Multi-Domain Learning for Accurate and Few-Shot Color Constancy

Jin Xiao[1,*]
[1]The Hong Kong Polytechnic University
csjxiao@comp.polyu.edu.hk

Shuhang Gu[2,*]
[2]CVL, ETH Zürich
shuhanggu@gmail.com

Lei Zhang[1,3,†]
[3]DAMO Academy, Alibaba Group
cslzhang@comp.polyu.edu.hk

Abstract

Color constancy is an important process in camera pipeline to remove the color bias of captured image caused by scene illumination. Recently, significant improvements in color constancy accuracy have been achieved by using deep neural networks (DNNs). However, existing DNN-based color constancy methods learn distinct mappings for different cameras, which require a costly data acquisition process for each camera device. In this paper, we start a pioneer work to introduce multi-domain learning to color constancy area. For different camera devices, we train a branch of networks which share the same feature extractor and illuminant estimator, and only employ a camera-specific channel re-weighting module to adapt to the camera-specific characteristics. Such a multi-domain learning strategy enables us to take benefit from cross-device training data. The proposed multi-domain learning color constancy method achieved state-of-the-art performance on three commonly used benchmark datasets. Furthermore, we also validate the proposed method in a few-shot color constancy setting. Given a new unseen device with limited number of training samples, our method is capable of delivering accurate color constancy by merely learning the camera-specific parameters from the few-shot dataset. Our project page is publicly available at https://github.com/msxiaojin/MDLCC.

1. Introduction

Human vision system naturally has the ability to compensate for different illuminants to a scene, named color constancy. The color of images captured by cameras, however are easily affected by different illuminants, and might appear "blueish" under sunlight and "yellowish" under indoor incandescent light. Aiming at estimating the scene illuminant from the captured image, color constancy is an important unit in camera pipeline to correct the color of captured images.

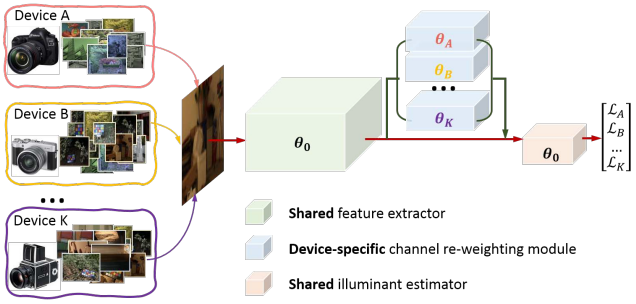


Figure 1. Overview of our proposed multi-domain learning color constancy method. We train color constancy networks for different devices simultaneously. Different networks share the same feature extractor and illuminant estimator with shared parameter θ_0, and only have their individual channel re-weighting module with parameters θ_A, θ_B and θ_K, respectively.

Classical color constancy methods utilize image statistics or physical properties to estimate illuminant of the scene. The performance of these approaches is highly dependent on the assumptions and these methods falter in cases where assumptions fail to hold [31]. In the last decade, another category of methods, i.e., the learning-based methods, have become more popular. Early learning-based methods [20, 15] adopt hand-crafted features and only learn the estimating function from the training data. Inspired by the success of deep neural networks (DNN) in other low-level vision tasks [25, 24, 16, 38], recently proposed DNN based approaches [9, 37, 26] learn image representation as well as the estimating function jointly, and have achieved state-of-the-art estimation accuracy.

DNN-based methods directly learn a mapping function between the input image and ground truth illuminant label. Given enough training data, they are able to use highly complex nonlinear function to capture the relationship between input images and the corresponding illuminants. However, the acquisition of data for training color constancy network is often costly: firstly, images, each contains the physical calibration objects, in a large variety of scenes under various illuminants must be collected; and then, ground-truth illuminant in each image needs to be estimated through the

*The first two authors contribute equally to this work.
†Corresponding author. This work is supported by China NSFC grant (no. 61672446) and Hong Kong RGC RIF grant (R5001-18).

corresponding calibration object. In addition, as raw data from different cameras exhibit distinct distributions, existing DNN-based color constancy approaches assume each camera has an independent network, and therefore require a large amount of labelled images for each camera. Due to the above reasons, the capacity of existing DNN-based color constancy methods are largely limited by the scale of training dataset. Great attempts have been made to improve the performance of color constancy models under insufficient training data.

In this paper, we proposed a multi-domain learning color constancy (MDLCC) method to leverage labelled color constancy data from different datasets and devices. Inspired by conventional imaging pipelines, which employ camera-specific estimation functions to estimate the illuminant from common low-level features, MDLCC adopts the same feature extractor to extract low-level features from input raw data, and use a camera-specific channel re-weighting module to transform device-specific features to a common feature space for adapting to different cameras. The common feature extractor is trained using data from different devices, and we train device-specific channel re-weighting module with data from different domains for domain adaptation. Such a strategy enables us to address the CSS difference among different cameras while leveraging different datasets to train a more powerful deep feature extractor. The proposed MDLCC framework learns most of the network parameters in each network with a much larger dataset, which significantly improves the color constancy accuracy of each camera.

Besides improving the color constancy performance of well established devices which already have a considerable amount of labelled data, our multi-domain network architecture also enables us to adapt our network to new cameras easily. Given insufficient number of labelled samples from a new camera device, MDLCC only needs to learn the device-specific parameters, and most of the network parameters are inherited from the meta-model which was trained on large scale dataset. Such a few-shot color constancy problem has been investigated in a recent paper [31]. McDonagh *et al*. [31] utilized the meta-learning technique [19] to learn a color constancy network which is easier to adapt to new cameras. However, as [31] still needs to fine-tune all the network parameters on the few-shot dataset, it has only achieved limited illuminant estimation performance in the few-shot setting. In contrast, the proposed MDLCC approach only needs to learn a small number of parameters from the few-show dataset, and is able to achieve higher few-shot estimation accuracy.

