# ON THE EFFICACY OF INTERMEDIATE REPRESENTATIONS IN KNOWLEDGE DISTILLATION FOR ROBUST GENERALIZATION

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

Domain generalization (DG) aims to learn a model that can generalize to unseen i.e. out-of-distribution (OOD) test domain. While large-capacity networks trained with sophisticated DG algorithms tend to achieve high robustness, they tend to be impractical in deployment. Typically, Knowledge distillation (KD) can alleviate this via an efficient transfer of knowledge from a robust teacher to a smaller student network. In our experiments, we observe that vanilla KD already provides strong OOD performance, often outperforming standalone DG algorithms. Motivated by this observation, we explore an adaptive distillation strategy that utilizes intermediate layer predictions in the student backbone to learn a meta network that effectively modulates the KD objective at an instance level. Our method adds no inference overhead and outperforms canonical ERM, and KD across vision, and text OOD generalization benchmarks.

## 1 INTRODUCTION

Deploying machine learning models in real-world scenarios requires robustness to distribution shifts (Koh et al., 2021; Huang et al., 2021), often referred to as domain generalization (DG) (Zhao et al., 2020; Robey et al., 2021) or out-of-distribution (OOD) generalization (Wald et al., 2021; Montasser et al., 2024). While larger parameter models trained with specialized DG algorithms (Gulrajani & Lopez-Paz, 2020) can achieve strong robustness, they are often prohibitively expensive in terms of computation and memory, making them impractical for deployment in resource-constrained environments.

**Efficacy of Knowledge Distillation** Knowledge distillation (KD) (Hinton et al., 2015a;b; Lopes et al., 2017), has emerged as a standard approach for improving efficiency by transferring knowledge from a large teacher model to a compact student model. Beyond efficiency aspect of KD, it has recently been explored for improving OOD robustness, where vanilla KD, as shown in (Zhou et al., 2022; 2023; Huang et al., 2023) tends to yield better OOD performance compared to models trained solely with DG algorithms. However, there is still room for improvement, as KD typically treats all samples uniformly and often overrelies on the teacher's implicit knowledge, making it prone to teacher-specific biases. Moreover, existing works that adapt KD for domain generalization primarily focus on using adversarially trained teachers (Nasery et al., 2022), multimodal teacher networks for additional supervision Huang et al. (2023) or ensemble of domain-specific teachers Zhao et al. (2025), leaving the role of the student network underexplored.

To address these limitations, we propose an adaptive distillation framework driven by a lightweight *forecaster*. Using intermediate student logits and uncertainty statistics, the forecaster estimates the correctness of the student's final prediction. This estimate determines a bounded, instance-wise coefficient on the supervised loss, assigning stronger label supervision to examples predicted to be difficult, while the distillation term retains a fixed coefficient and remains active for every example. The auxiliary readouts and forecaster are discarded after training, so the method adds no inference-time overhead.



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.

## 2 Background and Related Work

**Instance-Specific Learning and Adaptive Loss Mixing.** Prior works have explored instance-specific learning to improve neural network training under noisy conditions via instance-specific parameters such as temperature, smoothing factors, or weights. Saxena et al. (2019); Wang et al. (2018); Algan & Ulusoy (2021) introduce learnable sample and class weights to control the importance of each sample depending on reliability and label noise. Ren et al. (2019); Shu et al. (2019); Raghu et al. (2021); Jain et al. (2024) adopt meta-learning (Finn et al., 2017) to learn weighting functions. In the context of knowledge distillation, Zhao et al. (2021) propose curriculum distillation and Iliopoulos et al. (2022) reweigh student objectives to mitigate teacher bias. AMAL (Sivasubramanian et al., 2022) is closely related in spirit: it learns instance-specific mixing of supervised and teacher-matching losses using a bi-level meta-learning objective. In contrast, we (i) use early-layer readouts and uncertainty features to better capture instance difficulty under domain shift, and (ii) use an interleaved alternating schedule that requires only a held-out split from the training domains rather than a separate unbiased meta-dataset.

**Early Readouts.** Prior works on early readouts primarily focus on early-exiting (Han et al., 2021; Xu & McAuley, 2023; Laskaridis et al., 2021; Matsubara et al., 2022) to reduce inference cost using intermediate classifiers. Confidence/entropy-based early-exit criteria are studied for both language and vision models (Xin et al., 2020; Liu et al., 2020; Schwartz et al., 2020; Zhou et al., 2020). Most closely related, DEDIER (Tiwari et al., 2023) uses early readouts to modulate distillation weights via a hand-crafted rule designed to reduce featural (spurious) bias. Our method differs in that we learn the reweighting function end-to-end with a lightweight forecaster, avoiding hand-designed weighting heuristics.

**Distillation-based Domain Generalization and Stronger KD Objectives.** Distillation has shown promise for OOD generalization by transferring knowledge from robust teachers (Wang et al., 2021; Huang et al., 2023). Prior work studies improved teacher-student interactions, including gradient regularization (Wang et al., 2021), adversarially fine-tuned teachers (DAFT) (Nasery et al., 2022), and multimodal teacher supervision (Huang et al., 2023). Orthogonal to these choices, our contribution is student-centric and can, in principle, be combined with stronger KD losses (e.g., DKD (Zhao et al., 2022) or Wasserstein-based KD (Lv et al., 2024)) by treating the forecaster output as an instance-wise mixing coefficient.

## 3 Notation and Problem Setup

**Notations.** The first set of n natural numbers $\{1, 2, \ldots, n\}$ is denoted by $[n]$. The $n$-dimensional real vector space is denoted by $\mathbb{R}^n$. Vectors are typeset in lowercase bold (e.g., $\mathbf{x}$); matrices are in uppercase bold (e.g., $\mathbf{X}$); and elements are referenced by subscripts (e.g., $\mathbf{x}_i$, $\mathbf{X}_{ij}$). When needed for clarity, elements will be referenced by subscripts on square brackets (e.g., $[\mathbf{x}_1]_i$, $[\mathbf{X}_2]_{ij}$). We denote the sigmoid function by $\sigma(x) = (1 + e^{-x})^{-1}$.

**Knowledge Distillation** Knowledge distillation (Hinton et al., 2015b) provides an efficient framework for training compact models without the need to optimize over large-scale networks, thereby reducing both memory footprint and computational overhead. In this paradigm, the compact model (student) is trained to align with both the ground-truth labels and the predictive behavior of a larger reference model (teacher). Formally, the training objective for the student consists of two components:

1. **Supervised Loss** which measures the discrepancy between the student's predictions and the ground-truth labels using cross-entropy.
2. **Distillation Loss** defined as the Kullback-Leibler divergence between the student's and teacher's output distributions. This term encourages the student to replicate the softened predictive probabilities of the teacher, which encode richer information than hard labels alone and thus facilitate more effective knowledge transfer.

