Towards More Vibrant Image Colorization

Ahan Mukhopadhyay* **Kolin Guo*** **Kyle Lisenbee*** **Ulyana Tkachenko***
{amukhopa,ruguo,klisenbe,utkachen}@ucsd.edu

1 Introduction

The task of colorizing a grayscale image is both an unconstrained optimization problem over the human comprehension and an optimization problem over the images with potentially unknown ground truth. On the surface, hallucinating a plausible colored version of a grayscale image seems daunting, since two of the three dimensions of the image data have been lost. However, if done accurately and efficiently, such a task can have a wide range of applications in computer vision and graphics (*e.g.*, object detection and classification in grayscale images). Moreover, it can be used as a powerful pretext task for self-supervised feature learning (an autoencoder that outputs two color channels given one lightness channel input).

Related Works: The colorization of grayscale images has been a popular direction for researchers recently thanks to the development of deep convolution neural networks in computer vision. Instead of directly learning in the RGB color space, many approaches chose to learn in other color space: Dahl et al. utilizing VGG16 and VGG19 ImageNet pre-trained layers in the CIELUV color space [1, 2], Deshpande et al. transforms RGB color space to a de-correlated 2-channel normalized opponent color space [3], Zhang et al. implements a multinomial classification CNN network in the CIE *Lab* color space [4]. Others have developed instance-aware colorization approaches where the grayscale images are first passed through an object segmentation network and then a colorization network based on the segmentation map [5] for more coherent colorization within object instances. These instance-aware approaches have been successfully applied to image-to-image translation [6, 7].

Intended Contributions: For our project, we build on the CNN network shown in Figure 1 proposed in [4] by introducing an ℓ_2 norm constraint to encourage more vibrant colorful images. We also formulate the primal/dual problem and KKT conditions of the inherently non-convex optimization problem of the CNN network. To simplify our formulation, we focused on a single convolution block with two convolution-ReLU layers using gradients of the convolution layers as abstract functions. Also, we acknowledged the recent advancement in formulating a two-layer CNN as a convex optimization problem that can be solved in polynomial time [8] and also included the primal/dual/KKT using its results. From our experiments, the CNN network trained with our loss function can improve the washed-out results compared to the standard ℓ_2 norm loss in [4].

Task Assignment: **Ahan Mukhopadhyay** and **Ulyana Tkachenko** work on the mathematical formulation and deduction of the primal/dual problem and the KKT conditions of a single convolution block. **Kyle Lisenbee** works on code implementation and experiments on the Coco 2017 dataset. **Kolin Guo** works on code implementation, experiments on the Coco 2017 dataset, and formulates of the primal/dual/KKT using results in [8].

Paper Organization: The statement of the problem with primal/dual/KKT of a single convolution block is formulated in section [2]. The primal/dual/KKT of a two-layer CNN using results in [8] is formulated in Appendix [A.2-A.4]. The experimental approaches and results compared to the standard ℓ_2 norm loss are discussed in sections [3] and [4], respectively.

2 Statement of the Problem

Given an input lightness channel $\mathbf{L} \in \mathbb{R}^{H \times M \times 1}$, our objective is to learn a mapping $\hat{\mathbf{Y}} = \mathcal{F}(\mathbf{L}; \mathbf{\Theta})$ with the CNN network in Figure 1 to the two associated color channels *ab* of the CIE *Lab* color space $\mathbf{Y} \in \mathbb{R}^{H \times W \times 2}$, where H, W are image dimensions and $\mathbf{\Theta}$ is the parameters of the mapping function $\mathcal{F}(\cdot)$. We choose the CIE *Lab* color space because the distances in this space model perceptual color distances of human vision and direct relation to human color perception. In the loss functions below, we use subscript h, w to denote the value for the image pixel at location (h, w) (*e.g.*, $\mathbf{Y}_{h,w} \in \mathbb{R}^2$).

A straightforward objective function, as used in [4, 10], is the Euclidean ℓ_2 loss $J_{\ell_2}(\cdot, \cdot)$ between predicted and groundtruth *ab* colors:

$$J_{\ell_2}(\hat{\mathbf{Y}}, \mathbf{Y}) = \frac{1}{2} \sum_{h,w} \|\mathbf{Y}_{h,w} - \hat{\mathbf{Y}}_{h,w}\|_2^2 \tag{1}$$

*Authors are alphabetically ordered by first names. All authors belong to the Department of Computer Science and Engineering. GitHub repos for our code: https://github.com/KolinGuo/Colorization and https://github.com/klisenbee/Colorization

