Shift and Scale is Detrimental To Few-Shot Transfer

Anonymous Author(s)
Affiliation
Address
email

Abstract

Batch normalization is a common component in computer vision models, including ones typically used for few-shot learning. Batch normalization applied in convolutional networks consists of a normalization step, followed by the application of per-channel trainable affine parameters which shift and scale the normalized features. The use of these affine parameters can speed up model convergence on a source task. However, we demonstrate in this work that, on common few-shot learning benchmarks, training a model on a source task using these affine parameters is detrimental to downstream transfer performance. We study this effect for several methods on well-known benchmarks such as cross-domain few-shot learning (CD-FSL) benchmark and few-shot image classification on miniImageNet. We find consistent performance gains, particularly in settings with more distant transfer tasks. Improvements from applying this low-cost and easy-to-implement modifications are shown to rival gains obtained by more sophisticated and costly methods.

1 Introduction

Advances in few shot learning (FSL) have allowed deep learning models to adapt data representations with just a few labelled samples from novel classes [21, 19, 3]. Throughout the literature, most FSL methods make use of Batch-Norm (BN) [8] layers when training on source datasets to speed up model convergence and overcome the problem of internal covariate shift. Adding BN layers to deep neural networks stabilizes the distribution of layer inputs by controlling the mean and variance of these distributions, leading to improved performance across a plethora of computer vision tasks. Despite the ubiquity of BN layers, specially in deep convolutional neural networks (CNN), the source of their effectiveness is still poorly understood. Furthermore, recent studies have argued that this effectiveness may not be due to reducing the covariate shift, as originally believed [18].

Computationally, BN layers [8] consist of two steps. First, each input is normalized according to the mean and standard deviation across the spatial dimensions of a mini-batch for each of its channels. In this paper, we refer to this first process as "Feature-Normalization", or FN. These normalized inputs are then scaled and shifted by the trainable coefficient γ and bias β (the affine parameters). From the perspective of domain adaptation, Li et al. [12] state that "the label related knowledge is stored in the weight matrix of each layer, whereas domain related knowledge is represented by the statistics of the Batch-Normalization". In other words, affine parameters are responsible for adjusting the statistics of Batch-Norm (domain knowledge) according to the underlying correlation among features (label knowledge). This definition gives rise to a reasonable question: Are the affine parameters still useful when facing a novel target domain? From fig. 1, it is apparent that in the absence of these affine parameters, the preceding weight layers could substitute their role, resulting in fairly similar distributions from the output of BN layers. The auxiliary benefits of these affine parameters to weight layers has been studied in a recent work [4]. However, the negative effect of this biased adaptation to the training labels, when facing novel labels of a new target domain, has not been explored so far.

Submitted to 35th Conference on Neural Information Processing Systems (NeurIPS 2021). Do not distribute.

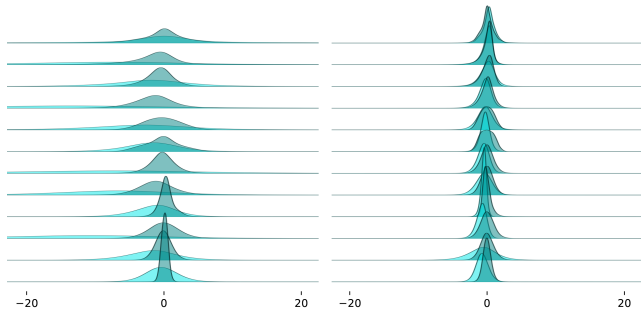

Figure 1: Distribution of input (left) and output (right) from the BN (light green) and FN (dark green) layers of a pre-trained ResNet10. All spatial values per channels are aggregated, resulting in one distribution per layer.

In this work, we investigate the effect of replacing the BN layers with Feature-Normalization on the generalizability of deep CNNs within the setting of cross domain few-shot transfer. In subsequent sections, we provide a formal definition of Feature-Normalization and then evaluate the effect of this replacement on some well-known cross domain few-shot methods and datasets.

2 Methods

Let S be a batch of labelled examples $\{(x_i^s, y_i^s)\}_{i=1}^N$ of size N from a source domain $D^s$and Θ be a deep convolutional neural network consisting of L layers with weight matrices θ^l where l represents the layer index. If h represents the intermediate features of Θ for layer l, the Feature-Normalization layer at layer l is computed for each channel and can be defined as[1]:

$$\mathrm{FN}(h_c) = \frac{h_c - \mu_c}{\sqrt{\sigma_c^2 + \epsilon}}.$$

Here, c is the channel, while μ_c and σ_c are the first and second moments of h_c respectively defined as:

$$\mu_c = \frac{1}{NHW} \sum\nolimits_{n,h,w} h_{nchw}, \quad \sigma_c = \sqrt{\frac{1}{NHW} \sum\nolimits_{n,h,w} (h_{nchw} - \mu_c)^2}$$

where H and W are the spatial dimensions of h_c.

