Affinity Graph Supervision for Visual Recognition

Chu Wang [1] Babak Samari [1] Vladimir G. Kim [2] Siddhartha Chaudhuri [2,3] Kaleem Siddiqi [1*]
[1]McGill University [2]Adobe Research [3]IIT Bombay
{chuwang,babak,siddiqi}@cim.mcgill.ca {vokim,sidch}@adobe.com

Abstract

Affinity graphs are widely used in deep architectures, including graph convolutional neural networks and attention networks. Thus far, the literature has focused on abstracting features from such graphs, while the learning of the affinities themselves has been overlooked. Here we propose a principled method to directly supervise the learning of weights in affinity graphs, to exploit meaningful connections between entities in the data source. Applied to a visual attention network [9], our affinity supervision improves relationship recovery between objects, even without the use of manually annotated relationship labels. We further show that affinity learning between objects boosts scene categorization performance and that the supervision of affinity can also be applied to graphs built from mini-batches, for neural network training. In an image classification task we demonstrate consistent improvement over the baseline, with diverse network architectures and datasets.

1. Introduction

Recent advances in graph representation learning have lead to principled approaches for abstracting features from such structures. In the context of deep learning, graph convolutional neural networks networks (GCNs) have shown great promise [3, 15]. The affinity graphs in GCNs, whose nodes represent entities in the data source and whose edges represent pairwise affinity, are usually constructed from a predefined metric space and are therefore fixed during the training process [3, 15, 22, 28]. In related work, self-attention mechanisms [26] and graph attention networks [27] have been proposed. Here, using pairwise weights between entities, a fully connected affinity graph is used for feature aggregation. In contrast to the graphs in GCNs, the parametrized edge weights change during the training of the graph attention module. More recent approaches also consider elaborate edge weight parametrization strategies [13, 18] to further improve the flexibility of graph structure learning. However, the learning of edge (attention) weights in the graph is entirely supervised by a main objective loss, to improve performance in a downstream task.

Whereas representation learning from affinity graphs has demonstrated great success in various applications [9, 33, 29, 11, 10], little work has been done thus far to directly supervise the learning of affinity weights. In the present article, we propose to explicitly supervise the learning of the affinity graph weights by introducing a notion of target affinity mass, which is a collection of affinity weights that need to be emphasized. We further propose to optimize a novel loss function to increase the target affinity mass during the training of a neural network, to benefit various visual recognition tasks. The proposed affinity supervision method is generalizable, supporting flexible design of supervision targets according to the need of different tasks. This feature is not seen in the related works, since the learning of such graphs are either constrained by distance metrics [13] or dependent on the main objective loss [26, 27, 18].

With the proposed supervision of the learning of affinity weights, a visual attention network [9] is able to compete in a relationship proposal task with the present state-of-the-art [32] without any explicit use of relationship labels. Enabling relationship labels provides an additional 25% boost over [32] in relative terms. This improved relationship recovery is particularly beneficial when applied to a scene categorization task, since scenes are comprised of collections of distinct objects. We also explore the general idea of affinity supervised mini-batch training of a neural network, which is common to a vast number of computer vision and other applications. For image classification tasks we demonstrate a consistent improvement over the baseline, across multiple architectures and datasets. Our proposed affinity supervision method leads to no computational overhead, since we do not introduce additional parameters.

2. Related Work

2.1. Graph Convolutional Neural Networks

In GCNs layer-wise convolutional operations are applied to abstract features in graph structures. Current approaches

*Corresponding author.

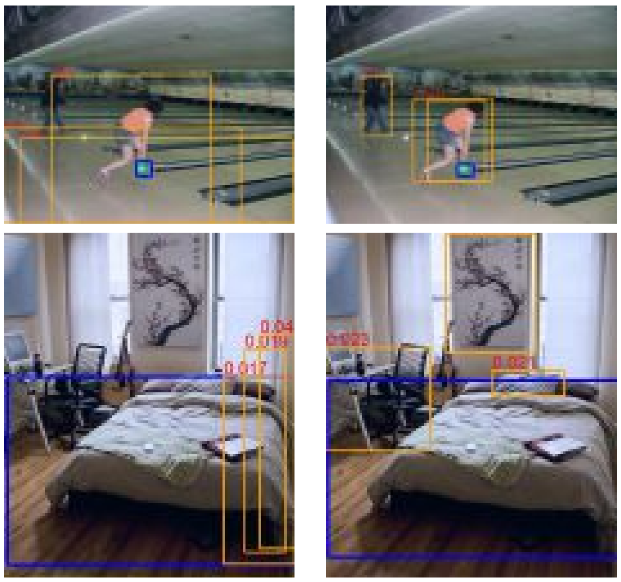


Figure 1: A comparison of recovered relationships on test images, with no relationship annotations used during training. We shows the reference object (blue box), regions with which it learns relationships (orange boxes) and the relationship weights in red text (zoom in on the PDF). Left: baseline visual attention networks [9] often recover relationships between a reference object and its immediate surrounding context. Right: our proposed affinity supervision better emphasizes *potential* relationships between distinct and spatially separated objects.

build the affinity graph from a predefined input [3, 22, 28] or embedding space [7, 26], following which features are learned using graph based filtering in either the spatial or spectral domain. Little work has been carried out so far to directly learn the structure of the affinity graph itself. In this article, we propose a generic method for supervising the learning of pairwise affinities in such a graph, without the need for additional ground truth annotations.

2.2. Visual Attention Networks

Attention mechanisms, first proposed in [26], have been successfully applied to a diverge range of computer vision tasks [9, 33, 29]. In the context of object detection [9], the attention module uses learned pairwise attention weights between region proposals, followed by per region feature aggregation, to boost object detection. The learned attention weights do not necessarily reflect relations between entities in a typical scene. In fact, for a given reference object (region), relation networks [9] tend to predict high attention weights with scaled or shifted bounding boxes surrounding the same object instance (Figure 1).

A present limitation of visual attention networks is their minimization of only the main objective loss during training [9, 33, 29], without any direct supervision of attention between entities. Whereas attention based feature aggregation has been shown to boost performance for general vision tasks [11, 10], the examples in Figure 1 provide evidence that relationships between distinct entities may not be sufficiently captured. In this paper we address this limitation by directly supervising the learning of attention. An affinity graph is first build from the pair-wise attention weights and a novel target affinity mass loss is then applied to guide the learning of attention between distinct objects, allowing the recovery of more plausible relationships.

2.3. Mini-batch Training

The training of a neural network often requires working with mini-batches of data, because typical datasets are too large for present architectures to handle. The optimization of mini-batch training is thus a research topic in its own right. Much work has focused on improving the learning strategies, going beyond stochastic gradient decent (SGD), including [23, 5, 1, 14]. In addition, batch normalization [12] has shown to improve the speed, performance, and stability of mini-batch training, via the normalization of each neuron's output to form a unified Gaussian distribution across the mini-batch.

In the present article we show that our affinity supervision on a graph built from mini-batch features can benefit the training of a neural network. By increasing the affinity (similarity) between mini-batch entries that belong to the same category, performance in image classification on a diverse set of benchmarks, is consistently improved. We shall discuss mini-batch affinity learning in more detail in Section 5.

3. Affinity Graph Supervision

We now introduce our approach to supervising the weights in an affinity graph. Later we shall cover two applications: affinity supervision on visual attention networks (built on top of Relation Networks [9]) in Section 4 and affinity supervision on a batch similarity graph in Section 5.

