Let There Be Color: Deep Learning Image Colorization

Justin Olah
Stanford University
jolah@stanford.edu

Jenny Yang
Stanford University
jjyang1@stanford.edu

Abstract

Colorization is a timeless task that has been around since the invention of art. With the improvement of artificial intelligence, colorization has become possible as a machine learning task. Colorization models take as input a black and white image and produce an image in color. In this paper, we define colorization as a regression and classification task and train models on two different datasets to tackle this problem. Additionally, we explore a weighting technique on rare colors and the implementation of skip-connections in the network. Through a combination of qualitative analysis involving tests of robustness and manual inspection, as well as quantitative analysis, we conclude that our U-Net model without class rebalancing weights produces the most accurate and colorful images. In addition, we observe that these models can learn semantic information about the scene without having any labels, inspiring future work of leveraging our present approach to tackle other tasks like semantic segmentation.

1. Introduction

Ever since the dawn of black and white photography, people have been searching to add color to decades of black and white photographs, videos, and sketches. [13] Traditionally, colorizing images has been the job of an artist where each pixel has to be individual changed. In recent years, however, deep learning computer vision models have greatly accelerated this task and have even automated the process.

Colorization as a machine learning task is defined where a model takes as input a black and white image and outputs the corresponding colored image. Colorization is often treated as a regression task or classification task, both of which we we will explore in this paper. We were motivated to explore colorization as a problem because of its fascinating implications beyond just color. Models that learn to color well often have learned other extremely relevant features about the image, such as segmentation, texture, and depth.

In our paper, we explore the use of regression and classification loss functions for a convolutional neural net based approach to image colorization. Beyond loss functions, we explore the effect that skip-connections and considering the frequency of different colors have on both learning information from the input data and applying that learned information in colorizing a photo. In addition, we quantitatively and qualitatively evaluate the performance of our models through robustness testing, error analysis, and other techniques.

2. Related Work

Early work on colorization involves non learning methods often using outside reference images. Gupta et al [4] introduced the method of colorizing images using super pixel extraction with a similar target image. Features are mapped from one image to the other and are subsequently colorized. Early machine learning colorizers utilize regression. Cheng et al [2] proposed using L2 regressive loss to train a deep learning network. However, the results were often desaturated as the model predicts the safest guess of color.

More state of the art methods treat colorization as a classification task. Zhang et al [14] and Charpiat et al [1] developed a method of treating each pixel as a classification task of quantized color bins in the AB color space, and they found that this produced more colorful images than traditional regression. Similarly Larson et al [9], predicted hue and lightness histograms to address the multi-modality of colorization. For example, a car can be red, blue, grey, etc. with different probabilities.

Some methods aid models with additional information. Zhang et al [15] assisted colorization models with sparse user input to guide what colors are chosen in certain sections of an image. Hung et al [6] similarly used feature engineering methods to input edges in addition to a black and white image with the goal of preventing color bleeding.

Colorization is also used as a pretext task for tasks like image classification and segmentation. Gonzales-Santiago et al. [3] used colorization as a pretext task for semantic segmentation of urban land color. A colorization model naturally must learn the semantics of a scene, like what the

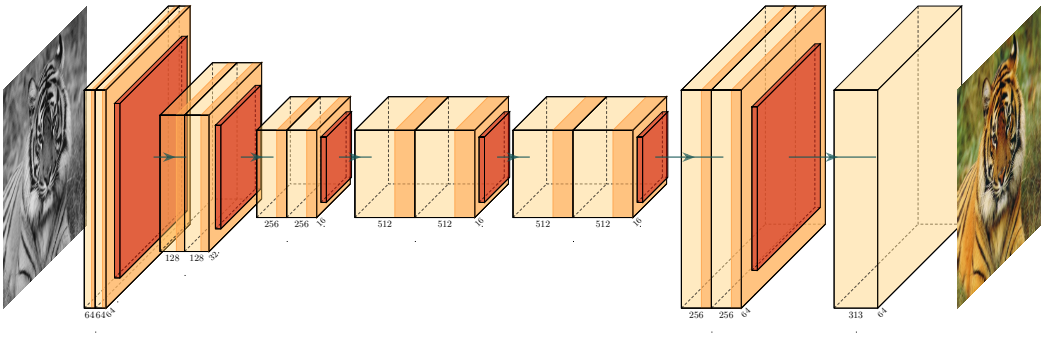


Figure 1. Our Classification Model. Between each group of convolution layer is a BatchNorm layer

difference between an apple and an orange is, in order to colorize each object. This knowledge can be leveraged to assist in other tasks.