3 Related work

Few shot learning Research on FSL has typically been made on the basis that there is little domain shift between the source and novel classes [3, 15, 20, 9, 5, 2, 21]. Recently, research has been carried out to tackle FSL settings where the domain gap between the source and target data is large [14]. Work in semi-supervised FSL [16, 10, 17, 22] utilizes unlabelled data to train the softmax classifier in the evaluation stages. Advances in self-training and self-supervised learning have led to promising solutions for FSL tasks with large domain gaps. A notable state-of-the-art approach in the setting of distant tasks is STARTUP [14], which employs a combination of self-training and self-supervised learning components for cross domain few-shot learning (CDFSL).

Batch-normalization BN layers enable the training of deeper networks while speeding up model convergence [8] and producing a slight regularization effect [13]. The authors of Batch-Norm [8] claim that it reduces internal covariate shift under the assumption that the normalization of features alleviates the dramatic changes in distribution to a weight layer's input. This assumption was cast into uncertainty in [18] where the authors manually induced internal covariate shift post batch-normalization to find minimal changes in training time. Further studies show that BN layers smoothen the optimization process [18] and help prevent the exploding gradients problem [1].

[1]For the sake of simplicity, and consistency with the current architectures we implement the Feature-Normalize layer using the available standard Batch-Norm modules with disabled affine parameters.

Li et al. [11] had used BN layers towards domain adaptation in their AdaBN method. This method relies on the assumption that data features from different domains will be transformed into representations with similar distributions. In image classification tasks where the source and target data are from the same distribution, Frankle et al. [4] highlight the expressive powers of the BN affine parameters. They conduct experiments which show that these affine parameters play a positive role towards improving model performance. However, this work does not take into consideration settings where there is a distributional gap between the training and target labels. In this paper, we explore the role of affine parameters towards the generizability of few shot learners in the presence of a distributional shift between the source training and target data.

4 Experiments

The experiments are conducted on the publicly available CDFSL benchmark [6], STARTUP [14] codes and provided datasets. For fairness and simplicity, in this work, we follow the same evaluation setting used for experiments on the CDFSL benchmark [6] and STARTUP [14]. The optimizer settings and learning rate used in this work follows STARTUP. The Baseline is standard transfer learning trained for 1000 epochs on miniImageNet with a batch size of 128. AdaBN utilizes the Baseline model and adapts to the novel domain using 20% of unlabelled samples from the target dataset, trained for an additional 10 epochs. STARTUP's teacher model is trained for 400 epochs on miniImageNet and its student model is trained for 1000 epochs on unlabelled samples from 20% of each target dataset, both using a batch size of 256. The remaining 80% of target datasets are utilized for fine-tuning, as described in section 4.1. All methods make use of the ResNet-10 architecture [7]. Results reported in table 2 are evaluated on samples from 900 unseen classes of ImageNet. In all experiments, the weights of the feature extractor are frozen after the representation learning phase. A linear classifier is then trained on the support set of the target dataset and the model is evaluated on the query set. All the experiments were carried out using the Tesla V100 SXM2 16 GB GPU.

4.1 Benchmark

We perform our experiments on the challenging CDFSL benchmark [6]. MiniImageNet [21] is used as the base representation learning dataset. The target dataset is comprised of 4 datasets, each with samples from very different distributions: EuroSAT (satellite imagery to determine land usage), CropDiseases (plant images to identify botanical diseases), ChestX (chest X-rays to determine diagnosis) and ISIC2018 (images of skin abrasions to detect melanoma).

Similar to [6], we perform experiments in an FSL linear classification setting where the support set is composed of 5 classes with k samples per class (5-way k-shot), where $k \in \{1, 5, 20, 50\}$ and the overall scores are an average of accuracies over all target datasets for $k \in \{5, 20, 50\}$. Evaluation is carried out over 600 episodes, 95% confidence intervals with reported mean accuracy. For the experiments on AdaBN and STARTUP, we randomly sample 20% of unlabelled images from novel classes in the target dataset, following a similar setup to that of STARTUP [14]. The remaining samples are used during the inference stage.

