Recognizing Instagram Filtered Images with Feature De-Stylization

Zhe Wu,[1*] **Zuxuan Wu,**[2†] **Bharat Singh,**[2] **Larry S. Davis**[2]
[1]Comcast Applied AI Research, [2]University of Maryland,College Park
zhe_wu@comcast.com, {zxwu, bharat, lsd}@cs.umd.edu

Abstract

Deep neural networks have been shown to suffer from poor generalization when small perturbations are added (like Gaussian noise), yet little work has been done to evaluate their robustness to more natural image transformations like photo filters. This paper presents a study on how popular pretrained models are affected by commonly used Instagram filters. To this end, we introduce ImageNet-Instagram, a filtered version of ImageNet, where 20 popular Instagram filters are applied to each image in ImageNet. Our analysis suggests that simple structure preserving filters which only alter the global appearance of an image can lead to large differences in the convolutional feature space. To improve generalization, we introduce a lightweight de-stylization module that predicts parameters used for scaling and shifting feature maps to "undo" the changes incurred by filters, inverting the process of style transfer tasks. We further demonstrate the module can be readily plugged into modern CNN architectures together with skip connections. We conduct extensive studies on ImageNet-Instagram, and show quantitatively and qualitatively, that the proposed module, among other things, can effectively improve generalization by simply learning normalization parameters without retraining the entire network, thus recovering the alterations in the feature space caused by the filters.

Introduction

Convolutional neural networks (CNNs) demonstrate impressive recognition accuracies on standard benchmarks like ImageNet (Deng et al. 2009), even surpassing human-level performance (He et al. 2015) especially when recognizing fine-grained objects; and as a result, these trained models are widely applied to a variety of applications in real-world. However, recent studies have shown CNNs are vulnerable to even small perturbations (Hendrycks and Dietterich 2019) like a Gaussian noise or blur, resulting in significant performance degradation, let alone maliciously injected adversarial noises (Goodfellow et al. 2014; Kurakin, Goodfellow, and Bengio 2016). This raises doubts about whether these models can offer reliable recognition performance, particularly when evaluated on images from social media, where images are generally modified (filtered) extensively by adjusting parameters like brightness, color, and contrast or their combinations. For example, Instagram provides 40 pre-defined filters and users can apply these filters to make photos look appealing with only a few clicks. These filtered images contain various visual effects, incurring far more complicated perturbations compared to Gaussian noises (Hendrycks and Dietterich 2019; Geirhos et al. 2018). Thus, models pretrained on natural images may fail when tested on these filtered images.

Extensive studies have been conducted to improve the generalization (Zhang et al. 2017) of deep neural networks and techniques like Dropout (Srivastava et al. 2014) and BatchNorm (Ioffe and Szegedy 2015) can effectively reduce overfitting. The generalization ability of deep networks is usually measured on a held-out testing set; or related tasks

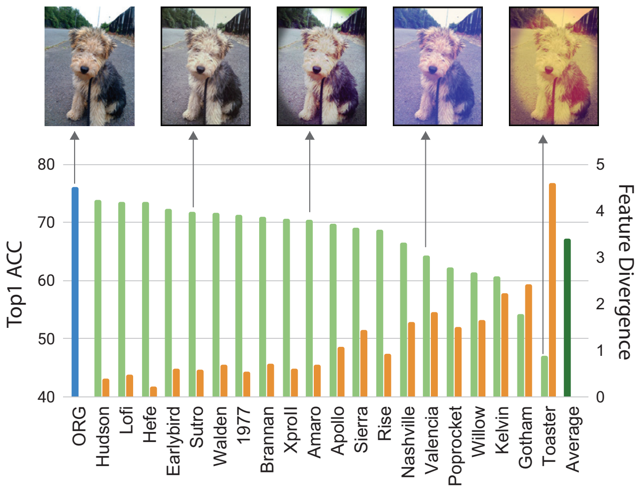

Figure 1: Performance of ResNet50 pretrained on ImageNet (Green) and `Conv5` feature divergence (Orange). X-axis: Different types of Instagram filters. Left-Y-axis: Top1 accuracy; right-Y-axis: Feature Divergence.

*This work was done when the author was at Univ. of Maryland.
†Corresponding author.

using features that are finetuned to be task-specific (Kornblith, Shlens, and Le 2018) or explicitly adapted for distribution alignment (Tzeng et al. 2017). In this paper, we estimate generalization from a different perspective—by testing on filtered images whose appearance is significantly modified but the structure and semantics are preserved. Such filtered images are prevalent on social media and it is critical to develop networks that generalize well on these images.

In light of this, we systematically study the robustness of modern CNN architectures to widely used Instagram filters for image classification, and introduce a simple yet effective approach that helps the generalization of these architectures on filtered images . To begin with, we create a new dataset, referred to as IMAGENET-INSTAGRAM, which contains images transformed from IMAGENET using 20 common Instagram filters—each original image in IMAGENET is applied with these 20 filters, generating 20 copies that share the same semantic content but differ in appearance (See Fig. 2 for an example). We then analyze the performance of several top-performing CNNs on the newly constructed dataset. The results suggest that dramatic changes in appearance lead to huge feature differences compared to those original images, which further result in significant performance drops (cf. "Toaster" and "Sutro" in Fig. 1).

