Enhancement Parameters and Perceptual Loss Functions for Personally Stylized Post-Processing

Stephanie Liu, Bharath Srivatsan
Department of Computer Science, Princeton University
{shliu, bharaths}@princeton.edu

Abstract

While most existing image enhancement applications offer automatic retouching options, users have unique aesthetic preferences that are different enough to warrant personalization [5]. We investigate the possibility of learning style transfer patterns for photographers and photo shoots in order to algorithmically post-process new images in a way that preserves the natural complexity of individual styles. We experiment with different models and ultimately contrast the effectiveness of a generalized enhancement vector learning approach with a perceptual feature loss function approach and a feature difference learning approach. In all three of our approaches, we use images from the MIT-Adobe FiveK dataset and a self-collected dataset drawn from campus photographers. Our best approach involves extracting enhancement parameters from images and matching new images to visually similar training examples. We find that by doing so, we are able to make realistic, incrementally improving edits.

1. Introduction

Photo editing is becoming an increasingly important part of modern digital literacy. Professionals have long used digital retouching and modification techniques to ready their pictures for sale, but the rise of photo-sharing applications like Instagram, Snapchat, and VSCO have meant that regular users are beginning to process their photos as well. Some of these applications, in an attempt to simplify the user experience, offer automated retouching options that can apply basic modifications to photos. However, many users attempt to create a cohesive 'aesthetic' across the photos they edit, just as a professional might attempt to ensure consistency in style among post-processed photos from an individual shoot. Furthermore, these consistent editing decisions often differ sharply among users—what is pleasing to one may not be to another. This means that existing, standardized features like auto-enhance do not successfully meet a core user need for personalized automated editing.

The central thrust of this work is to develop a model that can learn and replicate a specific individual's image processing style. The motivation for this is twofold: for one, regular users of services like Instagram edit many photos to match their unique styles, but often have to rely on standardized enhancements or filters as a baseline for their approach. Second, in our conversations with amateur and professional photographers alike, we discovered that the thousands of photos taken for individual shoots often end up being edited in consistent ways. Applying these intended changes algorithmically could dramatically improve this process. Given the comparative ease of collecting a dataset from the latter population, we focus on the second question in what follows.

1.1. Problem Statement

Put more formally, we attempt to study the following problem: Images are post-processed differently based on 1) their underlying visual components and 2) the aesthetic preferences of the editor e who is modifying them. Given a set of images $I_1 \dots I_n$ and their edited versions (tagged by editor) $I'_{1,e} \dots I'_{n,e}$, can we create a program that, given a new image I_{n+1}, can propose an edited version in the style of a particular editor $\hat{I'}_{n+1,e}$?

We constrain the types of edits we permit in a few ways:

- We focus on per-pixel adjustments. This means we allow for edits to tone, saturation, hue, sharpness, etc., but not for image crops or rotations that transpose or remove pixels. This is because the techniques necessary for identifying objects of interest in images (so as to make transposition decisions) are orthogonal to the stylistic exploration techniques we hope to use.

- We do not permit constructive edits, meaning we did not allow for edits that added components to images or that blended in elements from other images. Again,

the types of processing required to predict such edits are unrelated to the scope of this work.

1.2. Overview

We begin in Section 2 by outlining some related works. Section 3 describes our approach, including our dataset, each of three methodologies, and our implementation details. Section 4 overviews our results and evaluation. We finish in Sections 5 and 6 with a discussion of our future work and conclusions.

2. Related Work

Our work involves two approaches that are based on learning personalized image enhancements and a third on perceptual style transfer.

2.1. Personalization of Image Enhancement

For users who wish to replicate their unique, consistent styles across post-processed images, automatic image enhancement tools are severely lacking as they can only make generic edits. Kang *et al*. [5] determined that user preferences in image enhancement were different enough to warrant personalization, and that personalization is an important component for improving the subjective quality of images.