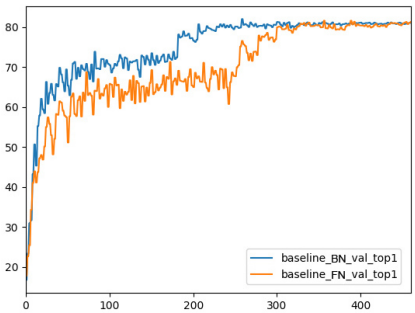

Figure 2: Learning accuracy on validation data for models with BN (blue) and FN (Orange). Despite the steeper rise for the model with BN, the model with FN reaches its best accuracy faster. The accuracy of both models converge to the same level and remain steady.

4.2 Results

Table 1 reports the results of our experiments. Across all datasets, and the 5, 20 and 50 levels of shot (as is in the CDFSL benchmark), the average performance of the models configured with FN exceeds that of BN models across all four datasets. FN outperforms BN models, notably surpassing the BN baseline by nearly 5 points for 1-shot classification on CropDisease. It can be observed that simply configuring the Baseline model with FN obtains results that rival (within error bars) the more

Table 1: The results of few-shot learning methods under extreme distribution shift.

	Baseline		AdaBN		STARTUP	
	BN	FN (ours)	BN	FN (ours)	BN	FN (ours)
	5-WAY, 1-SHOT					
EuroSAT	59.18±0.85	**63.54±0.84**	60.97±0.84	**64.95±0.84**	63.88±0.84	**64.00±0.88**
CropDisease	68.46±0.87	**73.33±0.83**	71.11±0.83	**73.15±0.82**	**75.93±0.80**	74.56±0.85
ISIC	33.10±0.59	**33.12±0.60**	33.10±0.59	**34.89±0.61**	32.70±0.60	**35.12±0.64**
ChestX	22.54±0.41	**22.72±0.42**	22.35±0.41	**22.49±0.40**	**23.09±0.43**	22.93±0.43
	5-WAY, 5-SHOT					
EuroSAT	79.98±0.37	**81.64±0.58**	81.44±0.63	**83.50±0.59**	82.29±0.60	**82.51±0.62**
CropDisease	89.85±0.49	**92.31±0.46**	91.38±0.47	**92.28±0.45**	**93.02±0.45**	92.86±0.43
ISIC	45.50±0.59	**46.50±0.61**	46.77±0.57	**48.76±0.65**	47.20±0.61	**48.54±0.63**
ChestX	26.65±0.42	**26.84±0.45**	26.28±0.41	**26.38±0.43**	26.94±0.44	**27.17±0.44**
	5-WAY, 20-SHOT					
EuroSAT	88.01±0.46	**88.83±0.43**	89.61±0.44	**90.63±0.42**	89.26±0.43	**89.63±0.43**
CropDisease	96.00±0.27	**97.23±0.24**	96.50±0.28	**97.44±0.22**	**97.51±0.21**	97.43±0.23
ISIC	55.60±0.56	**58.55±0.58**	57.66±0.55	**58.94±0.60**	58.60±0.58	**59.98±0.59**
ChestX	31.97±0.43	**32.31±0.46**	**31.26±0.44**	31.24±0.44	33.19±0.46	**33.54±0.46**
	5-WAY, 50-SHOT					
EuroSAT	91.03±0.37	**93.35±0.32**	**92.60±0.35**	92.51±0.34	91.99±0.36	**92.59±0.33**
CropDisease	97.58±0.21	**98.49±0.17**	98.02±0.19	**98.28±0.17**	98.45±0.17	**98.53±0.16**
ISIC	61.40±0.56	**63.76±0.58**	62.87±0.55	**64.98±0.57**	64.20±0.58	**65.90±0.56**
ChestX	**35.28±0.47**	35.22±0.45	34.37±0.47	**35.87±0.47**	36.91±0.50	**37.67±0.47**

Table 2: Evaluation of models trained on miniImageNet applied to novel classes of ImageNet using the classic evaluation protocol [21]

	1-shot	5-shot	20-shot
Baseline BN	54.56 ± 0.84	76.18 ± 0.69	84.53 ± 0.52
Baseline FN (Ours)	**56.48 ± 0.82**	**77.80 ± 0.68**	**85.69 ± 0.51**

complex and computationally expensive STARTUP, which employs a large amount of unlabelled data to bridge the domain gap.

