AdaShift: Learning Discriminative Self-Gated Neural Feature Activation With an Adaptive Shift Factor

SUPPLEMENTARY DOCUMENT

Sudong Cai
Graduate School of Informatics, Kyoto University
cai.sudong.t94@kyoto-u.jp

A. Diagrams of AdaShift-MA Derivatives

In Section 4.3 (Ablation Study), we validated the extensibility of our AdaShift prototype by presenting three new extended practical AdaShift derivatives based on AdaShift-MA, which we refer to as AdaShift-MA-N1, AdaShift-MA-N2, and AdaShift-MA-N3, respectively. In this Appendix, we depict the operational diagrams of AdaShift-MA-N1 and AdaShift-MA-N2 in Figures 1 and 2, correspondingly. Note that AdaShift-MA-N3 is a combined strategy of AdaShift-MA-N1 and AdaShift-MA-N2, *i.e.*, it employs AdaShift-MA-N1 at the nodes that converge both the expanded main and residual features and applies AdaShift-MA-N2 to process unexpanded features only.

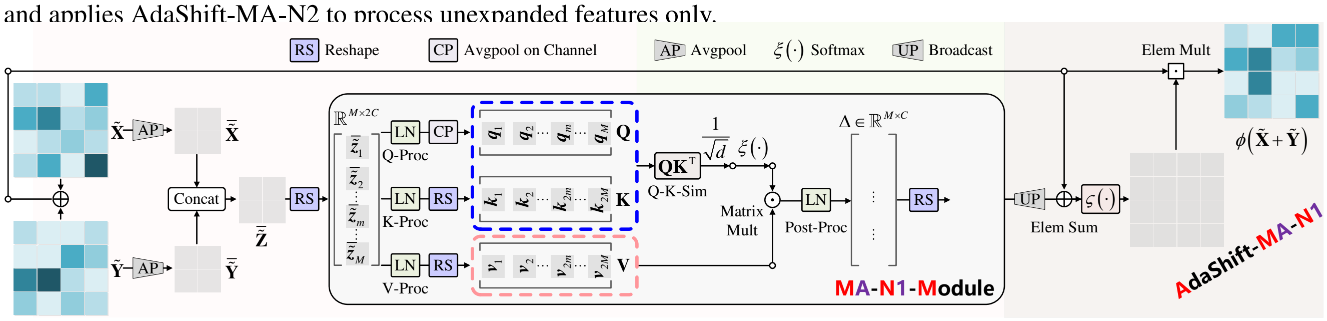

Figure 1. Illustration of AdaShift-MA-N1. $M = \lceil H/K_H \rceil \cdot \lceil L/K_L \rceil$. $\bar{\tilde{\mathbf{Y}}} \in \mathbb{R}^{C\times\lceil H/K_H\rceil\times\lceil L/K_L\rceil}$ denotes the residual feature map; $\bar{\tilde{\mathbf{Z}}} = \left[\bar{\tilde{\mathbf{X}}}; \bar{\tilde{\mathbf{Y}}}\right] \in \mathbb{R}^{2C\times\lceil H/K_H\rceil\times\lceil L/K_L\rceil}$ is produced by concatenating the channels of $\bar{\tilde{\mathbf{X}}}$ and $\bar{\tilde{\mathbf{Y}}}$. "Elem" denotes "Element-wise" and "Mult" denotes "Multiplication." Note that AdaShift-MA-N1 is only applicable to the nodes that converge both the main and residual features, otherwise regresses to AdaShift-MA.

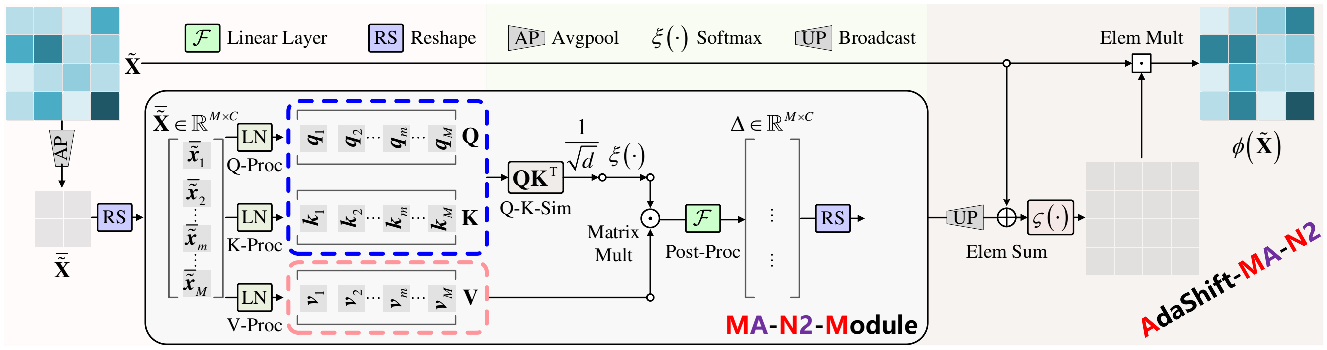

Figure 2. Illustration of AdaShift-MA-N2. $M = \lceil H/K_H \rceil \cdot \lceil L/K_L \rceil$. "Elem" denotes "Element-wise" and "Mult" denotes "Multiplication." Note that AdaShift-MA-N2 is only applied to the layers that process unexpanded features (*e.g.*, the second layer of a bottleneck residual block [9]) to avoid bringing excessive parameters.

B. Discussion of AdaShift: from MCDM-inspired Intuitions and Properties

In this Appendix, we discuss the attributes of AdaShift-B (as the representative of the practical AdaShift family) in light of the basic intuitions and assumed properties of neural activation inspired by the MCDM hypothesis [3]. Below we demonstrate that AdaShift-B is consistent with the basic MCDM-inspired intuitions and holds the corresponding (assumed) properties.

Retrospect.

Intuition 1. Let $\varrho\left(\tilde{x}\right)$ denote an unknown ideal similarity measure that reflects the unbiased feature importance score of the given feature vector $\boldsymbol{x}$ *w.r.t.* the given filter (vector) $\boldsymbol{w}$. Let $\hat{\varrho}\left(\left(\tilde{x}\right)\right.$ denote an updatable term that learns to gradually approximate $\varrho\left(\tilde{x}\right)$ in the training phase, namely, the *approximated similarity*.

Then, it is expected that:

Property 1. (The Directional Monotonicity $\wedge$ Sign Constraint of $\varsigma\left(\varrho_x\right)$ About ϱ_x) $\left|\varsigma\left(\varrho\left(\tilde{x}\right)\right)\right| \geqslant \left|\varsigma\left(\varrho\left(\tilde{y}\right)\right)\right|$ if $\varrho\left(\tilde{x}\right) \geqslant \varrho\left(\tilde{y}\right)$.

Particularly, Property 1 can be ensured by the simple condition

Proposition 1. *Property 1 $\Longleftrightarrow$ (1) $\varsigma\left(\varrho_x\right)$ is (monotonically) non-decreasing about ϱ_x $\wedge$ $\varsigma\left(\varrho_x\right) \geqslant 0$ $\vee$ (2) $\varsigma\left(\varrho_x\right)$ is (monotonically) non-increasing about ϱ_x $\wedge$ $\varsigma\left(\varrho_x\right) \leqslant 0$ (ϱ_x denotes $\varrho\left(\tilde{x}\right)$; $\wedge$ denotes "and;" $\vee$ denotes "or").*

Following we show that AdaShift-B satisfies holds Property 1.

Discussion. First, AdaShift-B, *i.e.*, $\phi\left(\tilde{x}\right) = \rho\left(\tilde{x}\right)\tilde{x} = \varsigma\left(\tilde{x} + \Delta\right)\tilde{x}$, learns to approximate the ideal similarity measure $\varrho\left(\tilde{x}\right)$ of the input $\tilde{x}$ by $\hat{\varrho}_x = \hat{\varrho}\left(\tilde{x}\right) = \tilde{x} + \Delta$, where the adaptive shift factor Δ is defined in Sec. 3.3 and the adjuster ς is an vanilla Sigmoid function of $\hat{\varrho}_x$, *i.e.*, $\varsigma\left(\hat{\varrho}_x\right) = \frac{\hat{\varrho}_x}{1+\hat{\varrho}_x}$. Therefore, $\forall \hat{\varrho}_x \in \mathbb{R}$, $\varsigma\left(\hat{\varrho}_x\right)$ is monotonically non-decreasing about $\hat{\varrho}_x$ $\wedge$ $\varsigma\left(\hat{\varrho}_x\right) \geqslant 0$. That is, AdaShift-B (with a Sigmoid adjuster) naturally satisfies the assumed conditions of Proposition 1. ***This ensures Proposition 1 for AdaShift-B***.

Intuition 2. ***CNI***: Any negative candidate is expected to be assigned with a limited weight to constrain its influence.

Property 2. ***(CNI)***

1. Basic case: $\exists \eta \in \mathbb{R}, \mathcal{M}_{x^-} \geqslant 0$ such that $\forall \varrho\left(\tilde{x}\right) < \eta$, we have (1) $\left|\varsigma\left(\varrho\left(\tilde{x}\right)\right)\right| \big|_{\varrho(\tilde{x})<\eta} \leqslant \mathcal{M}_{x^-}$; (2) Especially, we expect $\lim_{\varrho(\tilde{x})\to-\infty} \left|\varsigma\left(\varrho\left(\tilde{x}\right)\right)\right| = 0$.
2. Strict case: $\exists \eta \in \mathbb{R}, \mathcal{M}_{x^-} \geqslant 0$ such that $\forall \varrho\left(\tilde{x}\right) < \eta$, we have $\left|\rho\left(\tilde{x}\right)\tilde{x}\right| \big|_{\varrho(\tilde{x})<\eta} \leqslant \mathcal{M}_{x^-}$ (with the auxiliary conditions: $\varrho\left(\tilde{x}\right)$ is (a) (uniformly) continuous about $\tilde{x}$ on the domain; (b) differentiable about $\tilde{x}$ or at most has a finite number of points where the left- and right-hand limits exist but are unequal).

Following we show that AdaShift-B holds Property 2.

Discussion. The core spirit of Property 2 is to selectively constrain the influence of non-important (negative) alternative candidates (*i.e.*, features). We first analyze a simple case and then generalize it to an extended case.

Simple case. In the simple case, we take into account the re-scaling effect of the Z-Scoring of LayerNorm. That is, Δ (*i.e.*, Δ_c, which we specified as in the subsequent text for clarity) can be expressed by:

$$\Delta_c = \left[\mathrm{LN}\left(\mathrm{avgpool}_{H\times L}\left(\tilde{\mathbf{X}}\right)\right)\right]_c = \mathrm{LN}\left(\bar{\tilde{x}}_c\right) = \dot{\gamma}_c \frac{\bar{\tilde{x}}_c - \mu_{\bar{x}}}{\delta_{\bar{x}}} + \dot{\beta}_c\,, \tag{1}$$

where $\mu_{\bar{x}}$ and $\delta_{\bar{x}}$ denote the mean value and (unbiased estimation of) standard deviation of channel mean vector (*i.e.*, $\bar{\tilde{\boldsymbol{x}}}$, where $\bar{\tilde{x}}_c$ is the c-th element of $\bar{\tilde{\boldsymbol{x}}}$), respectively. $\dot{\gamma}_c$ and $\dot{\beta}_c$ denote the channel-wise scaling and shift factors (applied to introduce the parametric element-wise affine of LayerNorm), respectively.

