EEL: CALIBRATING EARLY-EXIT NEURAL NETWORKS VIA HIERARCHICAL SELF-SUPERVISED LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Early-exit neural networks reduce inference latency by enabling dynamic termination at intermediate exit branches. However, standard supervised training typically prioritizes the final-layer objective, leaving shallow exits with underdeveloped representations and poor calibration. To address this, we propose a Siamese hierarchical Self-Supervised Learning (SSL) paradigm for producing early-exit neural networks that treats intermediate exits as independent representation endpoints. By applying SSL regularization symmetrically across network depth, our framework promotes useful representations at all exits independently of task-specific classification heads. The resulting multi-branch encoder is then fine-tuned with early-exit classification heads to obtain the final early-exit network. Evaluation across convolutional and Vision Transformer backbones testing both contrastive and non-contrastive objectives demonstrates that the proposed approach improves the calibration of shallow exits, while maintaining a comparable efficiency trade-off.

1 INTRODUCTION

Deploying deep neural networks in latency-sensitive applications, ranging from real-time speech processing to edge computer vision, requires balancing computational cost and predictive performance (Rahmath P et al., 2024). Early-exit architectures, as BranchyNet Teerapittayanon et al. (2016), address this trade-off by allowing inputs to terminate at intermediate branches when a confidence criterion, such as entropy or maximum class probability, indicates that the prediction is sufficiently reliable. However, shallow exits often exhibit reduced predictive accuracy and poorly calibrated confidence estimates (Laskaridis et al., 2021). These exits may exhibit insufficient discriminative power and miscalibrated confidence estimates towards making the exiting decision Meronen et al. (2024). We argue that these limitations are not necessarily inherent to the multi-exit architecture itself, but instead arise in part from the conventional supervised optimization paradigm, which does not explicitly require intermediate representations to be independently useful for inference.

Currently, training strategies of inference-adaptive multi-exit DNNs remain mostly supervised. Because existing training strategies optimize intermediate predictors through task-specific supervised objectives at different exits, this approach shapes each depth's representation around its immediate classification task rather than explicitly building a useful endpoint. For example, criteria such as Cross-Entropy can bias shallow representations toward task-specific class discrimination. This tension may further amplify representational differences across depth Jiang et al. (2024), leaving shallow branches with weaker separability and lower feature rank for standalone inference (Lin et al., 2024). While contrastive self-supervised learning has been explored for multi-exit DNNs (Bae et al., 2023), to the best of our knowledge, prior work has not explicitly formulated intermediate exits as independent representation-learning endpoints for standalone inference.

In this work, we address this bottleneck of standard supervised training by introducing a hierarchical Siamese SSL framework. In comparison to previous hierarchical approaches focusing only on backbone stability (Jang et al., 2023a) we explicitly optimize a multi-branch encoder where each individual branch is a distinct representation endpoint. By applying regularization symmetrically across all exits, we encourage the representations at each exit depth to remain useful independently of task labels. The framework is architecture-agnostic and does not require knowledge distillation between exits. Instead, representational consistency is induced through the SSL objective and the symmetric Siamese structure. We hypothesize that explicitly learning useful representations at

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.

intermediate depths can improve the reliability of their corresponding predictions, which we evaluate through calibration while observing accuracy-efficiency trade-offs. The contributions of our work can be summarized as follows:

- We introduce a framework that leverages a hierarchical Siamese SSL setup – compatible with both contrastive and non-contrastive regularizers that prevent representation collapse – to train multi-branch encoders for early exiting. We additionally explore the potential benefits of exploiting this hierarchy through cross-depth alignment acting as an additional ablation layer.
- We demonstrate that the proposed framework is backbone-agnostic and regularization-agnostic, indicating that the benefits of our method are an outcome of the optimization objective leading to better Expected Calibration Error (ECE) of shallow branches.
- We demonstrate that our method's encoders, when fine-tuned with early-exit classification heads instead of trained from scratch, converge in substantially fewer epochs and achieve an accuracy-latency per-sample trade-off comparable to, or better than, the supervised early-exiting baseline.

2 Related Work

2.1 Multi-exit DNNs and Supervisory Tension

The need for efficient inference led to multi-exit DNNs, allowing for samples to terminate computation at intermediate branches when a confidence threshold is met. Initially, frameworks like BranchyNet (Teerapittayanon et al., 2016), MSDNet (Huang et al., 2017), and SDN (Kaya et al., 2019) established the feasibility of this approach for Convolutional Neural Networks (CNNs). This paradigm was extended to Transformer architectures via frameworks such as DeeBERT(Xin et al., 2020), FastBERT (Liu et al., 2020), PABEE (Zhou et al., 2020), and BEEM (Bajpai & Hanawal, 2025), which introduced additional exiting criteria like patience mechanisms (Zhou et al., 2020) or ensembles of intermediate confidence scores (Bajpai & Hanawal, 2025), thus balancing latency and predictive capacity during inference, along with concepts such as Knowledge Distillation (KD) during training (Liu et al., 2020).

Yet the optimization objectives for the majority of training regimes remain primarily based on supervised label learning across every exit point. The prevailing assumption in the literature is that intermediate representations can be effectively shaped by the same task-specific objectives applied to the final layer. This is often intensified through KD, where Intermediate Predictors (IPs) learn to mimic the behavior of another exit through self-distillation (Liu et al., 2020) or teacher-student KD.

However, jointly optimizing multiple supervised exits introduces a multi-objective optimization problem in which gradients associated with different exits may compete for shared parameters (Mokssit et al., 2025). Such gradient interference can be particularly relevant in early layers, which must simultaneously support predictions at multiple depths. This competition forces intermediate representations to be a byproduct of conflicting supervisory signals rather than optimized standalone endpoints. MSDNet (Huang et al., 2017) addresses this problem through architectural modifications; however, it still requires labels for supervised optimization. Unlike approaches that primarily modify exit criteria, branch placement, or supervised optimization, we instead investigate whether intermediate representations can be learned independently of task-specific supervision through hierarchical self-supervision.

2.2 Hierarchical Self-Supervised Learning

Self-supervised learning approaches can be broadly divided into contrastive and non-contrastive methods based on the underlying optimization objective. Contrastive methods such as SimCLR (Chen et al., 2020) and MoCo (He et al., 2020) maximize agreement between augmented views while contrasting positive pairs against negative samples, thereby avoiding degenerate solutions. Non-contrastive methods avoid collapse through distinct techniques. SimSiam (Chen & He, 2021) and BYOL (Grill et al., 2020) leverage asymmetric architectures with stop-gradient or momentum encoders. Barlow Twins (Zbontar et al., 2021), VICReg (Bardes et al., 2021), along with SIGReg (Balestriero & LeCun, 2025) encourage reduced redundancy through feature de-correlation or distri-

bution regularization. SwAV (Caron et al., 2020) assigns representations to learned prototypes via online clustering.

In many standard SSL formulations, the training objective is applied to a projection head attached to the final encoder representation (Chen et al., 2020; Bardes et al., 2021; Balestriero & LeCun, 2025). Prior analyses suggest that projection heads can absorb aspects of the geometric constraints imposed by SSL objectives while allowing the encoder representation to retain more task-relevant semantic structure (Ma et al., 2023). The projector is discarded after the pretraining phase is over, and no mechanism for shaping intermediate encoder representations is provided.

However, the use of individual projectors across the hierarchy of layers is not as explored. Some hierarchical extensions of SSL, such as Matryoshka Representation Learning (Kusupati et al., 2022), focus on encoding multi-granular information across embeddings and is therefore not directly applicable to multi-branch early-exit architectures. Other extensions focus on the hierarchy across the network depth. Prior work on hierarchical SSL across layers, such as SDSSL (Jang et al., 2023b) and SelfCon (Bae et al., 2023), utilize intermediate layers as training regularizers or supervised contrastive auxiliaries, respectively. Datta et al. (Datta et al., 2025) instead train networks layer-wise with local VICReg updates in place of backpropagation, producing progressively more compact per-layer representations not focusing on early-exit inference. (Yoshihashi et al., 2023) proposes the paradigm of Ladder Siamese Networks where intermediate SSL across the hierarchy of layers is used to produce better final layer representations using a non-contrastive approach and not independent representations across branches. Also, S-JEA (Manová et al., 2023) pursues the learning of hierarchical semantics through stacking encoders sequentially. To the best of our knowledge, no prior work pretrains multi-exit encoders with label-free objectives and evaluates them under confidence-threshold early exiting, including the calibration of shallow exits (We provide additional discussion on hierarchical SSL and early-exit neural networks in Appendix B.).

2.3 Overconfidence and Multi-Exit Neural Networks Calibration

A DNN whose confidence scores are representative of the actual likelihood of correct prediction is considered a well-calibrated model (Guo et al., 2017a). It is demonstrated that DNNs trained using cross-entropy on one-hot targets tend to produce overconfident predictions (Wang et al., 2021; Liu et al., 2022). In the context of multi-exit DNNs, miscalibration is further enhanced by shallow branches producing wrong predictions with high confidence. Difficult samples are predicted incorrectly with high confidence and are not deferred to subsequent deeper branches (Meronen et al., 2024; Bajpai & Hanawal, 2026), potentially degrading the trade-off between predictive performance and latency. Supervised cross-entropy training can therefore expose intermediate branches to the same overconfidence tendencies observed in final classifiers, while their representations may be less developed due to their shallower depth. Therefore, the resulting early exits are under-trained and miscalibrated. SSL objectives such as SimCLR and Barlow Twins (Chen et al., 2020; Zbontar et al., 2021) instead, learn representations through view invariance and redundancy reduction rather than one-hot class supervision, motivating their investigation as an alternative means of shaping intermediate representations and potentially improving exit calibration.

3 Preliminaries

In this section, we formalize the multi-branch encoder we propose and describe the optimization conflict described in the previous section that motivates our hierarchical self-supervised approach. In the context of this work an early-exit DNN is treated as a combination of a multi-branch encoder that has predictors (classification heads) attached at each branch.

3.1 Architecture Components Overview

- **Backbone (f_θ):** This component is a multi-branch encoder parameterized by a set of weights symbolized as θ and is treated as a series of N ordered blocks. For a given input sample x, the backbone produces a set of intermediate features $\mathcal{H} = \{h_1, h_2, ..., h_N\}$ across branches, where $h_i = f_\theta^{(i)}(x)$ symbolizes the output of the i-th block.
- **Branch-specific Projectors (g_i):** Each i-th branch of the backbone has a non-linear projector g_i attached, mapping features h_i into an embedding space $z_i = g_i(h_i) \in \mathbb{R}^{d_i}$, where $\mathbb{R}^{d_i}$

denotes the d_i-dimensional Euclidean space. The projectors are leveraged during pretraining to optimize towards the self-supervised objectives and are then discarded.

- **Linear Probes (c_i):** The discriminative maturity of representations across the backbone blocks during pretraining is evaluated by assigning lightweight linear classifiers c_i to each one of them that are discarded afterwards.
- **Alignment Projectors (g_{a_i}):** The challenge of facilitating alignment across depth is addressed by assigning one projector g_{a_i} per branch, that maps features h_i to a shared latent space $p_i = g_{a_i}(h_i) \in \mathbb{R}^{d_a}$. These projections are used for defining the alignment loss $\mathcal{L}_{align}$ and are discarded after pretraining. Figure 4 captures the broader architecture of the proposed training paradigm.

Each one of the components involved, complementary to the backbone, are parameterized by individual sets of weights separate from θ. Their respective weights are updated simultaneously based on their contribution to the total optimization loss along with the backbone. The use of alignment projectors aiming at cross-depth alignment is not inherently against our initial purpose of optimizing branches independently, as it enables for exploring the exploitation of features learned across the hierarchy of layers (Appendix C.1).

3.2 Multi-Exit DNNs and Conventional Supervised Optimization

In most existing frameworks, multi-exit DNNs are optimized using a joint supervised objective. Given a dataset $D = \{(x_j, y_j)\}_{j=1}^{K}$, where x_j denotes the feature representation of the j-th sample and y_j denotes its corresponding label, the standard approach minimizes the sum of Cross-Entropy losses (CE) across the network's depth as shown in Eq. 1 where w_i represents a task-specific weight for the i-th branch and k_i symbolizes a typical classification head of multi-exit neural networks in this context.

$$\mathcal{L}_{Joint} = \sum_{i=1}^{N} w_i \text{CE}(k_i(f_\theta^{(i)}(x)), y) \tag{1}$$

To be more specific, the backbone parameters of multi-exit DNNs θ are shared and the total gradient signal is a summation of conflicting vectors without an existing measure of encouraging standalone representation sufficiency. The failure of this paradigm can be considered as a Supervisory Tension problem. Shallow layers (e.g., h_1,h_2) produce features that discard non-discriminative information to satisfy the local optimization objective of the classification head. This can bias shallow representations toward task-specific discriminative directions and limit the diversity of the learned features.

3.3 Problem Formulation

Our framework aims to optimize parameters θ such that representations learned at each h_i remain informative when used independently. The broader optimization objective can be expressed as follows:

$$\theta^* = \arg\min_{\theta} [\frac{1}{N} \sum_{i=1}^{N} \mathcal{L}_{ssl}^{i} + \lambda \mathcal{L}_{align}] \tag{2}$$

where $\mathcal{L}_{ssl}$ is the SSL optimization objective at each layer i, and $\mathcal{L}_{align}$ is the alignment objective, regularized by coefficient λ. Although the projector and probe parameters are optimized jointly with θ during pretraining, we express the objective in terms of θ alone, as these auxiliary components are discarded after pretraining and only the backbone is retained for downstream use.

In addition, the alignment objective is not a plain knowledge distillation application but an attempt to align the representations of shallow branches with all of the other branches at deeper layers. We include this term primarily to explore whether cross-depth alignment offers benefits on top of independent per-branch SSL, rather than as a core requirement of the framework.

4 METHODOLOGY

In this work, multi-exit pretraining is formulated as hierarchical representation learning. Rather than optimizing intermediate branches through supervised objectives, each branch is explicitly treated as an independent representation-learning endpoint. To be more specific, the encoder is trained using branch-wise self-supervised objectives, while supervised labels are used only for detached linear probes monitoring the process of pretraining in terms of quality.

The proposed framework consists of three components: (i) a symmetric branch-wise SSL objective that learns invariant representations while preventing degenerate representations, (ii) isolated linear-probe evaluation providing a measure of intermediate representation quality based on labels without propagating the supervised gradients back to the multi-branch encoder, and finally (iii) a cross-depth representation-alignment objective encouraging shallow representations to be aligned with the representations of deeper branches. These are further described by Algorithm 1 and Fig. 4.

4.1 SYMMETRIC PER-BRANCH SSL OBJECTIVES

A symmetric Siamese configuration is leveraged to encourage view-invariance independently at each branch of the backbone. As shown in lines 3–5 of Algorithm 1, for each batch of samples $(x, y) \sim D$, two noisy transformed versions of the same input are generated. Specifically, two transformations $t_1, t_2 \sim T$, where T denotes the set of noise transformations, are independently sampled and applied to x, producing two related views $x_1 = t_1(x)$ and $x_2 = t_2(x)$. Although both views originate from the same underlying sample, they differ due to the applied noise transformations. The two transformed inputs are then processed by the same shared encoder f_θ, resulting in two corresponding sets of feature representations, $\{h_1^i\}_{i=1}^N$ and $\{h_2^i\}_{i=1}^N$, for the two branches, as shown in line 7. Sharing parameters θ across both views, the backbone is encouraged to learn a unified representation that is informative for both of them across individual backbone endpoints.