Generative adversarial networks (GAN) further accelerate colorization by training a discriminator which learns what images are real and fake. Zhu et al [17] mapped inputs to outputs and then outputs back to inputs to create a cyclic structure. This method is known as CycleGAN and allows for unpaired image-to-image translations, opening up a world of possibilities where ground truth photos are not necessary. For example, a black and white photo of a teddy bear can be colorized in the style of Van Gogh even though he has never painted that before. Nazeri et al. [11] similarly addressed the multi-modality of colorization by using a conditional GAN.

3. Methods

3.1. CIELAB Color Space

While most people are familiar with working in the Red-Green-Blue (RGB) color space, research in colorization is often developed in the Lightness-A-B (LAB) color space where A contributes to green-red and B contributes to blue-yellow. We input the black and white lightness channel of an image into the model and output the predicted A and B channels values of the image. The A and B channels can then be added to the inputted black and white image to produce a image in color.

3.2. Baseline Regression Model

Our baseline network models colorization as a regression problem. The model takes as input a continuous L-channel and outputs the continuous AB-channels. The loss is the L2 distance (also known as mean squared error) over the height and width of the image. We define the ground truth color $Y \in \mathbb{R}^{h \times w}$ and the predicted color as $\hat{Y} \in \mathbb{R}^{h \times w}$.

The L2 loss is as follows:

$$L_2(Y, \hat{Y}) = \frac{1}{2} \sum_{h,w} ||Y_{h,w} - \hat{Y}_{h,w}||_2^2$$

Our regression model consists of two main parts: the encoder and the decoder. In the encoder portion, the input is the black and white image and the model downsizes its size through convolutional blocks to learn its features. The decoder portion then upscales these features through upsampling and convolutional blocks in order to learn the colorization of the photo. The encoder portion of our model is the first 5 layers of Resnet-18 [5], and the decoder portion comprises of 5 convolutional blocks.

3.3. Classification Model

Instead of mapping continuous inputs to continuous output values, we can instead map the image to discrete values, making colorization a classification task. To do this, we quantize the ab color space into Q color bins. The model then outputs the probability of each pixel being a certain quanitized color. Each pixel for each color can have a value between 0 and 1. We define the ground truth color distribution as $Z \in [0, 1]^{h \times w \times q}$ and the predicted color distribution as $\hat{Z} \in [0, 1]^{h \times w \times q}$.

In our experiments, we follow the same quantization proposed by Zhang et al. [14] We divide the ab color spectrum into bins of size 10. In gamut or the visible human spectrum, this leaves $Q = 313$ color bins. In addition, instead of having a one hot ground truth, we calculate a soft encoding so the model can learn the proximity of similar colors. To do this we calculate the 10 nearest neighbors to the ground truth bin and weight using the Gaussian distance with $\sigma = 5$.

We then use cross entropy loss to train the model:

$$L_{cl}(\hat{Z}, Z) = -\sum_{h,w} \sum_{q} Z_{h,w,q} \log(\hat{Z}_{h,w,q})$$

To get a predicted color $\hat{Y}$, we then calculate the mode of the distribution by taking the argmax of $\hat{Z}$, which leaves us with $\hat{Y} \in Q^{h \times w}$.

Following Zhang et al, our classification model similarly uses convolutional blocks to encode and then decode the input. In particular, we use 4 convolutional blocks to first downsize the size of the inputs with the goal of encoding features. The second half the model uses 2 convolutional blocks and upsampling to increase the height and width of our model and add color. The exact dimensions of the convolutions are shown in Figure 1.

