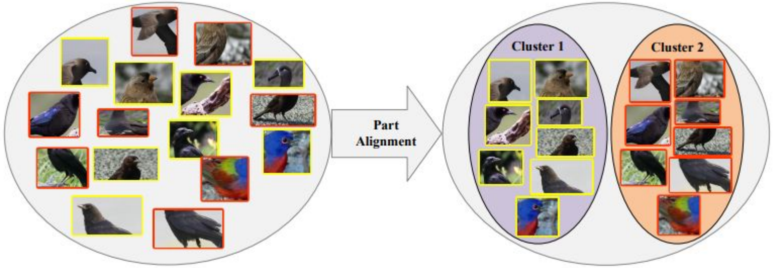
EECS 542: Class Presentation - Parts and Wholes

Attentional Constellation Nets for Few-shot Learning [2021]

Sachin Salim
sachinks@umich.edu

Problem Statement

Objective: To address the limitations of existing CNN frameworks in capturing explicit structured features, particularly object parts, for few-shot learning.



(a) Object-Level Attention Model

(b) Part-Level Attention Model

Image Courtesy: Object-Part Attention Model for Fine-grained Image Classification

Previous Work 1 : Constellation

Neural Activation Constellations [2015]

1. Deep neural activation maps identify CNN channels as part detectors.

2. Unsupervised data selection of part detectors based on similar relative locations forms part models.

3. Utilizing these part models extracts object part features for weakly-supervised classification.

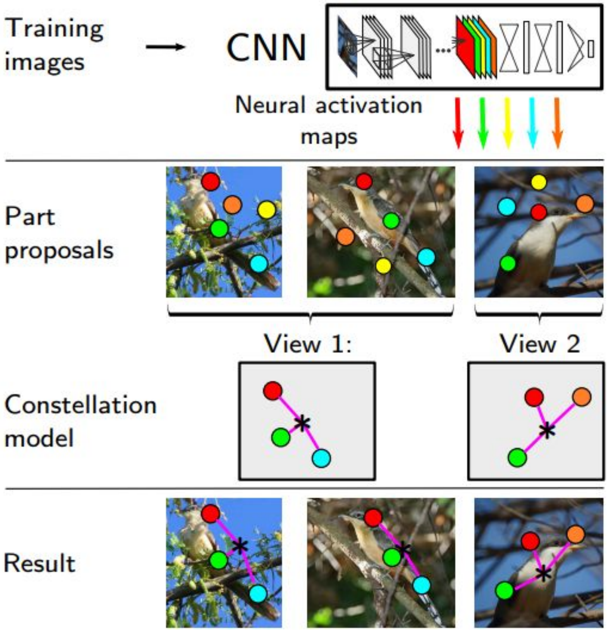


Image Courtesy: Neural Activation Constellations

Improvements in Constellation Architecture

Aspect	**Neural Activation Constellations**	**Attentional ConstellationNet (Ours)**
Part representation and spatial modeling	Gaussian-based constellation module	Cell feature clustering and self-attention
Integration and Optimization	Constellation module optimization is separate from CNN optimization	Seamlessly integrates constellation modules with CNNs, jointly optimizing them
Representation Utilization	Extracts sparse part representations	Utilizes dense cell features from CNN feature maps

Few-shot Learning

Standard Classification

- Train and test dataset have same categories
- C-base = C-novel

Few-shot Classification

- Train and test dataset have different categories to ensure fairness
- C-base ∩ C-novel = Ø