Each branch i is assigned a projector g_i mapping the intermediate features into an embeddings space, producing $z_1^i = g_i(h_1^i)$ and $z_2^i = g_i(h_2^i)$ (line 8). The SSL objective $\mathcal{L}_{ssl}^i$ per-branch i consists of a predictive loss $\mathcal{L}_{inv}^i$ and a regularization term $\mathcal{L}_{reg}^i$ (lines 9-11):

$$\mathcal{L}_{ssl}^i = \mathcal{L}_{reg}^i + \mathcal{L}_{inv}^i \tag{3}$$

where $\mathcal{L}_{inv}^i = \|z_1^i - z_2^i\|_2^2$ enforces invariance across views. We explicitly decouple the invariance term from the regularization objective responsible for collapse prevention. Therefore, the framework is regularization agnostic and different regularizers can be used for $\mathcal{L}_{reg}^i$. For the non-contrastive formulation, we instantiate $\mathcal{L}_{reg}$ using VICReg or SIGReg. We additionally evaluate the framework with a contrastive objective (NT-Xent per-branch loss) to assess whether the benefits extend beyond non-contrastive regularization. In this work mainly two regularizers are explored. VICReg (Bardes et al., 2021) prevents collapse $\mathcal{L}_{VICReg}$ through imposing variance $\mathcal{L}_{var}$ and covariance $\mathcal{L}_{cov}$ constraints across batches, being balanced by two positive coefficients as shown in Eq. 4. Within our decomposition, the VICReg regularization component comprises the variance and covariance terms, while the invariance component is represented separately by $\mathcal{L}_{inv}$.

$$\mathcal{L}_{VICReg} = a\mathcal{L}_{var} + b\mathcal{L}_{cov} \tag{4}$$

SIGReg regularizes the embedding distribution toward an isotropic Gaussian using random one-dimensional projections and the Epps-Pulley test (Balestriero & LeCun, 2025). In the context of this training objective, loss is computed on embedding features and their alignment with random 1D projections $d_m \sim \mathcal{N}(0, I)$ leveraging the Epps-Pulley (Balestriero & LeCun, 2025) test $\mathcal{G}$. The aforementioned can be expressed by Eq. 5 where Z corresponds to a batch of embeddings.

$$\mathcal{L}_{SIGReg} = \frac{1}{M} \sum_{m=1}^{M} \mathcal{G}(Zd_m) \tag{5}$$

These objectives are applied independently across branches as described by the inner loop of Algorithm 1 in lines 7-13, aiming for each branch to develop mature representations. The base loss is expressed as:

$$\mathcal{L}_{base} = \frac{1}{N} \sum_{i=1}^{N} \mathcal{L}_{ssl}^{i} \tag{6}$$

4.2 Isolated Supervised Monitoring

During pretraining, each branch i is monitored by a lightweight linear probe c_i trained on the detached features of its respective branch as described from Algorithm 1 (lines 12–13):

$$\mathcal{L}_{sup}^{i} = \text{CE}(c_i(stopgrad(h_1^i)), y) \tag{7}$$

By leveraging stopgrad() we ensure supervised signals do not propagate into the parameters θ. Probe accuracy at each branch i serves as a real-time proxy for representation quality.

4.3 Cross-depth Alignment

Local objectives $\mathcal{L}_{ssl}^{i}$ do not explicitly encourage coherence across the hierarchy. In CNNs particularly, feature rank across layers varies with depth, and shallow branches may converge to semantically sub-optimal solutions without soft alignment. Each branch i is therefore equipped with an alignment projector g_{a_i} producing normalized projections $\hat{p}_i = p_i/\|p_i\|$ where $p_i = g_{a_i}(h_i)$, aligned via (Algorithm 1 line 16):

$$\mathcal{L}_{align} = \frac{1}{|\mathcal{P}|} \sum_{i<j} \left(1 - \cos\left(\hat{p}_i, \text{stopgrad}(\hat{p}_j)\right)\right) \tag{8}$$

Stopgrad on $\hat{p}_j$ is deliberate: treating the deeper branch as a fixed target ensures only $\hat{p}_i$ is updated. The quality of deeper features is preserved while shaping earlier representations. The objective is normalized by the total number of pairs of projections that are present denoted as $\mathcal{P}$.

4.4 Total Objective and Training Protocol

The complete objective combines $\mathcal{L}_{base}$ and $\mathcal{L}_{align}$ via a balancing factor λ as described in Algorithm 1 (lines 18):

$$\mathcal{L}_{total} = \mathcal{L}_{base} + \lambda \mathcal{L}_{align} \tag{9}$$

Training proceeds in two phases. In the first phase, f_θ is pretrained using $\mathcal{L}_{total}$ per Algorithm 1, with probes $\{c_i\}$ serving as monitoring means and discarded afterwards along with all the projectors. The complementary components (e.g., projectors and probes) are optimized in relation to the loss objective they contribute to through the LARS optimizer used for the encoder. In the second phase, task-specific classification heads are attached to each branch, and the full model is fine-tuned under a standard supervised objective for a small number of epochs, with θ remaining trainable throughout. This two-phase design enables competitive multi-exit performance with substantially less downstream training than supervised baselines using one encoder as basis. The sensitivity of λ is analyzed in the ablation study in Appendix D.9.

5 Experiments

5.1 Setup

Data: Our experiments focus on image classification tasks as this is one of the most popular domains of application in early-exit neural networks. The datasets used vary in terms of number of samples,

dimensions, and domains: CIFAR10, CIFAR100 (Krizhevsky et al., 2009), CIFAR10.1v6 (Recht et al., 2018), STL-10 (Coates et al., 2011), Galaxy10 DeCALS (Leung & Bovy, 2019) and ImageNet100 (Tian et al., 2020). Extensive information regarding the data used during the experiments as well as the augmentation pipelines can be found in the Appendix C.5.

Encoders: We use ResNet18, ResNet34, ViT-S, and ViT-B as our backbone architectures, as detailed in Appendix C.4. In the first phase, each backbone is instantiated as a multi-branch encoder, with each branch stripped of any task-specific prediction head. In the second phase, we reuse the same multi-branch architecture, but now append a single fully connected linear layer to each branch to act as a classification head. Full architectural details of our multi-branch encoders are provided in Appendix C.4, and our implementation hyperparameters are largely based on (Terver et al., 2026), upon which we build our proposed training paradigm. The scope of the pretrained encoders used is presented in Appendix C.6

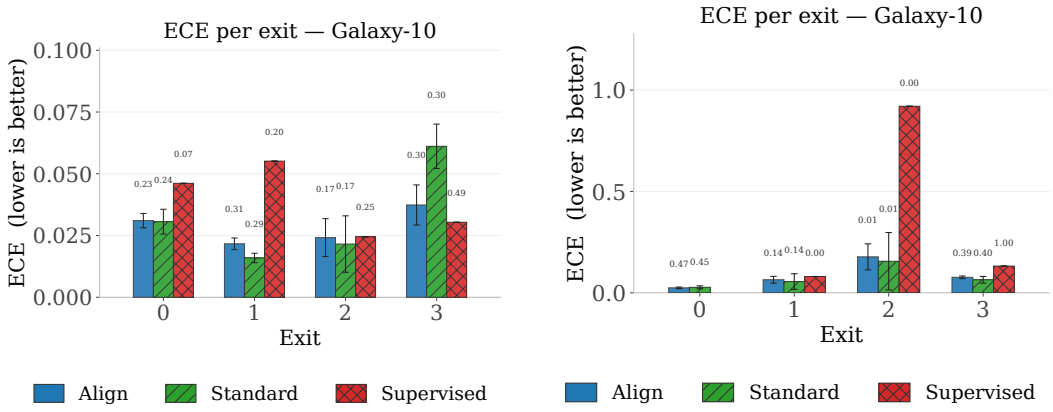
1st Phase - Self-Supervised Training: The data augmentation details of creating semantically aligned images can be found in Appendix C.5. The SSL procedure lasts 300 epochs across all the encoders where a combination of linear learning rate schedule followed by cosine annealing schedule is adopted while using the LARS optimizer. Across all exits the same type of regularizer is applied individually. Extensive information about the specifics of the pretraining can be found in Appendix C.3

2nd Phase - Supervised Training & Fine-Tuning: We consider three categories of models. **Supervised** models are trained from scratch using mainstream supervised learning. **Standard** and **Align** models are fine-tuned from our multi-branch encoders, with one linear classification head attached per branch. They differ in whether cross-depth alignment was used during the SSL procedure, which **Align** includes and **Standard** does not. Supervised models are trained for the same number of epochs as the self-supervised encoders (300). For **Standard** and **Align** models, all parameters remain trainable during fine-tuning, and none of our models requires more than 50 epochs to converge. Model confidence is quantified as the maximum value of the softmax output vector, and the experiments are conducted using three different seeds. More details about the choice of baselines and training overhead can be found in Appendix C.8 and AppendixC.7

5.2 ECE & Efficiency Tradeoff

Expected Calibration Error (ECE) (Guo et al., 2017b) is a metric used for measuring whether a model's predicted confidence matches its accuracy. It is leveraged to ensure that the confidence score of each exit is trustworthy and that it is separately computed for each exit branch. Overconfident exit points can lead to premature wrong exits, and underconfident exits can lead to unnecessary compute where samples do not exit at all, as confidence never exceeds the threshold. This is especially prominent for shallow exit branches that are usually trained on less discriminative intermediate features and are forced to produce a softmax distribution. Often such branches are wrong despite their high confidence. Unless otherwise stated in our experiments, we choose the threshold of the max probability exit criterion to be equal to 0.9 in our diagrams. As can be observed in Fig. 1, our method results in better ECE values for the shallow branches of both ViT-S and ResNet34 in comparison to the supervised baseline. In ResNet34 (Fig. 1a) the baseline shallow branches are overconfident compared to the proposed methodology. In contrast, the ECE value close to zero observed for the first branch of ViT-S indicates that the shallow exits are underconfident (Fig. 1b), which consequently limits the number of samples exiting early regarding the supervised baseline.

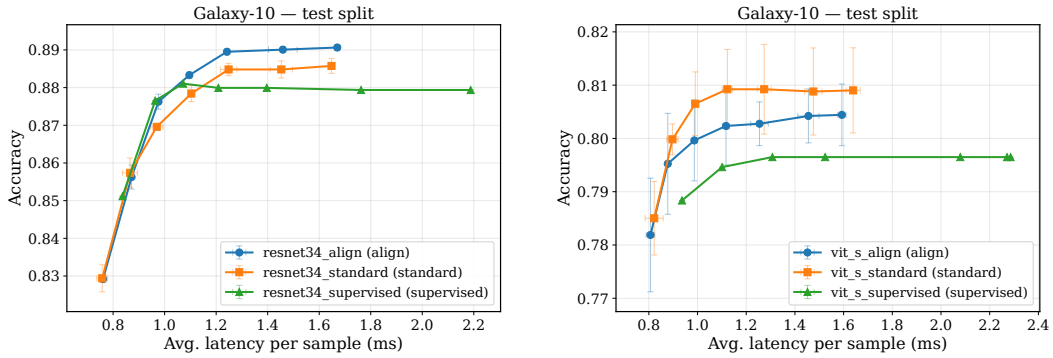
For Galaxy10 (Appendix D.4), the accuracy-latency Pareto frontier shows that our proposed method achieves a comparable or even superior efficiency trade-off across both architectures (Fig. 2). More results relevant to ECE and the corresponding Pareto frontiers for CIFAR10 and ImageNet100 can be found in Appendix D.1 and Appendix D.5 respectively, along with the fraction of exits per branch. Our experiments indicate that the application of cross-depth alignment affects the efficiency trade-off differently in comparison to our standard approach, as there are obvious deviations between the ECE of shallow branches and the corresponding Pareto frontier; however there is no discernible consistent pattern to report across architectures or datasets. In the scope of this work and in alignment with recent research we recognize that ECE is not always associated with the efficiency trade-off, and therefore these two aspects are observed individually (Kubaty et al., 2025).



(a) ResNet34 & VICReg - ECE

(b) ViT-S & VICReg - ECE

Figure 1: ECE across the shallow exit branches indicates that for both **Align** and **Standard** there is better calibration in comparison to **Supervised**. Above each bar the fraction of samples exiting is denoted floored to two decimal digits. The choice of a high confidence threshold demonstrates the behavior of shallow branches under demanding conditions.



(a) ResNet34 & VICReg - Pareto Frontier

(b) ViT-S & VICReg - Pareto Frontier

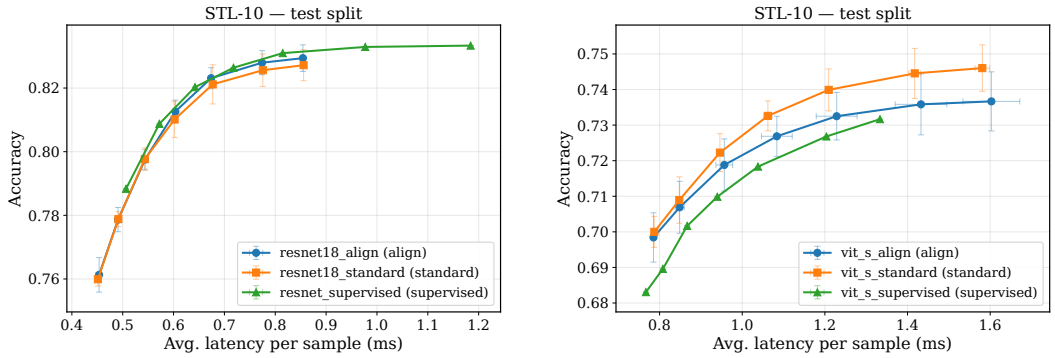
Figure 2: The Pareto frontiers suggest that the early-exit models trained with the proposed regime can achieve comparable accuracy with similar or smaller latency; this trend is more evident for bigger thresholds. We evaluate thresholds between 0.40 and 0.90 with step of 0.10, across all presented Pareto frontier diagrams.

5.3 Distribution Shift

Introducing distribution shift by testing our fine-tuned pretrained encoders along with the supervised baseline models can help in understanding if the representations learned during pretraining are of good quality, through the preservation of a healthy balance between accuracy and latency. Leveraging our fine-tuned early-exit encoders, based on ResNet18 and ViT-S, along with the supervised baselines trained on CIFAR10, we test our models directly on STL-10 as seen in Fig. 3. It is interesting to observe that for ResNet18 there are mainly latency gains and comparable accuracy, whereas this trend is reversed for ViT-S, where accuracy is improved however the corresponding latency is comparable. Additional results about introducing distribution shift using CIFAR10.1 and STL-10 can be found in Appendix D.2 and Appendix D.3.

5.4 Representation Quality & Alignment

Centered Kernel Alignment (CKA) (Kornblith et al., 2019) in the scope of this work is used to measure the similarity of representations across the layers of an encoder. Higher values of this



(a) ResNet18 & SIGReg - Pareto Frontier

(b) ViT-S & VICReg - Pareto Frontier

Figure 3: The proposed method across both **Align** and **Standard** provides comparable or improved accuracy-latency, reaching areas of the frontier that the **Supervised** does not. For example, ResNet18 is able to run with less latency and reduced accuracy, whereas ViT-S is able to reach higher accuracy across thresholds at slightly increased latency.

metric are interpreted as two layers of the encoder producing similar representations. As seen in Appendix D.8, Supervised branches of ResNet34 trained on CIFAR10 show lower inter-exit CKA at shallow depths (E1–E2: 0.78 vs. 0.84) but higher CKA at deeper pairs (E2–E3: 0.91 vs. 0.86; E2–E4: 0.92 vs. 0.83; E3–E4: 1.00 vs. 0.97). The near-unity E3–E4 CKA under supervision suggests representational redundancy, consistent with Supervisory Tension collapsing intermediate representations toward the final layer. **Standard** branches stay more distinct at depth, while their higher shallow-depth CKA suggests more coherent early representations than the locally conflicting gradients of supervised training produce.