Therefore, we posit that the visual effect brought by filters not only changes the style of original images but also injects style information into feature maps, resulting in shifts from original feature representations. If we can find a way to remove style information in these feature maps, they will be closer to those of the original samples. This is essentially the inverse process of style transfer tasks that aim to add style information into features, typically done with instance normalization (IN) (Ulyanov, Vedaldi, and Lempitsky 2016) to scale and shift feature maps at each channel. Then the question becomes, can we learn a set of parameters that re-normalize feature maps with IN by scaling and shifting to "undo" the changes caused by filters. If so, the performance on filtered images can be improved by simply tuning these parameters to produce re-normalized feature maps.

To this end, we propose a lightweight de-stylization module (DS), which contains a five-layer fully-connected network operating on feature maps encoded by a pretrained VGG encoder. The DS module outputs multiple sets of parameters, each of which is used to scale and shift feature maps in a corresponding IN layer of a base network. The DS module can be readily used in networks like IBN (Pan et al. 2018a) where IN layers are used for recognition tasks. To further extend the DS module to modern architectures without IN layers, we introduce a generalized DS (gDS), which performs IN with the de-stylization module on feature maps in modern networks but the normalized feature maps are further shortcut by skip connections. Such a design ensures style information in feature maps caused by filters can be removed with learned normalization parameters without destroying the optimized feature maps in the base network. We conduct extensive results on the newly proposed IMAGENET-INSTAGRAM dataset, and we demonstrate that both DS and gDS can effectively improve generalization when applying pretrained models on filtered images by simply learning normalization parameters without retraining the whole network and gDS is compatible with modern CNN architectures even without IN layers. Our qualitative results also suggest that gDS can indeed transform features of filtered image to be similar to those before filtering.

Related Work

Corruption Robustness. There are a few recent studies investigating the robustness of deep neural networks to corrupted or noisy inputs (Hendrycks and Dietterich 2019; Geirhos et al. 2018). Heydrycks *et al.* (Hendrycks and Dietterich 2019) introduce two variants of IMAGENET to benchmark the robustness of deep models to common corruptions and perturbations. The results suggest that deep models demonstrate instabilities to even small changes in input distributions. Geirhos *et al.* (Geirhos et al. 2018) study the robustness of humans and several CNNs on different types of degraded images. Our work differs from these methods as we focus on filtered images that contain a series of sophisticated and carefully designed transformations as opposed to basic transformations like Gaussian noise and rotations in (Hendrycks and Dietterich 2019; Geirhos et al. 2018). In addition, filtered images are created intentionally to make images aesthetically pleasing and thus improving generalization on these filtered images enjoys wider applications.

Domain Adaptation. Our work is also related to domain adaptation, or transfer learning, which aims to adapt a learned model to new tasks by aligning the feature distributions of the source task with that of the target task. Existing approaches usually minimize distances such as Maximum Mean Discrepancy (MMD) (Saito et al. 2018), first-order and second order statistics (Sun and Saenko 2016) or make features indistinguishable with adversarial loss functions (Ganin et al. 2016; Tzeng et al. 2017). However, these methods require training or finetuning the majority of weights in networks, which is computationally expensive particularly when using adversarial loss functions (Arjovsky and Bottou 2017). In contrast, we focus on how to improve generalization with a lightweight module. Recently, Li *et al.* (Li et al. 2016) introduce Adaptive Batch Normalization (AdaBN) that applies a trained network to a target domain without changing the model weights. AdaBN collects the statistics of Batch Normalization layer on target domains before final testing, and use the target domain statistics for normalization. However, it requires the distribution of test samples before testing, and is designed for adaptation to a unique domain. Our proposed approach, on the other hand, does not assume access to testing data beforehand (only access to the filtering function is needed), and performs normalization based on the appearance of the testing sample and thus can be applied to a mixture of different domains at the same time.

Image Synthesis for De-stylization. Recent advances in image-to-image translation provide a way to remove filtering effect such that de-stylized images can be directly input into the original model learned on natural images. In particular, image-to-image translation aims to generate images from a source domain in the style of a target domain. This can be achieved using generative models that enforce cycle-consistency (Zhu et al. 2017; Huang et al. 2018) or neural style transfer algorithms (Gatys, Ecker, and Bethge 2016; Huang and Belongie 2017). To remove the filtering effect with generative models is hard as adversarial loss functions are difficult to optimize and it requires modeling a many-to-one mapping if there are multiple source domains. De-stylization with style transfer is also challenging since it is difficult to select images from the source as references. In addition, both methods require training an additional large model to generate images while our approach aligns features maps with a lightweight network.

Feature Normalization. Feature normalization is an essential component in modern deep CNNs. Batch Normalization (Ioffe and Szegedy 2015) is widely used for faster convergence and better performance. Instance Normalization (Ulyanov, Vedaldi, and Lempitsky 2016) helps achieve good image stylization performance, as the channel-wise feature statistics are shown to contain sufficient style information (Ulyanov et al. 2016). In contrast to these methods operating on feature maps, there are some studies on conditional normalization which modulates feature maps with additional information. Conditional Batch Normalization (Dumoulin, Shlens, and Kudlur 2016) and Adaptive Instance Normalization (AdaIN) (Huang and Belongie 2017) adjust the normalization layer parameters based on external data inputs, and thus are able to achieve image stylization on a diverse set of styles. However, these normalization methods are mainly designed for generative tasks, and have not been used in discriminative models for recognition. In our work, we demonstrate the proposed de-stylization module is applicable to several modern deep networks to improve their generalization on filtered images for recognition.