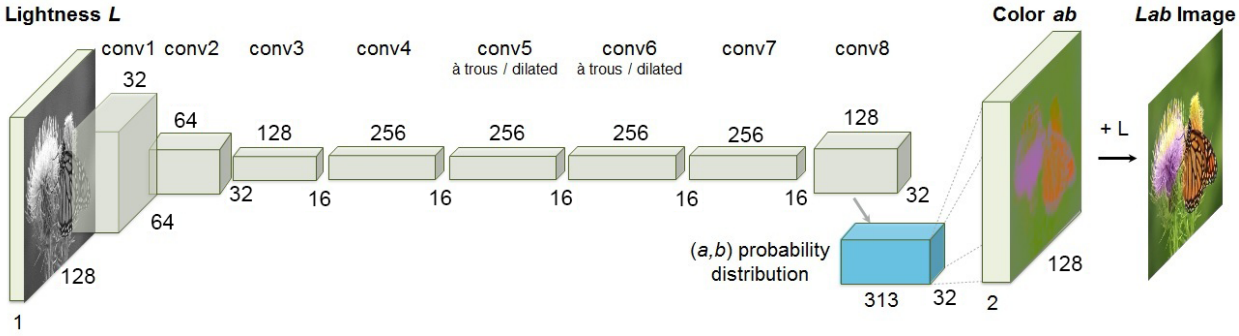

Figure 1: Our network architecture. Each `conv` layer refers to a block of 2 or 3 repeated `conv` and `ReLU` layers, followed by a `BatchNorm` [9] layer. The net has no `pool` layers. This figure is modified from Fig. 2 of [4] by reducing the number of `conv` filters by a half and the input/output images by a half.

However, the Euclidean ℓ_2 loss tends produce a desaturated color [4] because the optimal value to the loss function is the mean of the *ab* values. Since the *ab* values both have range $[-110, 110]$, their means would be $[0, 0]$ which represents a grayish color. To encourage more vibrant color, we incorporate an additional chroma term $\mathbf{C}$ and $\hat{\mathbf{C}}$ from the CIE *Lhc* color space [11] into equation [1] to ensure color vividness closer to ground truth. Our proposed loss function for encouraging color vividness is as follows:

$$J(\hat{\mathbf{Y}}, \mathbf{Y}) = \frac{1}{2} \sum_{h,w} \|\mathbf{Y}_{h,w} - \hat{\mathbf{Y}}_{h,w}\|_2^2 + \gamma \|\mathbf{C}_{h,w} - \hat{\mathbf{C}}_{h,w}\|_2^2 + \nu \|\mathbf{\Theta}\|_2^2 \tag{2}$$

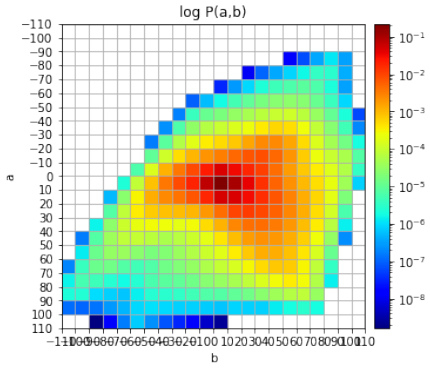

Figure 2: Empirical Probability Distribution in Quantized in-gamut *ab* color space with a grid size of 10. Most pixels have *ab* values around the center $[0, 0]$.

2.1 Primal Formulation of a Single Convolution Block

Using equation [2], we derive our primal formulation in standard form. We can rewrite this formulation with Θ as a constraint instead of directly in the objective function. These two problems are equivalent because their dual problems are equivalent [12]. By Lagrangian duality we can use the following formulation as our primal problem:

$$\begin{aligned} \text{minimize} \quad & \frac{1}{2} \sum_{h,w} \|\mathbf{Y}_{h,w} - \hat{\mathbf{Y}}_{h,w}(\mathbf{L}; \mathbf{\Theta})\|_2^2 + \gamma \sum_{h,w} (\mathbf{C}_{h,w} - \hat{\mathbf{C}}_{h,w})^2 \\ \text{subject to} \quad & \mathbf{C}_{h,w} = \|\mathbf{Y}_{h,w}\|_2 = \sqrt{(\mathbf{Y}_{h,w})^T \mathbf{Y}_{h,w}}, \quad \hat{\mathbf{C}}_{h,w} = \|\hat{\mathbf{Y}}_{h,w}\|_2 = \sqrt{(\hat{\mathbf{Y}}_{h,w})^T \hat{\mathbf{Y}}_{h,w}} \\ & \|\mathbf{\Theta}\|_2^2 \leq t, \qquad \gamma \geq 0 \end{aligned}$$

2.2 Subgradients of the CNN

We consider a small block of our CNN consisting of two convolution layers with intermediate ReLU layers. For specifics on how these layers are defined please refer to Appendix [A.5]. To calculate the partial gradients necessary for getting the dual, we first find the gradient of our proposed loss [2] with respect to the network output.

$$\frac{\partial J}{\partial \hat{Y}} = \hat{\mathbf{Y}}(1 + 2\gamma - \frac{2\gamma \|\mathbf{Y}\|_2}{\|\hat{\mathbf{Y}}\|_2}) - \mathbf{Y} \tag{3}$$

The gradient of the loss with respect to the network weights K_1 and K_2 is calculated using the chain rule and shown below with δ_{K_1}, δ_{K_2}, α_{K_1}, and α_{K_2} representing abstractions in the convolution layer gradients.

