EXIF Estimation With Convolutional Neural Networks

Divyahans Gupta
Stanford University
dgupta2@stanford.edu

Sanjay Kannan
Stanford University
skalon@stanford.edu

Abstract

In this paper, we explore a novel application of convolutional neural networks (CNNs): the task of estimating photographic metadata. Given the raw pixels of a digital photograph, we attempt to predict the camera settings it was taken with, including shutter speed, aperture size, and ISO level. We present some baseline results on this task using the MIRFLICKR dataset. We further contextualize these results in the realm of photography.

1. Introduction

Modern digital cameras store metadata, known as EXIF data, with every photo they capture. EXIF data stores information about the way a photo was taken, such as a camera's shutter speed, exposure level, and aperture size, as well as the particular setting (time and place).

EXIF fields store significant properties of an image (depth of field and brightness, to name two examples) that are recognizable to the human eye. Figures 1, 2, and 3 illustrate the visual effects relevant to the EXIF fields used in this study. Given only the pixels of a digital photograph as input, our task is to accurately estimate values for these fields.

Figure 1: Visual manifestations of shutter speed variance. The photograph on the left is taken with a high shutter speed while the photograph on the right is taken with a low shutter speed. Reproduced as educational fair use [1].

1.1. Motivation

While many computer vision tasks focus on recognizing aspects of a scene, we want to focus instead on aspects of the photography. Determining EXIF data from raw image pixels has several potential uses, including artistic imitation and digital forensics. Most images on the web are actually stripped of this data for this reason.

Moreover, solving this task may provide insight on what constitutes an image's *form*, rather than its *content*.

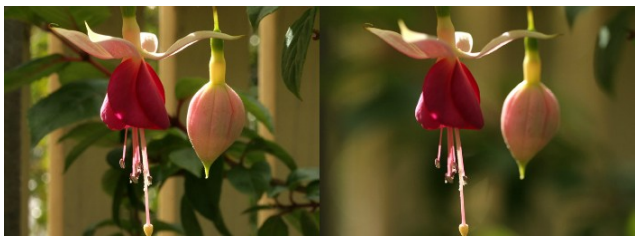


Figure 2: Visual manifestations of aperture variance. The photograph on the left is taken with a small aperture while the photograph on the right is taken with a large aperture. Reproduced as educational fair use [2].

1.2. Task

The input to our algorithm is an image. We then use three convolutional neural networks to output a predicted shutter speed, aperture level, and ISO level separately.

2. Literature Review

Prior research indicates that there has been significant interest in capturing elements of image style. In their paper on extracting and synthesizing texture, Gatys et al. [9] pass an image through the VGG network. They compute a Gram matrix between the filter activations in each layer, and use this Gram matrix as a proxy for the given image's texture.

The same team of researchers [10] was able to transfer artistic style *between* images. Applying methods from texture synthesis, the researchers minimized two parallel loss functions on an image generated from white noise: one with

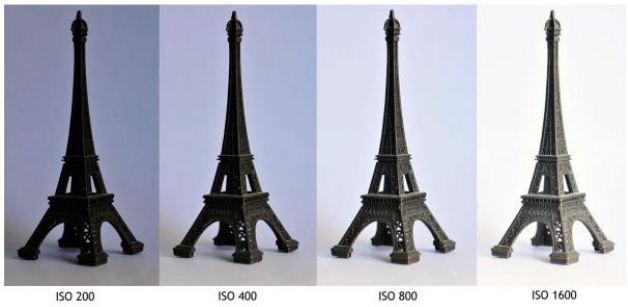


Figure 3: Visual manifestations of ISO level variance. Reproduced as educational fair use [4].

the content of an image, and one with the texture matrices of another image.

Much more recently, Luan et al. [14] added photorealism to the style transfer procedure. Their work adds a distortion penalty to the original style transfer loss function, which has the effect of making color transformations locally affine. As it pertains to our task, Luan et al. worked exclusively with digital photographs.

Apart from understanding style, there exists copious research on extracting content-related information from an image. For instance, Weyand et al. [19] trained a network on images from around the world, learning to approximate the global bounding box in which an image was taken. Their method relies on discovering the features of different territorial landscapes.

Meanwhile, Shih et al. [15] solve for affine color mappings from a database of time-lapse videos. These learned mappings are used to hallucinate time-of-day shifts on static images. Isola et al. [13] tackle this problem more generally, using convolutional neural networks to transform source image sets (daytime pictures, for instance) into target image sets (nighttime pictures).