Imagenet-Instagram

Many social media apps provide a variety of artistic image filters to help users editing the photos. For example, Instagram has 40 pre-defined photo filters. These filters are combinations of different effects such as curve profiles, blending modes, color hues, *etc.*, which make the filtered photos aesthetically appealing. We select 20 commonly used Instagram filters and apply them to each image in IMAGENET, the resulting new dataset is named as IMAGENET-INSTAGRAM. Figure. 2 illustrates one sampled image from IMAGENET and its 20 filtered versions with Instagram filters. As we can see, different Instagram filters generate different photo styles. For example, “Amaro” filter brightens the center of the image and adds vignetting to the border of the image. Some filters like “1977” and “Hefe” adjust the contrast of the image slightly without creating dramatic effects, while other filters like “Gotham” and “Willow”, discard some important information like color.

To evaluate the performance of modern CNN architectures on these filtered images, we run a ResNet50 (He et al. 2016) pretrained from IMAGENET on the validation set of IMAGENET-INSTAGRAM directly. The results are shown in Figure 1. We observe that the average Top-1 accuracy drops from 76.13% on IMAGENET validation set to 67.22% on IMAGENET-INSTAGRAM. Filters like “Gotham” and “Toaster” create significantly different appearances at the same time suffer from drastic performance drop (*eg.*, 21.13 and 29.03 absolute percentage points for “Gotham” and “Toaster” respectively). It is also surprising to see the performance of filters like “1977” which bring slight differences in color also drops by 5%.

To better understand why pre-trained ResNet50 suffers from poor performance on IMAGENET-INSTAGRAM, we analyze the feature divergence (Pan et al. 2018a) (see supplementary material for definition) of IMAGENET and IMAGENET-INSTAGRAM samples. Specifically, we compute features from the `Conv5` layer for images in both IMAGENET and IMAGENET-INSTAGRAM. For each filter type, we compute the feature divergence of `Conv5` between IMAGENET and IMAGENET-INSTAGRAM on validation set. Figure. 1 presents the results. We can clearly see the correlations between feature divergence and the performance on the validation set of IMAGENET-INSTAGRAM—large feature divergence translate to lower accuracies (see “Toaster”, “Gotham” and “Lord Kelvin”).

Method

Since feature divergence is positively correlated with performance drop, it would be ideal to reduce such mismatch induced by applying filters—the removal of style information encoded in feature maps. This is similar in spirit to style transfer tasks but in the reversed direction; style transfer approaches incorporate style information with instance transformation using a set of affine parameters that are either learned (Ulyanov, Vedaldi, and Lempitsky 2016) or computed from another image (Huang and Belongie 2017). Such an operation simply scales and shifts feature maps to add style information, which thus motivates us to invert this process in order to remove the style information. To this end, we introduce a lightweight de-stylization module (DS), which predicts a set of parameters used to normalize feature maps and hence further remove encoded style information therein. Then we discuss how to easily plug the module into modern architectures such that generalization of these networks can be improved on filtered images.

A lightweight de-stylization module (DS). As mentioned earlier, style transfer pipelines usually rely on instance normalization (IN) to normalize features (Ulyanov, Vedaldi, and Lempitsky 2016; Huang and Belongie 2017). In particular, IN normalizes features per channel for each sample separately using mean and variance computed in each channel. Denote the ith channel of feature map $\boldsymbol{x}$ as $\boldsymbol{x}_i$, the mean and variance of $\boldsymbol{x}_i$ as $\mu(\boldsymbol{x}_i)$ and $\sigma^2(\boldsymbol{x}_i)$, then the IN operation is defined as:

$$\boldsymbol{y}_i = \gamma_i \cdot \frac{\boldsymbol{x}_i - \mu(\boldsymbol{x}_i)}{\sigma(\boldsymbol{x}_i)} + \beta_i \tag{1}$$

$$\boldsymbol{y}_i = \gamma_i \cdot \frac{\boldsymbol{x}_i - \mu_B}{\sigma_B} + \beta_i \tag{2}$$

where γ_i and β_i are affine parameters for channel i. By learning a set of affine paramters γ and β, IN facilitates the transfer of the original image to a different style for image

