Context-Aware Graph Convolution for Image Retrieval

Ushasi Chaudhuri

Indian Institute of Technology Bombay (IITB)

Outline

- Retrieval problem in satellite images.
- Siamese GCN for image retrieval.
- Attention-based GCN.
- Experimental results.
- Looking ahead — Exploring the effects of 'semantics'.

Issues with Image Retrieval Problem

- Availability of wide range of satellite sensors, accumulation of an unprecedented volume of remote sensing images.
- Retrieval is necessary and challenging!
- Standard CNN based feature descriptors summarize the **global** scene level information from the images.
- Need to capture local regions in context in recognizing complex land-cover classes. Explore GCN!




Capturing local context

1. Represent the image **globally**, but highlight interactions among **local scene** constructs.
2. Discriminative learning of an embedding space from an **irregular spatial distribution** of the local constructs.

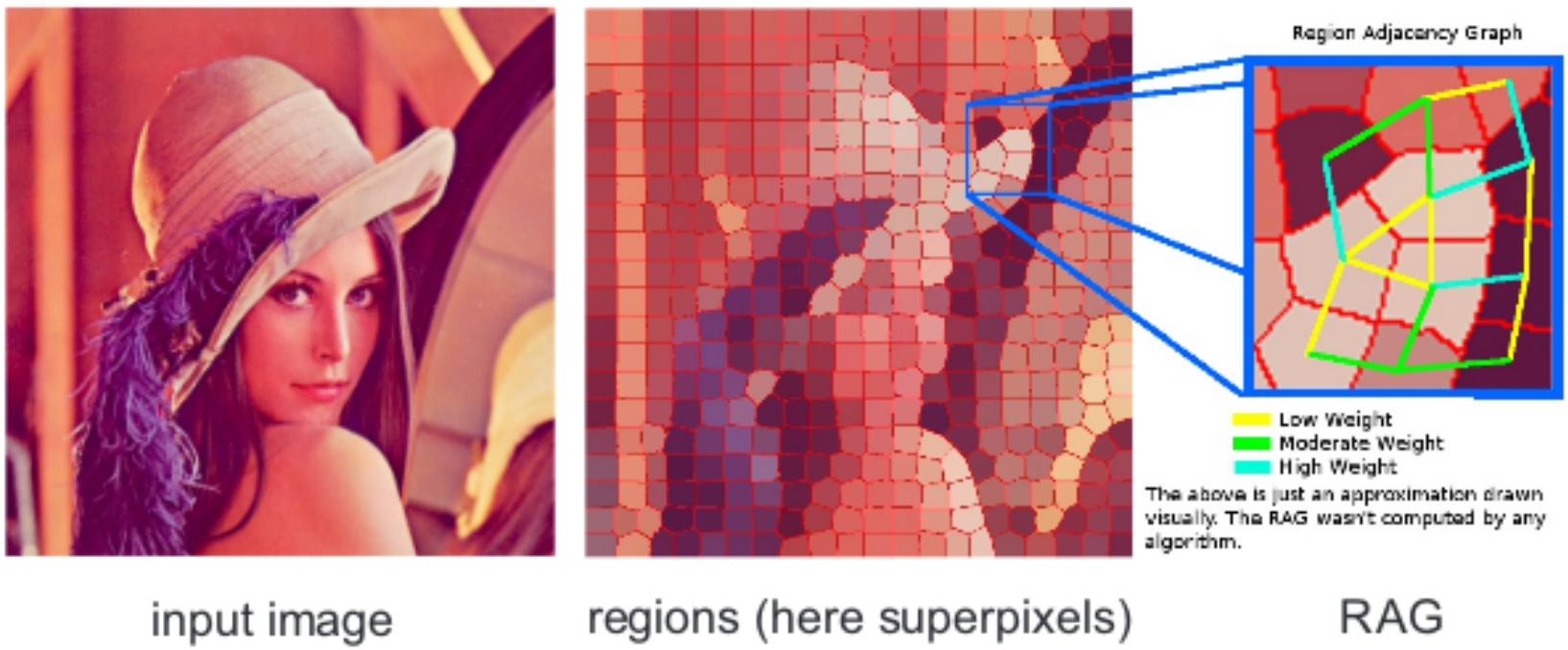
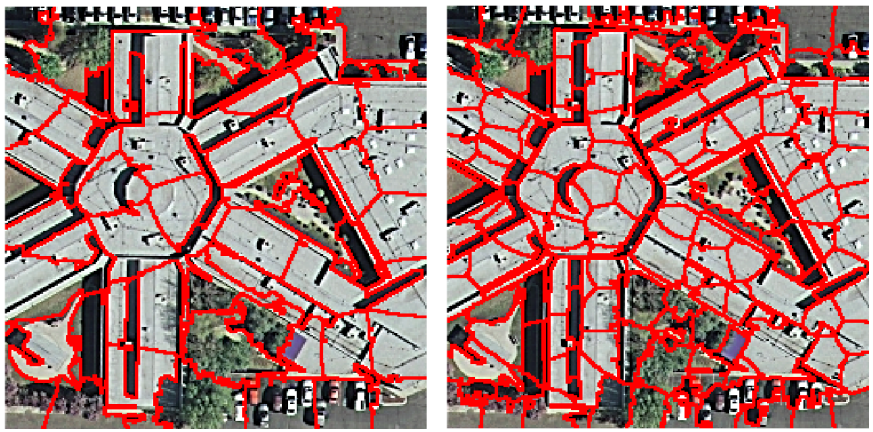


Figure: Region adjacency graph (RAG) construction.

Segmentation and Region Adjacency Graph

- Segmentation using SLIC superpixels (different region sizes).



(a) 50 (b) 150

- Contruct RAG.
- RAG for each image is different.
- Graph laplacian being different, must use spatial domain GCN instead of spectral domain.

Attributed Region Adjacency Graph

• Segment image x_i into n_i non-overlapping homogeneous super-pixel regions $\{r_i^1, r_i^2, \cdots, r_i^{n_i}\}$, using SLIC.

1. **Node:** Hand-crafted 356-d Shape, color, texture features.
2. **Weighted adjacency matrix:**
   - Edges exist between pairs of vertices if the corresponding regions are adjacent to each other; edge weight $E_i(r_i^j, r_i^k)$:

$$G_i(r_i^j, r_i^k) = \alpha_1 ||c_{r_i^j} - c_{r_i^k}||_2 + \alpha_2 |\theta_{r_i^j} - \theta_{r_i^k}|$$

i — image; j, k — nodes;

   - $c_{r_i^j}$ -centroid pixel of the j^{th} region. $\theta_{r_i^j}$ is the orientation angle (angle between horizontal axis and the major axis of the ellipse and $\theta \in [-90°, +90°]$).