Then, for $\hat{\varrho}_x \to -\infty$ (*i.e.*, $\tilde{x}_c + \Delta_c - \infty$), only two cases are possible, *i.e.*, (1) $\tilde{x}_c \to -\infty$; (2) $\tilde{x}_c \to +\infty$. Note that here we solely discuss the function of the concerned input, *i.e.*, $\tilde{x}_c$, so other terms (*e.g.*, $\tilde{x}_i, i \neq c$) are treated as known values.

Below we first consider the case (1) and generalize the deduced conclusion to the case (2). For case (1), we have:

$$\lim_{\tilde{x}_c\to-\infty} \mu_{\bar{x}} = \lim_{\tilde{x}_c\to-\infty} \bar{\tilde{x}}_c - \frac{1}{C}\bar{\tilde{x}}_c - \frac{1}{C}\sum_{i=1,i\neq c}^{C} \bar{\tilde{x}}_i = \frac{C-1}{C}\bar{\tilde{x}}_c - \frac{1}{C}\sum_{i=1,i\neq c}^{C} \bar{\tilde{x}}_i\,, \tag{2}$$

so that we only need to consider the term $\frac{C-1}{C}\bar{\tilde{x}}_c$ and denote this term as $\kappa_\mu \tilde{x}_c$, where $\lim_{\tilde{x}_c\to-\infty} \kappa_\mu = \frac{C-1}{C\cdot N}$, where $N = H\cdot L$ denotes the total number of pixel locations of the current feature map. Therefore, κ_μ is a finite value.

Similarly, for $\delta_{\bar{x}}$, we have:

$$\begin{aligned}
\lim_{\tilde{x}_c\to-\infty}\delta_{\bar{x}} &= \lim_{\tilde{x}_c\to-\infty}\sqrt{\frac{\sum\limits_{i=1}^{C}\left(\bar{\tilde{x}}_i-\mu_{\bar{x}}\right)^2}{C-1}} = \lim_{\tilde{x}_c\to-\infty}\frac{\sqrt{\left(\bar{\tilde{x}}_c-\bar{\tilde{x}}_c\right)+\sum\limits_{i=1,i\neq c}^{C}\left(\bar{\tilde{x}}_i-\mu_{\bar{x}}\right)^2}}{\sqrt{C-1}} \\
&= \lim_{\tilde{x}_c\to-\infty}\frac{\sqrt{\sum\limits_{i=1,i\neq c}^{C}\left(\bar{\tilde{x}}_i-\mu_{\bar{x}}\right)^2}}{\sqrt{C-1}} = \lim_{\tilde{x}_c\to-\infty}\frac{\sqrt{\sum\limits_{i=1,i\neq c}^{C}\left(\left(\kappa_\mu\tilde{x}_c\right)^2-2\bar{\tilde{x}}_i\left(\kappa_\mu\tilde{x}_c\right)+\bar{\tilde{x}}_i^2\right)}}{\sqrt{C-1}} \\
&\to \frac{1}{\sqrt{C-1}}\sqrt{\sum_{i=1,i\neq c}^{C}\left(\left(\kappa_\mu\tilde{x}_c\right)^2-2\bar{\tilde{x}}_i\left(\kappa_\mu\tilde{x}_c\right)\right)}\,. \qquad (3)
\end{aligned}$$

Note that as $\delta_{\bar{x}}$ is also essentially a first-order value of $\tilde{x}_c$ (like $\mu_{\bar{x}}$) and only $\left(\kappa_\mu\tilde{x}_c\right)^2$ is the first-order value of $\tilde{x}_c$ in $\delta_{\bar{x}}$, we can also represent it by $\kappa_\sigma\tilde{x}_c$, where

$$\begin{aligned}
\lim_{\tilde{x}_c\to-\infty}\left|\kappa_\sigma\right| &= \lim_{\tilde{x}_c\to-\infty}\frac{\frac{1}{\sqrt{C-1}}\sqrt{\left(\bar{\tilde{x}}_c-\mu_{\bar{x}}\right)^2+\sum_{i=1,i\neq c}^{C}\left(\left(\kappa_\mu\tilde{x}_c\right)^2-2\bar{\tilde{x}}_i\left(\kappa_\mu\tilde{x}_c\right)+\bar{\tilde{x}}_i^2\right)}}{\left|\tilde{x}_c\right|} \\
&= \lim_{\tilde{x}_c\to-\infty}\frac{\frac{1}{\sqrt{C-1}}\sqrt{\left(\bar{\tilde{x}}_c-\mu_{\bar{x}}\right)^2+\sum_{i=1,i\neq c}^{C}\left(\kappa_\mu\tilde{x}_c\right)^2}}{\left|\tilde{x}_c\right|} \\
&= \lim_{\tilde{x}_c\to-\infty}\frac{\frac{1}{\sqrt{C-1}}\sqrt{\left(\bar{\tilde{x}}_c-\mu_{\bar{x}}\right)^2+\left(C-1\right)\left(\kappa_\mu\tilde{x}_c\right)^2}}{\left|\tilde{x}_c\right|} \\
&= \lim_{\tilde{x}_c\to-\infty}\frac{1}{\sqrt{C-1}}\sqrt{\frac{\left(\bar{\tilde{x}}_c-\mu_{\bar{x}}\right)^2+\left(C-1\right)\left(\kappa_\mu\tilde{x}_c\right)^2}{\tilde{x}_c^2}} \\
&= \lim_{\tilde{x}_c\to-\infty}\frac{1}{\sqrt{C-1}}\sqrt{\left(\frac{1}{N}-\kappa_\mu\right)^2+\left(C-1\right)\left(\kappa_\mu\right)^2} \\
&= \sqrt{\frac{\left(\frac{1}{N}-\kappa_\mu\right)^2}{C-1}+\kappa_\mu^2}\,. \qquad (4)
\end{aligned}$$

That is, we can re-write Δ_c when $\tilde{x}_c\to-\infty$ as:

$$\lim_{\tilde{x}_c\to-\infty}\Delta_c = \dot{\gamma}_c\frac{\frac{1}{N}\tilde{x}_c-\kappa_\mu\tilde{x}_c}{-\kappa_\sigma\tilde{x}_c}+\dot{\beta}_c = \dot{\gamma}_c\frac{\frac{1}{N}\tilde{x}_c-\kappa_\mu\tilde{x}_c}{-\kappa_\sigma\tilde{x}_c}+\dot{\beta}_c = -\dot{\gamma}_c\frac{\frac{1}{N}-\kappa_\mu}{\kappa_\sigma}+\dot{\beta}_c\,, \qquad (5)$$

which regresses to a known value that consists of N, κ_μ, κ_σ, $\dot{\gamma}_c$, and $\dot{\beta}_c$.

With the above deduced conclusion, we have:

$$\begin{aligned}
\lim_{\hat{\varrho}(\tilde{x})\to-\infty}\left(\varsigma\left(\hat{\varrho}\left(\tilde{x}\right)\right)\tilde{x}\right) &= \lim_{(\tilde{x}+\Delta)\to-\infty}\mathrm{sigmoid}\left(\tilde{x}+\Delta\right)\tilde{x} = \lim_{(\tilde{x}+\Delta)\to-\infty}\frac{e^{\tilde{x}+\Delta}}{e^{\tilde{x}+\Delta}+1}\cdot\tilde{x} \\
&= \lim_{z\to+\infty}\frac{e^{-z}}{e^{-z}+1}\cdot\left(-z-\Delta\right)|_{z=-(\tilde{x}+\Delta)} = \lim_{z\to+\infty}\frac{-\frac{z+\Delta}{e^z}}{e^{-z}+1} \\
&= \frac{0}{0+1} = 0\,. \qquad (6)
\end{aligned}$$

This conclusion can be simply generalized to the case (2) (*i.e.*, $\tilde{x}_c\to+\infty$).

Therefore, AdaShift-B holds (the strict case of) Property 2.

Extended case. For more generality, we discuss an extended case where

$$\Delta_c = \dot{\gamma}_c\bar{\tilde{x}}_c+\dot{\beta}_c\,. \qquad (7)$$

That is, the Z-Scoring which regresses Δ_c to a known value is removed to relax the value constraint of Δ_c.

With the above condition, similar to the simple case, $\hat{\varrho}\left(\tilde{x}\right) \to -\infty$ is only possible for (1) $\tilde{x}_c \to -\infty$ and (2) $\tilde{x}_c \to +\infty$. Then, for $\tilde{x}_c \to -\infty$, we have:

$$\begin{aligned}
\lim_{\hat{\varrho}(\tilde{x}) \to -\infty} \left(\varsigma\left(\hat{\varrho}\left(\tilde{x}\right)\right)\tilde{x}\right) &= \lim_{(\tilde{x}+\Delta)\to-\infty} \mathrm{sigmoid}\left(\tilde{x}+\Delta\right)\tilde{x} \\
&= \lim_{(\tilde{x}+\Delta)\to-\infty} \frac{e^{\tilde{x}+\Delta}}{e^{\tilde{x}+\Delta}+1}\tilde{x} \\
&= \lim_{\left(\left(1+\frac{\dot{\gamma}_c}{N}\right)\tilde{x}_c+\dot{\beta}_c\right)\to-\infty} \frac{e^{\left(1+\frac{\dot{\gamma}_c}{N}\right)\tilde{x}_c+\dot{\beta}_c}}{e^{\left(1+\frac{\dot{\gamma}_c}{N}\right)\tilde{x}_c+\dot{\beta}_c}+1}\cdot\tilde{x} \\
&= \lim_{z\to+\infty} \frac{e^{-z}}{e^{-z}+1}\cdot -\frac{z+\dot{\beta}_c}{1+\frac{\dot{\gamma}_c}{N}} \Big|_{z=-\left(\left(1+\frac{\dot{\gamma}_c}{N}\right)\tilde{x}_c+\dot{\beta}_c\right)} \\
&= \lim_{z\to+\infty} \left(\frac{-N}{N+\dot{\gamma}_c}\cdot\frac{\frac{z+\dot{\beta}_c}{e^{z}}}{e^{-z}+1}\right) \\
&= \frac{-N}{N+\dot{\gamma}_c}\cdot\frac{0}{0+1} \\
&= 0\,.
\end{aligned} \tag{8}$$

This ensure that the (the strict case of) Property 2 holds for the extended case (of AdaShift-B).

