Conditional Group Normalization: An Empirical Study

Anonymous Authors

Abstract

Batch normalization has been widely used to improve optimization in deep neural networks. While the uncertainty in batch statistics can act as a regularizer, using these dataset statistics specific to the training set impairs generalization in certain tasks. Recently, alternative methods for normalizing feature activations in neural networks have been proposed. Among them, group normalization has been shown to yield competitive performance to batch normalization. All these methods utilize a learned affine transformation after the normalization operation to increase representational power. Methods used in conditional computation define the parameters of these transformations as learnable functions of conditioning information. In this work, we study whether and where the conditional formulation of group normalization can improve generalization compared to conditional batch normalization. We evaluate performances on the tasks of visual question answering, few-shot learning, and conditional image generation. Our experiments indicate that conditional group normalization is a reasonable replacement for conditional batch normalization and in some cases achieves improved performance.

In machine learning, the parameters of a model are typically optimized using a fixed training set. The model is then evaluated on a separate partition of the data to estimate its generalization capability. In practice, even under the i.i.d. assumption[1], the distribution of these two finite sets can *appear* quite different to the learning algorithm, making it challenging to achieve strong and robust generalization. This difference is often the result of the fact that a training set of limited size cannot adequately cover the cross-product of all relevant factors of variation. In other cases, the i.i.d. assumption is dropped on purpose, to study whether a model can capture regularity that generalize *out-of-distribution*. Several benchmarks for evaluating task specific models for their generalization capacity (Johnson et al. 2017; Kahou et al. 2017; Bahdanau et al. 2018) have been proposed recently. The problem of out-of-distribution generalization can in some cases be addressed by making strong assumptions that simplify discovering a family of patterns from limited data. Bahdanau et al. (2018), for example, show that their proposed synthetic relational reasoning task can be solved by a Neural Module Network (NMN) (Andreas et al. 2016) with fixed tree structure, while models without this structural prior fail. At the other end of the spectrum are more "generic" models as Feature-wise Linear Modulation (FiLM), which are able to uncover some of the compositionality in the tasks they are trained for.

In this paper, we focus on several of these generic models, that leverage conditional normalization methods for applications in visual question answering (VQA), few-shot classification and generative modeling of images. Despite our focus on these tasks, any improvement in this area can also benefit other domains such as reinforcement learning. We study strong deep neural network models for these tasks that employ Conditional Batch Normalization (CBN) (De Vries et al. 2017) for modulating normalized activations with contextual information.

Since Batch Normalization (BN) normalizes activations with statistics computed across multiple training samples, one has to precompute activation statistics over the training set to be used during inference. Due to this reliance on dataset statistics, it seems that BN (Ioffe and Szegedy 2015) (and thus also CBN) may be vulnerable to significant domain shifts between training and test data. To train models with BN one has to use a sufficiently large mini-batch size to limit the noise of activation statistics. Further potential issues with BN include limited diversity of samples generated by Generative Adversarial Networks (GANs) involving BN (Xiang and Li 2017) and vulnerability to adversarial examples (Galloway et al. 2019).

The recently proposed Group Normalization (GN) (Wu and He 2018) normalizes across groups of feature maps instead of across batch samples. Here, we explore whether a conditional formulation of GN is a viable alternative for CBN. GN is conceptually simpler than BN, as its function is the same during training and inference. Further, GN can be used with small batch sizes, which may help in applications with particularly large feature maps, such as medical imaging or video processing, in which the available memory can be a constraint.

Our contribution is an extensive empirical study of two

[1] All data samples are assumed to be drawn independently from an identical distribution (i.i.d.).

conditional normalization techniques over a broad range of tasks and benchmarks. To the best of our knowledge, the present work is the first to introduce Conditional Group Normalization (CGN). Our experiments indicate that it may have advantages for out-of-distribution generalization.

Background

Normalization Layers

Several normalization methods have been proposed to stabilize and speed-up the training of deep neural networks (Ioffe and Szegedy 2015; Wu and He 2018; Lei Ba, Kiros, and Hinton 2016; Ulyanov, Vedaldi, and Lempitsky 2016). To stabilize the range of variation of network activations x_i, methods such as BN (Ioffe and Szegedy 2015) first normalize the activations by subtracting mean μ_i and dividing by standard deviation σ_i:

$$\hat{x}_i = \frac{1}{\sigma_i}(x_i - \mu_i) \tag{1}$$

The distinction between different methods lies in how exactly these statistics are being computed. Wu and He (2018) aptly summarize several methods using the following notation. Let $i = (i_N, i_C, i_H, i_W)$ be a four-dimensional vector, whose elements index the features along the batch, channel, height and width axes, respectively. The computation of the statistics can then be written as

$$\mu_i = \frac{1}{m}\sum_{k\in\mathcal{S}_i} x_k, \quad \sigma_i = \sqrt{\frac{1}{m}\sum_{k\in\mathcal{S}_i}(x_k - \mu_i)^2 + \epsilon}, \tag{2}$$

where the set $\mathcal{S}_i$ of size m is defined differently for each method and ϵ is a small constant for numerical stability. BN, for instance, corresponds to:

$$\text{BN} \implies \mathcal{S}_i = \{k | k_C = i_C\}, \tag{3}$$

i.e. $\mathcal{S}_i$ is the set of all pixels sharing the same channel axis, resulting in μ_i and σ_i being computed along the (N, H, W) axes.

