Appendix of "Rethinking Uncertainty in Active Learning for Image Classification: When Confidence Is Misleading"

Zilong Ling
E125111005@stu.ahu.edu.cn
Anhui Provincial International Joint Research Center for Advanced Technology in Medical Imaging, School of Computer Science and Technology, Anhui University
Hefei, China

Huabin Wang
wanghuabin@ahu.edu.cn
Anhui Provincial International Joint Research Center for Advanced Technology in Medical Imaging, School of Computer Science and Technology, Anhui University
Hefei, China

Liang Du
duliang@sxu.edu.cn
School of Computer and Information Technology, Shanxi University
Taiyuan, China

Xuejun Li
xjli@ahu.edu.cn
Anhui Provincial International Joint Research Center for Advanced Technology in Medical Imaging, School of Computer Science and Technology, Anhui University
Hefei, China

Peng Zhou*
zhoupeng@ahu.edu.cn
Anhui Provincial International Joint Research Center for Advanced Technology in Medical Imaging, School of Computer Science and Technology, Anhui University
Hefei, China

ACM Reference Format:
Zilong Ling, Huabin Wang, Liang Du, Xuejun Li, and Peng Zhou. 2026. Appendix of "Rethinking Uncertainty in Active Learning for Image Classification: When Confidence Is Misleading". In . ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

A Weighted Wavelet Transform (WWT)

In this section, we provide more details about the Weighted Wavelet Transform (WWT) $\mathcal{H}(\mathbf{x})$ [11, 12, 17].

Given an image $\mathbf{X} \in \mathbb{R}^{H\times W\times C}$, we first flatten each channel into a vector $\mathbf{x} \in \mathbb{R}^{(H*W)}$ whose length is $H * W$. $\mathcal{H}$ is a linear transformation on $\mathbf{x}$, i.e., $\mathcal{H}(\mathbf{x}) = \mathbf{H}\mathbf{x}$. The transformation matrix $\mathbf{H} \in \mathbb{R}^{\frac{(H*W)}{4}\times(H*W)}$ is defined as:

$$\mathbf{H} = \mathbf{C}_{l2} \otimes \mathbf{C}_{l1} + \mathbf{C}_{h2} \otimes \mathbf{C}_{l1} + \mathbf{C}_{l2} \otimes \mathbf{C}_{h1} + \mathbf{C}_{h2} \otimes \mathbf{C}_{h1}, \tag{1}$$

where $\otimes$ denotes the Kronecker product. $\mathbf{C}_{l1}, \mathbf{C}_{l2}$ are two coefficent matrices for the low-frequency parts and $\mathbf{C}_{h1}, \mathbf{C}_{h2}$ are two coefficent matrices for the high-frequency parts, respectively. Their definitions are:

$$\mathbf{C}_{l1} = \begin{bmatrix} \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} & 0 & 0 & \cdots & 0 & 0 \\ 0 & 0 & \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} \end{bmatrix} \in \mathbb{R}^{\frac{W}{2}\times W},$$

$$\mathbf{C}_{l2} = \begin{bmatrix} \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} & 0 & 0 & \cdots & 0 & 0 \\ 0 & 0 & \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} \end{bmatrix} \in \mathbb{R}^{\frac{H}{2}\times H} \tag{2}$$

and

$$\mathbf{C}_{h1} = \begin{bmatrix} \frac{\sqrt{2}}{2} & -\frac{\sqrt{2}}{2} & 0 & 0 & \cdots & 0 & 0 \\ 0 & 0 & \frac{\sqrt{2}}{2} & -\frac{\sqrt{2}}{2} & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & \frac{\sqrt{2}}{2} & -\frac{\sqrt{2}}{2} \end{bmatrix} \in \mathbb{R}^{\frac{W}{2}\times W},$$

$$\mathbf{C}_{h2} = \begin{bmatrix} \frac{\sqrt{2}}{2} & -\frac{\sqrt{2}}{2} & 0 & 0 & \cdots & 0 & 0 \\ 0 & 0 & \frac{\sqrt{2}}{2} & -\frac{\sqrt{2}}{2} & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & \frac{\sqrt{2}}{2} & -\frac{\sqrt{2}}{2} \end{bmatrix} \in \mathbb{R}^{\frac{H}{2}\times H}. \tag{3}$$

More details of WWT can be found in [11, 12, 17].

B Proof of Lemma 3.1

Before proving Lemma 3.1, we reclaim our assumption of the dataset $\mathcal{X}$ with K classes. We assume the samples in the k-th class follow the Gaussian distribution $\mathcal{N}(\boldsymbol{\mu}_k, \Sigma)$, where $\boldsymbol{\mu}_k \in \mathbb{R}^p$ is the mean vector of the k-th class and $\Sigma \in \mathbb{R}^{p\times p}$ is the covariance matrix and p is the dimension of data. We denote the prior distribution of each class as $\pi_1, \pi_2, \cdots, \pi_K$. Then, we have the following Lemma:

Lemma B.1. *Suppose data $\mathcal{X}$ satisfies the above assumption. $\mathcal{F}(\cdot)$ is trained on $\mathcal{X}$ and $\mathcal{A}(\cdot)$ is trained on $\mathcal{H}(\mathcal{X})$. Then, for any $\mathbf{x} \in \mathcal{X}$, the predicted confidence to the k-th class of $\mathcal{F}(\mathbf{x})$, i.e., the k-th logit of $\mathcal{F}(\mathbf{x})$, which is denoted as $\mathcal{F}_k(\mathbf{x})$, is*

$$\mathcal{F}_k(\mathbf{x}) = \boldsymbol{\mu}_k^\top \Sigma^{-1}(\mathbf{x} - \frac{1}{2}\boldsymbol{\mu}_k) + \ln \pi_k, \tag{4}$$

and the predicted confidence to the k-th class of $\mathcal{A}(\mathcal{H}(\mathbf{x}))$, i.e., the k-th logit of $\mathcal{A}(\mathcal{H}(\mathbf{x}))$, which is denoted as $\mathcal{A}_k(\mathcal{H}(\mathbf{x}))$, is

$$\mathcal{A}_k(\mathcal{H}(\mathbf{x})) = \boldsymbol{\mu}_k^\top \mathbf{B}(\mathbf{x} - \frac{1}{2}\boldsymbol{\mu}_k) + \ln \pi_k, \tag{5}$$

*Corresponding author.

where $\mathbf{B} = \mathbf{H}^\top(\mathbf{H}\Sigma\mathbf{H}^\top)^{-1}\mathbf{H}$.

Proof. We first prove Eq.(4). $\mathcal{F}_k(\mathbf{x})$ denotes the logit that $\mathbf{x}$ belongs to the k-th class, and in this classifier, the model classifies $\mathbf{x}$ to the k-th class if $k = \arg\max_j \mathcal{F}_j(\mathbf{x})$. Therefore, the decision boundary between the k-th class and the j-th class is decided by the equation $\mathcal{F}_k(\mathbf{x}) = \mathcal{F}_j(\mathbf{x})$. To this end, to calculate $\mathcal{F}_k(\mathbf{x})$, we can calculate the decision boundary of the k-th class and the j-th class in our dataset $\mathcal{X}$.

Notice that we assume the samples in the k-th class should follow the Gaussian distribution $\mathcal{N}(\boldsymbol{\mu}_k, \Sigma)$. According to [5], we have that the decisoin boundary between the k-th and the j-th class is

$$\boldsymbol{\mu}_k^\top\Sigma^{-1}(\mathbf{x} - \frac{1}{2}\boldsymbol{\mu}_k) + \ln\pi_k = \boldsymbol{\mu}_j^\top\Sigma^{-1}(\mathbf{x} - \frac{1}{2}\boldsymbol{\mu}_j) + \ln\pi_j \tag{6}$$

Therefore, we have $\mathcal{F}_k(\mathbf{x}) = \boldsymbol{\mu}_k^\top\Sigma^{-1}(\mathbf{x} - \frac{1}{2}\boldsymbol{\mu}_k) + \ln\pi_k$.

Now, we prove Eq.(5). Consider $\mathcal{H}(\mathbf{x}) = \mathbf{H}\mathbf{x}$. Since $\mathbb{P}(\mathbf{x}|y = k)$ follows the Gaussian distribution $\mathcal{N}(\boldsymbol{\mu}_k, \Sigma)$, its linear transformation $\mathbb{P}(\mathbf{H}\mathbf{x}|y = k)$ also follows the Gaussian distribution, whose mean vector is $\hat{\boldsymbol{\mu}}_k = \mathbf{H}\boldsymbol{\mu}_k$ and covariance matrix $\hat{\Sigma} = \mathbf{H}\Sigma\mathbf{H}^\top$ [16]. Take them into the above decision boundary Eq.(6), we have that:

$$\begin{aligned}\mathcal{A}_k(\mathcal{H}(\mathbf{x})) &= \hat{\boldsymbol{\mu}}_k^\top\hat{\Sigma}^{-1}(\mathbf{H}\mathbf{x} - \frac{1}{2}\hat{\boldsymbol{\mu}}_k) + \ln\pi_k \\ &= (\mathbf{H}\boldsymbol{\mu}_k)^\top(\mathbf{H}\Sigma\mathbf{H}^T)^{-1}(\mathbf{H}\mathbf{x} - \frac{1}{2}\mathbf{H}\boldsymbol{\mu}_k) + \ln\pi_k \\ &= \boldsymbol{\mu}_k^\top\mathbf{H}^\top(\mathbf{H}\Sigma\mathbf{H}^T)^{-1}\mathbf{H}(\mathbf{x} - \frac{1}{2}\boldsymbol{\mu}_k) + \ln\pi_k \\ &= \boldsymbol{\mu}_k^\top\mathbf{B}(\mathbf{x} - \frac{1}{2}\boldsymbol{\mu}_k) + \ln\pi_k,\end{aligned} \tag{7}$$

where $\mathbf{B} = \mathbf{H}^\top(\mathbf{H}\Sigma\mathbf{H}^T)^{-1}\mathbf{H}$. This concludes the proof of Lemma B.1. □

Now, we are ready to prove Lemma 3.1. We compute the gradient of $\mathcal{F}_k(\mathbf{x})$ w.r.t. $\mathbf{x}$:

$$\frac{\partial\mathcal{F}_k(\mathbf{x})}{\partial\mathbf{x}} = \frac{\partial(\boldsymbol{\mu}_k^\top\Sigma^{-1}(\mathbf{x} - \frac{1}{2}\boldsymbol{\mu}_k) + \ln\pi_k)}{\partial\mathbf{x}} = \Sigma^{-1}\boldsymbol{\mu}_k. \tag{8}$$

This concludes the proof of Lemma 3.1.