$$\frac{\partial J}{\partial K_1} = (\hat{\mathbf{Y}}(1+2\gamma - \frac{2\gamma\|\mathbf{Y}\|_2}{\|\hat{\mathbf{Y}}\|_2}) - \mathbf{Y}) \cdot \text{sgn}(\text{ReLU}(\theta))\ \delta_{K_1} \cdot \text{sgn}(\text{ReLU}(h)) \cdot \alpha_{K_1} \tag{4}$$

$$\frac{\partial J}{\partial K_2} = (\hat{\mathbf{Y}}(1+2\gamma - \frac{2\gamma\|\mathbf{Y}\|_2}{\|\hat{\mathbf{Y}}\|_2}) - \mathbf{Y}) \cdot \text{sgn}(\text{ReLU}(\theta)) \cdot \alpha_{K_2} \tag{5}$$

2.3 KKT conditions and Dual Problem of a Single Convolution Block

First we construct the Lagrangian of the objective function $\mathbf{J}$. Note that $\mathbf{\Theta}$ in [2] is all the network parameters which is just the combination of $\mathbf{K_1}$ and $\mathbf{K_2}$ for our CNN block.

$$\begin{aligned}\mathbf{L_g}(\mathbf{K_1}, \mathbf{K_2}, \lambda_1, \lambda_2) = \frac{1}{2}\sum_{h,w}\|\mathbf{Y}_{h,w} - \hat{\mathbf{Y}}_{h,w}\|_2^2 \ &+\ \gamma\sum_{h,w}(\|\mathbf{Y}_{h,w}\|_2 - \|\hat{\mathbf{Y}}_{h,w}\|_2)^2 \\ &+\ \lambda_1(-\gamma) + \lambda_2(\|\mathbf{K_1}\|^2 + \|\mathbf{K_2}\|^2 - t)\end{aligned} \tag{6}$$

Assuming the Lagrangian is differentiable, the dual can be derived by solving for optimal values of K1 and K2. These steps also apply for simplifying the stationary KKT conditions [2].

$$\nabla_{\mathbf{K_1}}\mathbf{L_g}(K_1, K_2, \lambda_1, \lambda_2) = \frac{\partial J}{\partial K_1} + 2\lambda_2\mathbf{K_1} = 0,\ \mathbf{K_1} = \mathbf{K_1^*}$$

$$\nabla_{\mathbf{K_2}}\mathbf{L_g}(K_1, K_2, \lambda_1, \lambda_2) = \frac{\partial J}{\partial K_2} + 2\lambda_2\mathbf{K_2} = 0,\ \mathbf{K_2} = \mathbf{K_2^*}$$

Since our problem is non-convex due to the complexity of our CNN network, any $\mathbf{K_1}^*$ and $\mathbf{K_2}^*$ that are primal and dual optimal with zero duality gap will satisfy the KKT conditions above. However, we cannot claim vice versa: any $\mathbf{K_1}^*$ and $\mathbf{K_2}^*$ that satisfy the KKT conditions above will be primal and dual optimal with zero duality gap.
The dual problem solution is unconstrained due to the level of abstraction on our CNN gradients.

$$\mathbf{g}^*(\lambda_1, \lambda_2) = \max_{\lambda_1, \lambda_2} \mathbf{L_g}(\mathbf{K_1^*}, \mathbf{K_2^*}, \lambda_1, \lambda_2)$$

A more detailed primal/dual/KKT conditions analysis of a two-layer CNN can be found in A.2-A.4 using results in [8]

3 Experimental Approach

The goal of our approach is an optimization method over the colorization of an image such that we maximize the retention of its likely true colorfulness. Given the nature and complexity of this task, we base our method on the success of Convolutional neural networks in image orientated tasks.

3.1 Network Architecture

We adapt the overall network structure of [4] with a restructuring of the dimensions at each layer, where each image input is reduced in dimension by half. To maintain relative complexity, the number of nodes in each network layer are reduced by one half as well. We retain the networks number of convolution and activation layers to promote the necessary generalization of pixel-wise relationships across training images. Change of dimension between layers is through stride, kernel dimensions remain static.
Each network input is the corresponding lightness channel of an image in the LAB colorspace and each output is a predicted coloring of the a and b channels. As the lightness channel remains fixed, our loss is restricted to two channels. We treat the weight of the chroma loss over these two channels as a network parameter. Network weights are trained using Adam over mini-batches.

3.2 Evaluation

To evaluate the success of our model, we utilize the mathematical metrics area under the curve (AuC), rebalanced area under the curve (RebalancedAuC), peak signal-to-noise ratio (PSNR) and root mean squared error (RMSE).

AuC: We compute the percentage of predicted pixel colors within a thresholded L2 distance of the ground truth in *ab* color space. We then sweep across thresholds from 0 to 150 to produce a cumulative mass function, as introduced in [3], integrate the area under the curve, and normalize.

RebalancedAuC: However, we observed that the raw AuC metric is not representative of the network performance due to the class imbalance in the *ab* color space. Therefore, to best evaluate the model's performance, we re-weights the pixels inversely by the class probability in *ab* space shown in Figure 2.

[2] A full description of KKT conditions for the primal problem can be found in Appendix [A.6]

MSE: We use two mean squared error measures to evaluate the models from differing perspectives. RMSE gives us an idea of the models performance on a strict pixel by pixel basis. However, this measure struggles when images lack sufficient brightness, or when measuring images of different dimensions. To handle this, we use the Peak signal-to-Noise ratio to better scale the calculated metric over a diverse data set .