As Lei Ba, Kiros, and Hinton (2016) point out, the performance of BN is highly affected by the batch size hyperparameter. This insight led to the introduction of several alternative normalization schemes, that normalize per sample, i.e. not along batch axis N. Layer Normalization (LN) (Lei Ba, Kiros, and Hinton 2016), which normalizes activations within each layer, corresponds to the following set definition:

$$\text{LN} \implies \mathcal{S}_i = \{k | k_N = i_N\}. \tag{4}$$

Ulyanov, Vedaldi, and Lempitsky (2016) introduce Instance Normalization (IN) in the context of image stylization. IN normalizes separately for each sample and each channel along the spatial dimensions:

$$\text{IN} \implies \mathcal{S}_i = \{k | k_N = i_N, k_C = i_C\}. \tag{5}$$

Recently, Wu and He (2018) introduced GN, which draws inspiration from classical features such as HOG (Dalal and Triggs 2005). It normalizes features per sample, separately within each of G groups, along the channel axis:

$$\text{GN} \implies \mathcal{S}_i = \{k | k_N = i_N, \lfloor \frac{k_C}{C/G} \rfloor = \lfloor \frac{i_C}{C/G} \rfloor\} \tag{6}$$

GN can be seen as a way to interpolate between the two extremes of LN (corresponding to $G = 1$, i.e. all channels are in a single group) and IN (corresponding to $G = C$, i.e. each channel is in its own group).

After normalization, all above mentioned methods insert a scaling and shifting operation using learnable per-channel parameters γ and β:

$$y_i = \gamma\hat{x}_i + \beta \tag{7}$$

This "de-normalization" is done to restore the representational power of the normalized network layer (Ioffe and Szegedy 2015).

CBN (De Vries et al. 2017; Perez et al. 2018) is a conditional variant of BN, in which the learnable parameters γ and β in Equation 7 are replaced by learnable functions

$$\gamma(c_k) = W_\gamma c_k + b_\gamma, \;\; \beta(c_k) = W_\beta c_k + b_\beta \tag{8}$$

of some per-sample conditioning input c_k to the network with parameters W_γ, W_β, b_γ, b_β. In a VQA model, c_k would for instance be an embedding of the question (Perez et al. 2018). Before CBN was introduced, Dumoulin, Shlens, and Kudlur (2017) proposed Conditional Instance Normalization (CIN), a conditional variant of IN very similar to CBN, using IN instead of BN. In our experiments, we explore a conditional variant of GN, i.e. CGN.

Visual Question Answering

In VQA (Malinowski and Fritz 2014; Antol et al. 2015), the task is to answer a question about an image. This task is usually approached by feeding both image and question to a parametric model, which is trained to predict the correct answer, for instance via classification among all possible answers in the dataset. One recent successful model for VQA is the FiLM architecture (Perez et al. 2018), which employs CBN to modulate visual features based on an embedding of the question.

Few-Shot Classification

The task of few-shot classification consists in the challenge of classifying data given only a small set of support samples for each class. In episodic M-way, k-shot classification tasks, meta-learning models (Ravi and Larochelle 2016) learn to adapt a classifier given multiple M-class classification tasks, with k support samples for each class. The meta-learner thus has to solve the problem of generalizing between these tasks given the limited number of training samples. In this work we experiment with the recently proposed Task dependent adaptive metric (TADAM) architecture (Oreshkin, López, and Lacoste 2018). It belongs to the family of meta-learners that employ nearest neighbor classification within a learned embedding space. In the case of TADAM, the network providing this embedding is modulated by a task embedding using CBN.

Conditional Image Generation

Some of the most successful models for generating images are GANs (Goodfellow et al. 2014). This approach involves training a neural network (Generator) to generate an image,

while the only supervisory signal is that from another neural network (Discriminator) which indicates whether the image looks real or not. Several variants of GANs (Mirza and Osindero 2014; Odena, Olah, and Shlens 2017) have been proposed to condition the image generation process on a class label. More recently, the generators that work best stack multiple ResNet-style (He et al. 2016) architectural blocks, involving two CBN-ReLU-Conv operations and an upsampling operation. These blocks are followed by a BN-ReLU-Conv operation to transform the last features into the shape of an image. Such models can be trained as Wasserstein GANs using gradient penalty (WGAN-GP) as proposed by Gulrajani et al. (2017), which gives mathematically sound arguments for an optimization framework.

More recently, Spectral Norm GAN (SNGAN) (Miyato et al. 2018a) uses the aforementioned architecture with spectral normalization on the weights to stabilize training at each iteration. Two noteworthy GAN models that use architectures baselined on SNGAN are Self-Attention GAN (SAGAN) (Zhang et al. 2018a) and BigGAN (Brock, Donahue, and Simonyan 2019). SAGAN inserts a self-attention mechanism (Parikh et al. 2016; Vaswani et al. 2017; Cheng, Dong, and Lapata 2016) to attend over important parts of features during the generation process. The architecture of BigGAN is the same as for SAGAN, with the exception of an increase in batch size and channel widths, as well as some architectural changes to improve memory and computational efficiency. Both these models have been successfully used in generating high quality natural images. In our experiments, we use SNGAN and compare performance metrics of two types of normalization — CBN and CGN.