Siamese Graph Networks (SGCN)

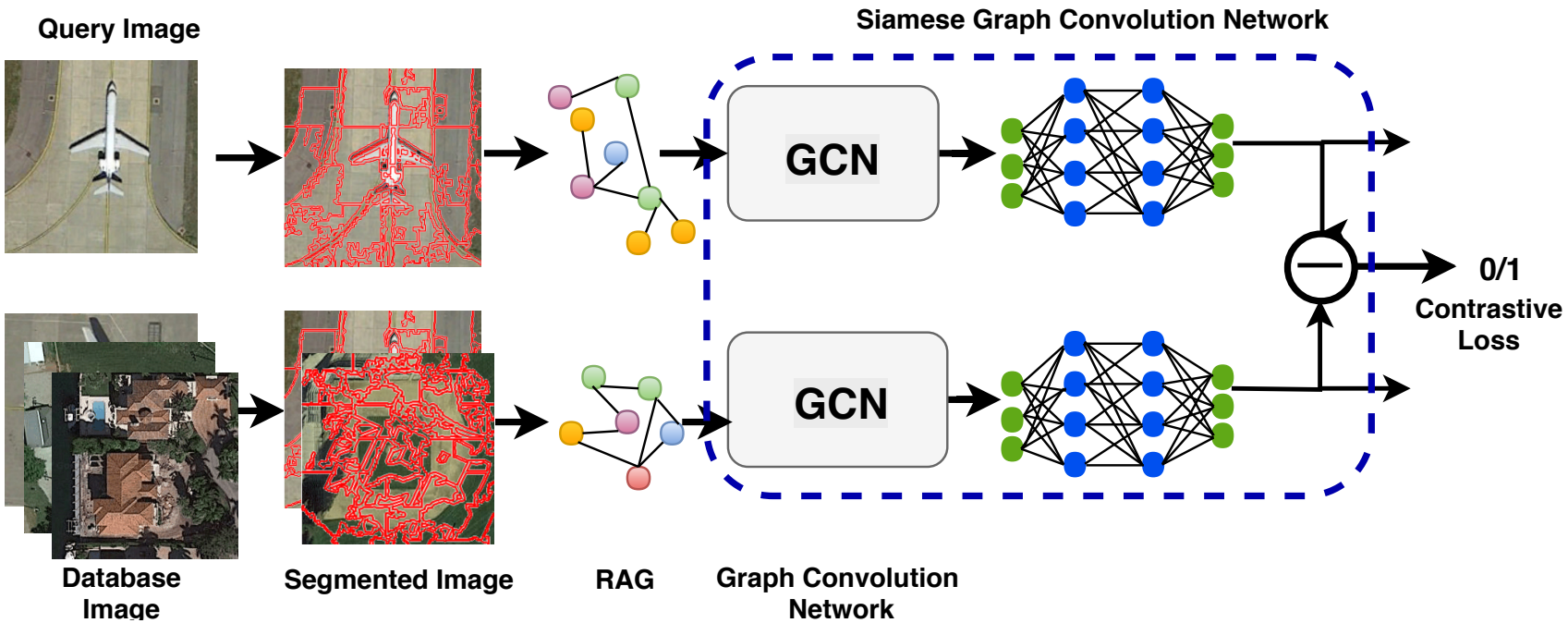


Figure: Pipeline of the proposed SGCN network. The SGCN layer is trained like a siamese twin pair with two images at a time, and the contrastive loss is minimized.

Siamese Graph Networks (SGCN)

- To learn an embedding where images from similar classes are forced to reside closely while different classes are pushed apart from each other.
- Contrastive loss is optimized:

$$\mathcal{L}^k = \sum_{n=1}^{N} \Big(t_{nij} \left\{ \max\left(0, m - (\widehat{x}_{ni} - \widehat{x}_{nj})^2\right) \right\} + (1 - t_{nij})(\widehat{x}_{ni} - \widehat{x}_{nj})^2 + \lambda_1 \left(|\widehat{x}_{ni}|_2 + |\widehat{x}_{nj}|_2 \right) + \lambda_2 |\phi|_2 \Big)$$

- Weights and feature norms are used for regularization.
- Positive class: $t_{ij} = 0$; Negative class: $t_{ij} = 1$.
- Ensure avoiding class-imbalance problem.

Selection of positive and negative exempler pairs

- **Hard negatives:** For the negative pair selection, images from a different class which lie closest, as well as the image which lie farthest from the other pair image are chosen randomly.

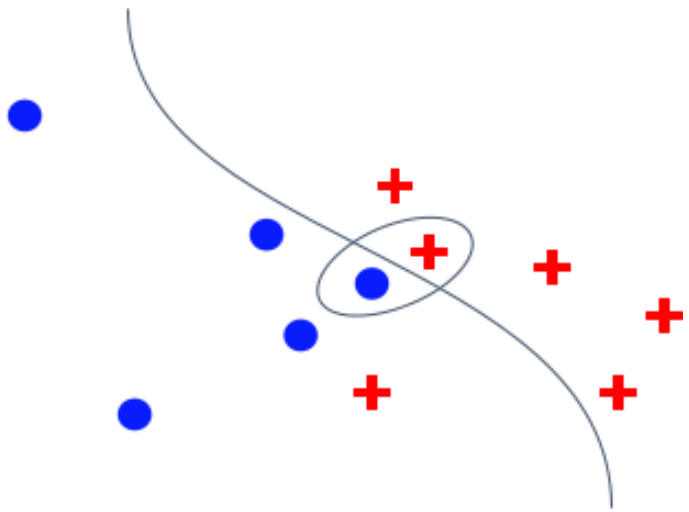


Image Retrieval:

- k-nearest neighbours are then found for a given query image x^{query} from all the classes.

Experiments and Results

Datasets:

1. UC Merced 21 land use classes, each contain 100 images.
2. PatternNet 38 land use classes, each contain 800 images.

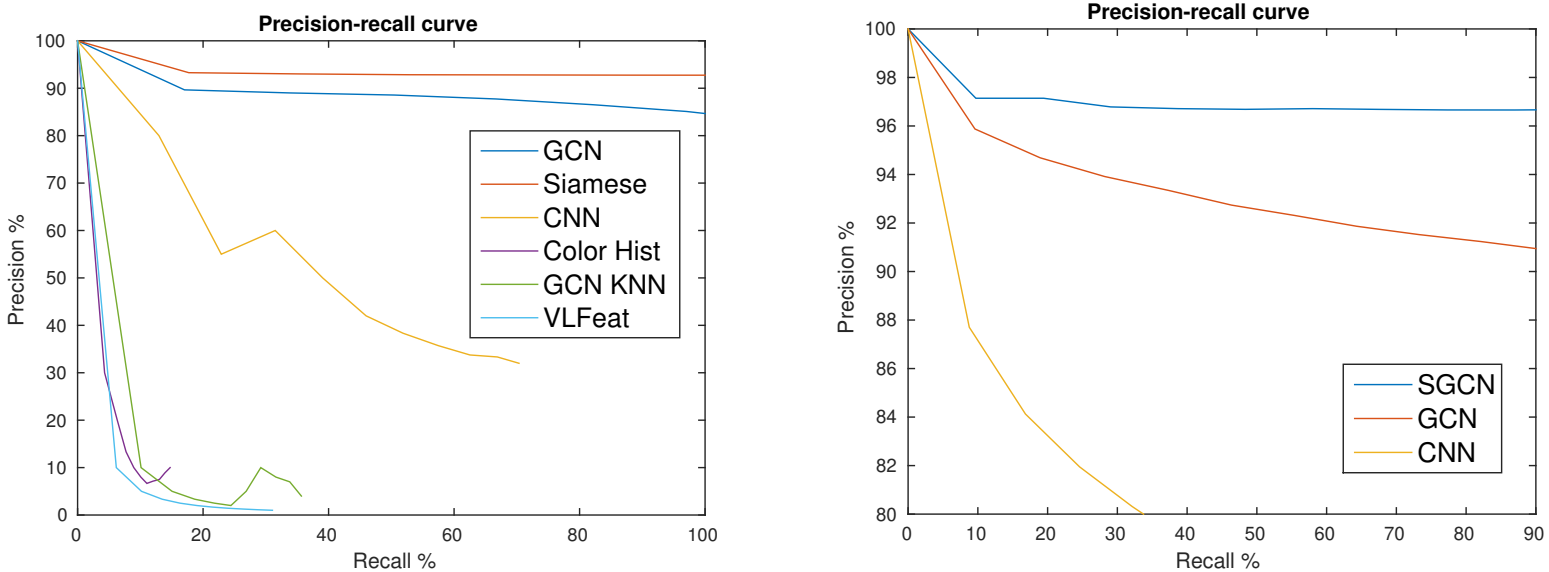
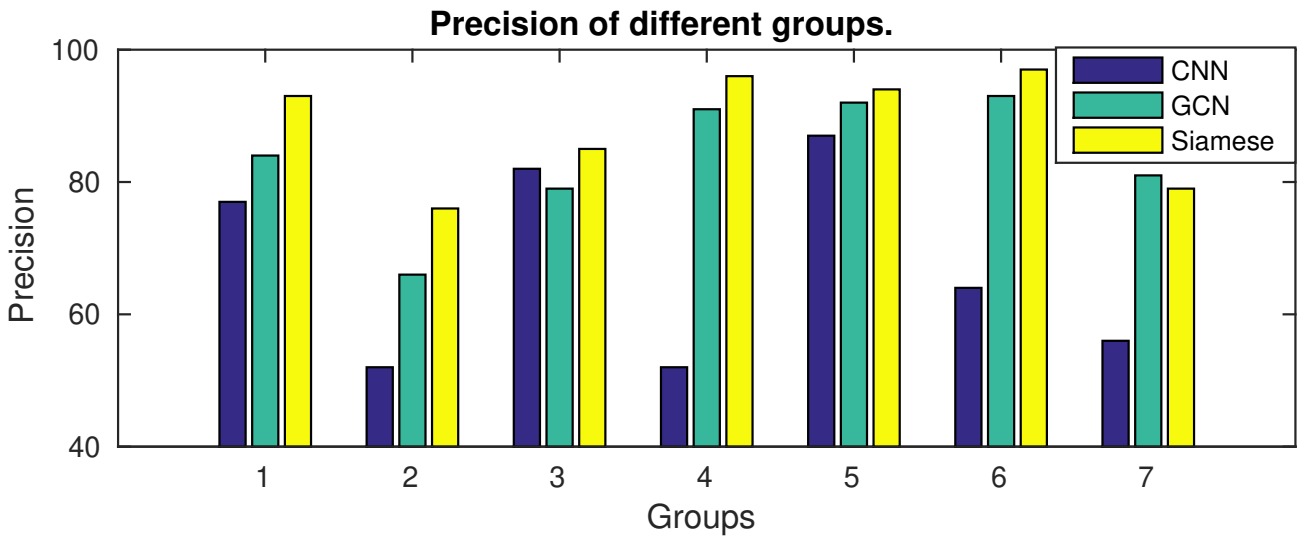


Figure: Precision recall curve of various methods on (a) UCMerced, (b) PatternNet dataset.

Experiments and Results

Model	UC Merced			PatternNet		
	ANMRR	mAP	P@10	ANMRR	mAP	P@10
G-KNN	0.92	7.50	10.12	0.88	12.35	13.24
RAG-KNN	0.75	26.74	24.90	0.69	22.56	37.70
VGG-VD16	0.38	53.71	78.34	0.33	59.86	92.04
VGG-VD19	0.39	53.19	77.60	0.34	57.89	91.13
GoogLeNet	0.39	53.13	80.96	0.29	63.11	93.31
GCN	0.33	64.81	87.12	0.28	73.11	95.53
SGCN	**0.30**	**69.89**	**93.63**	**0.21**	**81.79**	**97.14**