**Problem Setting.** Consider a standard supervised multi-classification setting with inputs $\mathbf{x} \in \mathcal{X}$ (the *input feature space*), outputs $y \in \mathcal{Y}$ (the *output space*), and the training data $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^N$ where $y_i \in [K]$, $K$ denoting the total number of classes and $\mathbf{x}_i \in \mathbb{R}^d$ denoting the $i$-th input feature. Suppose, we train a student network $\mathcal{M}_{\boldsymbol{\theta}_S}$ parameterized by $\boldsymbol{\theta}_S$ and let the corresponding

teacher network $\mathcal{M}_{\boldsymbol{\theta}_T}$ parameterized by $\boldsymbol{\theta}_T$ be used for teacher-student distillation. Let the student network consist of $L$ layers, where the output representation for the $\ell$-th layer $\ell \in [L]$ is given by $\mathbf{z}^{\ell}_{\boldsymbol{\theta}_S} = h^{\ell}_{\boldsymbol{\theta}_S}(\mathbf{x}_i) \in \mathbb{R}^{d_\ell}$ where $d_\ell$ denotes the $\ell$-th layer dimension. The final layer output logits from student network corresponding to an input $\mathbf{x}_i$ is denoted by $\mathbf{z}^{L}_{\boldsymbol{\theta}_S} := \mathcal{M}_{\boldsymbol{\theta}_S}(\mathbf{x}_i)$.

**Loss Formulation.** In KD, the student network $\mathcal{M}_{\boldsymbol{\theta}_S}$ is trained by minimizing the standard cross entropy loss between the model predictions and ground truth over the training data $\mathcal{D}$ as per Equation (1), while the teacher predictions are used to minimize the knowledge distillation term as shown in Equation (2), defined as the Kullback-Leibler (KL) divergence between teacher and student soft probabilities $p_{\boldsymbol{\theta}_T}(\mathbf{x})$ and $p_{\boldsymbol{\theta}_S}(\mathbf{x})$.

**Hard-label supervision**

$$\mathcal{L}_{\mathrm{CE}} = -\mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}}\left[\sum_{j=1}^{K} \mathbb{1}\{y=j\}\log p_{\boldsymbol{\theta}_S}(\mathbf{x})_j\right] \quad (1)$$

**Soft-target distillation**

$$\mathcal{L}_{\mathrm{KD}} = \tau^2 \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}}\left[D_{\mathrm{KL}}\big(p^{\tau}_{\boldsymbol{\theta}_T}(\mathbf{x}) \,\|\, p^{\tau}_{\boldsymbol{\theta}_S}(\mathbf{x})\big)\right] \quad (2)$$

where $p^{\tau}_{\boldsymbol{\theta}}(\mathbf{x}_i)_j = \frac{\exp(\mathcal{M}_{\boldsymbol{\theta}}(\mathbf{x}_i)_j/\tau)}{\sum_{k=1}^{K}\exp(\mathcal{M}_{\boldsymbol{\theta}}(\mathbf{x}_i)_k/\tau)}$ denotes $j$-th class probability where $j \in [K]$. Here, $\tau$ denotes the temperature used to soften the output probability distribution of both the student and the teacher, as well as rescale the gradients to allow the distillation signal to remain balanced against $\mathcal{L}_{\mathrm{CE}}$. Generally, a higher temperature allows for smoother distributions, which capture richer inter-class relationship from the teacher (Hinton et al., 2015a). This allows the student to capture more nuanced relationships between classes rather than focusing on a single class representing the highest probability.

**Student Training Objective.** The student network's loss is denoted by $\mathcal{L}^{(S)}(\mathcal{D};\boldsymbol{\theta}_S)$, where

$$\mathcal{L}^{(S)}(\mathcal{D};\boldsymbol{\theta}_S) = \lambda_{\mathrm{CE}}\mathcal{L}_{\mathrm{CE}} + \lambda_{\mathrm{KD}}\mathcal{L}_{\mathrm{KD}} \quad (3)$$

Here, $\mathcal{L}^{(S)}(\mathcal{D};\boldsymbol{\theta}_S)$ is expressed as a weighted combination of $\mathcal{L}_{\mathrm{CE}}$ and $\mathcal{L}_{\mathrm{KD}}$, with $\lambda_{\mathrm{CE}} \in \mathbb{R}^+$ and $\lambda_{\mathrm{KD}} \in \mathbb{R}^+$ being seen as two degrees of freedom, controlling the contribution of each loss component (Srivastava et al., 2015). The two degrees of freedom can be reduced to a single degree of freedom by bounding $\lambda_{\mathrm{CE}}$ and $\lambda_{\mathrm{KD}}$ with the condition: $\lambda_{\mathrm{CE}} + \lambda_{\mathrm{KD}} = 1$.

*Soft Distillation*. Moreover, in the absence of labeled data to train the student model, the distillation reduces to a soft-distillation setting, where $\lambda_{\mathrm{CE}} = 0$ . In this case, the soften teacher probabilities form the only source of supervision for the student model (Lopes et al., 2017).

**Student Capacity Limits and Teacher Confidence.** The student's ability to learn the teacher's deeper representational space is inherently bounded, and the student may fail to capture richer information beyond a threshold due to limited capacity (Cho & Hariharan, 2019). On the other hand, larger teachers are over-confident, yielding higher target logits and lower variance in predictions, resulting in less distinctive incorrect-class softmax probabilities. In this case, if distillation temperature is increased, teacher guidance becomes weaker, smoothing strengthens due to the softened distribution, and class discriminability measured by variance of incorrect-class probability initially rises then falls off. Together, both the student capacity and teacher confidence restrict the effectiveness of knowledge transfer (Li et al., 2022).

### 3.1 OOD Generalization in Distilled Models

**Bottlenecks with Vanilla KD.** While distillation serves as a good approach for model compression, the student model can be interpreted as a small surrogate of the teacher, overfitted to the teacher's learned patterns. The overfitting and the student model's limited capacity typically fails to generalize well with OOD samples at inference time (Yue et al., 2023). While there exist OOD generalization algorithms targeted towards OOD robustness in neural network (Gulrajani & Lopez-Paz, 2020), the lack of representation capacity affects the ability of the student to directly benefit from these algorithms.

**General Solution.** A general solution to improve generalization ability of the student is to train the teacher network with specialized domain generalization (DG) algorithms (Gulrajani & Lopez-Paz, 2020) or adversarial training (Goodfellow et al., 2015; Nasery et al., 2022) for OOD robustness. During distillation, the robust teacher can act as a "shortcut" that helps the student model bypass the requirement of complex, robust training. The student model inherits the teacher's ability to produce

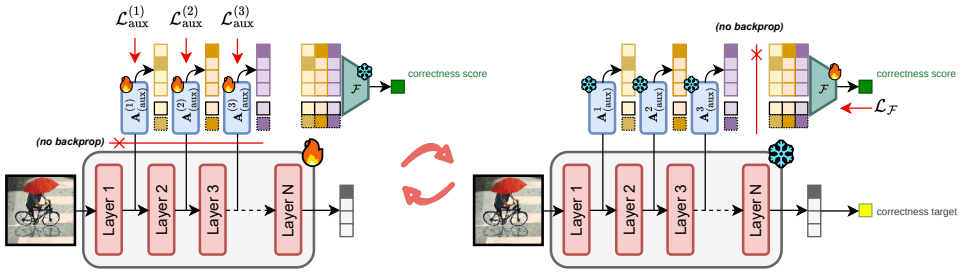


Figure 1: Overview of adaptive knowledge distillation that leverages intermediate layer predictions and uncertainty measures to learn a meta-network *forecaster* which modulates contribution of ground-truth supervision and teacher supervision in the distillation process at an instance level. Here, the student network, early layer auxiliary networks and *forecaster* are trained as per Algorithm 1.

well-calibrated outputs for seen and unseen input instances. This helps the student to be robust to distribution shifts in a deployment environment.