While the metrics above help to describe the relative true accuracy between an image and its predicted coloring, they do not provide an accurate measure of coloring quality. The same measured inaccuracy in a color space can as equally be caused by a change in color as it can by a lack of colorfulness. Subsequently, our evaluation lacks a strong correlation between subjective and objective measure. For our purposes, we note the mathematical metrics for comparison, but focus on perceived quality.

4 Results

Figure 3: Our model trained over 20 epochs on the COCO data set. Left is our input image and right the ground truth with increasing gamma respective of Table 1.

The evaluation results using the metrics above is summarized in Table 1. Training on the COCO image data set with over 118,000 images, we find our model increases the subjective colorfulness of the input image with respect to a base model of L_2 in equation [1]. With increasing gamma, the generated images appear to be colorful on average.

Table 1: Evaluation Metrics Results on Coco 2017 Dataset

Model/Metric	AuC	PSNR	RMSE	Model/Metric	RebalancedAuC
Pretrained	0.892	22.3	0.26		
L_2,$\gamma = 0$	0.91	23.4	24	L_2,$\gamma = 0$	0.3544
L_2,$\gamma = 0.25$	0.91	23.5	24		
L_2,$\gamma = 0.50$	0.9	23.2	0.25	L_2,$\gamma = 1$	0.3967
L_2,$\gamma = 1$	0.89	22.8	0.23	L_2,$\gamma = 2$	0.3669
Random	0.23	24.1	0.28		

By the results in Table 1, however, we can see the difficulty in quantifying our perception. Change in chroma is not clearly represented by a positive change across evaluation metrics. We can see this in comparison to the model described in [4]. While their model outperforms ours in a subjective perception test, our models score better across RMSE and AuC. All tested models outperformed a random colorization, however.

5 Conclusion

We developed a novel loss function [2] for a feed-forward CNN that ensured visually more vivid image colorization than comparable Euclidean loss methods. We rewrote this new objective into a standard primal form and derived an abstract dual formulation for a layer block specific to our CNN model [A.5] and novel loss function. Following, we discussed a closed form solution to the primal-dual formulation and KKT conditions of the strictly convex Euclidean loss in relation to our image colorization problem [A.2, A.3, A.4]. Future work to derive a similar closed form analysis specific to our

novel loss function will provide more insight into the relationship between loss convexity and our added constraint. Successive research into representing larger CNNs as closed form convex optimization problems is necessary for a complete reformulation of image colorization as a single convex optimization problem.

References

[1] R. Dahl, "Automatic colorization," 2016, https://tinyclouds.org/colorize.

[2] M. He, D. Chen, J. Liao, P. V. Sander, and L. Yuan, "Deep exemplar-based colorization," 2018.

[3] A. Deshpande, J. Rock, and D. Forsyth, "Learning large-scale automatic image colorization," in *2015 IEEE International Conference on Computer Vision (ICCV)*, 2015, pp. 567–575.

[4] R. Zhang, P. Isola, and A. A. Efros, "Colorful image colorization," in *ECCV*, 2016.

[5] J.-W. Su, H.-K. Chu, and J.-B. Huang, "Instance-aware image colorization," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[6] S. Ma, J. Fu, C. W. Chen, and T. Mei, "Da-gan: Instance-level image translation by deep attention generative adversarial networks (with supplementary materials)," 2018.

[7] Z. Shen, M. Huang, J. Shi, and X. Xue, "Towards instance-level image-to-image translation," 06 2019, pp. 3678–3687.

[8] T. Ergen and M. Pilanci, "Implicit convex regularizers of {cnn} architectures: Convex optimization of two- and three-layer networks in polynomial time," in *International Conference on Learning Representations*, 2021.

[9] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *CoRR*, vol. abs/1502.03167, 2015.

[10] Z. Cheng, Q. Yang, and B. Sheng, "Deep colorization," in *2015 IEEE International Conference on Computer Vision (ICCV)*, 2015, pp. 415–423.

[11] A. Ford, A. Roberts, and C. Poynton, "Colour space conversions," pp. 8–10, 1998.

[12] B. R. Gaines, J. Kim, and H. Zhou, "Algorithms for fitting the constrained lasso," *Journal of Computational and Graphical Statistics*, vol. 27, no. 4, pp. 861–871, 2018, pMID: 30618485.

[13] K. Clark, "Computing neural network gradients, stanford cs224n," 2019, https://web.stanford.edu/class/cs224n/readings/gradient-notes.pdf.

[14] A. Sahiner, T. Ergen, J. M. Pauly, and M. Pilanci, "Vector-output re{lu} neural network problems are copositive programs: Convex analysis of two layer networks and polynomial-time algorithms," in *International Conference on Learning Representations*, 2021.