C Proof of Theorem 3.2

According to Lemma B.1, we have

$$\begin{aligned}\|\mathcal{F}(\mathbf{x}) - \mathcal{A}(\mathcal{H}(\mathbf{x}))\|_2 &= \sqrt{\sum_{k=1}^{K}(\mathcal{F}_k(\mathbf{x}) - \mathcal{A}_k(\mathcal{H}(\mathbf{x})))^2} \\ &= \sqrt{\sum_{k=1}^{K}\left(\boldsymbol{\mu}_k^\top\Sigma^{-1}\left(\mathbf{x} - \frac{1}{2}\boldsymbol{\mu}_k\right) - \boldsymbol{\mu}_k^\top\mathbf{B}\left(\mathbf{x} - \frac{1}{2}\boldsymbol{\mu}_k\right)\right)^2} \\ &= \sqrt{\sum_{k=1}^{K}\left(\boldsymbol{\mu}_k^\top(\Sigma^{-1} - \mathbf{B})\left(\mathbf{x} - \frac{1}{2}\boldsymbol{\mu}_k\right)\right)^2}.\end{aligned} \tag{9}$$

We define $\mathcal{D}(\mathbf{x}) = \mathcal{F}(\mathbf{x}) - \mathcal{A}(\mathcal{H}(\mathbf{x}))$ as the difference vector function in the logit space. According to Eq.(9), $\mathcal{D}(\mathbf{x}) = \mathbf{M}(\Sigma^{-1} - \mathbf{B})(\mathbf{x} - \frac{1}{2}\boldsymbol{\mu}_k)$ (where $\mathbf{M} = [\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_K]^T$) is a linear function, and thus it is L-Lipschitz continuous. More specifically, for any two vectors $\mathbf{x}_1$ and $\mathbf{x}_2$, we have

$$\begin{aligned}\|\mathcal{D}(\mathbf{x}_1) - \mathcal{D}(\mathbf{x}_2)\|_2 &= \|\mathbf{M}(\Sigma^{-1} - \mathbf{B})(\mathbf{x}_1 - \mathbf{x}_2)\|_2 \\ &\le \|\mathbf{M}(\Sigma^{-1} - \mathbf{B})\|_2\|\mathbf{x}_1 - \mathbf{x}_2\|_2\end{aligned} \tag{10}$$

where $\|\mathbf{M}(\Sigma^{-1} - \mathbf{B})\|_2$ is the spectral norm of matrix $\mathbf{M}(\Sigma^{-1} - \mathbf{B})$, i.e., the largest singular value of $\mathbf{M}(\Sigma^{-1} - \mathbf{B})$ and we denote it as L.

Then, given a normal sample $\mathbf{x}$ that belongs to the k-th class, its perturbation is $\hat{\mathbf{x}} = \mathbf{x} - \eta\Sigma^{-1}\boldsymbol{\mu}_k$, and the corresponding misleading sample $\mathbf{x}^*$ satisfies $\|\mathbf{x}^* - \hat{\mathbf{x}}\| \le \epsilon$. Then, we have:

$$\begin{aligned}\|\mathcal{F}(\mathbf{x}^*) - \mathcal{A}(\mathcal{H}(\mathbf{x}^*))\|_2 &= \|\mathcal{D}(\mathbf{x}^*)\|_2 \\ &= \|\mathcal{D}(\hat{\mathbf{x}}) - (\mathcal{D}(\hat{\mathbf{x}}) - \mathcal{D}(\mathbf{x}^*))\|_2 \\ &\ge \|\mathcal{D}(\hat{\mathbf{x}})\|_2 - \|\mathcal{D}(\hat{\mathbf{x}}) - \mathcal{D}(\mathbf{x}^*)\|_2 \\ &\ge \|\mathcal{D}(\hat{\mathbf{x}})\|_2 - L\|\mathbf{x}^* - \hat{\mathbf{x}}\|_2 \\ &\ge \|\mathcal{D}(\hat{\mathbf{x}})\|_2 - L\epsilon \\ &= \|\mathcal{F}(\hat{\mathbf{x}}) - \mathcal{A}(\mathcal{H}(\hat{\mathbf{x}}))\|_2 - L\epsilon\end{aligned} \tag{11}$$

where the first inequality holds because of the triangle inequality, the second inequality holds because of the L-Lipschitz of $\mathcal{D}(\mathbf{x})$ (i.e., Eq.(10)), and the last inequality holds because of $\|\mathbf{x}^* - \hat{\mathbf{x}}\| \le \epsilon$.

To ensure that the misleading sample $\mathbf{x}^*$ satisfies $\|\mathcal{F}(\mathbf{x}^*) - \mathcal{A}(\mathcal{H}(\mathbf{x}^*))\|_2 \ge \delta$, we just need to make $\|\mathcal{F}(\hat{\mathbf{x}}) - \mathcal{A}(\mathcal{H}(\hat{\mathbf{x}}))\|_2 \ge \delta + L\epsilon$, which means:

$$\begin{aligned}&\mathbb{P}\left\{\left\|\mathcal{F}(\mathbf{x}^*) - \mathcal{A}\big(\mathcal{H}(\mathbf{x}^*)\big)\right\|_2 \ge \delta\right\} \\ &\ge \mathbb{P}\left\{\left\|\mathcal{F}(\hat{\mathbf{x}}) - \mathcal{A}\big(\mathcal{H}(\hat{\mathbf{x}})\big)\right\|_2 \ge \delta + L\epsilon\right\}.\end{aligned} \tag{12}$$

Hence, we compute the lower bound of $\mathbb{P}\{||\mathcal{F}(\hat{\mathbf{x}}) - \mathcal{A}(\mathcal{H}(\hat{\mathbf{x}}))||_2 \ge \delta + L\epsilon\}$ now.

