Mini-batch Similarity Graphs for Robust Image Classification

Arnab Mondal*
https://mila.quebec/en/person/arnab-mondal/
Vineet Jain*
https://mila.quebec/en/person/vineet-jain/
Kaleem Siddiqi
http://www.cim.mcgill.ca/~siddiqi/

Centre for Intelligent Machines
School of Computer Science
McGill University
Montréal, Canada

Abstract

Current deep learning models for image-based classification tasks are trained using mini-batches. In the present article, we show that exploiting similarity between samples in each mini-batch can significantly boost robustness to input perturbations, an often neglected consideration in the computer vision community. To accomplish this, we dynamically construct a similarity graph from the mini-batch samples and aggregate information using an attention module. Our experiments demonstrate an increase in robustness to local noise and black-box adversarial perturbations, when compared against a baseline model. Our approach also improves performance in diverse image-based object and scene classification tasks, when compared against baseline models and competitive recent methods.

1 Introduction

Supervised deep learning has had wide success in computer vision problems, including in object and scene classification [15, 29] and image segmentation [1, 12]. Models such as residual networks [8] have become standard and are now used as encoders to learn image based representations. In these settings, the training data is divided into mini-batches to accommodate limitations in computational and memory resources. Within a particular mini-batch, the input images may have varying degrees of similarity between them. Exploiting this variability during the feature encoding stage has the potential to improve the performance of downstream computer vision tasks.

A variety of different methods have been proposed to take advantage of the relationships between samples in a mini-batch for computer vision tasks, and in particular for image-based classification. These approaches all explicitly encourage the embeddings in the feature space to be close to one another when the underlying images are similar, by using an additional similarity-based loss term. As an example, [3] uses contrastive learning to encourage pairwise similarity between different augmentations of the same image. In [13]. this approach is extended to a supervised setting, such that the embeddings for instances within the same

* Equal Contribution.

class are nearby in the feature space. In a similar vein, in [24] the learning of representations is supervised by increasing the affinity between mini-batch samples that belong to the same class.

In the present article, we propose a more direct and flexible approach to information aggregation across each mini-batch of images. We construct a graph from each mini-batch of samples to aggregate information across those with similar features, using graph based aggregation and attention. As such, the requirement that similar images have similar embeddings is *implicit*, in that no additional loss term has to be optimized. Features are aggregated in training in a manner that adjusts dynamically to each particular mini-batch ensemble of images. A perturbation analysis explains how this, in turn, affords a degree of robustness to input image alterations and adversarial attacks.

Our experiments demonstrate the robustness of the proposed model against input perturbations and challenging black-box adversarial attacks. They also show a consistent improvement over the baseline and other related approaches across multiple architectures and datasets for the task of image classification. Our proposed mini-batch graph similarity method imposes little computational overhead, since it introduces only a small number of additional parameters for feature aggregation. Since the method is implemented as a modular layer (Figure 1) and training in mini-batches is not specific to image classification, with minor modifications, it can be used for other vision tasks including segmentation [1, 12], region proposal generation [18] and relationship modeling [28]. We include additional sections in the supplementary material exploring the connections of this approach to mini-batch discrimination used to train Generative Adversarial Networks (GANs). *We provide our implementation in the supplementary material.*

2 Related work

The modeling of relationships between samples in a mini-batch has already shown promise in computer vision tasks. In [24], the learning of affinity between samples is supervised by optimizing an affinity mass loss. Here the pairwise affinity between *all* samples in the mini-batch is considered, and the loss function encourages the model to increase the affinity between samples belonging to the same class. A different approach to exploit relationships between samples while training in mini-batches is to use supervised contrastive learning [13]. Here, the normalized embeddings from samples in the same class are encouraged to be closer to one another than to the embeddings of samples from different classes. This approach is related to another self-supervised contrastive method [3], where a model is trained to identify samples in a mini-batch that are different augmentations of the same image. These methods demonstrate improvement in image classification performance over standard networks.