Experiments and Results

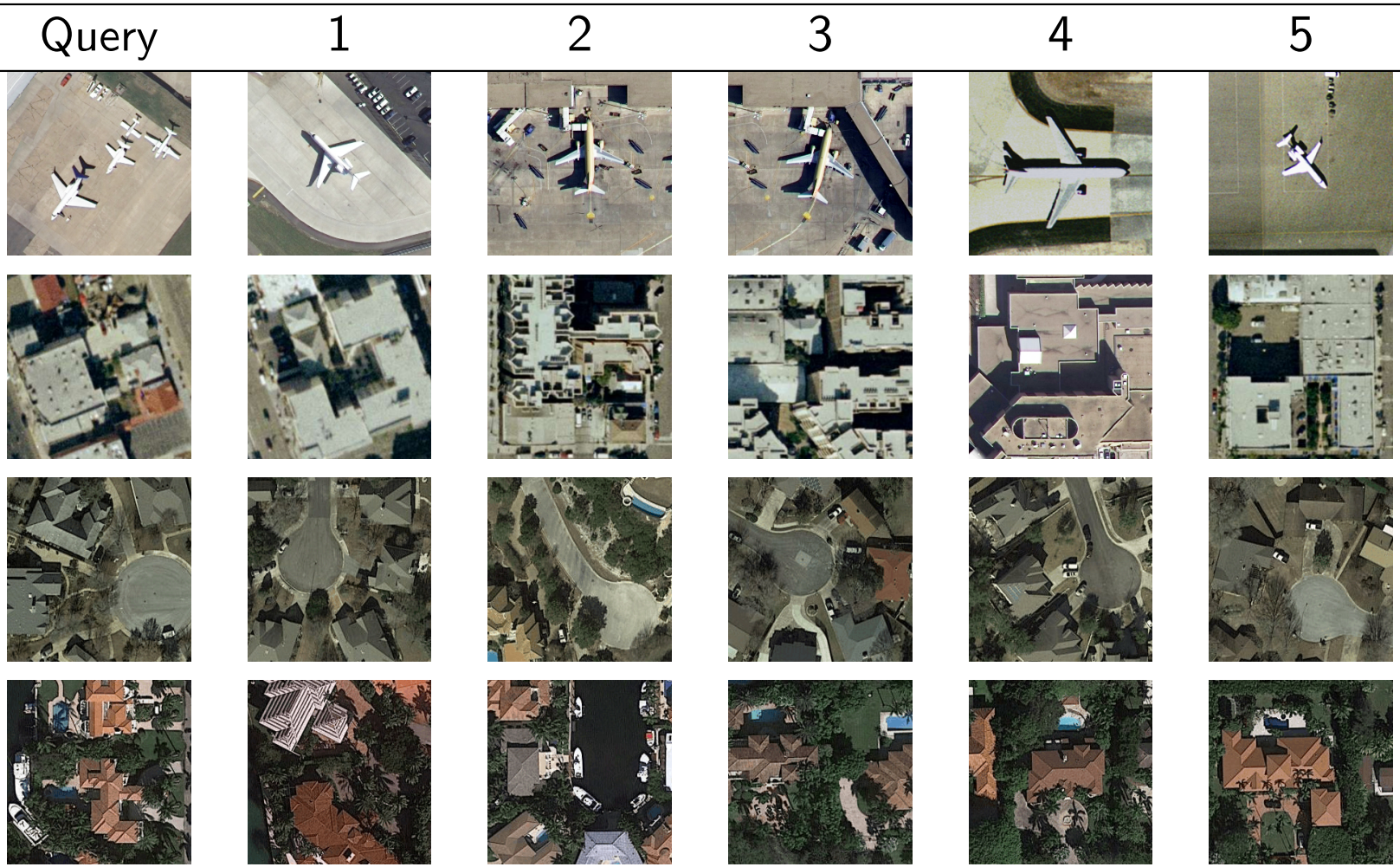
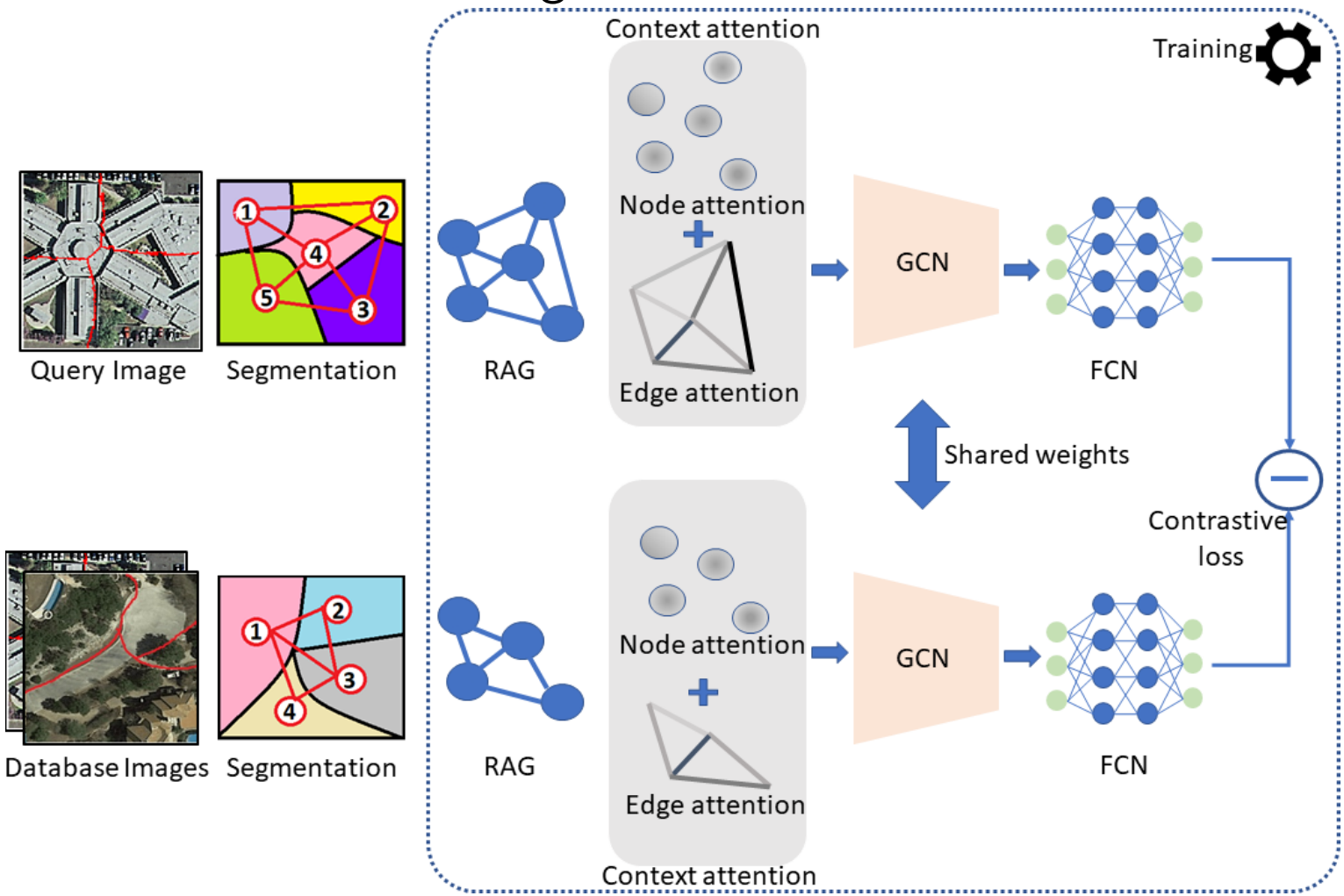


Figure: Query image versus the retried images from UC Merced and PatternNet. `Airplane`, `Buildings`, `ClosedRoad` and `CoastalMansion` classes.

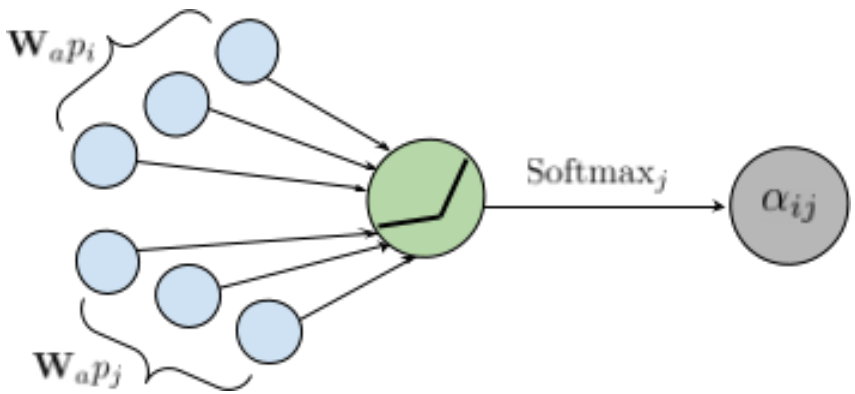
2. Context Attention

Node attention + Edge attention = Context-attention



Existing Node Attention

- **Image:** $p_i \times F$ node features of 356-d.
- **Input:** $\mathbf{p} = \{p_1, p_2, ..., p_R\}$, where R is the number of nodes.
- **Output:** node matrix of **p'** $= \{p_1', p_2', ..., p_R'\}$.
- To learn an attention $a : \mathbb{R}^{F'} \times \mathbb{R}^{F'} \mapsto \mathbb{R}$, train a learnable weight matrix $w_a \in \mathbb{R}^{F' \times F}$, for every node.