3.4. Classification Task Extentions

Class Rebalancing Pixels tend to have dull colors with ab values closer to 0, which will guide a model to predict dull colors. To counteract this, we introduce class rebalancing as proposed by Zhang et al. [14] During training, we weight the loss of each pixel based on the rarity of the quantized color bin. To calculate the $w(Z_{h,w})$, we first calculate p_q the probability of each color bin over the entire training dataset and smooth with a Gaussian of $\sigma = 5$. We also incorporate a uniform distribution of $\lambda = \frac{1}{2}$ which prevents the class weights from being too small, slowing down training. During prediction time, we take the argmax for each pixel $q = \text{argmax}(Z_{h,w})$. We then take the combination of the Gaussian and the uniform, normalize, then take the reciprocal leaving us with:

$$w_q = \left((1-\lambda)p_q + \frac{\lambda}{Q}\right)^{-1}$$

We normalize the sum w to equal 1:

$$\mathbb{E}[w] = \sum_q p_q w_q = 1$$

Adding the new weights, this changes the cross entropy loss to:

$$L_{cl,w}(\hat{Z}, Z) = -\sum_{h,w} w(Z_{h,w}) \sum_q Z_{h,w,q} \log(\hat{Z}_{h,w,q})$$

This means that the loss of pixels with less common colors will have higher loss, encouraging the model to predict brighter, more rare colors.

U-Net Classification Model To improve on the classification model, we draw inspiration from U-Net, which is a convolutional network architecture for fast and precise segmentation of images that was introduced for developed initially for segmenting biomedical data [12]. To do this, we modify our original classification model with the addition of skip-connections. These skip-connections are used when upsampling the model, by summing higher resolution feature maps from the encoder portion of the model with the upsampled features. Because upsampling is a sparse operation, using these good priors from earlier in the model gives the model more precise information.

Our U-Net model consists of 8 convolutional blocks. We trained the U-Net with skip connections that added the features of block 3 to block 6, block 2 to 7, and block 1 to 8.

4. Dataset and Features

In this paper we used the Tiny Imagenet dataset and the MIT Places dataset.

4.1. Tiny Imagenet

Stanford's Tiny Imagenet contains 64×64 images of 200 classes [10]. The training set comprises of 100,000 images, the validation set comprises 10,000 images, and the test set comprises of 10,000 images.

Our models trained on Tiny Imagenet overfit very quickly due to the small training set, as we will discuss further in our results section. This led us to also include our next dataset MIT Places.

4.2. MIT Places

The MIT-Places dataset [16] contains 1.8 million training images, 36,500 validation images, and 328,500 test images covering 365 classes. Each class contains between 3,068 and 5,000 images in the training set, 900 images in the test set, and 100 images in the validation set. Each image is has dimensions 256×256. The Places dataset includes outdoor landscapes that are inherently colorful and have recurring features like green grass and blue skies, as well as indoor scenes.

4.3. Pre-processing

To pre-process our data, we split each image in to the L and AB channels. To normalize our data, for the L channel we divided by 100 to get the input in the range [0,1] and for the A and B channels we divided by 128 to get into the [-1,1] range.

In addition, during training we performed random horizontal flip transformations with probability 0.5. This helps increase generalizability and essentially mimics adding new images to the dataset. For consistency and due to memory and time constraints, we also resize all input images to have dimensions 64×64.

5. Experiments

5.1. Evaluation method

For the baseline regression, the model was trained with L2 loss, and evaluated on both L2 and L1 loss.

For our classifiers, we trained on cross entropy loss and evaluated using cross entropy, L2, and L1 loss, and accuracy