Distinct from affinity graphs [24] and contrastive learning [3, 13], the use of graph based information aggregation in the present article encourages similar images to have similar embeddings in an implicit manner, while removing the need to optimize an additional loss term. Our method develops an extension of Graph Neural Networks (GNN) to mini-batch processing. GNNs were first proposed as deep learning architectures on graph-structured data in [6, 22] and have since been extended to include convolution operation on graphs in [2, 4, 9] or to combined locally connected regions in graphs[17]. The use of GNNs for semi-supervised classification was proposed in [14], following which several different variants of GNN models have been developed: Graph Attention Networks (GATs) [23], models to process graphs with edge information [5, 25] and GNNs that work under low

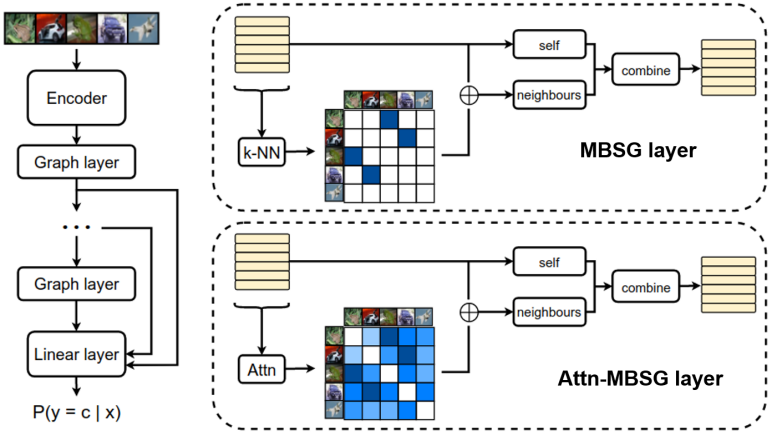

Figure 1: An illustration of the proposed MBSG model for mini-batch learning for image classification. Encoder representations are used to create an adjacency matrix based on k-nearest neighbours. The adjacency matrix defines a similarity graph on a mini-batch of representations, which are then combined based on this graph, as detailed in Section 3.

homophily [30]. GNNs have also been successfully applied to several other computer vision problems, including few-shot learning [19, 21] and semi-supervised learning [31]. In [21], a GNN is used to propagate label information from labeled samples in the support towards the unlabeled query image. In contrast, in [19, 31], a fixed graph is used to propagate embedding and label information, respectively.

3 Mini-batch Similarity Graphs

3.1 Proposed Method

Our network has two components: a feature encoder and a mini-batch similarity graph module (MBSG), as illustrated in Fig. 1. We obtain the encoder $f_\theta(\cdot)$ by removing the final layer of a standard vision network, such as Resnet-50. Consider a typical training setup which takes a mini-batch of samples as input for classification. We denote the input samples in a mini-batch as $\mathcal{X} = \{x_1, x_2, ..., x_B\}$, where B is the batch size. The encoder provides representations for each sample, $h_i^{(0)} = f_\theta(x_i), \forall i \in B$. The MBSG module induces a similarity graph on the set of encoded representations and combines them based on this graph.

We denote the representations in the l^{th} layer by the set $\mathcal{H}^{(l)} = \{h_1^{(l)}, h_2^{(l)}, ..., h_B^{(l)}\}$. To dynamically induce a graph on $\mathcal{H}^{(l)}$, we obtain the adjacency matrix $A^{(l)}$ by computing pairwise cosine similarity between representations and consider the top k similar representations for each sample as its neighbours, removing self connections. The extent of the neighbourhood for each node can be controlled with the parameter k. The layer-wise propagation rule of the MBSG in vector form, is given by,

$$\begin{aligned} \bar{H}^{(l)} &= H^{(l)} W^{(l)} + b^{(l)} \\ H^{(l+1)} &= \sigma\big(\text{combine}\big(\bar{H}^{(l)}, (1/k) A^{(l)} \bar{H}^{(l)}\big)\big). \end{aligned} \tag{1}$$

where $h_i^{(l)}$ is stacked row-wise to form $H^{(l)}$, $W^{(l)}$ is the weight matrix, $b^{(l)}$ is the bias, and $\sigma(\cdot)$ is a non-linear function, usually ReLU. In equation (1), $f_{\text{self}}^{(l)} = \bar{H}^{(l)}$ contains the 'self' information for each node, and $f_{\text{neigh}}^{(l)} = (1/k)A^{(l)}\bar{H}^{(l)}$ contains the 'neighbour' information, since it is based on the average of the encoded representations of the neighbors of each node, as reflected by the adjacency matrix $A^{(l)}$.

There are several methods to combine the self and neighbour information in the combine$(\cdot,\cdot)$ function, which are explained below. We explore each of these in our experiments.

Concatenation: Here the self features, $f_{\text{self}}^{(l)}$ and neighbour features $f_{\text{neigh}}^{(l)}$ are concatenated in the representation dimension. This allows the network the flexibility to learn separate weights for both.