Appendix

A Appendix

Table of Contents

A.1 Notation

Table 2: Notation

Notation	Description
$\mathbf{L}$	Lightness channel for input image
h, w	Pixel coordinates in the input image
$\mathbf{Y}_{h,w}$	Ground truth ab channel values
$\hat{\mathbf{Y}}_{h,w}$	Predicted ab channel values
$\mathbf{C}_{h,w}$	Ground truth chroma values
$\hat{\mathbf{C}}_{h,w}$	Predicted chroma values
$\mathbf{\Theta}$	CNN model parameters
γ	Weight for chroma L2 term

A.2 Primal Formulation of a two-layer CNN

Here, we assume that our model is a two-layer CNN: inputs are feed into a convolution layer with ReLU activation and then to a fully-connected layer with output dimensions. Also, since the loss function in equation [2] is non-convex (we cannot prove its convexity), we will focus on the Euclidean ℓ_2 loss function in equation [1] instead.

Notation and preliminaries: We denote the input lightness channel and the corresponding ab channels as $\mathbf{L} \in \mathbb{R}^{n \times d_{in}}$ and $\mathbf{Y} \in \mathbb{R}^{n \times d_{out}}$, where n is the number of training images, $d_{in} = (H_{in} \times W_{in})$ is the input dimensions and $d_{out} = (H_{out} \times W_{out} \times 2)$ is the output dimensions. Moreover, we denote the convolutional patch matrices, *i.e.*, subsets of input dimensions, extracted from $\mathbf{L}$ as $\mathbf{L}_k \in \mathbb{R}^{n \times h}$, $k \in [K]$, where h denotes the convolution kernel filter size, $[K]$ is the set of integers from 1 to K and K is the convolution layer output dimensions $K = (H_c, W_c)$. With this notation, $\{\mathbf{L}_k \mathbf{u}\}_{k=1}^{K}$ denotes a convolution operation between a filter $\mathbf{u} \in \mathbf{R}^h$ and the input data $\mathbf{L}$. In this section, the ReLU activation function is denoted as $(x)_+ = \max\{0, x\}$.

Using the result in Appendix A.5 of [8], the training problem of equation [1] with network parameters is as follows

$$p^* = \min_{\{\mathbf{u}_j, \{\mathbf{w}_{jk}\}_{k=1}^{K}\}_{j=1}^{m}} \frac{1}{2} \left\| \sum_{k=1}^{K} \sum_{j=1}^{m} (\mathbf{L}_k \mathbf{u}_j)_+ \mathbf{w}_{jk}^\top - \mathbf{Y} \right\|_F^2 + \frac{\beta}{2} \sum_{j=1}^{m} \left(\|\mathbf{u}_j\|_2^2 + \sum_{k=1}^{K} \|\mathbf{w}_{jk}\|_2^2 \right),$$

where $\| \cdot \|_F$ is the Frobenius norm, $\mathbf{w}_{jk} \in \mathbb{R}^{d_{out}}$ is the weights of the fully-connected layer, m is the number of convolution filters. Using a rescaling (as in Lemma A.1 of [8]) of $\bar{\mathbf{u}}_j = \gamma_j \mathbf{u}_j$ and $\bar{\mathbf{w}}_{jk} = \mathbf{w}_{jk} / \gamma_j$ to the primal problem, we obtain an equivalent primal formulation as follows

$$p^* = \min_{\substack{\{\mathbf{u}_j, \{\mathbf{w}_{jk}\}_{k=1}^{K}\}_{j=1}^{m} \\ \mathbf{u}_j \in \mathcal{B}_2}} \frac{1}{2} \left\| \sum_{k=1}^{K} \sum_{j=1}^{m} (\mathbf{L}_k \mathbf{u}_j)_+ \mathbf{w}_{jk}^\top - \mathbf{Y} \right\|_F^2 + \beta \sum_{j=1}^{m} \sum_{k=1}^{K} \|\mathbf{w}_{jk}\|_2, \tag{7}$$

where $\mathcal{B}_2 := \{\mathbf{u}_j \in \mathbb{C}^h : \|\mathbf{u}_j\| \leq 1\}$ is the unit ℓ_2 ball.

A.3 Dual Formulation of a two-layer CNN

The corresponding dual problem of equation [7] is given by

$$p^* \geq d^* = \max_{\mathbf{V}} -\frac{1}{2}\|\mathbf{V}-\mathbf{Y}\|_F^2 + \frac{1}{2}\|\mathbf{Y}\|_F^2 \quad \text{s.t.} \max_{\mathbf{u}\in\mathcal{B}_2} \sqrt{\sum_{k=1}^{K} \|\mathbf{V}^T(\mathbf{L}_k\mathbf{u})_+\|_2^2} \leq \beta. \tag{8}$$

The derivation of this dual formulation is similar to Problem 5 on the midterm.

A.4 KKT conditions of a two-layer CNN

The dual formulation in equation [8] is usually a weak dual. However, Theorem 2.1 of [8] shows that strong duality holds when the number of convolution filters m exceeds a threshold m^*. Due to the time constraint of this project, instead of deriving the strong duality case for equation [7], we here include the equivalent convex program for a two-layer CNN with average pooling and scalar output $\mathbf{y} \in \mathbb{R}^n$ and then formulate its KKT conditions.

