PIDSCORE: A STUDENT-CENTRIC ADAPTIVE FUSION METHOD FOR KNOWLEDGE DISTILLATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Knowledge distillation aims to train lightweight student models by balancing soft supervision from a teacher model with hard supervision from ground-truth labels, making it a key challenge to determine an appropriate fusion weight. Existing methods that use fixed fusion weights lack adaptability, while another class of strategies that rely on dynamic adjustments based on teacher signals fail to fully account for the learning state of the student model itself. To overcome these limitations, we propose PIDScore, a student-centric adaptive fusion method inspired by Proportional-Integral-Derivative (PID) controller theory. PIDScore relies solely on the student's own state and the ground-truth labels, dynamically adjusting the fusion weight at the sample level and operating completely independent of the teacher model. Specifically, it employs a PID branch that relies solely on the student model, integrating student's current prediction error (P), historical performance (I) and learning trend (D) to offer precise, fine-grained optimization guidance. As a teacher-agnostic, plug-and-play module, PIDScore demonstrates strong effectiveness and excellent generalization, consistently improving the performance of various state-of-the-art distillation methods on CIFAR-100, MS-COCO and ImageNet across extensive experiments. The code of this work will be released upon acceptance.

1 INTRODUCTION

With the rapid growth of deep neural network parameters and computational complexity, large-scale models achieve superior performance yet incur high costs, making them difficult to deploy on mobile terminals and edge devices, as well as in real-time applications. Therefore, making model lightweight while maintaining performance has become a key research problem. In this context, Knowledge Distillation (KD) Hinton et al. (2015), as a classic and effective model compression method, can transfer knowledge from large teacher models to lightweight student models, striking a balance between accuracy and efficiency.

In KD, the student model usually needs to learn from two supervisory signals simultaneously that are hard supervision from ground-truth labels and soft supervision from the teacher model's output distribution, intermediate features or structural relations. Regardless of whether a logit-based, feature-based or relation-based distillation paradigm is adopted, the essential issue is to coordinate the contributions of task and distillation supervision. Thus, the knowledge fusion weight, which is the trade-off coefficient between different supervisory signals, is a key issue across various KD frameworks. A reasonable fusion weight affects the knowledge extent to which the student model absorbs from the teacher and how well it learns from the true task objective. Ideally, this weight should be dynamically adjusted according to sample difficulty and the learning state of the student model at different training stages, achieving finer-grained and more effective knowledge transfer Li et al. (2023); Lu et al. (2021).

Current methods for setting the knowledge fusion weight fall into two categories. The first category adopts fixed empirical weights, such as KD Hinton et al. (2015), CTKD Li et al. (2023), DKD Zhao et al. (2022) and MLKD Jin et al. (2023). These methods are simple in form and straightforward to implement, but they use a fixed, uniform fusion weight throughout the training, failing to adapt to varying sample difficulties or the dynamic changes of the student model during training. The second category attempts to adaptively adjust the fusion weight according to the prediction discrepancy

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

between teacher and student models, such as ADA-KD Zhang et al. (2023) and RW-KD Lu et al. (2021). Unlike fixed-weight methods, these approaches offer some dynamic adjustment capability. However, their adjustment mechanisms mainly rely on the teacher-student discrepancy at the current moment and are essentially a proportional-response strategy based on instantaneous error, lacking systematic modeling of state information over the entire learning process.

Overall, most existing methods rely on single, local information sources and fail to exploit the student model's global state throughout the entire learning process. In fact, the historical performance of the student model reflects its long-term learning state, while current performance trends reveal the learning direction and speed. These two types of information correspond to the integral and differential states in control theory, and are crucial for more precisely adjusting the weights among different supervisory signals. However, most existing KD methods focus only on current error, ignoring historical accumulated information and dynamic trend information, which limits the effectiveness and robustness of the fusion weight adjustment.

To address the above issues, this paper leverages the proportional-integral-derivative (PID) control theory to reformulate the sample-wise knowledge fusion weight determination as a real-time dynamic adjustment problem, and proposes a student-centric adaptive weighting method, termed PIDScore. Unlike existing methods that rely on teacher-side information or fixed empirical rules, PIDScore dynamically adjust sample-level distillation weights using only the student's prediction error relative to ground truth. Therefore, PIDScore requires no access to the teacher model's state, making it a truly teacher-independent, plug-and-play general module. By jointly modeling the student's current error, historical accumulated performance and stage-wise changing trend, it integrates both local and global learning information, providing finer-grained and more robust distillation control. Moreover, it can be naturally adapted to various KD frameworks, thereby bringing more stable and efficient performance improvements. The main contributions are summarized as follows:

- We propose a student-driven, sample-level distillation weighting framework that adaptively adjusts the distillation weight by tracking the student's per-sample learning state, avoiding reliance on fixed rules, heuristics or teacher-side evaluation of existing methods.
- A student-feedback-driven PID mechanism is developed that feeds sample-wise student performance back into the PID terms for fine-grained dynamic weight updating, enabling more accurate distillation control across different samples and training stages.
- We unify different distillation paradigms by formulating KD as a dynamic trade-off between ground-truth and teacher supervision, enabling our method as a plug-and-play module for logit-based, feature-based, and relation-based distillation frameworks.
- Extensive experiments on CIFAR-100, MS-COCO and ImageNet with diverse teacher and student models across multiple distillation methods based on different strategies demonstrate the significant plug-and-play advantages of our method.

2 Related works

2.1 Knowledge Distillation

KD aims to transfer knowledge from a complex teacher network to a lightweight student network, improving the student's performance and generalization. Existing KD methods can be grouped into logit-based, feature-based or relation-based paradigms.

Logit-based methods rely solely on the final output logits of the teacher and student networks for knowledge transfer. They are simple, efficient, architecture-agnostic, and thus the most widely applicable. Hinton et al. Hinton et al. (2015) pioneered the use of temperature-scaled softmax and KL divergence to align the soft probability distributions of the teacher and student models. Subsequent improvements include CTKD Li et al. (2023), DKD Zhao et al. (2022) , MLKD Jin et al. (2023), Logit-stand Sun et al. (2024), and others, which further optimize the transfer process and improve the generalization of student models.

Feature-based methods transfer richer internal knowledge by aligning intermediate layer feature representations between teacher and student. Representative works include FitNets Romero et al.

(2014), AT Zagoruyko & Komodakis (2017), CRD Peng et al. (2019), ReviewKD Lu et al. (2021), SimKD Chen et al. (2022), CAT-KD Guo et al. (2023), activation boundary methods, etc. Among them, TopKD Kim et al. (2024) utilizes topological data analysis to capture the global structure of feature embeddings, constructs multi-scale geometric knowledge through persistence diagrams and images, and designs a dedicated distillation loss for alignment, achieving excellent performance in complex latent spaces.

Relation-based methods focus on transferring structured relations between samples or features through similarity matrices or relation graphs. Representative works include RKD Park et al. (2019), SP Tung & Mori (2019), FSP Yim et al. (2017) , CRD Tian et al. (2022), etc. NRKD Gou et al. (2025) is an efficient relation distillation method that focuses on local neighborhood structures of samples. It proposes two variants, which are Neighborhood Feature Relation Distillation (NFRD) and Neighborhood Logits Relation Distillation (NLRD), utilizing rich information from low-level and high-level layers to align local relations, respectively. It achieves excellent performance on multiple image classification datasets, especially for tasks requiring semantic relation capture.

2.2 Knowledge Fusion weight with Its Determination Methods

Although the three method categories differ in knowledge representation, they share a common core mechanism which utilizes a fusion weight α to balance the distillation loss $L_{distill}$ and the hard-label supervision loss L_{ce}, i.e., $L_{total} = \alpha L_{distill} + (1-\alpha)L_{ce}$. This weight directly controls the trade-off between teacher guidance and ground-truth labels, critically affecting KD performance. Existing methods for determining the fusion weight fall into two main categories, which are fiexed empricial weights and dynamic adjustment methods.

Fixed empirical weights, used in classic KD Hinton et al. (2015), CTKD Li et al. (2023), DKD Li et al. (2023), MLKD Jin et al. (2023), etc., are typically set via grid search or manual experience. They are simple to implement but lack theoretical foundation, rely heavily on manual tuning, and fail to adapt to different tasks, datasets or teacher-student architecture gaps, leading to unstable generalization across domains or heterogeneous models Li et al. (2023); Lu et al. (2021); Zhang et al. (2020).

Dynamic adjustment methods based on teacher-student prediction differences, such as ADA-KD Zhang et al. (2023), RW-KD Lu et al. (2021) and WLS-KD Zhou et al. (2021), can better adapt to training dynamics and sample heterogeneity, but their weight computation depends heavily on the quality of the teacher's predictions.

However, fixed empirical weights have certain limitations. Their values are typically selected for specific model pairs and distillation objectives. Moreover, these weights remain constant throughout training, limiting their ability to dynamically adjust the relative contributions of task and distillation supervision as the student's learning state evolves across training stages. Dynamic methods are more flexible but over-rely on the teacher model. If the teacher model contains bias, noise or overfitting, the student model will amplify incorrect guidance, thereby exacerbating the knowledge mismatch and ultimately harming generalization. This heavy reliance on the teacher model limits KD robustness in real-world deployment, especially when the teacher model is imperfect or faces distribution shift.

In summary, although KD has made significant progress under the three mainstream paradigms and the fusion weight is widely adopted as a core mechanism, current weight determination methods still suffer from insufficient transferability and over rely on the quality of the teacher model. This paper designs a fusion weight strategy that computes fusion weights without using teacher outputs and possesses strong cross-method transferability, aiming to further improve the generalization ability of KD.

3 Background and Notation

Distillation Objective. Let the transfer dataset be denoted as $\mathcal{D} = \{(x_n, y_n)\}_{n=1}^{N}$, where $x_n \in \mathbb{R}^{H\times W}$ represents the n-th input sample with spatial dimensions H and W, and $y_n \in \{0,1\}^K$ is the corresponding one-hot ground-truth vector over K classes. N is the total number of samples. For a given input x_n, the student model f_S and the teacher model f_T generate the corresponding logit vectors $z_n = f_S(x_n)$ and $v_n = f_T(x_n)$, where $z_n, v_n \in \mathbb{R}^K$. Following the standard KD paradigm,

the softened predictive distributions of the student and the teacher are defined via temperature scaling:

$$q(z_n)^{(k)} = \frac{\exp\left(z_n^{(k)}/T\right)}{\sum_{i=1}^{K}\exp\left(z_n^{(i)}/T\right)}, \qquad q(v_n)^{(k)} = \frac{\exp\left(v_n^{(k)}/T\right)}{\sum_{i=1}^{K}\exp\left(v_n^{(i)}/T\right)}, \tag{1}$$

where $q(z_n)$ denotes the predictive distribution of the student model, while $q(v_n)$ represents the soft target distribution produced by the teacher model. KD is performed by minimizing the Kullback–Leibler (KL) Kullback & Leibler (1951) divergence between these two distributions:

$$\mathcal{L}_{\mathrm{KD}} = \mathrm{KL}(q(v_n) \,\|\, q(z_n)) = \sum_{k=1}^{K} q(v_n)^{(k)} \log \frac{q(v_n)^{(k)}}{q(z_n)^{(k)}}. \tag{2}$$

Meanwhile, the cross-entropy loss with respect to the ground-truth label is defined as:

$$\mathcal{L}_{\mathrm{CE}} = -\sum_{k=1}^{K} y_n^{(k)} \log q(z_n)^{(k)}. \tag{3}$$

The overall training objective is formulated as:

$$\mathcal{L} = (1-\alpha)\mathcal{L}_{\mathrm{CE}} + \alpha\mathcal{L}_{\mathrm{KD}}, \tag{4}$$

where $\alpha \in [0, 1]$ is a trade-off coefficient.

PID-based Adaptive Weighting. Although instantaneous error-based weighting schemes are widely adopted in KD, these approaches typically rely on limited local information when determining the supervision weight, which restricts their ability to fully capture the student model's global learning dynamics throughout the training process. In particular, two types of information are especially important yet often overlooked: the student's accumulated historical learning performance, which reflects its long-term training status, and the changing trend of its current performance, which indicates whether the student is improving or deteriorating at the present stage. From a control-theoretic perspective, these two factors correspond to the integral and derivative states, respectively, and can provide valuable guidance for more accurate and stable supervision weight adjustment.