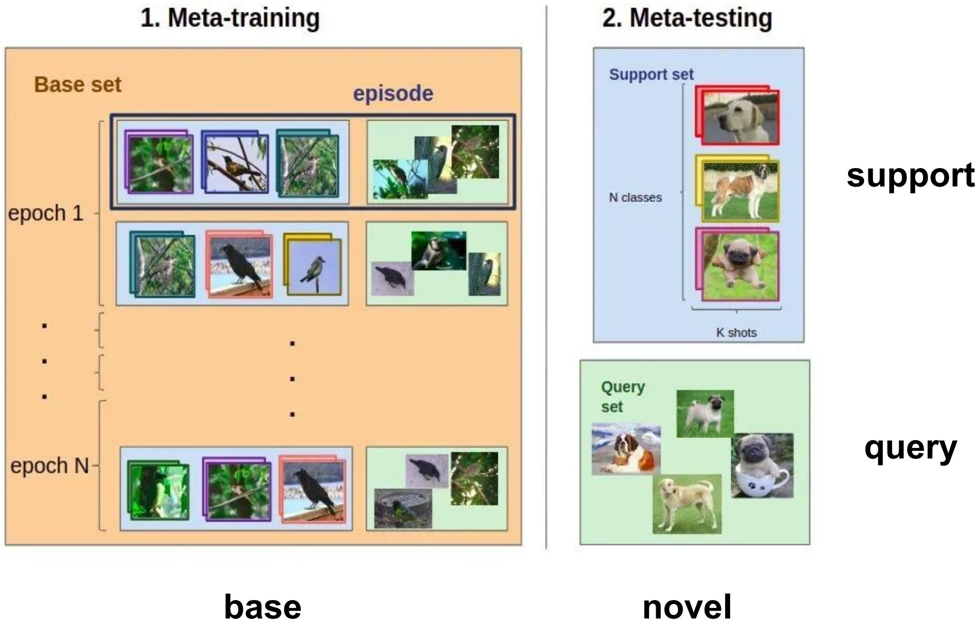


Figure Courtesy: Understanding Few-Shot Learning in Computer Vision: What You Need to Know, Neptune AI

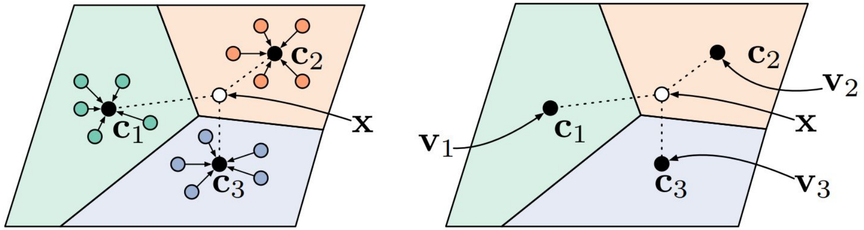
Previous Work 2 : Few-shot learning

Prototypical Networks [2017]

1. Utilizes a metric-based approach for few-shot learning.

2. Computes prototype representations of classes in a learned feature space.

3. Assigns new instances to the nearest prototype class for classification.



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

Algorithm 1 Training episode loss computation for Prototypical Networks. N is the number of examples in the training set, K is the number of classes in the training set, $N_C \leq K$ is the number of classes per episode, N_S is the number of support examples per class, N_Q is the number of query examples per class. RANDOMSAMPLE(S, N) denotes a set of N elements chosen uniformly at random from set S, without replacement.

Input: Training set $\mathcal{D} = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)\}$, where each $y_i \in \{1, \ldots, K\}$. $\mathcal{D}_k$ denotes the subset of $\mathcal{D}$ containing all elements $(\mathbf{x}_i, y_i)$ such that $y_i = k$.

Output: The loss J for a randomly generated training episode.

$V \leftarrow$ RANDOMSAMPLE$(\{1, \ldots, K\}, N_C)$ ▷ Select class indices for episode

for k in $\{1, \ldots, N_C\}$ **do**

$S_k \leftarrow$ RANDOMSAMPLE$(\mathcal{D}_{V_k}, N_S)$ ▷ Select support examples

$Q_k \leftarrow$ RANDOMSAMPLE$(\mathcal{D}_{V_k} \setminus S_k, N_Q)$ ▷ Select query examples

$\mathbf{c}_k \leftarrow \frac{1}{N_C} \sum_{(\mathbf{x}_i, y_i) \in S_k} f_\phi(\mathbf{x}_i)$ ▷ Compute prototype from support examples

end for

$J \leftarrow 0$ ▷ Initialize loss

for k in $\{1, \ldots, N_C\}$ **do**

for $(\mathbf{x}, y)$ in Q_k **do**

$J \leftarrow J + \frac{1}{N_C N_Q} \left[d(f_\phi(\mathbf{x}), \mathbf{c}_k)) + \log \sum_{k'} \exp(-d(f_\phi(\mathbf{x}), \mathbf{c}_{k'})) \right]$ ▷ Update loss

end for

end for

Figure Courtesy: Prototypical Networks

Improvements in Few-shot framework

Aspect	ProtoNet	Attentional ConstellationNet (Ours)
Explicit Structured Representations	Metric-based framework	Explicit structured representations
Distance	Euclidean distance	Cosine similarity
Training Scheme	Prototypical scheme (episodic learning)	Standard classification scheme