Our main contributions are summarized as follows:

1. This paper starts a pioneer work to leverage the multi-domain learning idea to improve the color constancy performance.
2. We propose a device-specific channel re-weighting module to adapt the features from different domains to a common estimator. This allows us to use the same feature extraction and illuminant estimation modules for different cameras.
3. The proposed MDLCC achieved state-of-the-art color constancy performance on benchmark datasets [36], [14] and [3], in both the standard and few-shot settings.

2. Related Work

In this section, we firstly provide an overview of color constancy and then introduce previous work of handling insufficient training data. Lastly, we present a brief introduction to the multi-domain methods, which is closely related to our contributions.

2.1. Color Constancy: An Overview

Existing color constancy methods can be divided into two categories: the statistics-based methods [12, 11, 18, 40] and the learning-based methods [15, 20, 8, 37, 26, 6, 7]. Based on different priors of the 'true' white-balanced image, statistics-based methods use statistics of the observed image to estimate the illuminant. Despite its fast estimating speed, the simple assumptions adopted in these approaches may not fit well the complex scenes, and thus limited the estimation performance of the statistics-based methods. The learning-based methods learn color constancy models from training data. Early works along this branch used handcraft features, followed by decision tree [15] or support vector regression approach [20] to regress the scene illuminants. To take full advantage of training data, recent works have started to learn features from data for color constancy. In [8], Bianco *et al*. used a 3 layer convolutional network to estimate local illuminants for image patches. Shi *et al*. [37] designed two sub-networks to adapt to the ambiguity of local estimates. In [26], Hu *et al*. proposed the FC^4 approach which introduced a confidence-weighted pooling layer in a fully convolutional network to estimate illuminants from images with arbitrary sizes. Besides extracting features from the raw image, [6, 7] constructed histograms in log-chromatic space, and then apply a learned conv filter to the histograms to estimate illuminant. In spite of the strong performances, learning-based color constancy methods often require a large amount of training data and have limited generalization capacity to new devices.

2.2. Color constancy with insufficient training data

Since the construction of large scale datasets with enough variety and manual annotations is often laborious and costly, a large number of approaches have been proposed to remedy the insufficiency of training data.

Data augmentation Data augmentation is a commonly used strategy for training models with insufficient data. Currently, most of the learning-based color constancy works have utilized the data augmentation strategy for improving the estimation accuracy. Specifically, random cropping [26] and image relighting [26, 9] are the most commonly used data augmentation schemes. However, as such simple augmentation schemes can not increase the diversity of scenes, they can only bring marginal improvement to the learned color constancy model. Recently, Banić *et al*. [2] designed a image generator to simulate images under various illuminants which however, is faced with the gap between synthetic and real data.

Pre-training Besides data augmentation, another strategy for improving color constancy performance is pre-training. FC^4 [26] started with the AlexNet, which is pre-trained on ImageNet dataset as feature extractor. A smaller learning rate is then used to fine-tune these parameters.

Weakly supervised learning Several works also resorted to unsupervised learning methods. In [39], Tieu *et al*. proposed to learn a linear statistical model on a single device from video frame observations. Banić *et al*. [3] utilize statistical approach to approximate the unknown ground-truth illumination of the training images, and learn color constancy model from approximated illumination values. Currently, the unsupervised learning approach has achieved better performance than conventional statistical-based methods, but is still not on par with supervised state-of-the-arts.

Inter-camera transformation Due to the distinction among raw images by different devices, large scale dataset needs to be collected for each device. Several work also focused on reducing the workload of constructing camera-specific dataset. Gao *et al*. [21] attempt to discount the variation among different devices by learning a transformation matrix based on camera spectral sensitivity. Banić *et al*. [3] proposed to learn transformation matrix among ground truth distributions of two cameras, before inter-camera experiments. The existing inter-camera approaches only study pairs of sensors and there has not been any works which could leverage data from a large number of devices.

Few-shot learning Recently, McDonagh *et al*. [31] have formulated the color constancy of different cameras and color temperature as a few-shot learning problem. The model-agnostic meta-learning method [19] has been adopted to learn a meta model which is capable of adapting to new cameras using only a small number of training samples. However, as McDonagh *et al*. did not exploit domain knowledge of color constancy and only rely on the adaptation capacity of MAML algorithm [31], only achieved limited performance in the few-shot setting.

2.3. Multi-domain Learning

Multi-domain learning aims to improve the performance for the same tasks with inputs from multiple domains, by exploiting correlation among the multi-domain datasets. In the last decade, a large amount of works [28, 33, 34, 35] have comprehensively shown that by jointly learning from multiple domains brings significant performance gains compared with individually learning for each domain. These methods usually incorporate an adaptation model, e.g., the domain-specific conv [34, 35] and batch normalization [10], to adapt to inputs from different domains. In this paper, we start from the commonality of different devices' color constancy problems, and design a camera-specific channel re-weighting layer for handling multi-device color constancy problem.

3. Multi-domain Learning Color Constancy

In this section, we introduce our proposed multi-domain learning color constancy (MDLCC) method. We start with the formulation of color constancy problem and the target of our MDLCC model. Then, we introduce the network architecture of MDLCC as well as how MDLCC could be utilized to solve the few-shot color constancy problem.

3.1. Problem Formulation

We focus on the single illuminant color constancy problem which assumes the scene illuminant is global and uniform. Under the Lambertian assumption, the image formation can be simplified as:

$$\mathbf{Y}_c = \Sigma_{n=1}^{N} \mathbf{C}_c(\lambda_n)\mathbf{I}(\lambda_n)\mathbf{R}(\lambda_n),\ c \in \{r, g, b\} \quad (1)$$

where $\mathbf{Y}$ is the observed raw image. λ_n for $n = 1, 2, ...N$ represents the discrete sample of wavelength λ. $\mathbf{C}_c(\lambda_n)$ represents the camera spectral sensitivity (CSS) of color channel c. $\mathbf{I}(\lambda_n)$ is the spectral power distribution of illuminant, and $\mathbf{R}(\lambda_n)$ denotes the surface reflectance of the scene. Color constancy aims to estimate the illuminant $\mathbf{L} = [L_r, L_g, L_b]$ given the observed image $\mathbf{Y}$. The latent 'white-balanced' image $\mathbf{W}$ can then be derived according to the von Kries model [41] by

$$\mathbf{W}_c = \mathbf{Y}_c / L_c,\ c \in \{r, g, b\}. \quad (2)$$

Since different cameras use distinct CSS, raw image $\mathbf{Y}$ by different camera occupies different color subspaces. Existing learning based methods generally train independent model for each device. In this work we combine raw images by different devices to jointly learn a color constancy model. Denote the training data from device k as $D_k = \{\mathbf{Y}_{k,i}, \mathbf{L}_{k,i}\}_{i=1}^{N_k}$, where the superscript k, i denote the device index and sample index, respectively, and N_k is the number of samples for D_k. The proposed multi-domain