Zhu et al. [21] extend the work of Isola et al. even further. By creating a novel adversarial architecture called CycleGAN, they are able to transform an input distribution of images into an output distribution of images. Moreover, they are able to do this without explicit input-output pairs of the desired style transformation.

Finally, we note the classical task of image in-painting, where the "content-related information" we are extracting is the image itself. Researchers like Bertalmio et al. [6] have previously approached this problem from a conventional optimization standpoint, but more recently Cai et al. [7] used convolutional networks to perform the in-painting procedure in a single network pass.

While all of these papers describe striking results, their authors work with metadata related to image content, rather than the structural characteristics of a photograph that exist in a vacuum. We are concerned with how a photo was taken, rather than anything present in the photo. The visual effects we hope to capture are functions of the way in which the image was taken, rather than its substance.

For prior work in this area, we turn to the field of digital forensics, which has long preceded the advent of deep learning methods. For instance, Fridrich [8] discusses photo-response nonuniformity (PRNU), a unique noise fingerprint left by camera sensors when taking photos. This fingerprint can be extracted and used to identify individual devices.

PRNU is one of the primary tasks in digital image forensics, and it appears rather suitable as a deep learning classification problem. In 2016, Baroffio et al. [5] trained a relatively simple convolutional network to discriminate photos between 27 camera models. While PRNU is not the specific task we tackle here, the motivation remains the same: extracting the latent properties of a digital sensor reading.

Finally, we mention a few studies related to image processing, particularly where there is no data-driven learning *per se*. These studies demonstrate an active research interest in photographic structure.

Yuan et al. [20] use a pair of images, one noisy (high ISO level and high shutter speed) and one blurred (low ISO level and low shutter speed), to output a de-blurred version of the original image. Such a procedure would be useful in low-lighting environments, where cameras require a trade-off between ISO level (a proxy for noise) and shutter speed (a proxy for crisp capture).

Other groups have worked with motion blur removal, which is today a critical feature on smartphone cameras. Sun et al. [17] use a convolutional network to predict the motion field on a blurred image, which is then used as part of a larger de-blurring pipeline. This approach aligns with the spirit of our project, where our networks must learn local structure in the pixels of images.

3. Dataset

Our primary dataset is MIRFLICKR-1M [12]. This dataset consists of richly annotated images from the photo sharing website Flickr, with a substantial amount of EXIF data retained on each image. We discarded all images that were missing EXIF data for the fields we were concerned with.

3.0.1 Aperture Size

The aperture of a camera is the opening through which light travels, and the term *aperture* is often synonymous with *aperture size*. The visual effects of aperture are usually noted as the depth-of-field, defined as the distance between the nearest and furthest objects in an image.

We noticed that the aperture size was typically reported in one of two formats. The first is the "f-stop" value familiar to photographers, which is the ratio of a camera's focal

length to the diameter of the entrance pupil. The second is a lesser-known format called APEX, or additive system of photographic exposure. APEX was developed to simplify aperture computation in the era of film cameras, and it is related logarithmically to f-stop values.

For our purposes, we convert all aperture readings to precise f-stop values. Aperture settings can vary in very minute gradations, even though these gradations are typically imperceptible to the human eye. To compute a manageable number of class labels from the numerous settings, we convert the target f-stop values to the APEX scale (where the most common values are integral units), and round each value to the nearest integer.

This left us with 13 classes. In practice, however, even these gradations were difficult for our model to learn. We ended up grouping several low aperture classes and several high aperture classes together, training our model on a binary classification task.

3.1. ISO Level

ISO levels measure a camera's sensitivity to light, and the term itself is an artifact of film photography. When lighting is poor and fast shutter speeds allow the entrance of little to no light, higher ISO levels allow these shutter speeds to be used. This creates a visual trade-off, since higher ISO levels tend to add noise, while lower shutter speeds produce blurry images.

Our dataset had a number of different ISO levels along a logarithmic scale, the most common being 100, 200, 400, 800, and 1600. Since there were very few ISO readings that did not conform to these values, we used only those images that did conform. Just like aperture size, however, these ISO gradations proved difficult for our models to learn, and we reduced the problem to one of two classes (ISO 100 and ISO 800).

3.2. Shutter Speed

Shutter speed is the duration for which a camera's sensor is exposed to light. Properly speaking, shutter speed is not a "speed," and for this reason it is often referred to as *exposure time*.