Intuition 3. ***PPI***: Any two important candidates with close importance are expected to be assigned with comparable weights to ensure comparable influence, *i.e.*, the relatively more important candidate will not cover the influence of another.

Property 3. ***(PPI)***

1. Basic case: $\exists\eta\in\mathbb{R}, \mathcal{M}_{x^+}\geqslant 0$ such that $\forall\varrho\left(\tilde{x}\right)>\eta$ we have $\left|\varsigma\left(\varrho\left(\tilde{x}\right)\right)\right|\,|_{\varrho(\tilde{x})>\eta}\leqslant\mathcal{M}_{x^+}$.
2. Strict case: $\exists\eta\in\mathbb{R}, \mathcal{M}_{x^+}\geqslant 0$, such that $\forall\varrho\left(\tilde{x}\right)>\eta$ we have $\left|\nabla_{\tilde{x}}\left(\rho\left(\tilde{x}\right)\tilde{x}\right)\right|\,|_{\varrho(\tilde{x})>\eta}\leqslant\mathcal{M}_{x^+}$ at any $\tilde{x}$ where $\phi\left(\tilde{x}\right)$ is differentiable (with the auxiliary conditions: $\varrho\left(\tilde{x}\right)$ is (a) (uniformly) continuous about $\tilde{x}$ on the domain; (b) differentiable about $\tilde{x}$ or at most has a finite number of points where the left- and right-hand limits exist but are unequal).

Following we show that AdaShift-B holds Property 3.

Discussion. The core spirit of Property 3 is to preserve the positive influence of different important alternative candidates (*i.e.*, features). We analyze this based on the simple case and the extended case of Δ_c introduced in the above discussion of Property 2.

In particular, Property 3 can be satisfied by the *Lipschitz continuity* of the overall activation function $\phi(\tilde{x}_c)$ according to its definition. Therefore, below we confirm the Lipschitz continuity of $\phi(\tilde{x}_c)$ of AdaShift-B.

First, the (partial) derivative of AdaShift-B about the concerned input $\tilde{x}_c$ can be calculated by:

$$\begin{aligned}
\nabla_{\tilde{x}_c}\phi\left(\tilde{x}_c\right) &= \frac{\partial\left(\mathrm{sigmoid}\left(\tilde{x}_c+\Delta\right)\cdot\tilde{x}_c\right)}{\partial\tilde{x}_c} \\
&= \frac{\partial\mathrm{sigmoid}\left(\tilde{x}_c+\Delta\right)}{\partial\tilde{x}_c}\cdot\frac{\partial\left(\tilde{x}_c+\Delta\right)}{\partial\tilde{x}_c}\cdot\tilde{x}_c+\mathrm{sigmoid}\left(\tilde{x}_c+\Delta\right)\,.
\end{aligned} \tag{9}$$

For the simple case where Z-Scoring is applied, Δ_c is always a bounded value that has a weak relationship to the input $\tilde{x}_c$ (*i.e.*, can be regarded as a zero-order value of $\tilde{x}_c$). Therefore, we can treat Δ_c as a known value in the calculation of $\nabla_{\tilde{x}_c}\phi\left(\tilde{x}_c\right)$, *i.e.*,

$$\begin{aligned}
\nabla_{\tilde{x}_c}\phi\left(\tilde{x}_c\right) &= \frac{\partial\mathrm{sigmoid}\left(\tilde{x}_c+\Delta\right)}{\partial\tilde{x}_c}\cdot\tilde{x}_c+\mathrm{sigmoid}\left(\tilde{x}_c+\Delta\right) \\
&= \frac{e^{\tilde{x}+\Delta}}{e^{\tilde{x}+\Delta}+1}\cdot\left(1-\frac{e^{\tilde{x}+\Delta}}{e^{\tilde{x}+\Delta}+1}\right)\cdot\tilde{x}_c+\mathrm{sigmoid}\left(\tilde{x}_c+\Delta\right) \\
&= \frac{\tilde{x}_c\cdot e^{\tilde{x}+\Delta}}{\left(e^{\tilde{x}+\Delta}+1\right)^2}+\mathrm{sigmoid}\left(\tilde{x}_c+\Delta\right) \\
&< \frac{\tilde{x}_c\cdot e^{\tilde{x}+\Delta}}{\left(e^{\tilde{x}+\Delta}+1\right)^2}+1\,.
\end{aligned} \tag{10}$$

So, the boundedness of $\nabla_{\tilde{x}_c}\phi\left(\tilde{x}_c\right)$ is only possibly violated when $\tilde{x}_c \to +\infty$. But, as

$$\begin{aligned}\lim_{\tilde{x}\to+\infty} \nabla_{\tilde{x}_c}\phi\left(\tilde{x}_c\right) &= \lim_{\tilde{x}\to+\infty}\left(\frac{\tilde{x}_c \cdot e^{\tilde{x}+\Delta}}{\left(e^{\tilde{x}+\Delta}+1\right)^2}+1\right) = \lim_{\tilde{x}\to+\infty}\left(\frac{\tilde{x}_c}{e^{\tilde{x}+\Delta}+1}\cdot\frac{e^{\tilde{x}+\Delta}}{e^{\tilde{x}+\Delta}+1}+1\right)\\ &= 0\cdot 1+1 = 1\,.\end{aligned} \tag{11}$$

Therefore, $\nabla_{\tilde{x}_c}\phi\left(\tilde{x}_c\right)$ is bounded on the whole domain, *i.e.*, $\phi(\tilde{x}_c)$ of AdaShift-B holds Lipschitz continuity. ***This ensures that Property 3 holds for AdaShift-B***.

Similarly, for the extended case, we have:

$$\begin{aligned}\nabla_{\tilde{x}_c}\phi\left(\tilde{x}_c\right) &= \frac{\partial \text{sigmoid}\left(\tilde{x}_c+\Delta\right)}{\partial\tilde{x}_c}\cdot\frac{\partial\left(\tilde{x}_c+\Delta\right)}{\partial\tilde{x}_c}\cdot\tilde{x}_c + \text{sigmoid}\left(\tilde{x}_c+\Delta\right)\\ &< \frac{\partial\text{sigmoid}\left(\left(1+\frac{\dot{\gamma}_c}{N}\right)\tilde{x}_c+\dot{\beta}_c\right)}{\partial\tilde{x}_c}\cdot\left(1+\frac{\dot{\gamma}_c}{N}\right)\cdot\tilde{x}_c+1\\ &= \frac{e^{\left(1+\frac{\dot{\gamma}_c}{N}\right)\tilde{x}_c+\dot{\beta}_c}}{e^{\left(1+\frac{\dot{\gamma}_c}{N}\right)\tilde{x}_c+\dot{\beta}_c}+1}\cdot\left(1-\frac{e^{\left(1+\frac{\dot{\gamma}_c}{N}\right)\tilde{x}_c+\dot{\beta}_c}}{e^{\left(1+\frac{\dot{\gamma}_c}{N}\right)\tilde{x}_c+\dot{\beta}_c}+1}\right)\cdot\left(1+\frac{\dot{\gamma}_c}{N}\right)\cdot\tilde{x}_c+1\\ &= \frac{\left(1+\frac{\dot{\gamma}_c}{N}\right)\cdot\tilde{x}_c}{e^{\left(1+\frac{\dot{\gamma}_c}{N}\right)\tilde{x}_c+\dot{\beta}_c}+1}\cdot\frac{e^{\left(1+\frac{\dot{\gamma}_c}{N}\right)\tilde{x}_c+\dot{\beta}_c}}{e^{\left(1+\frac{\dot{\gamma}_c}{N}\right)\tilde{x}_c+\dot{\beta}_c}+1}+1\,.\end{aligned} \tag{12}$$

Note that $|\dot{\gamma}_c|$ is a small value in common due to the $\mathcal{L}_2$ regularization applied on learnable parameters and N is typically a relatively big value (*i.e.*, $|\dot{\gamma}_c| \ll N$). That is, we have:

$$\begin{aligned}\lim_{\tilde{x}\to+\infty}\nabla_{\tilde{x}_c}\phi\left(\tilde{x}_c\right) &= \lim_{\tilde{x}\to+\infty}\left(\frac{\partial\text{sigmoid}\left(\left(1+\frac{\dot{\gamma}_c}{N}\right)\tilde{x}_c+\dot{\beta}_c\right)}{\partial\tilde{x}_c}\cdot\left(1+\frac{\dot{\gamma}_c}{N}\right)\cdot\tilde{x}_c+1\right)\\ &= \lim_{\tilde{x}\to+\infty}\left(\frac{\left(1+\frac{\dot{\gamma}_c}{N}\right)\cdot\tilde{x}_c}{e^{\left(1+\frac{\dot{\gamma}_c}{N}\right)\tilde{x}_c+\dot{\beta}_c}+1}\cdot\frac{e^{\left(1+\frac{\dot{\gamma}_c}{N}\right)\tilde{x}_c+\dot{\beta}_c}}{e^{\left(1+\frac{\dot{\gamma}_c}{N}\right)\tilde{x}_c+\dot{\beta}_c}+1}+1\right)\\ &= 0\cdot 1+1\\ &= 1\,.\end{aligned} \tag{13}$$

Therefore, $\nabla_{\tilde{x}_c}\phi\left(\tilde{x}_c\right)$ is also bounded, which ensures the Lipschitz continuity of $\phi(\tilde{x}_c)$ of AdaShift-B for the extended case. ***This generalizes Property 3 of AdaShift-B for the extended case***.

Intuition 4. ***OD***: The assigned weights are expected to differentiate the positive and the negative candidates (while avoiding gradient and feature vanishing or explosion).

Property 4. ***(OD)*** $\exists\eta\in\mathbb{R}$ and $\exists\epsilon_\rho,\delta_\rho>0$ such that if $\varrho\left(\tilde{x}\right)>\eta>\varrho\left(\tilde{y}\right)$, then, $\forall\varrho\left(\tilde{x}\right)-\varrho\left(\tilde{y}\right)>\epsilon_\rho$ we have $\varsigma\left(\varrho\left(\tilde{x}\right)\right)-\varsigma\left(\varrho\left(\tilde{y}\right)\right)>\delta_\rho$. (Note that δ_ρ is big enough to prevent gradient and feature vanishing)

Discussion. The core spirit of Property 4 is to prevent the gradient and feature vanishing after neural activation. It expects the space of the difference of the upper-bound and lower-bound of a re-weighting function to be adequate.

Note that Property 4 is a relaxed constraint since the trainable parameters of a (recent) neural network layer (*e.g.*, linear layer and parametric normalization layer) are capable of providing an extent of flexibility to the (intensities of) features.

For AdaShift-B, where $\rho\left(\tilde{x}\right)=\text{sigmoid}\left(\tilde{x}+\Delta\right)\in(0,1)$, it is easy to satisfy the condition of Property 4, as

$$\lim_{(\tilde{x}+\Delta)\to+\infty}\varsigma\left(\tilde{x}+\Delta\right)-\lim_{(\tilde{x}+\Delta)\to-\infty}\varsigma\left(\tilde{x}+\Delta\right)=1-0=1\,. \tag{14}$$

This shows that AdaShift-B holds Property 4.