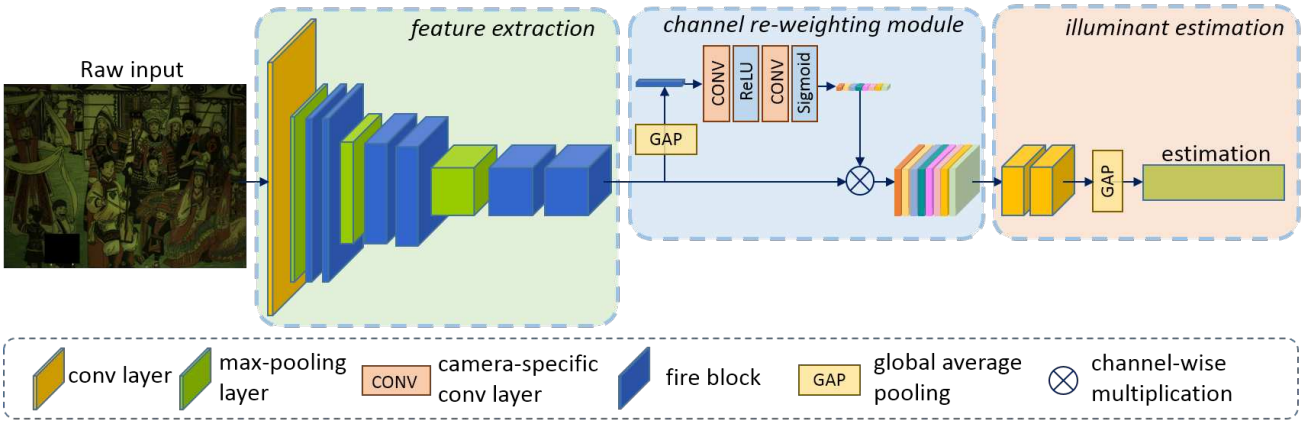


Figure 2. The proposed multi-domain color constancy network architecture. We used shared layers among multiple devices for feature extraction. A camera-specific channel re-weighting module was then used to adapt to each device. The illuminant estimation stage finally predicted the scene illuminant.

learning color constancy aims to learn a branch of networks which take raw images from different domains as inputs to estimate the illuminant of the scene:

$$\{\theta_0^*, \theta_k^*\} = \arg\min_{\theta_0,\theta_k} \sum_{k=1}^{K}\sum_{i=1}^{N_k} \mathcal{L}\left(\mathbf{L}_{k,i}, f(\mathbf{Y}_{k,i}; \theta_0, \theta_k)\right), \tag{3}$$

where the same network architecture $f(\cdot)$ is adopted for all the devices, and θ_0 and θ_k are the shared and device-specific parameters in the networks, respectively. $\mathcal{L}$ is the loss function which measures the difference between ground truth and estimated illuminants.

3.2. Network Architecture of MDLCC

As introduced in the previous section, we proposed to utilize the same network architecture and only use partial device-specific parameters to adapt to different devices. In order to validate our idea of using multi-domain learning to improve color constancy performance for different devices, we do not investigate new network architecture and utilize FC^4 (SqueezeNet model) as our backbone. Specifically, we assume FC^4 can be divided into two stages: 1) the first 10 layers of network, which gradually reduce the spatial resolution of feature maps, constitute a low-level feature extractor; 2) the last 2 layers of network constitute an estimator which summarizes the extracted feature to estimate the illuminant. Inspired by previous inter-camera approaches [21] which proposed to learn a transformation matrix to correlate different cameras, we propose a device-specific channel re-weighting module and apply different transforms, in the high dimensional feature space, for features extracted from different devices.

An illustration of our network architecture is presented in Fig. 2. For different devices, we employ the same feature extraction module to extract features from input images; and then use the device-specific channel re-weighting module to transform the features; finally, the same estimator is utilized to generate the final illuminant estimation. The details of the feature extraction, channel re-weighting and illuminant estimation modules are introduced as follows.

Feature extraction. We use the first 10 layers in FC^4 as our feature extractor. For the first layer, stride 2 convolution with 64 filters of size 3×3 is used to generate 64 feature maps. Then, 3 blocks, each consists of a max pooling layer and two fire blocks [27] are followed to increase receptive field and further reduce the spatial resolution of feature map by factor 8. The channel dimension of feature maps after each block is 128, 256 and 384 respectively. The ReLU [32] is used as activation function following each conv layer.

Channel re-weighting module. In order to adapt the low-level features from different domains to a common space, we propose a device-specific channel re-weighting module to transform features. Concretely, we derive the scaling factors from statistic of extracted features and device-specific parameters. Denote the output of feature extractor for image $\mathbf{Y}_{k,i}$ as $\mathbf{F}_{k,i}$, we use a global average pooling layer to calculate the mean values for each channel of $\mathbf{F}_{k,i}$. Then, the channel-wise scaling vector $\boldsymbol{\omega}_{k,i}$ can be obtained by:

$$\boldsymbol{\omega}_{k,i} = g_{sigmoid}(\mathbf{W}_{k,b} * g_{ReLU}(\mathbf{W}_{k,a} * \mathbf{z}_{k,i})), \tag{4}$$

where $\mathbf{z}_{k,i}$ is the mean values of $\mathbf{F}_{k,i}$, $\{\mathbf{W}_{k,a}, \mathbf{W}_{k,b}\}$ are device-specific parameters, $*$ is the convolution operator, g_{ReLU} and $g_{sigmoid}$ are the ReLU and sigmoid functions, respectively. Eq. (4) utilizes two device-specific fully connected layers to generate the channel scaling factors from the statistics of input feature map. Having $\boldsymbol{\omega}_{k,i}$, the transformed feature $\mathbf{G}_{k,i}$ can be obtained by:

$$\mathbf{G}_{k,i} = \boldsymbol{\omega}_{k,i} \otimes \mathbf{F}_{k,i}, \tag{5}$$

where $\otimes$ represents the channel-wise multiplication.