### 3.2 Early Readouts

Early-layer confidences in a neural network offer valuable insights about an input sample (Baldock et al., 2021). Low confidences at intermediate layers can suggest that the sample is complex or ambiguous, potentially making it difficult for a model to classify correctly. Conversely, higher confidence in the early layers indicate that the low-level representations already align well with a specific class, increasing the likelihood of a correct prediction from the network (Teerapittayanon et al., 2017). Motivated by these observations, we hypothesize that we can use intermediate-layer uncertainty as a model-dependent estimate of sample difficulty to learn to focus on examples that are difficult or ambiguous, consistent with the broader principle of emphasizing hard examples during training (Lin et al., 2018).

## 4 Proposed Methodology

In this work, we build upon the general solution outlined in Section 3.1 under the standard offline teacher–student distillation setting, introducing modifications primarily to the student architecture. First, we attach auxiliary networks a.k.a. *probes* on intermediate feature representations at specific layers of an $L$-layer student, denoted by $\mathcal{E} \subseteq [L]$. These auxiliary heads produce intermediate-layer predictions whose uncertainty statistics serve as signals of instance difficulty. Second, we introduce a meta-network that utilizes prediction statistics from the auxiliary classifiers to modulate the distillation gradients for each training instance. Unlike the use of hand-crafted weighting rules (Wang et al., 2021; Tiwari et al., 2023), the meta-network learns the reweighting mechanism in a data-driven manner using student's own internal representations.

Figure 1 provides a brief overview of our approach where we incorporate intermediate-layer predictions and uncertainty measures from the student model to dynamically reweight the distillation objective. In the following sub-sections, we elaborate on design and functionality of the auxiliary networks and the *forecaster* meta-network.

### 4.1 Auxiliary Network: Intermediate Layer Confidence Probe

**Design of Auxiliary Networks.** For each intermediate layer $\ell \in \mathcal{E}$, we attach a light-weight classifier $\mathbf{A}^{(\ell)}_{\text{aux}}(\bullet\,; \boldsymbol{\varphi}_\ell) : \mathbb{R}^{d_\ell} \to \boldsymbol{\Delta}^{K-1}$, where $\boldsymbol{\Delta}^{K-1}$ denotes the probability simplex over $K$ classes and the $\ell$-th auxiliary classifier is parameterized by $\boldsymbol{\varphi}_\ell$. The input to $\mathbf{A}^{(\ell)}_{\text{aux}}$ is the intermediate representation $\mathbf{z}^{\ell}_{\boldsymbol{\theta}_S}$, the $\ell$-th layer representation from the student network. The role of the auxiliary classifiers is to estimate the discriminative capacity of the intermediate representations up to layer $\ell$,

and their softmax probabilities serve as measures of early-layer prediction confidence. Such auxiliary predictors have been shown to be effective both for improving gradient flow and discriminative quality of learned features during training (Szegedy et al., 2015; Lee et al., 2015), and are also adopted for post-hoc analysis of representation quality in neural networks (Alain & Bengio, 2016).

**Training Objective.** On the training split $\mathcal{D}$, the auxiliary networks are trained using the same standard classification loss as the main network.

$$\mathcal{L}_{\text{aux}}^{(\ell)}(\boldsymbol{\varphi}_\ell) = \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}_s}\left[-\sum_{j=1}^{K}\mathbb{1}\{y=j\}\log\Big(\mathbf{A}_{\text{aux}}^{(\ell)}(\mathbf{z}_{\boldsymbol{\theta}_S}^{\ell}(\mathbf{x});\boldsymbol{\varphi}_\ell)\Big)_j\right] \tag{4}$$

**Confidence Estimation.** We derive uncertainty measures to quantify prediction confidence at early layers using the softmax outputs $(\mathbf{p}(\varphi_\ell))$ of the auxiliary networks. Specifically, we compute entropy $\mathcal{H}(\mathbf{p}(\varphi_\ell))$, maximum probability $c^{(\ell)}$, and confidence margin $(\varepsilon)$ (Tiwari et al., 2024; Xin et al., 2020; Liu et al., 2020) to quantify sample ambiguity using prediction confidence. The maximum probability and confidence margin of the $\ell$-th auxiliary classifier are quantified by,

$$c_i^{(\ell)} = p_{i,(1)}^{(\ell)}, \qquad \varepsilon_i^{(\ell)} = p_{i,(1)}^{(\ell)} - p_{i,(2)}^{(\ell)} \tag{5}$$

where $p_{\max}(\boldsymbol{\varphi}_l) = \max(p_1(\boldsymbol{\varphi}_l), p_2(\boldsymbol{\varphi}_l), \ldots, p_K(\boldsymbol{\varphi}_l))$ and $p_j(\cdot)$ indicates the probability of the $j$-th label, $\forall j \in [K]$.

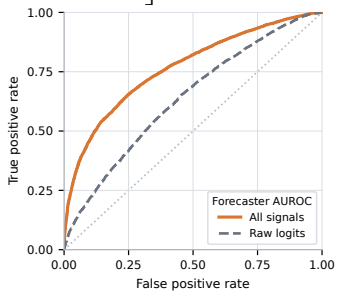


Figure 2: Forecaster correctness ROC on held-out OOD data, comparing raw readout logits against logits augmented with entropy, maximum confidence, and confidence margin.

### 4.2 Forecaster Meta-Network

Deep neural networks are prone to overfitting under label noise or class imbalance, where fixed reweighting schemes often fail due to their reliance on handcrafted weighting functions and hyperparameters (Tiwari et al., 2023; Wang et al., 2021). To overcome these limitations, we instead learn the reweighting function directly from data via a lightweight neural network, which we term the *forecaster*. The *forecaster* provides a flexible data-driven mechanism that generalizes beyond handcrafted weighting rules. In this work, we leverage the *forecaster* module to adaptively map sample-level statistics from the auxiliary networks to reweight the influence of the teacher network in the distillation process. We denote the *forecaster* as $\mathcal{F}(\bullet;\boldsymbol{\psi}^f)$, parameterized by $\boldsymbol{\psi}^f$.

**Design of Forecaster.** The *forecaster* is implemented as a lightweight classifier over features from the intermediate student layers. For each instance, we concatenate the raw logits from every auxiliary classifier with its entropy, maximum probability, and confidence margin,

$$\begin{aligned}\mathbf{u}_i &= \underset{\ell\in\mathcal{E}}{\text{Concat}}\left[\mathbf{a}_i^{(\ell)}, \mathcal{H}_i^{(\ell)}, c_i^{(\ell)}, \varepsilon_i^{(\ell)}\right] \\ w_{\boldsymbol{\psi}^f,i} &= \sigma\big(\mathcal{F}(\mathbf{u}_i;\boldsymbol{\psi}^f)\big)\end{aligned} \tag{6}$$

where $w_{\boldsymbol{\psi}^f,i} \in [0,1]$ estimates the likelihood that the student's final prediction is correct and therefore serves as a proxy for sample difficulty.