The range of typical shutter speeds starts at 1 second. From there, it doubles repeatedly to get slower (2, 4, 8 and so on), and halves repeatedly to get faster (0.5, 0.25, 0.125 and so on).

We saw that reported shutter speeds in our data were rather non-standard, perhaps due to floating-point miscalculation. As with ISO level and aperture size, we simply rounded shutter speeds to the nearest integer in the log-domain, and in this case we ended up with 20 classes. These were once again condensed for a binary classification setting, using several low and high shutter speeds while leaving out many intermediate shutter speeds.

3.3. Image Sizing

Apart from EXIF extraction, we encountered input images with highly non-standard sizes. MIRFLICKR contains a number of image shapes, ranging from about 250 pixels by 500 pixels to 500 pixels by 500 pixels (that is, the longer dimension of images was almost always 500 pixels).

Since it is hard to find image datasets with good EXIF data, using another dataset was mostly out of the question. We also could not simply resize images to a common shape, since doing so would noticeably distort perceived visual effects of the EXIF fields.

Our solution was to extract a number of random patches of size 128-by-128 from each image, which are fed independently to our convolutional network. The patches are averaged over their penultimate dense layers, and this averaged layer is connected to an output layer with class scores. In other words, we have several "sub-inputs" for each input image, but these "sub-inputs" share all parameters in the network.

The patch extraction procedure adds three model hyperparameters, which we handle generically in our network pipeline. These are the patch height p_H, the patch width p_W, and the number of patches k. For this paper, we fixed $p_H = p_W = 128$, which was an empirical compromise between discarding data and losing important non-localities.

Using image patches reduced the dimensionality of our input and worked around the spatial constraints of our problem. EXIF fields are a characteristic of an entire image, so samples from the image should, in expectation, represent the visual effects of its original EXIF settings.

Patches were centered and standardized across the dataset. We attempted trials both with and without this normalization, wondering if normalization might hurt the detection of brightness and contrast, but no improvements were to be had without normalization.

4. Methods

Given the novelty of our problem, we took an exploratory approach to understand how CNNs were applicable to our task (EXIF estimation). Therefore, we experimented with various deep convolutional neural network architectures to solve different tasks.

While we describe class extraction in the previous section, our holy-grail task was exact, continuous estimation of EXIF values. Our early attempts used regression-based neural network architectures, which would minimize the ℓ_2 loss between the output of a 1 unit fully-connected layer (our predicted value) and the target value.

Across all fields, our models learned to regress to the mean of the data. In other words, the models minimized the ℓ_2 loss by repeatedly predicting the mean target value. The models did not learn to discriminate between different

values and the unique features that define them.

Following those initial experiments, we decided to frame the problem as a classification problem, which convolutional neural networks are much more appropriate for. In concordance with the previous section, we designed our architectures to solve binary classification problems: our networks would predict high aperture versus low aperture, high shutter speed versus low shutter speed, and high ISO level versus low ISO level.

We believe that solving the task on these coarse buckets serves as a foundational proof-of-concept, and has the potential to spur future research.

Furthermore, stark differences in aperture size, ISO level, and shutter speed are easily identifiable to the human eye (as illustrated in Figures 1, 2, and 3), but minor gradations are difficult or even impossible to perceive. This is especially true on an uncontrolled photo-sharing dataset with fuzzy readings (as MIRFLICKR is). Therefore, it made sense to focus on classifying coarse buckets.

4.1. Model

The networks for each of the different EXIF fields were identical. The final architecture is illustrated in Figure 4. As we noted, input image are reduced into a set of k patches. Each patch is fed into the model separately.

The first layers of our network are taken from a pre-trained VGG-16, an image classification network that performed well on the ImageNet Challenge [16]. Specifically, we extracted the first three "blocks" of the VGG-16 network. The first and second blocks consist of two convolutional layers followed by a max-pool layer. The third block consists of three convolutional layers followed by a max-pool layer. The output of the third block is a tensor of dimension 32-by-32-by-256.

We chose to use VGG over other popular image classification networks due to its relative simplicity. Using pre-trained layers of the VGG network meant that we could extract latent features of the images without training on huge datasets for long periods of time. All parameters of the VGG network were frozen at training time.

On top of the VGG layers, the model consisted of four convolutional layers (kernel sizes 1-by-1, 2-by-2, 3-by-3, and 2-by-2), each with F filters and a ReLU activation layer. A fully-connected layer with 256 units followed the last ReLU activation. In order to output a single prediction for a set of patches, the outputs of the fully-connected layer were averaged for all the patches in the set. Because we chose to solve binary classification tasks, the averaging layer was followed by a ReLU, then a 2-unit fully connected layer. (The diagram depicts R bins in general, so we set $R = 2$ for the simplest classification task.)