Weighted Addition: This is a convex combination of self and neighbour features, which forces the network to use neighbour information. For $\beta \in [0,1]$, we have,

$$\text{combine}(f_{\text{self}}^{(l)}, f_{\text{neigh}}^{(l)}) = \beta f_{\text{self}}^{(l)} + (1-\beta) f_{\text{neigh}}^{(l)}. \tag{2}$$

This reduces to a standard graph convolution formulation if we set $\beta = 1/(k+1)$. We found that $\beta = 0.5$ provides best results, so all our experiments use this value unless otherwise stated.

Drop Features: We propose a different method to mitigate the effect of co-adaptation and to make the neighbours contribute meaningful information. During training, we drop the self features with probability p and the neighbour features with probability $1-p$. We then make the testing phase deterministic by taking the expected output feature, which leads to a similar expression as the sum combination,

$$\text{combine}(f_{\text{self}}^{(l)}, f_{\text{neigh}}^{(l)}) = p f_{\text{self}}^{(l)} + (1-p) f_{\text{neigh}}^{(l)}. \tag{3}$$

We determined empirically that a value of $p = 0.5$ gave good results, so all our experiments use this value unless otherwise stated. This expression is used during inference to account for the probability-based selection of self or neighbour information, similar to how parameters are scaled when using dropout in neural networks. Whereas this may look similar to weighted addition in appearance, it is in fact completely different.

Attention MBSG The model can be made more expressive by using an attention mechanism while aggregating from neighbours, which we refer to as an Attn-MBSG (see Fig. 1). This is done by changing the calculation of the adjacency matrix $\mathcal{A}^{(l)}$ to incorporate an attention coefficient between nodes. Let $\alpha_{ij}^{n(l)}$ denote the attention coefficient of node j to i for the n-th attention head, which can be computed as,

$$\alpha_{ij}^{n(l)} = \frac{\exp\left(\phi\left(\psi(W_n^{(l)} h_i^{(l)}, W_n^{(l)} h_j^{(l)})\right)\right)}{\sum_{m \in \mathcal{N}(i)} \exp\left(\phi\left(\psi(W_n^{(l)} h_i^{(l)}, W_n^{(l)} h_m^{(l)})\right)\right)}. \tag{4}$$

Here $\phi(\cdot)$ is a neural network, $W_n^{(l)}$ is a trainable matrix and ψ represents absolute difference. This is similar to the attention coefficient used in [23]. To form the weighted adjacency matrix, we first follow the same process as for MBSG by considering top k similar features based on cosine similarity to get the neighbourhood $\mathcal{N}(i)$ for each node i. To get the weighted adjacency matrix for the n-th head we set $\mathcal{A}_{ij}^{n(l)} = \alpha_{ij}^{n(l)}, \forall j \in \mathcal{N}(i)$ and $\mathcal{A}_{ij}^{n(l)} = 0, \forall j \notin \mathcal{N}(i)$.

We also remove self-connections by setting $\mathcal{A}_{ii}^{n(l)} = 0, \forall i$. The vectorized layer-wise propagation rule of the Attn-MBSG with N attention heads then becomes

$$\begin{aligned} \bar{H}^{(l)} &= H^{(l)} W^{(l)} + b^{(l)} \\ H^{(l+1)} &= \sigma\big(\text{combine}\big(\bar{H}^{(l)}, \frac{1}{N}\sum_{n=1}^{N} \mathcal{A}^{n(l)} \bar{H}^{(l)}\big)\big). \end{aligned} \tag{5}$$

We found that the number of attention heads, $N = 3$ provides best performance, so all our experiments use this value unless otherwise stated.

Once the intermediate representations (upto L layers) are obtained either using MBSG or Attn-MBSG, we get the final logits for the i-th input in the batch as

$$\ell_i = W_{final}(h_i^{(1)} \,\|\, h_i^{(2)} \,\|\, \ldots \,\|\, h_i^{(L)}), \tag{6}$$

where $\|$ denotes concatenation in the feature dimension. This captures the local and global information separately, and takes a weighted combination. This design choice has been used to increase the representation power of graph neural networks [26], by leveraging different neighbourhood ranges to better enable structure-aware representation. We can now compute the class probabilities by taking the softmax, $p(y_i = k|x_i) = \frac{\exp(\ell_k)}{\sum_{j=1}^{C}\exp(\ell_j)}$, where C is the total number of classes. We use cross entropy loss to train the encoder and the MBSG model end-to-end:

$$\mathcal{L} = -\sum_{i=1}^{B}\sum_{k=1}^{C} 1_{y_i=k} \log p(y_i = k|x_i). \tag{7}$$

Evaluation settings In the *transductive* setting, we predict the label for a single test image within a mini-batch graph consisting of one test image and multiple training images. In the *inductive* setting, we construct a graph on a full mini-batch consisting of test images and predict labels for all samples. We include results under both these settings.