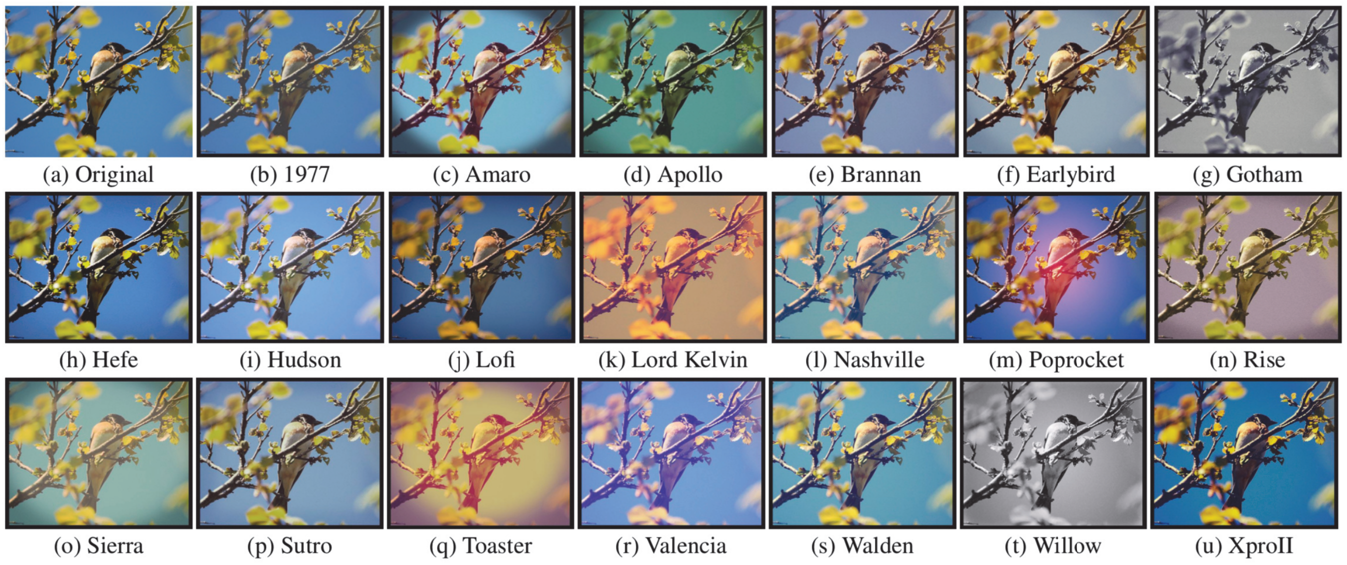

Figure 2: A sampled image from IMAGENET and its 20 Instagram filtered versions.

synthesis. A recently proposed IBN (Pan et al. 2018a) further demonstrates IN layers can be used in discriminative tasks like classification, by simply replacing half of BN layers in a ResNet BottleNeck with IN layers, and demonstrates good generalization across different domains.

Filtering a realistic image with an Instagram filter changes its intermediate feature maps when passing it through the CNN, and such changes will further lead to significant feature divergence as shown in Fig. 1. Given that IN layers can encode style information into feature maps with affine transformations, a natural question to ask is: Can we simply finetune the IN layers to obtain a different set of affine parameters that are able to remove style information in feature maps caused by applied filters? This allows a network to quickly generalize to filtered images without re-training the entire network. However, finetuning IN parameters indicates the same set of affine parameters of each channel are shared by all images, which might be viable if we are targeting at a single type of filter rather than 20 different filters.

Recall our goal is to tune IN parameters that are tailored for each filter, such that for each type of filtered image, the IN operation can successfully undo the changes in feature maps caused by input perturbations. This is the inverse process of *arbitrary* style transfer, where the feature maps from a source image is normalized using different affine parameters based on the styles to be transferred. Thus, we build upon adaptive instance normalization (AdaIn), which enables the transfer of the style of an arbitrary image to a source image (Huang and Belongie 2017). Formally, AdaIN is defined as follows:

$$\text{AdaIN}(\boldsymbol{x}_i, \boldsymbol{z}_m, \boldsymbol{z}_v) = \boldsymbol{z}_{m,i} \frac{\boldsymbol{x}_i - \mu(\boldsymbol{x}_i)}{\sigma(\boldsymbol{x}_i)} + \boldsymbol{z}_{v,i}, \tag{3}$$

where each feature map $\boldsymbol{x}_i$ is normalized separately, and then scaled and shifted using the corresponding mean $\boldsymbol{z}_{m,i}$ and variance $\boldsymbol{z}_{v,i}$ of feature maps $\boldsymbol{z}$ of a target image. In style transfer tasks (Huang and Belongie 2017), feature maps $\boldsymbol{z}$ are computed with a fixed VGG encoder and used for normalization for only once before generating an image. In contrast, we wish to adaptively normalize all Instance Normalization layers in a network to fully recover changes caused by filters.

To this end, we introduce a lightweight de-stylization module, which generates the affine parameters used for instance normalization in all IN layers. In particular, suppose there are L IN layers in a network, we use a five-layer fully connected network to map $\boldsymbol{z}$, the feature maps encoded by an VGG encoder, denoted as $H(\cdot)$, to L vectors $\{\boldsymbol{y}_l\}_{l=1}^{L}$, where each vector $\boldsymbol{y}_l \in \mathbb{R}^{2C_l}$ is used to normalize the corresponding IN layer of C_l channels with Eqn. 3. This is achieved by splitting $\boldsymbol{y}_l$ into two parts $\boldsymbol{y}_l = [\boldsymbol{y}_{lm}, \boldsymbol{y}_{lv}]$, with $\boldsymbol{y}_{lm}$ and $\boldsymbol{y}_{lv}$ denoting the predicted mean and variance for normalization. The first four fully-connected layers are used for transformation, denoted as $\mathcal{F}(\cdot)$. The final layer contains L heads, denoted as $\{f_l(\cdot)\}_{l=1}^{L}$, with each head corresponding to one of the L IN layers in the network. Formally, we define de-stylization module for layer l as $g_l(\cdot)$, then:

$$\boldsymbol{y}_l = g_l(\boldsymbol{z}) = f_l(\mathcal{F}(\boldsymbol{z})) \tag{4}$$

where $\boldsymbol{z} = H(I)$, I is the input Image. The DS module is illustrated in Fig. 3.