**Training Objective.** We train the *forecaster* jointly with the student $\mathcal{M}_{\boldsymbol{\theta}_S}$ and the auxiliary networks $\mathbf{A}_{\text{aux}}^{(\ell)}(\bullet;\boldsymbol{\varphi})$ using a binary classification objective, where the *forecaster* learns to predict whether student's prediction for a given input is correct. Specifically, we minimize $\mathcal{L}^{(f)}(\boldsymbol{\theta}_S;\boldsymbol{\psi}^f)$ on a held-out validation split $\mathcal{D}_f$.

$$\mathcal{L}^{(f)}(\boldsymbol{\psi}^f) = \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}_f}\left[-y_{\boldsymbol{\theta}_S}^g(\mathbf{x},y)\log w_{\boldsymbol{\psi}^f}(\mathbf{x}) - \Big(1-y_{\boldsymbol{\theta}_S}^g(\mathbf{x},y)\Big)\log\big(1-w_{\boldsymbol{\psi}^f}(\mathbf{x})\big)\right] \tag{7}$$

where $y_{\boldsymbol{\theta}_S}^g(\mathbf{x},y) = \mathbb{1}\left\{\arg\max_j[\mathcal{M}_{\boldsymbol{\theta}_S}(\mathbf{x})]_j = y\right\}$ is the binary correctness target and $w_{\boldsymbol{\psi}^f}(\mathbf{x}) \in (0,1)$ is the *forecaster* output parameterized by $\boldsymbol{\psi}^f$.

Ultimately, the student network is trained by minimizing,

$$\mathcal{L}^{(S)}(\mathcal{X}_\mathcal{B},\mathcal{Y}_\mathcal{B};\boldsymbol{\theta}_S) = \frac{1}{|\mathcal{B}|}\sum_{i\in\mathcal{B}}\Big[\widetilde{w}_i\,\mathcal{L}_{\text{CE}}[i] + \lambda_{\text{KD}}\,\mathcal{L}_{\text{KD}}[i]\Big] \tag{8}$$

**Algorithm 1:** Interleaved Training Schedule

**Models:** student $\mathcal{M}_{\boldsymbol{\theta}_S}$; fixed teacher $\mathcal{M}_{\boldsymbol{\theta}_T}$; readouts $\{\mathbf{A}_{\ell,\varphi_\ell}\}_{\ell\in\mathcal{E}}$; forecaster $\mathcal{F}_{\boldsymbol{\psi}^f}$
**Data:** student split $\mathcal{D}_s$; forecaster split $\mathcal{D}_f$; steps $T_s, T_f$
Initialize $\boldsymbol{\theta}_S$, $\{\varphi_\ell\}_{\ell\in\mathcal{E}}$, and $\boldsymbol{\psi}^f$
**while** *not converged* **do**

// Student and readout updates; freeze $\boldsymbol{\psi}^f$

**for** $t \leftarrow 1$ **to** $T_s$ **do**

Sample $\mathcal{B}_s \sim \mathcal{D}_s$

$\mathbf{z}_T \leftarrow \mathcal{M}_{\boldsymbol{\theta}_T}(\mathcal{B}_s)$; $[\{\mathbf{a}_i^{(\ell)}\}_{\ell\in\mathcal{E}}, \mathbf{z}_S] \leftarrow \mathcal{M}_{\boldsymbol{\theta}_S}(\mathcal{B}_s)$

$\mathbf{u}_i \leftarrow \mathrm{Concat}_{\ell\in\mathcal{E}}[\mathbf{a}_i^{(\ell)}, \mathcal{H}_i^{(\ell)}, c_i^{(\ell)}, \varepsilon_i^{(\ell)}]$

$w_{\boldsymbol{\psi}^f,i} \leftarrow \sigma\big(\mathcal{F}(\mathbf{u}_i;\boldsymbol{\psi}^f)\big)$; $\widetilde{w}_i \leftarrow \mathrm{clip}\big(1 - w_{\boldsymbol{\psi}^f,i}, w_{\min}, w_{\max}\big)$

$\boldsymbol{\theta}_S \leftarrow \boldsymbol{\theta}_S - \eta_S \nabla_{\boldsymbol{\theta}_S} \mathcal{L}^{(S)}$ // Eq. 8

$\varphi_\ell \leftarrow \varphi_\ell - \eta_A \nabla_{\varphi_\ell} \mathcal{L}_{\mathrm{aux}}^{(\ell)}, \forall \ell \in \mathcal{E}$ // Eq. 4

// Forecaster updates; freeze $\boldsymbol{\theta}_S, \{\varphi_\ell\}$

**for** $t \leftarrow 1$ **to** $T_f$ **do**

Sample $\mathcal{B}_f \sim \mathcal{D}_f$

$y_i^g \leftarrow \mathbb{1}\Big\{\arg\max_j [\mathbf{z}_S]_{ij} = y_i\Big\}, i \in \mathcal{B}_f$

$\boldsymbol{\psi}^f \leftarrow \boldsymbol{\psi}^f - \eta_F \nabla_{\boldsymbol{\psi}^f} \mathcal{L}^{(f)}$ // Eq. 7

where $w_{\boldsymbol{\psi}^f,i} = \sigma(z_{f,i})$ and $\widetilde{w}_i = \mathrm{clip}(1 - w_{\boldsymbol{\psi}^f,i}, w_{\min}, w_{\max})$. The bounds $0 < w_{\min} \leq w_{\max} \leq 1$ retain a non-zero weight for each sample while limiting the influence of individual examples. This weighting emphasizes hard-label supervision in lower-confidence samples, whereas the distillation loss is applied uniformly across all instances.

## 5 Interleaved Schedule

We adopt an interleaved training schedule, in which the student, auxiliary networks and the *forecaster* are optimized in alternating phases. For $T_s$ steps, we update the student model and the auxiliary networks using the training split of the training domains. Here, the *forecaster* parameters remain frozen, and it is used to provide instance-level weights to guide the distillation objective. Subsequently, for next $T_f$ steps, the *forecaster* is updated on a held-out validation split from the training data of the training domain(s) with the student network and the auxiliary classifiers frozen. These two phases are repeated until convergence. Algorithm 1 summarizes the interleaved schedule.

Importantly, the interleaved training schedule between the student network and the *forecaster* can be viewed as a bi-level optimization problem given by

$$\begin{aligned} &\min_{\boldsymbol{\psi}^f} \mathcal{L}^{(f)}\big(\boldsymbol{\theta}_S^\star(\boldsymbol{\psi}^f);\, \mathcal{D}_v\big) \\ &\text{s.t. } \boldsymbol{\theta}_S^\star(\boldsymbol{\psi}^f) = \arg\min_{\boldsymbol{\theta}_S} \frac{1}{|\mathcal{B}|} \sum_{i\in\mathcal{B}} [\widetilde{w}_i \mathcal{L}_{\mathrm{CE}}[i] + \lambda_{\mathrm{KD}} \mathcal{L}_{\mathrm{KD}}[i]] \end{aligned} \tag{9}$$

Rather than solving the bilevel problem directly, we employ the interleaved schedule in Algorithm 1 as a practical heuristic.

## 6 Experiments

**Evaluation.** We evaluate our methodology on OOD classification benchmarks spanning two modalities: vision and text. For vision, we use DomainBed (Gulrajani & Lopez-Paz, 2020); for text, we use representative ID–OOD task pairs from GLUE-X (Yang et al., 2023). We compare against vanilla KD and ERM, and report classification accuracy.

**Model Pool.** In our KD experiments, we distill from a larger teacher model. We experiment with both ResNet (He et al., 2015) and Vision Transformer (Dosovitskiy et al., 2021) as the teacher network. Specifically, we use `ResNet-50` and `ViT-L/16` as teacher network in the distillation process. For the student network, we select the smaller capacity models from the ResNet family: ResNet-10 and ResNet-18. For ViT-based student, we utilize the `ViT-B/16` model. For transformer-based natural language tasks, we leverage Encoder-only `bert-base-uncased` (Devlin et al., 2019) as the teacher network. In this case, the student network is a 4-layer compact BERT variant: `google/bert_uncased_L-4_H-256_A-4` (Turc et al., 2019).