Motivated by this observation, we introduce the proportional-integral-derivative (PID) control mechanism to model the adaptive weighting process in KD. Specifically, the control signal is constructed by jointly considering the current error, its accumulated history and its rate of change, which can be written as:

$$w(t) = K_p e(t) + K_i \int e(t)\,dt + K_d \frac{de(t)}{dt}, \tag{5}$$

where $w(t)$ denotes the control output or supervision weight, and $e(t)$ represents the error. The term $K_p e(t)$ is the proportional component, reflecting the instantaneous effect of the current error. $K_i \int e(t)\,dt$ is the integral component, which accumulates historical error to correct long-term deviation. And $K_d \frac{de(t)}{dt}$ is the derivative component, representing the trend of the error and predicting its future evolution, thereby enabling more accurate and stable control or supervision adjustment.

4 PIDSCORE

We propose to determine the balancing coefficient dynamically and individually for each sample. Specifically, PIDScore calculates the sample-specific fusion weight α_n based exclusively on the student's own state, which comprises its current prediction z_n and the corresponding ground-truth label y_n. Importantly, this computation does not require any access to the teacher model's internal states or outputs, rendering PIDScore a genuinely teacher-agnostic and plug-and-play module. The detailed procedure of PIDScore is illustrated in Figure 1.

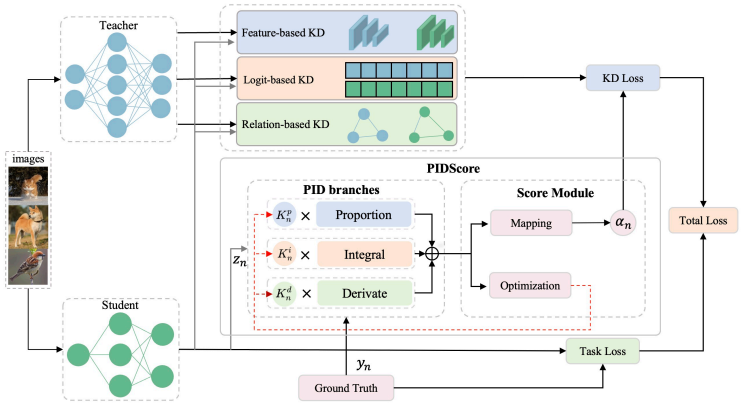

Figure 1: Framework of PIDScore.

4.1 PID Branch

The PID branch of PIDScore generates a control signal $w_n(t)$ from the student's predicted target discrepancy $e_n(t)$, where the target discrepancy $e_n(t)$ is defined as follows:

$$e_n(t) = y_n^{(k)} - z_n(t)^{(k)}, \tag{6}$$

where $y_n^{(k)}$ denotes the ground-truth label of the k-th class and $z_n(t)^{(k)}$ denotes the probability predicted by the student model for the target class at time t. This scalar discrepancy directly measures how well the student has mastered the current sample.

Why not use a combined or single loss? If we define $e_n(t)$ using a combination of $\mathcal{L}_{CE}$ and $\mathcal{L}_{KD}$, although it can to some extent integrate information from both classification supervision and distillation supervision, the resulting control signal struggles to maintain uniformity and stability. The fundamental reason is that $\mathcal{L}_{KD}$ varies significantly across different distillation methods in terms of its definition, numerical range and scale of variation, which undermines the comparability and consistency of the resulting signal. However, if only a single loss is used to define $e_n(t)$, it is difficult to fully characterize the student model's true learning state on the current sample. Relying solely on $\mathcal{L}_{CE}$ tends to bias the control process toward minimizing classification error while neglecting the supplementary benefit brought by distillation knowledge. Conversely, relying solely on $\mathcal{L}_{KD}$, on the other hand, tends to overemphasize fitting the teacher's output while weakening the contribution of ground-truth label supervision to improving the student's discriminative ability. Therefore, neither a combined loss nor a single loss is sufficient to serve as an ideal basis for characterizing the student model's learning state on the current sample. In contrast, defining $e_n(t)$ based on the predicted probability of the target class is more suitable, since it is naturally bounded in $[0, 1]$, provides good comparability and method-independence, and directly reflects the student's learning state from the student side, thereby offering a more stable and consistent basis for adaptive fusion-weight control (Empirical results are provided in Appendix D).

The PID branch then produces three control components (Proportional (P), Integral (I) and Derivative (D)) based on $e_n(t)$. The overall control signal is the sum of the three components:

$$w_n(t) = w_n^p(t) + w_n^i(t) + w_n^d(t), \tag{7}$$

where $w_n^p(t)$, $w_n^i(t)$ and $w_n^d(t)$ are the P, I and D control signals, respectively. Whereas a conventional PID controller directly adjusts $w_n(t)$, PIDScore instead uses $w_n(t)$ to guide the student model in determining the fusion weight α_n (see Section 4.2). The three branches are detailed below.

P branch. The proportional controller generates a control signal proportional to the current error $e_n(t)$ for rapid error correction:

$$w_n^p(t) = K_n^p e_n(t), \tag{8}$$

where K_n^p is a adaptable proportional coefficient (its adaptation is described in Section 4.2). The P-branch calculates the appropriate fusion weight for the student model in the current update by

leveraging the student model's instantaneous performance on the current sample. However, the P-branch relies solely on a single prediction, making it susceptible to noise interference and prone to falling into local optima. Moreover, it is unable to adapt to the dynamic requirements of different learning stages, which significantly limits the accuracy and robustness of the fusion weight calculator.

I branch. The integral branch accumulates error signals over a past period to leverage broader temporal context, thereby improving stability. The calculation formula is shown in Equation (9):

$$w_n^i(t) = K_n^i \int_0^t e_n(t)dt, \tag{9}$$

where K_n^i is a adaptable parameter. By integrating the error signals of the student model over the most recent t batches, the integral branch overcomes the limitations of relying on single information alone while effectively avoiding drastic updates caused by occasional prediction signals during the early stages of learning, thereby reducing fluctuations in the training process. Compared to using the P-branch alone, the integral branch enables more stable fusion weight calculation. A detailed analysis of the selection of the historical error window size (i.e., the time window t for the integral term and its impact on performance is provided in Appendix F.

D branch. The derivative controller generates control signals according to the rate of error variation, thereby enabling precise adjustment of the fusion weight. The dynamic evolution of the error is typically characterized by the n-th order difference:

$$w_d^n(t) = \sum_{i=0}^{n} (-1)^i \binom{n}{i} e_n(t-i). \tag{10}$$

The derivative term in conventional PID controllers only considers the variation trend between the two most recent errors, i.e., the first-order difference $(n = 1)$, which is insufficient to reflect the dynamic evolution of the student model's mastery level. Therefore, we adopt the second-order difference $(n = 2)$ to strike a balance between computational overhead and the ability to capture the student model's mastery trend over samples, while also smoothing out random fluctuations:

$$w_n^d(t) = K_n^d(e_n(t) - 2e_n(t-1) + e_n(t-2)), \tag{11}$$

where K_n^d is a adaptable parameter. See Appendix E for a comparison of first-order and second-order differences.

PIDScore introduces additional time overhead during training, and the specific training times for different integral and differential modules are provided in the Appendix.E and Appendix.F. However, in typical KD scenarios, both the teacher and student models are usually trained on cloud servers with abundant computational resources, making this extra training cost acceptable. Moreover, our method significantly improves the performance of the student model without increasing its inference time.

4.2 Score Module

The score module computes a comprehensive score $w_n(t)$ for each training sample by combining the P, I and D components (see Equation (7)). $w_n(t)$ is simultaneously used to determine the fusion weight between the teacher and student model predictions in KD, and serves as the primary basis for dynamically adjusting the PID controller gain parameters K_n^p, K_n^i and K_n^d. The method draws inspiration from fuzzy control principles, enabling adaptive adjustment of the gain parameters according to the current mastery level of each sample, thereby achieving more refined optimization of the fusion weight $w_n(t)$. The sample-specific fusion weight α_n is computed as follows:

$$\alpha_n = f\big(w_n(t)\big), \tag{12}$$

where $f(\cdot)$ is a precomputed lookup table. Specifically, after the $(t-1)$-th iteration, a comprehensive score $w_n(t) \in [0, W_m]$ is computed for each sample, where a lower score indicates better mastery and a higher score indicates poorer mastery. Since $e_n(t)$ is bounded in $[0, 1]$, the resulting $w_n(t)$ is also bounded and is further linearly rescaled to $[0, 100]$ for convenient lookup and mapping. Therefore, $W_m = 100$ in our implementation. It should be noted that W_m is not a tunable hyperparameter,

but is determined by the bounded range of $e_n(t)$ and the adopted rescaling scheme, where a lower score indicates better mastery of the sample, and a higher score indicates poorer mastery. The maximum possible score W_m is uniformly divided into five equal intervals, with threshold points: $S = [0, \Delta w, 2\Delta w, 3\Delta w, 4\Delta w, 5\Delta w], \quad \Delta w = W_m/5$, corresponding to five learning levels from $L = 1$ to $L = 5$. Since the PID objective is to minimize the error, smaller L indicates better mastery, while larger L indicates poorer mastery. Based on this reverse scoring definition, a step-size hyperparameter δ_p is introduced. Starting from an initial value K_{p0}, the current K_p for sample n is defined as:

$$K_p^n = K_{p0} - m(L) \cdot \delta_p, \tag{13}$$

where the adjustment multiplier $m(L)$ increases as the sample's mastery deteriorates, with specific values:$m(1) = 2, \quad m(2) = 3, \quad m(3) = 4, \quad m(4) = 5, \quad m(5) = 6$. This design ensures that samples with poorer mastery correspond to larger K_p^n, allowing faster error reduction, whereas well-mastered samples correspond to smaller K_p^n. After obtaining the current K_p^n, the total variation is defined as: $X = m(L) \cdot \delta_p$ Accordingly, the better the sample's mastery, the larger the value of X. This variation is not directly distributed to K_i^n and K_d^n in a fixed continuous ratio, instead, a discrete non-uniform allocation strategy is applied. Specifically, X is allocated in unit steps with priority given to K_i^n, while K_d^n is adjusted accordingly. For example, when $m(L) = 2, 3, 4$, the corresponding allocations are:$(\Delta K_i, \Delta K_d) = (\delta_p, \delta_p),\ (2\delta_p, \delta_p),\ (2\delta_p, 2\delta_p)$, and larger values of $m(L)$ follow the same pattern recursively. The updated parameters are then given by:

$$K_i^n = K_{i0} + \Delta K_i, \quad K_d^n = K_{d0} + \Delta K_d, \tag{14}$$

where K_{i0} and K_{d0} are preset base values, and it is ensured that $K_i^n \geq 0$ and $K_d^n \geq 0$. A sensitivity analysis of the hyperparameters K_{p0}, K_{i0}, K_{d0} and δ_p is provided in Appendix H.

5 Experiments

5.1 Experimental Setup

Datasets. We conduct experiments on CIFAR-100 Krizhevsky et al. (2009) , ImageNet Russakovsky et al. (2015) and MS-COCO Lin et al. (2014).

Baselines. We evaluated the performance improvement of PIDScore method on various KD methods. The feature-based KD methods include FitNet Romero et al. (2014) , RKD Park et al. (2019), CRD Peng et al. (2019), OFD Heo et al. (2019), ReviewKD Chen et al. (2021), SimKD Chen et al. (2022), AT Zagoruyko & Komodakis (2017), CAT-KD Guo et al. (2023)and TopKD Kim et al. (2024). The logit-based KD methods include KD Hinton et al. (2015), CTKD Li et al. (2023), DKD Zhao et al. (2022), MLKD Jin et al. (2023), LS Sun et al. (2024), EKD Xiang et al. (2025), SDD Wei et al. (2024) and TeKAP Hossain et al. (2025). The relation-based KD method is NRKD Gou et al. (2025).

Implementation Details. We follow the experimental settings of prior works Chen et al. (2021); Jin et al. (2023); Zhao et al. (2022). On CIFAR-100, we use SGD Sutskever et al. (2013) optimizer with 240 epochs (480 epochs for MLKD Jin et al. (2023)). The initial learning rate is 0.01 for MobileNet Howard et al. (2017); Sandler et al. (2018) and ShuffleNet Zhang et al. (2018), and 0.05 for other architectures (ResNet He et al. (2016b), WRN Zagoruyko & Komodakis (2016) and VGG Simonyan & Zisserman (2014)). More implementation details are provided in Appendix A.

5.2 Experimental Results

Results on CIFAR-100. We report the distillation results on CIFAR-100 under 14 teacher-student pairs, including 7 heterogeneous and 7 homogeneous settings. Due to space limitations, Table 1 in the main paper presents only representative results covering both heterogeneous and homogeneous teacher-student configurations. Additional experiments are provided in Appendix B, including more teacher-student pairs, ViT-based and cross-architecture distillation and MS-COCO experiments. We evaluate PIDScore on a broad range of representative KD baselines, covering feature-based, logit-based and relation-based paradigms. As shown in Table 1 and Appendix B, PIDScore consistently improves the performance of these baselines across different teacher-student settings, demonstrating its effectiveness as a general plug-in for various distillation methods.