The system in [5] for personalizing image enhancement learned a user's preferences by observing her edits on images. It then enhanced any (unseen) test image by finding the training image most similar to it and directly applying the training image's enhancement parameters. To get enough training images, [5] required that user edited 25 pictures (those considered the most 'unique'), and then extrapolated those edits to 5000 images. The image enhancement pipeline extracted 5 parameters associated with contrast and color correction from the user's edits: a power curve parameter, two S-curve parameters, temperature, and tint.

The learning component of the system applied to each new, unseen photo the same enhancement parameters previously applied to the "closest-looking" manually-edited picture. Thus, the central contribution of this system is in defining the "closest" picture via a distance metric. The model proposed learned its metric through minimizing a convex objective function that examined all pairs of images and measured how much the distance in image space differed from the distance in a "parameter space." This parameter space incorporated many visual features, including histograms in RGB, HSL (hue, saturation, lightness), and intensity channels. The training images for the distance metric were automatically enhanced, but it is assumed that the learned distance metric using these proxy parameters leads to reasonable estimates of the relevant distance metric in the image space. Once the distance metric was learned, given test image I_{n+1}, the model found the closest training image I_k and applied the enhancement parameters of I_k to it.

Caicedo *et al*. [2] built upon the work of Kang *et al*. [5] by proposing a probabilistic graphical model for jointly predicting enhancement preferences. This work explicitly encodes similarity across images, and groups users into clusters. In other words, the model showed that similar users have similar enhancement parameters for similar images; therefore, new images could be enhanced based on similarities to existing images edited by the cluster to which a given user belonged. The enhancement space was determined by the same 5 enhancement parameters as [5]. However, unlike in [5], Mahalanobis distance was used as the distance metric to measure similarities between images. Mahalanobis distance, applied to the visual feature space, learns a parametrized square matrix A, which encodes weights associated with different features and correlations amongst them in order to reflect the disparity in the enhancement parameter space. Then, a probabilistic model was trained to encode dependence of enhancement parameters on image content, as well as the clustering of users based on their editing styles. To enhance images from a new user, the new user was asked to enhance a subset of images from the training set to determine the user's cluster membership, which is the similar to the approach used in [5].

Bychkovsky *et al*. [1] refined this problem to three practical cases: 1) reproducing the adjustment of a single photographer given a large collection of examples 2) learning adjustment personalization using a carefully chosen set of training photos like in [5], and 3) introducing difference learning to free the user from using predetermined photos for training. To do so, they attempt to learn the remapping curve between input and output color luminances on the CIE-Lab color space. They then use supervised learning with a number of features (including intensity distributions, scene brightness, and equalization curves) to learn the adjustment of brightness, saturation, and contrast associated with various unique editing styles. Though they focus on tonal adjustments, they find that such modifications explain most of the variance between the edits made by different photographers in their set. Bychkovsky *et al*. [1] also contributed to the space of image enhancement by developing and releasing the MIT-Adobe FiveK dataset (see Figure 1 and Section 3.1).

2.2. Style Transfer with Perceptual Losses

The power of CNNs has been applied successfully to many image processing tasks, and has driven research in style transfer modules. The key insight of Gatys *et al*. [3] is that CNNs could separate the content and style of images. The result was a system that was able to synthesize images that combine the content of one photograph with the style representation of another. This model was trained by exam-

(a) Expert A (b) Expert C

Figure 1: An example of edited images from the MIT-Adobe FiveK dataset. Each original photo is edited by 5 different professional photographers.

ining features extracted from a pre-built VGG-16 Network, but is computationally expensive since each step of the optimization problem requires both a forward and a backward pass.

The neural network in Johnson *et al*. [4] is less computationally expensive because it trains feed-forward networks to quickly approximate the solutions to each optimization problem. Like [3], [4] utilizes perceptual loss functions instead of per-pixel loss functions. In other words, rather than looking at the per-pixel differences between images, the loss function uses the difference between high-level image feature representations extracted from pre-trained CNNs. By combining the benefits of perceptual loss functions and feed-forward transformation networks, the style transfer results they achieved were similar to those in [3], but three orders of magnitude faster.