5.4.1 SIGReg vs VICReg

In order to assess whether SIGReg or VICReg result in encoders with significant differences in performance, we study their application on the same type of encoder based on ResNet34 trained on CIFAR100. Our results indicate that the efficiency trade-off remains comparable, and their in-between differences for specific thresholds are not even discernible when observing the Pareto frontier found in Appendix D.6

5.4.2 NT-Xent Objective

It is important to verify that it is viable to substitute our combined invariance-regularization objective with a contrastive objective such as NT-Xent to demonstrate that our approach is agnostic to contrastive and non-contrastive regularization. Observing the Pareto frontier of ViT-S on Galaxy10 leveraging NT-Xent instead, provides comparable results when a batch size of 512 is leveraged during the 1st Phase of training (Appendix D.7). However, as expected, the paradigm of NT-Xent collapses when a batch size of 32 is leveraged due to an insufficient number of samples per batch (Appendix D.7). Therefore, the choice of non-contrastive regularizers should be preferred on low-resources training setups.

6 Conclusion

We propose a new method for training multi-branch encoders positioned for early-exit neural networks based on established mechanisms such as Siamese setups and well-known regularizers. The benefits of our method are improved shallow predictor calibration accompanied with efficiency trade-offs that are comparable to the typical supervised paradigm, providing different combinations of accuracy and latency across certain thresholds not accessible by **Supervised**. It is also demonstrated that the use of cross-depth alignment across all of our experiments is not necessarily providing significantly different results in a consistent manner. However, there are limitations too such as the effect of branch architecture and positioning to the proposed approach.

7 AI USE STATEMENT

In this work, we used generative AI tools to provide feedback on the clarity and readability of the manuscript as well as for the implementation of certain plotting functions. We have not used generative AI to help in matters such as developing theoretical models or conceptual frameworks, formulating mathematical claims, interpretation of results and data analysis. We have reviewed all AI-assisted work. For example, we assessed the feedback received and applied sparingly some of the provided suggestions and ensured that the plot functions render properly the diagrams of our data. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

8 REPRODUCIBILITY STATEMENT

In order to make our research readily available to readers it is important to promote transparency in terms of reproducibility. In Appendix C, we provide a diagram of the proposed pretraining architecture, an algorithm describing our method of pretraining, the pretraining hyperparameters, the architecture of our multi-branch encoder and auxiliary components as well as the data augmentation pipeline used during pretraining. In addition, we mention the hardware leveraged along with the total training duration across encoders.

REFERENCES

Sangmin Bae, Sungnyun Kim, Jongwoo Ko, Gihun Lee, Seungjong Noh, and Se-Young Yun. Self-contrastive learning: single-viewed supervised contrastive framework using sub-network. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pp. 197–205, 2023.

Divya Jyoti Bajpai and Manjesh Kumar Hanawal. BEEM: Boosting performance of early exit DNNs using multi-exit classifiers as experts. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=EzrZX9bd4G.

Divya Jyoti Bajpai and Manjesh Kumar Hanawal. Beyond greedy exits: Improved early exit decisions for risk control and reliability. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2026. URL https://openreview.net/forum?id=PNU2mPauvd.

Randall Balestriero and Yann LeCun. Lejepa: Provable and scalable self-supervised learning without the heuristics. *arXiv preprint arXiv:2511.08544*, 2025.

Adrien Bardes, Jean Ponce, and Yann LeCun. Vicreg: Variance-invariance-covariance regularization for self-supervised learning. *arXiv preprint arXiv:2105.04906*, 2021.

Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. *Advances in neural information processing systems*, 33:9912–9924, 2020.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pp. 1597–1607. PmLR, 2020.

Xinlei Chen and Kaiming He. Exploring simple siamese representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 15750–15758, 2021.

Adam Coates, Andrew Ng, and Honglak Lee. An analysis of single-layer networks in unsupervised feature learning. In Geoffrey Gordon, David Dunson, and Miroslav Dudík (eds.), *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, volume 15 of *Proceedings of Machine Learning Research*, pp. 215–223, Fort Lauderdale, FL, USA, 11–13 Apr 2011. PMLR. URL https://proceedings.mlr.press/v15/coates11a.html.

Joy Datta, Rawhatur Rabbi, Puja Saha, Aniqua Nusrat Zereen, M. Abdullah-Al-Wadud, and Jia Uddin. Deep representation learning using layer-wise VICReg losses. *Scientific Reports*, 15 (1):27049, July 2025. ISSN 2045-2322. doi: 10.1038/s41598-025-08504-2. URL https://doi.org/10.1038/s41598-025-08504-2.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 248–255. IEEE, 2009.

Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Guo, Mohammad Gheshlaghi Azar, et al. Bootstrap your own latent-a new approach to self-supervised learning. *Advances in neural information processing systems*, 33:21271–21284, 2020.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. On calibration of modern neural networks. ICML'17, pp. 1321–1330. JMLR.org, 2017a.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. On calibration of modern neural networks. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, pp. 1321–1330, Sydney, NSW, Australia, December 2017b. JMLR.org. URL https://dl.acm.org/doi/10.5555/3305381.3305518.

Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 9729–9738, 2020.

Gao Huang, Danlu Chen, Tianhong Li, Felix Wu, Laurens Van Der Maaten, and Kilian Q Weinberger. Multi-scale dense networks for resource efficient image classification. *arXiv preprint arXiv:1703.09844*, 2017.

Jiho Jang, Seonhoon Kim, Kiyoon Yoo, Chaerin Kong, Jangho Kim, and Nojun Kwak. Self-distilled self-supervised representation learning. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pp. 2829–2839, 2023a.

Jiho Jang, Seonhoon Kim, Kiyoon Yoo, Chaerin Kong, Jangho Kim, and Nojun Kwak. Self-distilled self-supervised representation learning. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pp. 2829–2839, 2023b.

Jiachen Jiang, Jinxin Zhou, and Zhihui Zhu. Tracing representation progression: Analyzing and enhancing layer-wise similarity. *arXiv preprint arXiv:2406.14479*, 2024.

Yigitcan Kaya, Sanghyun Hong, and Tudor Dumitras. Shallow-Deep Networks: Understanding and Mitigating Network Overthinking. In *Proceedings of the 36th International Conference on Machine Learning*, pp. 3301–3310. PMLR, May 2019. URL https://proceedings.mlr.press/v97/kaya19a.html.

Simon Kornblith, Mohammad Norouzi, Honglak Lee, and Geoffrey Hinton. Similarity of neural network representations revisited. In *International conference on machine learning*, pp. 3519–3529. PMlR, 2019.

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

Piotr Kubaty, Filip Szatkowski, Eric Nalisnick, Bartosz WĂłjcik, et al. Rethinking calibration for early-exit neural networks. *arXiv preprint arXiv:2508.21495*, 2025.

Aditya Kusupati, Gantavya Bhatt, Aniket Rege, Matthew Wallingford, Aditya Sinha, Vivek Ramanujan, William Howard-Snyder, Kaifeng Chen, Sham Kakade, Prateek Jain, et al. Matryoshka representation learning. *Advances in Neural Information Processing Systems*, 35:30233–30249, 2022.

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*, pp. 1–6, June 2021. doi: 10.1145/3469116.3470012. URL http://arxiv.org/abs/2106.05022. arXiv:2106.05022 [cs].

Henry W. Leung and Jo Bovy. Galaxy10 decals dataset. https://github.com/henrysky/Galaxy10, 2019. astroNN documentation: https://astronn.readthedocs.io/en/latest/galaxy10.html.

Shaohui Lin, Bo Ji, Rongrong Ji, and Angela Yao. A closer look at branch classifiers of multi-exit architectures. *Computer Vision and Image Understanding*, 239:103900, 2024. ISSN 1077-3142. doi: https://doi.org/10.1016/j.cviu.2023.103900. URL https://www.sciencedirect.com/science/article/pii/S1077314223002801.

Bingyuan Liu, Ismail Ben Ayed, Adrian Galdran, and Jose Dolz. The devil is in the margin: Margin-based label smoothing for network calibration. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 80–88, 2022.

Weijie Liu, Peng Zhou, Zhiruo Wang, Zhe Zhao, Haotang Deng, and Qi Ju. Fastbert: a self-distilling bert with adaptive inference time. In *Proceedings of the 58th annual meeting of the association for computational linguistics*, pp. 6035–6044, 2020.

Jiajun Ma, Tianyang Hu, and Wenjia Wang. Deciphering the projection head: Representation evaluation self-supervised learning. *arXiv preprint arXiv:2301.12189*, 2023.

Alžběta Manová, Aiden Durrant, and Georgios Leontidis. S-jea: Stacked joint embedding architectures for self-supervised visual representation learning. *arXiv preprint arXiv:2305.11701*, 2023.

Lassi Meronen, Martin Trapp, Andrea Pilzer, Le Yang, and Arno Solin. Fixing overconfidence in dynamic neural networks. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pp. 2680–2690, 2024.

Saad Mokssit, Ouassim Karrakchou, Alejandro Mousist, and Mounir Ghogho. Confidence-gated training for efficient early-exit neural networks. *arXiv preprint arXiv:2509.17885*, 2025.

Haseena Rahmath P, Vishal Srivastava, Kuldeep Chaurasia, Roberto G. Pacheco, and Rodrigo S. Couto. Early-Exit Deep Neural Network - A Comprehensive Survey. *ACM Comput. Surv.*, 57(3): 75:1–75:37, August 2024. ISSN 0360-0300. doi: 10.1145/3698767. URL https://dl.acm.org/doi/10.1145/3698767.

Benjamin Recht, Rebecca Roelofs, Ludwig Schmidt, and Vaishaal Shankar. Do cifar-10 classifiers generalize to cifar-10? *arXiv preprint arXiv:1806.00451*, 2018.

Surat Teerapittayanon, Bradley McDanel, and H.T. Kung. Branchynet: Fast inference via early exiting from deep neural networks. In *2016 23rd International Conference on Pattern Recognition (ICPR)*, pp. 2464–2469, 2016. doi: 10.1109/ICPR.2016.7900006.

Basile Terver, Randall Balestriero, Megi Dervishi, David Fan, Quentin Garrido, Tushar Nagarajan, Koustuv Sinha, Wancong Zhang, Mike Rabbat, Yann LeCun, et al. A lightweight library for energy-based joint-embedding predictive architectures. *arXiv preprint arXiv:2602.03604*, 2026.

Yonglong Tian, Dilip Krishnan, and Phillip Isola. Contrastive multiview coding. In *European Conference on Computer Vision (ECCV)*, 2020.

Deng-Bao Wang, Lei Feng, and Min-Ling Zhang. Rethinking calibration of deep neural networks: Do not be afraid of overconfidence. *Advances in Neural Information Processing Systems*, 34: 11809–11820, 2021.

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/.

Ryota Yoshihashi, Shuhei Nishimura, Dai Yonebayashi, Yuya Otsuka, Tomohiro Tanaka, and Takashi Miyazaki. Ladder siamese network: A method and insights for multi-level self-supervised learning. In *2023 IEEE International Conference on Image Processing (ICIP)*, pp. 211–215, 2023. doi: 10.1109/ICIP49359.2023.10222599.

Jure Zbontar, Li Jing, Ishan Misra, Yann LeCun, and Stéphane Deny. Barlow twins: Self-supervised learning via redundancy reduction. In *International conference on machine learning*, pp. 12310–12320. PMLR, 2021.

Wangchunshu Zhou, Canwen Xu, Tao Ge, Julian McAuley, Ke Xu, and Furu Wei. Bert loses patience: Fast and robust inference with early exit. *Advances in Neural Information Processing Systems*, 33: 18330–18341, 2020.

A INTRODUCTION

The appendix can be used complementary to the main content for reproducibility purposes and for purposes relevant to the close examination of model behavior across experiments along with additional ablation nuances. In addition, there is a short presentation of adjacent work in combination with our positioning to it.

B ADJACENT LITERATURE

Method	Label-free	Own SSL loss per depth	Depth early-exit eval.
Matryoshka RL (Kusupati et al., 2022)	○	×	×
SelfCon (Bae et al., 2023)	×	×	×
SDSSL (Jang et al., 2023b)	●	○	×
Ladder Siamese Net (Yoshihashi et al., 2023)	●	●	×
S-JEA (Manová et al., 2023)	●	●	×
Layer-wise VICReg (Datta et al., 2025)	●	●	×
EEL (Ours)	●	●	●

Table 1: Positioning of EEL against related hierarchical self-supervised learning work. ● = yes, ○ = partial, × = no. *Label-free*: pretraining uses no class labels. *Own SSL loss per depth*: each intermediate depth is trained with its own SSL objective (partial: the target is the final-layer output). *Depth early-exit eval.*: confidence-thresholded exiting at intermediate depths is evaluated.

Table 1 positions our framework against closely related work on hierarchical SSL and early-exit neural networks, to the best of our understanding.

Matryoshka RL: Matryoshka Representation Learning can be applied in both supervised and non-supervised setups. This approach operates across nested representation dimensions of the final output embedding vector. In the scope of this method, rather than exiting early from intermediate depth layers, a final output representation vector is produced and nested prefix sub-vectors across lower dimensions are evaluated.

SelfCon: The standard SelfCon approach described in this work leverages labels and rather than evaluating the completely independent, isolated SSL losses across depth, the objective contrasts representations from different depths of the same network. Specifically, the algorithm contrasts the representations across depth with those of the final layer. The use of sub-networks at each depth is inspired by multi-exit neural networks however it is not delving into the creation of early-exit inference-adaptive neural networks.

SDSSL: This is a label-free method however the training involves the SSL objective optimization of intermediate layers relevant to the final layer. Therefore, the intermediate branches are not primarily treated independently but they are largely dependent on the final layer. This work demonstrates that intermediate layers can produce representations of good quality however the efficiency trade-off of early-exit neural networks using threshold mechanisms to provide premature predictions is not studied.

Ladder Siamese Networks, S-JEA, Layer-wise VICReg: Ladder Siamese Networks (Yoshihashi et al., 2023), S-JEA (Manová et al., 2023) and Layer-wise VICReg (Datta et al., 2025) do apply a label-free SSL objective at intermediate depths, but they investigate the quality of the resulting representations rather than their use for early-exit inference. We demonstrate that using different backbones and types of regularizers can lead to multi-branch encoders positioned for early-exit inference unlike the current supervised paradigm.