Table 1: Top-1 Accuracy (%) on CIFAR-100 for different KD methods. KD methods are sorted by feature-based, logit-based and relation-based. We apply PIDScore to the KD methods and use ↑ to show its performance gain. **Bold** indicates the best result. Underline is the second best.

		To Similar Architecture				To Different Architecture		
Type	**Teacher**	**ResNet32×4**	**VGG13**	**WRN-40-2**	**ResNet56**	**ResNet32×4**	**ResNet32×4**	**ResNet50**
		79.42	**74.64**	**75.61**	**72.34**	**79.42**	**79.42**	**79.34**
	Student	**ResNet8×4**	**VGG8**	**WRN-40-1**	**ResNet20**	**SHN-V2**	**WRN-40-2**	**MN-V2**
		72.50	**70.36**	**71.98**	**69.06**	**71.82**	**75.61**	**64.60**
Feature	FitNet (ICLR'15)	73.50	71.02	72.24	69.21	73.54	77.69	63.16
	AT (ICLR'17)	73.44	71.43	72.77	70.55	72.73	77.43	58.58
	RKD (CVPR'19)	71.90	71.48	72.22	69.61	73.21	77.82	64.43
	CRD (CVPR'20)	75.51	73.94	74.14	71.16	75.65	78.15	69.11
	OFD (ICCV'19)	74.95	73.95	74.33	70.98	76.82	79.25	69.04
	ReviewKD (CVPR'21)	75.63	74.84	75.09	71.89	77.78	78.96	69.89
	SimKD (CVPR'22)	78.08	74.89	74.53	71.05	78.39	79.29	69.97
	CAT-KD (CVPR'23)	76.91	74.65	74.82	71.62	78.41	78.59	71.36
	CAT-KD+Ours	77.23↑0.32	74.90↑0.25	75.41↑0.59	72.05↑0.43	78.79↑0.38	79.47↑0.88	71.86↑0.50
	TopKD (ICML'24)	74.60	73.82	74.02	71.26	75.65	77.44	69.12
	TopKD+Ours	75.38↑0.78	73.97↑0.15	74.57↑0.55	71.93↑0.67	76.26↑0.61	78.01↑0.57	69.50↑0.38
Logit	KD (NeurIPS'14)	73.33	72.98	73.54	70.66	74.45	77.70	67.35
	KD+Ours	74.84↑1.51	73.68↑0.70	74.43↑0.89	72.15↑1.49	75.97↑1.52	78.26 ↑0.56	69.11↑1.76
	KD+LS (CVPR'24)	76.62	74.36	74.37	71.43	75.56	77.92	69.02
	KD+LS+Ours	76.83↑0.21	74.84↑0.48	75.00↑0.63	71.99↑0.56	76.61↑1.05	78.77↑0.85	70.30↑1.28
	DKD (CVPR'22)	76.32	74.68	74.81	71.97	77.07	78.46	70.35
	DKD+Ours	76.60↑0.28	74.78↑0.10	75.22↑0.41	72.32↑0.35	77.57↑0.50	78.94↑0.48	70.87↑0.52
	CTKD (AAAI'23)	73.39	73.52	73.93	71.19	75.37	77.66	68.67
	CTKD+Ours	73.75↑0.36	74.40↑0.88	74.21↑0.28	72.33↑1.14	76.41↑1.04	78.28↑0.62	69.49↑0.82
	MLKD (CVPR'23)	77.08	75.18	75.35	72.19	78.44	79.26	71.04
	MLKD+Ours	77.34↑0.26	**75.98**↑0.80	**76.33**↑0.98	**72.72**↑0.53	78.88↑0.44	**80.36**↑1.10	71.22↑0.18
	MLKD+LS (CVPR'24)	78.28	75.22	75.56	71.50	78.76	79.66	71.19
	MLKD+LS+Ours	**78.53**↑0.25	75.59↑0.37	76.13↑0.57	72.28↑0.78	**79.31**↑0.55	80.34↑0.68	**71.86**↑0.67
	SDD (CVPR'24)	75.33	73.66	74.16	70.72	76.49	78.54	69.31
	SDD+Ours	76.29↑0.96	74.06↑0.40	74.62↑0.46	71.60↑0.88	77.17↑0.68	78.77↑0.23	70.23↑0.92
	TeKAP (ICLR'25)	74.52	73.79	73.80	71.32	75.43	76.83	68.75
	TeKAP+Ours	75.42↑0.90	74.33↑0.54	74.66↑0.86	71.75↑0.43	76.29↑0.86	78.24↑1.41	69.51↑0.76
	EKD (ICCV'25)	77.21	74.71	73.95	69.97	77.65	77.76	70.38
	EKD+Ours	77.48↑0.27	74.95↑0.24	74.28↑0.33	70.58↑0.61	78.17↑0.52	79.32↑1.56	70.62↑0.24
Relation	NRKD (NN'25)	76.34	74.62	74.30	71.92	76.72	78.55	68.16
	NRKD+Ours	76.56↑0.22	75.19↑0.57	74.93↑0.63	72.02↑0.10	77.42↑0.70	79.35↑0.80	69.10↑0.94

Table 2: Top-1 Accuracy (%) of multi *Teacher-Student* pairs on CIFAR-100. The teacher models are RegNetY_400MF (79.77), RegNetX_400MF (79.54), ResNet32x4 (79.46) and WRN-28-4 (79.14).

Student	**Method**	**Accuracy**	**Method+Ours**	**Accuracy**	**Method**	**Accuracy**	**Method+Ours**	**Accuracy**
ResNet56	MMKD (ICME'23)	75.26	MMKD+Ours	**76.36**↑1.10	MTKD (AAAI'25)	75.11	MTKD+Ours	**75.29**↑0.18
ShuffleV2	MMKD (ICME'23)]	77.44	MMKD+Ours	**78.02**↑0.58	MTKD (AAAI'25)	78.17	MTKD+Ours	**78.42**↑0.25
MobileNetv2	MMKD (ICME'23)	73.85	MMKD+Ours	**74.04**↑0.19	MTKD (AAAI'25)	73.69	MTKD+Ours	**74.37**↑0.68
RegNetX_200MF	MMKD(ICME'23)	79.70	MMKD+Ours	**80.81**↑1.11	MTKD(AAAI'25)	80.25	MTKD+Ours	**80.61**↑0.36

Multi-Teacher Distillation on CIFAR-100. We further evaluate PIDScore in the multi-teacher distillation setting on CIFAR-100. Following MMKD Zhang et al. (2023) and MTKD Yang et al. (2025), we use four teacher models, i.e., RegNetY-400MF, RegNetX-400MF, ResNet32x4 and WRN-28-4, and conduct experiments on four student settings. As shown in Table 2, PIDScore consistently improves both multi-teacher baselines across all student models. These results show that PIDScore remains effective even when richer supervisory signals are already available, highlighting its robustness, complementarity and generality in more complex distillation scenarios.

Results on ImageNet. We compare the Top-1 and Top-5 accuracies of different methods on the ImageNet dataset in Table 3. Our method also achieves consistent performance improvements for the three KD methods on this large-scale dataset.

Feature Visualizations. Figure 2 shows the visualizations of deep feature representations. The results indicate that our proposed PIDScore improves the feature separability and discriminability of various methods, including KD, MLKD, TOPKD and NRKD.

Table 3: Top-1 Accuracy (%) and Top-5 Accuracy (%) on ImageNet for different KD methods.

Type	Teacher/Student	ResNet34/ResNet18		ResNet50/MN-V1	
	Accuracy	**Top-1**	**Top-5**	**Top-1**	**Top-5**
	Teacher	**73.31**	**91.42**	**76.16**	**92.86**
	Student	**69.75**	**89.07**	**68.87**	**88.76**
Feature	AT (ICLR'17)	70.69	90.01	69.56	89.33
	OFD (ICCV'19)	70.81	89.98	71.25	90.34
	CRD (CVPR'20)	71.17	90.13	71.37	90.41
	ReviewKD (CVPR'21)	71.61	90.51	72.56	91.00
	SimKD (CVPR'22)	71.59	90.48	72.25	90.86
	CAT-KD (CVPR'23)	71.26	90.45	72.24	91.13
Logit	KD (NeurIPS'14)	71.03	90.05	70.50	89.80
	KD+Ours	71.86↑0.83	90.18↑0.13	71.45↑0.95	90.13↑0.33
	KD+LS (CVPR'24)	71.42	90.29	72.18	90.80
	KD+LS+Ours	71.68↑0.26	90.45↑0.16	72.47↑0.29	91.08↑0.28
	DKD (CVPR'22)	71.70	90.41	72.05	91.05
	DKD+Ours	**72.06**↑0.36	**90.46**↑0.05	**72.66**↑0.61	**91.15**↑0.10

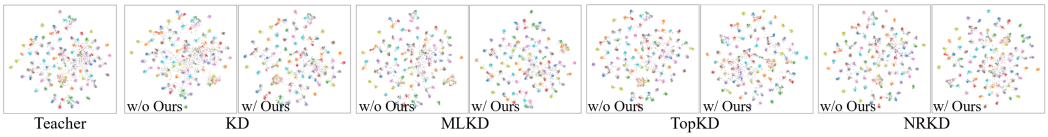

Figure 2: The t-SNE Maaten & Hinton (2008) visualization of features. The teacher and student are ResNet32×4 and WRN-40-2.

Ablation Study. To verify the effectiveness of each component in PID and analyze their performance under different distillation strategies, we conduct extensive experiments. As shown in Table 4, introducing any individual component among P, I or D alone already improves the performance of distillation methods, while the complete PID combination achieves the optimal performance. This indicates that jointly modeling all three components effectively leverages their complementary effects and thus yields superior results. Additional PID ablations, including component contributions and the hyperparameter sensitivity of PID-based weight adjustment, are provided in Appendix H.

Table 4: Ablation study on CIFAR-100.

P	I	D	Teacher/Student: ResNet32×4/WRN-40-2				ResNet56/ResNet20	
			KD	NRKD	CAT-KD	MLKD	KD	MLKD
			77.70	78.55	78.59	79.26	70.66	72.19
✓			77.82↑0.12	78.73↑0.18	78.92↑0.33	79.79↑0.53	71.31↑0.65	72.31↑0.12
	✓		77.88↑0.18	78.66↑0.11	79.14↑0.55	79.84↑0.58	71.59↑0.93	72.29↑0.10
		✓	78.14↑0.44	79.01↑0.46	78.65↑0.06	79.95↑0.69	71.74↑1.08	72.35↑0.16
✓	✓	✓	**78.26**↑0.56	**79.35**↑0.80	**79.47**↑0.88	**80.36**↑1.10	**72.15**↑1.49	**72.72**↑0.53

6 Conclusion

This paper proposed PIDScore, a student-driven adaptive weighting method inspired by PID control, to address the lack of dyanmic and fine-grained adjustment of fusion weights in KD. Unlike existing strategies that rely on fixed empirical rules or instantaneous teacher-student discrepancies, PIDScore leverages the student model's own learning feedback and jointly incorporates the current error, the accumulated historical error, and the trend of error variation to dynamically update the sample-wise distillation weight in real time. The proposed method requires no teacher-side information and enjoys the advantages of being teacher-independent, plug-and-play, and broadly applicable, making it naturally compatible with various KD frameworks. Experimental results show that PIDScore can stably improve the performance of existing distillation methods. Overall, our study demonstrates that introducing dynamic feedback mechanisms from control theory into weight allocation for KD is an effective direction for improving the efficiency and robustness of knowledge transfer.

AI USE STATEMENT

In this work, We used generative AI tools solely to polish author-written text and improve the readability and fluency of the manuscript. These tools were not used for any other aspect of this work, including research ideation, methodology development, theoretical analysis, code implementation, data preparation, experimental design or execution, or interpretation of results. The authors reviewed all AI-assisted edits to ensure that they preserved the intended meaning and technical accuracy of the manuscript. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This study uses publicly available datasets and does not involve human subjects or personal data.

REPRODUCIBILITY STATEMENT

Section 4 introduces the proposed PIDScore method. Section 5 and Appendix A provide the experimental setup and implementation details, including the datasets, model configurations, training procedures, and evaluation methods, to support the reproducibility of our results.

REFERENCES

Defang Chen, Jian-Ping Mei, Hailin Zhang, Can Wang, Yan Feng, and Chun Chen. Knowledge distillation with the reused teacher classifier. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 11933–11942, 2022.

Pengguang Chen, Shu Liu, Hengshuang Zhao, and Jiaya Jia. Distilling knowledge via knowledge review. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 5008–5017, 2021.

