RETHINKING IMAGE HISTOGRAM MATCHING FOR IMAGE CLASSIFICATION

Supplementary Materials

A. DIFFERENTIABILITY OF CONVENTIONAL HISTOGRAM MATCHING

We explain why it is difficult to modify conventional histogram matching (HM) to be differentiable. This challenge is due to conventional HM containing two non-differentiable processes. We first explain the processes of conventional HM. We then explain the two non-differentiable processes in conventional HM: (1) cumulative distribution function (CDF) calculation and (2) look up table (LUT) calculation.

A.1. Processes of Conventional Histogram Matching

Conventional HM [1] transforms the CDF of a source image to match the CDF of a target image. Conventional HM first computes the CDFs of both the source image $\mathbf{x}_S$ and the target image $\mathbf{x}_T$, for each color channel. We denote the CDF of $\mathbf{x}_S$ as $F_{\mathbf{x}_S}$ and the CDF of $\mathbf{x}_T$ as $F_{\mathbf{x}_T}$. Then, conventional HM creates a mapping function $M(p)$ of pixel value p to match $F_{\mathbf{x}_S}$ to $F_{\mathbf{x}_T}$. Theoretically, the mapping function $M(p)$ is expressed as $F_{\mathbf{x}_T}^{-1}(F_{\mathbf{x}_S}(p))$. However, in general, finding analytical expressions for $F_{\mathbf{x}_T}^{-1}$ is not a trivial task [1]. To address this, in practice, conventional HM uses an approximation method. The mapping function $M(p)$ is expressed as an LUT. Finally, the pixels of the source image $\mathbf{x}_S$ with value p are replaced with $M(p)$ to obtain $\mathbf{x}_M$ whose CDF is approximately equal to $F_{\mathbf{x}_T}$. Conventional HM involves two non-differentiable processes: (1) CDF calculation and (2) LUT calculation. We explain these processes in the following.

A.2. Non-Differentiable CDF Calculation

$F_{\mathbf{x}_T}$ and $F_{\mathbf{x}_S}$ are obtained as the cumulative sums of the histograms of $\mathbf{x}_T$ and $\mathbf{x}_S$. Let $\mathbf{x} = \{x_i \mid i = 1, 2, \cdots, H \times W\}$ denote the input N-bit digital image flattened to 1D, where H and W represent the height and width of the input image. Let $p \in \mathcal{P} = \{0, 1, \cdots, 2^N - 1\}$ denote the pixel value of the input image. The histogram of the input image $h_{\mathbf{x}}(p)$ is calculated using the bin assignment function $\delta(x_i, p)$:

$$h_{\mathbf{x}}(p) = \sum_{i=1}^{H \times W} \delta(x_i, p), \tag{1}$$

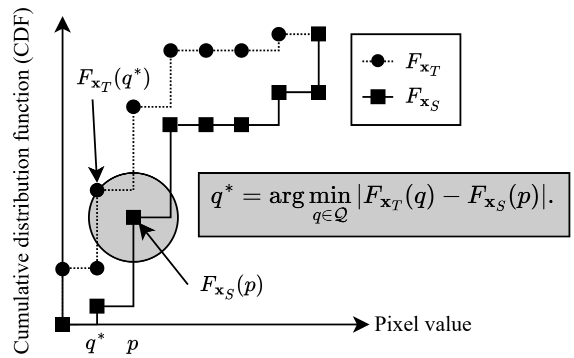

Fig. 5. The LUT calculation by finding the pixel value q^* such that $F_{\mathbf{x}_T}(q^*)$ is the nearest value to $F_{\mathbf{x}_S}(p)$.

where, the bin assignment function $\delta(x_i, p)$ is defined as:

$$\delta(x_i, p) = \begin{cases} 1 & \text{if } x_i = p, \\ 0 & \text{otherwise.} \end{cases} \tag{2}$$

For each pixel value $p \in \mathcal{P}$, the CDF $F_{\mathbf{x}}$ is defined as:

$$F_{\mathbf{x}}(p) = \frac{1}{HW} \sum_{k=0}^{p} h_{\mathbf{x}}(k), \tag{3}$$

The derivative of the bin assignment function $\delta(x_i, p)$ w.r.t. x_i is zero for all i. This feature makes the CDF calculation in HM non-differentiable. PyTorch [2], a widely used framework in deep learning research, provides histogram calculation functions such as `torch.histc()`, `torch.histogram()`, and `torch.bincount()`. However, these functions lack implementation of a backward function for floating-point inputs[1]. To address this zero gradient problem, some researchers [3, 4, 5, 6] make the histogram calculation differentiable by replacing the bin assignment function with another differentiable function.