C REPRODUCIBILITY

C.1 ARCHITECTURE DIAGRAM

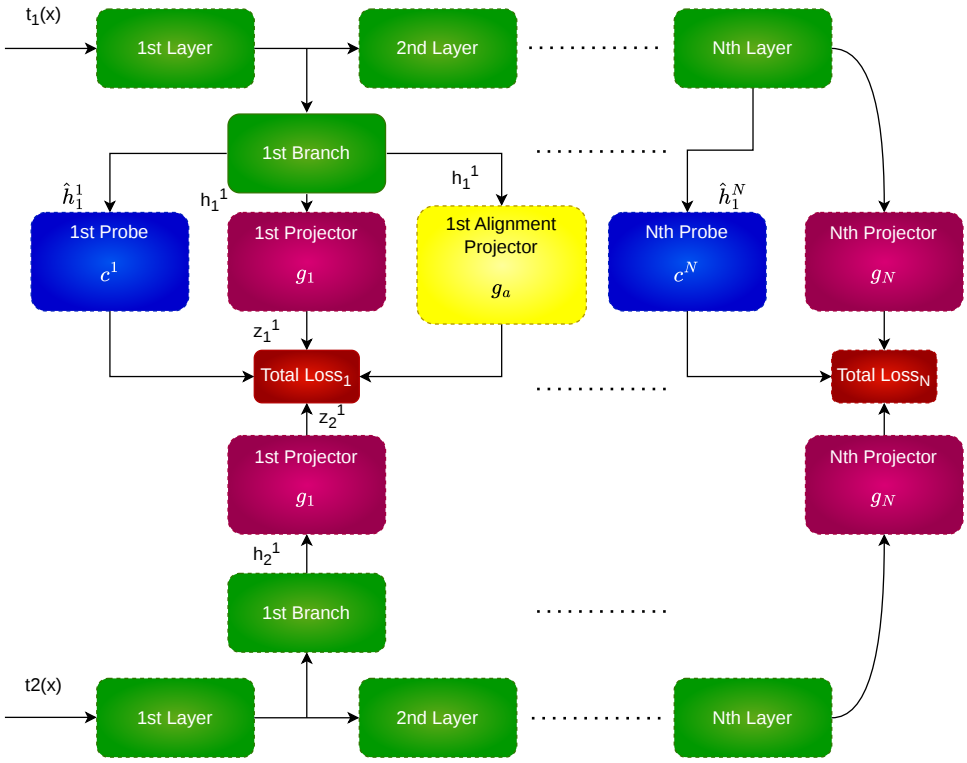


Figure 4: Siamese Hierarchical Self-Supervised Learning Setup

C.2 Algorithm Overview

Algorithm 1 Hierarchical Non-Contrastive SSL for Multi-Exit DNNs

Input: Training set D, transformation set T, backbone f_θ, SSL projectors $\{g_i\}$, alignment projectors $\{g_{a_i}\}$, linear probes $\{c_i\}$, balancing factor λ.
while not converged **do**
  Sample batch $(x, y) \sim D$
  Sample transformations $t_1, t_2 \sim T$
  Generate views: $x_1 = t_1(x), \quad x_2 = t_2(x)$
  for each branch $i = 1$ to N **do**
    $h_1^i, h_2^i \leftarrow f_\theta^{(i)}(x_1), f_\theta^{(i)}(x_2)$
    $z_1^i, z_2^i \leftarrow g_i(h_1^i), g_i(h_2^i)$
    $\mathcal{L}_{inv}^i \leftarrow \|z_1^i - z_2^i\|_2^2$
    $\mathcal{L}_{reg}^i \leftarrow$ VICReg or SIGReg(z_1^i, z_2^i)
    $\mathcal{L}_{ssl}^i = \mathcal{L}_{inv}^i + \mathcal{L}_{reg}^i$
    $\hat{h}^i \leftarrow$ stopgrad(h_1^i)
    $\mathcal{L}_{sup}^i =$ CE$(c_i(\hat{h}^i), y)$
    $p_i \leftarrow g_{a_i}(h_1^i), \quad \hat{p}_i \leftarrow p_i / \|p_i\|$
  end for
  $\mathcal{L}_{align} = \dfrac{1}{|\mathcal{P}|} \sum_{i<j} (1 - \cos(\hat{p}_i, \text{stopgrad}(\hat{p}_j)))$
  $\mathcal{L}_{base} = \dfrac{1}{N} \sum_{i=1}^N \mathcal{L}_{ssl}^i$
  $\mathcal{L}_{total} = \mathcal{L}_{base} + \lambda \mathcal{L}_{align} + \dfrac{1}{N} \sum_{i=1}^N \mathcal{L}_{sup}^i$
  Update $\theta, \{g_i\}, \{g_{a_i}\}, \{c_i\}$ jointly via LARS w.r.t. $\mathcal{L}_{total}$ {stopgrad blocks $\mathcal{L}_{sup}^i$, $\mathcal{L}_{align}$'s $\hat{p}_j$ term from updating θ}
end while

C.3 SSL Training Hyperparameters

Table 2: Pretraining hyperparameters across model families.

Hyperparameter	ResNet18	ResNet34	ViT-Small	ViT-Base
Number of exits	4	4	4	4
Exit feature dims	[64, 128, 256, 512]	[64, 128, 256, 512]	[384, 384, 384, 384]	[768, 768, 768, 768]
SSL objective (per branch)				
Regularizer	SIGReg	SIGReg	VICReg	VICReg
Per-branch params	num_slices = [64, 128, 256, 512]; λ = 10.0 (all branches)	num_slices = [64, 128, 256, 512]; λ = 10.0 (all branches)	std_coeff = 1.0, cov_coeff = 25.0; std_margin = 1.0 (all branches)	std_coeff = 1.0, cov_coeff = 25.0; std_margin = 1.0 (all branches)
Exit loss weights	[1.0, 1.0, 1.0, 1.0]	[1.0, 1.0, 1.0, 1.0]	[1.0, 1.0, 1.0, 1.0]	[1.0, 1.0, 1.0, 1.0]
Optimization				
Optimizer	LARS	LARS	LARS	LARS
LR – backbone + projectors	0.3	0.3	0.3	0.3
LR – linear probes	0.2	0.2	0.2	0.2
Weight decay	1e-4	1e-4	1e-4	1e-4
Momentum	0.9	0.9	0.9	0.9
LARS η	0.02	0.02	0.02	0.02
clip_lr	True	True	True	True
exclude_bias_n_norm	True	True	True	True
Scheduler				
Type	Warmup-cosine	Warmup-cosine	Warmup-cosine	Warmup-cosine
Warmup epochs	10	10	10	10
Total epochs	300	300	300	300
base_lr / warmup_start_lr / min_lr	0.3 / 3e-5 / 0.0	0.3 / 3e-5 / 0.0	0.3 / 3e-5 / 0.0	0.3 / 3e-5 / 0.0
Training				
Batch size	256	256	512	512
Number of views	2	2	2	2
AMP dtype	float16	float16	float16	float16
Alignment				
Alignment enabled	off / on	off / on	off / on	off / on
Alignment weight	0 / 0.2	0 / 0.2	0 / 0.2	0 / 0.2
Alignment projector LR	n/a / 0.2	n/a / 0.2	n/a / 0.2	n/a / 0.2

C.4 Component Architecture

ResNet34

Table 3: Backbone architecture for ResNet34.

Stage	Configuration
Backbone	`torchvision.models.resnet34`
MaxPool	Identity
layer1	64 channels
layer2	$64 \to 128$ channels (stride 2 in first block)
layer3	$128 \to 256$ channels (stride 2 in first block)
layer4	$256 \to 512$ channels (stride 2 in first block)
Final pool	AdaptiveAvgPool2d(1×1) $\to$ Flatten
FC head	Identity

ResNet18

Table 4: Backbone architecture for ResNet18.

Stage	Configuration
Backbone	`torchvision.models.resnet18`
MaxPool	Identity
layer1	64 channels
layer2	$64 \to 128$ channels (stride 2 in first block)
layer3	$128 \to 256$ channels (stride 2 in first block)
layer4	$256 \to 512$ channels (stride 2 in first block)
Final pool	AdaptiveAvgPool2d(1×1) $\to$ Flatten
FC head	Identity

Convolutional Neural Network Branches

Table 5: CNN branches (shared by ResNet18 and ResNet34).

Branch	Configuration
Branch 1 (after layer1, $d = 64$)	[Conv 3×3 $\to$ BN $\to$ ReLU] $\times$ 3 $\to$ AvgPool $\to$ Flatten
Branch 2 (after layer2, $d = 128$)	[Conv 3×3 $\to$ BN $\to$ ReLU] $\times$ 2 $\to$ AvgPool $\to$ Flatten
Branch 3 (after layer3, $d = 256$)	[Conv 3×3 $\to$ BN $\to$ ReLU] $\times$ 1 $\to$ AvgPool $\to$ Flatten

ViT-S

Table 6: Backbone architecture for ViT-Small.

Component	Configuration
Backbone	`torchvision.models.VisionTransformer`
Patch size	2×2 or 16×16 (Galaxy10 & ImageNet100)
Hidden dim	384
Number of layers	12
Number of heads	6
MLP dim	1536 ($= 4 \times 384$)
Heads	Identity

ViT-B

Table 7: Backbone architecture for ViT-Base.

Component	Configuration
Backbone	`torchvision.models.VisionTransformer`
Patch size	16×16
Hidden dim	768
Number of layers	12
Number of heads	12
MLP dim	3072 ($= 4 \times 768$)
Heads	Identity

Vision Transformer Branches

Table 8: Vision Transformer branches for ViT-Small.

Branch	Configuration
Branch 1 (after encoder block 2, $d = 384$)	LayerNorm$(384) \to$ Linear$(384,\ 384) \to$ GELU
Branch 2 (after encoder block 5, $d = 384$)	LayerNorm$(384) \to$ Linear$(384,\ 384) \to$ GELU
Branch 3 (after encoder block 8, $d = 384$)	LayerNorm$(384) \to$ Linear$(384,\ 384) \to$ GELU

Table 9: Vision Transformer branches for ViT-Base.

Branch	Configuration
Branch 1 (after encoder block 2, $d = 768$)	LayerNorm$(768) \to$ Linear$(768,\ 768) \to$ GELU
Branch 2 (after encoder block 5, $d = 768$)	LayerNorm$(768) \to$ Linear$(768,\ 768) \to$ GELU
Branch 3 (after encoder block 8, $d = 768$)	LayerNorm$(768) \to$ Linear$(768,\ 768) \to$ GELU

Linear Probes

Table 10: Linear probes. One linear probe per exit. For ResNet18/34, $d \in \{64, 128, 256, 512\}$ across the four exits.

Model	Configuration
ResNet18 / ResNet34	Linear$(d,$ `num_classes`$)$
ViT-Small	Linear$(384,$ `num_classes`$)$
ViT-Base	Linear$(768,$ `num_classes`$)$

Alignment Projectors

Table 11: Alignment projectors. One projector per exit. For ResNet18/34, $d \in \{64, 128, 256, 512\}$ across the four exits.

Model	Configuration
ResNet18 / ResNet34	Linear$(d,\ 256) \rightarrow$ BN $\rightarrow$ ReLU $\rightarrow$ Linear$(256,\ 256)$
ViT-Small	Linear$(384,\ 256) \rightarrow$ BN $\rightarrow$ ReLU $\rightarrow$ Linear$(256,\ 256)$
ViT-Base	Linear$(768,\ 256) \rightarrow$ BN $\rightarrow$ ReLU $\rightarrow$ Linear$(256,\ 256)$

SSL Projectors

Table 12: SSL projectors. One projector per exit. For ResNet18/34, $d \in \{64, 128, 256, 512\}$ across the four exits.

Model	Configuration
ResNet18 / ResNet34	Linear$(d,\ 2048) \rightarrow$ BN $\rightarrow$ ReLU $\rightarrow$ Linear$(2048,\ 2048) \rightarrow$ BN $\rightarrow$ ReLU $\rightarrow$ Linear$(2048,\ 128)$
ViT-Small	Linear$(384,\ 2048) \rightarrow$ BN $\rightarrow$ ReLU $\rightarrow$ Linear$(2048,\ 2048) \rightarrow$ BN $\rightarrow$ ReLU $\rightarrow$ Linear$(2048,\ 2048)$
ViT-Base	Linear$(768,\ 2048) \rightarrow$ BN $\rightarrow$ ReLU $\rightarrow$ Linear$(2048,\ 2048) \rightarrow$ BN $\rightarrow$ ReLU $\rightarrow$ Linear$(2048,\ 2048)$

C.5 Data Augmentation & Datasets

Table 13: Datasets and data augmentation for SSL pretraining. Augmentations are listed in application order.

	SSL pretraining
Views per image	2
Augmentations (in order)	RandomResizedCrop(image_size, scale=(0.2, 1.0)) ColorJitter(0.4, 0.4, 0.2, 0.1), p=0.8 RandomGrayscale, p=0.2 Solarization(thr. 128), p=0.1 RandomHorizontalFlip, p=0.5 ToTensor Normalize

Fine-tuning: During fine-tuning we ensure that augmentation is identical between both our encoders and the supervised baselines.

CIFAR10: This dataset consists of 60,000 32×32 color images from 10 classes, with 50,000 training images and 10,000 test images.

CIFAR10.1: This is an independently collected test set for CIFAR-10, designed to evaluate model generalization beyond the original CIFAR-10 test distribution. We use version v6, which contains 2,000 images.

CIFAR100: This dataset consists of 60,000 32×32 color images from 100 classes, with 50,000 training images and 10,000 test images.

STL-10: This dataset contains 10 classes of color images at a resolution of 96×96 (resized to 32×32), with 5,000 labeled training images, 8,000 test images, and 100,000 unlabeled images.

Galaxy10 DeCALS: This dataset contains 17,736 labeled galaxy images (256×256) belonging to 10 morphological classes, derived from the Galaxy Zoo project.

ImageNet100: This is a 100-class subset of ImageNet-1K (Deng et al., 2009), following the subsampling protocol of (Tian et al., 2020) (160×160).

C.6 Experimental Scope

The full scope of our experiments is mentioned below, aiming to show the effectiveness of the proposed method across various combinations of regularizers and architectures, running 3 seeds per case:

CIFAR10: ResNet18 (SIGReg), ResNet34 (SIGReg), ViT-S (SIGReg)

CIFAR100: ResNet34 (SIGReg), ResNet34(VICReg)

Galaxy10: ResNet34 (VICReg), ViT-S (VICReg), ViT-B (VICReg), ViT-S (NT-Xent)

ImageNet100: ViT-B (VICReg)

C.7 Hardware Specifications - Training Overhead

All the experiments were supported using one NVIDIA RTX Pro 6000 Blackwell. The aforementioned GPU was used during pretraining, and the figures shown in our manuscript mentioning latency are all based on results measured on this type of hardware too.

Our SSL pretraining of 300 epochs on average across datasets lasts approximately 3 hours for ResNet18, 6 hours for ResNet34, 5 hours for ViT-S and 6.5 hours for ViT-B.

The corresponding cost in hours of GPU usage for the fine-tuning procedure on average across datasets tested is approximately 10 minutes for ResNet18, 40 minutes for ResNet34, 13 minutes for ViT-S and 40 minutes for ViT-B.

Our supervised baselines were trained for 300 epochs and the average cost in hours of GPU usage across datasets tested per architecture is 1.5 hours for ResNet18, 5 hours for ResNet34, 2 hours for ViT-S and 5 hours for ViT-B.

The durations mentioned above are averaged across the datasets each architecture was evaluated on, per the experimental scope. CIFAR10 and CIFAR100 have comparable training costs due to similar image count and resolution. Galaxy10, however, has substantially larger/higher-resolution images; since convolutional cost scales with spatial resolution throughout the network, this significantly raises ResNet34's average duration relative to ResNet18 (trained on CIFAR10 only). ViT-B, in contrast, was evaluated on Galaxy10 and ImageNet100 (vs. ViT-S on CIFAR10 and Galaxy10): despite ImageNet100's larger scale, ViT-B's use of a 16×16 patch size keeps its token count low, so its average duration remains comparable to ResNet34's rather than disproportionately higher. Also, some results of reusing the proposed encoders are presented in Appendix E.

C.8 Baselines

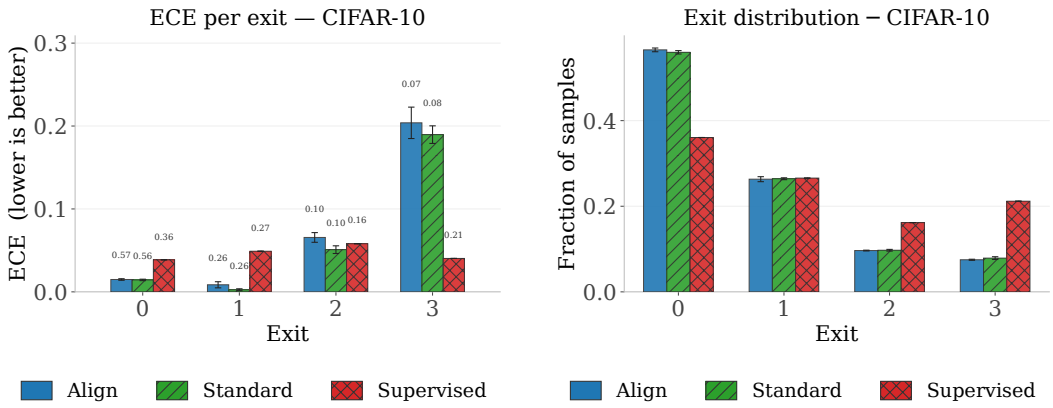
The reasoning behind choosing a typical joint optimization supervised training paradigm for our baselines is because of three factors. First, it is one of the most popular approaches in training early-exit neural networks across the related literature. Second, it highlights the problem of supervisory tension that resulted in so many researchers focusing on calibration in the domain of early-exit neural networks. Third, the use of KD in combination with typical supervised approaches as well as post-hoc calibration are orthogonal to our paradigm as it addresses the pretraining procedure leading to multi-branch encoders. These encoders during fine-tuning can be treated as any other early-exit neural network and integrate such orthogonal techniques.