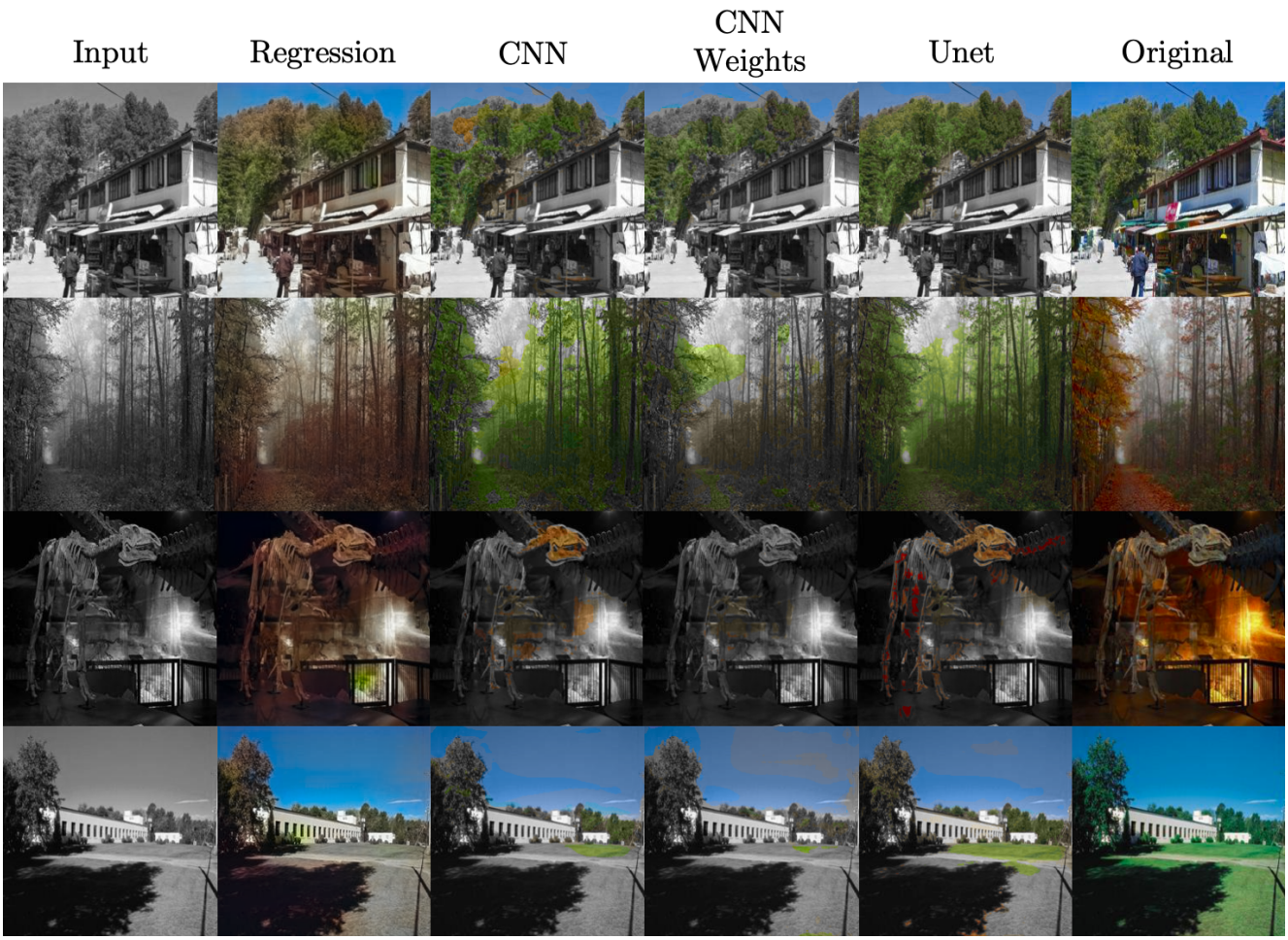


Figure 2. Results on MIT Places Dataset

on the validation set. While the classifier model is trained with cross entropy loss, we also wanted to see how L2 and L1 loss are affected as the model learns. Accuracy refers to the average accuracy of each pixel color classification. We also saved validation set predicted image every 50,000 steps to verify the improvement of the model.

In addition, we qualitatively evaluated the performance of the model by randomly selecting images and visually comparing them to the original color image, as we found that quantitative loss metrics did not always accurately capture model performance.

5.2. Experimental Details

In all of our experiments, we used the Adam optimizer with parameters $\beta 1 = 0.9$, $\beta 2 = 0.99$, weight decay $w = 0.001$, and a learning rate of 3e-5. We also clip the max gradient to 5 to prevent gradient explosion.

We ran experiments on AWS EC2 Deep Learning AMI (Ubuntu 18.04) Version 50.0 with instance type g4dn.xlarge. We also ran experiments on Google Cloud Platform Deep Learning VM with four NVIDIA Tesla T4 GPUs. Because we used varying machines, we will not draw strong conclusions about training time but will instead focus on the results.