Substituting $\hat{\mathbf{x}} = \mathbf{x} - \eta\Sigma^{-1}\boldsymbol{\mu}_k$ into Eq.(9), we have:

$$\begin{aligned}||\mathcal{F}(\hat{\mathbf{x}}) - \mathcal{A}(\mathcal{H}(\hat{\mathbf{x}}))||_2 &= \sqrt{\sum_{j=1}^{K}\left(\boldsymbol{\mu}_j^\top(\Sigma^{-1} - \mathbf{B})\left(\hat{\mathbf{x}} - \frac{1}{2}\boldsymbol{\mu}_j\right)\right)^2} \\ &\ge \left|\boldsymbol{\mu}_k^\top(\Sigma^{-1} - \mathbf{B})\left(\hat{\mathbf{x}} - \frac{1}{2}\boldsymbol{\mu}_k\right)\right| \\ &= \left|\boldsymbol{\mu}_k^\top(\Sigma^{-1} - \mathbf{B})\left(\mathbf{x} - \eta\Sigma^{-1}\boldsymbol{\mu}_k - \frac{1}{2}\boldsymbol{\mu}_k\right)\right|\end{aligned} \tag{13}$$

Let us denote $\mathbf{c}_k = (\Sigma^{-1} - \mathbf{B})\boldsymbol{\mu}_k$ and $\mathbf{z}_k = \mathbf{x} - \eta\Sigma^{-1}\boldsymbol{\mu}_k - \frac{1}{2}\boldsymbol{\mu}_k$. Then, Eq.(13) becomes to $||\mathcal{F}(\hat{\mathbf{x}}) - \mathcal{A}(\mathcal{H}(\hat{\mathbf{x}}))||_2 \ge |\mathbf{c}_k^\top\mathbf{z}_k|$. We denote $m_k = \mathbf{c}_k^\top\mathbf{z}_k$.

Notice that $\mathbf{x} \sim \mathcal{N}(\boldsymbol{\mu}_k, \Sigma)$, we have

$$\mathbf{z}_k \sim \mathcal{N}\left(\boldsymbol{\mu}_k - \eta\Sigma^{-1}\boldsymbol{\mu}_k - \frac{1}{2}\boldsymbol{\mu}_k, \Sigma\right) = \mathcal{N}\left(\frac{1}{2}\boldsymbol{\mu}_k - \eta\Sigma^{-1}\boldsymbol{\mu}_k, \Sigma\right) \tag{14}$$

Then, $m_k = \mathbf{c}_k^\top\mathbf{z}_k$ also follows a Gaussian distribution, i.e., $m_k \sim \mathcal{N}\left(\tilde{\mu}^{(k)}, \left(\tilde{\sigma}^{(k)}\right)^2\right)$, where $\tilde{\mu}^{(k)} = \frac{1}{2}\mathbf{c}_k^\top\boldsymbol{\mu}_k - \eta\mathbf{c}_k^\top\Sigma^{-1}\boldsymbol{\mu}_k$ and $\tilde{\sigma}^{(k)} = \sqrt{\mathbf{c}_k^\top\Sigma\mathbf{c}_k}$. Hence, we have

$$\mathbb{P}\{||\mathcal{F}(\hat{\mathbf{x}}) - \mathcal{A}(\mathcal{H}(\hat{\mathbf{x}}))||_2 \geq \delta + L\epsilon\} \geq \mathbb{P}\{|m_k| \geq \delta + L\epsilon\} \tag{15}$$
$$= \mathbb{P}\{m_k \geq \delta + L\epsilon\} + \mathbb{P}\{m_k \leq -\delta - L\epsilon\}$$
$$= \mathbb{P}\left\{\frac{m_k - \tilde{\mu}^{(k)}}{\tilde{\sigma}^{(k)}} \geq \frac{\delta + L\epsilon - \tilde{\mu}^{(k)}}{\tilde{\sigma}^{(k)}}\right\} + \mathbb{P}\left\{\frac{m_k - \tilde{\mu}^{(k)}}{\tilde{\sigma}^{(k)}} \leq \frac{-\delta - L\epsilon - \tilde{\mu}^{(k)}}{\tilde{\sigma}^{(k)}}\right\}$$
$$= \left(1 - \Phi\left(\frac{\delta + L\epsilon - \tilde{\mu}^{(k)}}{\tilde{\sigma}^{(k)}}\right)\right) + \Phi\left(\frac{-\delta - L\epsilon - \tilde{\mu}^{(k)}}{\tilde{\sigma}^{(k)}}\right)$$

where $\Phi(\cdot)$ is the cumulative distribution function (CDF) of the standard normal distribution $\mathcal{N}(0,1)$. The second equality holds because that $\frac{m_k - \tilde{\mu}^{(k)}}{\tilde{\sigma}^{(k)}}$ follows the standard normal distribution $\mathcal{N}(0,1)$. Furthermore, due to the symmetric of the Gaussian distribution, we have $1 - \Phi(a) = \Phi(-a)$ for any $a \in \mathbb{R}$. Eq.(15) can be rewritten as:

$$\mathbb{P}\{||\mathcal{F}(\hat{\mathbf{x}}) - \mathcal{A}(\mathcal{H}(\hat{\mathbf{x}}))||_2 \geq \delta + L\epsilon\} \tag{16}$$
$$\geq \left(1 - \Phi\left(\frac{\delta + L\epsilon - \tilde{\mu}^{(k)}}{\tilde{\sigma}^{(k)}}\right)\right) + \Phi\left(\frac{-\delta - L\epsilon - \tilde{\mu}^{(k)}}{\tilde{\sigma}^{(k)}}\right)$$
$$= \Phi\left(\frac{-\delta - L\epsilon + \tilde{\mu}^{(k)}}{\tilde{\sigma}^{(k)}}\right) + \Phi\left(\frac{-\delta - L\epsilon - \tilde{\mu}^{(k)}}{\tilde{\sigma}^{(k)}}\right)$$