Illuminant estimation. With the transformed feature $\mathbf{G}_{k,i}$, we utilize two convolution layers to estimate local illuminants and the final global illuminant value $\hat{\mathbf{L}}_{k,i}$ is achieved by a subsequent global average pooling layer.

During the training phase, all the training samples contribute to the training of feature extraction and illuminant estimation modules, while only the samples from device k affect the device-specific parameters $\{\mathbf{W}_{k,a}, \mathbf{W}_{k,b}\}$ in the channel re-weighting module.

3.3. MDLCC for few-shot color constancy

MDLCC learns shared and device-specific parameters to leverage the labelled data from different devices. Most of the parameters are shared by different devices and only a small portion (6.7%) of parameters are device-specific. Such a property of MDLCC makes it an ideal architecture for few-shot color constancy. Specifically, given limited number of training samples from a new unseen device, we only need to learn the device-specific parameters from these samples and the shared parameters can be inherited from existing MDLCC models. More details of our few-shot color constancy settings will be introduced in section 4.2.

4. Experiments

4.1. Datasets

We evaluate our proposed method using three widely-used color constancy datasets: the reprocessed [36] Gehler-Shi dataset [22], the NUS 8-camera dataset [14] and the Cube+ dataset [3]. The Gehler-Shi dataset was collected using two cameras, i.e., Canon 1D and Canon 5D. It contains both indoor and outdoor scenes, and comprises 568 scenes in total. The NUS dataset contains 1,736 images which were collected using 8 cameras in about 260 scenes. While the Cube+ dataset is a recently released large scale color constancy dataset. It contains 1,365 outdoor scenes and 342 indoor scenes. And all the images were captured by a Canon 550D camera. For each dataset, we follow previous work [6, 7, 26] to use the linear RGB images for experiments. The linear RGB images were obtained by applying a simple down-sample de-mosaicking operation to the raw images, followed by black-level subtraction and saturation pixel removal.

We follow previous works [7, 26, 14] to use 3-fold cross validation for each dataset. Specifically, for the Gehler-Shi dataset, we used the cross validation splits provided in the author's homepage. The subsets for each camera in NUS dataset contain images from the same scene. To ensure that the same scene would not be in both training and testing sets when combining multiple subsets in the NUS dataset, we split the training and testing set for NUS dataset according to scene content. As for the cube+, we randomly split the testing set into 3 folds for cross validation. We use the angular error in degree as quantitative measure, which has been utilized in previous methods [6, 7, 26, 14]. In all of our experiments, we report 5 metrics of the angular errors, i.e., the mean, median, tri-mean of all errors, mean of the lowest 25% of errors, and mean of the highest 25% of errors.

4.2. Implementing Details

We train our networks with the angular loss:

$$\mathcal{L}(\mathbf{L}, \hat{\mathbf{L}}) = cos^{-1}\left(\frac{\hat{\mathbf{L}} \odot \mathbf{L}}{||\hat{\mathbf{L}}|| \times ||\mathbf{L}||}\right), \tag{6}$$

where $\odot$ represents the inner product, and $cos^{-1}(\cdot)$ is the inverse of cosine function.

Our framework is implemented based on TensorFlow [1] with CUDA support. For both the multi-domain setting and few-shot setting, we train our networks with inputs of size $384 \times 384 \times 3$. Image random cropping and relighting [26] are used as data augmentations. We employ the Adam solver [30] as optimizer and set the learning rate as 1×10^{-4}. The weight decay value is set as 0.0001 and momentum is set as 0.9. For the experiments with all the training samples, we train our model for 750,000 iterations with batch size 8. While for few-shot experiments, we train our model for 15,000 iterations with batch size 8.

For the multi-domain setting, we train all the parameters from scratch and initialize them with normal distribution. For the few-shot setting, the shareable weights are directly inherited from the meta-model (more details of meta model will be introduced in section 4.5) and we only train camera-specific parameters. The camera-specific parameters are initialized with normal distribution.

4.3. Ablation Study and Analysis

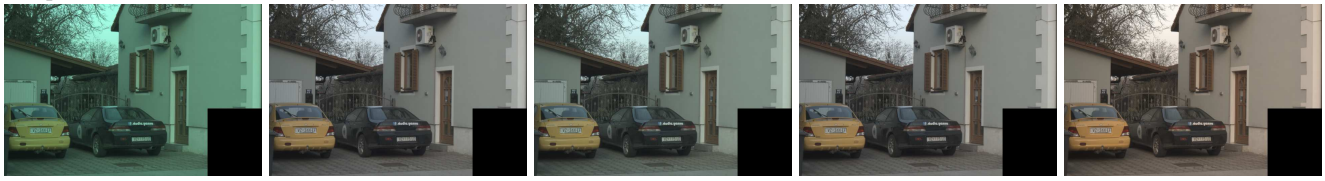
In this section, we carry out ablation study to evaluate the effectiveness of multi-domain learning as well as our proposed camera-specific channel re-weighting module.

To validate the effectiveness of multi-domain color constancy, we implement two variants: 1) single device color constancy and 2) multi-device combination model. Concretely, the single device color constancy model utilizes our network architecture and trains independently network for each device; the multi-device combination method collects training data from all devices and trains a unique network to process images from different devices. For fair comparison, all the hyper parameters are kept the same as in our MDLCC approach. Furthermore, in order to analyze the effect of device number for our multi-domain learning model, we present 4 groups of experiments which utilize images from different numbers of cameras for training. The details of the combined cameras are listed in Table 1. In the last group, we combine all the cameras from Gehler-Shi, NUS and Cube+ dataset, which contain 11 different cameras in total. The quantitative performances are listed in Table 1.