3. Approach

In developing an image enhancement algorithm to solve our specific problem statement (personalized processing over small sets of ground truth examples), we built off of the past research described above. Many algorithms that focus on image enhancement or style replication, though, require in-feasibly large datasets for training or are too expensive to run efficiently. We offer three possible architectures to tackle this problem, but ultimately find that only the third method is sufficiently efficient and effective.

3.1. Dataset

We rely on two sources of data for testing this project. First, we use the MIT-Adobe FiveK Dataset.[1] This is a collection of 5,000 photos that are each manually annotated and edited by 5 trained photographers [1]. Figure 1 shows an example of the images in this dataset. Because each photo included has both a baseline and various edited versions, we are able to compute perceptual or stylistic distance metrics across edits to identify the unique tendencies of individual editors.

In addition, we collected our own data set by compiling 250 pairs of pre- and post-edited photos from two campus photographers. In this dataset, split into different photo shoots (each with consistent internal editing styles), we are again able to map editing decisions for individual photos conditioned on editor.

These two datasets, in conjunction, allow us to test our work on two different scenarios. Using MIT-Adobe FiveK, we are able to assess if our methods can adequately capture the differences in processing that individual editors use for the same photo. Using our campus dataset, we can test how our algorithms perform on different pictures from photo shoots with consistent styles. More generally, though we have access to relatively large sets of images, we also evaluate our approaches on smaller subsets to assess performance for real-world photographers who may not be able to supply large training sets.

When training method 2, below, we also rely on a subset of images from the MS-COCO dataset.

3.2. Approaches

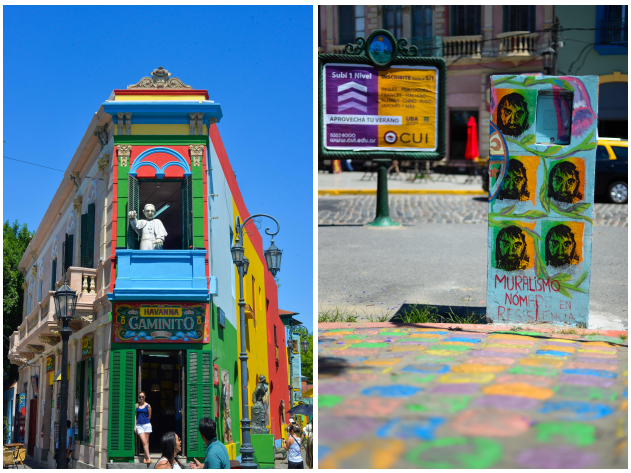
We propose three candidate approaches. First, we experimented with generating processing masks for individual photo shoots to then use in the perceptual style transfer module from [4]. Second, we attempted to directly learn feature differences between pre- and post-processed images, mapping image features to suggested stylistic changes. Finally, we learn image modification preferences for individual photographers by applying modifications from similar pictures that they had previously edited. This method is inspired by the approach in [5].

3.2.1 Perceptual Style Masking

Johnson *et al*. [4] released a Torch-based implementation of their feed-forward style transfer network based on perceptual losses. Their implementation takes as input a photograph and a piece of art, and outputs an image that matches the content representation of the photograph and the style representation of the piece of art.

In our first attempt, we iterated on this model. Intuitively, if an editor's typical post-processing behavior could

[1] https://data.csail.mit.edu/graphics/fivek/



(a) Photo A (b) Photo B

Figure 2: An example of two different images from our campus dataset. These photos were taken from the same shoot, and thus share many stylistic similarities. Each is edited by the original photographer (in this case, Bharath).

be represented as an artistic 'style,' it could then be applied directly to new images through the style transfer network. This implementation had three stages:

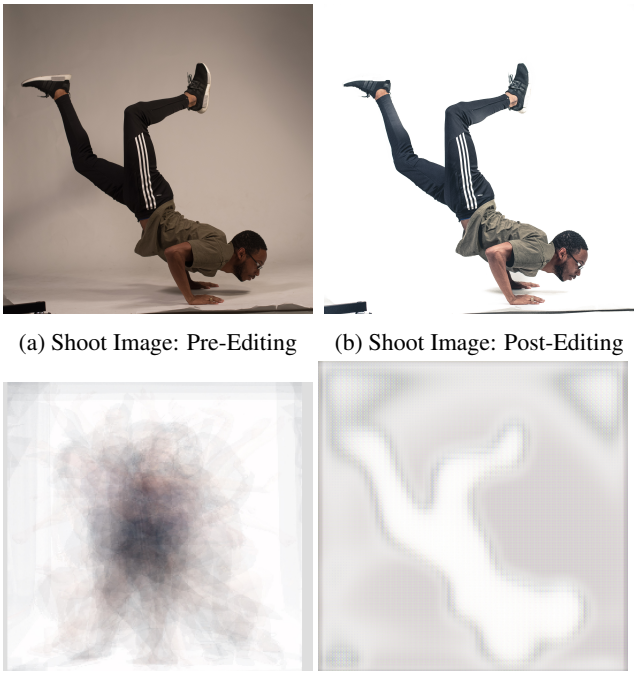
1. Style mask generation: First, we needed to generate stylistic 'mask' representations of the post-processing that individual artists performed. We experimented with two kinds of mask representations:

   (a) Pixel-difference mapping: This approach calculated per-pixel differences between pre- and post-processed images in a shoot and represented these differences visually, in the hopes of capturing specific processing steps to apply en masse in a style transfer.

   (b) Output mapping: This approach blended output images from a given shoot, in the hopes of retaining broad image characteristics (center-sharpness, background-brightness, etc.) that were common to post-processed images.

2. Perceptual loss-based training for model generation: Once a style mask was generated, we needed to build a style application model that could then be applied to images. As in [4], we used a perceptual loss function that assessed stylistic loss against various features of a pre-trained VGG-16 net to learn optimal stylistic transforms for a large subset of the MS-COCO dataset.

3. Style transferal: Finally, we used the pre-trained model and a new content image to generate a combined image output.



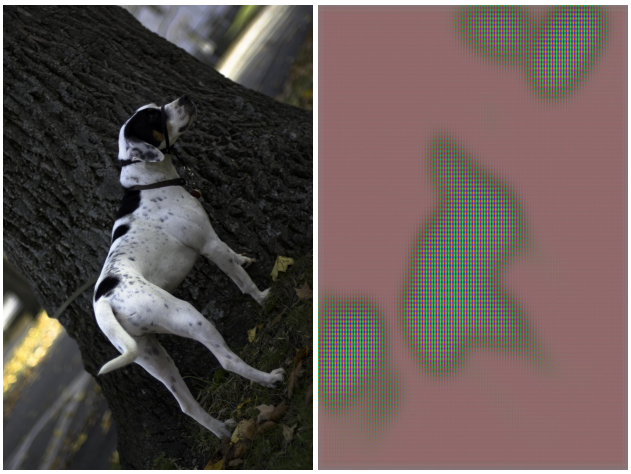
(a) Shoot Image: Pre-Editing (b) Shoot Image: Post-Editing

(c) Generated Mask (d) Mask Applied to Image

Figure 3: An example of an image from one of the photoshoots in our campus dataset (pre- and post-processing), the composite mask generated by averaging output image pixel values from all photos in the shoot, and the same image with the mask applied to it by the perceptual style transfer module. As 3c visually indicates, the mask seems to capture the dark-foreground, white-background style of the images in the shoot

For each mapping type, we experimented with different specific implementations, using various combinations of pixel and feature proportional averages. See Figure 3c for an example of this process applied to one shoot.

We ultimately found that this approach was not suited for the problem at hand. As we discovered, [4]'s perceptual loss function learns to apply style representations in a complex way. Rather than picking out simple image features from the proposed style in isolation, the loss function is trained to represent output images that encapsulate a combination of features and how they manifest in concert. We suspect that this is why, for instance, in Figure 3d, the output image uses the dark-light image boundaries and the fuzziness of the foreground to generate images that abstract various vital content features and preserve boundaries instead. Furthermore, the style reconstruction loss used in [4] penalizes differences in colors and textures. As a result, the composite mask that we believed was an accurate representation of a given photo shoot was unsuited for this loss, as the colors and patterns of the original image were lost. This



(a) Image from Different Shoot (b) Mask Applied to Sample

Figure 4: An example of an image from a different photo shoot with the same mask from 3c applied to it. As 4b demonstrates, applying masks from different shoots generates unintelligible results

explanation might also account for the unexpectedly varied results generated by applying masks to images from different shoots (see Figure 4b)—though poor performance was expected, the particular output (with its colors and patches) was not.