We calculated the classification error using the cross-entropy loss function on the Softmax of our output (specifically, our target distribution was the one-hot vector for the given class).

A deeper stack of smaller kernels has the same effective receptive field as a shallow stack of larger kernels, but is more memory-efficient and more expressive because it has more non-linearities. This motivated our use of relatively small kernel sizes.

We preferred to reduce the dimensionality of the filters between our convolutional units and correspondingly increase the number of filters, embedding more feature maps to capture a greater variety textures. Furthermore, a batch normalization layer was inserted after every convolutional layer. Batch normalization offered modest gains in performance.

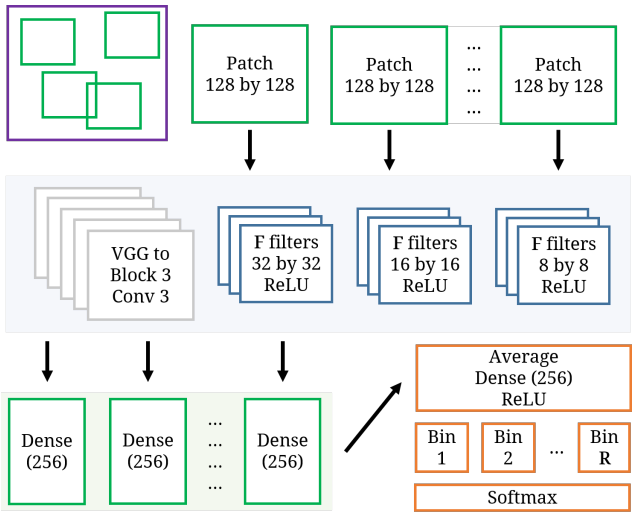


Figure 4: Illustration of our model. The fourth convolutional layer is not pictured due to space constraints on the diagram.

5. Experiments

5.1. ISO Speed and Aperture

Our initial experiments made very little progress on the ISO level and aperture size tasks. Binary classification on these two fields did a little better than random and neither achieved more than 55% accuracy on the validation set in the hyperparameter tuning process.

Therefore, we limit the following discussion to the shutter speed task, which the model performed significantly better on.

5.2. Training

Our final dataset consisted of 5406 images, which were split $80 - 10 - 10$ into 4324 training examples, 540 validation examples, and 542 test examples. We trained on batch

sizes of 32 images (each further subdivided into k patches). We used the Adam optimizer as a canonical choice.

5.3. Tuning

We tested a number of hyperparameters to obtain the best validation accuracy. These hyperparameters were the number of patches k, learning rate, regularization weight, and number of filters F in the convolutional layers. Each model trained for 10 epochs due to limitations in compute power and time. We also noticed that the model would often begin to overfit to the training data around then.

The best performing model trained with a learning rate of 0.001, a regularization weight of 0.001, 64 filters per convolutional layer, and $k = 8$ patches. This model achieved a validation accuracy of 79.44% and test accuracy of 70.66%.

Transfer learning with VGG-16 greatly improved our results. Our best model without transfer learning, which substituted the VGG layers with our own layers, achieved a validation accuracy of 71.11%.

Increasing the number of patches significantly improved the performance of the model. A model with $k = 2$ and all other hyperparameters equal to the best performing model achieved a validation accuracy of 72.03%. Figure 5 illustrates the effect that this hyperparameter had on the performance of the model.

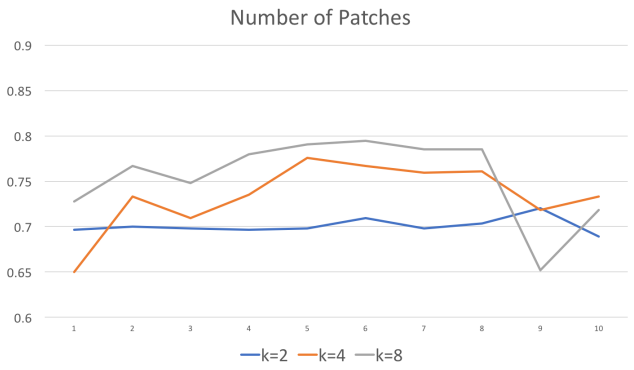
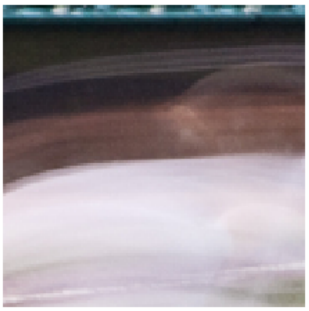