Table 1. Ablation study by comparing Single Device model, Multi-device Combination model and our proposed MDLCC model, under different combinations of cameras. The best is shown in red.

Method / Dataset	Single Device Color Constancy					Multi-device Combination					MDLCC				
	Mean	Med.	Tri.	Best 25%	Worst 25%	Mean	Med.	Tri.	Best 25%	Worst 25%	Mean	Med.	Tri.	Best 25%	Worst 25%
Gehler-Shi	1.66	1.14	1.24	0.38	3.86	1.91	1.34	1.41	0.42	4.47	1.62	1.10	1.17	0.36	3.79
NUS-C600D	1.97	1.39	1.54	0.47	4.37	1.92	1.34	1.47	0.44	4.26	1.82	1.26	1.39	0.44	4.15
Gehler-Shi	1.66	1.14	1.24	0.38	3.86	1.89	1.35	1.46	0.41	4.45	1.61	0.99	1.11	0.37	3.79
NUS-C1	2.04	1.45	1.60	0.50	4.55	1.98	1.42	1.54	0.48	4.35	1.87	1.33	1.48	0.46	4.19
Cube+	1.35	0.95	1.02	0.32	3.04	1.35	0.93	1.00	0.31	3.10	1.24	0.83	0.96	0.26	2.97
NUS-Fuj.	2.08	1.59	1.73	0.50	4.45	2.04	1.54	1.66	0.49	4.32	1.97	1.39	1.51	0.45	4.43
NUS-N52	2.33	1.65	1.82	0.50	5.34	2.21	1.53	1.73	0.45	4.89	2.00	1.47	1.53	0.45	4.59
Cube+	1.35	0.95	1.02	0.32	3.04	1.35	0.92	1.01	0.31	3.08	1.26	0.84	0.94	0.25	2.97
Gehler-Shi	1.66	1.14	1.24	0.38	3.86	1.87	1.33	1.46	0.43	4.40	1.59	0.95	1.11	0.37	3.77
NUS-C1	2.04	1.45	1.60	0.50	4.55	2.00	1.43	1.55	0.45	4.39	1.86	1.35	1.49	0.46	4.11
NUS-C600D	1.97	1.39	1.54	0.47	4.37	1.93	1.35	1.45	0.44	4.33	1.65	1.16	1.29	0.35	3.73
NUS-Fuj.	2.08	1.59	1.73	0.50	4.45	2.03	1.55	1.67	0.47	4.36	1.87	1.37	1.48	0.45	4.18
NUS-N52	2.33	1.65	1.82	0.50	5.34	2.25	1.66	1.79	0.44	5.01	1.96	1.38	1.52	0.44	4.54
NUS-Oly.	1.86	1.37	1.51	0.47	4.08	1.80	1.34	1.48	0.46	3.97	1.68	1.15	1.30	0.34	3.85
NUS-Pan.	1.98	1.41	1.48	0.41	4.52	1.90	1.38	1.46	0.42	4.37	1.69	1.20	1.33	0.45	3.73
NUS-Sam.	2.18	1.66	1.75	0.54	4.79	2.13	1.52	1.69	0.52	4.62	1.78	1.33	1.42	0.41	3.95
NUS-Son.	1.91	1.51	1.56	0.55	4.05	1.86	1.47	1.54	0.53	3.89	1.74	1.36	1.44	0.46	3.70
Cube+	1.35	0.95	1.02	0.32	3.04	1.36	0.92	1.05	0.33	3.15	1.24	0.84	0.95	0.27	2.95

Input from Gehler-Shi | Single device (6.04°) | MD-combination (8.80°) | MDLCC (3.39°) | GT



Input from Cube | Single device (6.19°) | MD-combination (10.62°) | MDLCC (5.25°) | GT

Figure 3. Visualization of color constancy results by single device color constancy model, multiple device combination model, and our proposed MDLCC model. Images are converted to sRGB for visualization.

Multi-domain learning Compared with single device approach which learns distinct network on each dataset, our method achieves better performance on all the sub-datasets. Even for the large scale Cube+ dataset which contains 1707 training samples, data from related domains is beneficial. This clearly demonstrates the effectiveness of multi-domain learning in the color constancy area.

Camera-specific channel re-weighting module By comparing single device results and multi-device combination results, we found that directly combining several datasets without the camera-specific module can not constantly improve the color constancy performance. It might lead to improved performance for one camera but degrades severely for the others. For example, when combining Gehler-Shi with NUS-C600D, the performance on Gehler-Shi dataset degrades dramatically from 1.66 to 1.91 in mean error. This reveals that directly combining multiple dataset without device-specific module can not take full advantage of the cross-device training data. While, by adopting the camera-specific channel re-weighting module, our MDLCC approach significantly outperforms the multi-device combination baseline.

Number of devices From Table 1 we also observed that by increasing the number of devices in MDLCC, the performance can be further improved. This is because more training samples comprise more scenes and illuminants, and are beneficial for learning more generalized representations. For example, the mean error of MDLCC on NUS-600D is

Table 2. Color constancy results by different methods on reprocessed Gehler-Shi [36], NUS [14] and Cube+ dataset [3]. The best and second metric is shown in red and blue respectively.