Extension to modern architectures. We are now able to generate IN parameters at each IN layer in the network to shift back feature maps, however an important question remains. IN layers are mainly used in style transfer networks, and only IBN has explored the use of IN layers for standard visual recognition tasks (Pan et al. 2018b), which significantly limits the applicability of the lightweight destylization module to state-of-the-art architectures. Fortunately, modern networks, differing in the number of layers and types of layers used, are similar to the seminal

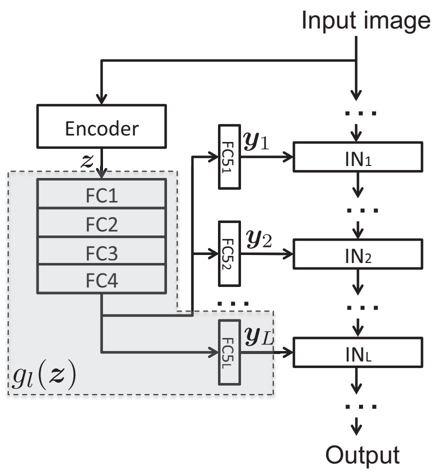

Figure 3: Our proposed framework, with DS module highlighted.

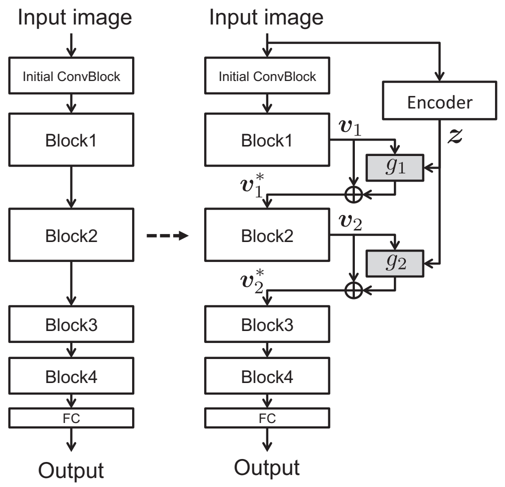

Figure 4: Generalized de-stylization. Left: Orginal 5-block CNN prototype. Right: gDS applied on Block1 and Block2 of the CNN.

AlexNet (Krizhevsky, Sutskever, and Hinton 2012) in design. They contain five blocks of convolution layers, topped by several fully connected layers for classification; and the blocks are different among different architectures. For example, ResNet50 contains BottleNeck blocks while Denseblocks in DenseNet (Huang et al. 2017). As a result, we can plug the output head of the de-stylization module after these convolutional blocks. We denote $\boldsymbol{v}_k$ as the feature maps from the k-th block in a network, and before it is sent to the next block, we perform the normalization with Eqn. 4, the outputs are defined as $g_k(\boldsymbol{v}_k)$. Doing this directly might destroy the feature maps in original networks, which are optimized without any IN layers. To mitigate this issue, we further shortcut these layers with skip connections, and now features sent to the next block become:

$$\boldsymbol{v}_k^* = g_k(\boldsymbol{v}_k) + \boldsymbol{v}_k \tag{5}$$

In this case, if the learned affine parameters are set to zero, then there is simply no normalization. And the network will degrade to the original network. We name the proposed extension as generalized de-stylization (gDS), illustrated in Fig. 4.

Discussions. The design of extending the de-stylization module with skip connections allows the model to remove style information in feature maps brought by applied filters and at the same time without hurting originally optimized features. Consequentially, we can simply optimize weights in the de-stylization module without re-training the base network. With less than 10% parameters compared to an entire network needed for finetuning, the module can be learned efficiently.

Experiments

We first study the the robustness of pretrained networks on IMAGENET-INSTAGRAM, and we get the upper bound performance by finetuning on these filtered images since their labels are readily available. Then we show the effectiveness of proposed DS and its generalized version gDS. After that, we show the performance when only a limited number of class samples are available, followed by some ablation studies and qualitative analysis.

Implementation details. For both training and finetuning, we set the number of epochs to be 15 and use SGD as the optimizer. The learning rate is set to 0.001 and is decreased by a factor of 10 after 10 epochs. All experiments are conducted on NVIDIA 4 Quadro P6000 GPUs. We use a VGG pretrained on IMAGENET as the encoder of DS, and we fixed its weights for all experiments. To test the performance of DS, we use an IBN with a pretrained ResNet50 as its backbone since it is the only network that contains IN layers used for recognition tasks. We compute Top1/Top5 accuracy for each type of filters in IMAGENET-INSTAGRAM and report the mean accuracies across all filters. For gDS, we only perform normalization at the end of `Conv1` and `Conv2`, as feature divergence caused by appearance changes is large in these layers.

Robustness of pretrained networks. Table. 1 presents the results of a pretrained ResNet50 and IBN model when applied to IMAGENET-INSTAGRAM. We can see significant performance drops for both ResNet50 and IBN. For example, Top1 accuracy dropped by 8.92 and 7.89 (absolute percentage) separately. The degradation in accuraries of IBN is less severe than ResNet50, confirming the fact that IN layers can indeed help nomalize style information.