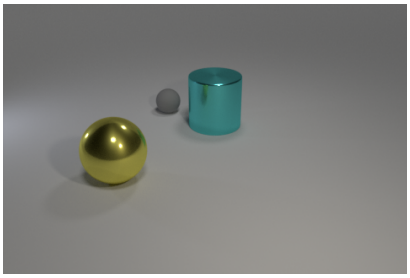
Experiments

Visual Question Answering

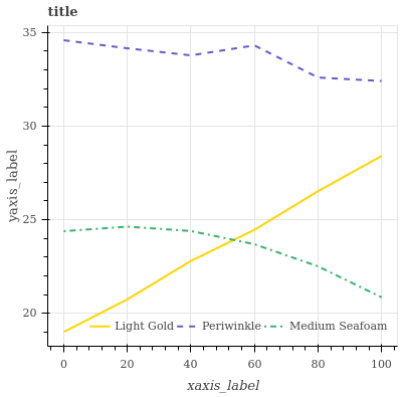
We study whether substituting CGN for CBN in the VQA architecture FiLM (Perez et al. 2018) yields comparable performance. We run experiments on the following recently proposed benchmarks for compositional generalization:
CLEVR Compositional Generalization Test (CLEVR-CoGenT) (Johnson et al. 2017) is a variant of the popular Compositional Language and Elementary Visual Reasoning (CLEVR) dataset (Johnson et al. 2017) that tests for compositional generalization. See Figure 1 (a) for an example from this dataset. The images consist of rendered three-dimensional scenes containing several shapes (small and large cubes, spheres and cylinders) of differing material properties (*metal* or *rubber*) and colors. Questions involve *queries* for object attributes, *comparisons*, *counting* of sets and combinations thereof. In contrast to the regular CLEVR dataset, the training set of CLEVR-CoGenT explicitly combines some shapes only with different subsets of four out of eight colors, and provides two validation sets: one with the same combinations (*valA*) and one in which the shape-color assignments are swapped (*valB*). To perform well on *valB*, the model has to generalize to unseen combinations of shapes and colors, i.e. it needs to capture the compositionality of the task.
Figure Question Answering (FigureQA) (Kahou et al. 2017) is a VQA dataset consisting of mathematical plots with templated yes/no question-answer pairs that address relations between plot elements. Figure 1 (b) shows a sample from the dataset. The dataset contains plots of five types (vertical/horizontal bar plots, line plots, pie charts and dot-line plots). Each plot has between 2 and 10 elements, each of which has one of 100 colors. Plot elements (e.g. a slice in a pie chart) are identified by their color names in the questions. Questions query for *one-vs-one* or *one-vs-all* attribute relations, e.g. "Is Lime Green less than WebGray?" or "Does Cadet Blue have the minimum area under the curve?". Similar to CLEVR-CoGenT, FigureQA requires compositional generalization. The overall 100 colors are split into two sets A and B, each containing 50 unique colors. During training, colors of certain plot types are sampled from set A, while the remaining plot types use colors from set B (*scheme 1*). There are two validation sets, one using the same color scheme, and one for which the plot-type to color assignments are swapped (*scheme 2*).
Spatial Queries On Object Pairs (SQOOP) (Bahdanau et al. 2018) is a recently introduced dataset that tests for systematic generalization. Figure 1 (c) shows an example from the training set. It consists of images containing five randomly chosen and arranged objects (digits and characters). Questions concern the four spatial relations *LEFT OF*, *RIGHT OF*, *ABOVE* and *BELOW* and the queries are all of the format "X R Y?", where X and Y are left-hand and right-hand objects and R is a relationship between them, e.g. "nine LEFT OF a?". To test for systematic generalization, only a limited number of combinations of each left-hand object with different right-hand objects Y are shown during training. In the hardest version of the task (1 rhs/lhs), only a single right-hand side object is combined with each left-hand side object. For instance, the training set of this version may contain images with the query "A RIGHT OF B", but no images with queries about relations of left-hand object A with any other object than B. The test set contains images and questions about all combinations, i.e. it evaluates generalization to relations between novel object combinations.

Model We experiment with three small variations of the FiLM architecture (Perez et al. 2018). The original architecture in Perez et al. (2018) consists of an unconditional *stem* network, followed by a core of four ResNet (He et al. 2016) blocks with CBN (De Vries et al. 2017), and finally a classifier. The stem network is either a sequence of residual blocks trained from scratch or a fixed pre-trained feature extractor followed by a learnable layer of 3×3 convolutions. The scaling and shifting parameters of the core layers are affine transforms of a question embedding provided by a gated recurrent unit (GRU) (Cho et al. 2014). The output of the last residual block is fed to the classifier, which consists of a layer of 512 1×1 convolutions, global max-pooling, followed by a fully-connected ReLU (Nair and Hinton 2010) layer using (unconditional) BN and a softmax layer, which outputs the probability of each possible answer. We train the following three variants that include CGN[2]:

[2]We always set the number of groups to 4, as the authors of Wu and He (2018) showed that this hyperparameter does not have a



(a) **CLEVR-CoGenT**



(b) **FigureQA**

(c) **SQOOP**

Figure 1: Examples of the VQA datasets used in our experiments. (a) **CLEVR-CoGenT**: Are there any gray things made of the same material as the big cyan cylinder? - No. (b) **FigureQA**: Does Medium Seafoam intersect Light Gold? - Yes. (c) **SQOOP**: X right_of J? - no

1. "all GN": all conditional and regular BN layers are replaced with corresponding conditional or regular GN layers.
2. "BN stem": all CBN layers are replaced with CGN, regular BN layers in the stem and classifier are left unchanged, except those in the fully-connected hidden layer in the classifier, for which we remove normalization.
3. "BN stem & cls": all CBN layers in the core ResNet blocks are replaced with CGN, regular BN in the stem and classifier are left unchanged.

