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 led to principled approaches for abstracting features from such structures. In the context of deep learning, graph convolutional neural 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, 29]. 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, 34, 30, 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. Our affinity supervision method is generalizable and supports the flexible design of supervision targets according to the needs of particular tasks. This feature is absent in related work, where the learning of affinity graphs is either constrained by distance metrics [13] or is 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 [33] without any explicit use of relationship labels. Enabling relationship labels provides an additional 25% boost over [33] 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.

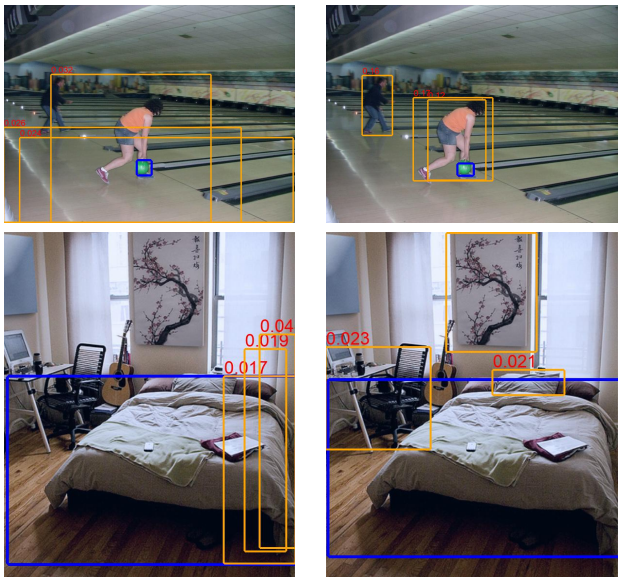

Figure 1: A comparison of recovered relationships on test images, with no relationship annotations used during training. We show the reference object (blue box), regions with which it learns relationships (orange boxes) and the relationship weights in red text (zoom in on the PDF). Left: baseline visual 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, 29] 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 diverse range of computer vision tasks [9, 34, 30]. 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, 34, 30], 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 built 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 descent (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 sim-

ple 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. A study on affinity mass design is in our arXiv version [28].

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 affinity 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:

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

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

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

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

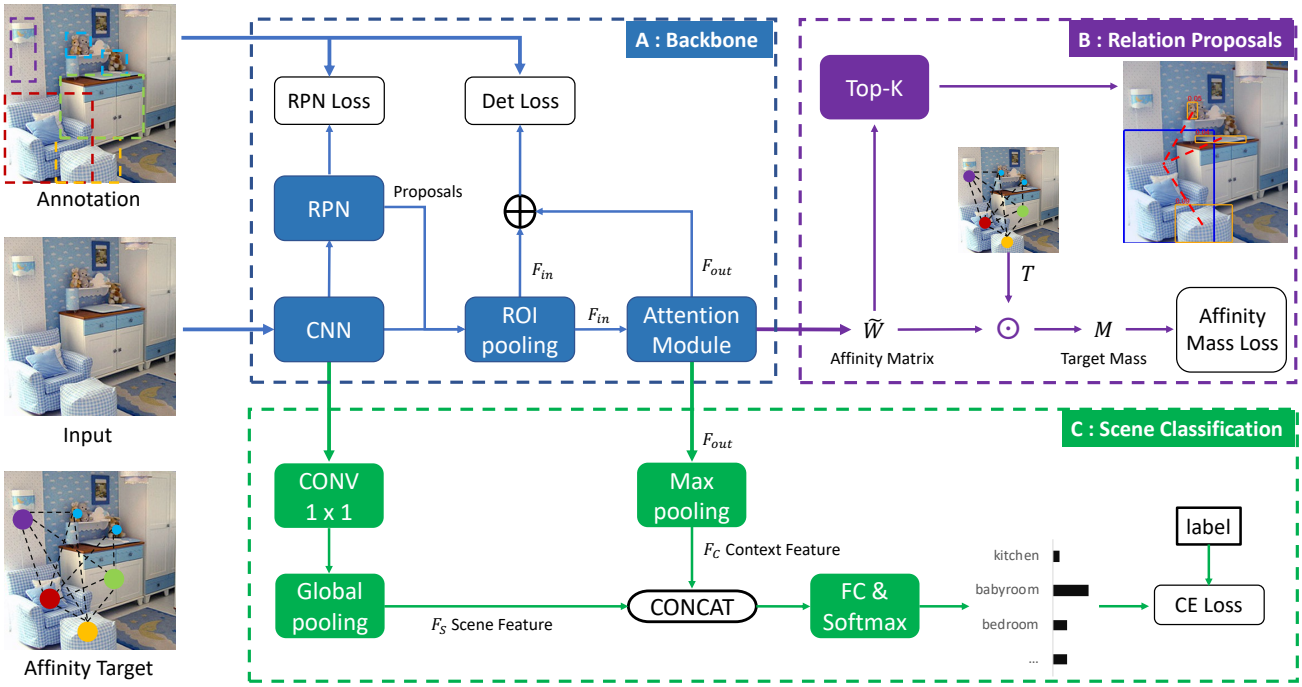

Figure 2: An overview of our affinity graph supervision in visual attention networks, in application to two tasks. The blue dashed box surrounds the visual attention network backbone, implemented according to Relation Networks [9]. The purple dashed box highlights our core component for affinity learning 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 of our arXiv version [28].

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-occurrence. We provide a visualization of the affinity target and additional studies, in the supplementary material of our arXiv version [28]. 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 Training

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

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

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

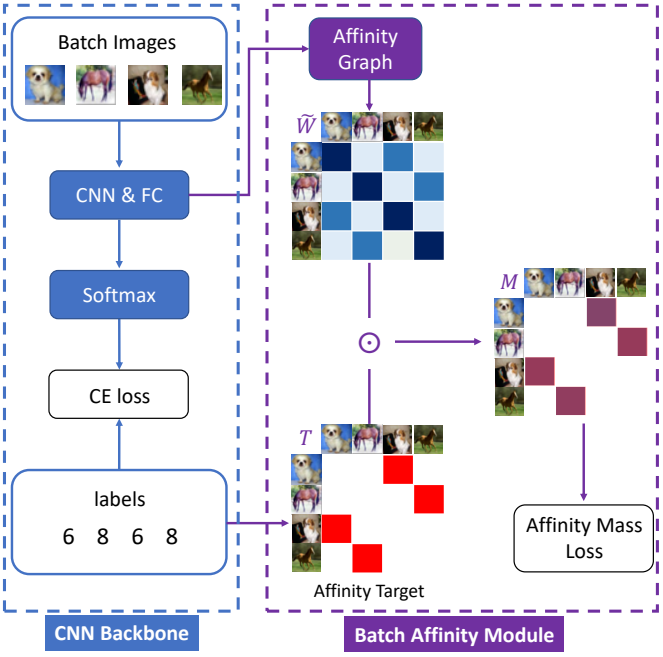

Figure 3: An overview of our affinity graph supervision in mini-batch training of a standard convolutional neural network. Blue box: CNN backbone for image classification. Purple box: Affinity supervision module for mini-batch training. The colored tiles represent entries of the affinity matrix $\tilde{\mathcal{W}}$ 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 representing 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 pairwise 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 30k validation images for training. 5k "minival" images are used for testing, as a 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 [32, 31].
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 each training session. We apply stochastic gradient descent (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]. 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. [2] During training we save the best performing model on validation set, and report its test set performance.

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 [33, 32, 31].
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 detection task. The results in Table 1 serve solely to determine the best loss configuration. Here we do not claim improve-

[2] 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 $\pm$ 0.1	48.0 $\pm$ 0.1	47.7 $\pm$ 0.2	47.9 $\pm$ 0.2	48.2 $\pm$ 0.1	**48.6 $\pm$ 0.1**
mAP@0.5 (%)	78.2	79.3 $\pm$ 0.2	79.6 $\pm$ 0.2	79.7 $\pm$ 0.2	79.4 $\pm$ 0.1	79.9 $\pm$ 0.2	**80.0 $\pm$ 0.2**
recall@5k (%)	-	43.5	60.3 $\pm$ 0.3	64.6 $\pm$ 0.5	62.1 $\pm$ 0.3	**69.9 $\pm$ 0.3**	66.8 $\pm$ 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 $\pm$ 0.3	77.1 $\pm$ 0.2	**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.

ment on detection tasks. The results of additional tests using ablated models are in the arXiv version of this article [28].

6.5. Relationship Proposal Task

Figure 4 compares the relationships recovered on the Visual Genome dataset, by a 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 [33], a state of the art level one-stage relationship learning network with strong supervision, using ground truth relationship annotations. Our affinity mass loss does not require potentially costly human annotated relationship labels for learning (only object class labels were used) and but matches the present state-of-the-art [33] (the blue curve in Figure 4) in performance. When supervised with a target built from the ground truth relation labels instead of the object labels, we outperform relation proposal networks (by 25% in relative terms for all K thresholds) with this recall metric (the red curve).