C. Generalizing AdaShift by Varying ς: Discussion and Validation

C.1. Discussion

We suppose our AdaShift prototype and practical derivatives can be generalized to various options of self-gated re-weighting functions (*i.e.*, ς) different from vanilla Sigmoid function (*e.g.*, ERF-based functions [10]). We identified this by referring to the Property 1 (clarified in Appendix B, with its induced Proposition 1 proved in [3]) of the **M**ulti-**C**riteria **D**ecision-**M**aking (**MCDM**) interpretation [3] which is suggested to explain the meaning/mechanism of a ς in an MCDM-based feature re-calibration process. That is, as a key, ς, the re-weighting function of activation, is expected to preserve the absolute relationships of different approximated similarities.

To start, for (practical) AdaShifts, we have $\phi\left(\tilde{x}\right) = \varsigma\left(\tilde{x}+\Delta\right)\tilde{x}$, *i.e.*, $\hat{\varrho}\left(\tilde{x}\right) = \tilde{x}+\Delta$. More specifically, we consider that $\Delta = \Delta\left(\tilde{\mathbf{X}}\right)$ has a scalar output, *e.g.*, the examples of AdaShift-B and -MA (presented by Equation (10) and Equation (11) in Section 3.3), where ς is a Sigmoid function. For this case, the Property 1 has been proved to be satisfied (in part-1, Appendix B, with an extended discussion) and expected to be generalizable to other ς with similar curve attributes (*e.g.*, the other three examples we demonstrate in this Appendix).

In particular, *Property 1 indicates that the practical AdaShift derivatives are possibly generalized further by applying various monotonic ς, besides the vanilla Sigmoid function.*

C.2. Validation

For further verification, we hereby investigate the generalizability of our AdaShift prototype about self-gated re-weighting functions with a targeted experiment on CIFAR-100 [12] using CIFAR-ResNet-56 [9, 25], where we compare the modified AdaShift-B(s) employing different ς with the counterparts of original popular/SOTA activation functions that propose/apply the corresponding ς, which include (1) GELU [10] with a ERF-based ς; (2) Mish [19] that suggested $\varsigma\left(\cdot\right) = \tanh\left(\mathrm{softplus}\left(\cdot\right)\right)$; (3) a specialized control group, namely, TanhGate which we modified on the vanilla Tanh function, where $\varsigma\left(\cdot\right) = 0.5\left(\tanh\left(\cdot\right)+1\right) = {}^{1}\!/_{1+e^{-2(\cdot)}}$. Note that we use ReLU and Tanh as the baselines and also show the results of the raw AdaShift-B (denoted by SiLU-Ada) with its counterpart model SiLU as a reference group.

Table 1 reports the comparative results of different groups of the AdaShift-B derivatives and the counterpart activation functions, where we have two major observations: (1) although differing each other by different ς, the actual performances of the original self-gated activation functions can be indistinguishable; (2) our original and the corresponding modified AdaShift-B(s) improve different self-gated activation function counterparts **significantly and consistently** with only negligible computational cost. These results validate the generalizability and effectiveness of our AdaShift prototype for discriminative self-gated neural feature activation.

Table 1. Evaluation on the generalizability of AdaShift prototype using different self-gated re-weighting functions ς. Activation functions with the suffix "-Ada" denote the modified AdaShift-B(s) that apply the corresponding ς.

Activation	Prototype $\phi\left(\tilde{x}\right)$		Re-weighting $\varsigma\left(\cdot\right)$	Backbone	#Params.	Top-1(%)↑
	$\varsigma\left(\tilde{x}\right)\tilde{x}$	$\varsigma\left(\tilde{x}+\Delta\right)\tilde{x}$				
ReLU [20]	—		—	CIFAR-ResNet-56	0.6M	74.4±0.3
Tanh	—		—	CIFAR-ResNet-56	0.6M	***72.3±0.3***
SiLU [8]	✓		$\mathrm{sigmoid}\left(\cdot\right)$	CIFAR-ResNet-56	0.6M	75.3±0.4
SiLU-Ada		✓	$\mathrm{sigmoid}\left(\cdot\right)$	CIFAR-ResNet-56	0.6M	**76.5±0.3**
GELU [10]	✓		$0.5\left(1+\mathrm{erf}\left(\cdot/\sqrt{2}\right)\right)$	CIFAR-ResNet-56	0.6M	75.3±0.3
GELU-Ada		✓	$0.5\left(1+\mathrm{erf}\left(\cdot/\sqrt{2}\right)\right)$	CIFAR-ResNet-56	0.6M	**76.3±0.2**
Mish [19]	✓		$\tanh\left(\mathrm{softplus}\left(\cdot\right)\right)$	CIFAR-ResNet-56	0.6M	75.2±0.3
Mish-Ada		✓	$\tanh\left(\mathrm{softplus}\left(\cdot\right)\right)$	CIFAR-ResNet-56	0.6M	**76.6±0.3**
TanhGate	✓		$0.5\left(\tanh\left(\cdot\right)+1\right)$	CIFAR-ResNet-56	0.6M	75.4±0.3
TanhGate-Ada		✓	$0.5\left(\tanh\left(\cdot\right)+1\right)$	CIFAR-ResNet-56	0.6M	**76.5±0.3**

D. Extended Ablation Study on AdaShift Prototype

We explained our main intuitions of the AdaShift prototype by rethinking a common Softmax-based classification process in Section 3.2. In this Appendix, we further discuss the complementary intuitive properties that help ensure a discriminative feature activation by extending the first ablation study (*i.e.*, **AdaShift prototype**) introduced in Section 4.3 (Ablation Study). To this end, we first conduct a tailored experiment and then identify the intuitive properties from the experimental results and new observations. Specifically, we compare our AdaShift-B with two different sets of targeted control groups:

1. A series of **C**ontrol **G**roups (**CG**s) of modified AdaShift-B(s) built on various prospective prototypes of activation functions (specified in Table 2, have introduced in the first ablation study of Section 4.3).
2. A set of SOTA self-gated activation functions, including ACON-C [18], Meta-ACON [18], SMU-1 [2], and SMU [2], constructed on a specialized self-gated prototype modified from the base prototype (Equation (1) in the manuscript):

$$\phi\left(\tilde{x}\right) = \eta\varsigma\left(\kappa\tilde{x}\right)\tilde{x} + \varepsilon\tilde{x}\,, \tag{15}$$

where η, κ, and ε are trainable coefficients (*e.g.*, SMU-1 [2], SMU [2], and ACON-C [18]) or content-aware modules (*e.g.*, Meta-ACON [18]).

Table 2. An extended ablation study on learnable activation prototypes, where ReLU is set as the baseline.

Activation	Prototype	Re-weighting $\varsigma\left(\cdot\right)$	Backbone	#Params.	Top-1(%)↑
ReLU [20]	—	—	CIFAR-ResNet-56	0.6M	74.4±0.3
Proto-CG1	$\phi\left(\tilde{x}\right) = \varsigma\left(\tilde{x}\right)\tilde{x}$	sigmoid $(\cdot)$	CIFAR-ResNet-56	0.6M	75.3±0.4
Proto-CG2	$\phi\left(\tilde{x}\right) = \varsigma\left(\kappa\tilde{x}\right)\tilde{x}$			0.6M	74.8±0.2
Proto-CG3	$\phi\left(\tilde{x}\right) = \varsigma\left(\Delta\tilde{x}\right)\tilde{x}$			0.6M	***73.4±0.3***
Proto-CG4	$\phi\left(\tilde{x}\right) = \varsigma\left(\kappa\tilde{x}+\Delta\right)\tilde{x}$			0.6M	***73.7±0.3***
Proto-CG5	$\phi\left(\tilde{x}\right) = \varsigma\left(\Delta_1\tilde{x}+\Delta_2\right)\tilde{x}$			0.6M	***73.6±0.2***
Proto-CG6	$\phi\left(\tilde{x}\right) = \varsigma\left(\tilde{x}+\Delta\right)\left(\tilde{x}+\Delta\right)$			0.6M	75.9±0.3
Proto-CG7	$\phi\left(\tilde{x}\right) = \varsigma\left(\tilde{x}+\Delta_1\right)\left(\tilde{x}+\Delta_2\right)$			0.6M	76.2±0.4
ACON-C [18]	$\phi\left(\tilde{x}\right) = \eta\varsigma\left(\kappa\tilde{x}\right)\tilde{x} + \varepsilon\tilde{x}$	sigmoid $(\cdot)$	CIFAR-ResNet-56	0.6M	74.1±0.3
Mt-ACON [18]				0.6M	75.7±0.2
SMU-1 [2]		erf $(\cdot)$		0.6M	74.7±0.2
SMU [2]				0.6M	74.9±0.3
AdaShift-B	$\phi\left(\tilde{x}\right) = \varsigma\left(\tilde{x}+\Delta\right)\tilde{x}$	sigmoid $(\cdot)$	CIFAR-ResNet-56	0.6M	**76.5±0.3**

In Table 2, we report the comparative results of different activation models on CIFAR100 [12] with CIFAR-ResNet-56 [9, 25] backbone, where ReLU serves as the baseline. Note that
(1) all the compared methods in the set-1 use Sigmoid as the ς;
(2) the ς of the SOTA self-gated activation functions in set-2 can be found in Table 2;
(3) Δ denotes the proposed shift factor of AdaShift-B;
(4) in particular, Δ_1 and Δ_2 are assigned independently (*i.e.*, employing independent parameters);
(5) κ is specified as channel-wise trainable parameters except for Meta-ACON [18] which learns SE-Net-style [11] channel weights with a lightweight MLP;
(6) CG-1 and CG-2 are equivalent to SiLU [8] and Swish [21], respectively.
Our major observations and the supposed explanations are 4-fold:
a) AdaShift yields the highest accuracy among all the compared prototypes and SOTA self-gated activation models. **This validates the effectiveness of our AdaShift prototype.**
b) CG6 which equals to $\phi\left(\tilde{x}'\right) = \varsigma\left(\tilde{x}'\right)\tilde{x}', \tilde{x}' = \tilde{x} + \Delta$ improves CG1 and CG2 but leads to accuracy drops to AdaShift-B. **This demonstrates that (a) the tensor-level non-local cues are contributing to adaptive feature translations; (b) the *mismatch feature scoring* problem of Act is hard to be eliminated by the direct adjustments on features outside ς and instead, the adaptive adjustments on the re-weighting curve about the input features can be more effective.**

c) CG7 which employs two ways of Δ(s) to shift features from both inside and outside of ς fails to improve AdaShift-B. **This validates that an activation process cannot cumulate the contributions led by the same non-local cues.**

d) CG3, CG4, and CG5 which try to combine channel scalings with the Δ factor fail to achieve practical improvements but demonstrate significant accuracy drops to CG1, CG2, and AdaShift-B. **This indicates that the raw $\tilde{x}$ can serve as an informative anchor for Δ to cast tailored adaptive adjustments on the feature re-weighting within the ς. Re-scaling the raw $\tilde{x}$ can be interfering since disrupting the original connections of Δ and $\tilde{x}$.**

As auxiliary explanations of the contributing properties of our AdaShift prototype, we qualitatively compare it with several important control groups that also leverage trainable parameters as follows.