ConstellationNet pipeline

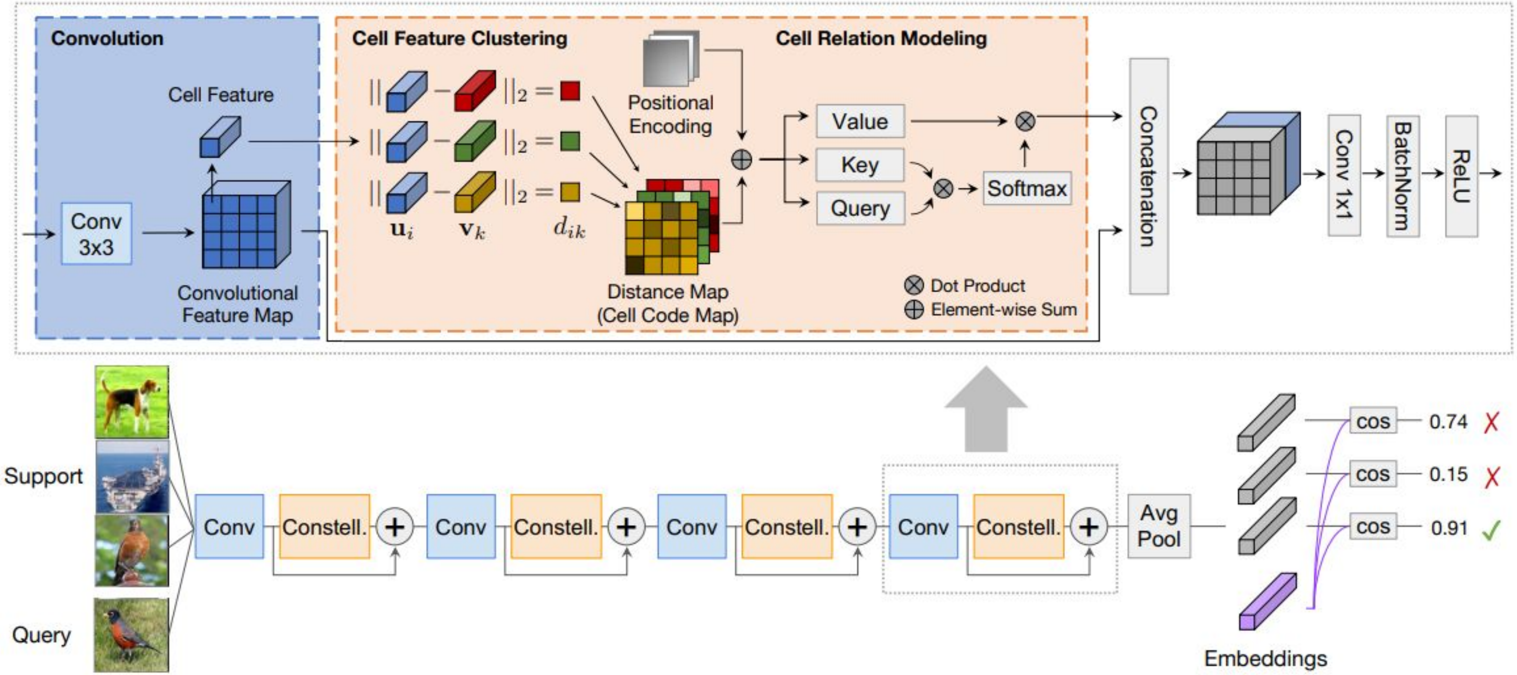
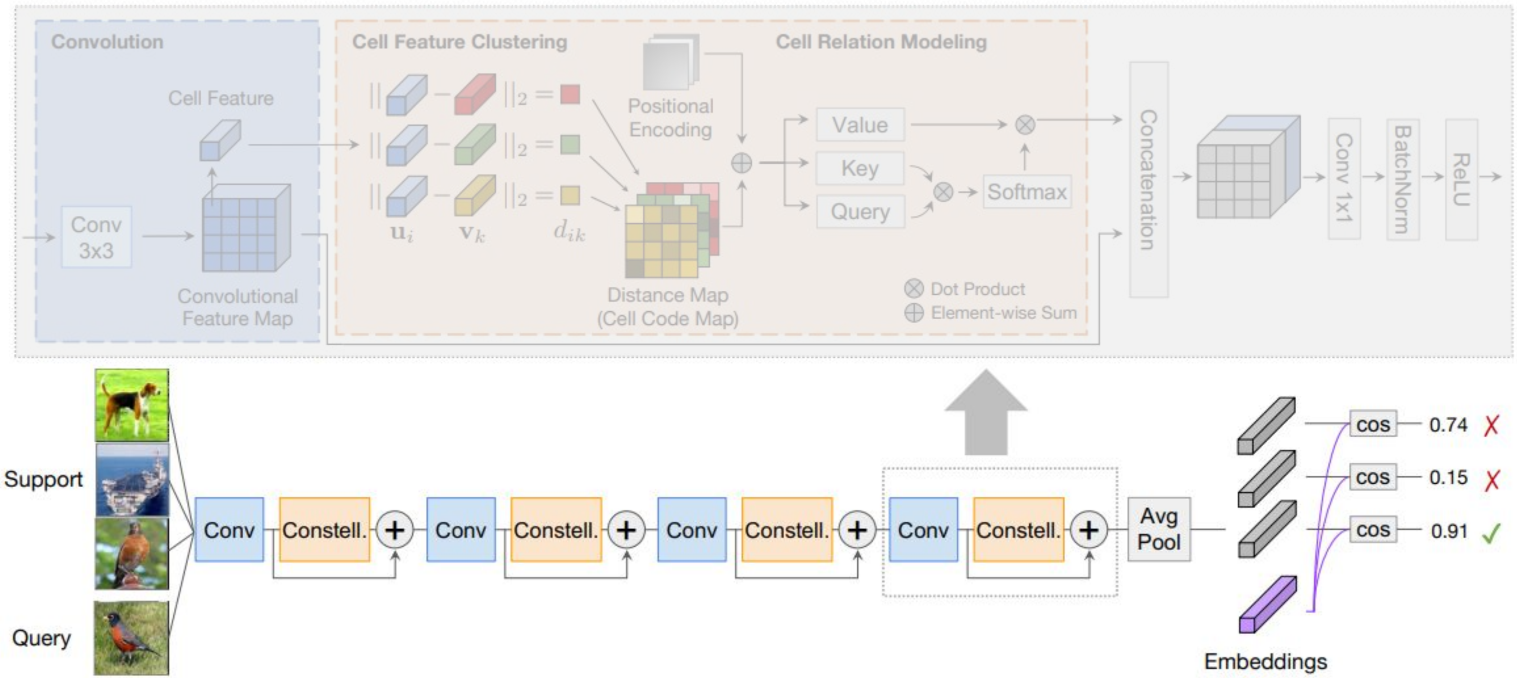