**Datasets.** We evaluate and compare our method on five benchmark datasets, including OfficeHome (Venkateswara et al., 2017), PACS (Li et al., 2017), and DomainNet. For DomainNet, we use the 4 domains (Real, Clipart, Painting, Sketch) and 126 classes subset following (Saito et al., 2019). For text-classification benchmarks, we choose the tasks of (1) paraphrase identification with MPRC as ID split and QQP as OOD split, and (2) natural language inference (NLI) with MNLI as ID split and MNLI-mismatched as OOD split. Table 1 provides the details of the domains, instances, and label counts for each dataset.

| Dataset | Domains | Instances | Classes |
|---|---|---|---|
| **Vision** | | | |
| DomainNet-126 | 4 | 142,334 | 126 |
| OfficeHome | 4 | 15,588 | 65 |
| PACS | 4 | 9,991 | 7 |
| **Text** | | | |
| Paraphrase | 2 | 44,118 | 2 |
| NLI | 2 | 402,517 | 3 |

Table 1: Dataset statistics used in our evaluation.

**Baselines.** We compare our method against three baselines. We include the Empirical Risk Minimization (ERM) (Vapnik, 1998) as a canonical domain generalization algorithm, supported by findings in Gulrajani & Lopez-Paz (2020). Additionally, we include vanilla KD (Hinton et al., 2015b), where the student is trained to mimic the teacher without any architectural modification on the student network. To compare against a hand-crafted weighing mechanism for adaptive KD, we compare our approach against our reproduction of KDDG (Wang et al., 2021).

Table 2: Domain-wise OOD test accuracy (%) across student architectures. Distillation methods use a ResNet-50 teacher. Each result shows the mean with ± standard deviation below, over three seeds. Checkpoints are selected using source-domain validation sets. Bold and underline identifies the highest and second-highest measured mean within each student–dataset group.

| | | DomainNet$_{126}$ | | | | | PACS | | | | | OfficeHome | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Student** | **Method** | Clipart | Painting | Real | Sketch | **Avg.** | Art | Cartoon | Photo | Sketch | **Avg.** | Art | Clipart | Product | Real World | **Avg.** |
| *ResNet-10* | ERM | 51.27 ± 0.93 | 42.30 ± 0.47 | 48.25 ± 0.27 | 45.80 ± 1.22 | 46.90 ± 0.47 | 60.18 ± 1.46 | **70.40** ± 2.21 | 79.92 ± 2.25 | 67.93 ± 0.96 | 69.61 ± 0.34 | 30.11 ± 1.06 | 31.37 ± 1.76 | 49.68 ± 0.63 | 48.63 ± 0.65 | 39.95 ± 0.55 |
| | KDDG | 54.90 ± 1.24 | 46.63 ± 0.49 | 52.22 ± 0.34 | 48.89 ± 0.82 | 50.66 ± 0.71 | 62.17 ± 0.58 | 66.77 ± 0.53 | 80.84 ± 0.13 | 66.47 ± 3.24 | 69.06 ± 0.82 | 31.86 ± 1.13 | 33.62 ± 1.16 | 53.25 ± 0.89 | 52.75 ± 1.49 | 42.87 ± 1.16 |
| | KD | 56.00 ± 0.21 | 47.70 ± 0.37 | 53.67 ± 0.27 | 50.65 ± 0.45 | 52.01 ± 0.16 | 67.20 ± 2.06 | 67.73 ± 2.82 | **82.81** ± 0.31 | **69.33** ± 1.25 | 71.77 ± 0.41 | 37.95 ± 1.57 | 37.24 ± 1.31 | 58.48 ± 0.59 | **58.69** ± 0.80 | 48.09 ± 0.66 |
| | **Ours** | **56.93** ± 0.61 | **48.53** ± 1.02 | **53.73** ± 0.57 | **52.73** ± 1.45 | **52.98** ± 0.54 | **70.31** ± 2.08 | 69.35 ± 2.31 | 82.41 ± 1.53 | 68.76 ± 0.74 | **72.71** ± 0.79 | **39.65** ± 0.73 | **38.78** ± 0.81 | **58.62** ± 0.99 | 57.90 ± 0.59 | **48.74** ± 0.63 |
| *ResNet-18* | ERM | 59.88 ± 0.37 | 51.95 ± 0.57 | 55.88 ± 1.07 | 55.61 ± 1.24 | 55.83 ± 0.69 | 74.94 ± 3.36 | 71.73 ± 2.06 | 89.37 ± 0.49 | 73.91 ± 5.24 | 77.49 ± 2.07 | 42.41 ± 2.34 | 46.32 ± 1.56 | 62.19 ± 0.93 | 63.73 ± 0.86 | 53.66 ± 0.66 |
| | KDDG | 63.71 ± 1.59 | 56.60 ± 0.34 | 61.37 ± 0.34 | 58.72 ± 1.55 | 60.10 ± 0.41 | 76.21 ± 2.41 | 75.44 ± 2.03 | 90.49 ± 1.82 | 74.50 ± 4.42 | 79.16 ± 0.84 | 47.08 ± 1.04 | 48.66 ± 0.78 | 67.80 ± 0.38 | 66.35 ± 1.65 | 57.47 ± 0.15 |
| | KD | **65.92** ± 0.56 | 59.06 ± 0.71 | **66.34** ± 0.56 | **60.51** ± 1.16 | **62.96** ± 0.32 | 82.73 ± 1.59 | **77.22** ± 1.10 | **94.09** ± 0.74 | 77.78 ± 1.34 | 82.95 ± 0.65 | **54.39** ± 0.21 | **50.47** ± 0.91 | **70.68** ± 0.16 | **71.70** ± 0.42 | **61.81** ± 0.25 |
| | **Ours** | 65.05 ± 0.08 | **59.62** ± 0.37 | 65.73 ± 0.18 | 60.06 ± 0.35 | 62.62 ± 0.06 | **84.20** ± 0.48 | 76.33 ± 0.56 | 93.94 ± 0.64 | **79.17** ± 1.33 | **83.41** ± 0.12 | 54.09 ± 0.67 | 49.37 ± 0.20 | 70.44 ± 0.41 | 71.47 ± 0.18 | 61.34 ± 0.10 |

**Training Protocol.** For a dataset comprising $N$ domains, we adopt the leave-one-domain-out setup where the network is trained on $N-1$ domains and evaluated on the held-out domain. We reserve 20% from each of training domain as a validation set for model selection. In our method, remaining 80% training split is further divided, with 90% used to train the student backbone and auxiliary networks, and 10% held out to train the forecaster. We leverage gold validation splits to train the *forecaster* in our text-modality evaluations. Within the distillation setup.