1) **Comparison with Swish [21] (*i.e.*, CG-2).** Swish facilitates a flexible Sigmoid-based re-weighting by adding channel-wise scaling factors on the inputs. However, we identify a critical weakness in the flexibility led by this paradigm. That is, although κ shows no limits on positivity or negativity, it can only be a positive or negative value for an individual channel in an iteration or inference. This attribute constrains Swish-like functions to cast fine-grained adjustments on different feature units within the same channel since we suppose that feature units of different spatial locations can have highly differentiated importance scores as for re-calibrations. Specifically, as for a given channel-c, the re-weighting driven by $\varsigma\left(\tilde{\boldsymbol{X}}_c\right) = \text{sigmoid}\left(\kappa_c\tilde{\boldsymbol{X}}_c\right)$ is still monotonic about the raw input channel slice (*i.e.*, matrix) $\tilde{\boldsymbol{X}}_c \in \mathbb{R}^{H\times L}$, therefore falling short in alleviating the critical *mismatched feature scoring* problem. We suppose this explanation can be generalized to SOTA Swish-like functions that suggest different ς yet demonstrate close results, *e.g.*, ErfAct [1] and Pserf [1] (their results can be found in Table 3 in Section 4.2 of the manuscript, which are clearly inferior to our AdaShift-B). In contrast, our AdaShift enables fine-grained flexible adaptive adjustments to different feature units in each channel by leveraging an addable translation factor Δ.

2) **Comparison with SOTA self-gated activation functions based on prototype 15.** To our understanding, the main change from functions of prototype 15 to Swish-like functions is the introducing of a leakage term $\varepsilon\tilde{x}$ which casts feature translations outside the re-weighting curve ς. However, this paradigm remains a critical weakness: The main re-weighting process, *i.e.*, $\eta\varsigma\left(\kappa\tilde{x}\right)$ preserves the monotonicity on the input $\tilde{x}$ while the leakage term $\varepsilon\tilde{x}$ which translates feature outside ς falls short in making use of the effective nonlinearity, thus resulting in limited feature adjustments.

Based on the above experimental results and qualitative analysis, we summarize our complementary intuitive properties of an effective self-gated neural activation as follows. That is, we expect a self-gated activation function capable of:

(a) casting intense yet flexible changes on the inputs, *i.e.*, capable of giving slight adjustments on the inputs while also capable of drastically varying inputs from positive values to negative depending on the corresponding learning states;

(b) realizing fine-grained adjustments to the inputs, *i.e.*, preserving the diversity of the intensities of feature units while avoiding neutralizing the differences of feature units.

(c) constraining the (main) adjustments within the re-weighting process.

E. Auxiliary Ablation Study Using AdaShift-MA

In this Appendix, we conduct the corresponding supplementary ablation studies with AdaShift-MA as auxiliary investigations of the effects of the AdaShift prototype.

Hypothesis: balanced summation of $\tilde{x}$ and Δ (AdaShift-MA version). In this supplementary ablation study for part-2 of Section 4.3, we further investigate the hypothesis of *balanced summation* by comparing the original AdaShift-MA to the abridged AdaShift-MA, *i.e.*, *Ada-MA-CG1* which replaces the post-LN with a vanilla parametric scaling-and-shift operation. The implementation protocols are identical to other experiments on CIFAR-100.

Table 3 reports the comparative results, which we identify to be consistent with the corresponding phenomena reported in the ablation study using AdaShift-B. This further validates the hypothesis.

Table 3. Ablation study on the hypothesis of imbalanced summation (AdaShift-MA version). We report the $\text{mean} \pm \text{std}$ of the Top-1.

Activation	Backbone	#Params.	FLOPs	Top-1(%)↑
ReLU [20]	CIFAR-ResNet-56	0.6M	90.7M	74.4±0.3
Ada-MA-CG1	CIFAR-ResNet-56	0.6M	92.2M	76.3±0.2
AdaShift-MA	CIFAR-ResNet-56	0.6M	92.2M	**77.0±0.4**

Table 4. Evaluation on the generalizability of AdaShift prototype using different self-gated re-weighting functions ς. Activation functions with the suffix "-Ada-MA" denote the modified AdaShift-MA(s) that apply the corresponding ς.

Activation	Prototype $\phi(\tilde{x})$		Re-weighting $\varsigma(\cdot)$	Backbone	#Params.	Top-1(%)↑
	$\varsigma(\tilde{x})\tilde{x}$	$\varsigma(\tilde{x}+\Delta)\tilde{x}$				
ReLU [20]	—		—	CIFAR-ResNet-56	0.6M	74.4±0.3
Tanh	—		—	CIFAR-ResNet-56	0.6M	***72.3±0.3***
SiLU [8]	✓		$\text{sigmoid}(\cdot)$	CIFAR-ResNet-56	0.6M	75.3±0.4
SiLU-Ada-MA		✓	$\text{sigmoid}(\cdot)$	CIFAR-ResNet-56	0.6M	**77.0±0.4**
GELU [10]	✓		$0.5(1+\text{erf}(\cdot/\sqrt{2}))$	CIFAR-ResNet-56	0.6M	75.3±0.3
GELU-Ada-MA		✓	$0.5(1+\text{erf}(\cdot/\sqrt{2}))$	CIFAR-ResNet-56	0.6M	**76.8±0.2**
Mish [19]	✓		$\tanh(\text{softplus}(\cdot))$	CIFAR-ResNet-56	0.6M	75.2±0.3
Mish-Ada-MA		✓	$\tanh(\text{softplus}(\cdot))$	CIFAR-ResNet-56	0.6M	**76.7±0.2**
TanhGate	✓		$0.5(\tanh(\cdot)+1)$	CIFAR-ResNet-56	0.6M	75.4±0.3
TanhGate-Ada-MA		✓	$0.5(\tanh(\cdot)+1)$	CIFAR-ResNet-56	0.6M	**76.9±0.3**

Generalizing AdaShift by varying ς (AdaShift-MA version). In this supplementary ablation study for Appendix C, we further validate the generalizability of our AdaShift prototype about self-gated re-weighting functions (*i.e.*, ς), where the setup strategies of the control groups are maintained.

In Table 4, we report the comparative results between different groups of the AdaShift-MA derivatives and their counterpart activation models, where phenomena are consistent with the corresponding ablation study on the AdaShift-B derivatives. This further validates the generalizability and effectiveness of the proposed AdaShift prototype.

Table 5. Ablation study on learnable activation prototypes. ReLU is set as the baseline.

Activation	Prototype	Re-weighting $\varsigma(\cdot)$	Backbone	#Params.	Top-1(%)↑
ReLU [20]	—	—	CIFAR-ResNet-56	0.6M	74.4±0.3
Proto-MA-CG1	$\phi(\tilde{x})=\varsigma(\tilde{x})\tilde{x}$	$\text{sigmoid}(\cdot)$	CIFAR-ResNet-56	0.6M	75.3±0.4
Proto-MA-CG2	$\phi(\tilde{x})=\varsigma(\kappa\tilde{x})\tilde{x}$	$\text{sigmoid}(\cdot)$	CIFAR-ResNet-56	0.6M	74.8±0.2
Proto-MA-CG3	$\phi(\tilde{x})=\varsigma(\Delta\tilde{x})\tilde{x}$	$\text{sigmoid}(\cdot)$	CIFAR-ResNet-56	0.6M	***74.2±0.3***
Proto-MA-CG4	$\phi(\tilde{x})=\varsigma(\kappa\tilde{x}+\Delta)\tilde{x}$	$\text{sigmoid}(\cdot)$	CIFAR-ResNet-56	0.7M	***74.6±0.2***
Proto-MA-CG5	$\phi(\tilde{x})=\varsigma(\Delta_1\tilde{x}+\Delta_2)\tilde{x}$	$\text{sigmoid}(\cdot)$	CIFAR-ResNet-56	0.7M	***74.4±0.2***
Proto-MA-CG6	$\phi(\tilde{x})=\varsigma(\tilde{x}+\Delta)(\tilde{x}+\Delta)$	$\text{sigmoid}(\cdot)$	CIFAR-ResNet-56	0.6M	76.4±0.2
Proto-MA-CG7	$\phi(\tilde{x})=\varsigma(\tilde{x}+\Delta_1)(\tilde{x}+\Delta_2)$	$\text{sigmoid}(\cdot)$	CIFAR-ResNet-56	0.7M	75.8±0.3
ACON-C [18]	$\phi(\tilde{x})=\eta\varsigma(\kappa\tilde{x})\tilde{x}+\varepsilon\tilde{x}$	$\text{sigmoid}(\cdot)$	CIFAR-ResNet-56	0.6M	74.1±0.3
Mt-ACON [18]	$\phi(\tilde{x})=\eta\varsigma(\kappa\tilde{x})\tilde{x}+\varepsilon\tilde{x}$	$\text{sigmoid}(\cdot)$	CIFAR-ResNet-56	0.6M	75.7±0.2
SMU-1 [2]	$\phi(\tilde{x})=\eta\varsigma(\kappa\tilde{x})\tilde{x}+\varepsilon\tilde{x}$	$\text{erf}(\cdot)$	CIFAR-ResNet-56	0.6M	74.7±0.2
SMU [2]	$\phi(\tilde{x})=\eta\varsigma(\kappa\tilde{x})\tilde{x}+\varepsilon\tilde{x}$	$\text{erf}(\cdot)$	CIFAR-ResNet-56	0.6M	74.9±0.3
AdaShift-MA	$\phi(\tilde{x})=\varsigma(\tilde{x}+\Delta)\tilde{x}$	$\text{sigmoid}(\cdot)$	CIFAR-ResNet-56	0.6M	**77.0±0.4**

Ablation study on AdaShift prototype (AdaShift-MA version). In this supplementary ablation study for Appendix D, we further validate our intuitions, design, and choice for the AdaShift prototype, where the setup strategies of the control groups in Appendix D are maintained.

The comparative results are reported in Table 5, demonstrating that the main experimental phenomena closely align with the counterpart ablation evaluations using AdaShift-B derivatives (in Appendix D). This further validates our intuitions and design choices for the AdaShift prototype.

In addition, two new observations are obtained: (1) MA operations introduce accuracy gains on different counterparts of AdaShift-B derivatives; (2) Proto-MA-CG6 outperforms Proto-MA-CG7 with reduced learning flexibility. The second observation merits further investigation (in future work).

F. Implementation Recipe

F.1. ImageNet Classification

For fair comparisons, we adopt the basic CNN training-evaluation protocols [18, 29] for all the implemented ResNets and MobileNetV2(s)/ShuffleNetv2(s), respectively.