We performed the following 5 experiments on the Tiny-Imagenet and MIT Places datasets:

1. Baseline Regression
2. CNN Classifier
3. CNN Classifier with Class weights
4. Unet Classifier
5. Unet Classifier with Class weights

6. Results

6.1. Quantitative Analysis

While it is difficult to draw conclusions from quantitative metrics for image colorization, they do help summarize

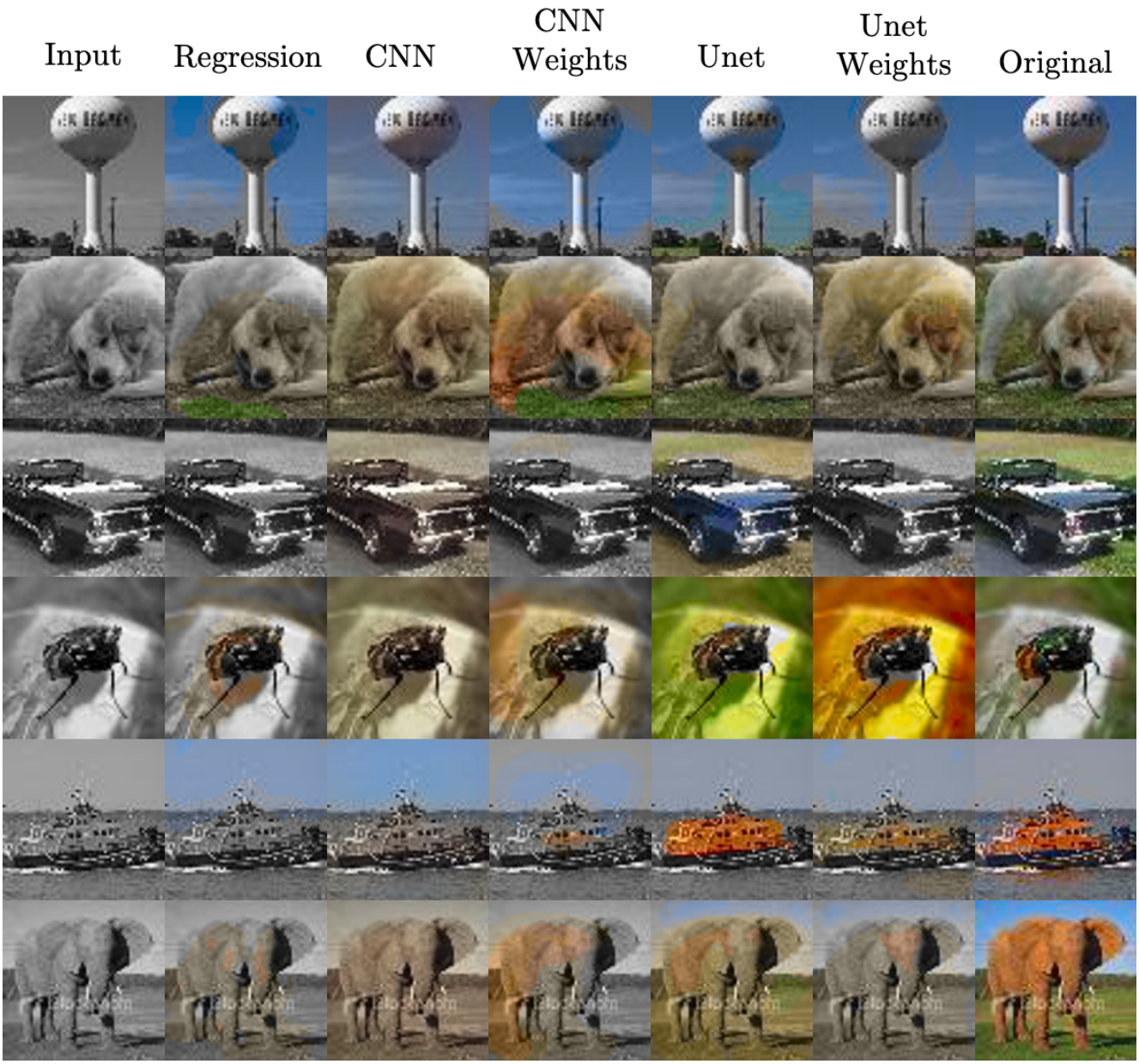


Figure 3. Results on Tiny ImageNet Dataset

macro trends in the data. In Table 1, we see that regression achieved the best L1 and L2 losses because they are not constrained by quantized color bins. In addition, the cross entropy loss increased for models with weights, but that was caused by the class rebalancing weights pushing the model to predict more rare colors. However, for L1, L2, and accuracy there was negligible difference with the addition of class weights.