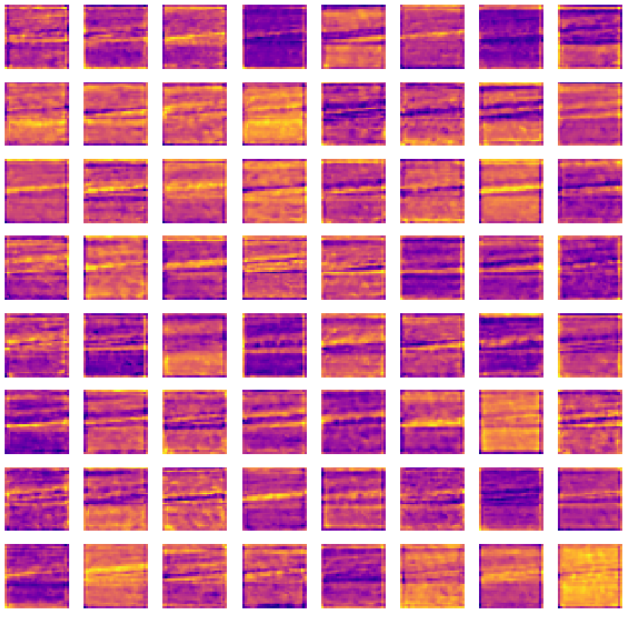
Figure 5: Illustration of the effect that the number of patches had on the performance of the model. Increasing the number of patches led to a not insignificant increase in validation accuracy.

A notable result was that increasing the number of filters on the best performing model (from $F = 64$ filters to $F = 128$) did not improve the validation accuracy after 10 epochs of training. Analysis of these results will follow in the section below.

(a) An example input image with low shutter speed. Reproduced as educational fair use [3].



(b) A patch of the input image from above. Note that it captures part of the shutter speed effect.



(c) Visualization of filter activations on the first non-VGG convolutional layer. Model was run on the patch shown above. Note that the layer filters activate on the horizontal shutter effect.

Figure 6: Visualization of activations for a given patch with low shutter speed.

6. Discussion

Our results were not extraordinary, but they were certainly insightful. We saw that convolutional neural networks have the ability to discern certain photographic elements but not others. There are a number of possible rea-

sons why our model was not able to accurately discern ISO level and aperture size. For instance, different lighting conditions with the same ISO level can produce very different photographs.

Aperture sizes may have been difficult to discern due to our patching method, which does not always capture background pixels. Seeing background textures is critical to understanding perceived depth-of-field in an image, the visual correlate of aperture. Foreground subtraction techniques might have aided in clearly establishing the background.

Our results with shutter speed showed promise. We believe this was because the model could learn to identify the textures caused by a really low shutter speed. Our use of transfer learning likely helped the model identify these textures, since texture activations are embedded in the latent features used by VGG to classify images. For instance, waterfalls are typically photographed with low shutter speeds to capture the effect of motion. Even though our work differentiates itself by not predicting properties of image content, these properties are clearly still useful in a correlative sense.

Increasing the number of patches on the shutter speed model improved our validation accuracy. More patches extract more of the original image in expectation, creating a lower variance estimate of the entire image's dense layer features when averaged. Ultimately, this dense layer is used to compute class scores for the entire image.

Increasing the number of filters improved accuracy, at least up to the point where more than 10 epochs would be required for adequate convergence. As discussed in He et al. [11], accuracy may degrade as models become deeper because they become harder to optimize.

7. Conclusion

We tackled a novel problem that had very little prior research. We showed that some EXIF characteristics are discernible by convolutional neural networks while others are not. We identified probable reasons for the failure of our aperture size and ISO level classifiers. We showed promising results with shutter speed classification and expounded on the reasons behind its performance. In the next section, we consider possible extensions of these results.

7.1. Further Work

Because this was only an initial exploration of the EXIF estimation task, there is ample opportunity for further work to build on top of these results with more expressive models.

One of the most obvious extensions of this work is that of accurately predicting more granular shutter speed classes, but to do so in a tractable way, we need to penalize nearby class predictions less than those that are completely wrong. Such predictions would still be quite useful.