	IMAGENET	IMAGENET-INSTAGRAM
Method	Top1/Top5 acc	Top1/Top5 acc
Resnet50	76.13/ 92.93	67.21/ 87.62
IBN	**77.44/93.69**	**69.55/ 89.32**

Table 1: Performance of pretrained models on IMAGENET-INSTAGRAM. Notice the significant performance drop.

Upper bound by finetuning. Since the semantics are preserved after applying filters, we finetune both ResNet50 and IBN on IMAGENET-INSTAGRAM with images in different types of filters together, denoted as ResNet50-ft and IBN-ft. Finally, we finetune ResNet50 for each filter type separately and refer this method as finetuning upperbound (UB). The results are summarized in Table. 2

It is worth mentioning that training models with all data is extremely computationally expensive and time consuming (IMAGENET-INSTAGRAM is $20\times$ size of IMAGENET). Besides, it also requires lots of space to store the data. Thus, we randomly select 10% of images from each object category in the IMAGENET training set, and transform each image with a random Instagram filter. As a result, we generated a mini version of IMAGENET-INSTAGRAM training set, which we named as IMAGENET-INSTAGRAM-mini. IMAGENET-INSTAGRAM-mini is only $1/200$ size of IMAGENET-INSTAGRAM. There are only around 6 images per object category for each Instagram filter type. We finetune ResNet50 and IBN using IMAGENET-INSTAGRAM-mini and show the results in the third column of Table. 2

	IMAGENET-INSTAGRAM	mini
Method	Top1/Top5 acc	Top1/Top5 acc
ResNet50-ft	74.52/ 92.08	72.53/ 91.02
IBN-ft	75.47/ 92.71	73.23/91.53
ResNet50-UB	75.62/ 92.64	-/ -

Table 2: Results of pretrained models *vs.* finetuned models on IMAGENET-INSTAGRAM and IMAGENET-INSTAGRAM-mini ($1/200$ size of IMAGENET-INSTAGRAM).

We can see with the entire IMAGENET-INSTAGRAM training set, simply finetuning can improve the Top1 accuracy from 67.21% to 74.52% and from 69.55% to 75.47% for ResNet50 and IBN, separately. Besides, by comparing the performances of ResNet50-ft and ResNet50-UB, we can see finetuning together is almost as good as finetuning separately, but finetuning separately is less practical as it requires 20 separate models. Furthermore, with IMAGENET-INSTAGRAM-mini as the training set, the Top1 accuracy of ResNet50 and IBN could improve by 5.32% and 3.68% compared to pretrained models. Although the Top1 accuracy is still 2% less than finetuning with the entire IMAGENET-INSTAGRAM, it is much more computationally feasible. Therefore, we report results on IMAGENET-INSTAGRAM-mini instead of the entire IMAGENET-INSTAGRAM in the remaining of the paper.

Effectiveness of DS. We evaluate the effectiveness of DS when plugged into an IBN on IMAGENET-INSTAGRAM-mini. In addition to comparing with pretrained ResNet50 and IBN, we also compare with IBN-IN, in which only IN parameters are finetuned while other weights fixed. Besides the mean accuracies on all 20 filters on IMAGENET-INSTAGRAM, we also show the averaged performance on 10 filters that lead to the most significant drops of ResNet50, named as Hard 10. The results are shown in Table. 3. We observe that the proposed DS achieves the best performance. On IMAGENET-INSTAGRAM-mini, IBN-IN improves the Top 1 accuracy of IBN by 1.2%, which demonstrates that simply finetuning IN can help generalization slightly. DS further improves the performance of IBN-IN by 1.2%. The effectiveness of DS is clearly visible in the subset of 10 hard filters, where DS improves by 5% over Top-1 accuracy compared with IBN.

Method	All 20 Top1/Top5 acc	Hard 10 Top1/Top5 acc
ResNet50	67.21/ 87.62	62.41/ 84.471
IBN	69.55/ 89.32	65.19/ 86.71
IBN-IN	70.73/ 90.08	67.68/ 88.39
DS	**71.96/ 90.93**	**70.31/ 90.06**

Table 3: Performances of DS on IMAGENET-INSTAGRAM-mini. Our proposed DS achieves best results. The improvement on Hard 10 filters is significant.

Effectiveness of gDS. We now investigate if we can extend gDS to modern architectures, by using three top-performing CNNs, *i.e.*, ResNet50, DenseNet121 (Huang et al. 2017) and SEResNet50 (Hu, Shen, and Sun 2018), whose Top1 accuracies on IMAGENET are 76.13%, 74.47% and 77.61%, respectively. We apply the gDS at the end of Block1 and Block2, as shown in Figure. 4. To be specific, the applied positions are the end of `Conv2_3` and `Conv3_4` in ResNet50, the end of DenseBlock1 and DenseBlock2 in DenseNet121, and the end of `Conv2_3` and `Conv3_4` in SeResNet50.

In addition to showing the results of pretrained networks and gDS, we also consider a simpler form of gDS—ResIN—which adds IN layers shortcut by skip connections at the same position of gDS to ResNet50. As in gDS, we only train the parameters of IN layers while keeping the remaining weights of the network fixed. The difference between ResIN and gDS is that, affine parameters of IN layers are conditioned on additional information. The results are summarized in Figure 5.