Method \ Dataset	Gehler-Shi					NUS					Cube+				
	Mean	Med.	Tri.	Best 25%	Worst 25%	Mean	Med.	Tri.	Best 25%	Worst 25%	Mean	Med.	Tri.	Best 25%	Worst 25%
White-Patch [11]	7.55	5.68	6.35	1.45	16.12	9.91	7.44	8.78	1.44	21.27	6.80	3.85	5.21	0.68	16.93
Grey-world [12]	6.36	6.28	6.28	2.33	10.58	4.59	3.46	3.81	1.16	9.85	3.52	2.55	2.82	0.60	7.98
Edge-based Gamut [4]	6.52	5.04	5.43	1.90	13.58	4.40	3.30	3.45	0.99	9.83	–	–	–	–	–
1st-order Gray-Edge [40]	5.33	4.52	4.73	1.86	10.03	3.35	2.58	2.76	0.79	7.18	3.06	2.05	2.32	0.55	7.22
2nd-order Gray-Edge [40]	5.13	4.44	4.62	2.11	9.26	3.36	2.70	2.80	0.89	7.14	3.28	2.34	2.58	0.66	7.44
Shades-of-Gray [18]	4.93	4.01	4.23	1.14	10.20	3.67	2.94	3.03	0.98	7.75	3.22	2.12	2.44	0.43	7.77
Bayesian [22]	4.82	3.46	3.88	1.26	10.49	3.50	2.36	2.57	0.78	8.02	–	–	–	–	–
General Gray-World [5]	4.66	3.48	3.81	1.00	10.09	3.20	2.56	2.68	0.85	6.68	–	–	–	–	–
Natural Image Statistics [23]	4.19	3.13	3.45	1.00	9.22	3.45	2.88	2.95	0.83	7.18	–	–	–	–	–
Spatio-spectral Statistics [13]	3.59	2.96	3.10	0.95	7.61	3.06	2.58	2.74	0.87	6.17	–	–	–	–	–
Intersection-based Gamut [4]	4.20	2.39	2.93	0.51	10.70	–	–	–	–	–	–	–	–	–	–
Pixels-based Gamut [4]	4.20	2.33	2.91	0.50	10.72	5.27	4.26	4.45	1.28	11.16	–	–	–	–	–
Cheng 2014 [14]	3.52	2.14	2.47	0.50	8.74	2.18	1.48	1.64	0.46	5.03	–	–	–	–	–
Exemplar-based [29]	2.89	2.27	2.42	0.82	5.97	–	–	–	–	–	–	–	–	–	–
Corrected-Moment [17]	2.86	2.04	2.22	0.70	6.34	2.95	2.05	2.16	0.59	6.89	–	–	–	–	–
Regression Tree [15]	2.42	1.65	1.75	0.38	5.87	–	–	–	–	–	–	–	–	–	–
CCC [6]	1.95	1.22	1.38	0.35	4.76	2.38	1.48	1.69	0.45	5.85	–	–	–	–	–
DS-Net (HypNet+SelNet) [37]	1.90	1.12	1.33	0.31	4.84	2.24	1.46	1.68	0.48	6.08	–	–	–	–	–
FC^4 (SqueezeNet-FC^4) [26]	1.65	1.18	1.27	0.38	3.78	2.23	1.57	1.72	0.47	5.15	1.35	0.93	1.01	0.30	3.24
FFCC (model J) [7]	1.80	0.95	1.18	0.27	4.65	1.99	1.31	1.43	0.35	4.75	1.38	0.74	0.89	0.19	3.67
FFCC+metadata+semantics [7]	1.61	0.86	1.02	0.23	4.27	–	–	–	–	–	–	–	–	–	–
MDLCC	1.58	0.95	1.11	0.37	3.77	1.78	1.29	1.40	0.42	3.97	1.24	0.83	0.92	0.26	2.91

1.82 when combining with Gehler-Shi, which can be further decreased to 1.65 when combining with all the other cameras. This also demonstrates the effectiveness of our proposed camera-specific channel re-weighting module. Our model is still effective in handling 11 devices.

4.4. Comparison with State-of-the-art

In this section, we compare our proposed multi-domain color constancy approach with other color constancy algorithms. We compare our approach with competing methods on the Gehler-Shi [36], NUS [14] and Cube+ [3] datasets. For the NUS dataset, we follow previous work [7, 26] and take the geometric mean of each metric over 8 cameras. We train our model by combining all the devices in the three datasets. The results of comparison methods on the Gehler-Shi dataset and NUS dataset are collected from [7, 26]. While, for the Cube+ dataset, we present the results using open source codes from the authors' webpages. We retrain the FFCC [7] and FC^4 [26] models on the Cube+ dataset, and the hyper-parameters have been carefully tuned to achieve the best performance.

The experimental results are listed in Table 2. Except the state-of-the-art FFCC approach, the proposed MDLCC outperforms all competing approaches in all metrics. Specifically, our model constantly outperforms our backbone architecture, i.e., the FC^4 approach, this clearly validates the effectiveness of multi-domain learning for color constancy. Compared to the FFCC approach, our model generally outperforms the base FFCC model which only exploits image content for color constancy, and is comparable to the full FFCC model which additionally takes the camera metadata (exposure setting and camera info) and semantic information as inputs. Concretely, our model shows better performance in terms of mean error and the worst 25% of mean errors, while inferior performances in the other three metrics. A possible reason is that our loss function has the tendency to reduce the average error over all training samples, which better fits the mean error and worst 25% metrics.

4.5. Few-shot Evaluations

In this section, we conduct experiments to validate the capacity of the proposed model for few-shot color constancy problem. We used the Gehler-Shi, Cube dataset and one subset from NUS (NUS-C1) as the few-shot testing datasets. Note that Cube dataset is a subset from Cube+ which contains only the outdoor scenes. We choose Cube instead of Cube+ for the purpose of directly comparing our method with the recently proposed Few-shot Meta-Learning Color Constancy method (FMLCC) [31]. For training our model, we use the remaining 7 datasets, i.e., 7 subsets from NUS dataset, as the training set and only finetune those device-specific parameters on the few-shot dataset. Specifically, we vary the number of few-shot sample K as 1, 5, 10 and 20 respectively, for thoroughly validat-

Table 3. Comparison of few-shot color constancy models.