3.2 Robustness to Input Perturbations

Many modern classification models have impressive performance on common datasets and benchmarks, but are brittle in that their performance can degrade severely when the input images are corrupted or perturbed [10, 16] (see Section 4.4) . We now show how our MBSG module provides robustness in the face of perturbations in the input, which is a highly desirable property for such models.

Proposition 1. *Consider a neural network comprised of an encoder and a fully connected layer, $g_{sup}(\cdot)$, and a MBSG neural network consisting of an encoder followed by a graph convolutional layer where each node has k neighbours, by $g_{MBSG}(\cdot)$. For transductive prediction, consider an input sample x, with some perturbation Δx. Let the associated perturbations in logits be $\Delta y_{sup} = g_{sup}(x+\Delta x) - g_{sup}(x)$ and $\Delta y_{MBSG} = g_{MBSG}(x+\Delta x) - g_{MBSG}(x)$. Then, $\Delta y_{MBSG} = \frac{1}{k+1}\Delta y_{sup}$.*

Proof. Let the encoder $e(\cdot)$ output a vector $e(x)$ for a given image x, and denote a given batch of samples as $\{x_1, x_2, ..., x_B\}$. For the standard supervised model, denoting the weight matrix of the final layer as W, we get the perturbation in the final pre-softmax logits, for perturbation in input x_B, as $\Delta y_{sup} = W^T e(x_B + \Delta x_B) - W^T e(x_B) = W^T[e(x_B + \Delta x_B) - e(x_B)]$.

Now, consider an MBSG model in which each sample is connected to k other samples in the mini-batch and has the same weight matrix W. For an MBSG with self connections, using the standard GCN update rule, we have $\Delta y_{MBSG} = \frac{1}{k+1} W^T [e(x_B + \Delta x_B) + \sum_{j \in \mathcal{N}(B)} e(x_j)] - \frac{1}{k+1} W^T [e(x_B) + \sum_{j \in \mathcal{N}(B)} e(x_j)] = \frac{1}{k+1} W^T [e(x_B + \Delta x_B) - e(x_B)] = \frac{1}{k+1} \Delta y_{sup}$ □

The above proposition effectively states that for any perturbation in the input, the corresponding perturbation in the output is inversely proportional to the number of neighbours, for each node in the mini-batch graph, when using MBSGs as opposed to standard networks. Similarity based aggregation aids in transductive inference, where a prediction is made for a single corrupted test image within a mini-batch of randomly sampled uncorrupted training set images. In Section 4.4 we carry out experiments to verify this property of robustness to image perturbations. This proposition only holds true for transductive prediction, where information from the uncorrupted training samples in the neighborhood is aggregated to improve robustness against noisy test samples. In the purely inductive case, the aggregation of features from noisy samples in the neighborhood might reduce performance.

We also test the robustness of the model against challenging black-box adversarial attacks. These adversaries craft perturbations which cause the model to classify legitimate looking input images incorrectly. Black-box adversaries do not have access to the model parameters and the gradients, and must query the model to observe the output class probabilities. They query the model repeatedly with a chosen image, but perturbing it with each iteration, based on the results of the previous query. These type of attacks are more severe than white-box attacks, where the adversary has access to the model parameters. A model which has a lower attack success rate and/or requires a higher number of queries on average against an adversary before a successful attack, is considered more robust. We test the MBSG model against two recently proposed and popular black-box adversarial attack methods, simBA [7] and Bandits-TD [11]. We choose these two methods since they use different methodologies - simBA uses local search in order to craft adversarial perturbations whereas Bandits-TD estimates the gradient by repeatedly querying the model to create the adversarial input. The results of these experiments are also provided in Section 4.4.

4 Experiments

4.1 Datasets

We perform experiments on two standard computer vision object classification datasets, CIFAR-10 and CIFAR-100. In order to expand the scope of the experiments to include scene classification, which is more complex, we also test our models on the MIT 67 scene dataset. For our experiments on robustness and image perturbations we us CIFAR-10, as is common in the machine learning literature [10, 16].

CIFAR-10 consists of 60,000 colour images in 10 classes, with 6,000 images per class. We use the standard split with 50,000 training images and 10,000 test images. Each image is 32 $\times$ 32.

CIFAR-100 is similar to CIFAR-10, except that it has 100 classes, which significantly increases the difficulty of the classification task. Each class contains 600 images with 500 training images and 100 testing images per class. Each image is 32 $\times$ 32.