Jianping Gou, Xiaomeng Xin, Baosheng Yu, Heping Song, Weiyong Zhang, and Shaohua Wan. Neighborhood relation-based knowledge distillation for image classification. *Neural Networks*, 188:107429, 2025.

Ziyao Guo, Haonan Yan, Hui Li, and Xiaodong Lin. Class attention transfer based knowledge distillation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 11868–11877, 2023.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016a.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016b.

Byeongho Heo, Jeesoo Kim, Sangdoo Yun, Hyojin Park, Nojun Kwak, and Jin Young Choi. A comprehensive overhaul of feature distillation. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 1921–1930, 2019.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

Md Imtiaz Hossain, Sharmen Akhter, Choong Seon Hong, and Eui-Nam Huh. Single teacher, multiple perspectives: Teacher knowledge augmentation for enhanced knowledge distillation. In *The Thirteenth International Conference on Learning Representations*, 2025.

Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017.

Ying Jin, Jiaqi Wang, and Dahua Lin. Multi-level logit distillation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 24276–24285, 2023.

Jungeun Kim, Junwon You, Dongjin Lee, Ha Young Kim, and Jae-Hun Jung. Do topological characteristics help in knowledge distillation? In *Forty-first International Conference on Machine Learning*, 2024.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

Solomon Kullback and Richard A. Leibler. On information and sufficiency. *Annals of Mathematical Statistics*, 22(1):79–86, 1951.

Zheng Li, Xiang Li, Lingfeng Yang, Borui Zhao, Renjie Song, Lei Luo, Jun Li, and Jian Yang. Curriculum temperature for knowledge distillation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pp. 1504–1512, 2023.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: Common objects in context. In *Proc. of the European Conference on Computer Vision (ECCV)*, pp. 740–755. Springer, 2014.

Peng Lu, Abbas Ghaddar, Ahmad Rashid, Mehdi Rezagholizadeh, Ali Ghodsi, and Philippe Langlais. Rw-kd: Sample-wise loss terms re-weighting for knowledge distillation. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pp. 3145–3152, 2021.

Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov):2579–2605, 2008.

Wonpyo Park, Dongju Kim, Yan Lu, and Minsu Cho. Relational knowledge distillation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 3967–3976, 2019.

Baoyun Peng, Xiao Jin, Jiaheng Liu, Dongsheng Li, Yichao Wu, Yu Liu, Shunfeng Zhou, and Zhaoning Zhang. Correlation congruence for knowledge distillation. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 5007–5016, 2019.

Adriana Romero, Nicolas Ballas, Samira Ebrahimi Kahou, Antoine Chassang, Carlo Gatta, and Yoshua Bengio. Fitnets: Hints for thin deep nets. arxiv 2014. *arXiv preprint arXiv:1412.6550*, 2014.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252, 2015.

Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4510–4520, 2018.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

Shangquan Sun, Wenqi Ren, Jingzhi Li, Rui Wang, and Xiaochun Cao. Logit standardization in knowledge distillation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 15731–15740, 2024.

Ilya Sutskever, James Martens, George Dahl, and Geoffrey Hinton. On the importance of initialization and momentum in deep learning. In *International conference on machine learning*, pp. 1139–1147. pmlr, 2013.

Yonglong Tian, Dilip Krishnan, and Phillip Isola. Contrastive representation distillation, 2022. URL https://arxiv.org/abs/1910.10699.

Frederick Tung and Greg Mori. Similarity-preserving knowledge distillation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019.

Shicai Wei, Chunbo Luo, and Yang Luo. Scaled decoupled distillation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 15975–15983, 2024.

Liangyu Xiang, Junyu Gao, and Changsheng Xu. Evidential knowledge distillation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 2814–2824, 2025.

Chuanguang Yang, Xinqiang Yu, Han Yang, Zhulin An, Chengqing Yu, Libo Huang, and Yongjun Xu. Multi-teacher knowledge distillation with reinforcement learning for visual recognition. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 9148–9156, 2025.

Junho Yim, Donggyu Joo, Jihoon Bae, and Junmo Kim. A gift from knowledge distillation: Fast optimization, network minimization and transfer learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. *arXiv preprint arXiv:1605.07146*, 2016.

Sergey Zagoruyko and Nikos Komodakis. Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer. In *ICLR*, 2017. URL https://arxiv.org/abs/1612.03928.

Hailin Zhang, Defang Chen, and Can Wang. Adaptive multi-teacher knowledge distillation with meta-learning. In *2023 IEEE International Conference on Multimedia and Expo (ICME)*, pp. 1943–1948. IEEE, 2023.

Weijia Zhang, Yuehao Liu, Wu Ran, and Chao Ma. Cross-architecture distillation made simple with redundancy suppression. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 23256–23266. IEEE, 2025a.

Weijia Zhang, Fei Xie, Weidong Cai, and Chao Ma. Vrm: Knowledge distillation via virtual relation matching. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 2707–2717. IEEE, 2025b.

Xiangyu Zhang, Xinyu Zhou, Mengxiao Lin, and Jian Sun. Shufflenet: An extremely efficient convolutional neural network for mobile devices. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 6848–6856, 2018.

Youcai Zhang, Zhonghao Lan, Yuchen Dai, Fangao Zeng, Yan Bai, Jie Chang, and Yichen Wei. Prime-aware adaptive distillation. In *European conference on computer vision*, pp. 658–674. Springer, 2020.

Borui Zhao, Quan Cui, Renjie Song, Yiyu Qiu, and Jiajun Liang. Decoupled knowledge distillation. In *Proceedings of the IEEE/CVF Conference on computer vision and pattern recognition*, pp. 11953–11962, 2022.

Helong Zhou, Liangchen Song, Jiajie Chen, Ye Zhou, Guoli Wang, Junsong Yuan, and Qian Zhang. Rethinking soft labels for knowledge distillation: A bias-variance tradeoff perspective. *arXiv preprint arXiv:2102.00650*, 2021.

A Implementation Details

We follow the experimental settings used in previous works Hinton et al. (2015); Sun et al. (2024); Zhao et al. (2022); Li et al. (2023); Jin et al. (2023); Wei et al. (2024); Hossain et al. (2025); Xiang et al. (2025); Guo et al. (2023); Kim et al. (2024); Zhang et al. (2023); Yang et al. (2025); Zhang et al. (2025b;a). **For experiments on CIFAR-100**, we use SGD Sutskever et al. (2013) as the optimizer and train for 240 epochs, except for MLKD Jin et al. (2023), which is trained for 480 epochs. The batch size is set to 64. The initial learning rate is 0.01 for MobileNets Howard et al. (2017); Sandler et al. (2018) and ShuffleNets Zhang et al. (2018), and 0.05 for other architectures such as ResNets He et al. (2016a), WRNs Zagoruyko & Komodakis (2016), and VGGs Simonyan & Zisserman (2014). The learning rate is reduced by a factor of 0.1 at the 150th, 180th and 210th epochs. The momentum and weight decay are set to 0.9 and 5e-4, respectively. The base temperature is set to $T = 4$. **For experiments on ImageNet**, we also use SGD Sutskever et al. (2013) with a total of 100 epochs. The batch size is 512, and the initial learning rate is 0.2, which is divided by 10 at the 30th, 60th and 90th epochs. The momentum and weight decay are 0.9 and 1e-4, respectively. The base temperature is set to $T = 1$. **For experiments on MS-COCO**, we similarly use SGD as the optimizer and train for 180,000 iterations, with a total batch size of 8 and an initial learning rate of 0.01. We apply linear warm-up during the first 1,000 iterations and reduce the learning rate by a factor of 0.1 at the 120,000th and 160,000th iterations. The momentum and weight decay are set to 0.9 and 1×10^{-4}, respectively, and the distillation temperature is set to $T = 1$. In all experiments, we fix δ_p to 0.05. Since our experiments involve various knowledge distillation (KD) methods, a single parameter configuration can hardly be optimal for every method. Therefore, we evaluate the parameter K_{p0} using values selected from the set $[1, 0.9, 0.8, 0.7, 0.6]$.

B More Experiments

B.1 Cross-Architecture Knowledge Distillation

We apply PIDScore to both vanilla KD Hinton et al. (2015) and RSD Zhang et al. (2025a) on 12 heterogeneous teacher-student pairs, covering CNN-based, Transformer-based and MLP-based students. As reported in Table 5, PIDScore consistently improves both methods across all teacher-student pairs, with accuracy gains of 0.92-9.10 percentage points over KD and 0.18-1.82 percentage points over RSD. These results demonstrate the consistent effectiveness of PIDScore across different teacher-student architectures.

Table 5: Cross-architecture knowledge distillation results on CIFAR-100 across 12 heterogeneous teacher-student pairs. We apply PIDScore to KD Hinton et al. (2015) and RSD Zhang et al. (2025a) and use ↑ to show its accuracy gains in percentage points. **Bold** indicates the best result for each teacher-student pair.

Teacher	Student	Teacher Acc.	Student Acc.	KD (NeurIPS'14)	KD+Ours	RSD (ICVV'25)	RSD+Ours
CNN-based Students							
Swin-T	ResNet18	89.26	74.01	78.74	79.66↑0.92	83.92	**84.28**↑0.36
ViT-S	ResNet18	92.44	74.01	77.26	78.29↑1.03	81.50	**82.11**↑0.61
Mixer-B/16	ResNet18	87.62	74.01	77.79	78.87↑1.08	81.85	**82.74**↑0.89
Swin-T	MN-V2	89.26	73.68	74.68	77.44↑2.76	83.68	**84.17**↑0.49
ViT-S	MN-V2	92.44	73.68	72.77	77.03↑4.26	81.68	**82.06**↑0.38
Mixer-B/16	MN-V2	87.62	73.68	73.33	77.56↑4.23	81.74	**81.98**↑0.24
Transformer-based Students							
ConvNeXt-T	DeiT-T	88.42	68.00	72.99	74.74↑1.75	82.46	**83.57**↑1.11
Mixer-B/16	DeiT-T	87.62	68.00	71.36	73.51↑2.15	78.50	**80.32**↑1.82
ConvNeXt-T	Swin-P	88.42	72.63	76.44	78.19↑1.75	82.21	**82.61**↑0.40
Mixer-B/16	Swin-P	87.62	72.63	75.93	77.92↑1.99	81.28	**81.78**↑0.50
MLP-based Students							
ConvNeXt-T	ResMLP-S12	88.42	66.56	72.25	81.35↑9.10	84.21	**84.39**↑0.18
Swin-T	ResMLP-S12	89.26	66.56	71.89	79.96↑8.07	82.67	**84.48**↑1.81

B.2 Relation-Based Knowledge Distillation

We further apply PIDScore to the relation-based distillation method VRM Zhang et al. (2025b). PIDScore adjusts the loss-fusion weights while preserving the original form of relation supervision. As shown in Table 6, PIDScore improves VRM on all seven teacher-student pairs, with accuracy gains of 0.21-0.70 percentage points, demonstrating its effectiveness and compatibility with relation-based distillation.

Table 6: Effect of PIDScore on the relation-based distillation method VRM on CIFAR-100. We apply PIDScore to VRM and use ↑ to show its accuracy gains. **Bold** indicates the best result

Method	Teacher	ResNet32×4	VGG13	WRN-40-2	ResNet56	ResNet32×4	ResNet32×4	ResNet50
		79.42	74.64	75.61	72.34	79.42	79.42	79.34
	Student	ResNet8×4	VGG8	WRN-40-1	ResNet20	SHN-V2	WRN-40-2	MN-V2
		72.50	70.36	71.98	69.06	71.82	75.61	64.60
VRM (ICCV'25)		78.76	76.19	76.46	72.09	79.34	80.22	72.30
VRM+Ours		**78.97**↑0.21	**76.79**↑0.60	**76.67**↑0.21	**72.68**↑0.59	**79.69**↑0.35	**80.92**↑0.70	**72.58**↑0.28

Table 7: Top-1 Accuracy (%) on CIFAR-100 for different KD methods. KD methods are sorted by feature-based, logit-based and relation-based. We apply PIDScore to the KD methods and use ↑ to show its performance gain. **Bold** indicates the best result. Underline is the second best.