Method \ Test set		NUS-C1					Cube					Gehler-Shi				
		Mean	Med.	Tri.	Best 25%	Worst 25%	Mean	Med.	Tri.	Best 25%	Worst 25%	Mean	Med.	Tri.	Best 25%	Worst 25%
Single device		2.04	1.45	1.60	0.50	4.55	1.21	0.85	0.90	0.23	2.85	1.66	1.14	1.24	0.38	3.86
FMLCC [31]	K=10	–	–	–	–	–	1.63	1.08	1.20	0.31	3.89	2.66	1.91	1.99	0.49	6.20
	K=20	–	–	–	–	–	1.59	1.02	1.15	0.30	3.85	2.57	1.84	1.94	0.47	6.11
MDLCC	K=1	2.93	2.27	2.40	0.95	6.05	2.02	1.75	1.83	0.85	3.67	3.00	2.32	2.49	0.88	6.24
	K=5	2.36	1.72	1.87	0.60	5.08	1.63	1.20	1.30	0.50	3.46	2.43	1.76	1.94	0.59	5.33
	K=10	2.27	1.61	1.81	0.57	4.97	1.56	1.14	1.24	0.43	3.33	2.32	1.68	1.83	0.57	5.17
	K=20	2.18	1.59	1.75	0.51	4.80	1.47	1.06	1.14	0.39	3.27	2.26	1.60	1.75	0.56	5.08

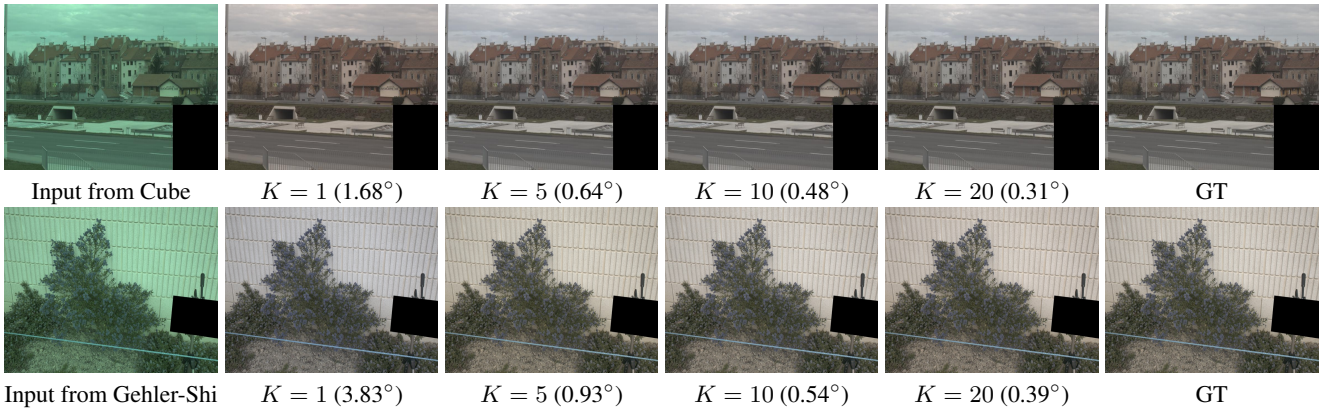


Figure 4. Visualization of few-shot color constancy results. Images are converted to sRGB for visualization. The two input images are taken from Cube and Gehler-Shi dataset respectively. We present the few-shot color constancy results with different training sample K. The angular error in degree is also given.

ing our method. We split each test dataset into three folds. For each fold, we randomly chose K samples from the remaining folds to construct the training samples, which were used to learn the camera-specific parameters. To avoid the randomness and disturbance by the selection of K training samples, we repeated the few-shot experiments for 10 times, each with different random choices of K images. We then present the average of each metric over 10 runs. The few-shot performances are listed in Table 3. We choose FMLCC [31] for comparison and the results of FMLCC are copied from the original paper [31]. The performance of the single device color constancy, which used the whole dataset for training, is also provided for reference.

Compared with previous few-shot color constancy approach FMLCC [31], our model achieved much better results in most of metrics. In addition, as FMLCC needs to fine-tune all the network weights, they might not be able to deliver good results for extreme few-shot cases, for example $K = 1$. While, as our model only requires retraining the camera-specific weights, we can still obtain good color constancy performance. From Table 3 and Table 2, one can see that with only single shot ($K = 1$), our model outperforms most of statistical-based approaches. Moreover, when using $K = 20$ training samples, our model achieves comparable performance with single device model, which used the whole dataset for training. Some visual examples of our few shot color constancy results are provided in Fig. 4.

5. Conclusions

Deep networks can largely improve the color constancy accuracy with large scale annotated dataset. However, the acquisition of such dataset is laborious and costly, especially for color constancy problem which requires independent dataset for each camera due to the distinction in devices. In this paper, we start a pioneer work to leverage the multi-domain learning method for color constancy problem. Specifically, we utilized training data by different devices to train a single model, to learn complementary representations and improve generalization capability. Experimental results show that with the proposed shareable modules and camera-specific module, our model achieves much better results than training independent model for each device, and also achieves state-of-the-art performance on three benchmark datasets. We also tested the color constancy performances under few-shot setting. Experimental results show that the proposed model can effectively adapt to a new device with only a few, e.g., 20, training samples.

References

[1] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. Tensorflow: A system for large-scale machine learning. In *12th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 16)*, pages 265–283, 2016.

[2] Nikola Banić, Karlo Koščević, Marko Subašić, and Sven Lončarić. Crop: Color constancy benchmark dataset generator. *arXiv preprint arXiv:1903.12581*, 2019.

[3] Nikola Banić and Sven Lončarić. Unsupervised learning for color constancy. *arXiv preprint arXiv:1712.00436*, 2017.

[4] Kobus Barnard. Improvements to gamut mapping colour constancy algorithms. In *European conference on computer vision*, pages 390–403. Springer, 2000.

[5] Kobus Barnard, Vlad Cardei, and Brian Funt. A comparison of computational color constancy algorithms. i: Methodology and experiments with synthesized data. *IEEE transactions on Image Processing*, 11(9):972–984, 2002.

[6] Jonathan T Barron. Convolutional color constancy. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 379–387, 2015.

[7] Jonathan T Barron and Yun-Ta Tsai. Fast fourier color constancy. In *IEEE Conf. Comput. Vis. Pattern Recognit*, 2017.