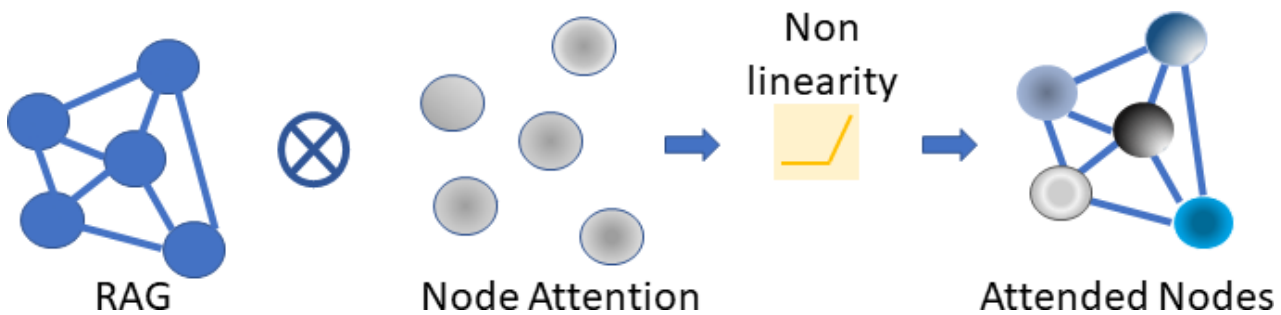


Existing Node Attention

- **Attention layer**: single layer feed-forward neural network, activated with a non-linearity LeakyReLU(·).
- **LeakyReLU** provides a soft attention on the input node features.

$$\alpha_{ij} = \frac{\exp(\text{LeakyRelu}(a[\mathbf{W}_a p_i \parallel \mathbf{W}_a p_j]))}{\sum_{k \in \mathcal{Z}_i} \exp(\text{LeakyRelu}(a[\mathbf{W}_a p_i \parallel \mathbf{W}_a p_j]))}$$

- Assigns higher importance to most important features; lower priority on the less important node features.



Proposed Edge Attention

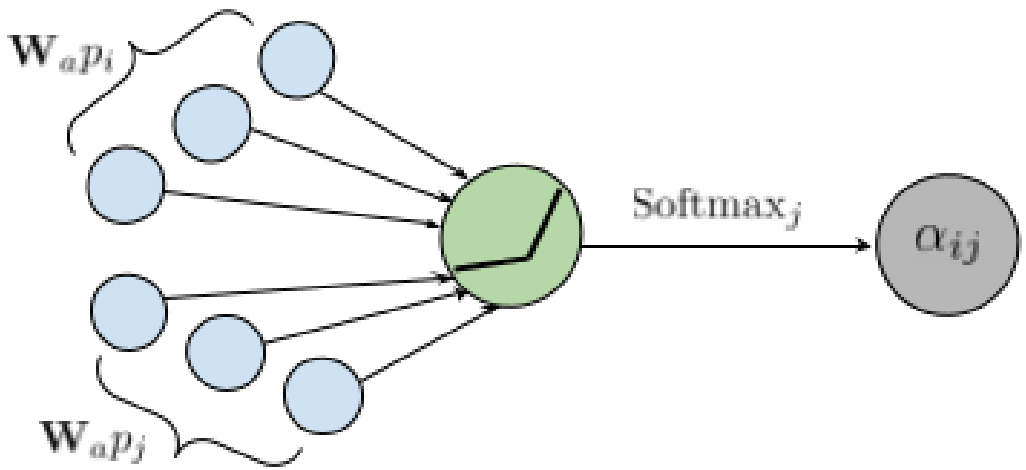
- **Image** p_i nodes, weighted-adjacency matrix: $p_i \times p_i$.
- **Edge matrix** $b : \mathbb{R}^{p_i} \times \mathbb{R}^{p_i} \mapsto \mathbb{R}$.
- To lean attention: $b : \mathbb{R}^{p_i} \times \mathbb{R}^{p_i} \mapsto \mathbb{R}$.

The importance of j^{th} edge to i^{th} edge is then given by:

$$h_{ij} = b(\mathbf{W}_e p_i, \mathbf{W}_e p_j)$$

Attention heads comparable across different edges (Normalize):

$$\beta_{ij} = \frac{\exp(h_i)}{\sum_{i \in \mathrm{R}} \exp(h_i)}$$

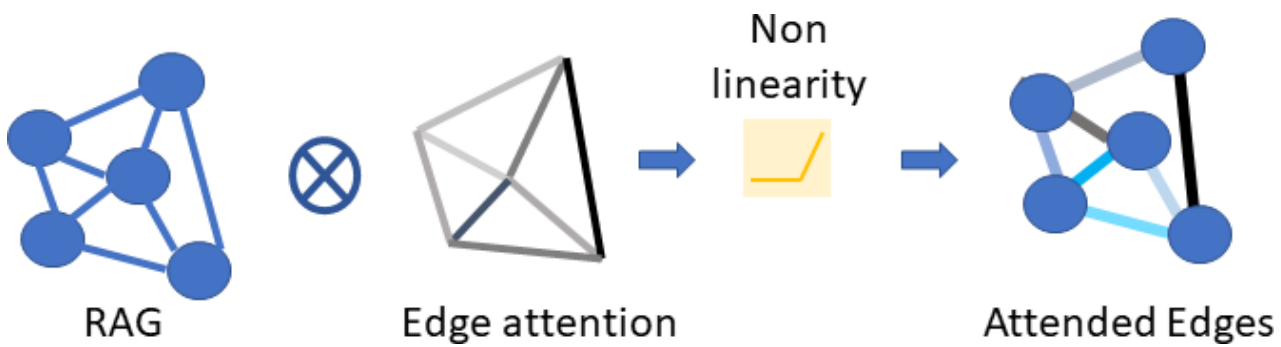


Proposed Edge Attention

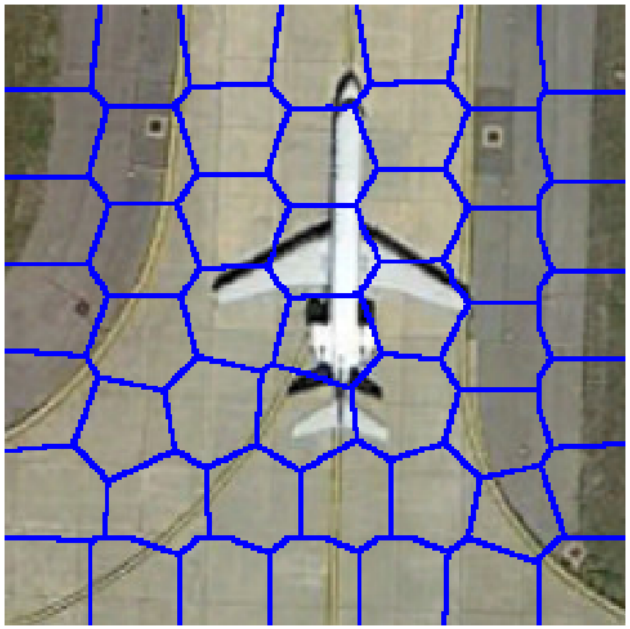
- **Attention layer**: single layer feed-forward neural network, activated with a non-linearity LeakyReLU(·).
- **LeakyReLU** provides a soft attention on the edges.