MIT 67 contains indoor scene images belonging to 67 categories, with a total of 15620 images. The number of images varies across categories, but there are at least 100 images per

Model	CIFAR 10		CIFAR 100		MIT 67	
	Inductive	Transductive	Inductive	Transductive	Inductive	Transductive
Supervised vanilla	94.69 ±0.17		74.48 ±0.25		64.48 ±0.27	
Supervised contrastive []	94.85 ±0.13		74.80 ±0.22		65.10 ±0.16	
Affinity supervision []	94.45 ±0.35		74.50 ±0.59		64.60 ±0.30	
Data-Distortion Guided Self-Distillation []	94.80 ±0.16		74.61 ±0.30		65.00 ±0.21	
MBSG (concat)	95.19 ±0.23	95.21 ±0.21	75.22 ±0.17	75.15 ±0.20	65.80 ±0.21	65.81 ±0.19
MBSG (sum)	95.02 ±0.19	95.02 ±0.13	75.18 ±0.21	75.20 ±0.15	65.68 ±0.17	65.70 ±0.18
MBSG (dropfeat)	**95.24** ±0.19	**95.22** ±0.25	75.21 ±0.17	75.25 ±0.19	65.82 ±0.18	65.84 ±0.20
Attn-MBSG (concat)	95.14 ±0.14	95.12 ±0.21	75.16 ±0.22	75.19 ±0.25	**65.86** ±0.15	**65.87** ±0.18
Attn-MBSG (sum)	94.95 ±0.18	94.98 ±0.15	75.23 ±0.15	75.25 ±0.17	65.72 ±0.19	65.73 ±0.17
Attn-MBSG (dropfeat)	95.05 ±0.25	95.06 ±0.22	**75.29** ±0.11	**75.26** ±0.18	**65.86** ±0.22	65.85 ±0.20

Table 1: Image classification results using a Resnet-50 encoder. The results are shown with 95% confidence intervals over 5 runs. The architectures are trained using a single MBSG layer (L=1), batch size of 256 and with $k = 16$ for CIFAR-10, and $k = 4$ for CIFAR-100 and MIT67. We provide results for different combine modes of our single layered mini-batch graph based models (rows 4-9). *Additional results using a Wide Resnet-28-10 encoder are in the supplementary material.*

category. For our experiments we reduce the image size by approximately a factor of 10 in each dimension to 64×64, to ease the computational resources. As a consequence the scene classification task becomes harder.

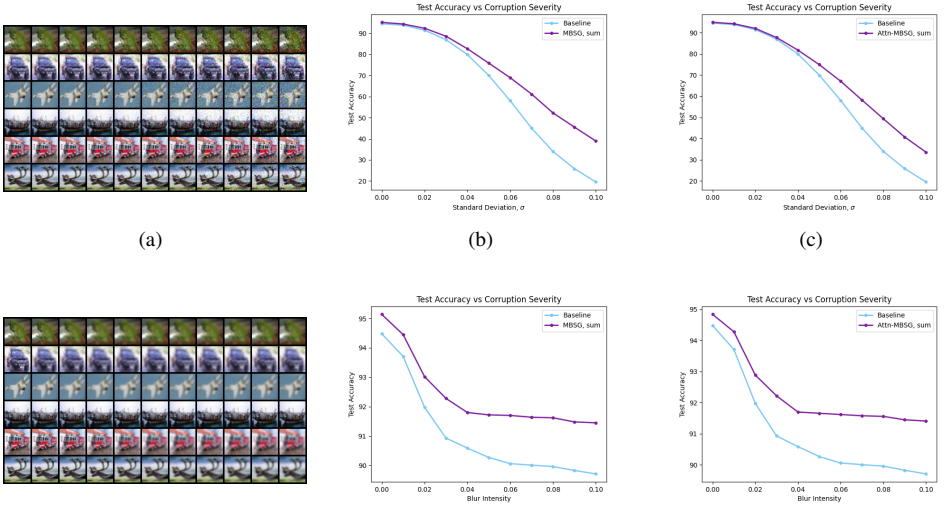

Figure 2: We encourage the reader to zoom-in on the PDF. Average test accuracy at different corruption severities for pixel-wise Gaussian noise (top) and Gaussian blurring (bottom) on CIFAR10, using a ResNet-50 encoder. Models using MBSGs (purple) maintain higher accuracy over the range of corruption severities when compared to the baseline model (blue) and have a lower reduction in accuracy for higher corruption levels. (a) Sample images with increasing level of Gaussian noise (left to right), (b) and (c) test accuracy plots for MBSG (sum) and Attn-MBSG (sum), (d) Sample images with increasing level of Gaussian blurring (left to right), (e) and (f) test accuracy plots for MBSG (sum) and Attn-MBSG (sum). *The plots for other models are provided in the supplementary material.*

4.2 Image classification results

Baselines. We use a ResNet-50 model trained using cross-entropy loss as the first baseline. We also reproduce the results of [13] with a batch size of 256 and reproduce results of [24] and [27] with our ResNet-50 baseline. We focus on relative improvement between our proposed model and these baselines.