3.2.2 Feature Difference Learning

Our second approach attempted to directly map various input image features to an enhancement vector that could represent the changes made during processing. In this way, we intended, we could create a model for each user that would incorporate both the visual features of an image and the stylizations that a given editor typically performed when deciding on how to process it.

More specifically, we wished to represent an image based on its visual features and/or per-pixel CIE-Lab data to train a model that could predict pixel-by-pixel CIE-Lab transformation suggestions. Given that this was a structured prediction problem, we investigated both a structured-kNN and a structured-SVM as candidate enhancement-prediction models. Unfortunately, both methods were difficult to train and neither successfully converged on the data at hand. This was unsurprising, as the dimensionality of the possible output space was almost as large as the input vector information available.

To reduce output feature dimensionality, we attempted to use a simplified enhancement vector that consisted of parameters for image-wide transformations instead. This way, we could learn a few values for brightness or saturation transforms that could represent the processing steps that needed to be taken. Unfortunately, this was not able to solve the issue—neither model converged on this problem setup either.

3.2.3 Generalized Enhancement Vector Learning

Based on the issues we found above, we settled on a third approach. This method aimed to build a reference set that learned a generalized enhancement feature vector for each image. Then, for each new (test) image, this module found the 'nearest' images in the historical set and generated a composite version of the enhancements applied to them. Thus, this process applied specific image modifications (rather than masks for entire photo shoots as in section 3.2.1) but did so without making the problem space too open ended (as was the issue with directly learning enhancements in section 3.2.2).

This approach is similar to [2], [5] but addresses shortcomings in both implementations. In [2], a editor cannot belong to multiple editor clusters, and is therefore limited to one image enhancement style over all images. We found that in the dataset we created, professional photographers varied their editing styles dramatically based on different photo shoots. Therefore, our approach can more optimally apply different enhancement transformations depending on the type of input image. Furthermore, since we can leverage an editor's current set of pre- and post-processed images, we do not need to generate the kind of training set used by [5]. Finally, though [5], [2] developed a system where each user enhanced < 25 carefully-chosen images, this small number of examples may not properly represent the kind of photos that each editor most commonly works with. By contrast, our methodology theoretically improves in accuracy as a user edits more photos.

Our processing phase learned four kinds of image-wide enhancements for each pair of $I_k, I'_{k,e}$: color (saturation), brightness, contrast, and sharpness. We used the Python Pillow module to compute each enhancement 'factor' according to the formulas below:

$$\alpha_{color} = \frac{I' - I_{greyscale}}{I - I_{greyscale}}$$

$$\alpha_{brightness} = \frac{I' - I_{black}}{I - I_{black}}$$

$$\alpha_{contrast} = \frac{I' - I_{greyscale,avg}}{I - I_{greyscale,avg}}$$

$$\alpha_{sharpness} = \frac{I' - I_{blur}}{I - I_{blur}}$$

where $I_{greyscale}$ is a greyscale (unsaturated) version of I, I_{black} is a black image of size I, $I_{greyscale,avg}$ is a greyscale version of I with the average pixel value placed throughout (no contrast), and I_{blur} is I with a blur filter applied to it.

Once we had, for each editor, a mapping of each of their input images to enhancement feature vectors α:

$$\alpha = \begin{bmatrix} \alpha_{color} \\ \alpha_{brightness} \\ \alpha_{contrast} \\ \alpha_{sharpness} \end{bmatrix}$$

we then could generate enhancement vectors for new images. We do so by developing a distance metric to group images by visual content and by using a lightweight clustering algorithm to identify the closest past images for each test picture. We decided to use a color histogram similarity metric due to color histogram importance in distinguishing between photos from different shoots. This distance metric methodology differs from both [5], [2].