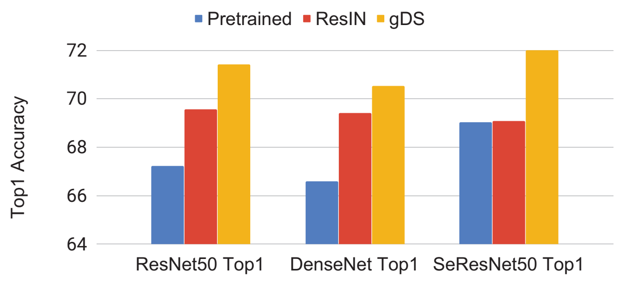

Figure 5: Performances of proposed gDS on different CNN architectures. We compare pretrained model, Finetuning ResIN and gDS for each CNN architecture.

We can see that, gDS outperforms pretrained models by $3\% - 4\%$ for all architectures, which verifies that gDS can be applied to modern networks that help the generalization on filtered images. Further, both ResIN and gDS improve the performance of pretrained models, confirming the fact that normalizing feature maps with IN layers is helpful. As gDS normalizes feature maps conditioned on style information in feature maps from input samples, it helps learning separate parameters for different filters. Interestingly, SeResNet50 didn't benefit from ResIN much, but still benefits from gDS by 3%.

We also evaluate the effectiveness of the designed skip-connection. To achieve this, we remove the skip-connection in ResIN and finetune the IN parameters. The resulting Top1 accuracy is only 36.16%, much worse than pretrained ResNet50. This indicates that without the skip-connection, the added normalization layer could even destroy the optimized feature.

The visualization of how gDS removed the style information brought by filters is shown in Fig. 6.

Generalization to unseen classes. We demonstrate parameters used for scaling and shifting learned by gDS can generalize to unseen classes. During training, we randomly select half of the categories from IMAGENET-INSTAGRAM-mini and use them to train the gDS. At test time, we evaluate the learned model on the other half of categories that have never been seen before from IMAGENET-INSTAGRAM validation set, and on all categories as well. We compare with a pretrained ResNet50 and ResNet50-ft under the same settings and the results are shown in Table. 4.

We can see that on unseen categories, the performance of ResNet50 finetuned using filtered images on seen categories decreases, while gDS increases the performance by 4.24% compared to ResNet50. This demonstrates that gDS not only improves the generalization of CNNs on filtered images but also across different categories. When tested on all categories, finetuned ResNet50 improves upon ResNet50 slightly, but still worse than gDS.

Method	Other Half Top1/Top5 acc	All Top1/Top5 acc
ResNet50	67.64/ 87.85	67.25/ 87.67
ResNet50 ft	59.96/ 84.67	68.89/ 89.10
gDS	**71.40/ 90.49**	**71.08/ 90.30**

Table 4: Performance of models learned with only half of the object categories, and evaluated on the other half of the object categories as well as all categories.

Comparisons with alternative methods. We compare gDS with two alternative methods based on ResNet50, *i.e.*, AdaBN (Li et al. 2016) and AdaIN (Huang and Belongie 2017) that perform alignment without retraining the weights of the whole network. AdaBN (Li et al. 2016) accumulates BN layer statistics on the target domain, without accessing training samples from IMAGENET-INSTAGRAM. However, since AdaBN is designed for a single target domain, we apply AdaBN on the validation set of each filter type separately. In this way, AdaBN targets each Instagram filter type at a time, thus obtaining better performance than applying AdaBN on entire IMAGENET-INSTAGRAM. On the other hand, AdaIN performs style transfer on filtered images such that the styles generated by filters are removed by an image generator. More specifically, we randomly select 100 images per Instagram filter from IMAGENET-INSTAGRAM training set and 200 images from IMAGENET to train a generator. Then the generator is used to sysnthesize filter-free images, upon which the original pretrained CNN is applied. The results are shown in Table. 5.

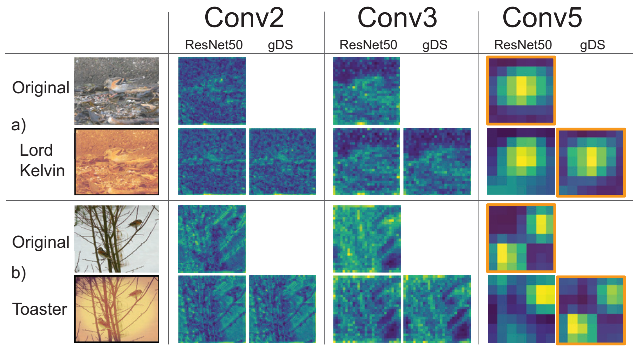

Figure 6: Feature maps comparison of ResNet50 and ResNet50+gDS. Two cases from two most challenging Instagram filers are shown. (a) "Lord Kelvin", (b) "Toaster". For each example case, we show the original IMAGENET image and its Instagram-filtered version. For each image, we also show the activation feature maps after `Conv2`, `Conv3` and `Conv5`, generated by ResNet50 and Resnet50+gDS. gDS succesfully transforms the features of filtered images to be close to original images (See the diagonal comparisons).

Method	Top1/Top5 acc
ResNet50	67.21/ 87.62
AdaBN (Li et al. 2016)	68.87/ 88,74
AdaIN (Huang and Belongie 2017)	35.45/ 58.86
gDS	**71.41/ 90.46**

Table 5: Comparison with other alternative methods on IMAGENET-INSTAGRAM-mini.