3.1. Affinity Graph

We assume that there are N entities generated by a feature embedding framework, for example, a region proposal network (RPN) together with ROI pooling on a single image [25], or a regular CNN applied over a batch of images. Let $\mathbf{f}^i$ be the embedding feature for the i-th entity. We define an affinity function $\mathcal{A}$ which computes an affinity weight between a pair of entities m and entity n, as

$$\omega^{mn} = \mathcal{A}(\mathbf{f}^m, \mathbf{f}^n). \tag{1}$$

A specific form of this affinity function applied in attention networks [9, 26] is reviewed in Section 4, and another simple form of this affinity function applied in batch training is defined in section 5.

We now build an affinity graph G whose vertices m represent entities in the data source with features $\mathbf{F}_{in} = \{\mathbf{f}^m\}$ and whose edge weights $\{\omega^{mn}\}$ represent pairwise affinities between the vertices. We define the graph adjacency matrix for this affinity graph as the $N \times N$ matrix $\mathcal{W}$ with entries $\{\omega^{mn}\}$. We propose to supervise the learning of $\mathcal{W}$ so that those matrix entries ω^{mn} selected by a customized supervision target matrix $\mathcal{T}$ will increase, thus gaining emphasis over the other entries.

3.2. Affinity Target $\mathcal{T}$

We now explain the role of a supervision target matrix $\mathcal{T}$ for affinity graph learning. In general, $\mathcal{T} \in \mathbf{R}^{N \times N}$ with

$$\mathcal{T}[i,j] = \begin{cases} 1 & \text{if } (i,j) \in \mathcal{S} \\ 0 & \text{otherwise}, \end{cases} \tag{2}$$

where $\mathcal{S}$ stands for a set of possible connections between entities in the data source.

Target Affinity Mass We would like $\mathcal{W}$ to have higher weights at those entries where $\mathcal{T}[i,j] = 1$, to place emphasis on the entries that are selected by the supervision target. We capture this via a notion of *target affinity mass* $\mathcal{M}$ of the affinity graph, defined as

$$\mathcal{M} = \sum \tilde{\mathcal{W}} \odot \mathcal{T}, \tag{3}$$

where $\tilde{\mathcal{W}} = softmax(\mathcal{W})$ is a matrix-wise softmax operation. A study on affinity mass design is available in the supplementary material.

3.3. Affinity Mass Loss $\mathcal{L}_G$

We propose to optimize the learning of the parameters θ of a neural network to achieve

$$\max_{\theta} \quad \mathcal{M}. \tag{4}$$

Our aim is to devise a strategy to maximize $\mathcal{M}$ with an empirically determined choice of loss form. There are several loss forms that could be considered, including smooth $L1$ loss, $L2$ loss, and a focal loss variant. Defining $x = 1 - \mathcal{M} \in [0,1]$, we define losses

$$L_2(x) = x^2 \tag{5}$$

and

$$\text{smooth}_{L_1}(x) = \begin{cases} x^2 & \text{if} \quad |x| < 0.5 \\ |x| - 0.25 & \text{otherwise}. \end{cases} \tag{6}$$

The focal loss on $\mathcal{M}$ is a negative log likelihood loss, weighted by the focal normalization term proposed in [19], which is defined as

$$\mathcal{L}_G = L_{\text{focal}}(\mathcal{M}) = -(1 - \mathcal{M})^{\gamma} \log(\mathcal{M}). \tag{7}$$

The focal term $(1-\mathcal{M})^{\gamma}$ [19] helps narrow the gap between well converged affinity masses and those that are far from convergence.

Empirically, we have found that the focal loss variant gives the best results in practice, as described in the ablation study reported in Section 6.4. The choice of the γ term depends on the particular tasks, so we provide experiments to justify our choices in Section 6.4.

3.4. Optimization and Convergence of $\mathcal{L}_G$

The minimization of the affinity mass loss $\mathcal{L}_G$ places greater emphasis on entries in $\mathcal{W}$ which correspond to ground truth connections in $\mathcal{S}$, through network training. However, when optimized in conjunction with a main objective loss, which could be an object detection loss $\mathcal{L}_{main} = \mathcal{L}_{det} + \mathcal{L}_{rpn}$ in visual attention networks or a cross entropy loss $\mathcal{L}_{main} = \mathcal{L}_{class}$ in mini-batch training, a balance between $\mathcal{L}_{main}$ and $\mathcal{L}_G$ is required. The total loss can be written as

$$\mathcal{L} = \mathcal{L}_{main} + \lambda \mathcal{L}_G. \tag{8}$$

Empirically, we choose $\lambda = 0.01$ for visual attention networks and for mini-batch training, we choose $\lambda = 0.1$. Figure 5 demonstrates the convergence of the target mass, justifying the effectiveness of using loss $\mathcal{L}_G$ in the optimization of equation 4.

4. Affinity in Attention Networks

We review the computation of attention weights in [26], given a pair of nodes from the attention graph defined in Section 3.1. Let an entity node m consist of its feature embedding, defined as $\mathbf{f}^m$. The collection of input features of all the nodes then becomes $\mathbf{F}_{in} = \{\mathbf{f}^m\}$. Consider node m as a reference object with the attention weight $\tilde{\omega}^{mn}$ indicating its affinity to a surrounding entity node n. This affinity is computed as a softmax activation over the scaled dot products ω^{mn} defined as:

$$\tilde{\omega}^{mn} = \frac{\exp(\omega^{mn})}{\sum_k \exp(\omega^{kn})}, \quad \omega^{mn} = \frac{< W_K \mathbf{f}^m, W_Q \mathbf{f}^n >}{\sqrt{d_k}}. \tag{9}$$

Both W_K and W_Q are matrices and so this linear transformation projects the embedding features $\mathbf{f}^m$ and $\mathbf{f}^n$ into metric spaces to measure how well they match. The feature dimension after projection is d_k. With the above formulation, the attention graph affinity matrix is defined as