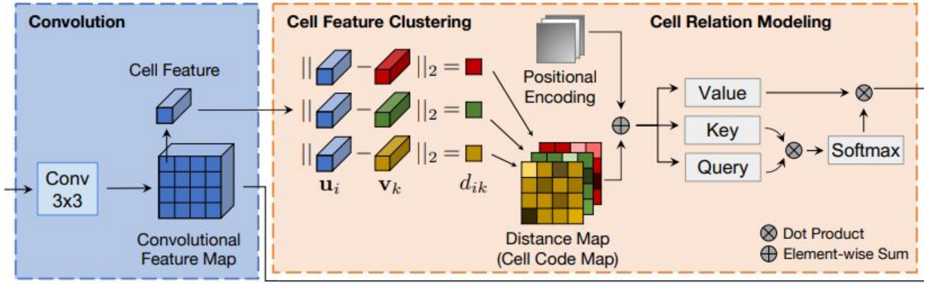
Figure Courtesy: Attentional Constellation Nets for Few-shot Learning

ConstellationNet pipeline - Network architecture



Constellation block - Overview

- **Cell Features**: Dense, individual local feature at a position in the feature map
- **Clustering**: Generates cell codes to model the underlying distribution of input cell features
- **Spatial Position Encoding**: Includes positional encoding for spatial information.
- **Tokenized Representation**: Represents cells as code + positional encoding.
- **Self-Attention**: Captures part relationships and spatial configurations using self-attention.



Cell Feature Clustering

Input cell features $\mathcal{U} = \{\mathbf{u}_1, \mathbf{u}_2, ..., \mathbf{u}_n\}$

1. **Initialization**

$$\mathcal{V} = \{\mathbf{v}_1, \mathbf{v}_2, ..., \mathbf{v}_K\} \qquad \mathbf{s} = (s_1, s_2, ..., s_K) = \mathbf{0}$$

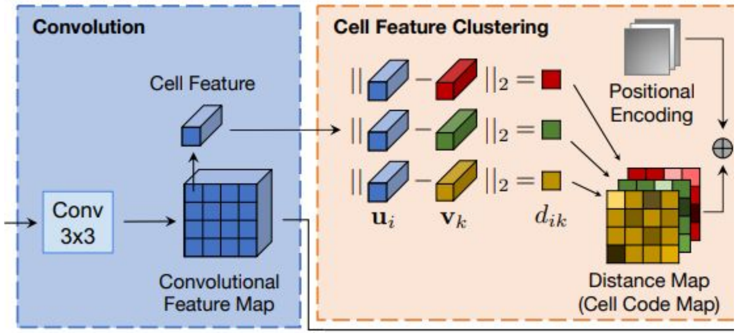
2. **Cluster Assignment**

$$d_{ik} = ||\mathbf{u}_i - \mathbf{v}_k||_2^2, \qquad m_{ik} = \frac{e^{-\beta d_{ik}}}{\sum_j e^{-\beta d_{ij}}}, \qquad \mathbf{v}'_k = \frac{\sum_i m_{ik}\mathbf{u}_i}{\sum_i m_{ik}}$$