Once a proposed enhancement vector was generated for a test image, we then applied each image enhancement sequentially to the pre-processed picture using built in methods from Pillow. Select results from this approach are shown in Figure 5.

3.3. Implementation

All of our code was run on an AWS p2.xlarge Deep Learning AMI (Ubuntu) Version 8.0 instance, which is preinstalled with popular deep learning frameworks. For our feature difference learning approach with style transfer, we use open-sourced implementations by Johnson *et al*. and antlerros.

4. Results and Evaluation

We evaluated our generalized enhancement vector approach in three ways.

1. We assessed the quality and 'naturalness' of our post-processed images by surveying 20 amateur photographers
2. We assessed whether our images fit with editor aesthetics by surveying our campus photographers
3. We assessed our images' conformity to an editor's specific stylizations by computing color space differences to ground truth post-processed pictures

We used three metrics because we wanted our output images to at the very least be high quality and seem natural, and ideally match an editor's general preferences (if not the exact modifications that they would have made).

First, we attempted to evaluate our output images by asking 20 amateur photographers to rank the pre-processed, ground truth (editor post-processed), and generated versions of a randomly-selected subset of images. Each respondent picked the image they liked the best and least from each set of three. If our generated images were regularly selected over the ground truth versions, that would suggest that we had created natural-looking images. If, on the other hand, our images were the worst of the three, it would suggest that we had applied 'enhancements' that had in fact lowered the baseline aesthetic of the image. If our enhancement ability was perfect, our images would be selected in equal measure as better than/worse than the ground-truth versions.

We found that across 95% of images surveyed, our post-processed version was preferred as least as much as the original image (ie. it was voted 'worst' no more than the baseline version). Furthermore, we found that 40% of our generated images in the survey won more votes for 'best' than their ground truth post-processed versions. Taken together, these findings imply that we broadly have successfully enhanced images naturally and without massive imperfections. Oddly enough, 100% of generated images based on the MIT-Adobe FiveK dataset were ranked as better than the versions edited by Retoucher C.