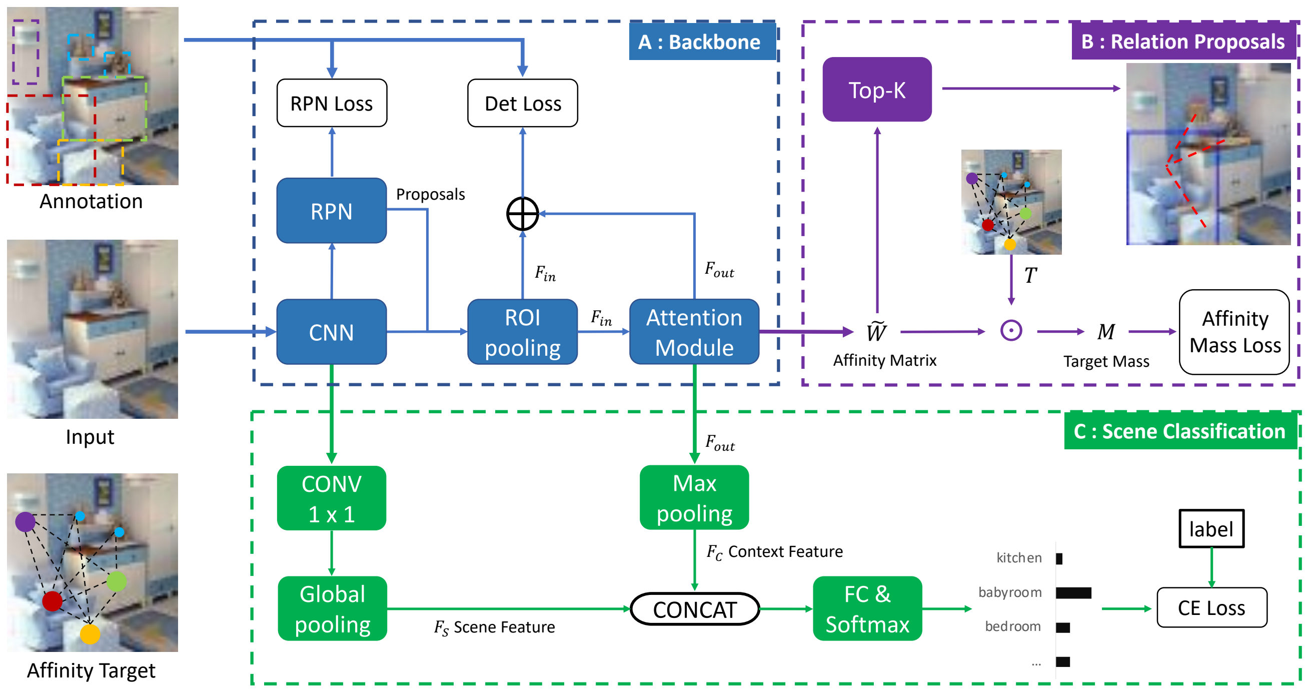


Figure 2: An overview of our affinity graph supervision in visual attention networks, in application to two tasks. The blue dashed box surrounds the visual attention network backbone, implemented according to Relation Networks [9]. The purple dashed box highlights our core component for affinity learning and for relation proposal generation. The green dashed box surrounds the branch for scene categorization. An example affinity target is visualized in the bottom left corner, with solid circles representing ground truth objects colored by their class. The dashed lines between pairs of solid circles give rise to a value of 1 for the corresponding entry in matrix $\mathcal{T}$. See the text in Section 4.1 for a detailed description. A detailed illustration of the attention module is in the supplementary material.

$\mathcal{W} = \{\omega^{mn}\}$. For a given reference entity node m, the attention module also outputs a weighted aggregation of m's neighbouring nodes' features, which is

$$\mathbf{f}^m_{out} = \sum_n \tilde{\omega}^{mn} f^n. \qquad (10)$$

The set of eature outputs for all nodes is thus defined as $\mathbf{F}_{out} = \{\mathbf{f}^m_{out}\}$. Additional details are provided in [26, 9].

4.1. Affinity Target Design

For visual attention networks, we want our attention weights to focus on relationships between objects from different categories, so for each entry $\mathcal{T}[a, b]$ of the supervision target matrix $\mathcal{T}$, we assign $\mathcal{T}[a, b] = 1$ only when:

1. proposal a overlaps with ground truth object α's bounding box with intersection over union > 0.5.

2. proposal b overlaps with ground truth object β's bounding box with intersection over union > 0.5.

3. ground truth objects α and β are two different objects coming from different classes.

Note that NO relation annotation is required to construct such supervision target.

We choose to emphasize relationships between exemplars from different categories in the target matrix, because this can provide additional contextual features in the attention aggregation (equation 10) for certain tasks. Emphasizing relationships between objects within the same category might be better suited to modeling co-occurence. We provide a visualization of the affinity target and additional studies, in the supplementary material. We now discuss applications that could benefit from affinity supervision of the attention weights: object detection, relationship proposal generation, and scene categorization.

4.2. Object Detection and Relationship Proposals

In Figure 2 (part A to part B) we demonstrate the use of attention networks for object detection and relationship proposal generation. Here part A is identical to Relation Networks [9]. The network is end-to-end trainable with detection loss, RPN loss and the target affinity mass loss. In addition to the ROI pooling features $\mathbf{F}_{in} \in \mathcal{R}^{N_{obj} \times 1024}$ from the Faster R-CNN backbone of [25], contextual features $\mathbf{F}_{out}$ from attention aggregation are applied to boost detection performance. The final feature descriptor for the detection head is $\mathbf{F} + \mathbf{F}_c$, following [9]. In parallel, the attention matrix output $\mathcal{W} \in \mathcal{R}^{N \times N}$ is used to generate relationship proposals by finding the top K weighted pairs in the matrix.

4.3. Scene Categorization

In Figure 2 (part A to part C) we demonstrate an application of visual attention networks to scene categorization. Since there are no bounding box annotations in most scene recognition datasets, we adopt a visual attention network (described in the previous section), pretrained on the MSCOCO dataset, in conjunction with a new scene recognition branch (part C in Figure 2), to perform scene recognition. From the CNN backbone, we apply an additional 1×1 convolution layer, followed by a global average pooling to acquire the scene level feature descriptor $\mathbf{F}_s$. The attention module takes as input the object proposals' visual features $\mathbf{F}_{in}$, and outputs the aggregation result as the scene contextual feature $\mathbf{F}_c$. The input to the scene classification head thus becomes $\mathbf{F}_{meta} = concat(\mathbf{F}_s, \mathbf{F}_c)$, and the class scores are output. In order to maintain the learned relationship weights from the pre-trained visual attention network, which helps encode object relation context in the aggregation result $\mathbf{F}_{out}$, we fix the parameters in part A (blue box), but make all other layers in part C trainable.

5. Affinity in mini-Batch

Moving beyond the specific problems of object detection, relationship proposal generation and scene categorization, we now turn to a more general application of affinity supervision, that of mini-batch training in neural networks. Owing to the large size of most databases and limitations in memory, virtually all deep learning models are trained using mini-batches. We shall demonstrate that emphasizing pairwise affinities between entities during training can boost performance for a variety of image classification tasks.

5.1. Affinity Graph

We consider image classification over a batch of N images, processed by a convolutional neural network (CNN) to generate feature representations. Using the notation in Section 3, we denote the feature vectors of this batch of images as $\mathbf{F}_{in} = \{\mathbf{f}^i\}$, where $i \in 1...N$ is the image index in the batch. We then build a batch affinity graph G whose nodes represent images, and the edge $\omega^{mn} \in \mathcal{W}$ encode pairwise feature similarity between node m and n.

Distance Metric. A straightforward $L2$ distance based measure [1] can be applied to compute the edge weights as

$$\omega^{mn} = \mathcal{A}(f^m, f^n) = -\frac{\|f^m - f^n\|_2^2}{2}. \tag{11}$$

[1]More elaborate distance metrics could also be considered, but that is beyond the focus of this article.

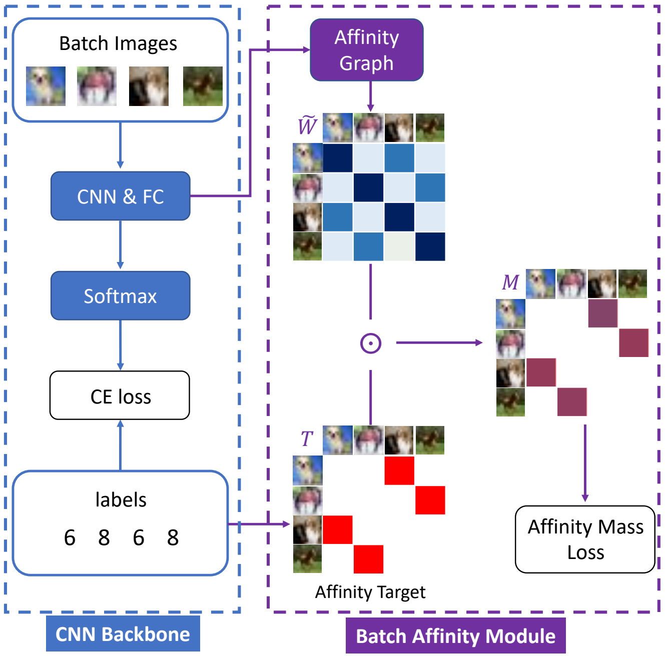


Figure 3: An overview of our affinity graph supervision in mini-batch training of a standard convolutional neural network. Blue box: CNN backbone for image classification. Purple box: Affinity supervision module for mini-batch training. The colored tiles represent entries of the affinity matrix $\tilde{\mathcal{W}}$ and target $\mathcal{T}$, where a darker color denotes a larger numerical value. Minimization of the affinity mass loss aims to increase the value of the purple squares representin entries in mass $\mathcal{M}$ (see equation 3).