Overfitting As seen in the training curve figures in the Appendix, we observed strong overfitting while training, especially when training TinyImagenet on the CNN. At 500k steps the validation loss has already been minimized, while the training loss was continuing to decrease. However, while the validation loss began to grow, the colorization performance still improved visually and we saw that the L1 validation loss continued to decrease. Observing overfitting led us to experiment with the U-Net and adding class weights, both of which reduced overfitting by introducing more complexity to the classification task. Additionally, we decided to train on the MIT Places dataset, which has 18 times as many images. Our runs on MIT Places did not overfit because of the sheer volume of data.

6.2. Qualitative Analysis

Some examples from our results on MIT Places and Tiny Imagenet can be found in Figures 2 and 3. We saved 100 images from the test set of each dataset and chose to display a variety of the most interesting results over different objects and places.

The regression model trained on MIT Places produced the smoothest images but also generally the least colorful. It is apparent that it took advantage of the larger number of training examples compared to tiny image which can be seen by the worse performance on the Tiny Imagenet Regression model.

Unlike Zhang et al [14], our class rebalancing weights in general did not help make the images brighter and more colorful. There are exceptions like the fly in Figure 3 which is on a bright orange background.

The U-Net classifier for both datasets seem to outperform all other models, meaning that the skip connections were helpful in accelerating learning by improving upsampling and preventing neurons from dying.

Robustness While a model may generalize well to data in the validation or test set, it is also important that it generalizes to other datasets and modes. There is overlap between datasets; both contain landscapes and objects but Places leans towards landscapes and TinyImagenet leans more towards objects. We evaluate our models trained on TinyImagenet on MIT Places and vice versa. Our results shows that the model trained on TinyImagenet can better generalize to MIT Places, both visually in Figure 6 and quantitatively in Table 1. We reason that because MIT Places contains many outdoor scenes, the model gained strong confidence in coloring green grass and blue sky, while leaving the remainder of the image grey. More specifically, the U-Net with weights trained on TinyImagenet generalized best onto the MIT Places dataset, maintaining accurate and vibrant colorizations.

Figure 4. Bad colorization from regression and classification models trained on MIT Places

Error Analysis Manual inspection of the model predictions yielded insights on what each model struggled on. Figure 4 shows a sampling of failed colorizations. We can see that the regression model tends to color everything desaturated and brown as a way of directly minimizing mean-squared error due to the ubiquitous and uncontroversial nature of the color. On the other hand, the classification model

	Trained on Tiny ImageNet Evaluated on Tiny ImageNet				Trained on MIT Places Evaluated on MIT Places			
Method	L2	L1	Cross Entropy	Accuracy	L2	L1	Cross Entropy	Accuracy
Baseline Regression	0.013	0.076	-	-	0.011	0.067	-	-
CNN	0.016	0.081	3.239	0.280	0.011	0.066	2.917	0.333
CNN with Weights	0.018	0.089	12.019	0.237	0.011	0.067	14.414	0.328
Unet	0.015	0.080	3.647	0.252	0.013	0.075	2.949	0.318
Unet with Weights	0.015	0.080	17.019	0.277	0.014	0.075	9.086	0.301
	Trained on Tiny ImageNet Evaluated on MIT Places				Trained on MIT Places Evaluated on Tiny ImageNet			
Method	L2	L1	Cross Entropy	Accuracy	L2	L1	Cross Entropy	Accuracy
Baseline Regression	0.011	0.074	-	-	0.011	0.074	-	-
CNN	0.013	0.072	3.195	0.299	0.017	0.085	3.669	0.281
CNN with Weights	0.020	0.096	8.892	0.219	0.017	0.085	25.243	0.281
Unet	0.013	0.076	3.559	0.249	0.017	0.085	3.415	0.280
Unet with Weights	0.013	0.076	12.681	0.253	0.017	0.085	15.383	0.277

Table 1. L2, L1, Cross Entropy and Accuracy Values evaluated on the Tiny ImageNet and MIT Places Test Set. The top table is evaluating on the same dataset as the model was trained, and the bottom graph is a robustness test, evaluating on a different dataset than the model was trained. (Weights refers to Class Rebalancing Weights mentioned in section 3.4)

Figure 5. Colorized faces on a CNN classifier trained on MIT Places.

trained on MIT Places tended to color mostly green and blue. While the photo on the bottom left of Figure 4 is supposed to be pizza, the model interpreted it to be grass. Similarly, in the bottom right photo, the model assumes that the ceiling of the room is the sky because the model has learned that the top portion of the sky is blue.