D ADDITIONAL RESULTS

D.1 CIFAR10

D.1.1 RESNET18 & SIGREG

ECE & Exit Distribution



(a) ResNet18 - ECE

(b) ResNet18 - Exit Distribution

Figure 5: ECE across the shallow exit branches indicates that for both **Align** and **Standard** there is better calibration in comparison to **Supervised**. The exit distribution indicates that in comparison to the baseline more samples exit early.

Pareto Frontier

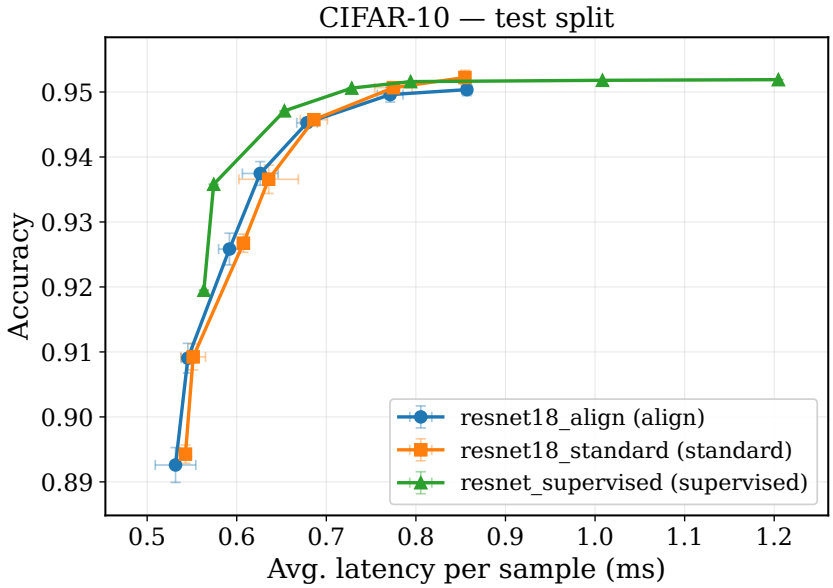
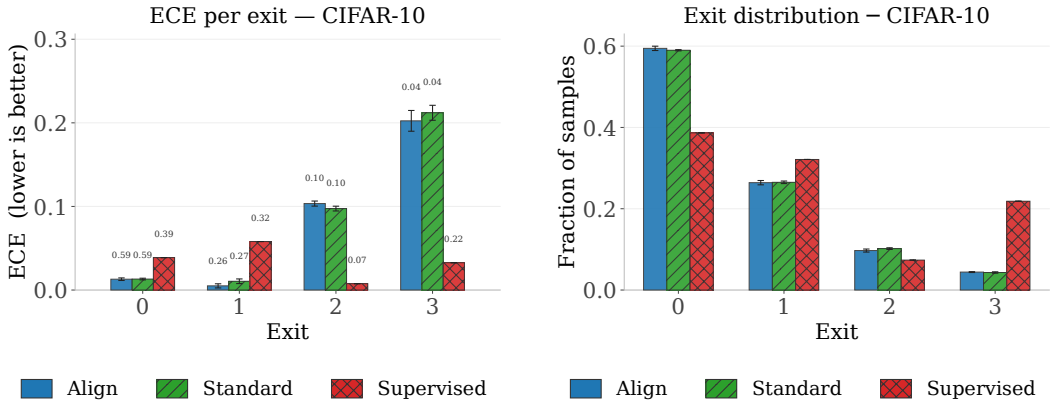


Figure 6: The Pareto frontier indicates that the proposed method across both **Align** and **Standard** is able to achieve comparable accuracy at reduced latency. In addition, it seems that for lower thresholds the proposed method is able to achieve a small speedup accompanied by a drop of accuracy.

D.1.2 RESNET34 & SIGREG

ECE & Exit Distribution



(a) ResNet34 - ECE

(b) ResNet34 - Exit Distribution

Figure 7: ECE across the shallow exit branches indicates that for both **Align** and **Standard** there is better calibration in comparison to **Supervised**. The exit distribution indicates that in comparison to the baseline more samples exit early.

Pareto Frontier

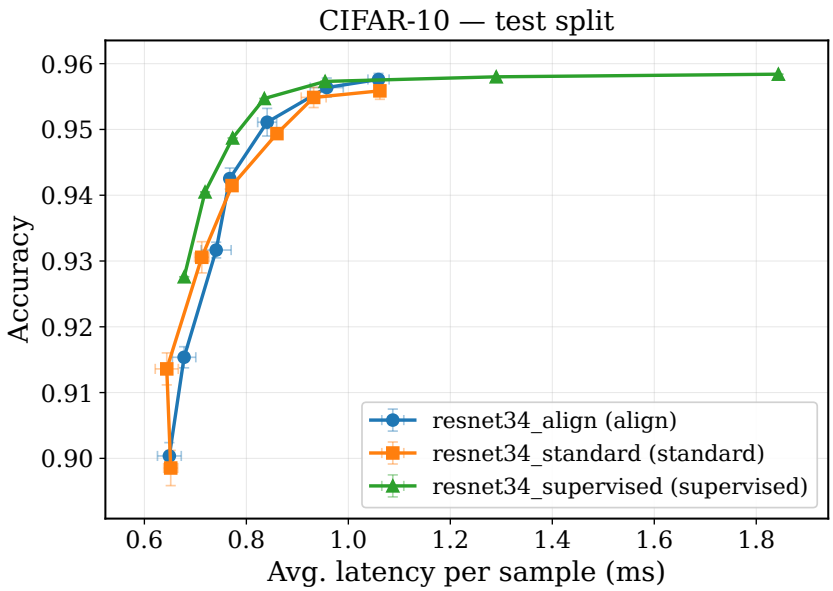
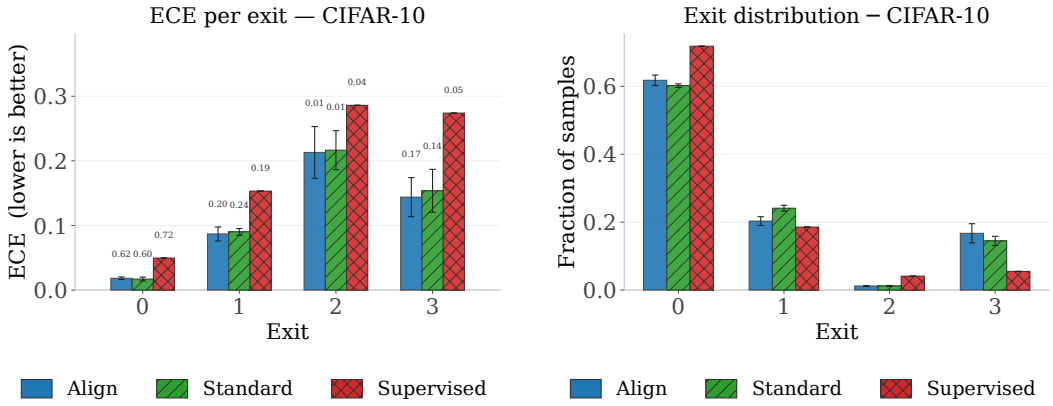


Figure 8: The Pareto frontier indicates that the proposed method across both **Align** and **Standard** is able to achieve comparable accuracy at reduced latency. In addition, it seems that for lower thresholds the proposed method is able to achieve a small speedup accompanied by a drop of accuracy.

D.1.3 ViT-S & VICReg

ECE & Exit Distribution



(a) ViT-S - ECE

(b) ViT-S - Exit Distribution

Figure 9: ECE across the shallow exit branches indicates that for both **Align** and **Standard** there is better calibration in comparison to **Supervised**. However, it seems that better calibration is not directly translated into more samples exiting early as explained in the main text. The reasons behind this result may be the fact that supervised training of a ViT architecture, without pretrained weights, on a dataset such as CIFAR10 could be unstable. This dataset consists of images of small dimensions and the number of samples might not be sufficient for such architectures. The results can help us understand the impact of using such datasets on ViT architectures using our methodology.

Pareto Frontier

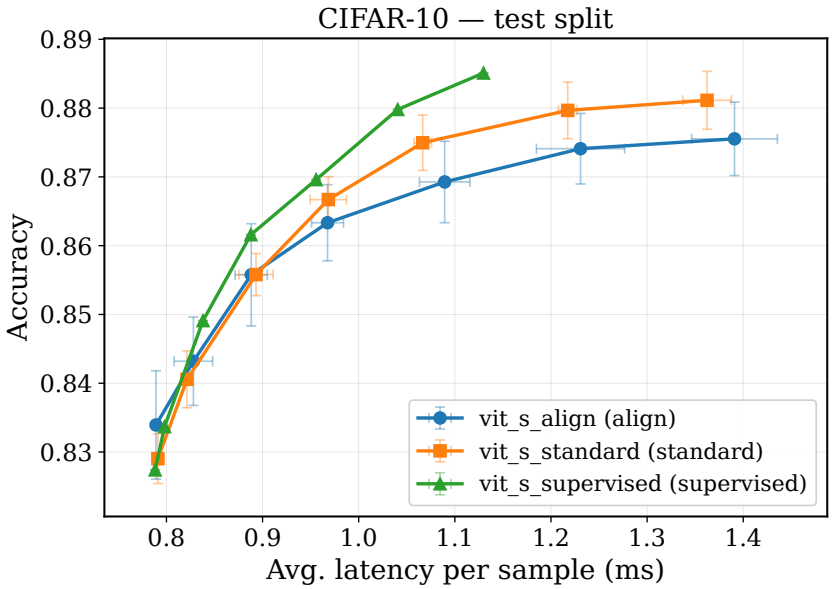
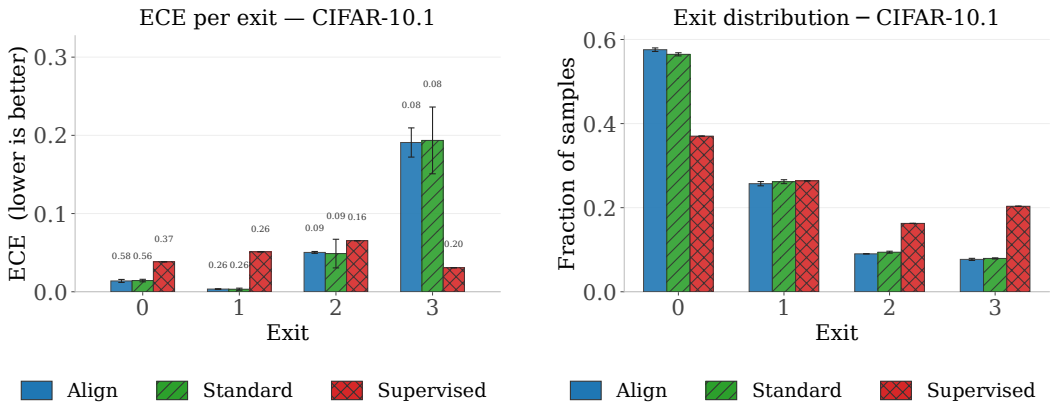


Figure 10: The Pareto frontier indicates that despite the better calibration of shallow exits the efficiency trade-off is not superior. This is a case that further validates the decision to observe efficiency and calibration individually with no assumption of correlation.

D.2 CIFAR10.1

D.2.1 RESNET18 & SIGREG

ECE & Exit Distribution



(a) ResNet18 - ECE

(b) ResNet18 - Exit Distribution

Figure 11: ECE across the shallow exit branches indicates that for both **Align** and **Standard** there is better calibration in comparison to **Supervised**. The exit distribution indicates that in comparison to the baseline more samples exit early.

Pareto Frontier

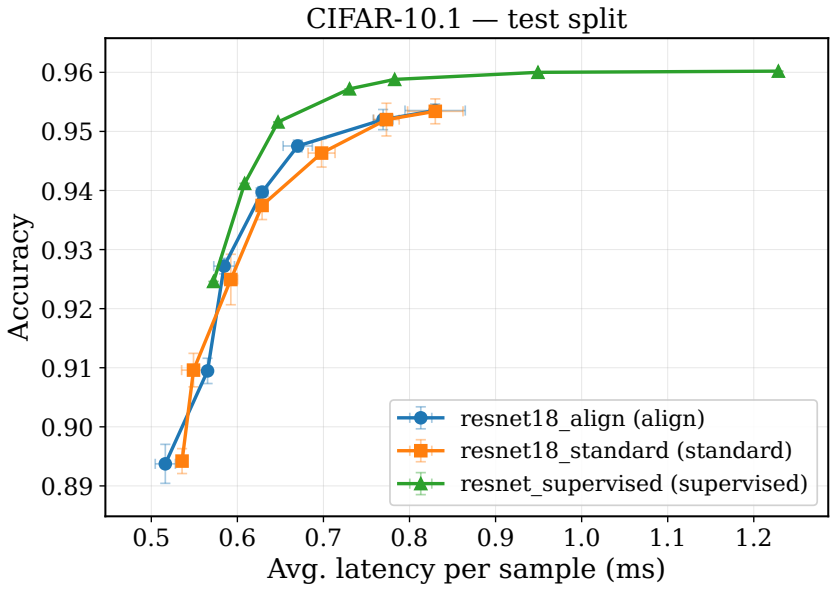
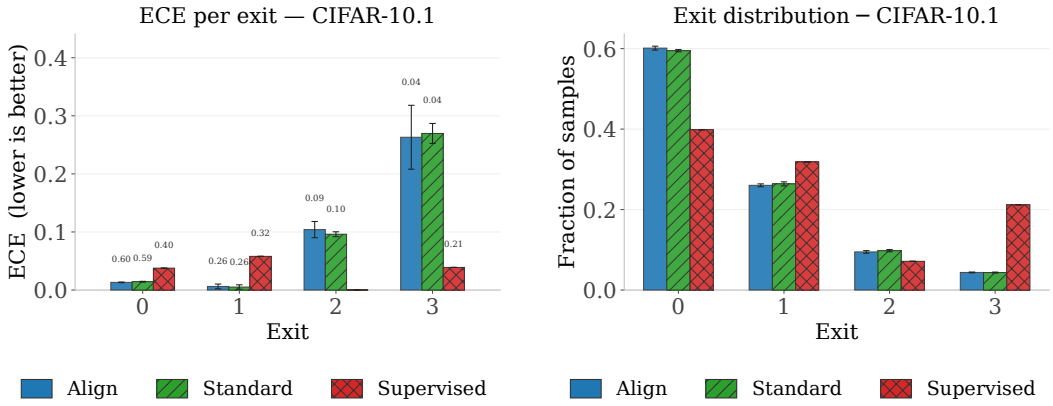


Figure 12: The Pareto frontier indicates that the proposed method across both **Align** and **Standard** is able to achieve comparable accuracy at reduced latency. In addition, it seems that for lower thresholds the proposed method is able to achieve a small speedup accompanied by a drop of accuracy.

D.2.2 RESNET34 & SIGREG

ECE & Exit Distribution



(a) ResNet34 - ECE

(b) ResNet34 - Exit Distribution

Figure 13: ECE across the shallow exit branches indicates that for both **Align** and **Standard** there is better calibration in comparison to **Supervised**. The exit distribution indicates that in comparison to the baseline more samples exit early.