5.2. Affinity Target Design

In the mini-batch training setting, we would like feature representations from the same class to be closer to each other in a metric space, with those from different classes being spread apart. To this end, we build the affinity target matrix $\mathcal{T}$ as follows. For each entry $\mathcal{T}[a, b]$ in the matrix, we assign $\mathcal{T}[a, b] = 1$ only when mini-batch node a and b belong to the same category. Thus, the affinity target here selects those entries in $\mathcal{W}$ which represent pairwise similarity between images from the same class. During the optimization of the affinity mass loss (defined in Section 3.3), the network will increase the affinity value from the entries in $\mathcal{W}$ selected by T, while suppressing the other ones. This should in principle leads to improved representation learning and thus benefit the underlying classification task.

5.3. Overview of Approach

A schematic overview of our mini-batch affinity learning approach is presented in Figure 3. Given a batch of N images, we first generate the feature representations $\mathbf{F}_{in}$

from a CNN followed by fully connected layers. We then send $\mathbf{F}_{in}$ to an affinity graph module, which contains a pair-wise distance metric computation followed by a matrix-wise softmax activation, to acquire the affinity graph matrix $\tilde{\mathcal{W}}$. Next, we built the affinity target matrix $\mathcal{T}$ from the image category labels following Section 5.2. An element-wise multiplication with $\tilde{\mathcal{W}}$ is used to acquire the target affinity mass $\mathcal{M}$, which is used in computing the affinity mass loss. During training, the network is optimized by both cross entropy loss $\mathcal{L}_{class}$ and the target affinity loss $\mathcal{L}_G$, using the balancing scheme discussed in Section 3.4.

6. Experiments

6.1. Datasets

VOC07:, which is part of the PASCAL VOC detection dataset [6], with 5k images in trainval and 5k in test set. We used this trainval/test split for model ablation purposes.
MSCOCO: which consists of 80 object categories [20]. We used the 30k validation images for training and the 5k "minival" images for testing, which is common practice [9].
Visual Genome: which is a large relationship understanding benchmark [16], consisting of 150 object categories and human annotated relationship labels between objects. We used 70k images for training and 30K for testing, as in the scene graph literature [31, 30].
MIT67: which is a scene categorization benchmark with 67 scene categories, with 80 training images and 20 test images in each category [24]. We used this official split.
CIFAR10/100: which is a popular benchmark dataset containing 32 by 32 tiny images from 10 or 100 categories [17]. We used the official train/test split and we randomly sampled 10% of train set to form a validation set.
Tiny Imagenet: which is a simplified version of the ILSVRC 2012 image classification challenge [4] containing 200 classes [2] with 500 training images and 50 validation images in each class. We used the official validation set as the test set since the official test set is not publicly available. For validation, we randomly sample 10% of the training set.

6.2. Network Training Details

Visual Attention Networks. We first train visual attention networks [9] end-to-end, using detection loss, RPN loss and affinity mass loss (Figure 2 parts A and B). The loss scale for affinity loss is chosen to be 0.01 as discussed in Section 3.4. Upon convergence, the network can be directly applied for object detection and relationship proposal tasks. For scene categorization, we first acquire a visual attention network that is pretrained on the COCO dataset, and then use the structural modification in Section 6.6 (Figure 2 parts A and C) to fine tune it on the MIT67 dataset. Unless stated otherwise, all visual attention networks are based on a ResNet101 [8] architecture, trained with a batch size of 2 (images), using a learning rate of $5e-4$ which is decreased to $5e-5$ after 5 epochs. There are 8 epochs in total for the each training session. We apply stochastic gradient decent (SGD) with momentum optimizer and set the momentum to 0.9. We evaluate the model at the end of 8 epochs on the test set to report our results.

Mini-batch Affinity Supervision. We applied various architectures including ResNet-20/56/110 for CIFAR and ResNet-18/50/101 for tiny ImageNet, as described in [8]. [2] The CIFAR networks are trained for 200 epochs with a batch size of 128. We set the initial learning rate to 0.1 and reduce it by a factor of 10 at epochs 100 and 150, respectively. The tiny ImageNet networks are trained for 90 epochs with a batch size of 128, an initial learning rate of 0.1, and a factor of 10 reduction at epochs 30 and 60. For all experiments in mini-batch affinity supervision, the SGD optimizer with momentum is applied, with the weight decay and momentum set to $5e-4$ and 0.9. For data augmentation during training, we have applied random horizontal flipping. [3] During training we save the best performing model on the validation set, and report test set performance on this model.

6.3. Tasks and Metrics

We evaluate affinity graph supervision on the following tasks, using the associated performance metrics.
Relationship Proposal Generation. We evaluate the learned relationships on the Visual Genome dataset, using a recall metric which measures the percentage of ground truth relations that are covered in the predicted top K relationship list, which is consistent with [32, 31, 30].
Classification. For the MIT67, CIFAR10/100 and Tiny ImageNet evaluation, we use classification accuracy.
Object Detection. For completeness we also evaluate object detection on VOC07, using mAP (mean average precision) as the evaluation metric [6, 20]. Additional detection results on MSCOCO are in the supplementary material.

6.4. Ablation Study on Loss Functions

We first carry out ablation studies to examine different loss functions for optimizing the target affinity mass $\mathcal{M}$ as well as varying focal terms r, as introduced in Section 3.3. The results in Table 1 show that focal loss is in general better than smooth L1 and L2 losses, when supervising the target mass. In our experiments on visual attention networks, we therefore apply focal loss with $\gamma = 2$, which empirically gives the best performance in terms of recovering relationships while still maintaining a good performance in detec-

[2]The network architectures are exactly the same as those in the original ResNet paper [8].

[3]For the CIFAR datasets, we also applied 4-pixel padding, followed by 32×32 random cropping after horizontal flipping, following [8].

VOC07 Ablation	F-RCNN [25]	RelNet [9]	smooth L1	L2	$\gamma = 0$	$\gamma = 2$	$\gamma = 5$
mAP@all (%)	47.0	47.7 ± 0.1	48.0 ± 0.1	47.7 ± 0.2	47.9 ± 0.2	48.2 ± 0.1	$\mathbf{48.6 \pm 0.1}$
mAP@0.5 (%)	78.2	79.3 ± 0.2	79.6 ± 0.2	79.7 ± 0.2	79.4 ± 0.1	79.9 ± 0.2	$\mathbf{80.0 \pm 0.2}$
recall@5k (%)	-	43.5	60.3 ± 0.3	64.6 ± 0.5	62.1 ± 0.3	$\mathbf{69.9 \pm 0.3}$	66.8 ± 0.2

Table 1: An ablation study on loss functions comparing against the baseline faster RCNN [25] and Relation Networks [9], using the VOC07 database. The results are reported as percentages (%) averaged over 3 runs. The relationship recall metric is also reported with ground truth relation labels constructed as described in Section 4.1, *using only object class labels*.