$$\begin{aligned} &\min_{\substack{\{\mathbf{c}_i,\mathbf{c}_i'\}_{i=1}^{P_{conv}} \\ \mathbf{c}_i,\mathbf{c}_i'\in\mathbb{R}^h,\forall i}} f(\mathbf{c}_i,\mathbf{c}_i') = \min_{\substack{\{\mathbf{c}_i,\mathbf{c}_i'\}_{i=1}^{P_{conv}} \\ \mathbf{c}_i,\mathbf{c}_i'\in\mathbb{R}^h,\forall i}} \frac{1}{2}\left\|\sum_{i=1}^{P_{conv}}\sum_{k=1}^{K}\mathbf{D}(S_i^k)\mathbf{L}_k(\mathbf{c}_i'-\mathbf{c}_i)-\mathbf{y}\right\|_2^2 + \beta\sum_{i=1}^{P_{conv}}(\|\mathbf{c}_i\|_2+\|\mathbf{c}_i'\|_2) \\ &\text{s.t. } (2\mathbf{D}(S_i^k)-\mathbf{I}_n)\mathbf{L}_k\mathbf{c}_i \succeq 0, \quad (2\mathbf{D}(S_i^k)-\mathbf{I}_n)\mathbf{L}_k\mathbf{c}_i' \succeq 0, \quad \forall i,k, \end{aligned} \tag{9}$$

where $\mathbf{D}(S) \in \mathbb{R}^{n\times n}$ is a diagonal mask matrix defined as $\mathbf{D}(S)_{ii} := \mathbb{1}[i \in S]$, $\mathbf{I}_n$ is the identity matrix of size n, and P_{conv} is the cardinality of the convolutional hyperplane arrangements for the patch matrices $\{\mathbf{L}_k\}_{k=1}^K$.
Introducing Lagrange multipliers $\boldsymbol{\lambda}_{ik}, \boldsymbol{\lambda}_{ik}' \in \mathbb{R}^n$ for the two inequality constraints, the KKT conditions for equation [9] is as follows

$$\begin{aligned} (\mathbf{I}_n - 2\mathbf{D}(S_i^k))\mathbf{L}_k\mathbf{c}_i^* \preceq 0, \qquad (\mathbf{I}_n - 2\mathbf{D}(S_i^k))\mathbf{L}_k\mathbf{c}_i'^* \preceq 0, \qquad \boldsymbol{\lambda}_{ik}^* \succeq 0, \quad \boldsymbol{\lambda}_{ik}'^* \succeq 0, \\ \boldsymbol{\lambda}_{ik}^{*\top}(\mathbf{I}_n - 2\mathbf{D}(S_i^k))\mathbf{L}_k\mathbf{c}_i^* = 0, \qquad \boldsymbol{\lambda}_{ik}'^{*\top}(\mathbf{I}_n - 2\mathbf{D}(S_i^k))\mathbf{L}_k\mathbf{c}_i'^* = 0, \qquad \forall i,k \\ \nabla_{\mathbf{c}_i} f(\mathbf{c}_i^*,\mathbf{c}_i'^*) + \sum_{i=1}^{P_{conv}}\sum_{k=1}^{K}\left((\mathbf{I}_n - 2\mathbf{D}(S_i^k))\mathbf{L}_k\right)^\top \boldsymbol{\lambda}_{ik}^* = 0, \\ \nabla_{\mathbf{c}_i'} f(\mathbf{c}_i^*,\mathbf{c}_i'^*) + \sum_{i=1}^{P_{conv}}\sum_{k=1}^{K}\left((\mathbf{I}_n - 2\mathbf{D}(S_i^k))\mathbf{L}_k\right)^\top \boldsymbol{\lambda}_{ik}'^* = 0, \end{aligned} \tag{10}$$

where $\nabla_{\mathbf{c}_i} f(\mathbf{c}_i^*,\mathbf{c}_i'^*)$ and $\nabla_{\mathbf{c}_i'} f(\mathbf{c}_i^*,\mathbf{c}_i'^*)$ are not expanded out for simplicity. Since our formulation in equation [9] is convex and finite dimensional ($2hP_{conv}$ variables and $2nP_{conv}K$ constraints), any points $\mathbf{c}_i^*$, $\mathbf{c}_i'^*$, $\boldsymbol{\lambda}_{ik}^*$, $\boldsymbol{\lambda}_{ik}'^*$ that satisfy the KKT conditions in equation [10] are primal and dual optimal, with zero duality gap. The optimal filter weights $\mathbf{u}_j^*$ and fully-connected layer weights w_j^* can be constructed from $\mathbf{c}_i^*$ and $\mathbf{c}_i'^*$ following Theorem 2.1 of [8].

A.5 Computing Gradients of CNN Layer Block

Let us consider a layer block of the CNN that we used:

$$\begin{aligned} \mathbf{L} &= \text{input} \\ \mathbf{z} &= conv(\mathbf{K_1}, L) \\ \mathbf{h} &= \text{ReLU}(z) \\ \theta &= conv(\mathbf{K_2}, h) \\ \mathbf{x} &= \text{ReLU}(\theta) \\ \hat{\mathbf{Y}} &= (\hat{a}, \hat{b}) = (x) \\ \mathbf{J} &= J_{\text{loss}}(\hat{Y}) \end{aligned}$$

where $conv(K,X)_{i,j} = \sum_{u,v} K_{i-u,j-v} \cdot X_{u,v}$ represents the mathematical operation of a convolution layer. We use K_1 and K_2 to represent the weight and bias terms for the respective convolution layers. Following the gradient analysis steps in [13] we consider the useful network gradients that we would like to compute for our network:

$$\frac{\partial J}{\partial L}, \frac{\partial J}{\partial z}, \frac{\partial J}{\partial h}, \frac{\partial J}{\partial \theta}, \frac{\partial J}{\partial \hat{Y}}$$