Pareto Frontier

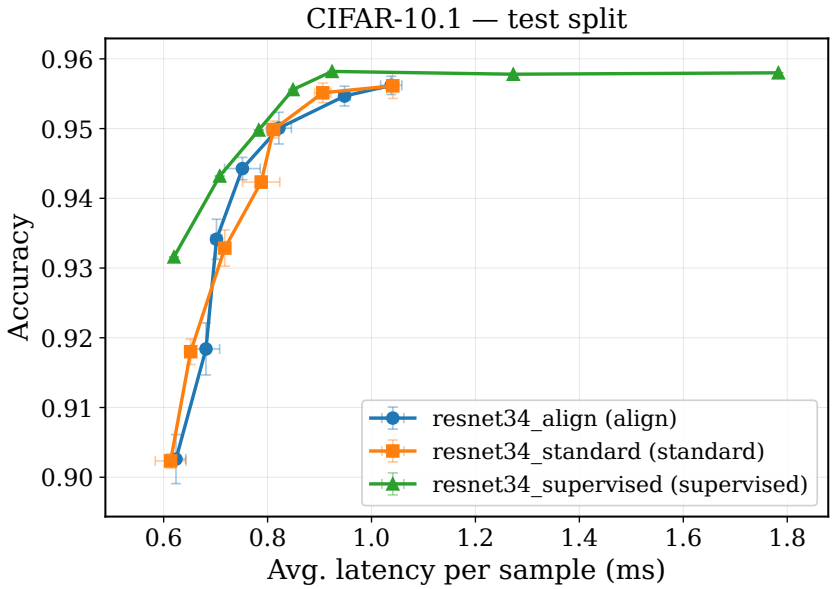
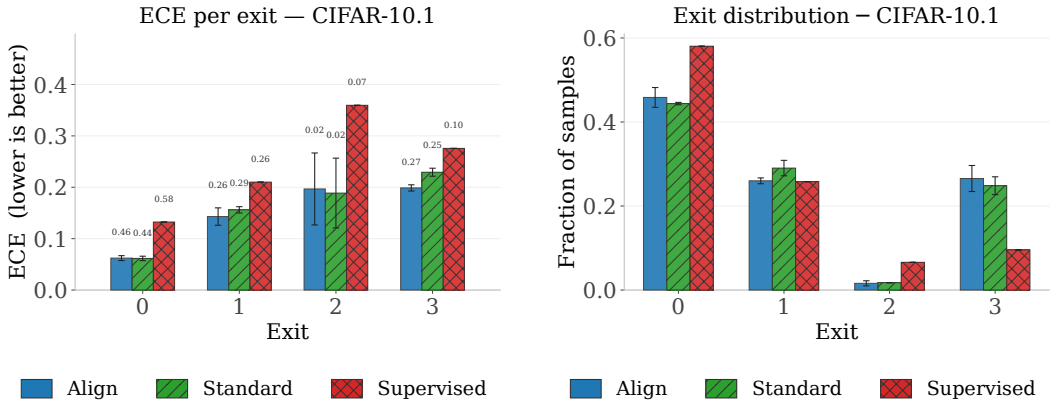


Figure 14: The Pareto frontier indicates that the proposed method across both **Align** and **Standard** is able to achieve comparable accuracy at reduced latency. In addition, it seems that for lower thresholds the proposed method is able to achieve a small speedup accompanied by a drop of accuracy.

D.2.3 ViT-S & VICReg

ECE & Exit Distribution



(a) ViT-S - ECE

(b) ViT-S - Exit Distribution

Figure 15: ECE across the shallow exit branches indicates that for both **Align** and **Standard** there is better calibration in comparison to **Supervised**. However, it seems that better calibration is not directly translated into more samples exiting early as explained in the main text. The reasons behind this result may be the fact that supervised training of a ViT architecture, without pretrained weights, on a dataset such as CIFAR10 could be unstable. This dataset consists of images of small dimensions and the number of samples might not be sufficient for such architectures. The results can help us understand the impact of using such datasets on ViT architectures using our methodology.

Pareto Frontier

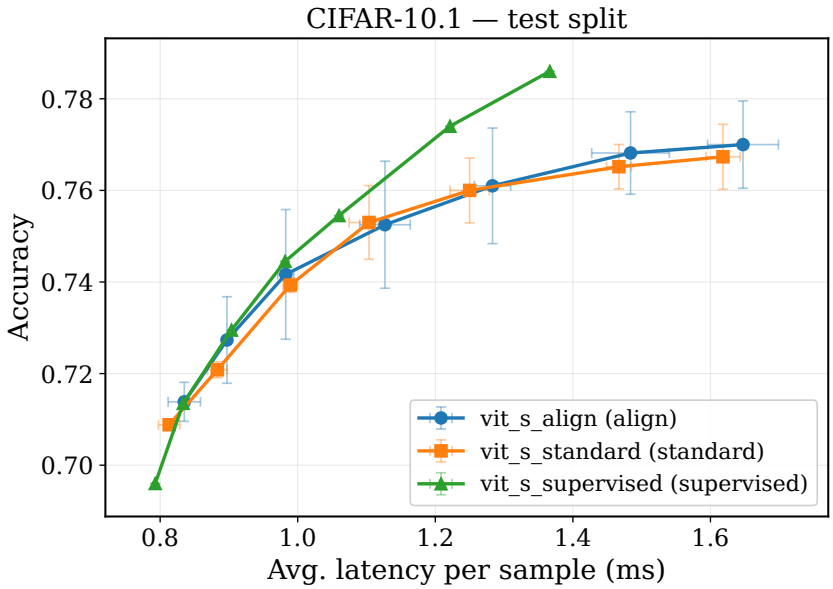
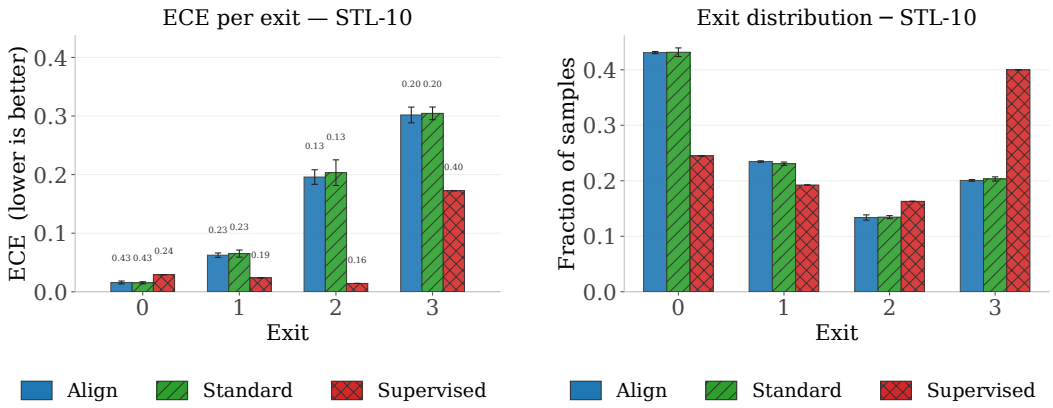


Figure 16: The Pareto frontier indicates that despite the better calibration of shallow exits the efficiency trade-off is not superior.

D.3 STL-10

D.3.1 RESNET18 & SIGREG

ECE & Exit Distribution



(a) ResNet18 - ECE

(b) ResNet18 - Exit Distribution

Figure 17: ECE across the shallow exit branches indicates that for both **Align** and **Standard** there is better calibration for the shallowest exit in comparison to **Supervised** while more samples exit early. This case further highlights the main text claims that calibration and efficiency trade-offs should be assessed individually in confidence-threshold early-exit neural networks.

Pareto Frontier

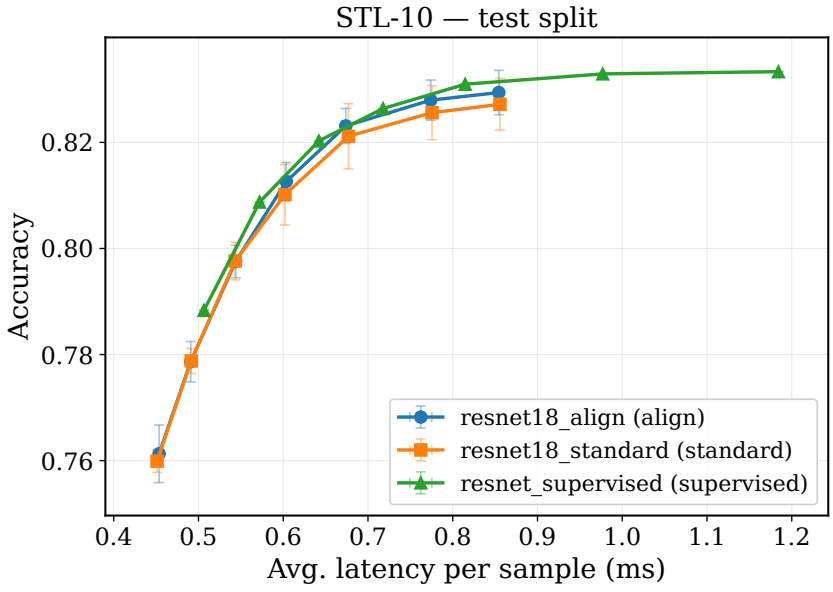
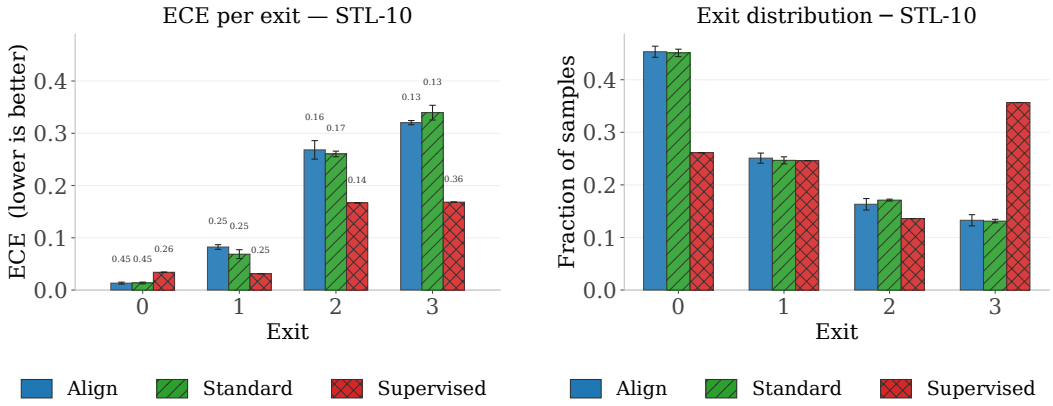


Figure 18: The Pareto frontier indicates that the proposed method across both **Align** and **Standard** is able to achieve comparable accuracy at reduced latency. In addition, it seems that for lower thresholds the proposed method is able to achieve a small speedup accompanied by a drop of accuracy.

D.3.2 RESNET34 & SIGREG

ECE & Exit Distribution



(a) ResNet34 - ECE

(b) ResNet34 - Exit Distribution

Figure 19: The Pareto frontier indicates that the proposed method across both **Align** and **Standard** is able to achieve comparable accuracy at reduced latency. In addition, it seems that for lower thresholds the proposed method is able to achieve a small speedup accompanied by a drop of accuracy.

Pareto Frontier

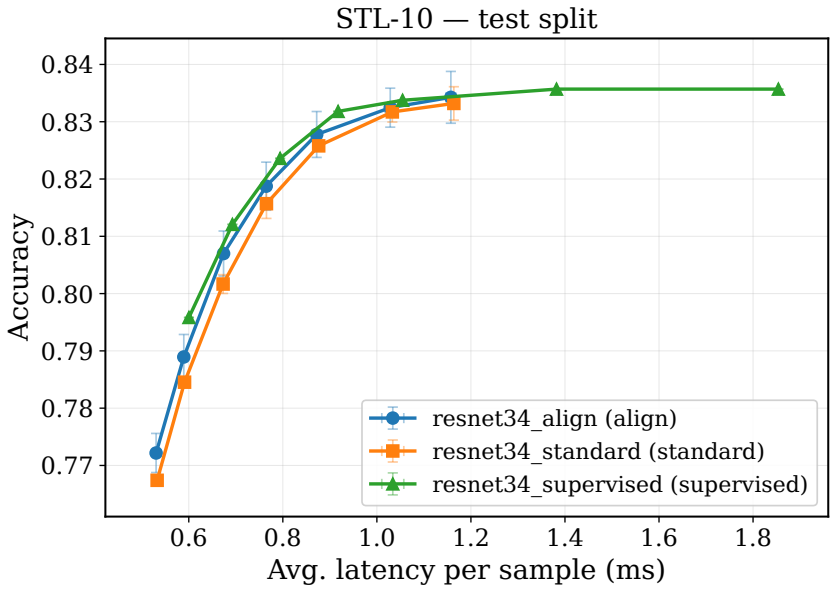
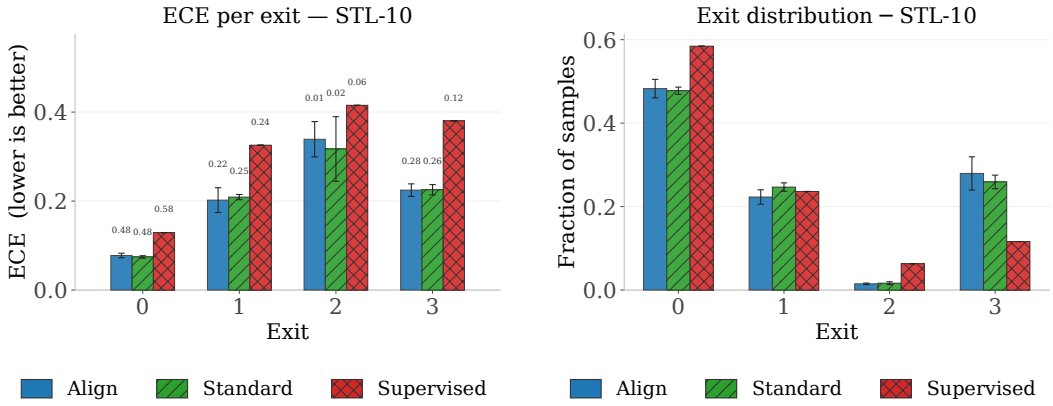


Figure 20: The Pareto Frontier indicates that the proposed methodology achieves comparable accuracy with lower latency per sample whereas for smaller thresholds there is an accuracy drop.

D.3.3 ViT-S & VICReg

ECE & Exit Distribution



(a) ViT-S - ECE

(b) ViT-S - Exit Distribution

Figure 21: ECE across the shallow exit branches indicates that for both **Align** and **Standard** there is better calibration in comparison to **Supervised**. However, it seems that better calibration is not directly translated into more samples exiting early as explained in the main text. The reasons behind this result may be the fact that training a ViT architecture, without pretrained weights, on a dataset such as CIFAR10 could be unstable. This dataset consists of images of small dimensions and the number of samples might not be sufficient for such architectures. The results can help us understand the impact of using such datasets on ViT architectures using our methodology under distribution shift.