MIT67	CNN	CNN	CNN + ROIs	CNN + Attn	CNN + Attn + $\mathcal{L}_G$
Pretraining	Imgnet	Imgnet+COCO	Imgnet+COCO	Imgnet+COCO	Imgnet+COCO
Features	$\mathbf{F}_S$	$\mathbf{F}_S$	$\mathbf{F}_S, max(\mathbf{F_{in}})$	$\mathbf{F}_S, \mathbf{F}_C$	$\mathbf{F}_S, \mathbf{F}_C$
Accuracy (%)	75.1	76.8	78.0 ± 0.3	77.1 ± 0.2	$\mathbf{80.2 \pm 0.3}$

Table 2: MIT67 Scene Categorization Results, averaged over 3 runs. A visual attention network with affinity supervision gives the best result (the boldfaced entry), with an improvement over a non-affinity supervised version (4-th column) and the baseline methods (columns 1 to 3). See the text in Section 6.6 for details. F_s, F_c and F_{in} are described in Section 4.3.

tion task. The results in Table 1 serve solely to determine the best loss configuration. Here we do not claim improvement on detection tasks. The results of additional tests using ablated models will be updated in the supplementary material.

6.5. Relationship Proposal Task

Figure 4 compares the relationships recovered on the Visual Genome dataset, by a default visual attention network "baseline" model (similar to [9]), our affinity supervised network with affinity targets built using only object class labels "aff-sup-obj" (see Section 4.1), and an affinity target built from human annotated ground truth relation labels "aff-sup-rel". We also include the reported recall metric from Relationship Proposal Networks [32], which is a state-of-the-art level one-stage relationship learning network with strong supervision, using ground truth relationship annotations. Notably, our proposed affinity mass loss does not require potentially costly human annotated relationship labels for learning (only object class labels were used) and yet it achieves the same level of performance as the present state-of-the-art [32] (the blue curve in Figure 4). When supervised with a target built from the ground truth relation labels instead of the object labels, we considerably outperform relation proposal networks (by 25% in relative terms for all K thresholds) with this recall metric (the red curve).

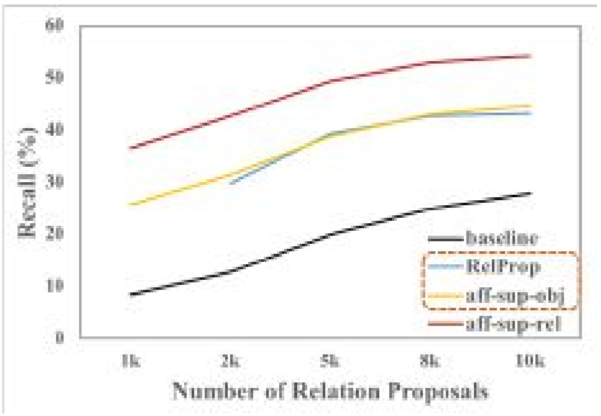


Figure 4: We show the percentage of the true relations that are in the top K retrieved relations, with varying K, in a relation proposal task. We compare a baseline network (black), Relation Proposal Networks [32] (blue), our affinity supervision using object class labels (but no explicit relations) (orange) and our affinity supervision with ground truth relation labels (red). We match the state of the art *with no ground truth relation labels used* (the overlapping blue and orange curves). We out perform the state of the art by a large margin (25% in relative terms) when ground truth relations are used.

6.6. Scene Categorization Task

For scene categorization we adopt the base visual attention network (Figure 2, part A), and add an additional scene task branch (Figure 2, part C) to fine tune it on MIT67, as discussed in Section 4.3. Table 2 shows the results of applying this model to the MIT67 dataset. We refer to the baseline CNN as "CNN" (first column), which is an ImageNet pretrained ResNet101 model directly applied to an image classification task. In the second column, we first acquire a COCO pretrained visual attention network (Figure 2, part A), and fine tune it using only the scene level feature F_S (Figure 2, part C). In the third column, for the same COCO pretrained visual attention network, we concatenate object proposals' ROI pooling features with F_S to serve as meta scene level descriptor. In the fourth and fifth columns, we apply the full scene architecture in Figure 2

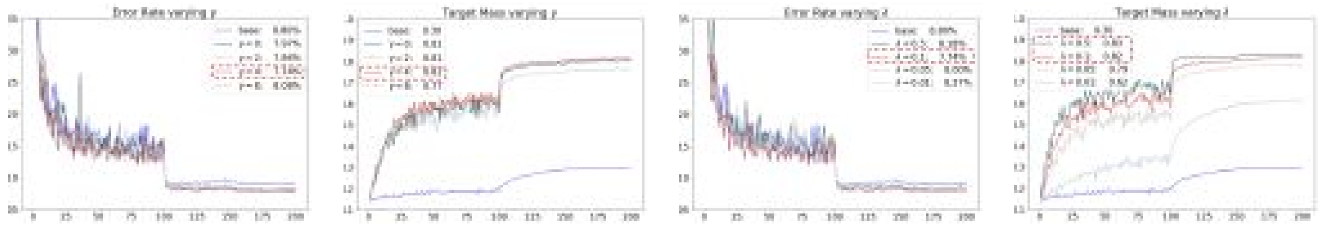


Figure 5: An ablation study on mini-batch affinity supervision, with the evaluation metric on a test set over epochs (horizontal axis), with the best result highlighted with a red dashed box. Left Plots: classification error rates and target mass with varying focal loss' γ parameter. Right Plots: error rates and target mass with varying loss balancing factor λ (defined in section 3.4).

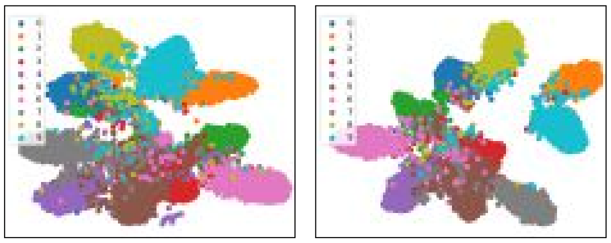


Figure 6: Left: t-SNE plot of learned feature representations for a baseline ResNet20 network on CIFAR10 dataset. Right: t-SNE plot for affinity supervised ResNet20 network.

CIFAR-10	ResNet 20	ResNet 56	ResNet 110
base CNN	91.34 $\pm$ 0.27	92.24 $\pm$ 0.48	92.64 $\pm$ 0.59
Affinity Sup	92.03 $\pm$ 0.21	92.90 $\pm$ 0.35	93.42 $\pm$ 0.38
CIFAR-100	**ResNet 20**	**ResNet 56**	**ResNet 110**
base CNN	66.51 $\pm$ 0.46	68.36 $\pm$ 0.68	69.12 $\pm$ 0.63
Affinity Sup	67.27 $\pm$ 0.31	**69.79 $\pm$ 0.59**	**70.5 $\pm$ 0.60**
Tiny Imagenet	**ResNet 18**	**ResNet 50**	**ResNet 101**
base CNN	48.35 $\pm$ 0.27	49.86 $\pm$ 0.80	50.72 $\pm$ 0.82
Affinity Sup	**49.30 $\pm$ 0.21**	**51.04 $\pm$ 0.68**	**51.82 $\pm$ 0.71**

Table 3: Batch Affinity Supervision results. Numbers are classification accuracy in percentages. CIFAR results are reported over 10 runs and tiny ImageNet over 5 runs.

part C, but with a visual attention network that is pretrained without and with (supervised) target affinity loss, respectively. The affinity supervised case (fifth column) demonstrates a non-trivial improvement over the baseline (first to third columns) and also significantly outperforms the unsupervised case (fourth column). This demonstrates that the attention weights learned solely by minimizing detection loss do not generalize well to a scene task, whereas those learned by affinity supervision can.

6.7. Mini-Batch Affinity Supervision

We conducted model ablation study on γ and λ parameters, introduced in section 3, as summarized in Figure 5. In the subsequent experiments we chose $\gamma = 4$ and $\lambda = 0.1$ based on the ablation plots for error rates in Figure 5.