For our proposed model, we use a ResNet-50 network without the final layer as an encoder and add a single layer (L=1) MBSG with different combine methods. The entire model is trained end-to-end using cross-entropy loss. We provide the results for both MBSG and Attn-MBSG (with 95% confidence intervals over 5 runs) with all the combine options in Table 1, under both the inductive and the transductive settings. Our models perform better than all three baselines across the datasets considered. We observe that there is no significant difference in test accuracy between the inductive and transductive settings. Among the different combine methods, the drop feature seems to perform best in general; however, the difference in performance between these variations is small. Also, Attn-MBSG is slightly better, owing to the model's higher expressivity due to learnable attention weights. We take the best value for k to be close to the batch size divided by the number of classes, since this is the expected number of samples in a mini-batch having the same class label.

4.3 Ablation studies

Here we present several ablation studies on various hyperparameters, to understand how they influence the performance of the model. *Additional ablations on the number of attention heads for Attn-MBSG, the β parameter for weighted addition and probability p for drop features are presented in the supplementary material.*

The most important parameters for graph-based learning are the size of the graph and the degree of each node, i.e., the neighborhood size k. Here we use the ResNet-50 encoder and the CIFAR-10 dataset for all our experiments. We expect the best value for k to be close to the batch size divided by the number of classes, since this is the expected number of samples in a mini-batch having the same class label. For CIFAR-10 with a batch size of 256, this value is $256/10 \approx 25$. We also expect performance to improve with larger batch sizes, since this translates to larger graphs, and hence more samples per class. These expectations are verified by the results of our experiments shown in Table 2 and Table 3.

Model	k=0	k=4	k=8	k=16	k=32	k=64	k=128	k=256
MBSG (dropfeat)	94.44	94.91	95.10	95.24	95.22	95.02	94.11	92.88
Attn-MBSG (dropfeat)	94.30	94.79	94.88	95.05	94.97	94.90	94.43	93.52

Table 2: Image classification results on CIFAR-10 using a Resnet-50 encoder and MBSG, while varying the neighbourhood size k. All the networks are trained with a batch size of 256 using a single layer GNN.

Model	BS=32	BS=64	BS=128	BS=256	BS=512
MBSG (dropfeat)	94.34	94.88	95.10	95.24	95.28
Attn-MBSG (dropfeat)	94.38	94.82	94.95	95.05	95.12

Table 3: Image classification results on CIFAR-10 using a Resnet-50 encoder and an MBSG, with different batch sizes. All the networks are trained with a neighbourhood size of $k = 16$, using a single layer GNN.

4.4 Robustness experiments

The performance of models on standard well-curated datasets is an important consideration, but equally important is their robustness to perturbations in the input and to adversarial attacks. We use CIFAR-10 for these experiments, as is common in the literature [10, 16], where image classification is an easy task since the dataset has a small number of visually distinct classes. Despite this, models such as ResNet-50 can be very sensitive to local perturbations (see Figure 2), hence this lack of robustness is a serious issue.

In order to test the robustness of MBSGs to image perturbations, we first consider random (pixel-wise) Gaussian noise and local Gaussian blurring on input images, with varying levels of corruption severity. The evaluation of test accuracy is done via transductive testing, where a mini-batch consists of a single corrupted image along with a training set of uncorrupted images. The class label prediction made by the model for the corrupted image is compared against the true label. Figure 2 shows plots of test accuracy, measured in the manner described above, for the best performing variations of MBSG (sum) and Attn-MBSG (sum) on the CIFAR-10 dataset for different levels of corruption severity. We observe that models using MBSGs are far better at accommodating the effects of local perturbations to the images (Fig. 2 first row) with 50% or better test accuracy in relative terms over the baseline, at higher levels of corruption. Although the effects of local Gaussian blur are less harmful, models based on MBSGs are still better by about 2% over the range of corruption severities we have considered (Fig. 2 second row).