**Auxiliary Network Design.** For the ResNet student, each residual block stage yields feature maps that encode progressively abstract representations of the input (He et al., 2015). To obtain class-wise representations from these intermediate features, we employ lightweight auxiliary heads following the design principles of prior works on multi-layer feature supervision (Szegedy et al., 2015; Lee et al., 2015) and early-exits (Xin et al., 2020; Liu et al., 2020). Each

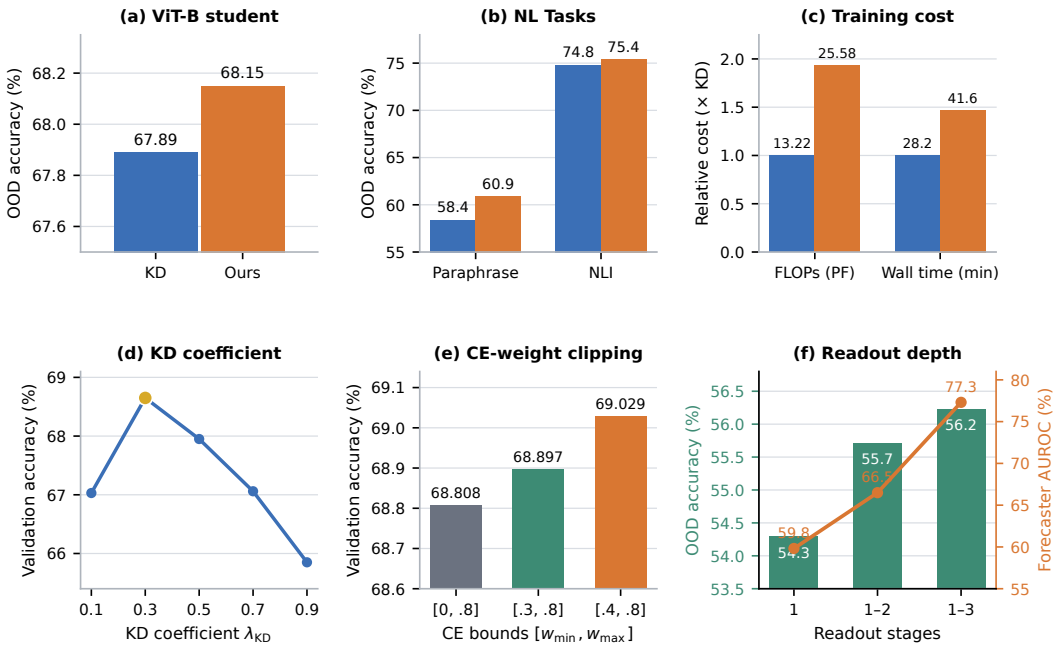


Figure 3: Compact evaluation and sensitivity results. (a) OOD accuracy for the ViT-B student. (b) KD and our method on the paraphrase and NLI tasks. (c) total supported-operation FLOPs and end-to-end wall time for 10k optimizer steps for student. (d) KD-coefficient grid search. (e) forecaster weight clipping. (f) Impact on OOD accuracy and forecaster AUROC as readout depth increases.

auxiliary head comprises two components: (i) a pooling operation to reduce the spatial dimensionality, and (ii) a feed-forward projection layer that maps the pooled features to a vector of dimension equal to the number of target classes. For transformer-based backbones, the auxiliary networks are simple linear projections i.e. one layer feed-forward block.

## 7 Results

Table 2 reports the domain generalization performance across three benchmark datasets. Here, ERM Vapnik (1998) corresponds to standard empirical risk minimization, where the network is trained solely on ground-truth supervision. In the KD (Hinton et al., 2015a) setting, the student is trained with an additional supervision from a larger network fine-tuned with ERM.

We observe that KD consistently outperforms both, the canonical empirical risk minimization (ERM), and hand-crafted gradient weighting mechanism in KD. These highlights the effectiveness of a tuned knowledge transfer from a robust teacher network. Building on the strong KD baseline, we evaluate our adaptive distillation setup, which augments KD with the *forecaster* meta-network that utilizes early readout signals from the student network. Our method improves over KD in majority of dataset-domain pairs under both, a convolutional `ResNet` student and transformer based `ViT` student evaluated on DomainNet (Refer Figure 3(a)). These consistent improvements across datasets showcase our approach as a principled extension to the standard KD problem for domain generalization.

Figure 3(b) further reports the domain generalization results in the text modality with a transformer-based student architecture. Here, our method yields an average increase of 1.6% in accuracy, showcasing robustness of the *forecaster*-based reweighting mechanism across both modality and student model architecture.

**The Role of Uncertainty Signals.** We further analyze the effect of intermediate layer prediction confidence signals on *forecaster* quality. As shown in Figure 2, augmenting auxiliary-classifier logits with entropy $\mathcal{H}(p)$, maximum confidence, and confidence margin $(\varepsilon)$ improves correctness AUROC from $63.7\%$ to $76.7\%$. This demonstrates that intermediate readouts provide rich signals of

sample ambiguity, which are then leveraged by the forecaster to effectively modulate the supervised-loss weight.

**Choice of Intermediate Layers.** Figure 3(f) shows that progressively incorporating intermediate readouts improves both OOD accuracy and forecaster AUROC, indicating that layers at different depths in the ResNet student provide complementary signals about sample difficulty. While later layers provide increasingly task-aligned evidence. Aggregating all intermediate readouts therefore gives the forecaster a richer view of the student's evolving prediction state.

**Hyperparameter Selection and Transfer.** We tune the student learning rate, distillation temperature $\tau$, KD coefficient $\lambda_{\text{KD}}$, and forecaster weight clipping bounds using source-domain validation accuracy on the DomainNet dataset (Refer Table 3). Within each student architecture, the selected configuration is fixed and transferred unchanged across all datasets. In our experiments, we observe that hyperparameters do not transfer uniformly across architectures. In particular, clipping forecaster weights stabilizes adaptive supervision, but overly restrictive bounds can suppress useful instance-level gradient (Refer Figure 3(e)).

**Stability in Training Schedule.** Further, we update the student model and the *forecaster* with equal update frequency, $T_s = T_f = 1$ in our experiments. We simply ensure that the *forecaster* is updated as frequently as the student, preventing the learned *forecaster* representations from going stale as the student representation evolves during training. To assess how the frequency of updates to the *forecaster* affect the downstream OOD performance of the student network, we conduct a sensitivity analysis by varying $T_s$ and $T_f$ across several configurations; ranges shown in Table 3. In these alternative schedules observe a small but consistent drop in average ID and OOD performance in these alternative configurations. These observations suggest that the *forecaster* network is negatively affected in both cases of (1) infrequent updates $(T_s = 10, T_f = 1)$ where the *forecaster* is lagging behind current student configuration and too frequent updates $(T_s = 10, T_f = 10$ or $T_s = 1, T_f = 10)$ where the *forecaster* potentially overfits to current student configuration. These results suggest that balanced updates allow for stable and effective training dynamics in the interleaved setting.

## 8 Conclusion

In this work, we introduced a *forecaster*-based reweighting approach to offline knowledge distillation, where the *forecaster* leverages intermediate layer readouts from the student model to adaptively modulate the distillation objective, improving the generalizability of the student network over vanilla KD. Our experiments across multiple benchmarks (vision and text) demonstrate efficacy over vanilla KD while adding no inference-time overhead. Together, these results establish a practical student-centric extension to standard KD for domain generalization.

## 9 Limitations and Future Work

First, our approach introduces additional training-time components (auxiliary heads and the forecaster) and relies on an interleaved optimization schedule. As shown in Figure 3(c), our interventions in KD result in approximately $2\times$ the PetaFLOPs and $1.5\times$ wall time as opposed to vanilla KD. However, these remain a one-time training cost, and do not introduce an inference-time overhead. Towards the experimental choices, we use ERM for the student-only baseline and the teacher algorithm. We believe that distilling knowledge from a teacher trained with a specialized DG algorithm similarly yields better student generalization than training the student directly with the same DG algorithm. Evaluating this hypothesis remains future work. Additionally, incorporating different modes of teacher supervision, including language-guided distillation (Huang et al., 2023) and domain-specialized online teachers (Zhao et al., 2025), into our uncertainty-aware framework and evaluating their complementary benefits remain open for future work.