Convergence of Target Mass. We plot results showing convergence of the target affinity mass during learning in Figure 5. There is a drastic improvement over the baseline target mass convergence, when affinity supervision is enabled. The chosen $\lambda = 0.1$ empirically gives sufficient convergence on Target mass (right-most in Figure 5).

Per-class feature separation. A comparison of t-SNE [21] plots on learned feature representations from 1) baseline CNN and 2) CNN supervised with affinity mass loss is presented in Figure 6. Note that the feature separation between different classes is better in our case.

Results. We now summarize the results for mini-batch affinity learning on CIFAR10, CIFAR100 and TinyImageNet in Table 3. Overall, we have a consistent improvement over the baseline, when using the affinity supervision in mini-batch training. In particular, for datasets with a large number of categories, such as CIFAR100 (100-classes) and tiny ImageNet (200-classes), the performance gain is above 1%. Another advantage of affinity supervision is that we do not introduce any additional network layers or parameters, except for the need for computing the $N \times N$ affinity matrix and its loss. Therefore, the we found the training run-time of affinity supervision very close to the baseline CNN.

7. Conclusion

In this paper we have addressed an overlooked problem in the computer vision literature, which is the direct supervision of affinity graphs applied in deep models. Our main methodological contribution is the introduction of a novel target affinity mass, and its optimization using an affinity mass loss. These novel components lead to demonstrable improvements in relationship retrieval. In turn, we have shown that the improved recovery of relationships between objects boosts scene categorization performance. We have also explored a more general problem, which is the supervision of affinity in mini-batches. Here, in diverse visual recognition problems, we see improvements once again. Given that our affinity supervision approach introduces no additional parameters or layers in the neural network, it adds little computational overhead to the baseline architecture. Hence it can be adopted by the community for affinity based training in other computer vision applications as well.

Acknowledgments

We thank the Natural Sciences and Engineering Research Council of Canada (NSERC) and Adobe Research, for research funding.

References

[1] RMSprop optimizer. http://www.cs.toronto.edu/~tijmen/csc321/slides/lecture_slides_lec6.pdf. Accessed: 2019-11-11. 2

[2] Tiny imagenet visual recognition challenge. https://tiny-imagenet.herokuapp.com/. Accessed: 2019-11-11. 6

[3] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. *NIPS*, 2016. 1, 2

[4] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. *CVPR*, 2009. 6

[5] John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 2011. 2

[6] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The pascal visual object classes (voc) challenge. *IJCV*, 2010. 6

[7] Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. pages 1025–1035, 2017. 2

[8] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. 2016. 6, 10

[9] Han Hu, Jiayuan Gu, Zheng Zhang, Jifeng Dai, and Yichen Wei. Relation networks for object detection. *CVPR*, 2018. 1, 2, 3, 4, 6, 7, 10, 11, 12, 13

[10] Jie Hu, Li Shen, Samuel Albanie, Gang Sun, and Andrea Vedaldi. Gather-excite: Exploiting feature context in convolutional neural networks. *NIPS*, 2018. 1, 2

[11] Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. *CVPR*, 2018. 1, 2

[12] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015. 2

[13] Bo Jiang, Ziyan Zhang, Doudou Lin, Jin Tang, and Bin Luo. Semi-supervised learning with graph learning-convolutional networks. *CVPR*, 2019. 1, 13

[14] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *ICLR*, 2015. 2

[15] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *ICLR*, 2017. 1

[16] Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A Shamma, et al. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *IJCV*, 2017. 6

[17] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. Technical report, Citeseer, 2009. 6

[18] Ruoyu Li, Sheng Wang, Feiyun Zhu, and Junzhou Huang. Adaptive graph convolutional neural networks. *AAAI*, 2018. 1

[19] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. *CVPR*, 2017. 3

[20] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. *ECCV*, 2014. 6

[21] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 2008. 8

[22] Federico Monti, Davide Boscaini, Jonathan Masci, Emanuele Rodolà, Jan Svoboda, and Michael M Bronstein. Geometric deep learning on graphs and manifolds using mixture model cnns. *CVPR*, 2017. 1, 2

[23] Ning Qian. On the momentum term in gradient descent learning algorithms. *Neural networks*, 1999. 2

[24] Ariadna Quattoni and Antonio Torralba. Recognizing indoor scenes. *CVPR*, 2009. 6

[25] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *NIPS*, 2015. 2, 4, 7, 12

[26] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *NIPS*, 2017. 1, 2, 3, 4, 13

[27] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. *arXiv preprint arXiv:1710.10903*, 2017. 1, 13

[28] Chu Wang, Babak Samari, and Kaleem Siddiqi. Local spectral graph convolution for point set feature learning. *ECCV*, 2018. 1, 2

[29] Xiaolong Wang, Ross Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks. *CVPR*, 2018. 1, 2

[30] Danfei Xu, Yuke Zhu, Christopher Choy, and Li Fei-Fei. Scene graph generation by iterative message passing. *CVPR*, 2017. 6

[31] Rowan Zellers, Mark Yatskar, Sam Thomson, and Yejin Choi. Neural motifs: Scene graph parsing with global context. *CVPR*, 2018. 6

[32] Ji Zhang, Mohamed Elhoseiny, Scott Cohen, Walter Chang, and Ahmed Elgammal. Relationship proposal networks. *CVPR*, 2017. 1, 6, 7

[33] Hengshuang Zhao, Yi Zhang, Shu Liu, Jianping Shi, Chen Change Loy, Dahua Lin, and Jiaya Jia. Psanet: Point-wise spatial attention network for scene parsing. *ECCV*, 2018. 1, 2

Supplementary Material

A. Runtime Comparisons

VOC07-Res101	1-epoch Runtime	GPU Memory
Baseline	79.7 minutes	3137 MB
Baseline + $\mathcal{L}_G$	82.3 minutes	3141 MB

CIFAR10-Res110	1-epoch Runtime	GPU Memory
Baseline	34.7 seconds	2117 MB
Baseline + $\mathcal{L}_G$	35.9 seconds	2139 MB

Table 4: Training time (runtime) versus GPU memory consumption between the baseline and our affinity supervised network (denoted as "Baseline + $\mathcal{L}_G$"). For VOC07 we apply RelNet [9] as the baseline. Its affinity supervised version is discussed in Section 4.2 of our article. For CIFAR-10, the baseline is a ResNet-110 network [8] and its affinity supervised version is discussed in Section 5.3 of our article.

We provide efficiency analysis for visual attention networks as well as mini-batch training, with and without the affinity loss. The results are summarized in Table (4), with a ResNet101 structure trained on VOC07 dataset for attention networks and a ResNet110 structure trained on CIFAR10 for mini-batch training. For all experiments reported in Table (4), we used a machine configuration of a single Titan XP GPU, an Intel Core i9 CPU and 64GBs of RAM. The affinity supervision did introduce a small percentage increment to runtime and GPU memory, but the benefits are appreciable, as reflected in the multiple experiments reported in our article.

B. Object Detection Results

As promised in our article, we present the object detection results of affinity supervised attention networks in Table 5. We report results on VOC07 and the COCO split we applied in our article. In both cases we improve upon the baseline and slightly outperform the unsupervised case (similar to Relation Networks [9]). This suggests that relation weights learned using affinity supervision are at least as good as those from the unsupervised case [9], in terms of object detection performance.

Convergence of Target Mass We also provide results showing the convergence of target mass in the context of visual attention networks, in Table (6). It is evident that the affinity mass loss succeeds in optimizing the target mass, when compared with a baseline Relation Network model [9]. This is also indirectly supported by the dramatic improvement in the recall metric, reported in Table 1 of the main article and Table (8), when comparing between the baseline visual attention network and its affinity supervised version.