Besides the described changes in the normalization layers, the architecture and hyperparameters are the same as used in Perez et al. (2018) for all experiments, except for SQOOP where they are the same as in Bahdanau et al. (2018). The only difference is that we set the constant ϵ of the Adam optimizer (Kingma and Ba 2014) to $1e{-}5$ to improve training stability[3]. For SQOOP, the input to the residual network are the raw image pixels. For all other networks, the input is features extracted from layer *conv4* of a ResNet-101 (He et al. 2016), pre-trained on ImageNet (Russakovsky and others 2015), following Perez et al. (2018).

Results Tables 1, 2 and 3 show the results of training FiLM with CBN and CGN on the three considered datasets. In the experiments on CLEVR-CoGenT, all three CGN variants of FiLM achieve a slightly higher average accuracy. On FigureQA, CBN outperforms CGN slightly. Note that for the SQOOP dataset we rerun the original CBN experiments by Bahdanau et al. (2018) and observe significantly higher accuracy on all versions of the task. In the hardest SQOOP variant with only one right-hand side object per left-hand side object (*1 rhs/lhs*), all three variants of CGN achieve a higher performance than the CBN experiments (both original and ours). For SQOOP variants whose training sets contain more combinations, CGN did not converge in some cases. Learning curves of models successfully trained on SQOOP seem to follow the same pattern: For a relatively large number of gradient updates there is no significant improvement. Then, at some point, almost instantly the model achieves 100% training accuracy. It is possible that a hyperparameter search or additional regularization is required to guarantee convergence in these cases.

Table 1: Classification accuracy on CLEVR-CoGenT *valB*. Mean and standard deviation of three runs with early stopping on *valA* are reported for the models we trained.

Model	Accuracy (%)
CBN (FiLM (Perez et al. 2018))	75.600
CBN (FiLM, our results)	75.539 ± 0.671
CGN (all GN)	75.758 ± 0.356
CGN (BN stem)	75.703 ± 0.571
CGN (BN stem & cls)	$\mathbf{75.807 \pm 0.511}$

Table 2: Classification accuracy on FigureQA *validation2*, mean and standard deviation of three runs after early stopping on *validation1*.

Model	Accuracy (%)
CBN (FiLM, our results)	$\mathbf{91.618 \pm 0.132}$
CGN (all GN)	91.343 ± 0.436
CGN (BN stem)	91.080 ± 0.166
CGN (BN stem & cls)	91.317 ± 0.514

Few-Shot Learning

CBN has also been used in recent methods for few-shot learning (Oreshkin, López, and Lacoste 2018; Jiang et al. 2018). We replicate the experiments of Oreshkin, López, and Lacoste (2018) on Mini-ImageNet and Fewshot-CIFAR100

large influence on the performance. This number was selected using uniform sampling from the set $\{2, 4, 8, 16\}$.

[3]The authors of Perez et al. (2018) confirmed occasional gradient explosions with the original setting of $1e-8$.

Table 3: Test accuracies on several versions of SQOOP. Mean and standard deviation of three runs after early stopping on the validation set are reported for the models we trained. Here, FiLM refers to the model specified in (Bahdanau et al. 2018), whereas "FiLM, ours" indicates our run of the same.

Dataset	Model	Accuracy (%)
1 rhs/lhs	CBN (FiLM)	65.270 ± 4.610
	CBN (FiLM, ours)	72.369 ± 0.529
	CGN (all GN)	74.020 ± 2.814
	CGN (BN stem)	73.824 ± 0.334
	CGN (BN stem & cls)	$\mathbf{74.929 \pm 3.888}$
2 rhs/lhs	CBN (FiLM)	80.200 ± 4.320
	CBN (FiLM, ours)	84.966 ± 4.165
	CGN (all GN)	$\mathbf{86.689 \pm 6.308}$
	CGN (BN stem)	83.109 ± 0.381
	CGN (BN stem & cls)	85.859 ± 5.318
4 rhs/lhs	CBN (FiLM)	90.420 ± 1.000
	CBN (FiLM, ours)	97.043 ± 1.958
	CGN (all GN)	91.404 ± 0.318
	CGN (BN stem)	91.601 ± 1.937
	CGN (BN stem & cls)	$\mathbf{99.474 \pm 0.254}$
35 rhs/lhs	CBN (FiLM)	99.803 ± 0.219
	CBN (FiLM, ours)	$\mathbf{99.841 \pm 0.043}$
	CGN (all GN)	99.755 ± 0.025
	CGN (BN stem)	99.815 ± 0.122
	CGN (BN stem & cls)	99.782 ± 0.155

(FC100) using their code for TADAM and compare the results with a version that uses CGN instead of CBN.