		To Similar Architecture			To Different Architecture			
Type	Teacher	WRN-40-2	ResNet110	ResNet110	ResNet32×4	WRN-40-2	WRN-40-2	VGG13
		75.61	74.31	74.31	79.42	75.61	75.61	74.64
	Student	WRN-16-2	ResNet32	ResNet20	WRN-16-2	ResNet8×4	MN-V2	MN-V2
		73.26	71.14	69.06	73.26	72.50	64.60	64.60
Feature	FitNet (ICLR'15)	73.58	71.06	68.99	74.70	74.61	68.64	64.16
	AT (ICLR'17)	74.08	72.31	70.65	73.91	74.11	60.78	59.40
	RKD (CVPR'19)	73.35	71.82	69.25	74.86	75.26	69.27	64.52
	CRD (CVPR'20)	75.48	73.48	71.46	75.65	75.24	70.28	69.73
	OFD (ICCV'19)	75.24	73.23	71.29	76.17	74.36	69.92	69.48
	ReviewKD (CVPR'21)	76.12	73.89	71.34	76.11	74.34	71.28	70.37
	SimKD (CVPR'22)	75.53	73.92	71.06	77.17	75.29	70.10	69.44
	CAT-KD (CVPR'23)	75.60	73.62	71.37	76.82	75.38	70.24	69.13
	CAT-KD+ours	76.07↑0.47	74.06↑0.44	71.78↑0.41	77.00↑0.18	75.66↑0.28	70.65↑0.41	69.58↑0.45
	TopKD (ICML'24)	75.32	73.68	71.28	74.97	75.17	68.96	68.83
	TopKD+ours	75.53↑0.21	74.03↑0.35	71.65↑0.37	75.34↑0.37	75.62↑0.45	69.25↑0.29	69.31↑0.48
Logit	KD (NeurIPS'14)	74.92	73.08	70.67	74.90	73.97	68.36	67.37
	KD+Ours	75.69↑0.77	73.98↑0.90	71.46↑0.79	75.34↑0.44	75.23↑1.26	69.17↑0.81	68.99↑1.62
	KD+LS(CVPR'24)	76.11	73.73	71.48	75.26	77.11	69.23	68.61
	KD+LS+ours	76.41↑0.30	74.09↑0.36	71.89↑0.41	76.12↑0.86	77.32↑0.21	70.03↑0.80	69.76↑1.15
	DKD (CVPR'22)	75.55	74.11	71.06	75.70	75.56	69.28	69.71
	DKD+ours	75.96↑0.41	74.57↑0.46	72.08↑1.02	76.32↑0.62	75.92↑0.36	70.21↑0.93	70.56↑0.85
	CTKD (AAAI'23)	75.45	73.52	70.99	74.57	74.61	68.34	68.50
	CTKD+ours	75.94↑0.49	74.46↑0.94	71.25↑0.26	74.79↑0.22	74.92↑0.31	68.55↑0.21	68.65↑0.15
	MLKD (CVPR'23)	76.63	74.11	71.89	76.52	77.33	70.78	70.57
	MLKD+Ours	77.31↑0.68	**75.20**↑1.09	**72.81**↑0.92	76.95↑0.43	**77.94**↑0.61	**71.85**↑1.07	70.97↑0.40
	MLKD+LS (CVPR'24)	76.95	74.32	72.27	77.38	77.68	70.50	70.94
	MLKD+LS+ours	**77.53**↑0.58	74.96↑0.64	72.54↑0.27	**77.54**↑0.16	77.89↑0.21	70.89↑0.39	**71.22**↑0.28
	SDD (CVPR'24)	75.81	73.90	71.43	75.58	74.83	69.43	69.06
	SDD+ours	76.05↑0.24	74.24↑0.34	71.77↑0.34	75.99↑0.41	75.43↑0.60	70.16↑0.73	69.28↑0.22
	TeKAP (ICLR'25)	75.21	73.09	70.82	75.26	74.99	68.65	68.27
	TeKAP+ours	75.81↑0.60	73.78↑0.69	71.57↑0.75	75.67↑0.41	76.07↑1.08	69.48↑0.83	69.10↑0.83
	EKD (ICCV'25)	75.66	72.17	71.18	76.51	76.10	68.49	69.94
	EKD+ours	76.25↑0.59	73.11↑0.94	71.43↑0.25	76.84↑0.33	76.57↑0.47	69.00↑0.51	70.20↑0.26
Relation	NRKD (NN'25)	75.94	74.04	70.22	76.06	76.18	68.06	67.43
	NRKD+ours	76.58↑0.64	74.48↑0.44	71.86↑1.64	76.37↑0.31	76.35↑0.17	69.12↑1.06	68.21↑0.78

B.3 More Experiments on CIFAR-100

We further conduct several experiments on CIFAR-100 with different teacher-student architectures. As shown in Table 7. The results show that PIDScore consistently improves the performance of these

baseline methods across different teacher-student configurations, indicating that it can serve as a general plug-in and be effectively applied to a variety of distillation methods.

B.4 More Experiments on Distilling ViT

We further verify the applicability of PIDScore to ViT distillation. Owing to its model-agnostic design, the proposed method can be seamlessly extended to ViT architectures without any modification. As shown in Table 8, PIDScore consistently yields substantial performance improvements over KD Hinton et al. (2015) across different ViT models, demonstrating its generalization capability beyond CNN-based architectures. Moreover, it is worth noting that the gains achieved by PIDScore on KD Hinton et al. (2015) substantially surpass those obtained by LS Sun et al. (2024).

Table 8: The Top-1 Accuracy (%) of KD methods on CIFAR-100. Teacher is ResNet56. ↑ denotes the performance gain over the corresponding KD baseline . **Bold** indicates the best result.

Student	Size	Vanilla	AT	LG	AutoKD	KD	KD+LS	KD+Ours
PiT-Ti	5M	73.58	76.03	78.48	78.51	75.47	78.76↑3.29	**79.63**↑4.16
PVT-Ti	13M	69.22	74.66	77.07	77.48	73.60	78.43↑4.83	**79.01**↑5.41

B.5 Extension to Object Detection

For object detection, we replace the sample-level score used in classification with an image-level score. Specifically, the predictions of sampled foreground RoIs from each image are aggregated to estimate the image-level learning state, and historical states are maintained using a stable image identifier. The resulting image-level weight is expanded to all RoIs belonging to the same image and is used to weight the RoI classification supervision and the corresponding distillation loss. The RPN and bounding-box regression losses remain unchanged.

As shown in Table 9, PIDScore improves the KD baseline by 0.68 AP, 0.92 AP_{50} and 0.48 AP_{75} for ResNet101→ResNet18. For ResNet50→MobileNetV2, the corresponding gains are 1.55, 2.31 and 1.91 points. These results demonstrate that PIDScore can be extended from image classification to object detection without changing the main Faster R-CNN pipeline.

Table 9: Object detection results on MS-COCO Lin et al. (2014). The ↑ indicate the improvements produced by PIDScore over the corresponding KD baseline.

Method	ResNet101→ResNet18			ResNet50→MN-V2		
	AP	AP_{50}	AP_{75}	AP	AP_{50}	AP_{75}
Teacher	42.04	62.48	45.88	40.22	61.02	43.81
Student	33.26	53.61	35.26	29.47	48.87	30.90
KD	33.97	54.66	36.62	30.13	50.28	31.35
KD+Ours	**34.65**↑0.68	**55.58**↑0.92	**37.10**↑0.48	**31.68**↑1.55	**52.59**↑2.31	**33.26**↑1.91

C Comparison with Existing Adaptive Fusion Strategies

To evaluate the effectiveness and compatibility of PIDScore, we compare it with ADA-KD Zhang et al. (2023), RW-KD Lu et al. (2021) and WLS-KD Zhou et al. (2021) on four homogeneous and three heterogeneous teacher–student pairs, applying each weighting strategy to both vanilla KD Hinton et al. (2015) and DKD Zhao et al. (2022). The complete results are reported in Table 10. PIDScore achieves the highest accuracy on all seven teacher-student pairs under both distillation frameworks, outperforming the best competing configurations by 0.34-0.89 percentage points for vanilla KD and 0.10-0.51 percentage points for DKD. These results demonstrate that PIDScore provides consistent performance gains across different model pairs and distillation objectives with different internal structures.

Table 10: Top-1 accuracy (%) of different adaptive fusion-weight strategies on CIFAR-100. ↑ and ↓ indicate increases and decreases in accuracy. **Bold** indicates the best result.

		To Similar Architecture				To Different Architecture		
Method	**Teacher**	**ResNet32×4**	**VGG13**	**WRN-40-2**	**ResNet56**	**ResNet32×4**	**ResNet32×4**	**ResNet50**
		79.42	**74.64**	**75.61**	**72.34**	**79.42**	**79.42**	**79.34**
	Student	**ResNet8×4**	**VGG8**	**WRN-40-1**	**ResNet20**	**SHN-V2**	**WRN-40-2**	**MN-V2**
		72.50	**70.36**	**71.98**	**69.06**	**71.82**	**75.61**	**64.60**
KD	KD	73.33	72.98	73.54	70.66	74.45	77.70	67.35
	KD+ADA	73.81↑0.48	72.95↓0.03	73.70↑0.16	71.04↑0.38	75.17↑0.72	77.59↓0.11	68.31↑0.96
	KD+RW	73.78↑0.45	71.78↓1.20	73.94↑0.40	70.86↑0.20	74.52↑0.07	77.73↑0.03	67.54↑0.19
	KD+WLS	74.16↑0.83	73.16↑0.18	73.97↑0.43	71.26↑0.60	75.29↑0.84	77.46↓0.24	68.77↑1.42
	KD+Ours	**74.84**↑1.51	**73.68**↑0.70	**74.43**↑0.89	**72.15**↑1.49	**75.97**↑1.52	**78.26**↑0.56	**69.11**↑1.76
DKD	DKD	76.32	74.68	74.81	71.97	77.07	78.46	70.35
	DKD+ADA	75.00↓1.32	74.20↓0.48	74.21↓0.60	71.81↓0.16	76.75↓0.32	78.39↓0.07	69.95↓0.40
	DKD+RW	74.51↓1.81	73.75↓0.93	73.52↓1.29	70.53↓1.44	75.15↓1.92	78.29↓0.17	68.12↓2.23
	DKD+WLS	76.20↓0.12	74.52↓0.16	74.47↓0.34	71.51↓0.46	76.23↓0.84	78.30↓0.16	70.36↑0.01
	DKD+Ours	**76.60**↑0.28	**74.78**↑0.10	**75.22**↑0.41	**72.32**↑0.35	**77.57**↑0.50	**78.94**↑0.48	**70.87**↑0.52

Table 11: Effects of different input signals for the PID controller. Top-1 accuracy (%) is reported under four distillation methods. $\mathcal{L}_{CE}$ and $\mathcal{L}_{KD}$ denote the task and distillation losses, respectively. $\mathcal{L}_G$ denotes the teacher-student ground-truth-class logit difference and $\mathcal{L}_C$ combines the proposed confidence signal with $\mathcal{L}_G$. The ↑ indicate improvements over the corresponding baseline without PIDScore.

Method	**Teacher**	**ResNet56**	**ResNet32×4**	**Method**	**Teacher**	**ResNet56**	**ResNet32×4**
		72.34	**79.42**			**72.34**	**79.42**
	Student	**ResNet20**	**WRN-40-2**		**Student**	**ResNet20**	**WRN-40-2**
		69.06	**75.61**			**69.06**	**75.61**
KD		70.66	77.70	TeKAP		71.32	76.83
KD+Ours ($\mathcal{L}_{CE}$)		70.96↑0.30	78.06↑0.36	TeKAP+Ours ($\mathcal{L}_{CE}$)		71.46↑0.14	77.77↑0.94
KD+Ours ($\mathcal{L}_{KD}$)		71.00↑0.34	77.80↑0.10	TeKAP+Ours ($\mathcal{L}_{KD}$)		71.60↑0.28	77.42↑0.59
KD+Ours ($\mathcal{L}_{CE}+\mathcal{L}_{KD}$)		71.63↑0.97	77.85↑0.15	TeKAP+Ours ($\mathcal{L}_{CE}+\mathcal{L}_{KD}$)		71.49↑0.17	77.61↑0.78
KD+Ours ($\mathcal{L}_G$)		71.19↑0.53	77.94↑0.24	TeKAP+Ours ($\mathcal{L}_G$)		71.38↑0.06	77.74↑0.91
KD+Ours ($\mathcal{L}_C$)		71.52↑0.86	77.96↑0.26	TeKAP+Ours ($\mathcal{L}_C$)		71.48↑0.16	77.66↑0.83
KD+Ours		**72.15**↑1.49	**78.26**↑0.56	TeKAP+Ours		**71.75**↑0.43	**78.24**↑1.41
MLKD		72.19	79.26	MLKD+LS		71.50	79.66
MLKD+Ours ($\mathcal{L}_{CE}$)		72.39↑0.20	80.09↑0.83	MLKD+LS+Ours ($\mathcal{L}_{CE}$)		72.00↑0.50	79.95↑0.29
MLKD+Ours ($\mathcal{L}_{KD}$)		72.30↑0.11	79.85↑0.59	MLKD+LS+Ours ($\mathcal{L}_{KD}$)		71.95↑0.45	79.88↑0.22
MLKD+Ours ($\mathcal{L}_{CE}+\mathcal{L}_{KD}$)		72.38↑0.19	79.97↑0.71	MLKD+LS+Ours ($\mathcal{L}_{CE}+\mathcal{L}_{KD}$)		71.97↑0.47	80.11↑0.45
MLKD+Ours ($\mathcal{L}_G$)		72.35↑0.16	80.12↑0.86	MLKD+LS+Ours ($\mathcal{L}_G$)		72.09↑0.59	79.79↑0.13
MLKD+Ours ($\mathcal{L}_C$)		72.45↑0.26	79.63↑0.37	MLKD+LS+Ours ($\mathcal{L}_C$)		72.19↑0.69	79.90↑0.24
MLKD+Ours		**72.72**↑0.53	**80.36**↑1.10	MLKD+LS+Ours		**72.28**↑0.78	**80.34**↑0.68