1. For ResNets [9], we adopt the common data augmentation strategy [29] and basic SGD optimizer to train each implemented model with the standard cosine scheduler through 120 epochs (including 5 linear warm-up epochs), where the learning rate starts from 0.1 with a batch size of 256 by default and decays to 1×10^{-5}, gradually. The applied momentum and weight decay are 0.9 and 1^{-4}, respectively. We follow the common practice to stabilize the model weights by 10 cool-down epochs with the minimum learning rate 1×10^{-5} after the main epochs.
2. For MobileNetV2(s) [22] and ShuffleNetv2(s) [17], we use SGD optimizer with a momentum of 0.9 and a weight decay of 4×10^{-5}. The common recipe and data augmentation [17] are applied to train each model with the linear scheduler with a batch size of 1024 by default, where the learning rate starts from 0.5 and decreases to 1×10^{-5} in 240 epochs (*i.e.*, 300k iterations as for the training set of ImageNet).

By following the common practice, we (1) train and test all the implemented models with an image size of 224×224; (2) report the results of our implemented models and the official results for the compared methods by Top-1 Accuracy with one decimal place.

F.2. CIFAR-100 Classification

On CIFAR-100, we apply the same training-evaluation protocols suggested in [13] for CIFAR-ResNets [9, 25] with different activation functions

For fair comparisons, we adopt the same standard training-evaluation protocols and data augmentations suggested in [13] to train all the networks with our practical AdaShifts and the compared activation functions by a basic SGD optimizer with the momentum of 0.9 and weight decay of 5×10^{-4}. Each model is trained for 350 epochs with a batch size of 256. The learning rate starts from 0.1 and decreases to 1^{-6} by following the standard cosine learning rate schedule. All the input images are fixed to the size of 32×32 by following the common practice.

G. Supplementary Results on ImageNet Classification

G.1. Evaluation on Practical Efficiency

Table 6. Supplementary evaluation on practical efficiency using ResNet-50 backbone. The image *throughput* is measured on a single RTX A6000 GPU with pure FP32 inputs with a batch size of 128.

Backbone	Activation Function	Image Resolution	#Params.	FLOPs	Throughput (image / s)
ResNet-50 [9]	ReLU [20]	224×224	25.6M	4.1G	**1482.5**
	Swish [21]	224×224	25.6M	4.1G	**1476.1**
	IIEU [3]	224×224	25.6M	4.2G	1242.6
	Pserf [1]	224×224	25.6M	4.1G	1045.3
	SMU [2]	224×224	25.6M	4.1G	1046.1
	AdaShift-B (Ours)	224×224	25.6M	4.1G	**1352.8**

Despite that SOTA activation functions often add slight parameters and theoretical computational overheads on the ReLU [20] baseline, the actual burdens on practical throughput can be heavier. To discuss this phenomenon, we conduct a supplementary experimental analysis on the practical efficiency (measured by throughput) by comparing our AdaShift-B to SOTA activation functions on ResNet-50 [9] backbone, which include Swish [21], Pserf [1], SMU [2], and IIEU [3], where we use ReLU as the reference with comparatively highest speed (due to the relatively simplest operations).

We report the comparative practical image throughput of different models in Table 6, where our AdaShift-B adds marginal practical efficiency overhead on the ReLU baseline yet demonstrates competitive speed among SOTAs. It is worth noting that AdaShift-B enjoys significant improvements over SOTA activation functions in accuracy (as detailed in Section 4). This validates AdaShift for practical application.

G.2. Training with Advanced Recipe

In this Appendix, we demonstrate the further potential of practical AdaShifts, capable of boosting a standard CNN (*i.e.*, ResNet [9]) to match/chase advanced vision Transformer counterparts of a similar level of parameters and FLOPs, by simply replacing ReLU [20] in each of the corresponding layers and by applying an advanced 300-epoch recipe [26] for standard CNN training (for fair comparisons). Unlike the basic 120-epoch CNN training procedure [29] adopted in the main ImageNet experiments in Section 4.1, this new training procedure includes more diverse data augmentations and longer epochs, which is similar to the most prevalent 300-epoch training procedure [24] for Vision Transformer(s).

In particular, by taking into account the accuracy-efficiency trade-off, we conduct this comparative evaluation with two modified ResNet-50(s) enhanced by the targeted practical AdaShift derivatives: (1) AdaShift-B, *i.e.*, the simplest practical AdaShift and (2) a new **Hyb**rid AdaShift derivative, namely, AdaShift-**Hyb**, which hybridly using a modified AdaShift-MA, a modified AdaShift-B, and an original AdaShift-B to activate the feature maps outputted from the convolutional-layer-1, -2, and -3 (with BN operated) of each building block, respectively. Note that (1) each (bottleneck) residual block has three convolutional layers in total; (2) here, the *modification* to an AdaShift-B/-MA refers to adding a linear layer before LN. So, the overall modifications to the residual block only bring SE-Net [11] level parameters and cost. Then, these two AdaShift-enhanced ResNet-50(s) are compared with the representative Transformers, including ViT [7], PoolFormer [28], and Swin-Transformer [15], where ViT is set as the baseline model.

Table 7 reports the comparative results on ImageNet, where we have three major observations: (1) AdaShift-B can boost ResNet-50 to outperform ViT-B/16 and PoolFormer-S24 with a similar level of parameters and theoretical computational cost (measured by FLOPs) while demonstrating superior practical efficiency (measured by throughput). (2) ResNet-50 enhanced by AdaShift-B approaches the strong Swin-T in accuracy with fewer parameters, lower theoretical FLOPs, and notably higher practical speed. (3) AdaShift-Hyb enables ResNet-50 to outperform Swin-T with lower cost and higher efficiency.

This demonstrates the critical potential and meaning of effective neural feature activation and validates our intuitions and design of AdaShift for learning improved self-gated neural activation models.

Table 7. Comparison of AdaShift-enhanced ResNet-50(s) to representative vision Transformer counterparts. "$\star$" denotes the improved ViT trained with an extra regularization [23]. The practical image *throughput* is measured on a single RTX A6000 GPU with pure FP32 inputs with a batch size of 128.

Network	Activation Function	Image Resolution	#Params.	FLOPs	Throughput (image / s)	Top-1(%)$\uparrow$
ViT-B/16 [15]$\star$	GELU [10]	224 × 224	86.6M	16.9G	775.6	79.7
PoolFormer-S24 [28]	GELU [10]	224 × 224	21.4M	3.5G	1144.6	80.3
Swin-T [15]	GELU [10]	224 × 224	28.3M	4.5G	1052.2	**81.3**
ResNet-50 [9]	**AdaShift-B (Ours)**	224 × 224	25.6M	4.1G	**1352.8**	**80.8**
ResNet-50 [9]	**AdaShift-Hyb (Ours)**	224 × 224	28.2M	4.2G	1201.6	**81.7**

G.3. Evaluation on Lightweight Backbone

As a supplementary experiment on ImageNet [6] classification, we evaluate our AdaShift design by comparing the practical AdaShift-MA with other popular/SOTA activation functions with two lightweight networks, *i.e.*, MobileNetV2 [22] and ShuffleNetv2 [17], where the baseline networks are activated by ReLU [20] function and we show the improvements in Top-1 accuracy of our method over the ReLU baselines in "$(+\cdot)$". We report the implemented results for our AdaShift-MA and the official results for other compared models, respectively. Note that the basic training-evaluation protocols adopted in this evaluation are described in the configure 2, Appendix F.1.

report the comparative results, where our practical AdaShift achieves clear accuracy improvements over all the other compared models with negligible parameters added on the ReLU baselines. These phenomena are in line with the main evaluations on ResNets with different sizes (demonstrated in Section 4 of the manuscript). This validates the scalability of our AdaShift to various types of neural networks.

Table 8. Comparison of different activation functions with ShuffleNetv2 $1.0\times$ [17] backbone on ImageNet.

Backbone	FLOPs	Activation	ReLU [20]	Mish [19]	Mt-ACON [2]	SMU-1 [2]	SMU [2]	**AdaShift-MA**
ShuffleNetV2 1.0× [17]	153.5M	#Params.	2.3M	2.3M	2.6M	2.3M	2.3M	2.3M
		Top-1(%)↑	69.4	70.5	72.1	71.2	71.9	**72.3(*+2.9*)**

Table 9. Comparison of different activation functions with MobileNetV2 [22] 1.0× backbone on ImageNet. "N/A" denotes non-applicable (*i.e.*, not mentioned in the official publication.)

Backbone	FLOPs	Activation	ReLU [20]	ACON-C [2]	Mt-ACON [2]	PWLU [30]	**AdaShift-MA**
MobileNetV2 1.0× [22]	320.3M	#Params.	3.5M	3.6M	3.9M	N/A	3.6M
		Top-1(%)↑	72.1	73.6	75.0	74.7	**75.7(*+3.6*)**

G.4. Practical AdaShift For MetaFormer-Like Networks

In Section 3.3, we clarified that AdaShift-B and -MA were tailored to normalized activation inputs (otherwise encountering the *imbalanced summation* problem), hence unsuitable for MetaFormer layers where the activation inputs are commonly un-normalized in **F**eed-**F**orward-**N**etworks (**FFN**s). To this end, we introduce a new version of practical AdaShift, which we refer to as **AdaSfhit-X**, modified from AdaShift-MA yet focusing on enhancing MetaFormer-style layers that encode features with FFNs. Specifically, as a main change, AdaSfhit-X adds a post-linear-projection at the top of MA-based Δ to avoid imbalanced summation when casting adaptive translations on un-normalized FFN inputs. Note that this extra projection may be removable if applied on typical Transformers that naturally possess the post-linear-projections in the token mixers (such that only adds negligible parameters).

By taking into account the computational resource constraints, we choose to evaluate this new AdaShift derivative with ConvNext-T [16], a highly efficient advanced ConvNet inspired by MetaFormer architecture with a close size to ResNet-50 but way stronger. We report the evaluation results in Table 10, where ConvNext equipped with AdaShift-X enjoys significant improvements in accuracy on the original GELU-based counterpart (taking into account the diminishing returns effect on a strong network backbone). It is worth noting that **our AdaShift-X, to the best of our knowledge, is the first and only existing activation function that can add practical accuracy gains (*i.e.*, $\geqslant$ 0.3%) on advanced MetaFormer-like networks** (originally using GELU [10] activation in common). **This validates the versatility and scalability of our AdaShift prototype.**

Table 10. ImageNet evaluation of AdaShift-X on ConvNext-T [16]. The practical image *throughput* is measured on a single RTX A6000 GPU with pure FP32 inputs with a batch size of 128.