Pareto Frontier

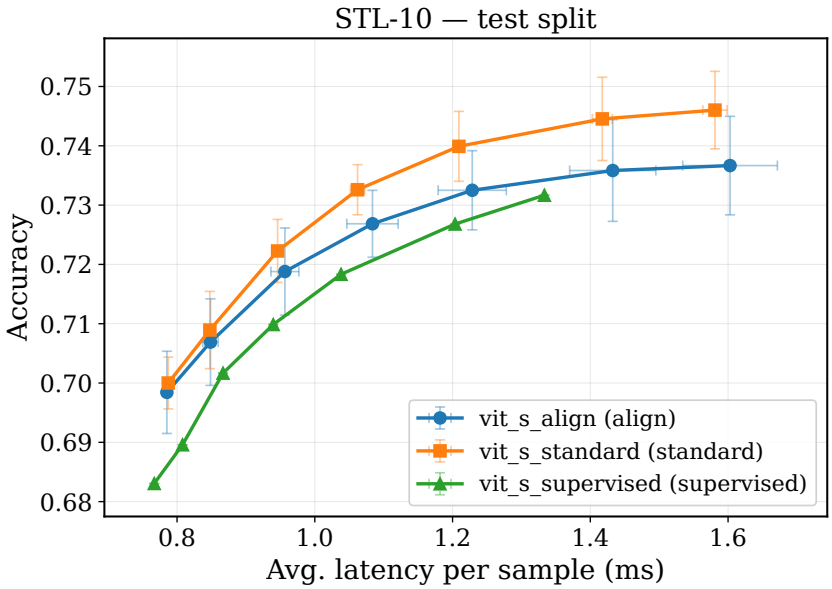
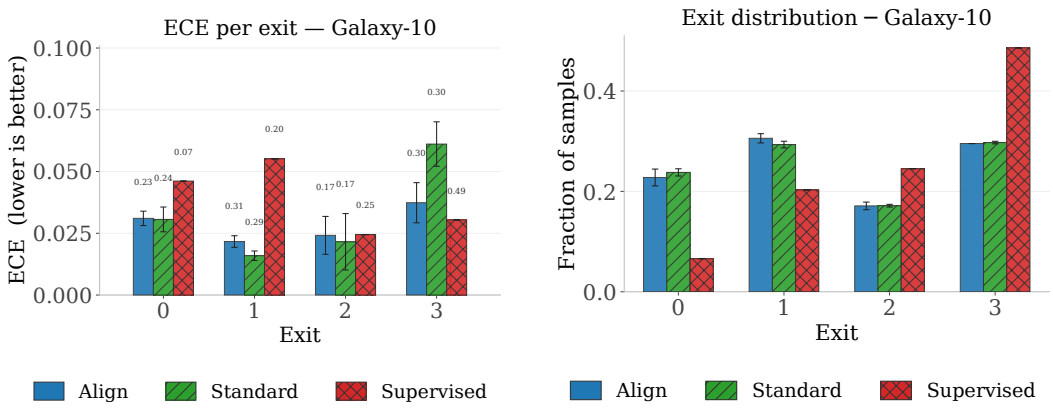


Figure 22: The Pareto frontier suggests that across thresholds the proposed method outperforms the supervised baseline in terms of accuracy however it appears that latency is increased.

D.4 GALAXY10

D.4.1 RESNET34 & VICREG

ECE & Exit Distribution



(a) ResNet34 - ECE

(b) ResNet34 - Exit Distribution

Figure 23: ECE across the shallow exit branches indicates that for both **Align** and **Standard** there is better calibration in comparison to **Supervised** while using a dataset with samples of bigger dimensions (256×256). In addition, it seems that more samples exit early in comparison to the baseline.

Pareto Frontier

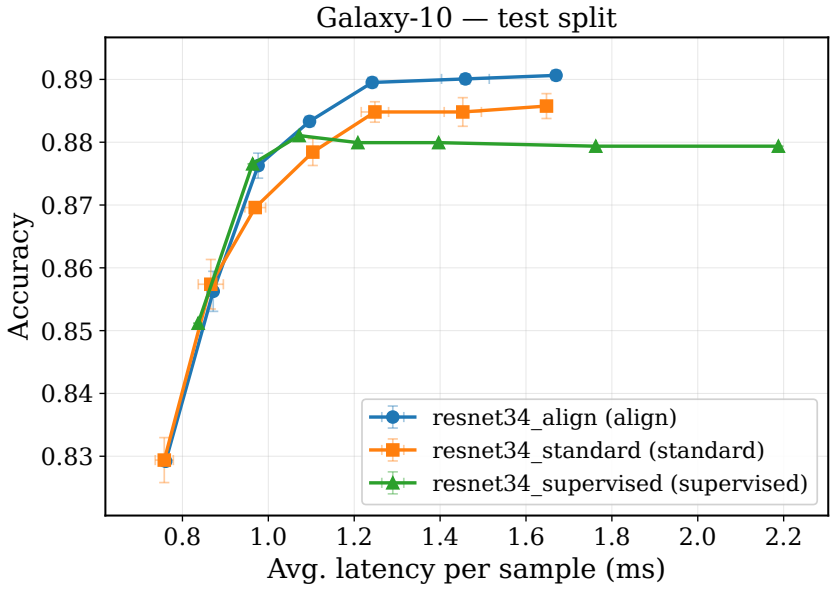
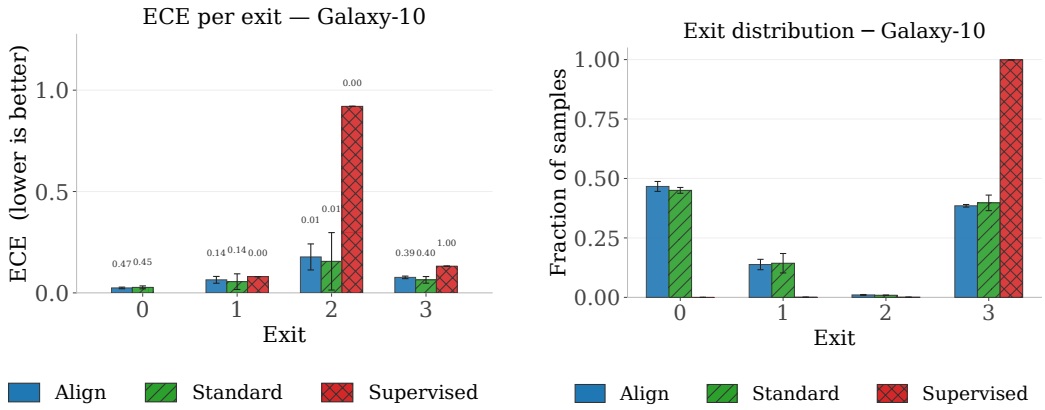


Figure 24: The Pareto frontier indicates that at lower thresholds there is an accuracy drop in favor of slightly reduced latency. In contrast, for larger thresholds the accuracy is increased compared to the supervised baseline accompanied by reduced latency too.

D.4.2 ViT-S & VICReg

ECE & Exit Distribution



(a) ViT-S - ECE

(b) ViT-S - Exit Distribution

Figure 25: ECE across the shallow exit branches indicates that for both **Align** and **Standard** there is better calibration for the shallowest exit branch in comparison to **Supervised** while more samples exit early. However, as explained for CIFAR10, datasets such as Galaxy10 might be challenging for ViT architectures without pretrained weights because of data scarcity. However, our results demonstrate that our SSL approach is not as affected as the baseline.

Pareto Frontier

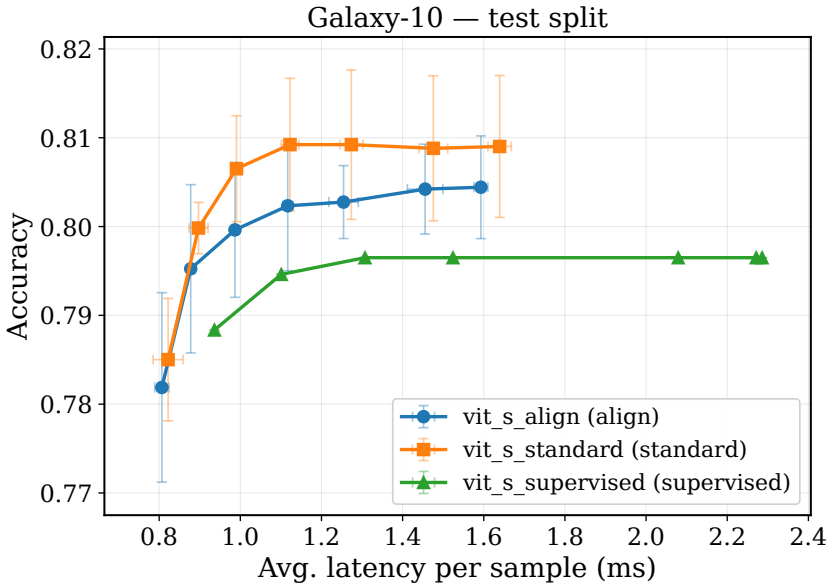
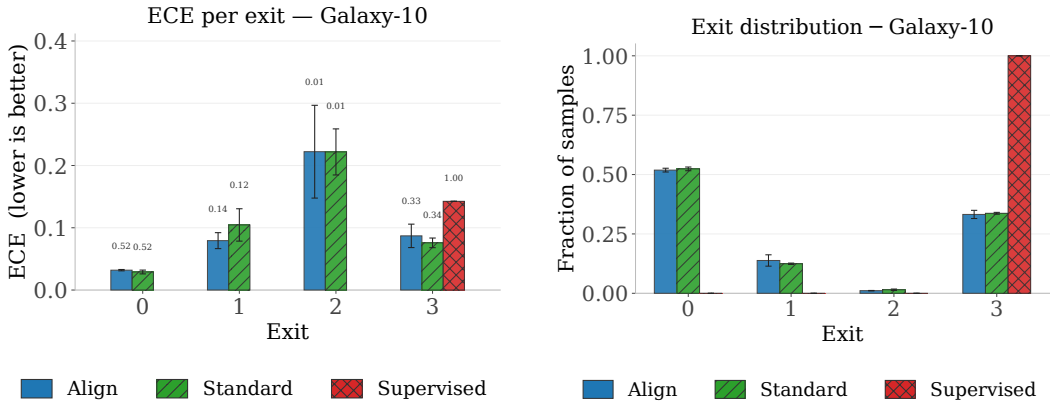


Figure 26: The Pareto frontier indicates that in terms of accuracy the results across all approaches are comparable, however the proposed method achieves reduced latency.

D.4.3 ViT-B & VICReg

ECE & Exit Distribution



(a) ViT-B - ECE

(b) ViT-B - Exit Distribution

Figure 27: ECE across the shallow exit branches indicates that for both **Align** and **Standard** there is better calibration for the shallowest exit branch in comparison to **Supervised** while more samples exit early. However, as explained for CIFAR10, datasets such as Galaxy10 might be challenging for ViT architectures without pretrained weights because of data scarcity. However, our results demonstrate that our SSL approach is not as affected as the baseline.

Pareto Frontier

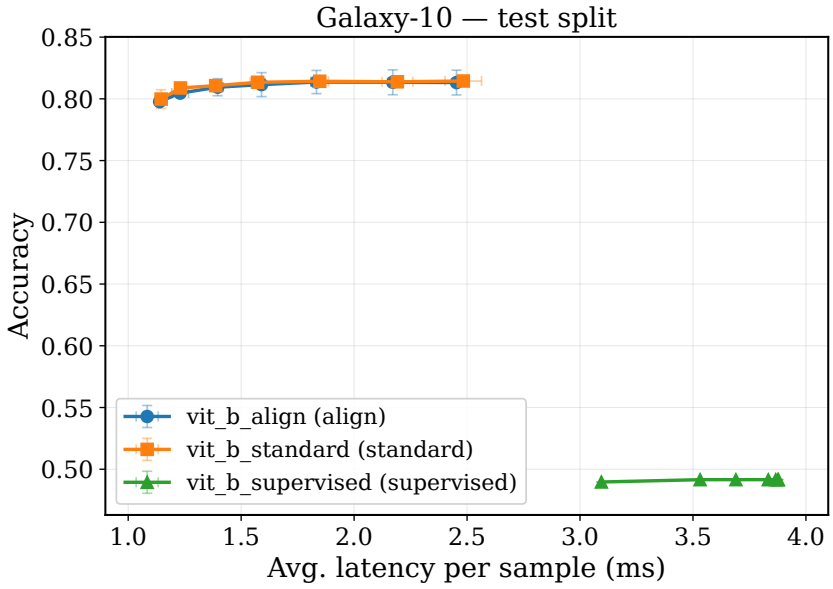
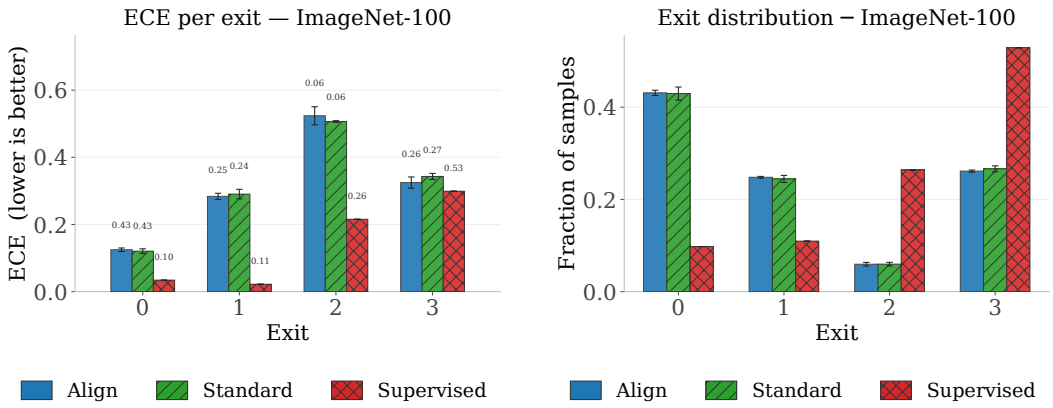


Figure 28: The Pareto frontier indicates that in terms of accuracy the SSL pretraining leads not only to better downstream accuracy but reduced latency too. The ViT-B supervised baseline is not able to integrate early-exit behavior or achieve better accuracy through the mainstream joint optimization paradigm, even using Mixup or Label Smoothing.

D.5 IMAGENET100

D.5.1 VIT-B & VICREG

ECE & Exit Distribution



(a) ViT-B - ECE

(b) ViT-B - Exit Distribution

Figure 29: ECE across the shallow exit branches indicates that for both **Align** and **Standard** there is better calibration in comparison to **Supervised** while more samples exit early. The existence of more training samples is affecting positively the **Supervised** results in comparison to the experiments of ViT-B on Galaxy10, as it seems that some of the samples manage to exit early.

Pareto Frontier

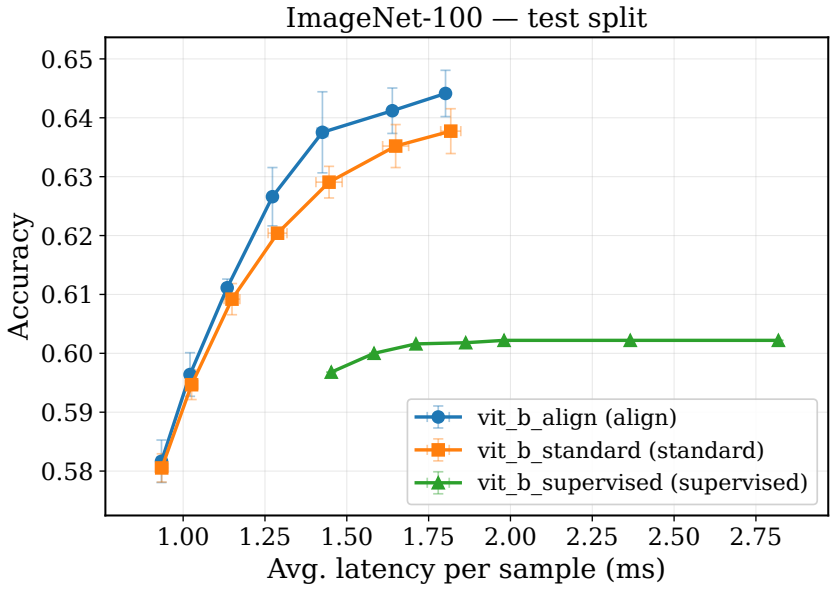


Figure 30: The Pareto frontier indicates that for lower thresholds the proposed method results in accuracy drop accompanied with latency reduction in comparison to the supervised baseline whereas for higher thresholds there is greater accuracy and reduced latency.