Semantic information learned From Figures 2 and 3, we can see that our models have learned semantic information from this unlabeled task of colorization. The models have learned that trees and grass are green and can distinguish background from buildings and objects. From Figure 5, we can see that they have even learned to recognizes human faces. However, some objects like the pizza in 4 have not been learned. We believe that with a larger dataset, our models could learn more objects and places.

7. Conclusion

To summarize, in this project we investigate the problem of colorization through several lens. We begun with a baseline regression model and then explored defining colorization as a classification task instead. In doing so, we experiment with class weights to encourage the model to pick more vibrant and rare colors. However, in practice, its effects became either too vibrant or not noticeable. Additionally, we used skip-connections in the model to help the model upsampling. We applied each of our models on two datasets, Tiny ImageNet and MIT Places, and evaluated the predictions rigorously through both quantitative and qualitative methods. We conclude that the U-Net without class weights performs the best.

While we saw promising results in our models, there are certainly room for improvements. One future addition could be removing gray and very dull images from the training set, as mapping a black and white input to a black and white output does not help the model learn color. Another consideration we will have for future projects is the importance of training on a large amount of data. In future experiments we could train on the MIT Places365-Challenge 2016 dataset [16] which contains 8 million training images.

Figure 6. Robustness: TinyImage Test Images colorized by a Unet Classifier trained on MIT Places

In addition, we could train on an ensemble of datasets to improve robustness. Better preprocessing and image transforms could also help with overfitting by bootstrapping new training examples. Another idea we have is to pretrain sections of our model on a semantic task like classification or segmentation. We believe this could speed up training and even boost the semantic knowledge of the final model. Another way we could evaluate our colorization models is observing the semantic information learned in our models. To do this we could use the autoencodings generated from our models to train a classifier.

7.1. Ethics

While image colorization may seem harmless, it actually creeps on some ethical gray areas. For example, the colorization of historical black and white images and videos may corrupt the artistic value of the original form; colorizing can go against the wishes of the original artist. For example, movies by the the famous Japanese filmmaker Akira Kurosawa have a distinct style, with its black and white color forming the heart of his art. Many Kurosawa fans would find it would be disrespectful to colorize his movies. In addition, historical images contain real people whose families may not wish their likeness altered. In 2021, Irish artist Matt Loughrey came under fire after colorzing historical photos of Cambodian torture victims [8]. While the colorization was not done using machine learning, increased prevalence and availability of deep learning models like the ones we have created would make it easier for anyone to colorize photos. Like any technology, it is crucial to keep in mind the potentially dangerous applications of one's work.

8. Contributions

8.1. Justin Olah

Justin coded the CNN classifier, class weights, and testing framework. He made the figures, and wrote the related work, methods, analysis, and conclusion.

8.2. Jenny Yang

Jenny coded the Unet classifier, class weights, and training framework. She wrote the abstract, introduction, methods, and experiments sections.

8.3. Codebases

We would like to acknowledge Zhang et al. [14] for providing a starting point with their color quantization code, and HarisIqbal88 [7] for their PlotNeuralNet Github repository for making network diagrams.

References

[1] G. Charpiat, M. Hofmann, and B. Schölkopf. Automatic image colorization via multimodal predictions. In D. Forsyth, P. Torr, and A. Zisserman, editors, *Computer Vision – ECCV 2008*, pages 126–139, Berlin, Heidelberg, 2008. Springer Berlin Heidelberg. 1

[2] Z. Cheng, Q. Yang, and B. Sheng. Deep colorization. *CoRR*, abs/1605.00075, 2016. 1

[3] J. González–Santiago, F. Schenkel, and W. Middelmann. Self-supervised image colorization for semantic segmentation of urban land cover. In *2021 IEEE International Geoscience and Remote Sensing Symposium IGARSS*, pages 3468–3471, 2021. 1

[4] R. K. Gupta, A. Y.-S. Chia, D. Rajan, E. S. Ng, and H. Zhiyong. Image colorization using similar images. In *Proceedings of the 20th ACM International Conference on Multimedia*, MM '12, page 369–378, New York, NY, USA, 2012. Association for Computing Machinery. 1