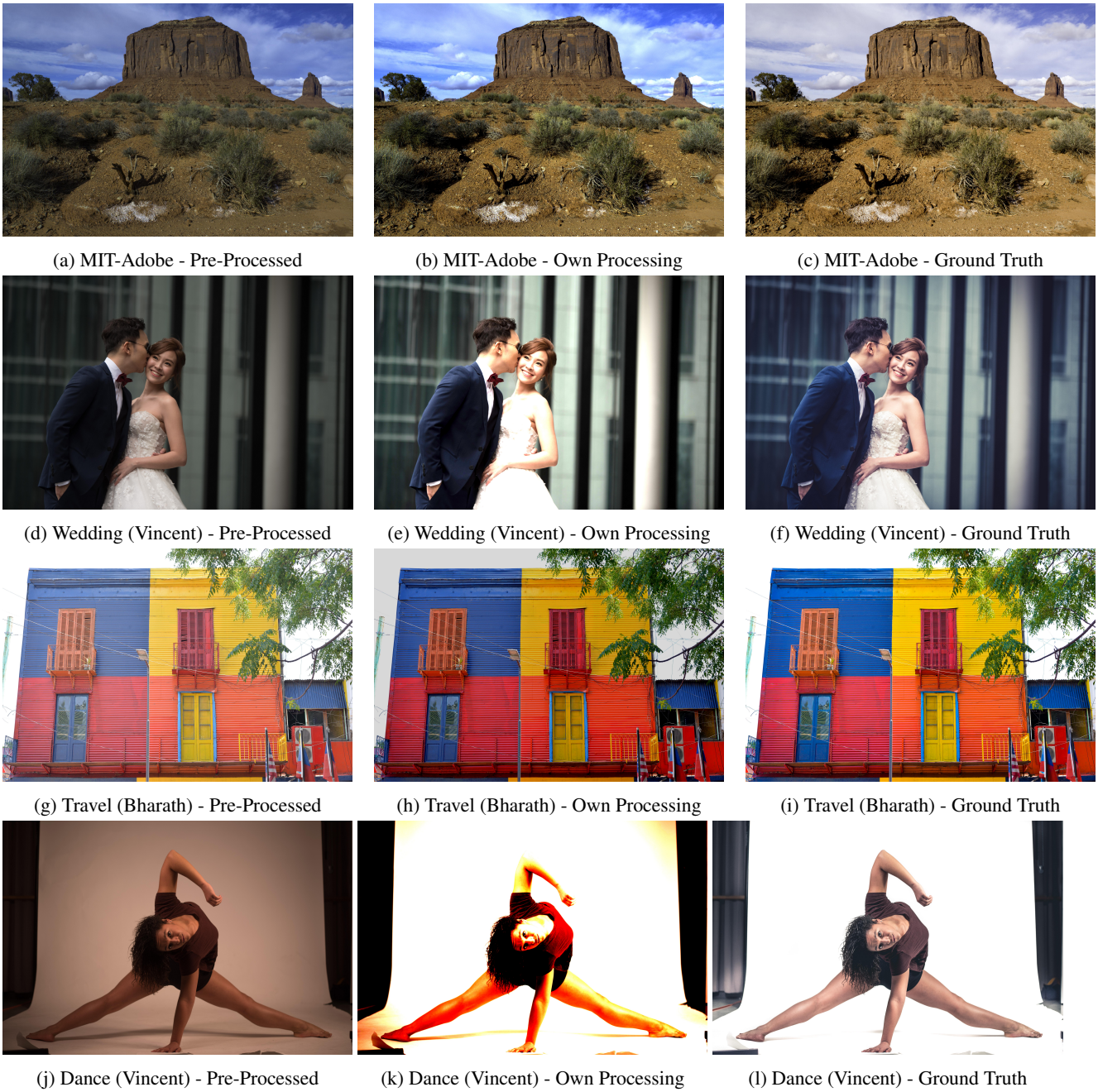
Our second evaluation metric was intended to assess whether our post-processed images broadly fit with an editor's aesthetic. Even if an image was objectively ranked as decent, if its particular editor found that they'd prefer the baseline version to it, we'd likely have generated images that substantially diverged from their unique styles. To evaluate this, we asked both campus dataset contributors to indicate for a subset of images whether they thought our generated outputs were "closer to their ideally enhanced versions" than the pre-processed images.

This time, we found that 86% of our post-processed images were preferred to the originals. This suggested that most of our edits did not stray sharply from the unique styles of editors. When asked about some examples (c.f. figure 5k) that seemed to exhibit worse-than-baseline performance but were still selected as closer, one of our campus photographers noted that these images were more in line with the intended features of their output, and in fact exhibited interesting tendencies of their own.

Finally, we attempted to quantitatively assess regeneration accuracy by computing color space differences to the ground truth edited versions. We did so by computing the L1-norm difference between the CIE-Lab color space vectors of our output images and the ground truth (editor post-processed) versions.

In doing so, we found an average difference of 4.74 for a randomly selected subset of images. For context, an average CIE-Lab difference of 2.3 is a just-noticeable difference, while the difference between black and white images is 100 [1]. While it is difficult to definitively benchmark our results, this suggests that these are perceptible yet not excessive differences.

This, though, is an imperfect metric. While CIE-Lab difference serves as a functional numerical metric, comparing



(a) MIT-Adobe - Pre-Processed (b) MIT-Adobe - Own Processing (c) MIT-Adobe - Ground Truth

(d) Wedding (Vincent) - Pre-Processed (e) Wedding (Vincent) - Own Processing (f) Wedding (Vincent) - Ground Truth

(g) Travel (Bharath) - Pre-Processed (h) Travel (Bharath) - Own Processing (i) Travel (Bharath) - Ground Truth

(j) Dance (Vincent) - Pre-Processed (k) Dance (Vincent) - Own Processing (l) Dance (Vincent) - Ground Truth

Figure 5: Sample results from our third approach. The first column shows pre-processed images, the second column shows the results of our module, and the third column contains the ground truth edited versions (by the editor whose style we attempted to emulate). The first three rows demonstrate successful attempts (to varying degrees), while the final row exemplifies a failed shoot (see discussion of structural masks in section 5)

tonal representations does not allow for more holistic assessments of image identity. There is a lot to enhancement quality that may not be captured by this approach.