We also test the robustness of the model to two recently proposed and popular black-box adversarial attack methods, simBA [7], and Bandits-TD [11]. Table 4 shows the mean and median number of queries before a successful attack, and the attack success rate, for different MBSG models for the two attack methods. The MBSG models have lower attack success rates and higher mean/median queries before a successful attack when compared to the baseline ResNet model. The large increase in mean queries can be attributed to a heavier tail in the distribution of queries, as can be seen from the histogram plots in Figure 3, which show results for the best performing variations of MBSG (dropfeat) and Attn-MBSG (dropfeat). *Plots for other variations are provided in the supplementary material.* The Attention MBSG models outperform the baseline model but do not perform as well as the MBSG models. One reason for this might be that any perturbation in the input samples has a compounding effect on calculating the attention weights and, therefore, the aggregation. Overall, using the

Model	SimBA			Bandits-TD		
	Mean Queries	Median Queries	Attack Success rate	Mean Queries	Median Queries	Attack Success rate
Baseline ResNet-50	357.31	302	100.00 %	564.22	524	87.93 %
MBSG, concat	508.53	388	99.79 %	768.86	620	87.77 %
MBSG, sum	520.46	**394**	99.89 %	708.55	638	87.74 %
MBSG, dropfeat	**572.15**	387	**99.68 %**	**787.69**	**659**	**87.66 %**
Attn MBSG, concat	**415.11**	**342**	99.89 %	**613.15**	574	87.92 %
Attn MBSG, sum	358.88	319	100.00 %	558.31	542	87.94 %
Attn MBSG, dropfeat	392.19	338	**99.79 %**	590.17	**580**	**87.84 %**

Table 4: Black-box adversarial attack results for the baseline and the proposed models using simBA [7] and Bandits-TD [11] on the CIFAR10 dataset. A higher number of queries is better, and a lower attack success rate is better. Here the mean and median number of queries is calculated over successful attacks only.

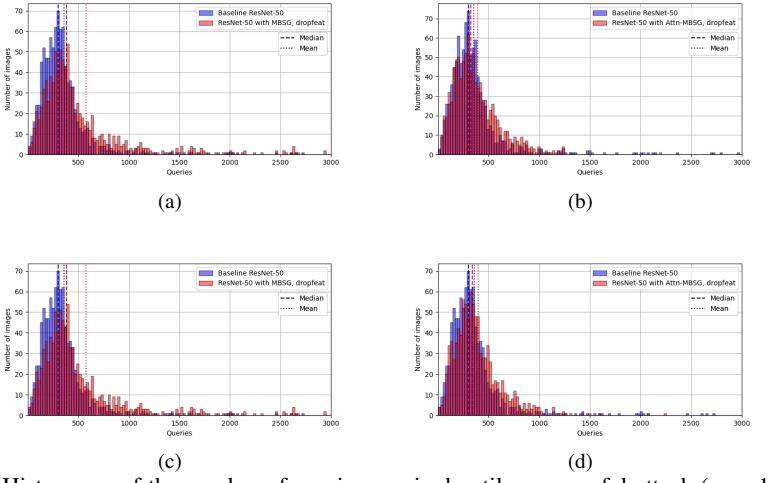

Figure 3: Histograms of the number of queries required until a successful attack (over 1000 target images) on the CIFAR10 dataset using dropfeat MBSG and Attn-MBSG for simBA (top) and Bandits-TD (bottom). The queries axis is limited to 3000 queries for clarity of presentation. Models using MBSGs (red) require more queries on average for a successful attack as compared to the baseline model (blue). *The plots for other model variations are provided in the supplementary material.*

MBSG module can help significantly improve robustness against common noise perturbation as well as black-box adversarial attacks.

5 Discussion

Our proposed MBSG method for mini-batch training shows great promise in application to image classification with respect to performance and robustness. MBSGs can be used with the most popular network models and require a modification of only their last layer. As such, we anticipate that they could find use for diverse computer vision tasks beyond classification as well, including segmentation [1, 12], region proposal detection [18] and relationship modeling [28]. We also show how a mini-batch graph can be used for GAN training and connect this with previous work on improved training for GANs [20], in Section 3 in the supplementary material. We show that using mini-batch graphs in the discriminator mitigates model collapse in GANs.

In ongoing work, we are extending our method so as to be able to apply it to larger datasets. The present limitation is not conceptual, but rather, has to do with computational requirements. From an implementation standpoint, we are working on parallelization so that the samples in a mini-batch can be spread across multiple GPUs. We are also working towards a more efficient use of the available memory.

There is an increasing interest in the vision community to build systems that can perform well in a variety of real world applications, where inputs may have unwanted perturbations or the model itself may be subject to attacks from adversaries. This work is a step towards achieving that goal by improving upon raw classification performance, and providing appreciable added robustness to image perturbations and adversarial attacks.

Acknowledgments We are grateful to the Natural Sciences and Engineering Research Council of Canada for support.