C. Additional Illustrations

Additional examples of visual relationships, recovered using baseline Relation Networks [9] and its affinity supervised version (discussed in Section 4.2 of our article), are provided in Figure 7. Here we allow all regions with different object class labels to have a potential relation with one another. No human annotated relationship labels are used during training. We include both examples showing improvement and ones where the results are comparable. Whereas the baseline method can be effective at times (Figure (7) right half), the affinity supervision improves its consistency. This claim is also supported by the relationship proposal results on Visual Genome reported in Section 6.5 of the main article.

D. Structure inside Attention Module

A structural overview of the visual attention module used throughout the main article is presented in Figure (8).

E. Supervision Targets

The proposed loss requires a task specific design of a supervision target $\mathcal{T}$. The flexibility in choosing this target is one of the core advantages of our affinity supervision, that is, it can be any user designed matrix target. In fact, in the experiments reported in the main article, we construct this matrix automatically using only object class labels, i.e., no labeled relations are required. In the context of mini-batch training, the target design is straightforward. Here we aim to reduce the within-class feature distances between batch images. Thus, a same-category target $\mathcal{T}$ is adopted. This target increases the similarity metric between same-class connections in the weight matrix $\tilde{\mathcal{W}}$ and because of the matrix-wise softmax activation, connections between instances from different classes are suppressed.

For the case of visual attention networks, various supervision targets can be applied to adapt the method for different downstream applications. In the main draft, our goal is to improve visual recognition using contextual features aggregated by the attention module, with improved object-wise relationship recovery. Thus, the supervision target emphasizes relationships between instances from different categories. However, given a distinct vision task, such as learning human-to-human interaction or human-to-X interaction, the target $\mathcal{T}$ could also be constructed by only selecting human-to-human or human-to-X instance connections, while suppressing other possibilities.

To support the idea that the target $\mathcal{T}$ is adaptive, we provide an exemplar ablation study on VOC07 detection

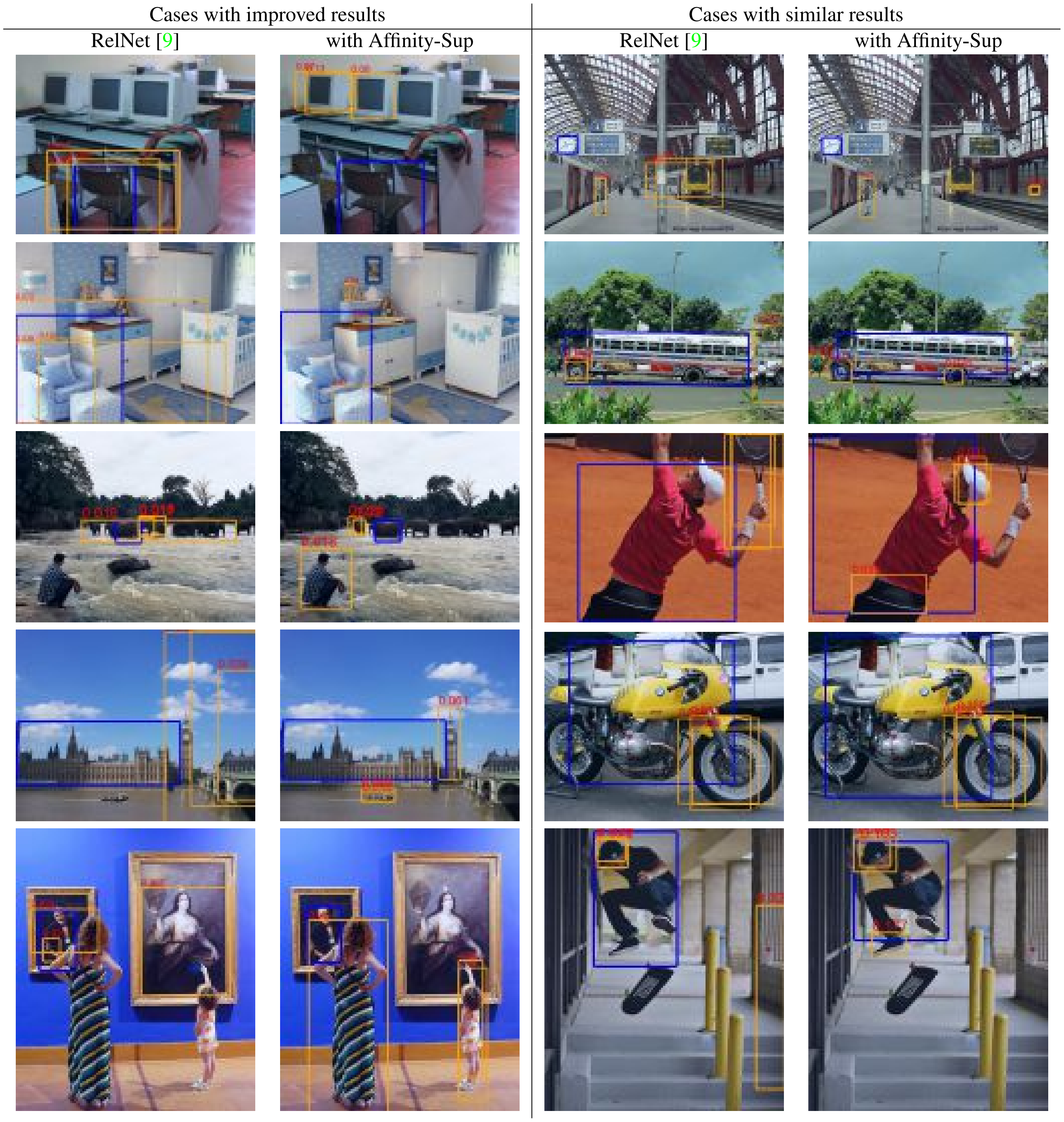


Figure 7: Additional comparisons of recovered relationships on test images, including cases with a clear improvement over the baseline (left) and cases where the results are comparable). The affinity supervision applied to acquire these results do not use human annotated relationship labels. See the text in Figure 1 of the main article for details about the representation. Zoom in on the PDF file to see the attention weight values.

task. We first consider the supervision target proposed in the main draft as **different-category supervision**. We now consider the case where attention between distinct objects belonging to the same category is *also* of interest, leading

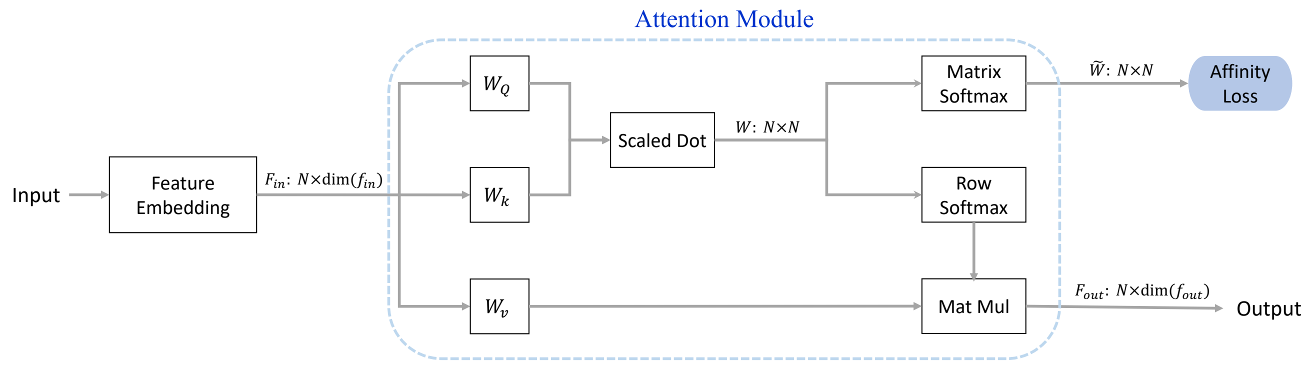


Figure 8: A detailed illustration of the structure inside the "attention module" shown in Figure 2 of the main article. An explanation of each step in this figure is provided at the beginning of Section 4 in the article.