A.3. Non-Differentiable LUT Calculation

Conventional HM calculates an LUT $M(p)$ using the CDF of the target image $F_{\mathbf{x}_T}$ and the CDF of the source image $F_{\mathbf{x}_S}$. Then the conventional HM applies the LUT to the source image. Fig. 5 shows the process of LUT calculation. The LUT

[1]This behavior was confirmed using PyTorch 2.5.0+cu124.

Algorithm 1 Practical LUT calculation process in HM

Require: $F_{\mathbf{x}_S}, F_{\mathbf{x}_T}, \mathcal{P} = \mathcal{Q} = \{0, 1, \cdots, 2^N - 1\}$
Ensure: LUT M
for each $p \in \mathcal{P}$ **do**
    $d_{min} \leftarrow \infty$
    $q^* \leftarrow 0$
    for each $q \in \mathcal{Q}$ **do**
        $d \leftarrow |F_{\mathbf{x}_T}(q) - F_{\mathbf{x}_S}(p)|$
        if $d < d_{min}$ **then**
            $d_{min} \leftarrow d$
            $q^* \leftarrow q$
        end if
    end for
    $M(p) \leftarrow q^*$
end for
return M

of pixel value $M(p)$ is calculated by finding the pixel value q^* such that $F_{\mathbf{x}_T}(q^*)$ is the nearest value to $F_{\mathbf{x}_S}(p)$. Specifically, q^* is obtained as:

$$q^* = \arg\min_{q \in \mathcal{Q}} |F_{\mathbf{x}_T}(q) - F_{\mathbf{x}_S}(p)|. \tag{4}$$

where $\mathcal{Q} = \{0, 1, \cdots, 2^N - 1\}$ is the set of possible pixel values (domain of the $F_{\mathbf{x}_T}$). For each pixel value $p \in \mathcal{P}$, LUT $M(p)$ is defined as:

$$M(p) = q^* \tag{5}$$

The optimal q^* is given by $F_{\mathbf{x}_T}^{-1}(F_{\mathbf{x}_S}(p))$. However, in general, finding analytical expressions for $F_{\mathbf{x}_T}^{-1}$ is not a trivial task [1]. In practice, to obtain q^* for each $p \in \mathcal{P}$, conventional HM calculate $|F_{\mathbf{x}_T}(q) - F_{\mathbf{x}_S}(p)|$ for each $q \in \mathcal{Q}$ and find the q that minimizes $|F_{\mathbf{x}_T}(q) - F_{\mathbf{x}_S}(p)|$. Algorithm 1 shows this practical LUT calculation process. In this practical LUT calculation, the derivative of $M(p)$ w.r.t. $F_{\mathbf{x}_T}(k)$ is zero for all $k \in \mathcal{P}$. This is because $M(p)$ is not a function of $F_{\mathbf{x}_T}(k)$, for all k. This feature makes the LUT calculation in HM non-differentiable.

B. REFERENCES

[1] Rafael C. Gonzalez, "Intensity transformations and spatial filtering," in *Digital Image Processing*, chapter 3, pp. 119–153. Pearson Deutschland, 4th edition, 2022.

[2] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer, "Automatic differentiation in pytorch," in *Adv. Neural Inform. Process. Syst. Worksh.*, 2017.

[3] Evgeniya Ustinova and Victor Lempitsky, "Learning deep embeddings with histogram loss," in *Adv. Neural Inform. Process. Syst.*, D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, Eds., 2016, vol. 29.

[4] Zhe Wang, Hongsheng Li, Wanli Ouyang, and Xiaogang Wang, "Learnable histogram: Statistical context features for deep neural networks," in *Eur. Conf. Comput. Vis.*, Cham, 2016, pp. 246–262.

[5] Ibrahim Yusuf, George Igwegbe, and Oluwafemi Azeez, "Differentiable histogram with hard-binning," *arXiv preprint arXiv:2012.06311*, 2020.

[6] Mor Avi-Aharon, Assaf Arbelle, and Tammy Riklin Raviv, "Differentiable histogram loss functions for intensity-based image-to-image translation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 45, no. 10, pp. 11642–11653, 2023.