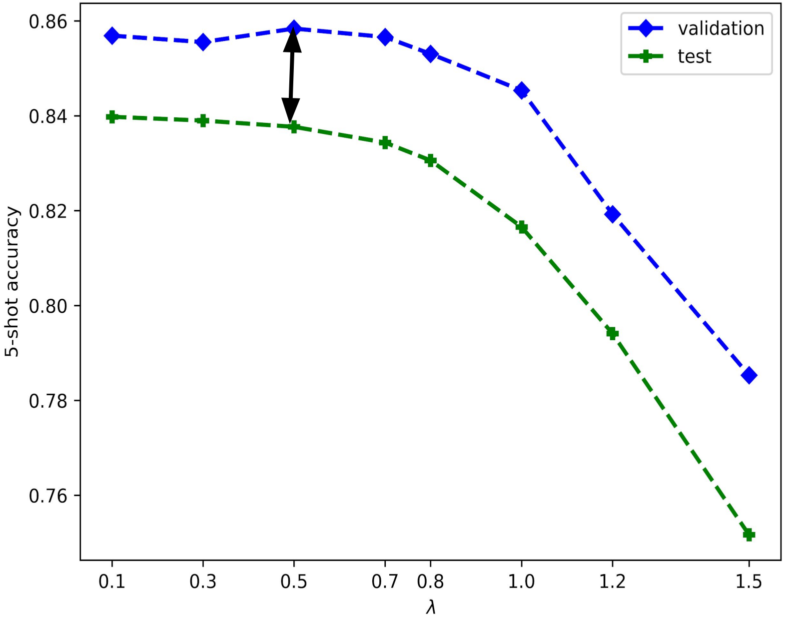
Cross Domain

Results (iNat)

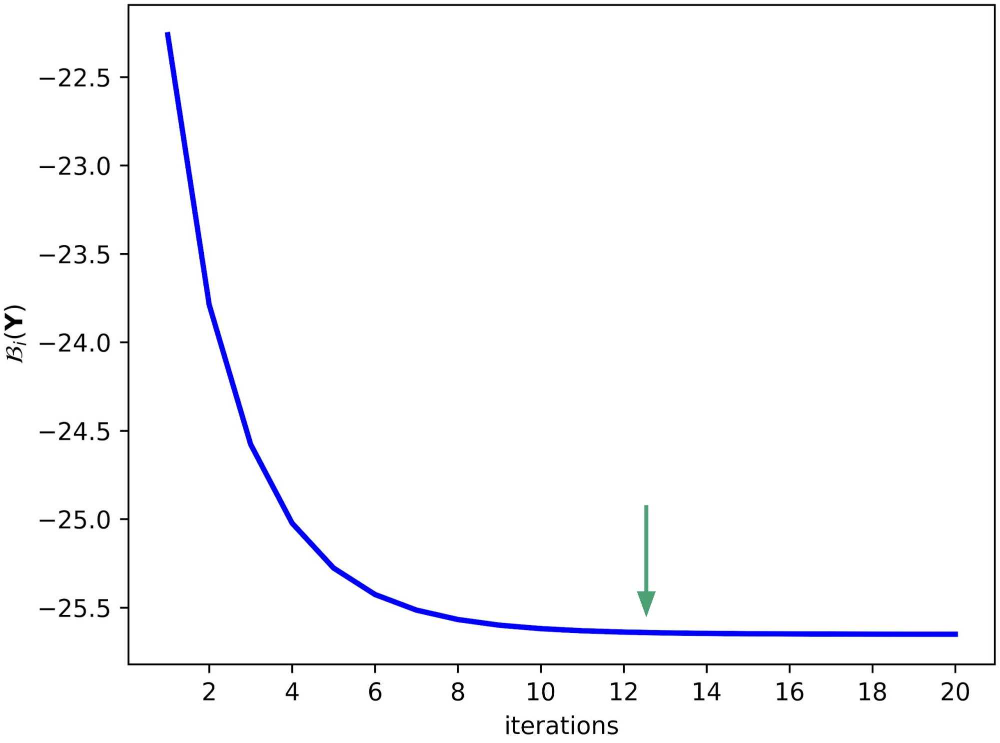
Methods	Network	Per Class	Mean
SimpleShot [Wang et al., 2019]	ResNet-18	55.80	58.56
LaplacianShot	ResNet-18	**62.80**	**66.40**
PN+BF+fsL+CP [Wertheimer and Hariharan, 2019]	ResNet-50	46.04	51.25
SimpleShot [Wang et al., 2019]	ResNet-50	58.45	61.07
LaplacianShot	ResNet-50	**65.96**	**69.13**
SimpleShot [Wang et al., 2019]	WRN	62.44	65.08
LaplacianShot	WRN	**71.55**	**74.97**

Ablation: Choosing λ





Ablation: Convergence



Ablation: Average Inference time

Methods	Network	inference time (s)
SimpleShot [Wang et al., 2019]	WRN	0.009
Transductive tuning [Dhillon et al., 2020]	WRN	20.7
LaplacianShot	WRN	0.012

Transductive

LaplacianShot Takeaways

- ✓ **SOTA results** without bell and whistles.
- ✓ Simple **constrained graph clustering** works very well.
- ✓ **No** network **fine-tuning**, neither **meta-learning**
- ✓ **Model Agnostic:** during inference with any training model and gain up to 4/5%!!!
- ✓ **Fast** transductive inference: almost inductive time

Thank you

Code On:

https://github.com/imtiazziko/LaplacianShot