We can see AdaBN improves the performance of pretrained models by only 1% where as gDS obtained a 4.2% performance gain. The results of AdaIN are worse than directly applying ResNet50. A reason could be that the synthesis process with an image generator is far from perfect and further introduces artifacts and distribution shifts.

Conclusion

We presented a study on how popular filters that are prevalent on social media affect the performance of pretrained modern CNN models. We created IMAGENET-INSTAGRAM, by applying 20 pre-defined ImageNet filters to each image in IMAGENET. We found that filters induce significant differences in feature maps compared to those of original images, and further lead to significant drops when directly applying CNNs pretrained on IMAGENET. To improve generalization, we introduced a lightweight de-stylization module, which produces parameters used to scale and shift feature maps in order to recover changes brought by filters. Combining the lightweight module together with skip connections, we presented gDS that can be plugged into modern CNN networks. Extensive studies are conducted on IMAGENET-INSTAGRAM and the results confirm the effectiveness of the proposed method.

Acknowlegement

This research was funded by ARO Grant W911NF1610342.

References

Arjovsky, M., and Bottou, L. 2017. Towards principled methods for training generative adversarial networks. In *ICLR*.

Deng, J.; Dong, W.; Socher, R.; Li, L.-J.; Li, K.; and Fei-Fei, L. 2009. ImageNet: A large-scale hierarchical image database. In *CVPR*.

Dumoulin, V.; Shlens, J.; and Kudlur, M. 2016. A learned representation for artistic style. In *ICLR*.

Ganin, Y.; Ustinova, E.; Ajakan, H.; Germain, P.; Larochelle, H.; Laviolette, F.; Marchand, M.; and Lempitsky, V. 2016. Domain-adversarial training of neural networks. *JMLR*.

Gatys, L. A.; Ecker, A. S.; and Bethge, M. 2016. Image style transfer using convolutional neural networks. In *CVPR*.

Geirhos, R.; Temme, C. R.; Rauber, J.; Schütt, H. H.; Bethge, M.; and Wichmann, F. A. 2018. Generalisation in humans and deep neural networks. In *Advances in Neural Information Processing Systems*, 7549–7561.

Goodfellow, I.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.; and Bengio, Y. 2014. Generative adversarial nets. In *NIPS*.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2015. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, 1026–1034.

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*, 770–778.

Hendrycks, D., and Dietterich, T. 2019. Benchmarking neural network robustness to common corruptions and perturbations. In *ILCR*.

Hu, J.; Shen, L.; and Sun, G. 2018. Squeeze-and-excitation networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 7132–7141.

Huang, X., and Belongie, S. 2017. Arbitrary style transfer in real-time with adaptive instance normalization. In *ICCV*.

Huang, G.; Liu, Z.; Van Der Maaten, L.; and Weinberger, K. Q. 2017. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 4700–4708.

Huang, X.; Liu, M.-Y.; Belongie, S.; and Kautz, J. 2018. Multimodal unsupervised image-to-image translation. In *ECCV*.

Ioffe, S., and Szegedy, C. 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*.

Kornblith, S.; Shlens, J.; and Le, Q. V. 2018. Do better imagenet models transfer better? *arXiv preprint arXiv:1805.08974*.

Krizhevsky, A.; Sutskever, I.; and Hinton, G. E. 2012. Imagenet classification with deep convolutional neural networks. In *NIPS*.

Kurakin, A.; Goodfellow, I.; and Bengio, S. 2016. Adversarial examples in the physical world. *arXiv preprint arXiv:1607.02533*.

Li, Y.; Wang, N.; Shi, J.; Liu, J.; and Hou, X. 2016. Revisiting batch normalization for practical domain adaptation. *arXiv preprint arXiv:1603.04779*.

Pan, X.; Luo, P.; Shi, J.; and Tang, X. 2018a. Two at once: Enhancing learning and generalization capacities via ibn-net. In *ECCV*.

Pan, X.; Luo, P.; Shi, J.; and Tang, X. 2018b. Two at once: Enhancing learning and generalization capacities via ibn-net. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 464–479.

Saito, K.; Watanabe, K.; Ushiku, Y.; and Harada, T. 2018. Maximum classifier discrepancy for unsupervised domain adaptation. In *CVPR*.

Srivastava, N.; Hinton, G.; Krizhevsky, A.; Sutskever, I.; and Salakhutdinov, R. 2014. Dropout: A simple way to prevent neural networks from overfitting. *JMLR*.

Sun, B., and Saenko, K. 2016. Deep CORAL - correlation alignment for deep domain adaptation. In *ECCV*.

Tzeng, E.; Hoffman, J.; Saenko, K.; and Darrell, T. 2017. Adversarial discriminative domain adaptation. In *CVPR*.

Ulyanov, D.; Lebedev, V.; Vedaldi, A.; and Lempitsky, V. S. 2016. Texture networks: Feed-forward synthesis of textures and stylized images. In *ICML*.

Ulyanov, D.; Vedaldi, A.; and Lempitsky, V. 2016. Instance normalization: The missing ingredient for fast stylization. *arXiv preprint arXiv:1607.08022*.

Zhang, C.; Bengio, S.; Hardt, M.; Recht, B.; and Vinyals, O. 2017. Understanding deep learning requires rethinking generalization. In *ICLR*.

Zhu, J.-Y.; Park, T.; Isola, P.; and Efros, A. A. 2017. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *ICCV*.