$$\beta_{ij} = \frac{\exp(\text{LeakyRelu}(h[\mathbf{W}_e p_i \parallel \mathbf{W}_e p_j]))}{\sum_{i \in R} \exp(\text{LeakyRelu}(h[\mathbf{W}_e p_i \parallel \mathbf{W}_e p_j]))}$$

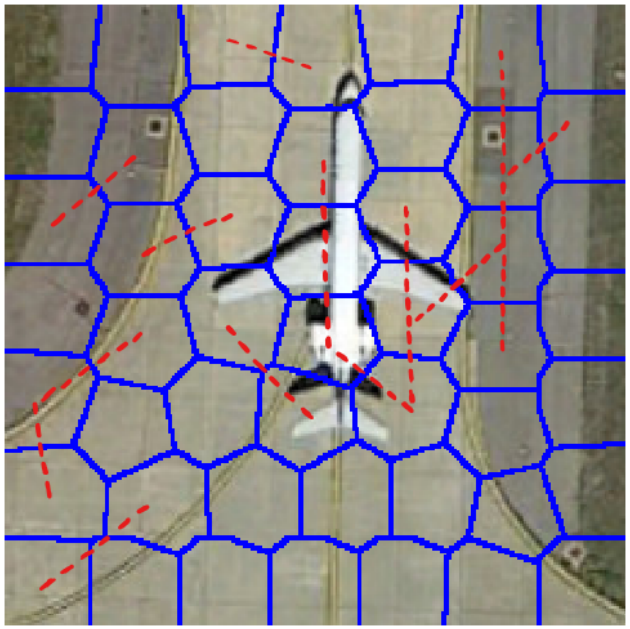
- Assigns higher importance to most important edges; lower priority on the less important edges.



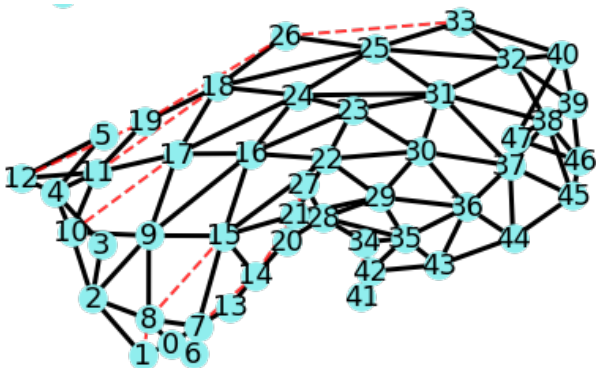
Experiments and Results



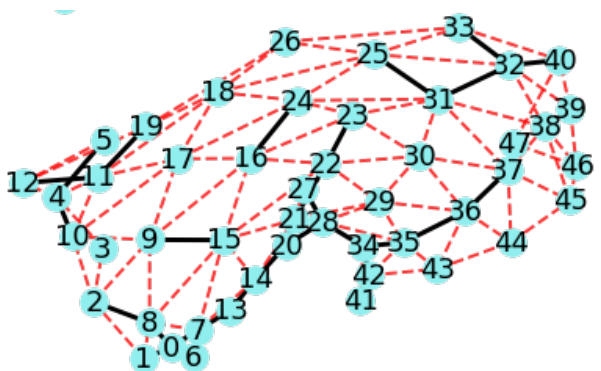
(a) Segmented image.



(b) Attended edges.



(c) Original edge weights.



(d) Attended edge weights.

Summary thus far—

Model	UC Merced			PatternNet		
	ANMRR	mAP	P@10	ANMRR	mAP	P@10
SGCN	0.30	69.89	93.63	0.21	81.79	97.14
Node Attended-SGCN	0.30	70.81	93.86	0.20	83.14	97.76
Edge Attended-SGCN	0.30	71.86	93.89	0.20	82.93	97.71
Context Attended-SGCN	**0.29**	**73.22**	**94.62**	**0.19**	**84.02**	**98.37**

- Demostrated the efficacy of GCN over CNNs in RS scene data.
- Proposed a novel edge-attention mechanism and demonstrated its efficacy in conjugation with a node-attention on GCN frameworks.
- Also proposed a multi-label classification and siamese retrieval architecture of RS data using GCNs.

Exploring Utility of Semantics

For a segmentation type of a problem, localization of sofa:

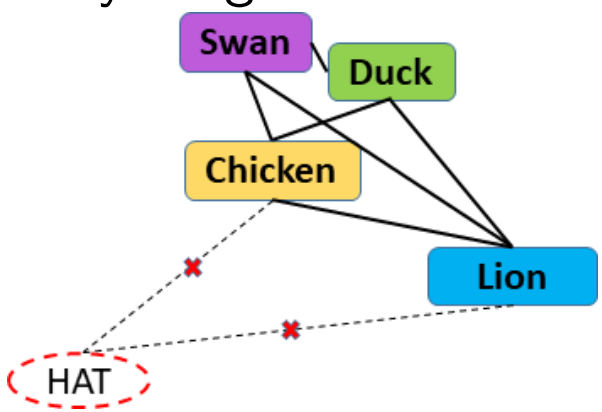
Observations:

- Acc[Image(sofa)] + w2v(sofa)] > Acc[Image(sofa)]
- Acc[Image(cat) + w2v(cat)] > Acc[Image(cat)]]

What about?

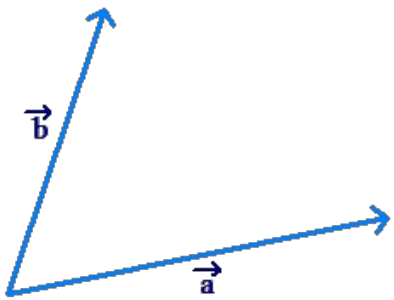
- Acc[Image(sofa)] + w2v(cat)] >? Acc[Image(sofa)]
- Acc[Image(cat) + w2v(sofa)] >? Acc[Image(cat)]]

Observation: Semantics provide additional separability in the feature space. Is there anything sacrosanct about w2v?