Mini-ImageNet was proposed by Vinyals et al. (2016) as a benchmark for few-shot classification. It contains 100 classes, for each of which there are 600 images of resolution 84×84. To generate five-way five-shot classification tasks five classes and five support samples for each class are sampled uniformly. The remaining images are used to compute the accuracy. Using the proposed split by Ravi and Larochelle (2016), training tasks are sampled from a subset of 64 classes. The remaining 36 classes are divided into 16 for meta-validation and 20 for meta-testing.

Fewshot-CIFAR100 (Oreshkin, López, and Lacoste 2018) is a few-shot classification version of the popular CIFAR100 data set (Krizhevsky 2009). Similarly to Mini-ImageNet, it contains 100 classes and 600 samples per class. The resolution of the images is 32×32. The classes are split by superclasses to reduce information overlap between data set partitions, which makes the task more challenging than Mini-ImageNet. The training partition contains 60 classes belonging to 12 superclasses. The validation and test partitions contain 20 classes belonging to 5 superclasses each. The tasks are sampled uniformly as in Mini-ImageNet.

Model TADAM (Oreshkin, López, and Lacoste 2018) is a metric-based few-shot classifier, i.e. it learns a measure of similarity between query samples and class representations. The metric is based on a learned image embedding $\mathrm{f}_\phi(x, c)$ provided by a residual network. Figure 2 shows a diagram of the overall architecture. Each class template is computed as the average embedding of all support samples for the respective class. The Euclidean distances between the embedding of a query sample and each of the class templates, weighted by a learned scaling factor α, is then used to classify the query sample x^*. The embedding network f_ϕ (see the dashed boxes in Figure 2) is modulated using CBN with a conditioning input c. In the computation of the similarity metric, c is fed by a task embedding Γ provided by a task embedding network (TEN), which reads the average embeddings of support samples from all classes of the task. Note that f_ϕ is evaluated without conditioning (i.e. by setting c to a zero vector[4]) in the computation of the task embedding Γ (see bottom of Figure 2). For the GN version we replaced all conditional and regular BN layers with their corresponding conditional or regular GN version (with the number of groups set to 32). For a complete description of the experimental setup, including all other hyperparameters, we refer the reader to Oreshkin, López, and Lacoste (2018).

Table 4: Five-way five-shot classification accuracy on Fewshot-CIFAR100 (Oreshkin, López, and Lacoste 2018) and Mini-Imagenet (Vinyals et al. 2016), mean and standard deviation of ten runs. The reported CBN numbers were achieved using the TADAM implementation released by Oreshkin, López, and Lacoste (2018).

Dataset	Model	Accuracy (%)
FC100	TADAM (CBN)	$\mathbf{52.996 \pm 0.610}$
	TADAM (CGN)	52.807 ± 0.509
Mini-Imagenet	TADAM (CBN)	$\mathbf{76.414 \pm 0.499}$
	TADAM (CGN)	74.032 ± 0.373

Results We see that using CGN instead of CBN yields only slightly reduced performance on FC100. There is a considerable 2.4% gap for Mini-ImageNet, which may in part be due to suboptimal hyperparameter settings, noting that the used settings were tuned for the CBN version.

Conditional Image Generation

Here we compare CBN and CGN on the task of generating images conditioned on their class label using the SNGAN (Miyato et al. 2018a) architecture.

CIFAR-10 (Krizhevsky 2009) is a data set containing 60000 32×32 images, 6000 for each of 10 classes. The dataset is split into 50000 training and 10000 test samples.

Model We use the official implementation of SNGAN (Miyato et al. 2018b). We replace the BN modules in the residual blocks with CBN and CGN for the respective cases, with number of groups set to 4 in case of CGN. We retain the optimization setup of a learning rate of $2\mathrm{e}^{-4}$ for both generator and discriminator, five discriminator updates per generator update using Adam optimizer (Kingma

[4]The conditioning input is implemented as a deviation from the identity transform (unity scaling and zero shift), so setting it to zero does not change the normalized activations.

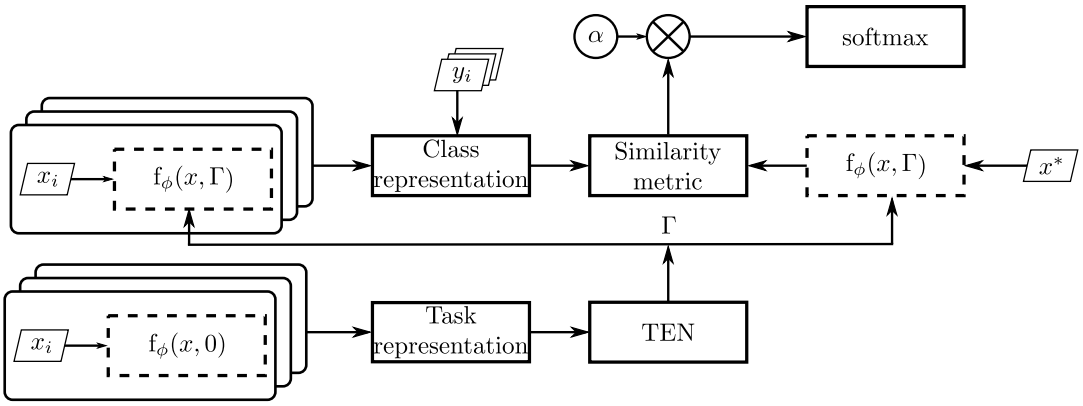


Figure 2: Architecture of TADAM (Oreshkin, López, and Lacoste 2018). Boxes with dashed border share parameters. Figure adapted from (Oreshkin, López, and Lacoste 2018).

and Ba 2014). We train using a single GPU (NVIDIA P100) and a batch size of 64. We also perform experiments where we use smaller batch sizes, to show the effects of CBN and CGN in helping generalization.