Let's start by finding the derivative of the ReLU function. Considering $\text{ReLU}(x) = max(x, 0)$, the derivative of the ReLU function with respect to x is

$$\text{ReLU}'(x) = \begin{cases} 1 & \text{if } x \geq 0 \\ 0 & \text{if otherwise} \end{cases} = \text{sgn}(\text{ReLU}(x))$$

where $sgn(.)$ is the signum function. Now let's use chain rule to see the relationship between these gradients

$$\frac{\partial J}{\partial \theta} = \frac{\partial J}{\partial \hat{Y}} * \frac{\partial \hat{Y}}{\partial \theta}$$
$$\frac{\partial J}{\partial h} = \frac{\partial J}{\partial \theta} * \frac{\partial \theta}{\partial h}$$
$$\frac{\partial J}{\partial z} = \frac{\partial J}{\partial h} * \frac{\partial h}{\partial z}$$

Notice that the $\frac{\partial \hat{Y}}{\partial \theta} = \text{ReLU'}(\theta) = \text{sgn}(\text{ReLU}(\theta))$ and similarly $\frac{\partial h}{\partial z} = \text{sgn}(\text{ReLU}(h))$. We also assume the following gradients:

$$\frac{\partial z}{\partial L} = \delta_{\mathbf{K_1}} \quad \frac{\partial z}{\partial K_1} = \alpha_{\mathbf{K_1}}$$
$$\frac{\partial \theta}{\partial h} = \delta_{\mathbf{K_2}} \quad \frac{\partial \theta}{\partial K_2} = \alpha_{\mathbf{K_2}}$$

For more details on how to take gradients of two or three layer ReLU convolution networks please refer to [14]. Now, we will compute $\frac{\partial J}{\partial \hat{Y}}$, we start with our objective loss [2] and take the gradient as followed. Note that the partial gradient with respect to $\hat{Y}_{h',w'}$ is 0 at any point $h, w \in R^{H \times W}$ where $h, w \neq h', w'$ and the summation can be removed.

$$\begin{aligned} \mathbf{J} &= \frac{1}{2} \sum_{h,w} \|\mathbf{Y}_{h,w} - \hat{\mathbf{Y}}_{h,w}(\mathbf{L}; \mathbf{\Theta})\|_2^2 + \gamma \sum_{h,w} (\mathbf{C}_{h,w} - \hat{\mathbf{C}}_{h,w})^2 \\ &= \frac{1}{2} \sum_{h,w} \|\mathbf{Y}_{h,w} - \hat{\mathbf{Y}}_{h,w}\|_2^2 + \gamma \sum_{h,w} (\|\mathbf{Y}_{h,w}\|_2 - \|\hat{\mathbf{Y}}_{h,w}\|_2)^2 \end{aligned}$$

$$\begin{aligned} \frac{\partial J}{\partial \hat{Y}_{h',w'}} &= \frac{\partial}{\partial \hat{Y}_{h',w'}} \left[\frac{1}{2} \sum_{h,w} \|\mathbf{Y}_{h,w} - \hat{\mathbf{Y}}_{h,w}\|_2^2 + \gamma \sum_{h,w} (\|\mathbf{Y}_{h,w}\|_2 - \|\hat{\mathbf{Y}}_{h,w}\|_2)^2 \right] \\ &= \frac{\partial}{\partial \hat{Y}_{h',w'}} \left[\frac{1}{2} \|\mathbf{Y}_{h',w'} - \hat{\mathbf{Y}}_{h',w'}\|_2^2 + \gamma (\|\mathbf{Y}_{h',w'}\|_2 - \|\hat{\mathbf{Y}}_{h',w'}\|_2)^2 \right] \\ &= (\hat{\mathbf{Y}}_{h',w'} - \mathbf{Y}_{h',w'}) + 2\gamma (\|\mathbf{Y}_{h',w'}\|_2 - \|\hat{\mathbf{Y}}_{h',w'}\|_2) \frac{\partial}{\partial \hat{Y}_{h',w'}} [\|\mathbf{Y}_{h',w'}\|_2 - \|\hat{\mathbf{Y}}_{h',w'}\|_2] \\ &= \hat{\mathbf{Y}}_{h',w'} - \mathbf{Y}_{h',w'} - 2\gamma (\|\mathbf{Y}_{h',w'}\|_2 - \|\hat{\mathbf{Y}}_{h',w'}\|_2) \frac{\hat{\mathbf{Y}}_{h',w'}}{\|\hat{\mathbf{Y}}_{h',w'}\|_2} \\ &= \hat{\mathbf{Y}}_{h',w'}(1 + 2\gamma) - \mathbf{Y}_{h',w'} - \frac{2\gamma \|\mathbf{Y}_{h',w'}\|_2 \hat{\mathbf{Y}}_{h',w'}}{\|\hat{\mathbf{Y}}_{h',w'}\|_2} \end{aligned}$$

$$\frac{\partial J}{\partial \hat{Y}} = \hat{\mathbf{Y}}(1 + 2\gamma - \frac{2\gamma \|\mathbf{Y}\|_2}{\|\hat{\mathbf{Y}}\|_2}) - \mathbf{Y}$$