VOC07	base F-RCNN	RelNet [9]	RelNet + $\mathcal{L}_G$
avg mAP (%)	47.0	47.7 ± 0.1	**48.2 ± 0.1**
mAP@0.5 (%)	78.2	79.3 ± 0.2	**79.9 ± 0.2**
mini COCO	base F-RCNN	RelNet [9]	RelNet + $\mathcal{L}_G$
avg mAP (%)	26.8	27.5	**27.9**
mAP@0.5 (%)	46.6	47.4	**47.8**

Table 5: Object Detection Results. mAP@0.5: average precision over a bounding box overlap threshold set to $IOU = 0.5$. avg mAP: averaged mAP over multiple bounding box overlap thresholds. VOC07 experiments are reported over 3 runs, demonstrating stability. $\mathcal{L}_{det}$ stands for detection task loss as defined in [25] and $\mathcal{L}_G$ for the target affinity mass loss defined in Section 3.3 of the main article.

COCO $\mathcal{M}$	Training	Testing
RelNet [9]	0.020	0.013
RelNet + $\mathcal{L}_G$	0.747	0.459

Table 6: We compare target mass values for a visual attention network supervised with (RelNet + $\mathcal{L}_G$) and without (RelNet) the affinity mass loss, using the target constructed in Section 4.1 of our main article. The values reported are evaluated on the COCO split, that is described in the experiment section.

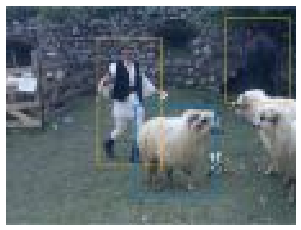
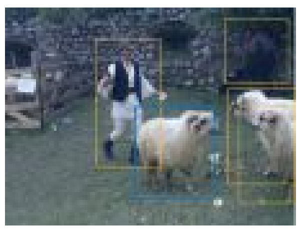
Different-Category	Different-Instance
	

Figure 9: The visualization of supervision targets for attention networks. The blue box indicates a fixed reference object a and the orange boxes indicate the objects b that have a ground truth relationship with a, for which we assign $\mathcal{T}[a, b] = 1$. Left: different category supervision. Note that the sheep in the blue box is *not* related to the other sheep in the image. Right: different instance supervision.

VOC07 varying $\mathcal{T}$	Diff-Instance	Diff-Category
avg mAP (%)	47.6 ± 0.1	48.2 ± 0.2
mAP@0.5 (%)	79.5 ± 0.2	79.9 ± 0.2

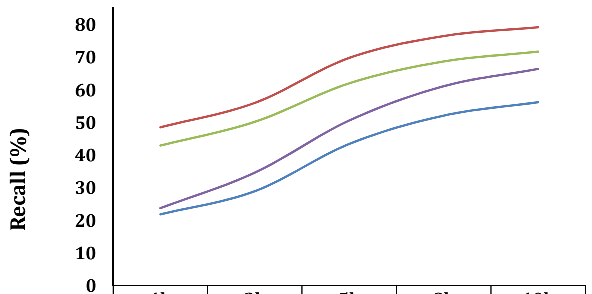
Table 7: Detection results on the VOC07 dataset when varying supervision targets, where we show mean accuracy over 3 runs.

to same-category connections in the target matrix $\mathcal{T}$. We refer to this as **different-instance supervision**. We provide a visual example of the above mentioned supervision targets in Figure 9. In Table 7, we provide object detection results on VOC07 when supervising the affinity graph using different targets.

In summary, the affinity supervision can be adapted to different targets, to achieve various goals or to handle distinct downstream tasks. However, the successful construction of such a target is task dependent.

F. Target Mass Definition

The definition of target mass, as in Section 3.2 of the main article, could have slightly different variations. We defined it as a summation over selected entries, in a matrix-scale. However, it is entirely possible to define such a summation over a row of matrix $\mathcal{W}$, when the softmax activation applied is a row-wise operation. That is we only consider a



	1k	2k	5k	8k	10k
— unsup	21.9	29	43.5	52.2	56.3
— row	23.8	34.7	50.7	61.2	66.5
— mat	43	50.3	62.1	68.8	71.8
— mat-focal	48.6	56.1	69.9	76.6	79.3

Table 8: Evaluating different target mass definitions. Results reported are relationship recall metric with varying top K, using the VOC07 test set. The supervision target $\mathcal{T}$ is constructed following Section 4.1 of main draft.

row of matrix $\mathcal{W}$ during the softmax:

$$\tilde{\omega}_{ij} = \frac{\exp \omega_{ij}}{\sum_x \exp \omega_{ix}} \tag{12}$$

We would build the target as we originally proposed, but compute the target mass in a row-wise manner and apply the affinity mass loss over the row-aggregated mass. For a given row i, we define its target mass as $\mathcal{M}_i$, and thus the affinity mass loss $\mathcal{L}_G$ can be written as

$$\mathcal{L}_G = -\sum_i (1 - \mathcal{M}_i)^\gamma \log \mathcal{M}_i. \tag{13}$$

To justify the selected matrix-wise formulation that we proposed in the main draft, we provide the following ablation study, on relationship recovery recall metric using the VOC07 dataset. In the reported results here, we define the aforementioned row-wise target mass formulation as "row" and the matrix-wise version used in the main draft as "mat", but supervised with only the log-loss of $\mathcal{L}_G = -\log \mathcal{M}$. Lastly, to demonstrate the benefit of focal terms in the final loss form, which is

$$\mathcal{L}_G = -(1 - \mathcal{M})^\gamma \log \mathcal{M}, \tag{14}$$

we define the matrix-wise supervision with focal term as "mat-focal". The recall measurement results are summarized in Table (8). We emphasize that recall@k reported here is always based on ranking the affinity weights post the matrix-wise softmax, ensuring fairness of the comparisons.

The results suggest that the affinity weights, when supervised using the affinity mass loss regardless of its form, are better than the unsupervised case (similar to Relation Networks [9]). Between different variations of target mass and loss forms, the choice of matrix-wise formulation with focal term gives the best results.

One can further simplify the definition of target mass to a single entry in matrix $\mathcal{W}$, and use a binary cross entropy loss over the Sigmoid activation of ω_{ij}, which is $p_{ij} = \frac{1}{1+\exp -\omega_{ij}}$. The loss can be written as

$$\mathcal{L}_G = -\sum_{ij} \left[\mathcal{T}_{ij} \log p_{ij} + (1 - \mathcal{T}_{ij}) \log (1 - p_{ij})\right], \tag{15}$$

where $\mathcal{T}_{ij}$ simply stands for the i, j-th entry of target matrix $\mathcal{T}$. Within multiple trials of a wide range of choices for the λ defined in Section 3.4 of the main article, we found that this single entry based formulation does not converge to a sufficiently large target mass value and the recall metric is very close to the baseline unsupervised case, thus these results are inferior to the earlier formulations.

In our loss design, the distinction between matrix softmax in affinity loss and row softmax in feature aggregation is essential, see Figure 8. In affinity learning we care about accurately representing the strength of node-to-node connection. For instance, if a node has weak connection to all its neighbors, its edges should have relatively small weights. Following related work [13, 27, 26], a row-wise softmax is applied in feature aggregation. This ensures a unified scaling of the aggregation result, so that a node with low affinity weights is not suppressed, and one with high weights is not dominant.