Table 2 reports the performance of the FN and BN configurations on the experiment with the unseen classes of ImageNet. Although the accuracies of both models on validation data are almost the same (fig. 2), in FSL settings where there is a relatively smaller domain shift, FN consistently surpasses BN models by an average of 1.56%. However, from fig. 2, it can be inferred that the FN does not improve performance on non-Few-Shot settings, and the affine parameters indeed aid convergence on in-distribution validation data.

5 Conclusion

Feature-Normalization layer improves the few-shot generalization accuracy on shifted domains by leveraging a lower amount of the model's parameters. By stabilizing the output distribution of convolutional layers, Feature-Normalization improves robustness against distributional shifts. It captures and normalizes the statistical distribution of data features while preventing the affine from overfitting to the training source labels. Feature-Normalization is completely consistent with well known Batch-Norm implementations. It could be easily integrated in existing CNN architectures and used by SOTA methods. It is observed that the proposed normalization technique only helps in few-shot transfer and the effect is more pronounced as the data distribution shifts. This is an initial study and we will consider further experiments on the effect of affine parameters in other few shot benchmarks and methods.

References

[1] Johan Bjorck et al. "Understanding Batch Normalization". In: *Proceedings of the 32nd International Conference on Neural Information Processing Systems*. 2018, pp. 7705–7716.

[2] Wei-Yu Chen et al. "A closer look at few-shot classification". In: *arXiv preprint arXiv:1904.04232* (2019).

[3] Chelsea Finn, Pieter Abbeel, and Sergey Levine. "Model-agnostic meta-learning for fast adaptation of deep networks". In: *International Conference on Machine Learning*. PMLR. 2017, pp. 1126–1135.

[4] Jonathan Frankle, David J Schwab, and Ari S Morcos. "Training batchnorm and only batchnorm: On the expressive power of random features in cnns". In: *arXiv preprint arXiv:2003.00152* (2020).

[5] Spyros Gidaris and Nikos Komodakis. "Dynamic few-shot visual learning without forgetting". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 4367–4375.

[6] Yunhui Guo et al. "A broader study of cross-domain few-shot learning". In: *European Conference on Computer Vision*. Springer. 2020, pp. 124–141.

[7] Kaiming He et al. "Deep residual learning for image recognition". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 770–778.

[8] Sergey Ioffe and Christian Szegedy. "Batch normalization: Accelerating deep network training by reducing internal covariate shift". In: *International conference on machine learning*. PMLR. 2015, pp. 448–456.

[9] Kwonjoon Lee et al. "Meta-learning with differentiable convex optimization". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2019, pp. 10657–10665.

[10] Xinzhe Li et al. "Learning to self-train for semi-supervised few-shot classification". In: *Advances in Neural Information Processing Systems* 32 (2019), pp. 10276–10286.

[11] Yanghao Li et al. "Adaptive batch normalization for practical domain adaptation". In: *Pattern Recognition* 80 (2018), pp. 109–117.

[12] Yanghao Li et al. "Revisiting batch normalization for practical domain adaptation". In: *arXiv preprint arXiv:1603.04779* (2016).

[13] Ping Luo et al. "Towards Understanding Regularization in Batch Normalization". In: *ArXiv* abs/1809.00846 (2019).

[14] Cheng Perng Phoo and Bharath Hariharan. "Self-training for few-shot transfer across extreme task differences". In: *arXiv preprint arXiv:2010.07734* (2020).

[15] Sachin Ravi and Hugo Larochelle. "Optimization as a model for few-shot learning". In: (2016).

[16] Mengye Ren et al. "Meta-learning for semi-supervised few-shot classification". In: *arXiv preprint arXiv:1803.00676* (2018).

[17] Pau Rodrıguez et al. "Embedding propagation: Smoother manifold for few-shot classification". In: *European Conference on Computer Vision*. Springer. 2020, pp. 121–138.

[18] Shibani Santurkar et al. "How does batch normalization help optimization?" In: *Proceedings of the 32nd international conference on neural information processing systems*. 2018, pp. 2488–2498.

[19] Jake Snell, Kevin Swersky, and Richard Zemel. "Prototypical networks for few-shot learning". In: *Proceedings of the 31st International conference on neural information processing systems*. 2017, pp. 4080–4090.

[20] Qianru Sun et al. "Meta-transfer learning for few-shot learning". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2019, pp. 403–412.

[21] Oriol Vinyals et al. "Matching networks for one shot learning". In: *Advances in neural information processing systems* 29 (2016), pp. 3630–3638.

[22] Yikai Wang et al. "Instance credibility inference for few-shot learning". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020, pp. 12836–12845.