Taking it back into Eq.(12), we have

$$\mathbb{P}\{||\mathcal{F}(\mathbf{x}^*) - \mathcal{A}(\mathcal{H}(\mathbf{x}^*))||_2 \geq \delta\} \tag{17}$$
$$\geq \Phi\left(\frac{-\delta - L\epsilon + \tilde{\mu}^{(k)}}{\tilde{\sigma}^{(k)}}\right) + \Phi\left(\frac{-\delta - L\epsilon - \tilde{\mu}^{(k)}}{\tilde{\sigma}^{(k)}}\right)$$
$$\geq \min_{k=1,\ldots,K}\left[\Phi\left(\frac{-\delta - L\epsilon + \tilde{\mu}^{(k)}}{\tilde{\sigma}^{(k)}}\right) + \Phi\left(\frac{-\delta - L\epsilon - \tilde{\mu}^{(k)}}{\tilde{\sigma}^{(k)}}\right)\right].$$

This concludes the proof of Theorem 3.2.

D Proof of Lemma 3.3

Define function $f(x) = \Phi(\frac{x-a}{b}) + \Phi(\frac{-x-a}{b})$, where $\Phi(\cdot)$ is the CDF of the standard normal distribution $\mathcal{N}(0,1)$, a and b are two positive constants, we now prove that $f(x)$ increases with increasing $|x|$.

It is easy to show that $f(x) = f(-x)$. Therefore, we just need to prove that, when $x > 0$, $f(x)$ increases monotonically with x. We denote the derivative of $\Phi(x)$ w.r.t. as $\phi(x)$. Since $\Phi(x)$ is the CDF of $\mathcal{N}(0,1)$, $\phi(x)$ is the PDF of $\mathcal{N}(0,1)$, i.e.,

$$\phi(x) = \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}}. \tag{18}$$

Then, the derivative of $f(x)$ w.r.t. x is:

$$\frac{\mathrm{d}f(x)}{\mathrm{d}x} = \frac{1}{b}\phi\left(\frac{x-a}{b}\right) - \frac{1}{b}\phi\left(\frac{-x-a}{b}\right) \tag{19}$$
$$= \frac{1}{b}\left(\phi\left(\frac{x-a}{b}\right) - \phi\left(\frac{x+a}{b}\right)\right) = \frac{1}{\sqrt{2\pi}b}\left(e^{-\frac{(x-a)^2}{2b^2}} - e^{-\frac{(x+a)^2}{2b^2}}\right)$$

Notice that $(x-a)^2 < (x+a)^2$ when $x, a > 0$. Therefore, $e^{-\frac{(x-a)^2}{2b^2}} - e^{-\frac{(x+a)^2}{2b^2}} > 0$, and thus $\frac{\mathrm{d}f(x)}{\mathrm{d}x} > 0$. It means that $f(x)$ increases monotonically with x when $x > 0$, which concludes the proof of Lemma 3.3.

E Algorithm of BUAL

Algorithm 1 shows the whole process of our BUAL method.

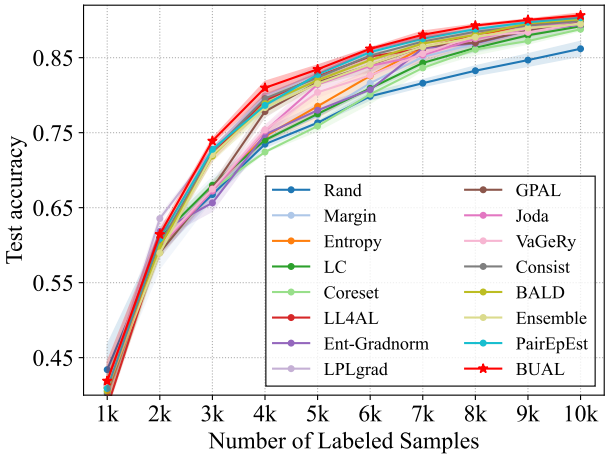
Algorithm 1 BUAL Algorithm

Input: Unlabeled dataset $\mathcal{U}$, inital labeled dataset $\mathcal{L}$, number of selection batches T, and the budget B of each selection batch.
Output: The trained model $\mathcal{F}$.
Initialize the networks $\mathcal{F}(\cdot)$, $\mathcal{A}(\cdot)$, and $\mathcal{P}(\cdot)$.
for $t = 1$ **to** T **do**
  Randomly select candidate subset C from $\mathcal{U}$.
  Compute the final score s_{fin} according to Eq.3 in Section 3.3.
  Select B samples of the smallest s_{fin} for annotation.
  Move the B annotated samples from $\mathcal{U}$ to $\mathcal{L}$.
  Train the nerworks $\mathcal{F}(\cdot)$, $\mathcal{A}(\cdot)$, and $\mathcal{P}(\cdot)$ with $\mathcal{L}$.
end for