References

[1] Rodrigo Benenson, Stefan Popov, and Vittorio Ferrari. Large-scale interactive object segmentation with human annotators. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[2] Joan Bruna, Wojciech Zaremba, Arthur Szlam, and Yann LeCun. Spectral networks and locally connected networks on graphs. *arXiv preprint arXiv:1312.6203*, 2013.

[3] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. *arXiv preprint arXiv:2002.05709*, 2020.

[4] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. In *Advances in neural information processing systems*, pages 3844–3852, 2016.

[5] Liyu Gong and Qiang Cheng. Exploiting edge features for graph neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9211–9219, 2019.

[6] Marco Gori, Gabriele Monfardini, and Franco Scarselli. A new model for learning in graph domains. In *Proceedings. 2005 IEEE International Joint Conference on Neural Networks, 2005.*, volume 2, pages 729–734. IEEE, 2005.

[7] Chuan Guo, Jacob R. Gardner, Yurong You, A. Wilson, and Kilian Q. Weinberger. Simple black-box adversarial attacks. In *ICML*, 2019.

[8] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[9] Mikael Henaff, Joan Bruna, and Yann LeCun. Deep convolutional networks on graph-structured data. *arXiv preprint arXiv:1506.05163*, 2015.

[10] Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. In *International Conference on Learning Representations*, 2018.

[11] Andrew Ilyas, Logan Engstrom, and Aleksander Madry. Prior convictions: Black-box adversarial attacks with bandits and priors. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=BkMiWhR5K7.

[12] Or Isaacs, Oran Shayer, and Michael Lindenbaum. Enhancing generic segmentation with learned region representations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[13] Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. Supervised contrastive learning. *arXiv preprint arXiv:2004.11362*, 2020.

[14] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

[15] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In *Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 1*, NIPS'12, page 1097–1105, Red Hook, NY, USA, 2012. Curran Associates Inc.

[16] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representations*, 2018.

[17] Mathias Niepert, Mohamed Ahmed, and Konstantin Kutzkov. Learning convolutional neural networks for graphs. volume 48 of *Proceedings of Machine Learning Research*, pages 2014–2023, New York, New York, USA, 20–22 Jun 2016. PMLR. URL http://proceedings.mlr.press/v48/niepert16.html.

[18] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015.

[19] Pau Rodríguez, Issam Laradji, Alexandre Drouin, and Alexandre Lacoste. Embedding propagation: Smoother manifold for few-shot classification. *arXiv preprint arXiv:2003.04151*, 2020.

[20] Tim Salimans, Ian J. Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. *CoRR*, abs/1606.03498, 2016. URL http://arxiv.org/abs/1606.03498.

[21] Victor Garcia Satorras and Joan Bruna Estrach. Few-shot learning with graph neural networks. In *International Conference on Learning Representations*, 2018.

[22] Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. The graph neural network model. *IEEE Transactions on Neural Networks*, 20(1):61–80, 2008.

[23] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=rJXMpikCZ.

[24] Chu Wang, Babak Samari, Vladimir G Kim, Siddhartha Chaudhuri, and Kaleem Siddiqi. Affinity graph supervision for visual recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8247–8255, 2020.

[25] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E Sarma, Michael M Bronstein, and Justin M Solomon. Dynamic graph cnn for learning on point clouds. *Acm Transactions On Graphics (tog)*, 38(5):1–12, 2019.

[26] Keyulu Xu, Chengtao Li, Yonglong Tian, Tomohiro Sonobe, Ken-ichi Kawarabayashi, and Stefanie Jegelka. Representation learning on graphs with jumping knowledge networks. *arXiv preprint arXiv:1806.03536*, 2018.

[27] Ting-Bing Xu and Cheng-Lin Liu. Data-distortion guided self-distillation for deep neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 5565–5572, 2019.

[28] Ji Zhang, Mohamed Elhoseiny, Scott Cohen, Walter Chang, and Ahmed Elgammal. Relationship proposal networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5678–5686, 2017.

[29] B. Zhou, A. Lapedriza, A. Khosla, A. Oliva, and A. Torralba. Places: A 10 million image database for scene recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(6): 1452–1464, 2018. doi: 10.1109/TPAMI.2017.2723009.

[30] Jiong Zhu, Yujun Yan, Lingxiao Zhao, Mark Heimann, Leman Akoglu, and Danai Koutra. Generalizing graph neural networks beyond homophily. *arXiv preprint arXiv:2006.11468*, 2020.

[31] Chengxu Zhuang, Xuehao Ding, Divyanshu Murli, and Daniel Yamins. Local label propagation for large-scale semi-supervised learning, 2019.