Results Figure 3 shows samples from conditional SNGAN trained using each of the two normalization methods.

For both normalization methods, in addition to a qualitative check of the generated samples, we calculate two scores that are widely used in the community to evaluate image generation Inception Score (IS) (Salimans et al. 2016; Barratt and Sharma 2018) and Fréchet Inception Distance (FID) (Heusel et al. 2017). Since we are using publicly available PyTorch (Paszke et al. 2017) implementations to compute these scores, the values for real data may differ slightly from scores computed using the original TensorFlow (Abadi and others 2015) implementation. However, we compare scores using the same implementation of these metrics for true and generated data.

IS is meant to measure the natural-ness of an image by checking the embedding of the generated images on a pre-trained Inception network (Szegedy et al. 2016). Although the suitability of the IS for this purpose has been rightfully put into question (Barratt and Sharma 2018), it continues to be used frequently. FID measures how similar two sets of images are, by computing the Fréchet distance between two multivariate Gaussians fitted to the embeddings of the images from the two sets. The embeddings are obtained from a pre-trained InceptionV3 network (Szegedy et al. 2016). In this case, we measure the distance between the real CIFAR-10 images, and the generated ones. This is a better metric than IS, since there is no constraint on the images being natural, and it is able to quantify not only their similarity to the real images, but also diversity in the generated images.

We trained SNGAN models to generate CIFAR-10 images conditioned on the class, for different batch sizes, viz. 64, 32, 16, 8, 4. In each case, we trained one model with CBN and another with CGN to compare them. We calculated IS and FID on these models at different stages of training, as can be seen in Figure 4 (a) and (b).

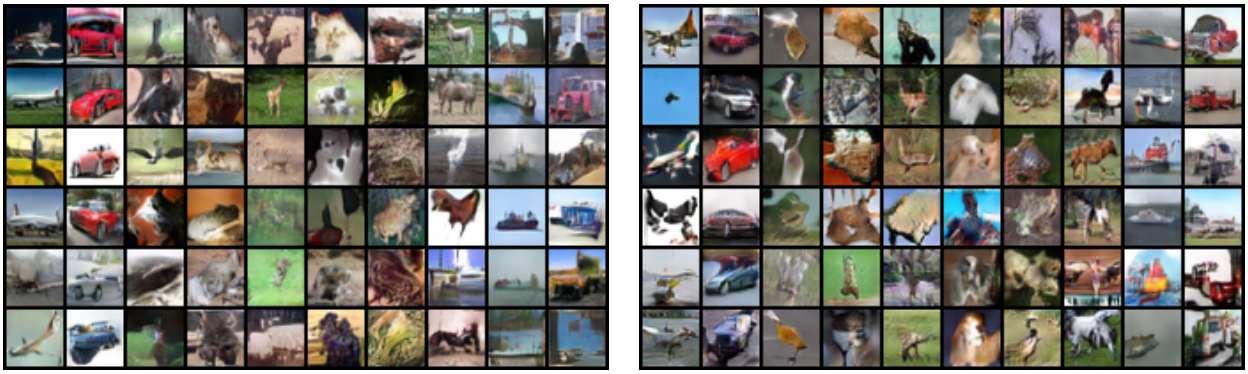
We see that the models trained with CGN achieve much better values for IS and FID than those with CBN. In the case of batch size 64, CGN performs very closely as CBN, and in all other cases of smaller batch sizes, CGN clearly outperforms CBN. This indicates the heavy dependence of CBN on batch size, which prevents it from generalizing well. Thus, we believe CGN is a better candidate as a module to use in a deep neural network-based generative models than CBN.

The SNGAN model architecture consists of a series of residual blocks followed by bn-relu-conv layers. Each residual block contains two bn-relu-conv modules, with an optional upsampling layer. Since the architectures of more recent models such as SAGAN (Zhang et al. 2018a) and BigGAN (Brock, Donahue, and Simonyan 2019) are very similar to that of the one we used, it is likely that the conclusions we draw from the SNGAN experiments transfer to them.