## AI Use Statement

We used generative AI tools to assist with manuscript editing and grammatical consistency checks. All citations and reference metadata were independently verified by the authors. The research hypothesis, theoretical development, experimental design, results, analysis, and interpretation were

conducted by the authors. We reviewed all AI-assisted content and take full responsibility for the final manuscript and its claims.

## REFERENCES


Guillaume Alain and Yoshua Bengio. Understanding intermediate layers using linear classifier probes. *arXiv preprint arXiv:1610.01644*, 2016.

Görkem Algan and Ilkay Ulusoy. Meta soft label generation for noisy labels, 2021. URL https://arxiv.org/abs/2007.05836.

Robert Baldock, Hartmut Maennel, and Behnam Neyshabur. Deep learning through the lens of example difficulty. *Advances in Neural Information Processing Systems*, 34:10876–10889, 2021.

Jang Hyun Cho and Bharath Hariharan. On the efficacy of knowledge distillation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding, 2019. URL https://arxiv.org/abs/1810.04805.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale, 2021. URL https://arxiv.org/abs/2010.11929.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks, 2017. URL https://arxiv.org/abs/1703.03400.

Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples, 2015. URL https://arxiv.org/abs/1412.6572.

Ishaan Gulrajani and David Lopez-Paz. In search of lost domain generalization, 2020. URL https://arxiv.org/abs/2007.01434.

Yizeng Han, Gao Huang, Shiji Song, Le Yang, Honghui Wang, and Yulin Wang. Dynamic neural networks: A survey, 2021. URL https://arxiv.org/abs/2102.04906.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015. URL https://arxiv.org/abs/1512.03385.

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

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network, 2015b. URL https://arxiv.org/abs/1503.02531.

Rui Huang, Andrew Geng, and Yixuan Li. On the importance of gradients for detecting distributional shifts in the wild. *Advances in Neural Information Processing Systems*, 34:677–689, 2021.

Zeyi Huang, Andy Zhou, Zijian Lin, Mu Cai, Haohan Wang, and Yong Jae Lee. A sentence speaks a thousand images: Domain generalization through distilling clip with language guidance, 2023. URL https://arxiv.org/abs/2309.12530.

Fotis Iliopoulos, Vasilis Kontonis, Cenk Baykal, Gaurav Menghani, Khoa Trinh, and Erik Vee. Weighted distillation with unlabeled examples, 2022. URL https://arxiv.org/abs/2210.06711.

Nishant Jain, Karthikeyan Shanmugam, and Pradeep Shenoy. Selective classification using a robust meta-learning approach, 2024. URL https://arxiv.org/abs/2212.05987.

Pang Wei Koh, Shiori Sagawa, Henrik Marklund, Sang Michael Xie, Marvin Zhang, Akshay Balsubramani, Weihua Hu, Michihiro Yasunaga, Richard Lanas Phillips, Irena Gao, et al. Wilds: A benchmark of in-the-wild distribution shifts. In *International conference on machine learning*, pp. 5637–5664. PMLR, 2021.

Stefanos Laskaridis, Alexandros Kouris, and Nicholas D. Lane. Adaptive inference through early-exit networks: Design, challenges and directions. In *Proceedings of the 5th International Workshop on Embedded and Mobile Deep Learning*, MobiSys '21, pp. 1–6. ACM, June 2021. doi: 10.1145/3469116.3470012. URL http://dx.doi.org/10.1145/3469116.3470012.

Chen-Yu Lee, Saining Xie, Patrick Gallagher, Zhengyou Zhang, and Zhuowen Tu. Deeply-supervised nets. In *Artificial intelligence and statistics*, pp. 562–570. Pmlr, 2015.

Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy M. Hospedales. Deeper, broader and artier domain generalization, 2017. URL https://arxiv.org/abs/1710.03077.

Xin-Chun Li, Wen-Shu Fan, Shaoming Song, Yinchuan Li, Shao Yunfeng, De-Chuan Zhan, et al. Asymmetric temperature scaling makes larger networks teach well again. *Advances in neural information processing systems*, 35:3830–3842, 2022.

Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection, 2018. URL https://arxiv.org/abs/1708.02002.

Weijie Liu, Peng Zhou, Zhiruo Wang, Zhe Zhao, Haotang Deng, and Qi Ju. FastBERT: a self-distilling BERT with adaptive inference time. In Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel Tetreault (eds.), *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 6035–6044, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.537. URL https://aclanthology.org/2020.acl-main.537/.

Raphael Gontijo Lopes, Stefano Fenu, and Thad Starner. Data-free knowledge distillation for deep neural networks, 2017. URL https://arxiv.org/abs/1710.07535.

Jiaming Lv, Haoyuan Yang, and Peihua Li. Wasserstein distance rivals kullback-leibler divergence for knowledge distillation. *Advances in Neural Information Processing Systems*, 37:65445–65475, 2024.

Yoshitomo Matsubara, Marco Levorato, and Francesco Restuccia. Split computing and early exiting for deep learning applications: Survey and research challenges. *ACM Computing Surveys*, 55 (5):1–30, December 2022. ISSN 1557-7341. doi: 10.1145/3527155. URL http://dx.doi.org/10.1145/3527155.

Omar Montasser, Han Shao, and Emmanuel Abbe. Transformation-invariant learning and theoretical guarantees for ood generalization. *Advances in Neural Information Processing Systems*, 37: 108649–108673, 2024.

Anshul Nasery, Sravanti Addepalli, Praneeth Netrapalli, and Prateek Jain. Daft: Distilling adversarially fine-tuned models for better ood generalization, 2022. URL https://arxiv.org/abs/2208.09139.

Aniruddh Raghu, Maithra Raghu, Simon Kornblith, David Duvenaud, and Geoffrey Hinton. Teaching with commentaries, 2021. URL https://arxiv.org/abs/2011.03037.

Mengye Ren, Wenyuan Zeng, Bin Yang, and Raquel Urtasun. Learning to reweight examples for robust deep learning, 2019. URL https://arxiv.org/abs/1803.09050.

Alexander Robey, George J Pappas, and Hamed Hassani. Model-based domain generalization. *Advances in Neural Information Processing Systems*, 34:20210–20229, 2021.

Kuniaki Saito, Donghyun Kim, Stan Sclaroff, Trevor Darrell, and Kate Saenko. Semi-supervised domain adaptation via minimax entropy. *CoRR*, abs/1904.06487, 2019. URL http://arxiv.org/abs/1904.06487.

Shreyas Saxena, Oncel Tuzel, and Dennis DeCoste. Data parameters: A new family of parameters for learning a differentiable curriculum. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper_files/paper/2019/file/926ffc0ca56636b9e73c565cf994ea5a-Paper.pdf.

Roy Schwartz, Gabriel Stanovsky, Swabha Swayamdipta, Jesse Dodge, and Noah A. Smith. The right tool for the job: Matching model and instance complexities. In Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel Tetreault (eds.), *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 6640–6651, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.593. URL https://aclanthology.org/2020.acl-main.593/.

Jun Shu, Qi Xie, Lixuan Yi, Qian Zhao, Sanping Zhou, Zongben Xu, and Deyu Meng. Meta-weight-net: Learning an explicit mapping for sample weighting, 2019. URL https://arxiv.org/abs/1902.07379.