Geometric Interpretation

Aim: We want to maximize the distance between the vectors for better separability in the feature space.



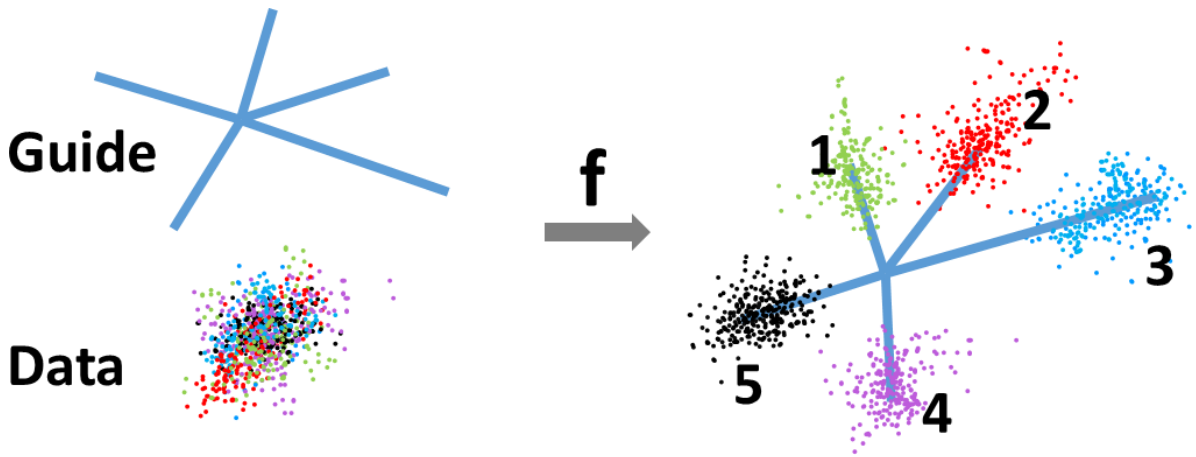
$\vec{a}$ - w2c(cat); $\vec{b}$ - w2v(sofa)

How to construct better separable 'semantic' representation?

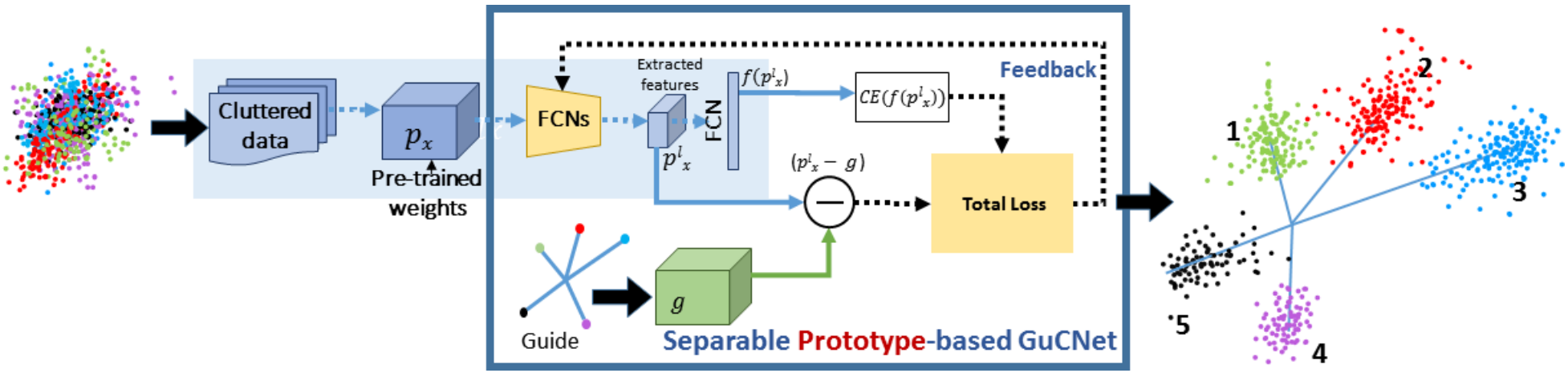
- Select highly separable prototypes as 'semantics'.
  - Use multi-hot vectors and try to increase the Hamming distance among vectors.
  - Else, choose points on a smaller dimension hyper-cube; But NP-complete problem.

Application in classification

- **Guide data ($\mathcal{X}$):** 'Semantic' or prototype vector.
- **Cluttered data ($\mathcal{Y}$):** to be classified.
- Embed class-wise features of the cluttered data to the distinct clusters of the guide data, to make them more separable.



Separable Prototype-based Guiding



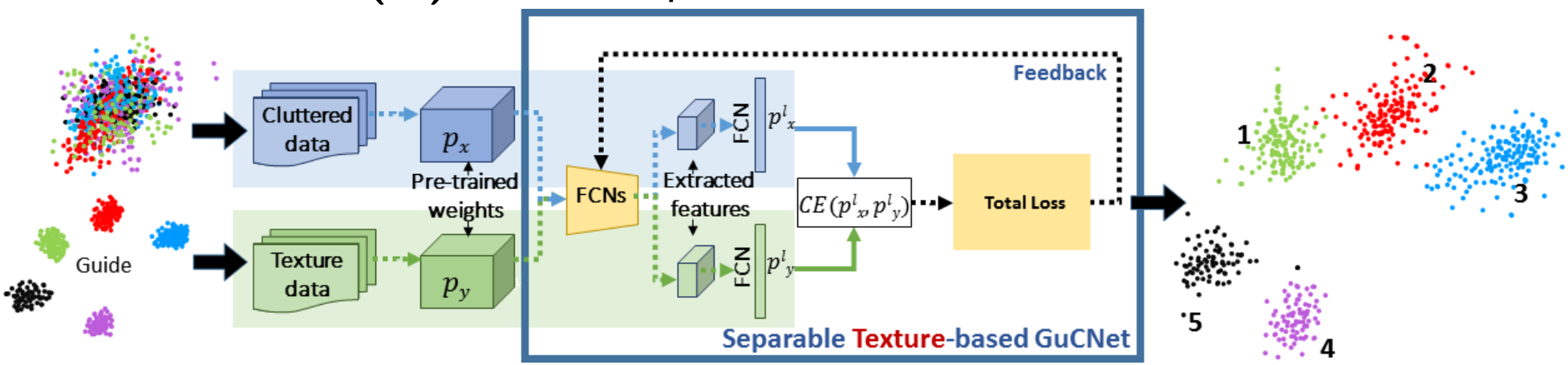
- Extract initial level features from cluttered data using a pre-trained network.
- Choose K ($K \geq C$) dimensional vectors (called *prototypes* g).
- **Matching loss:** ($\mathcal{L}_{\text{ml}} = \mid p^l_x - g \mid$) to bring the dataset closer to the prototype vectors.
- Minimize $\mathcal{L}_{\text{ml}} = \mid p^l_x - g \mid + \text{CE}(p^l_x, p^l_y)$.

Can I borrow your good dataset?