3. **Centroid Movement & Counter Update**

$$\mathbf{v}_k \leftarrow (1-\eta)\mathbf{v}_k + \eta\mathbf{v}'_k, \qquad \eta = \frac{\lambda}{s_k + \Delta s_k}$$

$$\Delta\mathbf{s} = \sum_i \mathbf{m}_i \qquad \mathbf{s} \leftarrow \mathbf{s} + \Delta\mathbf{s}$$



Cell Relation & Spatial Configuration Modeling

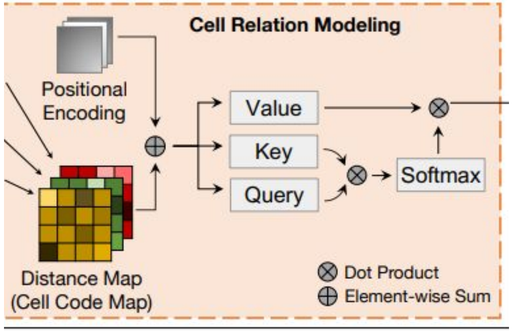
Self-attention mechanism is used to build the spatial relationship

$$\mathbf{F}_{\mathrm{I}} = \mathrm{SpatialFlatten}(\mathbf{D} + \mathbf{P}) \in \mathbb{R}^{B \times HW \times K}, \quad \mathbf{F}'_{\mathrm{I}} = \mathrm{SpatialFlatten}(\mathbf{D}) \in \mathbb{R}^{B \times HW \times K}$$

$$[\mathbf{F}^q, \mathbf{F}^k, \mathbf{F}^v] = [\mathbf{F}_{\mathrm{I}}\mathbf{W}^q, \mathbf{F}_{\mathrm{I}}\mathbf{W}^k, \mathbf{F}'_{\mathrm{I}}\mathbf{W}^v]$$

$$\mathbf{F}_{\mathrm{A}} = \mathrm{Att}(\mathbf{F}^q, \mathbf{F}^k, \mathbf{F}^v) = \mathrm{softmax}\left(\frac{\mathbf{F}^q(\mathbf{F}^k)^\top}{\sqrt{K}}\right)\mathbf{F}^v$$

$$\mathbf{F}_{\mathrm{MHA}} = \mathrm{MultiHeadAtt}(\mathbf{F}^q, \mathbf{F}^k, \mathbf{F}^v) = [\mathbf{F}_1, ..., \mathbf{F}_J]\mathbf{W}, \qquad \mathbf{F}_j = \mathrm{Att}(\mathbf{F}_j^q, \mathbf{F}_j^k, \mathbf{F}_j^v)$$