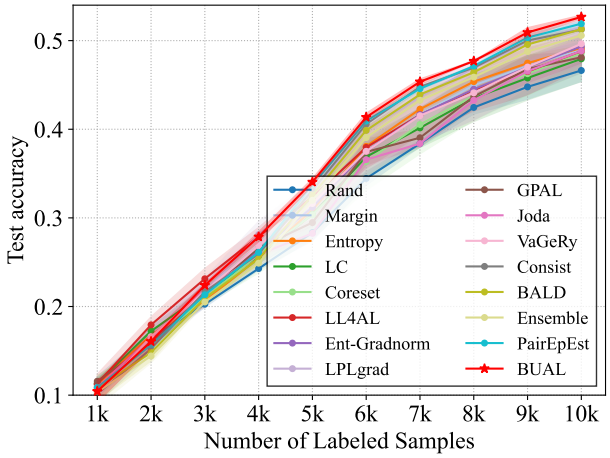
F Implementation Details

To comprehensively evaluate the effectiveness of our proposed method, we conduct experiments on four widely used image classification benchmarks: Cifar10[8], Cifar100[8], SVHN[13], and Tiny-ImageNet[9]. Both Cifar10 and Cifar100 consist of 60000 color images evenly distributed across the classes. Cifar10 contains 10 distinct classes, while Cifar100 contains 100 classes. Specifically, Cifar10 has 6000 samples per class, and Cifar100 has 600 samples per class. All images are 32×32 pixels and depict a variety of objects, including animals, vehicles, and more. Tiny-ImageNet is a reduced subset of ImageNet [3]. It consists of 200 categories, with a total of 120000 color images of size 64×64 pixels. Each category contains 500 training images, 50 validation images, and 50 test images, covering a wide range of visual objects. SVHN is derived from Google Street View images and consists of 10 classes. It contains 73257 images used for training and 26032 images used for testing. A brief summary of these datasets is provided in Table 1.

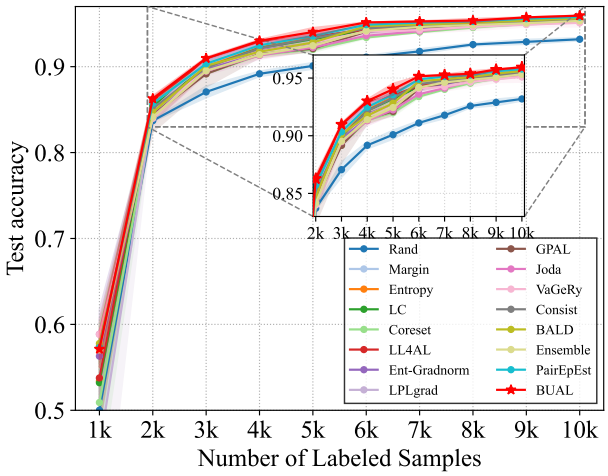
Following [6, 18, 19], we use ResNet-18 as the classification model for all the experiments. We adopt the wavelet package sym17 [10] to implement WWT. The parameter settings of our method and other compared methods are the same as [6]. In more detail, the models are trained for 200 epochs, and the batch size is fixed to 128. For the Cifar10, Cifar100, and SVHN datasets, we use stochastic gradient descent (SGD) as the optimizer with a momentum of 0.9 and a weight decay of 5×10^{-4}. Following [6, 14], for Cifar10 and Cifar100, we set the initial learning rate to 0.1 and reduce it to 0.01 after 160 epochs. For SVHN, the learning rate starts at 0.01 and is decayed to 0.001 after 40 epochs. For the Tiny-ImageNet dataset, we train the model for 100 epochs and adopt the Adam optimizer with a weight decay of 5×10^{-4}, where the learning rate is initialized to 0.001 and decayed by a factor of 0.1 at 60 epochs. All experiments are conducted on a workstation equipped with an Intel i7-12700K CPU and an NVIDIA GeForce RTX 3090 GPU, utilizing the PyTorch framework. Finally, to mitigate the impact of random initialization and ensure reproducibility, all experiments are repeated 5 times with different random seeds, and the average results are reported.



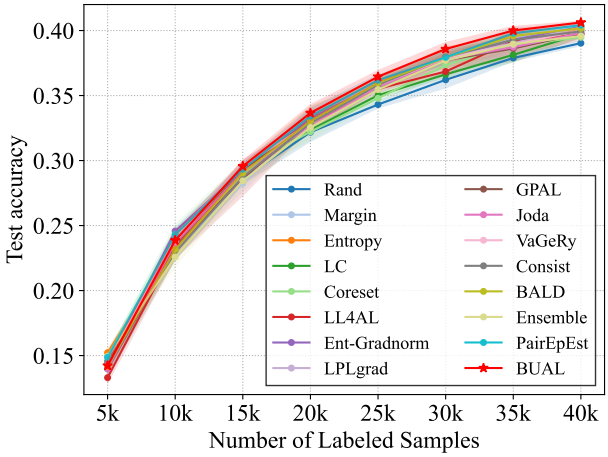
(a) Cifar10 when the labeling budget is 10000.



(b) Cifar100 when the labeling budget is 10000.



(c) SVHN when the labeling budget is 10000.



(d) Tiny-ImageNet when the labeling budget is 40000.

Figure 1: Classification results with ResNet-50 as the backbone on Cifar10, Cifar100, SVHN and Tiny-ImageNet.

Table 1: Summary of CIFAR-10, CIFAR-100, SVHN, and Tiny-ImageNet datasets.

Dataset	Classes	Total Images	Image Size
CIFAR-10	10	60 000	32×32
CIFAR-100	100	60 000	32×32
SVHN	10	99 289	32×32
Tiny-ImageNet	200	120 000	64×64

G Experimental results with ResNet-50 as Backbone

In our experiments (Section 4.2), we use ResNet-18 as the backbone for fair comparison. However, our active learning method can be applied to any other image classification backbones easily. To validate this, in this section, we replace the backbone network with a more widely-used image classification network ResNet-50. The other experimental settings are identical to our previous settings.