D Analysis of the Controller Input Signal

To validate the effectiveness and advantages of the proposed error-signal design, we compare the default input signal of PIDScore with five alternative signals, including $\mathcal{L}_{CE}$ and $\mathcal{L}_{KD}$. We denote their direct combination as $\mathcal{L}_{CE}+\mathcal{L}_{KD}$, the teacher-student logit difference on the ground-truth class as $\mathcal{L}_G$, and the combination of the proposed confidence-based signal and the ground-truth-class logit difference as $\mathcal{L}_C$. Although the proposed confidence-based signal and $\mathcal{L}_{CE}$ contain related information because both depend on the probability assigned by the student to the ground-truth class, they are not equivalent as PID inputs. In particular, the PID operation is not invariant to nonlinear monotonic transformations. The negative logarithm in $\mathcal{L}_{CE}$ amplifies low-confidence predictions and changes the numerical scales of both the accumulated and difference terms. Moreover, directly combining $\mathcal{L}_{CE}$ and $\mathcal{L}_{KD}$ makes the resulting signal dependent on the definition, numerical range, and variation scale of the specific distillation objective. Using either loss alone also captures only one aspect of the supervision. By contrast, the proposed confidence-based signal is bounded in $[0,1]$, varies linearly with the student's confidence in the ground-truth class, and is independent of the definition and numerical scale of the distillation objective, thereby providing a more stable and consistent input for adaptive fusion-weight control. As shown in Table 11, the default confidence-

based signal of PIDScore achieves the highest accuracy under KD, TeKAP, MLKD and MLKD+LS for both teacher-student pairs. Experimental results demonstrate that the proposed error signal is bounded and independent of the distillation method, thereby providing a stable controller input for different distillation objectives.

E Analysis of First-Order and Second-Order Differences on Derivative Terms

To further investigate the role of the derivative term in the PID controller, we conducted an ablation study to systematically compare first-order and second-order derivatives in terms of model performance and computational overhead. It should be noted that introducing a PID controller slightly increases the training time. However, in knowledge distillation tasks, model training is typically performed in cloud environments with relatively abundant computational resources, after which the trained model is deployed on resource-constrained devices. Therefore, this additional computational cost mainly arises during the offline training stage and is acceptable in practical applications.Specifically, we recorded the time consumption per training epoch and GPU memory consumption for both the first-order and second-order derivatives, as shown in Table 12. Compared with the original KD method, incorporating either the first-order or second-order derivative only slightly increases the training time per epoch. Although the second-order derivative utilizes more historical information than the first-order derivative and consequently introduces a small amount of additional computation, the overall increase in training time remains limited. Moreover, the GPU memory consumption remains identical across the original KD method and the two derivative settings, indicating that neither the first-order nor the second-order derivative introduces additional GPU memory overhead. These computational costs are limited to the training phase and do not impose any additional overhead on model inference.

Table 12: Effects of first-order and second-order differences under KD on CIFAR-100. We report Top-1 accuracy, training time per epoch on one GPU, and peak allocated GPU memory.

Method	Teacher	ResNet32×4	VGG13	WRN-40-2	WRN-40-2	ResNet56	ResNet110	ResNet110
		79.42	74.64	75.61	75.61	72.34	74.31	74.31
	Student	ResNet8×4	VGG8	WRN-40-1	WRN-16-2	ResNet20	ResNet32	ResNet20
		72.50	70.36	71.98	73.26	69.06	71.14	69.06
Accuracy								
KD		73.33	72.98	73.54	74.92	70.66	73.08	70.67
KD+Ours ($n = 1$)		74.48$_{\uparrow 1.15}$	73.43$_{\uparrow 0.45}$	73.82$_{\uparrow 0.28}$	75.14$_{\uparrow 0.22}$	71.65$_{\uparrow 0.99}$	73.62$_{\uparrow 0.54}$	71.38$_{\uparrow 0.71}$
KD+Ours ($n = 2$)		**74.84**$_{\uparrow 1.51}$	**73.69**$_{\uparrow 0.71}$	**74.43**$_{\uparrow 0.89}$	**75.69**$_{\uparrow 0.77}$	**72.15**$_{\uparrow 1.49}$	**73.98**$_{\uparrow 0.90}$	**71.46**$_{\uparrow 0.79}$
Training Time Cost (Per Epoch/GPU)								
KD		10.78 s	7.64 s	19.96 s	12.18 s	16.55 s	24.28 s	20.76 s
KD+Ours ($n = 1$)		11.28 s	7.99 s	21.22 s	13.78 s	16.89 s	24.60 s	21.23 s
KD+Ours ($n = 2$)		11.60 s	8.25 s	21.37 s	13.99 s	17.15 s	24.81 s	21.84 s
GPU Memory Cost (MB)								
KD		341.215	264.686	249.210	201.146	145.534	206.393	148.988
KD+Ours ($n = 1$)		341.215	264.686	249.210	201.146	145.534	206.393	148.988
KD+Ours ($n = 2$)		341.215	264.686	249.210	201.146	145.534	206.393	148.988

From the perspective of model performance, the results in Table 12 show that both the first-order and second-order derivatives improve model performance across different teacher–student model pairs. Furthermore, the second-order derivative consistently outperforms the first-order derivative across all evaluated model pairs. The reason is that the first-order derivative only reflects the change between two adjacent time steps, whereas the second-order derivative further characterizes the temporal evolution of the rate of change by incorporating information from three consecutive time points. As a result, it can more comprehensively capture the changing trend of learners' mastery levels. In addition, by incorporating richer historical observations, the second-order derivative can reduce the influence of random fluctuations, thereby improving the stability of the control process.

F Analysis of the Historical Error Window Size of the Integral Term

To further investigate the effect of different window sizes, we conducted an ablation study on the window setting used for accumulating historical errors in the integral term and systematically analyzed its impact on model performance and computational overhead, as shown in Table 13. Essentially, the choice of the integral window involves a trade-off among historical information utilization, control stability, and training-time cost. A smaller window requires slightly less training time and enables faster responses to recent changes, but it is insufficient to fully capture the long-term accumulation trend of errors and is therefore more susceptible to local fluctuations. In contrast, a larger window helps smooth random noise and strengthens the modeling of long-term bias, accompanied by a gradual yet modest increase in the training time per epoch. However, the peak allocated GPU memory remains unchanged across all window sizes, indicating that increasing the window size does not introduce additional GPU memory overhead. Moreover, excessive historical information does not necessarily lead to performance gains and may even reduce the model's sensitivity to current state changes.

In the experiments conducted in this study, the time window was set to $t = 3$. This configuration achieves the highest accuracy across all evaluated teacher–student model pairs. Compared with the original KD method, it introduces only a slight increase in the training time per epoch and does not incur any additional peak GPU memory consumption. Therefore, a window size of $t = 3$ provides an appropriate balance between control response speed and the utilization of historical information, while effectively capturing accumulated errors over a certain historical range without substantially compromising computational efficiency, thereby achieving robust model performance.

Table 13: Effects of different historical error window sizes under KD on CIFAR-100. We report Top-1 accuracy, training time per epoch on one GPU, and peak allocated GPU memory.

Method	**Teacher** **Student**	**ResNet32×4** **79.42** **ResNet8×4** **72.50**	**ResNet56** **72.34** **ResNet20** **69.06**	**ResNet32×4** **79.42** **SHN-V2** **71.82**	**WRN-40-2** **75.61** **ResNet8×4** **72.50**
Accuracy					
KD		73.33	70.66	74.45	73.97
KD+Ours ($t = 2$)		74.46↑1.13	71.83↑1.17	75.51↑1.06	74.72↑0.75
KD+Ours ($t = 3$)		**74.84**↑1.51	**72.15**↑1.49	**75.97**↑1.52	**75.23**↑1.26
KD+Ours ($t = 4$)		74.57↑1.24	71.33↑0.67	75.66↑1.21	74.68↑0.71
KD+Ours ($t = 5$)		74.80↑1.47	71.65↑0.99	75.40↑0.95	74.79↑0.82
KD+Ours ($t = 6$)		74.64↑1.31	72.09↑1.43	75.85↑1.40	74.65↑0.68
Training Time Cost (Per Epoch/GPU)					
KD		10.78 s	16.55 s	26.55 s	10.82 s
KD+Ours ($t = 2$)		11.18 s	16.74 s	26.74 s	11.12 s
KD+Ours ($t = 3$)		11.28 s	16.89 s	27.08 s	11.24 s
KD+Ours ($t = 4$)		11.61 s	17.04 s	27.34 s	11.37 s
KD+Ours ($t = 5$)		11.72 s	17.24 s	27.60 s	11.69 s
KD+Ours ($t = 6$)		11.95 s	17.39 s	28.10 s	11.74 s
GPU Memory Cost (MB)					
KD		341.215	145.534	514.167	256.835
KD+Ours ($t = 2$)		341.215	145.534	514.167	256.835
KD+Ours ($t = 3$)		341.215	145.534	514.167	256.835
KD+Ours ($t = 4$)		341.215	145.534	514.167	256.835
KD+Ours ($t = 5$)		341.215	145.534	514.167	256.835
KD+Ours ($t = 6$)		341.215	145.534	514.167	256.835

G Ablation Study on the Impact of Interval Partitioning Strategies

To further investigate the effect of interval granularity on model performance, we conduct an ablation study on the discretization strategy of the maximum possible score W_m. In the baseline PIDScore method, W_m is uniformly divided into five equidistant intervals, corresponding to five discrete knowledge levels. The threshold vector is defined as $S = [0, \Delta w, 2\Delta w, 3\Delta w, 4\Delta w, 5\Delta w]$, where

Table 14: Effects of the sample size in the integral term, the choice of n, and the number of historical performance records used where K_{p0}=0.8,K_{i0}=0.15,K_{d0}=0.05,δ_p=0.05

Method	Teacher	ResNet32×4	ResNet56	ResNet32×4	WRN-40-2
		79.42	72.34	79.42	75.61
	Student	ResNet8×4	ResNet20	SHN-V2	ResNet8×4
		72.50	69.06	71.82	72.50
Accuracy					
KD		73.33	70.66	74.45	73.97
KD+Ours ($m = 2$)		74.06↑0.73	71.49↑0.83	75.31↑0.86	74.99↑1.02
KD+Ours ($m = 3$)		74.63↑1.30	71.55↑0.89	75.52↑1.07	74.77↑0.80
KD+Ours ($m = 4$)		74.45↑1.11	71.73↑1.07	75.80↑1.35	75.00↑1.03
KD+Ours ($m = 5$)		**74.84**↑1.51	**72.15**↑1.59	**75.97**↑1 .52	**75.23**↑1.26
KD+Ours ($m = 6$)		74.66↑1.33	71.57↑0.91	75.78↑1.33	74.69↑0.72
KD+Ours ($m = 7$)		74.51↑1.18	71.73↑1.07	75.25↑0.70	75.10↑1.13
KD+Ours ($m = 8$)		74.55↑1.22	71.45↑0.79	75.48↑1.03	74.79↑0.82
KD+Ours ($m = 9$)		74.52↑1.12	71.65↑0.99	75.83↑1.38	74.70↑0.73
KD+Ours ($m = 10$)		74.63↑1.30	71.98↑1.32	74.97↑0.52	74.84↑0.87

$\Delta w = W_m/5$. We further generalize the interval partitioning scheme by introducing a variable partition number m. When W_m is uniformly divided into p equidistant intervals, the model correspondingly forms m discrete knowledge levels. Therefore, the number of intervals directly determines the number of levels.