Now that we have most of the intermediate network gradients, we can substitute them into our chain rule equations.

$$\frac{\partial J}{\partial \theta} = \frac{\partial J}{\partial \hat{Y}} * \frac{\partial \hat{Y}}{\partial \theta} = (\hat{\mathbf{Y}}(1 + 2\gamma - \frac{2\gamma \|\mathbf{Y}\|_2}{\|\hat{\mathbf{Y}}\|_2}) - \mathbf{Y}) * \text{sgn}(\text{ReLU}(\theta))$$

$$\frac{\partial J}{\partial h} = \frac{\partial J}{\partial \theta} * \frac{\partial \theta}{\partial h} = (\hat{\mathbf{Y}}(1 + 2\gamma - \frac{2\gamma \|\mathbf{Y}\|_2}{\|\hat{\mathbf{Y}}\|_2}) - \mathbf{Y}) * \text{sgn}(\text{ReLU}(\theta)) * \delta_{K_1}$$

$$\frac{\partial J}{\partial z} = \frac{\partial J}{\partial h} * \frac{\partial h}{\partial z} = (\hat{\mathbf{Y}}(1 + 2\gamma - \frac{2\gamma \|\mathbf{Y}\|_2}{\|\hat{\mathbf{Y}}\|_2}) - \mathbf{Y}) * \text{sgn}(\text{ReLU}(\theta)) * \delta_{K_1} * \text{sgn}(\text{ReLU}(h))$$

Finally, we can compute the gradient of the loss with respect to the network weights K_1 and K_2.

$$\frac{\partial J}{\partial K_1} = \frac{\partial J}{\partial z} * \frac{\partial z}{\partial K_1} = (\hat{\mathbf{Y}}(1 + 2\gamma - \frac{2\gamma\|\mathbf{Y}\|_2}{\|\hat{\mathbf{Y}}\|_2}) - \mathbf{Y}) * \text{sgn}(\text{ReLU}(\theta)) * \delta_{K_1} * \text{sgn}(\text{ReLU}(h)) * \alpha_{K_1}$$

$$\frac{\partial J}{\partial K_2} = \frac{\partial J}{\partial \theta} * \frac{\partial \theta}{\partial K_2} = (\hat{\mathbf{Y}}(1 + 2\gamma - \frac{2\gamma\|\mathbf{Y}\|_2}{\|\hat{\mathbf{Y}}\|_2}) - \mathbf{Y}) * \text{sgn}(\text{ReLU}(\theta)) * \alpha_{K_2}$$

These are exactly the gradients we need to start deriving the dual problem from our primal problem formulation.

A.6 KKT Conditions for Primal Formulation

1. Stationary Conditions

$$\nabla_{\mathbf{K_1}} \mathbf{L_g}(K_1, K_2, \lambda_1, \lambda_2) = 0$$
$$\nabla_{\mathbf{K_2}} \mathbf{L_g}(K_1, K_2, \lambda_1, \lambda_2) = 0$$

2. Complementary Slackness

$$\lambda_1(-\gamma) = 0$$
$$\lambda_2(\|\mathbf{K_1}\|^2 + \|\mathbf{K_2}\|^2 - t) = 0$$

3. Primal Feasibility

$$-\gamma \leq 0$$
$$\|\mathbf{K_1}\|^2 + \|\mathbf{K_2}\|^2 - t \leq 0$$

4. Dual Feasibility

$$\lambda_1 \geq 0$$
$$\lambda_2 \geq 0$$

Kernel Gradient Full Formulas

$$\begin{aligned}\nabla_{\mathbf{K_1}} \mathbf{L_g}(K_1, K_2, \lambda_1, \lambda_2) &= \frac{\partial J}{\partial K_1} + 2\lambda_2 \mathbf{K_1} \\ &= (\hat{\mathbf{Y}}(1 + 2\gamma - \frac{2\gamma\|\mathbf{Y}\|_2}{\|\hat{\mathbf{Y}}\|_2}) - \mathbf{Y}) * \text{sgn}(\text{ReLU}(\theta)) * \delta_{K_1} * \text{sgn}(\text{ReLU}(h)) * \alpha_{K_1} + 2\lambda_2 \mathbf{K_1}\end{aligned}$$

$$\begin{aligned}\nabla_{\mathbf{K_2}} \mathbf{L_g}(K_1, K_2, \lambda_1, \lambda_2) &= \frac{\partial J}{\partial K_2} + 2\lambda_2 \mathbf{K_2} \\ &= (\hat{\mathbf{Y}}(1 + 2\gamma - \frac{2\gamma\|\mathbf{Y}\|_2}{\|\hat{\mathbf{Y}}\|_2}) - \mathbf{Y}) * \text{sgn}(\text{ReLU})(\theta)) * \alpha_{K_2} + 2\lambda_2 \mathbf{K_2}\end{aligned}$$