Experiments with Standard Benchmarks

Datasets: CIFAR-FS, FC100, mini-ImageNet

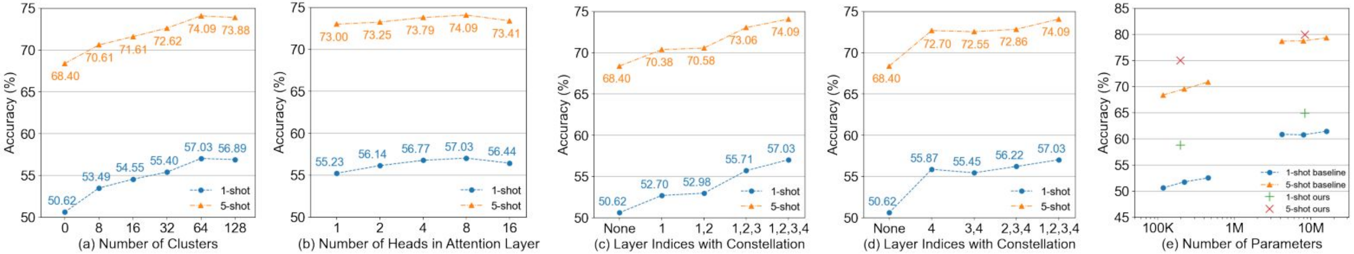
Model	Backbone	*mini*-ImageNet 5-way	
		1-shot	5-shot
Meta-Learning LSTM (Ravi & Larochelle, 2016)	Conv-4	43.44 ± 0.77	60.60 ± 0.71
Matching Networks (Vinyals et al., 2016)	Conv-4	43.56 ± 0.84	55.31 ± 0.73
Prototypical Networks (Snell et al., 2017)	Conv-4	49.42 ± 0.78	68.20 ± 0.66
Transductive Prop Nets (Liu et al., 2018)	Conv-4	55.51 ± 0.86	69.86 ± 0.65
MetaOptNet (Lee et al., 2019)	Conv-4	52.87 ± 0.57	68.76 ± 0.48
Negative Margin (Liu et al., 2020)	Conv-4	52.84 ± 0.76	70.41 ± 0.66
ConstellationNet (ours)	Conv-4	$\mathbf{58.82 \pm 0.23}$	$\mathbf{75.00 \pm 0.18}$
SNAIL (Mishra et al., 2018)	ResNet-12	55.71 ± 0.99	68.88 ± 0.92
TADAM (Oreshkin et al., 2018)	ResNet-12	58.50 ± 0.30	76.70 ± 0.30
TapNet (Yoon et al., 2019)	ResNet-12	61.65 ± 0.15	76.36 ± 0.10
Variational FSL (Zhang et al., 2019)	ResNet-12	61.23 ± 0.26	77.69 ± 0.17
MetaOptNet (Lee et al., 2019)	ResNet-12	62.64 ± 0.61	78.63 ± 0.46
CAN (Hou et al., 2019)	ResNet-12	63.85 ± 0.48	79.44 ± 0.34
SLA-AG (Lee et al., 2020)	ResNet-12	62.93 ± 0.63	79.63 ± 0.47
Meta-Baseline (Chen et al., 2020)	ResNet-12	63.17 ± 0.23	79.26 ± 0.17
AM3 (Xing et al., 2019) †	ResNet-12	65.21 ± 0.30	75.20 ± 0.27
ProtoNets + TRAML (Li et al., 2020)	ResNet-12	60.31 ± 0.48	77.94 ± 0.57
AM3 + TRAML (Li et al., 2020) †	ResNet-12	$\mathbf{67.10 \pm 0.52}$	79.54 ± 0.60
Negative Margin (Liu et al., 2020)	ResNet-12	63.85 ± 0.81	$\mathbf{81.57 \pm 0.56}$
ConstellationNet (ours)	ResNet-12	64.89 ± 0.23	79.95 ± 0.17

Model	Backbone	CIFAR-FS 5-way		FC100 5-way	
		1-shot	5-shot	1-shot	5-shot
MAML (Finn et al., 2017)	Conv-4	58.9 ± 1.9	71.5 ± 1.0	-	-
Prototypical Networks (Snell et al., 2017)	Conv-4	55.5 ± 0.7	72.0 ± 0.6	-	-
Relation Networks (Sung et al., 2018)	Conv-4	55.0 ± 1.0	69.3 ± 0.8	-	-
R2D2 (Bertinetto et al., 2018)	Conv-4	65.3 ± 0.2	79.4 ± 0.1	-	-
SIB (Hu et al., 2020)	Conv-4	68.7 ± 0.6	77.1 ± 0.4	-	-
ConstellationNet (ours)	Conv-4	$\mathbf{69.3 \pm 0.3}$	$\mathbf{82.7 \pm 0.2}$	-	-
Prototypical Networks (Snell et al., 2017)	ResNet-12	72.2 ± 0.7	83.5 ± 0.5	37.5 ± 0.6	52.5 ± 0.6
TADAM (Oreshkin et al., 2018)	ResNet-12	-	-	40.1 ± 0.4	56.1 ± 0.4
MetaOptNet-RR (Lee et al., 2019)	ResNet-12	72.6 ± 0.7	84.3 ± 0.5	40.5 ± 0.6	55.3 ± 0.6
MetaOptNet-SVM (Lee et al., 2019)	ResNet-12	72.0 ± 0.7	84.2 ± 0.5	41.1 ± 0.6	55.5 ± 0.6
ConstellationNet (ours)	ResNet-12	$\mathbf{75.4 \pm 0.2}$	$\mathbf{86.8 \pm 0.2}$	$\mathbf{43.8 \pm 0.2}$	$\mathbf{59.7 \pm 0.2}$

- ConstellationNet significantly performance better across various few-shot learning settings
- Improvement over both shallow & deep networks across all three datasets shows generality