It should be emphasized that changing m does not alter the weight allocation rules associated with each level. The update logic of K_n^p, K_n^i and K_n^d remains consistent with the strategy defined in the methodology. In other words, different values of m only change the discretization granularity of W_m and the resulting number of levels, while the weight allocation strategy for each level remains unchanged. Under different interval partitioning schemes, the same model output score may fall into different intervals and thus be mapped to different knowledge levels, which further leads to different magnitudes of weight updates. Therefore, this ablation study aims to examine how changes in interval granularity affect level mapping and, consequently, model performance. As shown in Table 14, the granularity of interval partitioning has a significant impact on model performance. When the number of intervals is too small and the granularity is coarse, minor variations in sample scores fail to trigger updates in interval mapping, resulting in delayed updates of the weights K_n^p, K_n^i and K_n^d. Conversely, when the number of intervals is too large and the granularity is fine, slight fluctuations in output scores can cause samples to be frequently mapped to different intervals, leading to overly frequent and excessively large weight updates, which destabilize the training process. Therefore, the number of partitions $m = 5$ should be chosen to balance the sensitivity of the mapping and the smoothness of weight updates, in order to achieve optimal model performance.

H Additional Ablation Studies

H.1 Hyperparameter Sensitivity Analysis

To further investigate the effect of key hyperparameters in PIDScore on KD performance, we conduct systematic analysis on the K_{p0}, K_{d0}, K_{i0} and δ. Since the proportional, differential and integral coefficients in PIDScore satisfy the normalization constraint $K_{p0} + K_{d0} + K_{i0} = 1$, we take K_{p0} as the primary controllable variable in the PID-related experiments and examine how different weight allocation strategies influence the distillation performance. Specifically, once K_{p0} is determined, K_{i0} and K_{d0} are no longer set independently. They are automatically assigned according to the remaining weight $1 - K_{p0}$:

$$K_{i0} = \frac{3}{4}(1 - K_{p0}), \qquad K_{d0} = \frac{1}{4}(1 - K_{p0}). \tag{15}$$

Therefore, adjusting K_{p0} essentially redefines the overall weight allocation structure of the PID components. Based on this, we conduct extensive ablation experiments on δ_p to comprehensively evaluate the effectiveness of PIDScore under different parameter combinations. As shown in Table 15., it can be observed that under all tested parameter combinations, PIDScore consistently improves the

Table 15: Effect of δ_p, K_{p0}, K_{i0} and K_{d0} on CIFAR-100. The teacher model is ResNet56 and the student model is ResNet20. The accuracy of the baseline KD Hinton et al. (2015) is 70.66%.

K_{p0}	K_{i0}	K_{d0}	δ_p									
			0.01	0.02	0.03	0.04	0.05	0.06	0.07	0.08	0.09	0.10
0.2	0.6	0.2	71.09	71.70	71.18	70.93	71.51	71.03	70.99	71.10	71.22	71.46
0.3	0.525	0.175	71.49	71.53	71.17	71.21	71.09	71.60	71.21	71.73	71.56	71.58
0.4	0.45	0.15	71.72	71.60	71.34	71.10	71.07	71.41	71.48	71.39	71.74	71.52
0.5	0.375	0.125	71.19	71.43	71.36	71.40	71.27	71.46	71.72	71.89	71.27	71.16
0.6	0.3	0.1	71.73	71.13	71.17	71.60	71.66	71.51	71.39	71.70	71.39	71.05
0.7	0.225	0.075	72.04	71.35	71.49	71.76	71.36	71.14	71.18	71.68	71.70	71.84
0.8	0.15	0.05	71.25	71.59	71.84	71.85	72.15	71.86	71.50	72.07	71.55	71.63
0.9	0.075	0.025	71.53	71.37	71.41	71.75	71.63	71.76	71.68	71.61	71.47	71.60
1.0	0	0	71.82	71.39	72.09	71.49	71.89	71.84	71.79	71.55	71.74	71.60

performance of the baseline KD method. This demonstrates that the proposed method not only brings stable performance gains over the baseline distillation framework, but also exhibits strong robustness and adaptability across different hyperparameter settings.

H.2 Analysis of the Contributions of PID Components

We evaluate the sensitivity of PIDScore to the initial PID coefficients and the adaptive update hyperparameter δ_p on seven teacher-student pairs. In both experiments, all three branch coefficients are adaptively updated during training. For each model pair, we report Top-1 accuracy and the relative contributions of the P, I and D branches to the fusion weight α as mean $\pm$ standard deviation over three random seeds.

For the sensitivity analysis of the initial PID coefficients, the three coefficients are not searched independently. Following the coefficient constraint, we select $K_{p0} \in \{1.0, 0.9, 0.8, 0.7, 0.6, 0.5\}$ and determine the other two initial coefficients using $K_{i0} = 0.75(1 - K_{p0})$ and $K_{d0} = 0.25(1 - K_{p0})$, while fixing $\delta_p = 0.05$. As shown in Table 16, the default initialization $(K_{p0}, K_{i0}, K_{d0}) = (0.8, 0.15, 0.05)$ achieves the highest mean accuracy on all seven teacher-student pairs. For each pair, the difference between the highest and lowest mean accuracies across the six initializations ranges from 0.37 to 0.73 percentage points, indicating limited sensitivity to initialization over the evaluated range. Even when $K_{i0} = K_{d0} = 0$, the measured contributions of the I and D branches remain nonzero because the adaptive mechanism updates all three coefficients during training. Therefore, this initialization does not constitute a fixed P-only ablation.

For the sensitivity analysis of the adaptive update hyperparameter, we fix the initial coefficients to $(K_{p0}, K_{i0}, K_{d0}) = (0.8, 0.15, 0.05)$ and evaluate $\delta_p \in \{0.01, 0.05, 0.10\}$. As shown in Table 17, $\delta_p = 0.05$ achieves the highest mean accuracy on all seven teacher-student pairs. For each pair, the difference between the highest and lowest mean accuracies across the three values ranges from 0.26 to 0.48 percentage points, indicating limited sensitivity to δ_p within the tested range.

To further summarize the changes in branch contributions in both experiments, we compute unweighted averages of the reported contribution means across the seven model pairs. Table 19 summarizes the average relative contributions under different initial coefficient settings. As K_{p0} decreases from 1.0 to 0.5, the average P contribution decreases from 73.27% to 26.44%, while the average I and D contributions increase from 7.62% and 19.11% to 38.93% and 34.63%, respectively. Under the default initialization, the average P, I and D contributions are 54.54%, 20.68%, and 24.78%, with a combined average I and D contribution of 45.46%. The per-pair results in Table 16 show that their combined contribution ranges from 40.32% to 54.16% across the seven model pairs, indicating that these branches account for a substantial portion of the fusion weight.

Table 18 summarizes the average relative contributions under different values of δ_p. Increasing δ_p from 0.01 to 0.10 decreases the average P contribution from 72.10% to 35.27%, while increasing the average I contribution from 15.76% to 26.17% and the average D contribution from 12.13% to 38.57%. At $\delta_p = 0.05$, the average P, I, and D contributions are 54.54%, 20.68%, and 24.78%, respectively. Together, these results show that both the initialization and δ_p substantially modulate

Table 16: Sensitivity to the initial PID coefficients with $\delta_p = 0.050$. $P(\alpha)$, $I(\alpha)$ and $D(\alpha)$ are branch contributions to α (all in %). Results are mean ± SD over three random seeds. Parentheses give model reference accuracies. **Bold** marks the best mean accuracy.

K_{p0}	K_{i0}	K_{d0}	Accuracy	$P(\alpha)$	$I(\alpha)$	$D(\alpha)$
Teacher/Student: ResNet56 (Acc=72.34) / ResNet20 (Acc=69.06)						
1.000	0.000	0.000	71.26 ± 0.17	78.55 ± 0.08	7.72 ± 0.01	13.73 ± 0.07
0.900	0.075	0.025	71.61 ± 0.14	68.54 ± 0.09	14.88 ± 0.01	16.58 ± 0.10
0.800	0.150	0.050	$\mathbf{71.95 \pm 0.19}$	59.68 ± 0.07	21.84 ± 0.02	18.48 ± 0.09
0.700	0.225	0.075	71.56 ± 0.11	48.93 ± 0.06	28.80 ± 0.03	22.27 ± 0.09
0.600	0.300	0.100	71.47 ± 0.25	39.31 ± 0.08	35.69 ± 0.04	25.00 ± 0.12
0.500	0.375	0.125	71.44 ± 0.20	30.08 ± 0.05	42.48 ± 0.021	27.44 ± 0.01
Teacher/Student: ResNet32×4 (Acc=79.42) / ResNet8×4 (Acc=72.50)						
1.000	0.000	0.000	74.35 ± 0.08	75.10 ± 0.02	7.87 ± 0.03	17.03 ± 0.03
0.900	0.075	0.025	74.42 ± 0.32	65.14 ± 0.03	14.69 ± 0.01	20.16 ± 0.03
0.800	0.150	0.050	$\mathbf{74.78 \pm 0.25}$	56.27 ± 0.01	21.26 ± 0.01	22.45 ± 0.01
0.700	0.225	0.075	74.32 ± 0.08	45.70 ± 0.01	27.76 ± 0.01	26.52 ± 0.02
0.600	0.300	0.100	74.57 ± 0.42	36.39 ± 0.03	34.16 ± 0.02	29.45 ± 0.05
0.500	0.375	0.125	74.54 ± 0.27	27.48 ± 0.02	40.34 ± 0.02	32.18 ± 0.04
Teacher/Student: ResNet32×4 (Acc=79.42) / WRN-40-2 (Acc=75.61)						
1.000	0.000	0.000	77.71 ± 0.12	71.12 ± 0.04	7.45 ± 0.00	21.43 ± 0.04
0.900	0.075	0.025	77.84 ± 0.12	61.17 ± 0.02	13.94 ± 0.00	24.89 ± 0.02
0.800	0.150	0.050	$\mathbf{78.13 \pm 0.20}$	52.39 ± 0.06	20.04 ± 0.01	27.57 ± 0.07
0.700	0.225	0.075	77.78 ± 0.19	42.13 ± 0.05	25.98 ± 0.02	31.89 ± 0.07
0.600	0.300	0.100	77.87 ± 0.16	33.31 ± 0.05	31.72 ± 0.03	34.97 ± 0.07
0.500	0.375	0.125	77.79 ± 0.34	24.96 ± 0.05	37.18 ± 0.04	37.86 ± 0.09
Teacher/Student: VGG13 (Acc=74.64) / VGG8 (Acc=70.36)						
1.000	0.000	0.000	73.04 ± 0.18	72.24 ± 0.04	7.61 ± 0.00	20.16 ± 0.04
0.900	0.075	0.025	73.15 ± 0.24	62.28 ± 0.06	14.21 ± 0.01	23.50 ± 0.06
0.800	0.150	0.050	$\mathbf{73.50 \pm 0.24}$	53.47 ± 0.03	20.48 ± 0.01	26.05 ± 0.04
0.700	0.225	0.075	73.26 ± 0.25	43.15 ± 0.03	26.62 ± 0.01	30.23 ± 0.05
0.600	0.300	0.100	73.31 ± 0.14	34.19 ± 0.01	32.55 ± 0.01	33.27 ± 0.02
0.500	0.375	0.125	73.12 ± 0.07	25.66 ± 0.02	38.24 ± 0.02	36.10 ± 0.04
Teacher/Student: ResNet32×4 (Acc=79.42) / SHN-V2 (Acc=71.82)						
1.000	0.000	0.000	74.94 ± 0.08	64.07 ± 0.17	7.40 ± 0.01	28.53 ± 0.18
0.900	0.075	0.025	75.32 ± 0.10	54.28 ± 0.13	13.18 ± 0.02	32.54 ± 0.15
0.800	0.150	0.050	$\mathbf{75.67 \pm 0.36}$	45.84 ± 0.22	18.50 ± 0.07	35.66 ± 0.29
0.700	0.225	0.075	75.43 ± 0.16	36.22 ± 0.28	23.55 ± 0.14	40.22 ± 0.42
0.600	0.300	0.100	75.43 ± 0.12	28.17 ± 0.01	28.38 ± 0.02	43.45 ± 0.03
0.500	0.375	0.125	75.21 ± 0.33	20.61 ± 0.12	32.86 ± 0.14	46.53 ± 0.26
Teacher/Student: WRN-40-2 (Acc=75.61) / WRN-40-1 (Acc=71.98)						
1.000	0.000	0.000	73.82 ± 0.41	76.35 ± 0.02	7.68 ± 0.01	15.98 ± 0.02
0.900	0.075	0.025	73.92 ± 0.09	66.32 ± 0.02	14.66 ± 0.00	19.02 ± 0.02
0.800	0.150	0.050	$\mathbf{74.18 \pm 0.31}$	56.45 ± 0.06	21.40 ± 0.01	22.15 ± 0.07
0.700	0.225	0.075	74.01 ± 0.31	46.89 ± 0.14	28.12 ± 0.04	24.99 ± 0.18
0.600	0.300	0.100	73.98 ± 0.02	37.57 ± 0.15	34.70 ± 0.08	27.73 ± 0.22
0.500	0.375	0.125	73.81 ± 0.11	28.49 ± 0.07	41.00 ± 0.06	30.51 ± 0.13
Teacher/Student: ResNet50 (Acc=79.34) / MN-V2 (Acc=64.60)						
1.000	0.000	0.000	68.53 ± 0.08	75.45 ± 0.11	7.64 ± 0.01	16.91 ± 0.11
0.900	0.075	0.025	68.51 ± 0.10	65.66 ± 0.17	14.60 ± 0.01	19.74 ± 0.18
0.800	0.150	0.050	$\mathbf{68.84 \pm 0.32}$	57.65 ± 0.07	21.23 ± 0.02	21.12 ± 0.09
0.700	0.225	0.075	68.25 ± 1.05	46.32 ± 0.30	27.91 ± 0.13	25.77 ± 0.43
0.600	0.300	0.100	68.69 ± 0.23	36.86 ± 0.23	34.29 ± 0.14	28.85 ± 0.37
0.500	0.375	0.125	68.54 ± 0.22	27.83 ± 0.06	40.39 ± 0.05	31.78 ± 0.12

the relative contributions of the PID branches, while the resulting changes in mean accuracy remain modest over the evaluated settings.