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

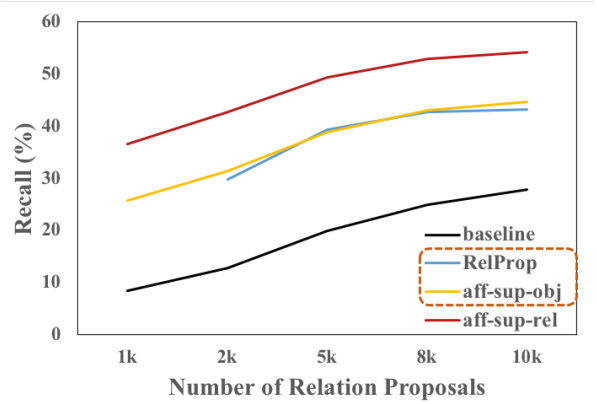

Figure 4: The percentage of the true relations that are in the top K retrieved relations, with varying K, in a relation proposal task. We compare a baseline network (black), Relation Proposal Networks [33] (blue), our affinity supervision using object class labels (but no explicit relations) (orange) 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) and improve on it by a large margin (25% in relative terms) when ground truth relations are used.

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 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

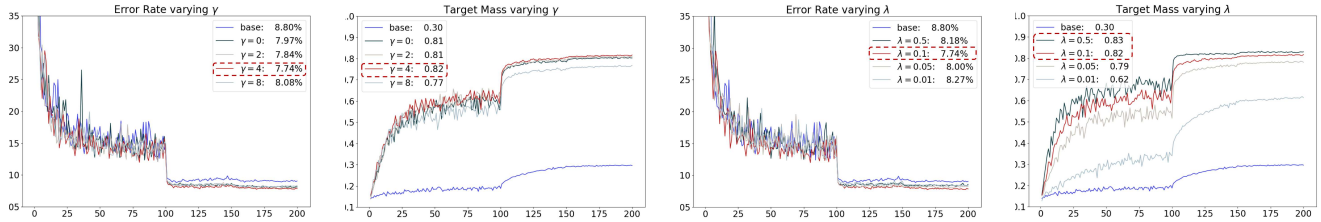

Figure 5: An ablation study on mini-batch affinity supervision, with the evaluation metric on a test set over epochs (horizontal axis), with the best result highlighted with a red dashed box. Left Plots: classification error rates and target mass with varying focal loss' γ parameter. Right Plots: error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

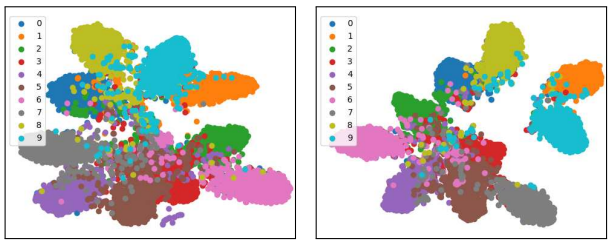

Figure 6: Left: t-SNE plot of learned feature representations for a baseline ResNet20 network on CIFAR10 dataset. Right: t-SNE plot for affinity supervised ResNet20 network.

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

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.

(fourth column). 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 a model ablation study on the γ and λ parameters introduced in Section 3 (summarized in Figure 5) and subsequently chose $\gamma = 4$ and $\lambda = 0.1$ for our experiments, based on the associated error rates.

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 provides acceptable convergence rates (right-most in Figure 5).

Feature Separation Between Classes. A comparison of t-SNE [21] plots on learned feature representations from 1) baseline CNN and 2) a 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 observe a consistent improvement over the baseline, when using the affinity supervision in mini-batch training. For datasets with a large number of categories, such as CIFAR100 (100-classes) and tiny ImageNet (200-classes), the performance gain is above 1%. Affinity supervision does not introduce any additional network layers or parameters beyond the construction of the $N \times N$ affinity matrix and its loss. Hence, training time with affinity supervision is very close to that of the baseline CNN.

7. Conclusion

We have addressed the overlooked problem of directly supervising the learning of affinity graph weights for deep models in computer vision. Our main methodological contribution is the introduction of a novel target affinity mass, and its optimization using an affinity mass loss, which leads to demonstrable improvements in relationship retrieval. We have also shown that the improved recovery of relationships between objects boosts scene categorization performance. Finally, we have 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 shows promise 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

[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] 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

[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

[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

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

[28] Chu Wang, Babak Samari, Vladimir G.Kim, Siddhartha Chaudhuri, and Kaleem Siddiqi. Affinity graph supervision for visual recognition. *arXiv preprint arXiv:2003.09049*, 2020. https://arxiv.org/abs/2003.09049 (visited: 2020-03-27). 3, 4, 7

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

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

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

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

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

[34] 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