Effectiveness of modules

Baseline	Cell Feature Clustering	Cell Relation Modeling	Multi Branch	Feature Augment	Extra Channels	1x1 Convolution	#Params	Conv-4		ResNet-12	
							Conv-4/Res-12	1-shot	5-shot	1-shot	5-shot
✓							117K/8.0M	50.62 ± 0.23	68.40 ± 0.19	60.77 ± 0.22	78.76 ± 0.17
✓					✓		222K/16M	51.76 ± 0.22	69.54 ± 0.18	61.45 ± 0.22	79.33 ± 0.16
✓	✓						146K/8.3M	53.34 ± 0.23	70.61 ± 0.19	62.24 ± 0.23	79.55 ± 0.16
✓		✓					184K/9.7M	55.92 ± 0.23	73.02 ± 0.18	62.75 ± 0.23	79.21 ± 0.17
✓		✓				✓	192K/8.4M	55.46 ± 0.23	72.52 ± 0.18	61.54 ± 0.24	76.51 ± 0.18
✓	✓	✓					200K/8.4M	57.03 ± 0.23	74.09 ± 0.18	63.36 ± 0.23	79.72 ± 0.17
✓	✓	✓	✓				200K/8.4M	58.37 ± 0.23	74.52 ± 0.18	64.62 ± 0.23	79.60 ± 0.17
✓	✓	✓	✓	✓			200K/8.4M	**58.82 ± 0.23**	**75.00 ± 0.18**	**64.89 ± 0.23**	**79.95 ± 0.17**
							WRN			WideResNet-28-10	
✓					✓		36.5M			61.54 ± 0.25	79.41 ± 0.23



Performance gain of ConstellationNet is unmatched by increasing the model complexity of baselines