Conclusion

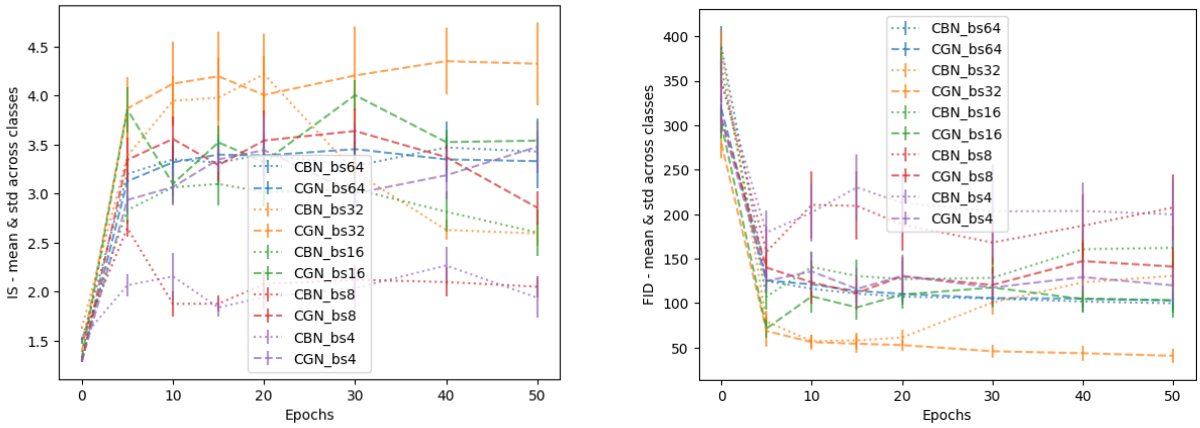
Because the performance of CBN heavily depends on the batch size and on how well training and test statistics match, we investigate the use of CGN as a potential alternative for CBN.

We experimentally show that the effect of this substitution is task-dependent, with performance increases in some VQA tasks that focus on systematic generalization and comparable performance in few-shot learning. In conditional image generation, we show that CGN can be trained with significantly smaller batch sizes than CBN, sometimes even with increased performance as measured by the IS and FID metrics. CGN's simpler implementation, its consistent behaviour during training and inference time, as well as its independence from batch sizes, are all good reasons to explore its adoption instead of CBN in tasks that require out-of-distribution generalization. That being said, further analysis is required to be able to confidently suggest one method over the other. For instance, a hyperparameter search for each of the normalization methods would be required to provide a more detailed performance comparison. As shown in our conditional image generation experiments, CGN is more amenable to training with small batch sizes. This suggests investigating applications in do-



(a) CBN (b) CGN

Figure 3: Samples from models trained with different normalization techniques. The images in each column belong to the same class, ordered as 'airplane', 'automobile', 'bird', 'cat', 'deer', 'dog', 'frog', 'horse', 'ship', 'truck'. Samples are not cherry-picked.



(a) IS (b) FID

Figure 4: (a) Inception Score (IS) (Salimans et al. 2016) (higher is better), (b) Frechét Inception Distance (FID) (Heusel et al. 2017) (lower is better), each averaged across all classes for one run each of CBN- and CGN-based models, for different batch sizes — 64, 32, 16, 8, 4

mains where efficient large-batch training is non-trivial, such as medical imaging or video processing.

Lastly, since some of the success of BN (and consequently also CBN) can be attributed to the regularization effect introduced by noisy batch statistics, it seems worthwhile to explore combinations of CGN with additional regularization as suggested for GN by Wu and He (2018). This is also motivated by recent successful attempts at replacing (unconditional) BN with careful network initialization (Zhang, Dauphin, and Ma 2019), which relies on additional regularization (Zhang et al. 2018b) to match generalization performance.

References

Abadi, M., et al. 2015. TensorFlow: Large-scale machine learning on heterogeneous systems. Software available from tensorflow.org.

Andreas, J.; Rohrbach, M.; Darrell, T.; and Klein, D. 2016. Neural module networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Antol, S.; Agrawal, A.; Lu, J.; Mitchell, M.; Batra, D.; Lawrence Zitnick, C.; and Parikh, D. 2015. Vqa: Visual question answering. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*.

Bahdanau, D.; Murty, S.; Noukhovitch, M.; Nguyen, T. H.; de Vries, H.; and Courville, A. 2018. Systematic generalization: What is required and can it be learned? *arXiv preprint arXiv:1811.12889*.

Barratt, S., and Sharma, R. K. 2018. A note on the inception score. *CoRR* abs/1801.01973.

Brock, A.; Donahue, J.; and Simonyan, K. 2019. Large scale gan training for high fidelity natural image synthesis. *International Conference on Learning Representations*.

Cheng, J.; Dong, L.; and Lapata, M. 2016. Long short-term memory-networks for machine reading. *arXiv preprint arXiv:1601.06733*.

Cho, K.; Van Merriënboer, B.; Bahdanau, D.; and Ben-

gio, Y. 2014. On the properties of neural machine translation: Encoder-decoder approaches. *arXiv preprint arXiv:1409.1259*.

Dalal, N., and Triggs, B. 2005. Histograms of oriented gradients for human detection. In *international Conference on computer vision & Pattern Recognition (CVPR'05)*, volume 1. IEEE Computer Society.

De Vries, H.; Strub, F.; Mary, J.; Larochelle, H.; Pietquin, O.; and Courville, A. C. 2017. Modulating early visual processing by language. In *Advances in Neural Information Processing Systems*.

Dumoulin, V.; Shlens, J.; and Kudlur, M. 2017. A learned representation for artistic style. *International Conference on Learning Representations (ICLR)*.

Galloway, A.; Golubeva, A.; Tanay, T.; Moussa, M.; and Taylor, G. W. 2019. Batch normalization is a cause of adversarial vulnerability. *arXiv preprint arXiv:1905.02161*.