Backbone	FLOPs	Activation Function	Image Resolution	#Params.	Throughput (image / s)	Top-1(%)↑
ConvNext-T [16]	4.5G	GELU [10]	224 × 224	28.6M	1220.1	82.1
		AdaShift-X (Ours)	224 × 224	32.0M	1075.3	**82.8**

G.5. Convergence Attribute

We show the convergence curves of ResNet-14(s) and -26(s) [9] enhanced by our AdaShift-B, AdaShift-MA, and other baseline/popular/SoTA activation functions. Note that each model is trained by the basic 120-epoch recipe (introduced in Appendix F.1) from scratch to convergence, respectively.

Figure 3 depicts the convergence trends in Top-1 accuracy (the higher the better) and training loss (the lower the better) of the ResNet-14 and ResNet-26 equipped with our IIEUs and other activation functions, respectively, where ReLU networks are the baselines. Our AdaShift-B and AdaShift-MA achieve the relatively highest Top-1 accuracies and lowest loss values over the varying of epochs. This validates the favorable convergence attributes of our activation models.

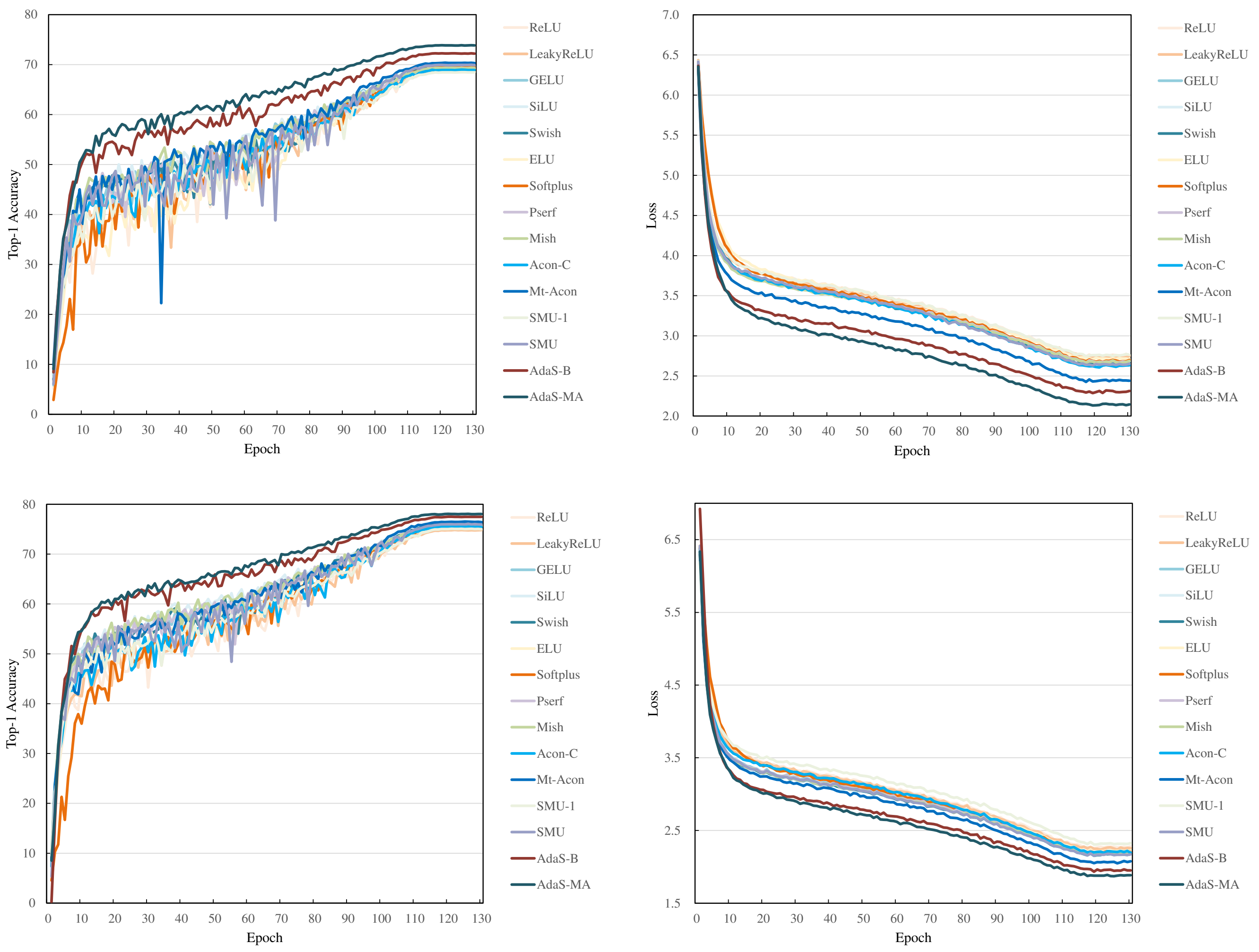

Figure 3. Top: the accuracy curve (left) and loss curve (right) of ResNet-14 backbone with different activation models. Bottom: the accuracy curve (left) and loss curve (right) of ResNet-26 backbone with different activation models.

G.6. Fine-tuning

In this Appendix, we conduct a special experiment to further investigate the generalizability and scalability (compatibility) of AdaShift activation process, where we fine-tune the AdaShift-B-enhanced ResNet-50 [9] network on the raw ResNet-50 backbone which uses ReLU activation. It is worth noting that AdaShift-B and ReLU may yield significantly different activation outputs for the same inputs, and AdaShift-B includes parameters (*i.e.*, LN) to be updated and these parameters cannot find useful weights from the raw ResNet(-50). We suppose this to be a critical challenge for fine-tuning.

In this experiment, we consider four different experiment settings which are the combinations of the two epoch settings for training (30-epoch and 45-epoch, respectively) and two settings for initial learning rate (0.05 and 0.03 for mini-batch of

256, respectively).

Table 11 reports the experimental results of different fine-tuning settings, where we have four major observations: (1) Without fine-tuning, AdaShift-B ResNet-50 that directly loads the trained raw ResNet-50 model weights yields extremely low results. (2) AdaShift-B ResNet-50 with each designated fine-tuning setting can outperform the raw ReLU ResNet-50 significantly. (3) Different learning rates of 0.05 and 0.03 can lead to close final accuracies (this demonstrates the training adaptability of the model) (4) The network performance can be improved by stretching the fine-tuning epochs. In particular, with only 45-epoch fine-tuning, AdaShift-B ResNet-50 approaches the original results of AdaShift-B obtained by the 120-epoch training from scratch.

This further validates the generalizability and compatibility of AdaShift for neural activation.

Table 11. Fine-tuning evaluation of AdaShift-B ResNet-50 on ReLU ResNet-50. "FT" denotes "Fine-tuning;" "Init" denotes "Initial;" "LR" denotes "Learning Rate." "W/o," "W/," and "Orig" denote "Without," "With," and "Original," respectively.

Activation	Backbone	Image Size	#Params.	FLOPs	FT Epochs		Init LR		Top-1(%)↑
					30	45	0.03	0.05	
ReLU [20]	ResNet-50 [20]	224 × 224	25.6M	4.1G	——		——		77.2
AdaShift-B (W/o FT)	ResNet-50 [20]	224 × 224	25.6M	4.1G	——		——		**11.3**
AdaShift-B (W/ FT)	ResNet-50 [20]	224 × 224	25.6M	4.1G	✓		✓		79.2
		224 × 224	25.6M	4.1G	✓			✓	79.1
		224 × 224	25.6M	4.1G		✓	✓		**79.5**
		224 × 224	25.6M	4.1G		✓		✓	**79.7**
AdaShift-B (Orig)	ResNet-50 [20]	224 × 224	25.6M	4.1G	——		——		**79.9**

G.7. Visualization

In this Appendix, we show a visualized example of activation map, consisting of 256 channels, extracted from the learning hierarchy-1 of AdaShift-B ResNet-50 [9]. The raw input image and the generated activation map (displayed by channels) are shown in Figure 4 and Figure 5, respectively, where we observe that the activation map concentrates on different informative textures and patterns and extracts them in different channels with rich details.

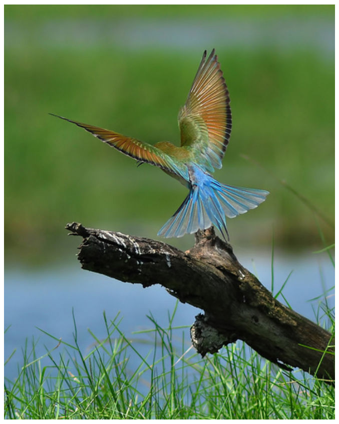

Figure 4. The raw input image (from the validation set of ImageNet) of the extracted activation map.

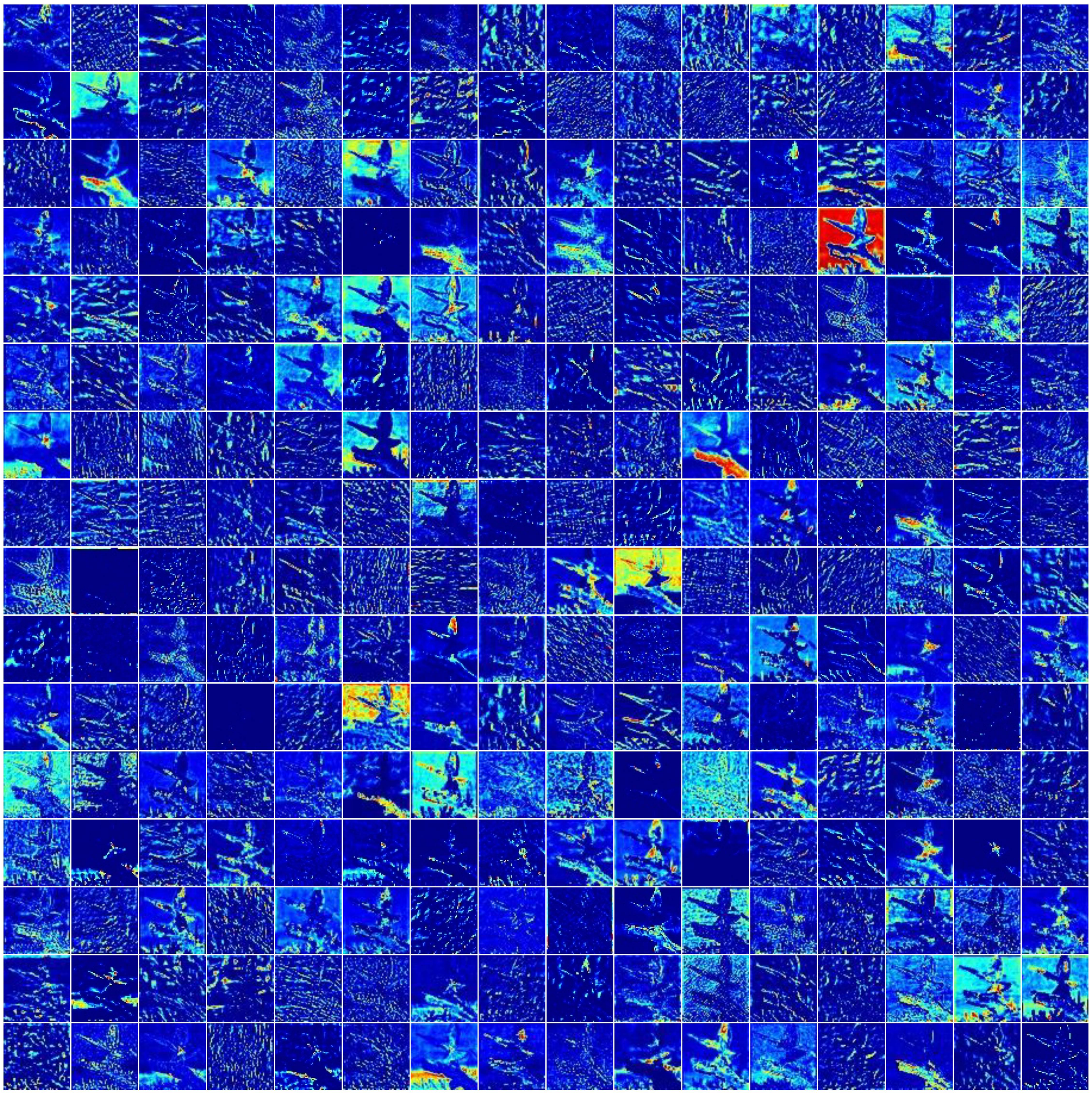

Figure 5. Visual example of the activation channels extracted from the learning hierarchy-1 of AdaShift-B ResNet-50.