Durga Sivasubramanian, Ayush Maheshwari, Pradeep Shenoy, Prathosh AP, and Ganesh Ramakrishnan. Adaptive mixing of auxiliary losses in supervised learning, 2022. URL https://arxiv.org/abs/2202.03250.

Rupesh K Srivastava, Klaus Greff, and Jürgen Schmidhuber. Training very deep networks. *Advances in neural information processing systems*, 28, 2015.

Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1–9, 2015.

Surat Teerapittayanon, Bradley McDanel, and H. T. Kung. Branchynet: Fast inference via early exiting from deep neural networks, 2017. URL https://arxiv.org/abs/1709.01686.

Rishabh Tiwari, Durga Sivasubramanian, Anmol Mekala, Ganesh Ramakrishnan, and Pradeep Shenoy. Using early readouts to mediate featural bias in distillation, 2023. URL https://arxiv.org/abs/2310.18590.

Rishabh Tiwari, Durga Sivasubramanian, Anmol Mekala, Ganesh Ramakrishnan, and Pradeep Shenoy. Using early readouts to mediate featural bias in distillation. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 2638–2647, 2024.

Iulia Turc, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Well-read students learn better: On the importance of pre-training compact models, 2019. URL https://arxiv.org/abs/1908.08962.

V.N. Vapnik. *Statistical Learning Theory*. Adaptive and learning systems for signal processing, communications, and control. Wiley, 1998. ISBN 9788126528929. URL https://books.google.co.in/books?id=RWrlkQEACAAJ.

Hemanth Venkateswara, Jose Eusebio, Shayok Chakraborty, and Sethuraman Panchanathan. Deep hashing network for unsupervised domain adaptation, 2017. URL https://arxiv.org/abs/1706.07522.

Yoav Wald, Amir Feder, Daniel Greenfeld, and Uri Shalit. On calibration and out-of-domain generalization. *Advances in neural information processing systems*, 34:2215–2227, 2021.

Yisen Wang, Weiyang Liu, Xingjun Ma, James Bailey, Hongyuan Zha, Le Song, and Shu-Tao Xia. Iterative learning with open-set noisy labels, 2018. URL https://arxiv.org/abs/1804.00092.

Yufei Wang, Haoliang Li, Lap pui Chau, and Alex C. Kot. Embracing the dark knowledge: Domain generalization using regularized knowledge distillation, 2021. URL https://arxiv.org/abs/2107.02629.

Ji Xin, Raphael Tang, Jaejun Lee, Yaoliang Yu, and Jimmy Lin. DeeBERT: Dynamic early exiting for accelerating BERT inference. In Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel Tetreault (eds.), *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 2246–2251, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.204. URL https://aclanthology.org/2020.acl-main.204/.

Canwen Xu and Julian McAuley. A survey on dynamic neural networks for natural language processing, 2023. URL https://arxiv.org/abs/2202.07101.

Linyi Yang, Shuibai Zhang, Libo Qin, Yafu Li, Yidong Wang, Hanmeng Liu, Jindong Wang, Xing Xie, and Yue Zhang. Glue-x: Evaluating natural language understanding models from an out-of-distribution generalization perspective, 2023. URL https://arxiv.org/abs/2211.08073.

Xinli Yue, Mou Ningping, Qian Wang, and Lingchen Zhao. Revisiting adversarial robustness distillation from the perspective of robust fairness. *Advances in Neural Information Processing Systems*, 36:30390–30401, 2023.

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.

Di Zhao, Jingfeng Zhang, Hongsheng Hu, Philippe Fournier-Viger, Gillian Dobbie, and Yun Sing Koh. Balancing invariant and specific knowledge for domain generalization with online knowledge distillation. In James Kwok (ed.), *Proceedings of the Thirty-Fourth International Joint Conference on Artificial Intelligence, IJCAI-25*, pp. 2440–2448. International Joint Conferences on Artificial Intelligence Organization, 8 2025. doi: 10.24963/ijcai.2025/272. URL https://doi.org/10.24963/ijcai.2025/272. Main Track.

Haoran Zhao, Xin Sun, Junyu Dong, Zihe Dong, and Qiong Li. Knowledge distillation via instance-level sequence learning, 2021. URL https://arxiv.org/abs/2106.10885.

Shanshan Zhao, Mingming Gong, Tongliang Liu, Huan Fu, and Dacheng Tao. Domain generalization via entropy regularization. *Advances in neural information processing systems*, 33:16096–16107, 2020.

Andy Zhou, Jindong Wang, Yu-Xiong Wang, and Haohan Wang. Distilling out-of-distribution robustness from vision-language foundation models. *Advances in Neural Information Processing Systems*, 36:32938–32957, 2023.

Kaiyang Zhou, Yuanhan Zhang, Yuhang Zang, Jingkang Yang, Chen Change Loy, and Ziwei Liu. On-device domain generalization. *arXiv preprint arXiv:2209.07521*, 2022.

Wangchunshu Zhou, Canwen Xu, Tao Ge, Julian McAuley, Ke Xu, and Furu Wei. Bert loses patience: Fast and robust inference with early exit, 2020. URL https://arxiv.org/abs/2006.04152.

# A Appendix

## A.1 Hyperparameter Search

We tune optimization and loss-balancing hyperparameters using only source-domain validation accuracy. For each student architecture, we select one configuration by averaging validation accuracy over its leave-one-domain-out folds on the DomainNet dataset, then reuse that configuration unchanged across all held-out domains and random seeds of the same architecture across datasets. Held-out OOD data are evaluated only after selection. Table 3 lists the search spaces.

Table 3: Hyperparameter search space.

| **Component** | **Hyperparameter** | **Search range / fixed value** |
|---|---|---|
| Student | Learning rate | $\{1, 2, 4\} \times 10^{-4}$ |
| Distillation | Temperature $\tau$ | $\{2, 4\}$ |
| Adaptive KD | Persistent coefficient $\lambda_{\mathrm{KD}}$ | $\{0.1, 0.3, 0.5, 0.7, 0.9\}$ |
| Vanilla KD | Coefficient $\lambda_{\mathrm{KD}}$ | $\{0.1, 0.3, 0.5, 0.7, 0.9\}$ |
| CE clipping | Lower bound $w_{\min}$ | $\{0, 0.2, 0.3, 0.4\}$ |
| CE clipping | Upper bound $w_{\max}$ | $\{0.6, 0.8, 0.9, 1.0\}$ |
| Readout | Learning rate | $10^{-3}$ (fixed) |
| Forecaster | Learning rate | $10^{-3}$ (fixed) |
| Interleaving | $(T_s, T_f)$ | $(1, 1), (1, 10), (10, 1), (10, 10)$ |

### A.2 Prompt Details for Text Classification

We use the following fixed instruction templates for the paraphrase-identification and natural-language-inference experiments.

```
Paraphrase Identification: Training Prompt (MRPC)

Sentence 1: {{sentence1}}
Sentence 2: {{sentence2}}
Question: Do both sentences mean the same thing?
Answer:

Paraphrase Identification: Test Prompt (QQP)

Question 1: {{question1}}
Question 2: {{question2}}
Question: Do both questions ask the same thing?
Answer:

Natural Language Inference: Train/Test Prompt (MNLI)

{{premise}}
Question: {{hypothesis}} True, False or Neither?
Answer:
```