[8] Simone Bianco, Claudio Cusano, and Raimondo Schettini. Color constancy using cnns. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 81–89, 2015.

[9] Simone Bianco, Claudio Cusano, and Raimondo Schettini. Single and multiple illuminant estimation using convolutional neural networks. *IEEE Transactions on Image Processing*, 26(9):4347–4362, 2017.

[10] Hakan Bilen and Andrea Vedaldi. Universal representations: The missing link between faces, text, planktons, and cat breeds. *arXiv preprint arXiv:1701.07275*, 2017.

[11] David H Brainard and Brian A Wandell. Analysis of the retinex theory of color vision. *JOSA A*, 3(10):1651–1661, 1986.

[12] Gershon Buchsbaum. A spatial processor model for object colour perception. *Journal of the Franklin institute*, 310(1):1–26, 1980.

[13] Ayan Chakrabarti, Keigo Hirakawa, and Todd Zickler. Color constancy with spatio-spectral statistics. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(8):1509–1519, 2012.

[14] Dongliang Cheng, Dilip K Prasad, and Michael S Brown. Illuminant estimation for color constancy: why spatial-domain methods work and the role of the color distribution. *JOSA A*, 31(5):1049–1058, 2014.

[15] Dongliang Cheng, Brian Price, Scott Cohen, and Michael S Brown. Effective learning-based illuminant estimation using simple features. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1000–1008, 2015.

[16] Chao Dong, Chen Change Loy, Kaiming He, and Xiaoou Tang. Image super-resolution using deep convolutional networks. *IEEE transactions on pattern analysis and machine intelligence*, 38(2):295–307, 2015.

[17] Graham D Finlayson. Corrected-moment illuminant estimation. In *Computer Vision (ICCV), 2013 IEEE International Conference on*, pages 1904–1911. IEEE, 2013.

[18] Graham D Finlayson and Elisabetta Trezzi. Shades of gray and colour constancy. In *Color and Imaging Conference*, volume 2004, pages 37–41. Society for Imaging Science and Technology, 2004.

[19] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1126–1135. JMLR. org, 2017.

[20] Brian Funt and Weihua Xiong. Estimating illumination chromaticity via support vector regression. In *Color and Imaging Conference*, volume 2004, pages 47–52. Society for Imaging Science and Technology, 2004.

[21] Shao-Bing Gao, Ming Zhang, Chao-Yi Li, and Yong-Jie Li. Improving color constancy by discounting the variation of camera spectral sensitivity. *JOSA A*, 34(8):1448–1462, 2017.

[22] Peter Vincent Gehler, Carsten Rother, Andrew Blake, Tom Minka, and Toby Sharp. Bayesian color constancy revisited. In *2008 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE, 2008.

[23] Arjan Gijsenij and Theo Gevers. Color constancy using natural image statistics and scene semantics. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(4):687–698, 2011.

[24] Shuhang Gu, Shi Guo, Wangmeng Zuo, Yunjin Chen, Radu Timofte, Luc Van Gool, and Lei Zhang. Learned dynamic guidance for depth image reconstruction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2019.

[25] Shuhang Gu, Yawei Li, Luc Van Gool, and Radu Timofte. Self-guided network for fast image denoising. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2511–2520, 2019.

[26] Yuanming Hu, Baoyuan Wang, and Stephen Lin. Fc 4: Fully convolutional color constancy with confidence-weighted pooling. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4085–4094, 2017.

[27] Forrest N Iandola, Song Han, Matthew W Moskewicz, Khalid Ashraf, William J Dally, and Kurt Keutzer. Squeezenet: Alexnet-level accuracy with 50x fewer parameters and¡ 0.5 mb model size. *arXiv preprint arXiv:1602.07360*, 2016.

[28] Mahesh Joshi, William W Cohen, Mark Dredze, and Carolyn P Rosé. Multi-domain learning: when do domains matter? In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 1302–1312. Association for Computational Linguistics, 2012.

[29] Hamid Reza Vaezi Joze and Mark S Drew. Exemplar-based color constancy and multiple illumination. *IEEE transactions on pattern analysis and machine intelligence*, 36(5):860–873, 2014.

[30] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[31] Steven McDonagh, Sarah Parisot, Zhenguo Li, and Gregory Slabaugh. Meta-learning for few-shot camera-adaptive color constancy. *arXiv preprint arXiv:1811.11788*, 2018.

[32] Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th international conference on machine learning (ICML-10)*, pages 807–814, 2010.

[33] Sylvestre-Alvise Rebuffi, Hakan Bilen, and Andrea Vedaldi. Learning multiple visual domains with residual adapters. In *Advances in Neural Information Processing Systems*, pages 506–516, 2017.

[34] Sylvestre-Alvise Rebuffi, Hakan Bilen, and Andrea Vedaldi. Learning multiple visual domains with residual adapters. In *Advances in Neural Information Processing Systems*, pages 506–516, 2017.

[35] Amir Rosenfeld and John K Tsotsos. Incremental learning through deep adaptation. *IEEE transactions on pattern analysis and machine intelligence*, 2018.

[36] Lilong Shi. Re-processed version of the gehler color constancy dataset of 568 images. *http://www. cs. sfu. ca/~ color/data/*, 2000.

[37] Wu Shi, Chen Change Loy, and Xiaoou Tang. Deep specialized network for illuminant estimation. In *European Conference on Computer Vision*, pages 371–387. Springer, 2016.

[38] Xin Tao, Hongyun Gao, Xiaoyong Shen, Jue Wang, and Jiaya Jia. Scale-recurrent network for deep image deblurring. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8174–8182, 2018.

[39] Kinh Tieu and Erik G Miller. Unsupervised color constancy. In *Advances in neural information processing systems*, pages 1327–1334, 2003.

[40] Joost Van De Weijer, Theo Gevers, and Arjan Gijsenij. Edge-based color constancy. *IEEE Transactions on image processing*, 16(9):2207–2214, 2007.

[41] J von Kries. Chromatic adaptation, festschrift der alberchtludwig-universität, 1902.