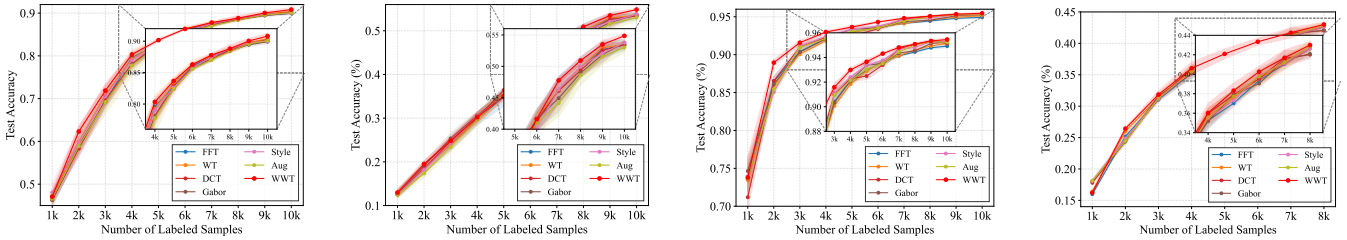
The results are shown in Figure 1. It shows that our method also outperforms other active learning methods at most time on all datasets. For example, in Cifar10, when we annotate 3000 samples (6% samples), our method achieves an average accuracy of 73.90%, surpassing the second-best method by 1.11%. In Cifar100, when we annotate 10000 samples (20% of samples), ours achieves an accuracy of 52.67%, while the second-best one only achieves 51.51%. In SVHN, when annotating 6000 samples (8.19% of samples), our accuracy is 95.15%, and the second-best one is 94.66%. In Tiny-ImageNet, when annotating 30000 samples (30% of samples), our accuracy is 38.58%, and the second-best one is 37.95%. It well demonstrates the effectiveness of our active learning method in ResNet-50.

H Hyperparameter Study

In our adaptive weighting strategy, we define the weight α as $\alpha = \frac{1}{1+\exp^{m-t}}$, where m is a parameter to control when the misleading score has the same weight as the boundary score. In this section, we conduct ablation experiments to investigate the effect of m. We denote T as the total number of the active selection batches, and

Table 2: Ablation Study on the effect of different m in adaptive weighting stratey on Cifar10 (Test accuracy ± standard deviation).

m	1k	2k	3k	4k	5k	6k	7k	8k	9k	10k
$0.1T$	47.39±2.81	61.28±1.39	71.73±1.10	79.72±2.96	79.72±2.96	86.04±0.25	87.55±0.21	88.64±0.14	89.79±0.09	90.52±0.04
$0.2T$	46.11±1.68	58.85±2.37	69.26±2.47	79.28±0.63	82.93±0.22	85.96±0.06	87.45±0.18	88.58±0.03	89.51±0.06	90.16±0.03
$0.3T$	46.14±1.06	58.34±2.91	68.98±3.55	78.79±2.08	82.91±0.27	85.90±0.20	87.44±0.03	88.74±0.05	89.68±0.07	90.29±0.19
$0.4T$	47.92±2.09	61.78±1.68	72.40±2.96	80.54±1.16	83.42±0.86	86.11±0.33	87.65±0.21	88.65±0.21	90.00±0.05	90.65±0.12
$0.5T$	**47.07±1.02**	**62.30±1.32**	**71.87±1.11**	**80.33±0.53**	**83.68±0.34**	**86.29±0.23**	**87.78±0.25**	**88.81±0.62**	**90.04±0.39**	**90.82±0.45**
$0.6T$	45.99±2.53	59.64±3.37	70.28±1.63	70.28±7.63	83.42±0.71	85.97±0.20	87.69±0.04	88.92±0.09	90.04±0.10	90.61±0.14
$0.7T$	46.86±2.34	60.25±2.10	71.38±1.33	79.56±1.75	83.23±0.56	85.92±0.13	85.92±0.13	88.83±0.30	89.92±0.17	90.58±0.03
$0.8T$	47.56±2.57	61.11±2.40	71.04±1.88	80.36±1.01	83.42±0.19	86.12±0.07	86.88±0.15	87.88±0.15	89.75±0.01	90.47±0.05
$0.9T$	48.09±1.46	60.80±2.15	70.13±1.60	79.02±0.78	83.33±0.15	86.34±0.21	87.92±0.06	88.98±0.07	89.92±0.07	90.64±0.20



(a) Cifar10 when the labeling budget is 10000. **(b) Cifar100 when the labeling budget is 10000.** **(c) SVHN when the labeling budget is 10000.** **(d) Tiny-ImageNet when the labeling budget is 40000.**

Figure 2: Classification results with different number of annotations on Cifar10, Cifar100, SVHN, and Tiny-ImageNet.

show the test accuracy on Cifar10 dataset with different m in Table 2. We show the results when m changes from $0.1T$ to $0.9T$. From Table 2, we can see that model performs best when $m = 0.5T$. That is why we fix $m = 0.5T$ in our implementation. In the early training stages ($t < m = 0.5T$), a higher weight is assigned to the boundary score, enabling rapid performance improvement. In the middle of training, i.e., $t = m = 0.5T$, the boundary score and misleading score have the same weight. In later stages ($t > m = 0.5T$), the misleading score will have a larger weight, encouraging the selection of misleading samples that the model fails to classify correctly.

I Comparison with Other Frequency-domain Transform

In this paper, we apply the weighted wavelet transform (WWT) to transfer the data into the frequency domain. In this section, we replace it with some other widely used transformation, such as the Fast Fourier Transform (FFT), the conventional Wavelet Transform (WT), Discrete Cosine Transform (DCT), Gabor Transform (Gabor), style transformation (Style [7]), and data augmentation (Aug [2]), to validate the effects of WWT. Figure 2 shows the classification results on cifar10, cifar100, SVHN and Tiny-ImageNet datasets. It can be seen that, at most times, our WWT outperforms other transformation methods, demonstrating the effectiveness of WWT. That is why we use WWT in our method.