In their paper describing the Inception V3 neural network, Szegedy et al. [18] note that one-hot classifications can also cause overfitting. Intuitively, the network will learn to be overconfident in its predictions. The authors propose smoothing the target labels, assigning a high value to a correct class without creating a one-hot vector. We hope to try this in the next iterations of our model.

In addition, many EXIF variables have co-dependent effects on the latent structure of an image. In the future, we want to build models for some of these (ISO and shutter speed in particular), and observe whether a joint prediction model for these characteristics could do better than the individual models. Indeed, photographers will point out that an image is really an inseparable combination of EXIF settings.

As we develop more expressive models, we hope to extract low-dimensional representations of photographic style. These low-dimensional representations could be used to cluster and find images that are similar in the sense we describe.

Evaluating photographic style would also be valuable as an image editing effect: understand changes in shutter speed (or some other characteristic), and make a photo look like it was taken with a different shutter speed. (In our literature review, we noted the recent work on CycleGAN [21], and view their approach as potentially relevant to this goal.)

7.2. Acknowledgements

We would like to thank the entire course staff for exposing us to a rich field of research. We never thought we would be able to understand state-of-the art papers after only after a quarter of work.

7.3. Contribution

The authors of this paper contributed equally.

References

[1] Wikimedia Commons, 2004. Available at https://commons.wikimedia.org/wiki/File:Windflower-05237-nevit.JPG.

[2] *Image of flowers at two different aperture levels.* Aug 2006. Available at https://i2.wp.com/digital-photography-school.com/wp-content/uploads/2006/08/aperture-depth-of-field.jpg.

[3] *Low shutter speed image of baseball player.* Jul 2009. Available at http://photobooksolutions.com/blog/wp-content/uploads/2009/07/shutterspeed-5.jpg.

[4] *Image of Eiffel Tower model at different ISO levels.* Jan 2017. Available at http://skillonpage.com/wp-content/uploads/2017/01/camera-iso-speed.jpg.

[5] L. Baroffio, L. Bondi, P. Bestagini, and S. Tubaro. Camera identification with deep convolutional networks. *arXiv preprint arXiv:1603.01068*, 2016.

[6] M. Bertalmio, G. Sapiro, V. Caselles, and C. Ballester. Image inpainting. In *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, pages 417–424. ACM Press/Addison-Wesley Publishing Co., 2000.

[7] N. Cai, Z. Su, Z. Lin, H. Wang, Z. Yang, and B. W.-K. Ling. Blind inpainting using the fully convolutional neural network. *The Visual Computer*, 33(2):249–261, 2017.

[8] J. Fridrich. Digital image forensics. *IEEE Signal Processing Magazine*, 26(2), 2009.

[9] L. Gatys, A. S. Ecker, and M. Bethge. Texture synthesis using convolutional neural networks. In *Advances in Neural Information Processing Systems*, pages 262–270, 2015.

[10] L. A. Gatys, A. S. Ecker, and M. Bethge. A neural algorithm of artistic style. *arXiv preprint arXiv:1508.06576*, 2015.

[11] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Computer Vision and Pattern Recognition (CVPR), 2016 IEEE Conference on*, 2016.

[12] M. J. Huiskes and M. S. Lew. The mir flickr retrieval evaluation. In *Proceedings of the 1st ACM international conference on Multimedia information retrieval*, pages 39–43. ACM, 2008.

[13] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros. Image-to-image translation with conditional adversarial networks. *arXiv preprint arXiv:1611.07004*, 2016.

[14] F. Luan, S. Paris, E. Shechtman, and K. Bala. Deep photo style transfer. *arXiv preprint arXiv:1703.07511*, 2017.

[15] Y. Shih, S. Paris, F. Durand, and W. T. Freeman. Data-driven hallucination of different times of day from a single outdoor photo. *ACM Transactions on Graphics (TOG)*, 32(6):200, 2013.

[16] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[17] J. Sun, W. Cao, Z. Xu, and J. Ponce. Learning a convolutional neural network for non-uniform motion blur removal. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 769–777, 2015.

[18] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2818–2826, 2016.

[19] T. Weyand, I. Kostrikov, and J. Philbin. Planet-photo geolocation with convolutional neural networks. In *European Conference on Computer Vision*, pages 37–55. Springer, 2016.

[20] L. Yuan, J. Sun, L. Quan, and H.-Y. Shum. Image deblurring with blurred/noisy image pairs. In *ACM Transactions on Graphics (TOG)*, volume 26, page 1. ACM, 2007.

[21] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. *arXiv preprint arXiv:1703.10593*, 2017.