[5] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015. 2

[6] Y.-C. Huang, Y.-S. Tung, J.-C. Chen, S.-W. Wang, and J.-L. Wu. An adaptive edge detection based colorization algorithm and its applications. In *Proceedings of the 13th Annual ACM International Conference on Multimedia*, MULTIMEDIA '05, page 351–354, New York, NY, USA, 2005. Association for Computing Machinery. 1

[7] H. Iqbal. Harisiqbal88/plotneuralnet v1.0.0, Dec. 2018. 8

[8] S. LaBarre. The controversial history of colorizing black-and-white photographs, May 2021. 7

[9] G. Larsson, M. Maire, and G. Shakhnarovich. Learning representations for automatic colorization. *CoRR*, abs/1603.06668, 2016. 1

[10] Y. Le and X. S. Yang. Tiny imagenet visual recognition challenge. 2015. 3

[11] K. Nazeri and E. Ng. Image colorization with generative adversarial networks. *CoRR*, abs/1803.05400, 2018. 2

[12] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. *CoRR*, abs/1505.04597, 2015. 3

[13] S. Tsaftaris, F. Casadio, J.-L. Andral, and A. Katsaggelos. A novel visualization tool for art history and conservation: Automated colorization of black and white archival photographs of works of art. *Studies in Conservation*, 59:125–135, 05 2014. 1

[14] R. Zhang, P. Isola, and A. A. Efros. Colorful image colorization. In *ECCV*, 2016. 1, 2, 3, 6, 8

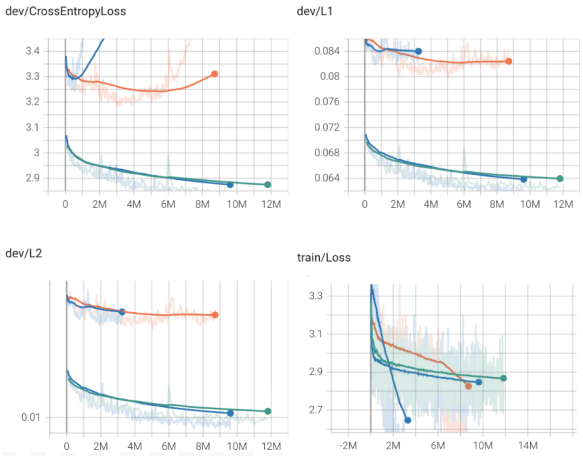
[15] R. Zhang, J. Zhu, P. Isola, X. Geng, A. S. Lin, T. Yu, and A. A. Efros. Real-time user-guided image colorization with learned deep priors. *CoRR*, abs/1705.02999, 2017. 1

[16] B. Zhou, A. Lapedriza, A. Khosla, A. Oliva, and A. Torralba. Places: A 10 million image database for scene recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2017. 3, 7

[17] J. Zhu, T. Park, P. Isola, and A. A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. *CoRR*, abs/1703.10593, 2017. 2

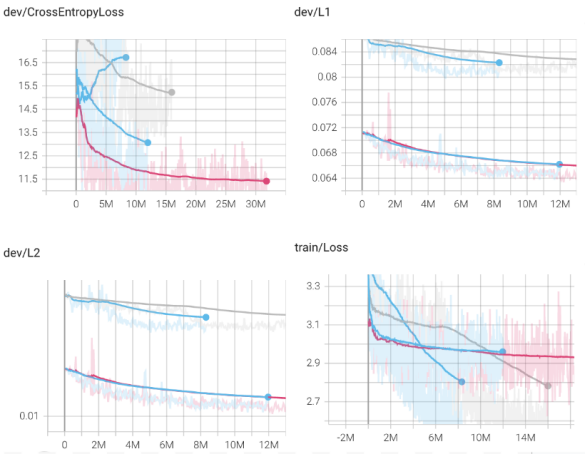
9. Appendix

We have also included the training curves for our CNN and Unet classifiers without class rebalancing weights:



Upper blue: CNN on Tiny Imagenet, Orange: Unet on Tiny Imagenet, Lower blue: Unet on MIT Places, Green: CNN on MIT Places.

Below is the training curves for CNN and U-Net classifiers with class rebalancing weights:



Upper blue: Unet on Tiny Imagenet, Grey: CNN on Tiny Imagenet, Lower blue: CNN on MIT Places, Unet on MIT Places.