Visualization of cluster centers

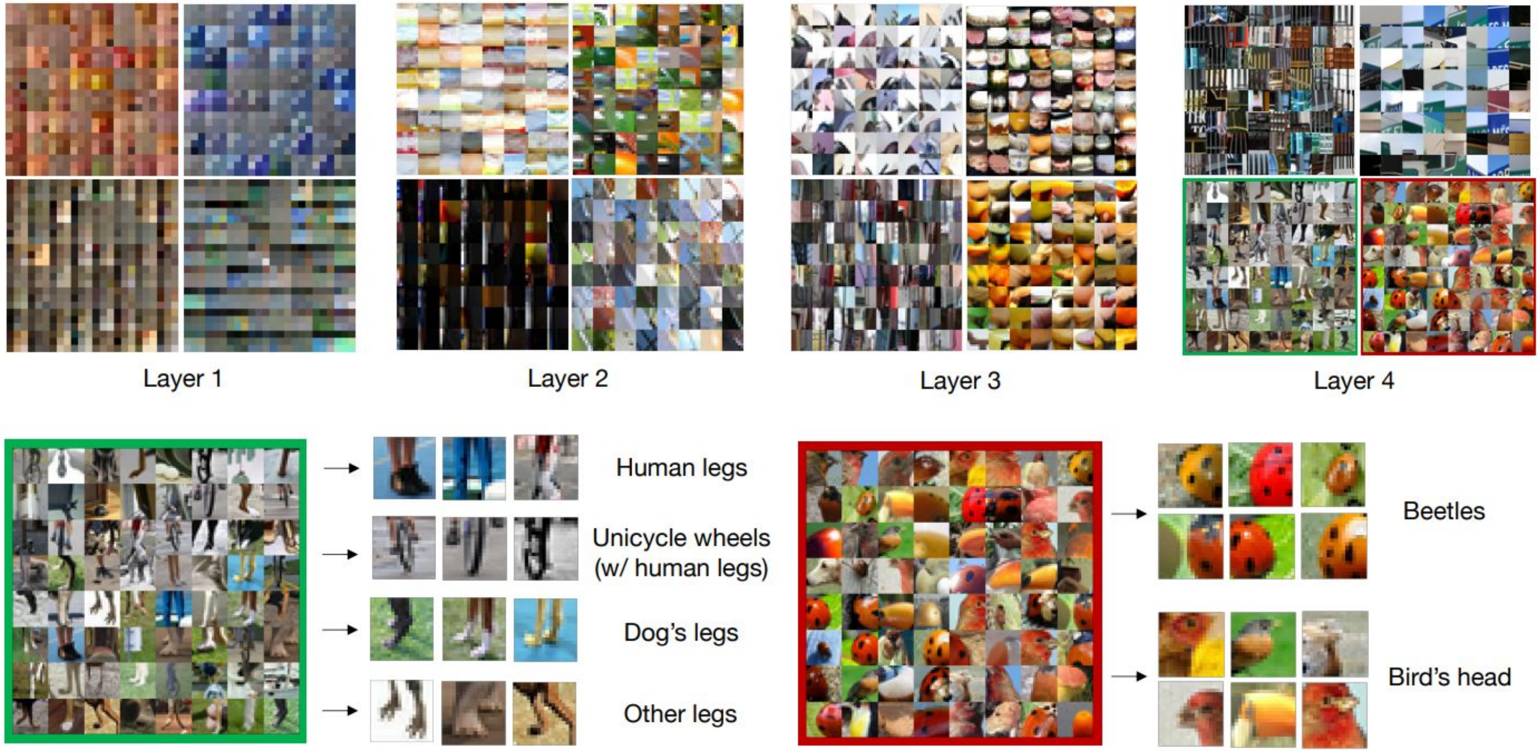


Figure Courtesy: Attentional Constellation Nets for Few-shot Learning

Visualization of cells assignment and attention maps

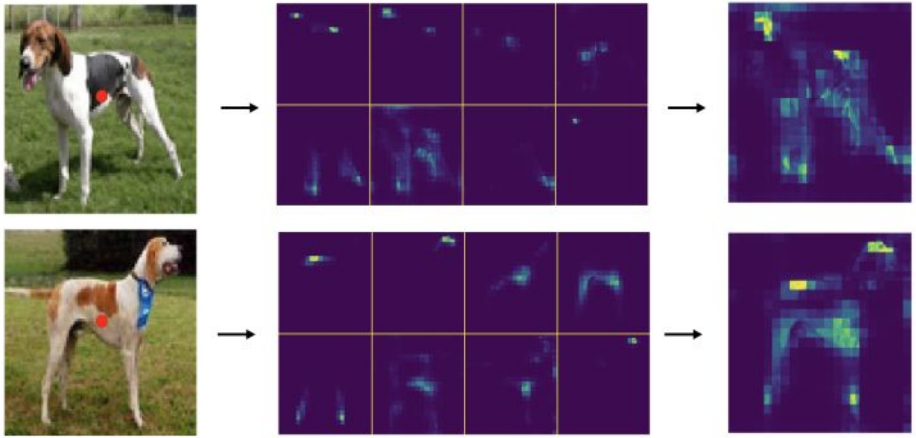


Figure Courtesy: Attentional Constellation Nets for Few-shot Learning

Review

Summary
The paper introduces Attentional Constellation Nets as a novel framework for few-shot learning. By integrating CNNs with a constellation model featuring cell-wise clustering, self-attention, and dense part representations, ConstellationNet enhances structured features, improving CNNs' robustness in few-shot scenarios. The method demonstrates significant performance gains over existing approaches, showcasing its promise in addressing few-shot learning challenges in computer vision.

Strengths

- Innovative Integration: merges CNNs with a constellation model, addressing structured feature limitations for few-shot learning
- Extensive Evaluation: Comprehensive assessment on benchmarks demonstrates significant performance enhancements over existing methods.
- Clear Explanations: Provides clear descriptions of framework components, aiding reader comprehension

Weaknesses

- Ablation Study Detail: Lacks detailed discussion on ablation study outcomes to reinforce robustness.
- Comparative Analysis: Requires deeper comparison against state-of-the-art few-shot learning methods to highlight competitiveness.
- Computational Efficiency Insight: Absence of discussion on computational efficiency limits insights into practical feasibility

Final Rating: Weak Accept
Confidence: 4

Justification: The approach is with promising results but lacks in-depth comparative analysis and details on computational efficiency

Future Directions

1. **Hierarchical Constellation Modeling** to capture more intricate spatial relationships and part-based representations for enhanced feature learning.

2. **Dynamic Attention Mechanisms** to enable the network to dynamically adjust attention weights based on the importance of spatial configurations and parts.

3. **Incorporating Few-shot Regression** could open avenues for broader applications in diverse domains beyond classification.

Discussions

1. How might the **hierarchical integration** of constellation modeling impact the interpretability and generalization of learned representations, especially when handling complex visual tasks in diverse domains beyond image classification?

2. What are the potential implications of incorporating **dynamic attention mechanisms** within ConstellationNet for addressing the scalability and adaptability of few-shot learning in real-world scenarios?

3. Considering the reliance on **unsupervised part-discovery strategies** in ConstellationNet, how could this approach influence the network's robustness to domain shifts or data augmentation, and what implications does it have for transfer learning and model adaptation across various datasets?

Answers for Discussion

1. Hierarchical modeling might improve interpretability but could pose challenges in computational efficiency, demanding exploration for optimal task-specific levels.
2. Dynamic attention could enhance adaptability by allowing ConstellationNet to adjust focus based on data complexity, potentially improving generalization on novel tasks.
3. Relying on unsupervised part-discovery may confer robustness to domain shifts, but its effectiveness across diverse real-world scenarios needs further investigation.

Piazza Discussions

1. What is the reason/intuition behind the performance being much better by the combination of explicit structured features and spatial relations, rather than simply increasing CNN channels? 116_f5
    a. Comment: Increasing the number of CNN channels can capture more patterns in the data, but it may not necessarily encode specific structured information or spatial relationships explicitly.
2. The visualizations confirm the clustering is learning semantically meaningful parts. Quantitatively evaluating the part quality could be an interesting analysis. 116_f10
3. Why cell features from a convolutional layer are representative for relational modeling? 116_f7