Goodfellow, I. J.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A. C.; and Bengio, Y. 2014. Generative adversarial nets. In *Advances in Neural Information Processing Systems*.

Gulrajani, I.; Ahmed, F.; Arjovsky, M.; Dumoulin, V.; and Courville, A. C. 2017. Improved training of wasserstein gans. In *Advances in Neural Information Processing Systems*.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*.

Heusel, M.; Ramsauer, H.; Unterthiner, T.; Nessler, B.; and Hochreiter, S. 2017. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *Advances in Neural Information Processing Systems*.

Ioffe, S., and Szegedy, C. 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning (ICML)*.

Jiang, X.; Havaei, M.; Varno, F.; Chartrand, G.; Chapados, N.; and Matwin, S. 2018. Learning to learn with conditional class dependencies.

Johnson, J.; Hariharan, B.; van der Maaten, L.; Fei-Fei, L.; Zitnick, C. L.; and Girshick, R. 2017. Clevr: A diagnostic dataset for compositional language and elementary visual reasoning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Kahou, S. E.; Michalski, V.; Atkinson, A.; Kádár, Á.; Trischler, A.; and Bengio, Y. 2017. Figureqa: An annotated figure dataset for visual reasoning. *Workshop in the International Conference on Learning Representations*.

Kingma, D. P., and Ba, J. 2014. Adam: A method for stochastic optimization. *International Conference on Learning Representations (ICLR)*.

Krizhevsky, A. 2009. Learning multiple layers of features from tiny images.

Lei Ba, J.; Kiros, J. R.; and Hinton, G. E. 2016. Layer normalization. *arXiv preprint arXiv:1607.06450*.

Malinowski, M., and Fritz, M. 2014. A multi-world approach to question answering about real-world scenes based on uncertain input. In *Advances in neural information processing systems*.

Mirza, M., and Osindero, S. 2014. Conditional generative adversarial nets. *CoRR* abs/1411.1784.

Miyato, T.; Kataoka, T.; Koyama, M.; and Yoshida, Y. 2018a. Spectral normalization for generative adversarial networks. *ArXiv* abs/1802.05957.

Miyato, T.; Kataoka, T.; Koyama, M.; and Yoshida, Y. 2018b. Spectral normalization for generative adversarial networks. *International Conference on Learning Representations (ICLR)*.

Nair, V., and Hinton, G. E. 2010. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th international conference on machine learning (ICML-10)*.

Odena, A.; Olah, C.; and Shlens, J. 2017. Conditional image synthesis with auxiliary classifier gans. In *International Conference on Machine Learning (ICML)*.

Oreshkin, B.; López, P. R.; and Lacoste, A. 2018. Tadam: Task dependent adaptive metric for improved few-shot learning. In *Advances in Neural Information Processing Systems*.

Parikh, A. P.; Täckström, O.; Das, D.; and Uszkoreit, J. 2016. A decomposable attention model for natural language inference. *arXiv preprint arXiv:1606.01933*.

Paszke, A.; Gross, S.; Chintala, S.; Chanan, G.; Yang, E.; DeVito, Z.; Lin, Z.; Desmaison, A.; Antiga, L.; and Lerer, A. 2017. Automatic differentiation in PyTorch. In *NeurIPS Autodiff Workshop*.

Perez, E.; Strub, F.; De Vries, H.; Dumoulin, V.; and Courville, A. 2018. Film: Visual reasoning with a general conditioning layer. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Ravi, S., and Larochelle, H. 2016. Optimization as a model for few-shot learning.

Russakovsky, O., et al. 2015. Imagenet large scale visual recognition challenge. *International journal of computer vision* 115(3).

Salimans, T.; Goodfellow, I.; Zaremba, W.; Cheung, V.; Radford, A.; Chen, X.; and Chen, X. 2016. Improved techniques for training gans. In *Advances in Neural Information Processing Systems*.

Szegedy, C.; Vanhoucke, V.; Ioffe, S.; Shlens, J.; and Wojna, Z. 2016. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*.

Ulyanov, D.; Vedaldi, A.; and Lempitsky, V. S. 2016. Instance normalization: The missing ingredient for fast stylization. *CoRR* abs/1607.08022.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, Ł.; and Polosukhin, I. 2017. Attention is all you need. In *Advances in neural information processing systems*.

Vinyals, O.; Blundell, C.; Lillicrap, T.; Wierstra, D.; et al. 2016. Matching networks for one shot learning. In *Advances in neural information processing systems*.

Wu, Y., and He, K. 2018. Group normalization. In *Proceedings of the European Conference on Computer Vision (ECCV)*.

Xiang, S., and Li, H. 2017. On the effects of batch and weight normalization in generative adversarial networks. *arXiv preprint arXiv:1704.03971*.

Zhang, H.; Goodfellow, I. J.; Metaxas, D. N.; and Odena, A. 2018a. Self-attention generative adversarial networks. *International Conference on Learning Representations (ICLR)*.

Zhang, H.; Cisse, M.; Dauphin, Y. N.; and Lopez-Paz, D. 2018b. mixup: Beyond empirical risk minimization. In *International Conference on Learning Representations*.

Zhang, H.; Dauphin, Y. N.; and Ma, T. 2019. Fixup initialization: Residual learning without normalization. In *International Conference on Learning Representations*.