5. Discussion and Future Work

Our generalized enhancement vector learning model was much more successful at creating subjectively better enhancements on the MIT-Adobe FiveK dataset than on the dataset that we created from campus photographers. From

observation, the enhancement of images in the MIT-Adobe FiveK dataset by the 5 trained photographers seemed more muted and conservative than the enhancements made by campus photographers. This suggests that our enhancement parameters improved upon muted images to the tastes of our survey respondents but could not do so for more heavily-stylized enhancements.

Furthermore, our applied enhancement parameters typically over-exaggerated the contrast and brightness of images. Since we predicted enhancement parameters independently of each other, and since photo post-processing often increases each of the parameters, the over-exposure of images in our post-processing can likely be attributed to the fact that our parameters interface with one another but are learned separately.

We also identify that the biggest shortcoming of our generalized enhancement feature vector approach is that it performs poorly on images that were heavily post-processed based on image structure and semantics. For example, in Figure 4a, the background of the dancer is made all white, while the saturation of the main subject of the image is enhanced. Here, different modifications were made to different structural components of the image; our image-wide enhancement parameters could not capture this nuance. To resolve this issue, we could have further limited edits in our dataset by permitting only generalized adjustments, rather than masks that applied only to specific components of pictures. This would still have allowed for isolated processing of particular colors or tone curves.

Future work ought incorporate more versatile mechanisms to adapt to these sorts of enhancements. Yan *et al*. [6] recognized that artistic enhancement is typically semantics-aware. They trained a deep neural network (DNN) on image descriptors that accounted for the local semantics of an images when generating enhancements. Although this approach is limited to learning only one enhancement style, it provides a good example of effective image descriptors and their applications to this space.

6. Conclusion

In this paper, we present and detail three approaches to learning the post-processing styles of different photographers and photo shoots. In our perceptual style masking approach, we discovered that our approach of applying a perceptual loss function for minimizing mask style loss was not suited for preserving the integrity of original images. During our feature difference learning approach, we were unable to train a model that could predict enhancement vectors with per-pixel transformation suggestions given our data and the scale of the optimization problems it created.

Our third and most successful approach involved generalized enhancement vector learning. We learned the enhancement feature vectors for each image in our dataset. Then, given a test image I_{n+1}, we found the closest training image I_k and applied its enhancement feature vector to the test image. To locate the closest training image, we used a distance metric based on the color histograms of images. We found that this approach yielded natural images that were preferred in most cases to pre-processed versions by both a set of surveyed photographers and our original campus dataset contributors. We suggest that mechanisms to improve the robustness of our enhancement feature vectors be explored in future work.

7. Acknowledgements

Thank you to Professor Olga Russakovsky for your support and guidance, and to the other members of COS 598B for your helpful feedback and insightful comments this past semester. Thanks to Vincent Po for providing us with many stylized images.

References

[1] V. Bychkovsky, S. Paris, E. Chan, and F. Durand. Learning photographic global tonal adjustment with a database of input/output image pairs. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 97–104. IEEE, 2011.

[2] J. C. Caicedo, A. Kapoor, and S. B. Kang. Collaborative personalization of image enhancement. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 249–256. IEEE, 2011.

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

[4] J. Johnson, A. Alahi, and L. Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *European Conference on Computer Vision*, pages 694–711. Springer, 2016.

[5] S. B. Kang, A. Kapoor, and D. Lischinski. Personalization of image enhancement. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 1799–1806, June 2010.

[6] Z. Yan, H. Zhang, B. Wang, S. Paris, and Y. Yu. Automatic photo adjustment using deep neural networks. *ACM Transactions on Graphics (TOG)*, 35(2):11, 2016.