- 'Semantics' doesn't have to be a single point!
- Many well-separable datasets are available.

Leverage well classifiability of any existing dataset.

Guide data ($\mathcal{X}$): A well separable data.



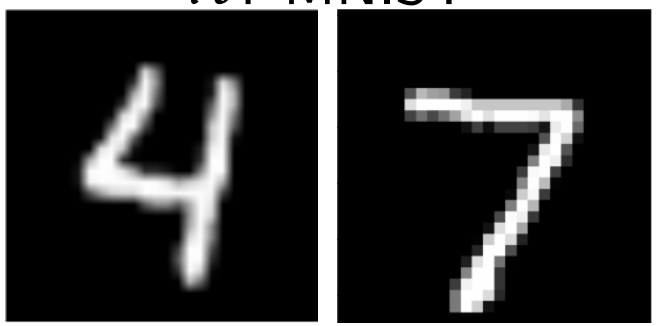
- Extract initial level features from both data using a pre-trained network.
- Feed samples of class-c of both $\mathcal{X}$ and $\mathcal{Y}$ together as the same class label in the unified space.
- Minimize cross-entropy $\mathcal{L}_{\text{CE}} = \text{CE}(p^l_x, p^l_y)$.

Results - RSSCN aerial scene dataset

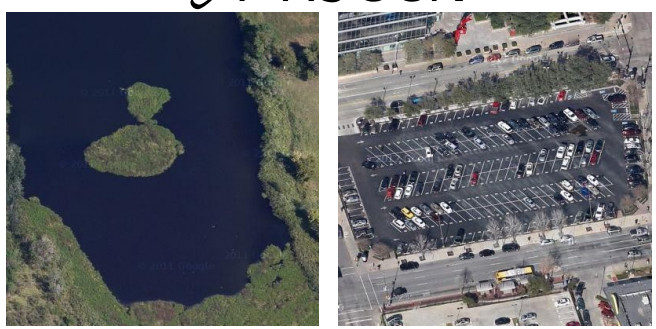
Classification performance of the proposed GuCNet architecture on RSSCN dataset. Here baseline for guide data (MNIST) is 99.80%. Has 7 classes.

Model	**Accuracy(%)**
LLC (CH)	79.94%
SpLSA (SIFT)	79.37%
VLAD (SIFT)	79.34%
RGSIR	81.00%
AlexNet	88.80%
CaffeNet	88.60%
GoogleNet	79.80%
VGG-M	87.30%
VGG-VD16	85.60%
Conv5-MSP5-FV	95.40%
Baseline	88.39%
GuCNet (Prototype)	**97.36%**
GuCNet (Texture)	**99.11%**

$\mathcal{X}$: MNIST



$\mathcal{Y}$: RSCCN

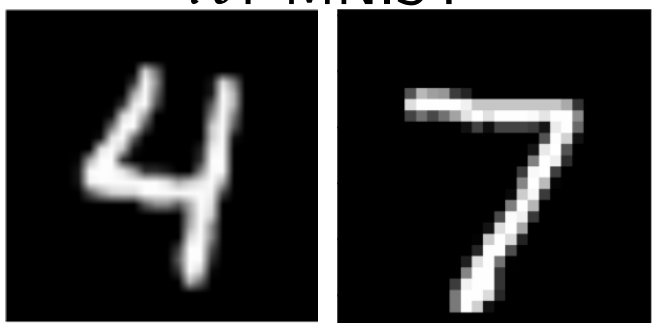


Results - LSUN Outdoor scene dataset

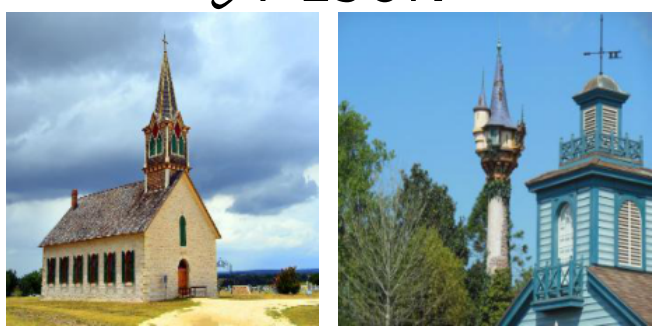
Classification performance of GuCNet on LSUN dataset with the same guide data. Has 10 classes.

Model	**Accuracy(%)**
Vanilla GAN	70.50%
Labeled-samples	77.00%
ds-cube	83.00%
Hybrid GAN	83.20%
Normal BN-Inception+scene n/w	90.40%
Deeper BN-Inception+scene n/w	90.90%
SJTU-ReadSense	90.40%
SIAT MMLAB	91.60%
Baseline	83.75%
GuCNet (Prototype)	**95.03%**
GuCNet (Texture)	**94.86%**

$\mathcal{X}$: MNIST



$\mathcal{Y}$: LSUN



Obtained with just 40% of the entire dataset.

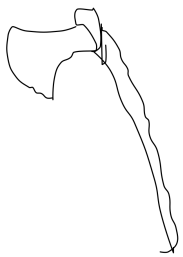
Results - TU-Berlin sketch dataset

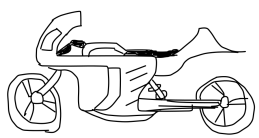
Performance comparison on TU-Berlin dataset for classification accuracy. Here baseline accuracy for guide data is 84.54%. Total 250 classes.

Model	Accuracy(%)
AlexNet-SVM	67.10%
AlexNet-Sketch	68.60%
Sketch-A-Net SC	72.20%
Sketch-A-Net-Hybrid	73.10%
ResNet18-Hybrid	73.80%
Humans	73.10%
Sketch-A-Net-Hybrid	77.00%
Sketch-A-Net	77.00%
Alexnet-FC-GRU	79.95%
Zhang *et. al.*	82.95%
Baseline	69.90%
GuCNet (Prototype)	**86.63%**
GuCNet (Texture)	**89.26%**

$\mathcal{X}$: TU-Berlin Images

$\mathcal{Y}$: TU-Berlin Sketches





Some Interesting Ablation Studies:

Table: Effect of different types of **co-binning** of texture classes from guide set.

Dataset (TU-Berlin)	**Accuracy(%)**
GuCNet (Texture):	
Same class binning	89.26%
Dissimilar class binning	90.05%

Table: Effect of separability of prototypes in terms of Hamming distance (H) on GuCNet performance.

Dataset	**Separation of prototypes**			
	w2v	$H = 2$	$\frac{H_{\max}}{2}$	$H_{\max}$
GuCNet (Prototype):				
RSCCN dataset	96.20%	96.02%	96.27%	97.36%
LSUN dataset	92.71%	94.60%	94.92%	95.03%

Conclusion and future work

- Discussed usefulness of GCNs with attention networks.
- Demonstrated the usefulness of abstracted concept of 'semantics' for improved classification.
- Exploring the possibility of combining the GuCNet with GCN.

Thank You