H. KITTI-Materials Road Scene Material Segmentation

Implementation details. We conduct a supplemental experiment to evaluate the versatility and generalizability of our practical method by comparing AdaShift-B with popular/SOTA activation functions on KITTI-Materials [4] road scene material segmentation, different from the prevailing object recognition problems, serving as a prospective base task for downstream applications related to materials in road scenes. To ensure fair comparisons, we adopt the official training and evaluation configures [4] with a common framework composed of ResNet-50 encoder [9] and the multi-level All-MLP segmentation head [27].

Experimental results. Table 12 demonstrates the comparative results of different popular/SOTA activation functions and the ReLU baseline, where our AdaShift-B outperforms compared methods by a clear margin. This validates the versatility and generalizability of our practical design.

Table 12. Comparison of popular/SOTA activation functions on KITTI-Materials [4] road scene material segmentation.

Activation Function	Encoder	SegHead	#Params.	mIoU(%)↑
ReLU [20]	ResNet-50 [9]	All-MLP [27]	31.7M	40.2
Mt-ACON [18]			31.9M	41.7
SMU [2]			31.7M	40.6
Swish [21]			31.7M	41.2
AdaShift-B (Ours)	ResNet-50 [9]	All-MLP [27]	31.7M	**42.2**

I. MS COCO Object Detection

Implementation details. In this Appendix, we aim to further demonstrate the versatility and generalizability of our generic activation prototype, *i.e.* AdaShift, for various vision tasks. To this end, we evaluate AdaShift-B on MS COCO [14] object detection by comparing it with the ReLU [20] baseline and other popular/SOTA activation functions, *i.e.*, Meta-ACON [18], SMU [2], IIEU [3], Swish [21]. For fair comparisons, we apply the default implementation procedure (1× schedule) in MMDetection toolbox [5] and report the results on the standard evaluation metrics, *i.e.*, mAP as the primary metric of averaged precisions and AP_{50}, AP_{75}, AP_S, AP_M, AP_L as the specific APs at different scales. We use a popular efficient detector RetinaNet that extracts feature maps with ResNet-50 encoders equipped with different activation functions, each of which is applied with their corresponding ImageNet pre-trained weights. By following the common practice, we ensure reproducibility by keeping on the deterministic mode in each implementation. Note that we report the official results for Meta-ACON as our re-implemented results are lower, possibly led by the different implementation environments.

Experimental results. The experimental results are shown in Table 13, where our AdaShift-B achieves significant gains in accuracy over different popular/SOTA activation functions. It is worth noting that the highly consistent and significant improvements over the baseline and popular/SOTA activation functions on various vision benchmarks verify the strong versatility and generalizability of AdaShift for effective self-gated neural activation.

Table 13. Comparison of different activation functions on COCO [14] object detection.

Activation Function	Encoder	#Params.	FLOPs	mAP (%)↑	AP_{50}(%)↑	AP_{75}(%)↑	AP_S(%)↑	AP_M(%)↑	AP_L(%)↑
ReLU [20]	ResNet-50 [9]	37.7M	238.9G	36.7	56.0	39.3	21.0	40.2	48.2
IIEU [3]		37.7M	239.0G	38.2	58.2	40.6	**23.2**	42.1	49.2
SMU [2]		37.7M	238.9G	37.5	56.6	40.2	21.5	41.5	48.4
Mt-ACON [18]		37.9M	238.9G	36.5	55.9	38.9	19.9	40.7	**50.6**
Swish [21]		37.7M	238.9G	37.2	56.3	39.9	21.0	41.1	47.8
AdaShift-B (Ours)	ResNet-50 [9]	37.7M	238.9G	**38.8**	**58.8**	**41.5**	22.4	**42.9**	50.0

References

[1] Koushik Biswas, Sandeep Kumar, Shilpak Banerjee, and Ashish Kumar Pandey. Erfact and pserf: Non-monotonic smooth trainable activation functions. In *Proc. AAAI Conference on Artificial Intelligence (AAAI)*, 2022. 8, 10, 11

[2] Koushik Biswas, Sandeep Kumar, Shilpak Banerjee, and Ashish Kumar Pandey. Smooth maximum unit: Smooth activation function for deep networks using smoothing maximum technique. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 7, 9, 10, 11, 12, 16

[3] Sudong Cai. Iieu: Rethinking neural feature activation from decision-making. In *Proc. IEEE International Conference on Computer Vision (ICCV)*, pages 5796–5806, 2023. 2, 6, 10, 11, 16

[4] Sudong Cai, Ryosuke Wakaki, Shohei Nobuhara, and Ko Nishino. Rgb road scene material segmentation. In *Proc. Asian Conference on Computer Vision (ACCV)*, 2022. 15, 16

[5] Kai Chen, Jiaqi Wang, Jiangmiao Pang, Yuhang Cao, Yu Xiong, Xiaoxiao Li, Shuyang Sun, Wansen Feng, Ziwei Liu, Jiarui Xu, Zheng Zhang, Dazhi Cheng, Chenchen Zhu, Tianheng Cheng, Qijie Zhao, Buyu Li, Xin Lu, Rui Zhu, Yue Wu, Jifeng Dai, Jingdong Wang, Jianping Shi, Wanli Ouyang, Chen Change Loy, and Dahua Lin. MMDetection: Open mmlab detection toolbox and benchmark. *arXiv preprint arXiv:1906.07155*, 2019. 16

[6] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009. 11

[7] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. In *Proc. International Conference on Learning Representations (ICLR)*, 2021. 11

[8] Stefan Elfwing, Eiji Uchibe, and Kenji Doya. Sigmoid-weighted linear units for neural network function approximation in reinforcement learning. *Neural Networks*, 107:3–11, 2018. 6, 7, 9

[9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016. 1, 6, 7, 10, 11, 13, 14, 15, 16

[10] Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016. 6, 9, 11, 12

[11] Jie Hu, Li Shen, Samuel Albanie, Gang Sun, and Enhua Wu. Squeeze-and-excitation networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 42(8):2011–2023, 2020. 7, 11

[12] Alex Krizhevsky. Learning multiple layers of features from tiny images. Master's thesis, University of Toronto, 2009. 6, 7

[13] Xiang Li, Wenhai Wang, Xiaolin Hu, and Jian Yang. Selective Kernel Networks. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 510–519, 2019. 10

[14] Tsung-Yi Lin, Michael Maire, Serge J. Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft coco: Common objects in context. In *Proc. European Conference on Computer Vision (ECCV)*, pages 740–755, 2014. 16

[15] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin Transformer: Hierarchical Vision Transformer Using Shifted Windows. In *Proc. IEEE International Conference on Computer Vision (ICCV)*, 2021. 11

[16] Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A convnet for the 2020s. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 12

[17] Ningning Ma, Xiangyu Zhang, Hai-Tao Zheng, and Jian Sun. Shufflenet v2: Practical guidelines for efficient cnn architecture design. In *Proc. European Conference on Computer Vision (ECCV)*, pages 116–131, 2018. 10, 11, 12

[18] Ningning Ma, Xiangyu Zhang, Ming Liu, and Jian Sun. Activate or not: Learning customized activation. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8032–8042, 2021. 7, 9, 10, 16

[19] Diganta Misra. Mish: A self regularized non-monotonic neural activation function. In *Proc. British Machine Vision Conference (BMVC)*, 2020. 6, 9, 12

[20] Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *Proc. International Conference on Machine Learning (ICML)*, 2010. 6, 7, 8, 9, 10, 11, 12, 14, 16

[21] Prajit Ramachandran, Barret Zoph, and Quoc V Le. Searching for activation functions. In *Proc. Workshop Track of the 6th International Conference on Learning Representations (ICLR)*, 2018. 7, 8, 10, 11, 16

[22] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4510–4520, 2018. 10, 11, 12

[23] Ilya O Tolstikhin, Neil Houlsby, Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Thomas Unterthiner, Jessica Yung, Andreas Steiner, Daniel Keysers, Jakob Uszkoreit, et al. Mlp-mixer: An all-mlp architecture for vision. *Advances in neural information processing systems*, 34:24261–24272, 2021. 11

[24] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training Data-Efficient Image Transformers & Distillation Through Attention. In *Proc. International Conference on Machine Learning (ICML)*, 2021. 11

[25] Weiaicunzai. pytorch-cifar100. `https://github.com/weiaicunzai/pytorch-cifar100`. 6, 7, 10

[26] Ross Wightman, Hugo Touvron, and Herve Jegou. Resnet strikes back: An improved training procedure in timm. In *NeurIPS 2021 Workshop on ImageNet: Past, Present, and Future*, 2021. 11

[27] Enze Xie, Wenhai Wang, Zhiding Yu, Anima Anandkumar, Jose M Alvarez, and Ping Luo. Segformer: Simple and Efficient Design for Semantic Segmentation with Transformers. In *Proc. Advances in Neural Information Processing Systems (NeurIPS)*, 2021. 15, 16

[28] Weihao Yu, Mi Luo, Pan Zhou, Chenyang Si, Yichen Zhou, Xinchao Wang, Jiashi Feng, and Shuicheng Yan. Metaformer is actually what you need for vision. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 11

[29] Jingkai Zhou, Varun Jampani, Zhixiong Pi, Qiong Liu, and Ming-Hsuan Yang. Decoupled Dynamic Filter Networks. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 10, 11

[30] Yucong Zhou, Zezhou Zhu, and Zhao Zhong. Learning specialized activation functions with the piecewise linear unit. In *Proc. IEEE International Conference on Computer Vision (ICCV)*, pages 12095–12104, 2021. 12