J Efficiency Results

We compare the training time of our method with that of the baselines in Figure 3. Note that in this experiment, we count the data selection time as part of the training time. In Cifar10, we run each method for 10 selection batches (i.e., annotation budget varies from 2% to 20% with an incremental size of 2%).

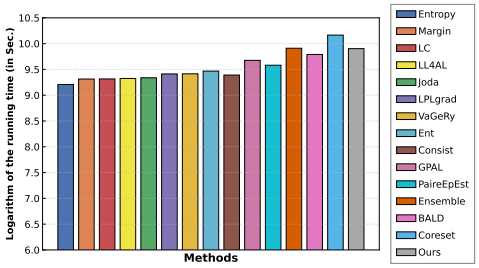


Figure 3: Training time (in Sec.) of the AL methods on the Cifar10 datasets. All the results are measured on a Nvidia RTX 3090 GPU with a 12th Gen Intel Core i7-12700K CPU.

In Figure 3, we report the logarithm of the running time (in seconds) for the evaluated methods. We can see that the Coreset method [15] is the slowest method. It requires solving the K-center problem iteratively during sample selection, which inherently introduces a time overhead. With the inconsistency-based baselines, we observe that methods relying on multiple models or multiple forward passes, such as BALD [4] and Ensemble [1], also exhibit relatively high computational costs. In this context, the running time of our method is comparable to these advanced baselines, and is even slightly more efficient than the Ensemble method. The computational cost of our approach primarily stems from the current implementation, which needs to train two network architectures

(i.e., the spatial-domain main model and the frequency-domain auxiliary model) serially. While this inevitably incurs an additional time cost compared to lightweight single-modal methods, it is worth noting that the computations of these two models are strictly decoupled and isolated during both the training and sample selection phases. Therefore, in practical deployments, their execution can be easily parallelized (e.g., distributed across multi-GPU setups). With parallel computing, the actual training time of our method could be significantly reduced, improving its efficiency while retaining the powerful bi-modal analytical capability.

References

[1] William H Beluch, Tim Genewein, Andreas Nürnberger, and Jan M Köhler. 2018. The power of ensembles for active learning in image classification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 9368–9377.

[2] Ekin D Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V Le. 2020. Randaugment: Practical automated data augmentation with a reduced search space. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*. 702–703.

[3] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*. Ieee, 248–255.

[4] Yarin Gal, Riashat Islam, and Zoubin Ghahramani. 2017. Deep bayesian active learning with image data. In *International conference on machine learning*. PMLR, 1183–1192.

[5] Benyamin Ghojogh and Mark Crowley. 2019. Linear and Quadratic Discriminant Analysis: Tutorial. arXiv:1906.02590 [stat.ML] https://arxiv.org/abs/1906.02590

[6] Shreen Gul, Mohamed Elmahallawy, Sanjay Madria, and Ardhendu Tripathy. 2024. LPLgrad: Optimizing Active Learning Through Gradient Norm Sample Selection and Auxiliary Model Training. In *2024 IEEE International Conference on Big Data (BigData)*. IEEE, 900–909.

[7] Philip TG Jackson, Amir Atapour Abarghouei, Stephen Bonner, Toby P Breckon, and Boguslaw Obara. 2019. Style augmentation: data augmentation via style randomization.. In *CVPR workshops*, Vol. 6. 10–11.

[8] Alex Krizhevsky, Geoffrey Hinton, et al. 2009. Learning multiple layers of features from tiny images.(2009).

[9] Yann Le, Xuan Yang, et al. 2015. Tiny imagenet visual recognition challenge. *CS 231N* 7, 7 (2015), 3.

[10] Gregory Lee, Ralf Gommers, Filip Waselewski, Kai Wohlfahrt, and Aaron O'Leary. 2019. PyWavelets: A Python package for wavelet analysis. *Journal of Open Source Software* 4, 36 (2019), 1237.

[11] Adrian S Lewis and G Knowles. 1992. Image compression using the 2-D wavelet transform. *IEEE Transactions on image Processing* 1, 2 (1992), 244–250.

[12] Waleed A Mahmoud, Ahmed S Hadi, and Talib M Jawad. 2012. Development of a 2-D Wavelet Transform based on Kronecker Product. *Al-Nahrain Journal of Science* 15, 4 (2012), 208–213.

[13] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Baolin Wu, Andrew Y Ng, et al. 2011. Reading digits in natural images with unsupervised feature learning. In *NIPS workshop on deep learning and unsupervised feature learning*, Vol. 2011. Granada, 7.

[14] Sebastian Schmidt, Leonard Schenk, Leo Schwinn, and Stephan Günnemann. 2025. Joint out-of-distribution filtering and data discovery active learning. In *Proceedings of the Computer Vision and Pattern Recognition Conference*. 25677–25687.

[15] Ozan Sener and Silvio Savarese. 2017. Active learning for convolutional neural networks: A core-set approach. *arXiv preprint arXiv:1708.00489* (2017).

[16] Marvin K Simon. 2002. *Probability distributions involving Gaussian random variables: A handbook for engineers and scientists*. Springer.

[17] Jinyu Tian, Jiantao Zhou, Yuanman Li, and Jia Duan. 2021. Detecting adversarial examples from sensitivity inconsistency of spatial-transform domain. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 35. 9877–9885.

[18] Tianyang Wang, Xingjian Li, Pengkun Yang, Guosheng Hu, Xiangrui Zeng, Siyu Huang, Cheng-Zhong Xu, and Min Xu. 2022. Boosting active learning via improving test performance. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 36. 8566–8574.

[19] Donggeun Yoo and In So Kweon. 2019. Learning loss for active learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 93–102.