Table 17: Sensitivity to δ_p with $(K_{p0}, K_{i0}, K_{d0}) = (0.800, 0.150, 0.050)$. $P(\alpha)$, $I(\alpha)$ and $D(\alpha)$ are branch contributions to α (all in %). Results are mean ± SD over three random seeds. Parentheses give model reference accuracies. **Bold** marks the best mean accuracy.

δ_p	Accuracy	$P(\alpha)$	$I(\alpha)$	$D(\alpha)$
Teacher/Student: ResNet56 (Acc=72.34) / ResNet20 (Acc=69.06)				
0.010	71.56 ± 0.14	74.82 ± 0.04	16.15 ± 0.01	9.03 ± 0.04
0.050	$\mathbf{71.95 \pm 0.19}$	59.68 ± 0.07	21.84 ± 0.02	18.48 ± 0.09
0.100	71.60 ± 0.19	41.11 ± 0.09	28.60 ± 0.02	30.29 ± 0.12
Teacher/Student: ResNet32×4 (Acc=79.42) / ResNet8×4 (Acc=72.50)				
0.010	74.49 ± 0.12	73.19 ± 0.01	15.89 ± 0.00	10.92 ± 0.01
0.050	$\mathbf{74.78 \pm 0.25}$	56.27 ± 0.01	21.26 ± 0.01	22.45 ± 0.01
0.100	74.32 ± 0.39	36.97 ± 0.06	27.39 ± 0.01	35.64 ± 0.07
Teacher/Student: ResNet32×4 (Acc=79.42) / WRN-40-2 (Acc=75.61)				
0.010	77.92 ± 0.20	70.91 ± 0.03	15.61 ± 0.00	13.48 ± 0.03
0.050	$\mathbf{78.13 \pm 0.20}$	52.39 ± 0.06	20.04 ± 0.01	27.57 ± 0.07
0.100	77.87 ± 0.11	32.79 ± 0.09	24.74 ± 0.04	42.48 ± 0.12
Teacher/Student: VGG13 (Acc=74.64) / VGG8 (Acc=70.36)				
0.010	73.17 ± 0.45	71.61 ± 0.03	15.71 ± 0.01	12.68 ± 0.04
0.050	$\mathbf{73.50 \pm 0.24}$	53.47 ± 0.03	20.48 ± 0.01	26.05 ± 0.04
0.100	73.06 ± 0.25	33.88 ± 0.03	25.71 ± 0.01	40.42 ± 0.05
Teacher/Student: ResNet32×4 (Acc=79.42) / SHN-V2 (Acc=71.82)				
0.010	75.48 ± 0.09	67.08 ± 0.15	14.91 ± 0.02	18.02 ± 0.18
0.050	$\mathbf{75.67 \pm 0.36}$	45.84 ± 0.22	18.50 ± 0.07	35.66 ± 0.29
0.100	75.38 ± 0.32	26.16 ± 0.14	22.03 ± 0.08	51.81 ± 0.22
Teacher/Student: WRN-40-2 (Acc=75.61) / WRN-40-1 (Acc=71.98)				
0.010	73.86 ± 0.23	73.71 ± 0.06	16.05 ± 0.01	10.24 ± 0.06
0.050	$\mathbf{74.18 \pm 0.31}$	56.45 ± 0.06	21.40 ± 0.01	22.15 ± 0.07
0.100	73.79 ± 0.32	38.42 ± 0.08	27.54 ± 0.03	34.03 ± 0.10
Teacher/Student: ResNet50 (Acc=79.34) / MN-V2 (Acc=64.60)				
0.010	68.36 ± 0.52	73.41 ± 0.14	16.03 ± 0.02	10.56 ± 0.16
0.050	$\mathbf{68.84 \pm 0.32}$	57.65 ± 0.07	21.23 ± 0.02	21.12 ± 0.09
0.100	68.76 ± 0.13	37.55 ± 0.17	27.16 ± 0.04	35.29 ± 0.21

Table 18: Average relative contributions (%) of the P, I and D branches under different adaptive update hyperparameters.

δ_p	$P(\alpha)$	$I(\alpha)$	$D(\alpha)$
0.01	72.10	15.76	12.13
0.05	54.54	20.68	24.78
0.10	35.27	26.17	38.57

H.3 Stability Analysis of PID-Based Weight Adjustment

We compare the temporal variation of the fusion weights produced by the P-only controller and the complete PID controller. Total variation is defined as the sum of the absolute changes between consecutive epoch-level means. As shown in Table 20, the complete PID controller reduces the standard deviation across epochs, the signal range, and the total variation by 18.40% 24.68%, and

Table 19: Average relative contributions (%) of the P, I and D branches under different coefficient initializations.

K_{p0}	K_{i0}	K_{d0}	$P(\alpha)$	$I(\alpha)$	$D(\alpha)$
1.000	0.000	0.000	73.27	7.62	19.11
0.900	0.075	0.025	63.34	14.31	22.35
0.800	0.150	0.050	54.54	20.68	24.78
0.700	0.225	0.075	44.19	26.96	28.84
0.600	0.300	0.100	35.11	33.07	31.82
0.500	0.375	0.125	26.44	38.93	34.63

Table 20: Temporal stability of the fusion weights generated by the P-only and complete PID controllers. Lower values indicate smoother adjustment.

Controller	**Std. Across Epochs**	**Signal Range**	**Total Variation**
P-only	11.410	69.056	88.726
Full PID	**9.310**	**52.012**	**68.948**
Relative change	-18.40%	-24.68%	-22.29%

22.29%, respectively. These results indicate that incorporating the I and D branches leads to smoother and more stable weight adjustment throughout training.

I Unified Loss Weight Mapping Mechanism

Due to the differences in loss composition and weight settings across various KD methods, we do not modify the original KD loss computation logic of each method in order to achieve a unified weight mapping scheme. Instead, we introduce an additional unified weight for each loss term at the final loss aggregation stage, which reflects the importance of that loss to the current learning process of the student model. Following the implementation paradigm of mainstream distillation frameworks, we divide the losses into two categories: the hard supervision loss derived from ground-truth labels and the soft distillation loss derived from the teacher model. Furthermore, we set the mapping range of the additional weight to $[0, 2]$ rather than $[0, 1]$. This is because when the weight range is limited to $[0, 1]$, the original loss can only be suppressed under the premise of keeping the default weights of existing methods unchanged. In contrast, using the range $[0, 2]$ allows the original loss ratio to remain unchanged when the weight is set to 1, while also enabling dynamic adjustment of the relative contributions of hard supervision and soft distillation losses according to the student model's state at different training stages, thereby better satisfying the learning requirements across different periods.

J Analysis of PID Control Mechanism, Weight Allocation and Output Results

Analysis of PID Component Outputs To further examine the practical behavior of our method during training, we provide a supplementary analysis of the outputs of different PID components. As shown in Figure 3 that all components in PIDScore remain generally stable throughout training, without obvious abnormal fluctuations, indicating that the PID structure can participate in optimization in a stable manner. In particular, the derivative term exhibits relatively small variations, suggesting that it mainly plays an auxiliary regulating role without introducing noticeable disturbance to the overall training process. Overall, the observed trends are consistent with the settings described in the main text, showing that our method maintains good coordination and stability during training.

Analysis of PID Output Results To further evaluate the regulation effect of our method during training, we analyze the variation of the PID outputs. As shown in Figure 4 that the overall PID output in our method gradually decreases as training proceeds, indicating that the corresponding error is continuously reduced during optimization. This suggests that our method remains effective

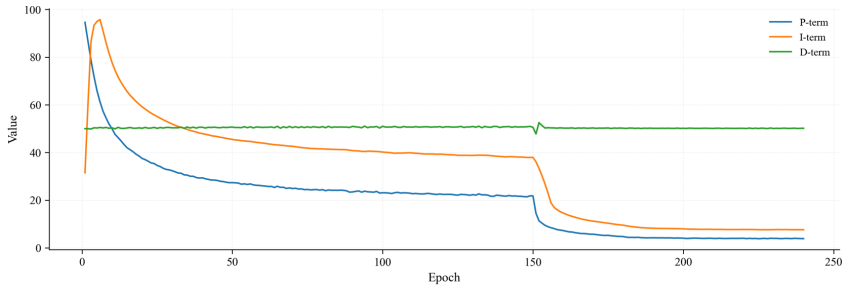

Figure 3: Variation curves of the outputs of different PID components during training

throughout training and exhibits favorable stability in the later stages. Overall, the evolution of PID outputs indicates that our method consistently plays a positive role during optimization and contributes to model convergence.

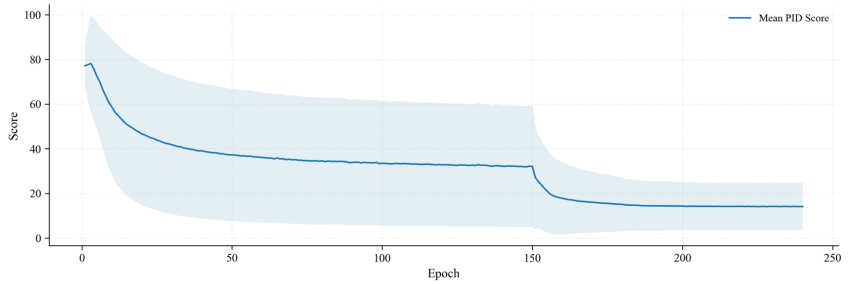

Figure 4: Distributional evolution of PID outputs in our method during training.

Analysis of PID Weights In addition to the output behavior, we further analyze the evolution of PID weights in our method. As shown in Figure 5 that K_p remains dominant throughout the training process, indicating that the current error continues to serve as the primary signal for model adjustment. Meanwhile, K_i consistently stays higher than K_d, which is in line with the weight setting in the main text and suggests that the integral term plays a stronger compensatory role than the derivative term in our method. In contrast, K_d remains relatively low and stable, indicating that the derivative term mainly functions as an auxiliary trend-regulation factor. Together, these observations show that the weight allocation in our method helps maintain training stability and is consistent with the design expectation in the main text.

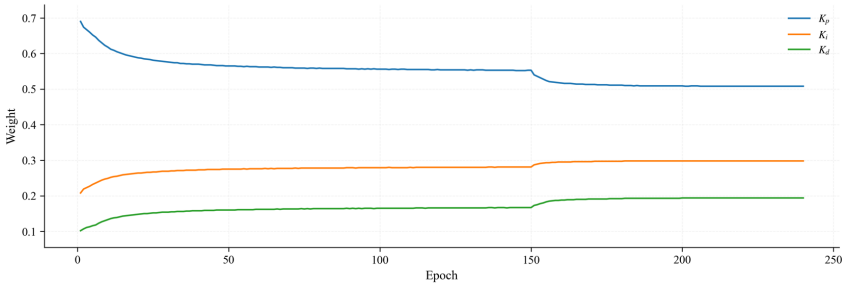

Figure 5: Evolution of the adaptive weights of different PID terms in our method over training epochs.

K LIMITATIONS

The limitation of PIDScore lies in the fact that, while it exhibits strong generality and consistently improves performance across various model architectures and knowledge distillation methods, this generality comes at the expense of deep adaptation to specific distillation frameworks. Because it is not specifically optimized for any particular method, PIDScore cannot fully exploit the unique designs of a given method in terms of loss formulation, feature interaction, or temperature strategy, which may result in suboptimal performance gains on certain methods.