D.6 VICREG & SIGREG

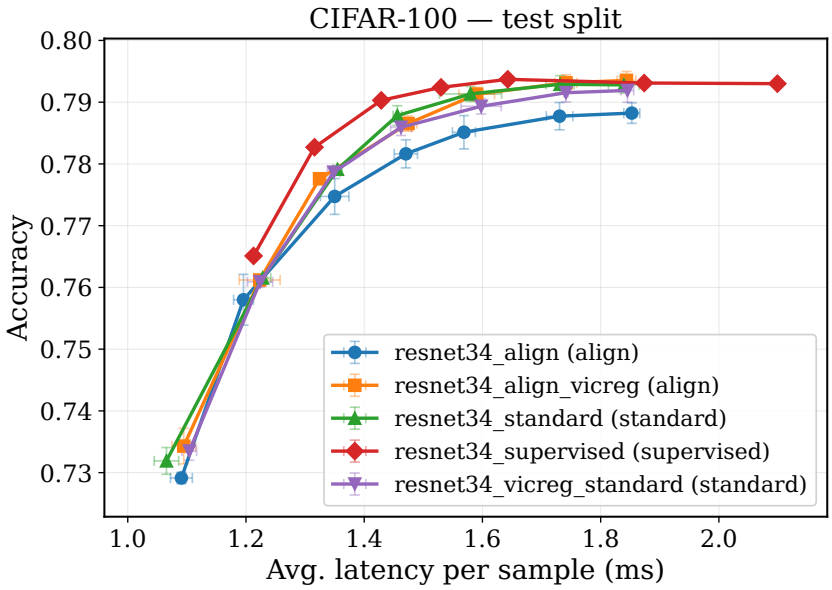


Figure 31: The Pareto frontier indicates that the regardless of the non-contrastive regularizer choice studied in this work the efficiency trade-off results are comparable.

D.7 NT-XENT

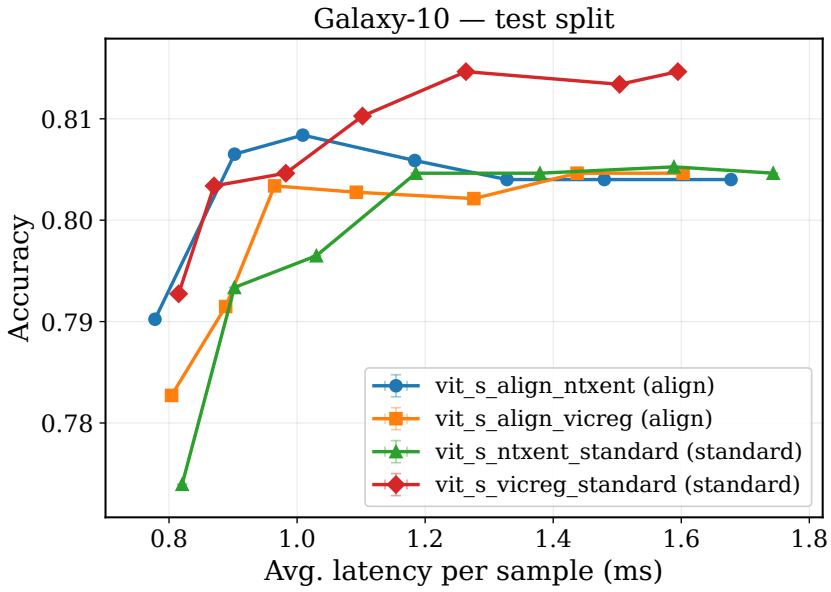


Figure 32: NT-Xent Batch Size 512 - The Pareto frontier indicates that the contrastive objective leads to comparable efficiency trade-off.

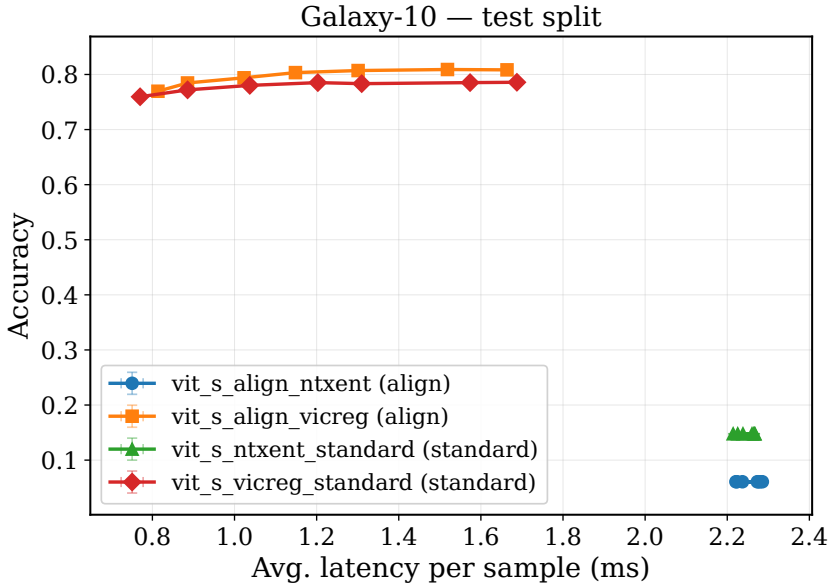


Figure 33: NT-Xent Batch Size 32 - The Pareto frontier indicates that the use of the contrastive objective results in collapse for small patch size values.

D.8 Representation Quality

D.8.1 CKA

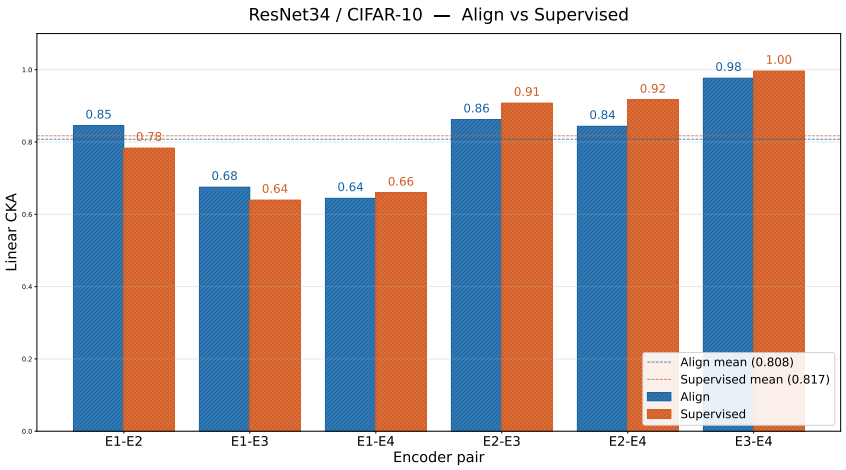


Figure 34: Inter-layer CKA comparison between **Supervised** and **Align**.

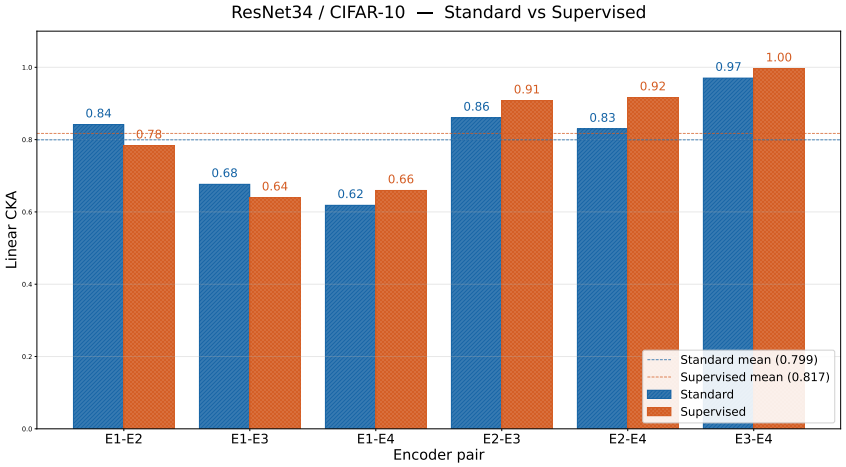


Figure 35: Inter-layer CKA comparison between **Supervised** and **Standard**.

D.8.2 tSNE & Probing

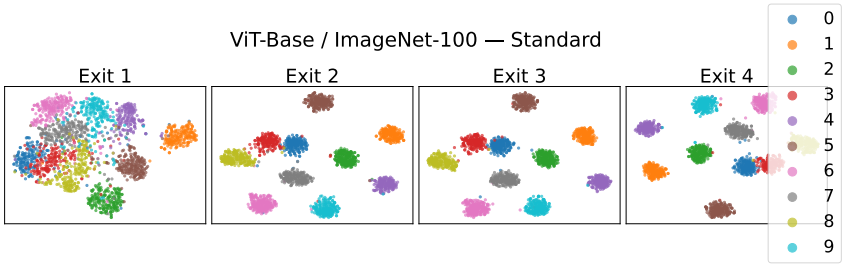


Figure 36: tSNE per exit-branch for ViT-B on 10 randomly selected classes of ImageNet100 for the **Standard** variant of the proposed method.

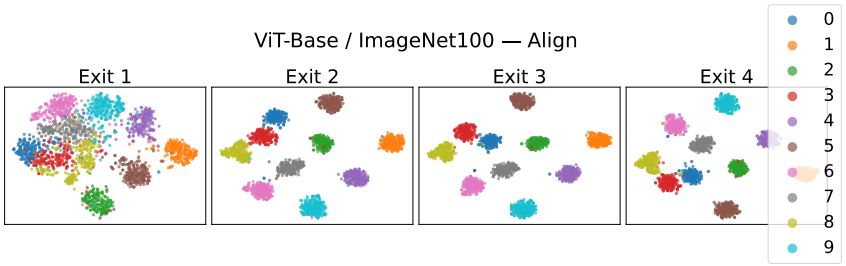


Figure 37: tSNE per exit-branch for ViT-B on 10 randomly selected classes of ImageNet100 for the **Align** variant of the proposed method.

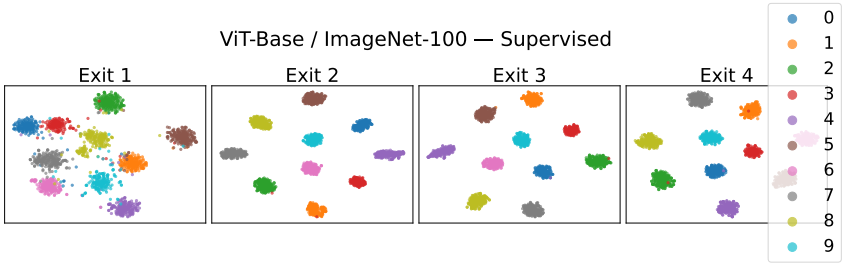


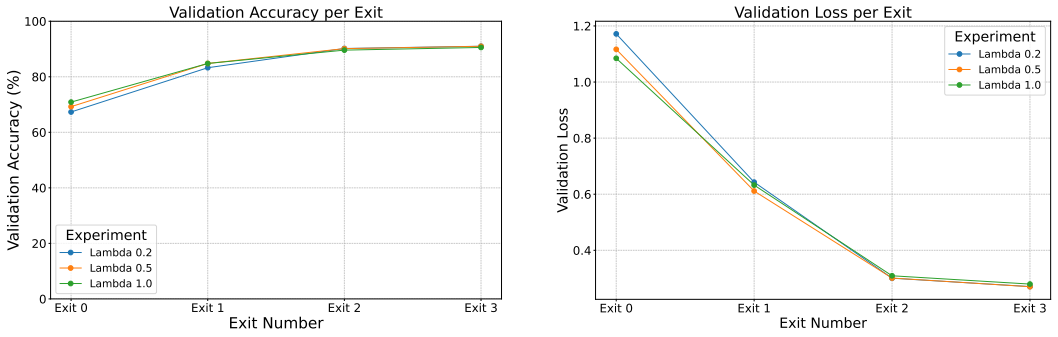
Figure 38: tSNE per exit-branch for ViT-B on 10 randomly selected classes of ImageNet100 for the **Supervised** approach.

Table 14: Linear-probe and kNN (k=20) accuracy on the features of each exit of ViT-Base, evaluated on a held-out set of 250 images from 10 randomly chosen ImageNet-100 classes. Probes are fit on non-augmented training features. The best value per row is in bold. With 250 test images, the 95% sampling interval on each accuracy is roughly ±5 points.

	Linear probe acc.			kNN acc.		
Exit	Align	Standard	Supervised	Align	Standard	Supervised
1	0.772	**0.796**	0.744	0.752	**0.784**	0.708
2	0.832	**0.860**	0.748	0.828	**0.840**	0.736
3	0.836	**0.848**	0.704	0.824	**0.832**	0.676
4	0.840	**0.848**	0.692	0.812	**0.836**	0.656

Table 14 reports how much class information is linearly and locally accessible in the features of each exit. Align and Standard outperform Supervised at Exits 2–4 by 8–15 percentage points on both metrics, well beyond the sampling noise of the 250-image test set. At Exit 1 the gap is smaller (3–5 points) and we do not consider it conclusive. The quality of Supervised features also decreases with depth (linear probe: 0.748 at Exit 2 vs. 0.692 at Exit 4), whereas Align and Standard improve or plateau. Align and Standard are statistically indistinguishable in our experiments: Standard is nominally ahead by 0.8–2.8 points, but these differences are within the sampling interval, so we conclude that Align matches Standard rather than improves on it. Finally, both methods already reach near-final accuracy at Exit 2 (e.g. 0.832 vs. 0.840 for Align in the linear probe), which is favorable for early exiting.

D.9 λ SENSITIVITY



(a) Accuracy - (3 λ values)

(b) Validation Loss - (3 λ values)

Figure 39: The effect of changing the λ parameter seems to have negligible impact for ResNet34 (SIGReg) on CIFAR10.

The cross-depth alignment term is regularized by a factor λ. Since, we are introducing this term and there is no prior point of reference it was necessary to explore the impact of its value on the performance of the encoder. This ablation study, explores the performance of the encoder, based on the probes used for monitoring during the SSL training. The set of λ values chosen is 0.2, 0.5, 1.0, where the validation loss and accuracy of the individual exits at the last epoch of training are used as evaluation metrics. We observe that the values of the evaluation metrics are almost overlapping and therefore there is not significant impact regarding the resulting encoder. Our results however, indicate that cross-depth alignment impacts the performance in itself in a manner comparable to the **Standard** approach.

E CIFAR10 Adjacent Classification

E.1 Animal to Animal

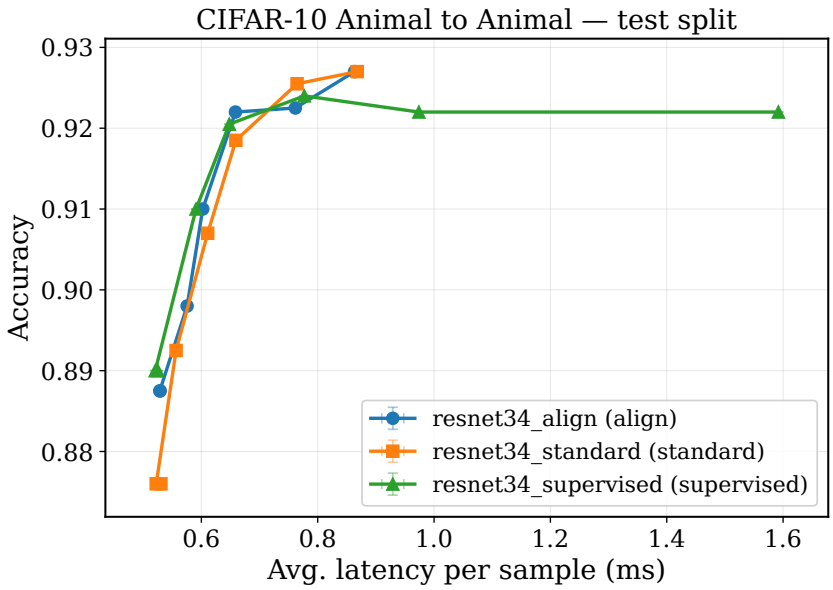


Figure 40: The Pareto frontier on the cat and dog labels, indicates that both models are comparable in terms of efficiency, however the proposed encoder needs at most 50 epoch of training.

E.2 Vehicle to Vehicle

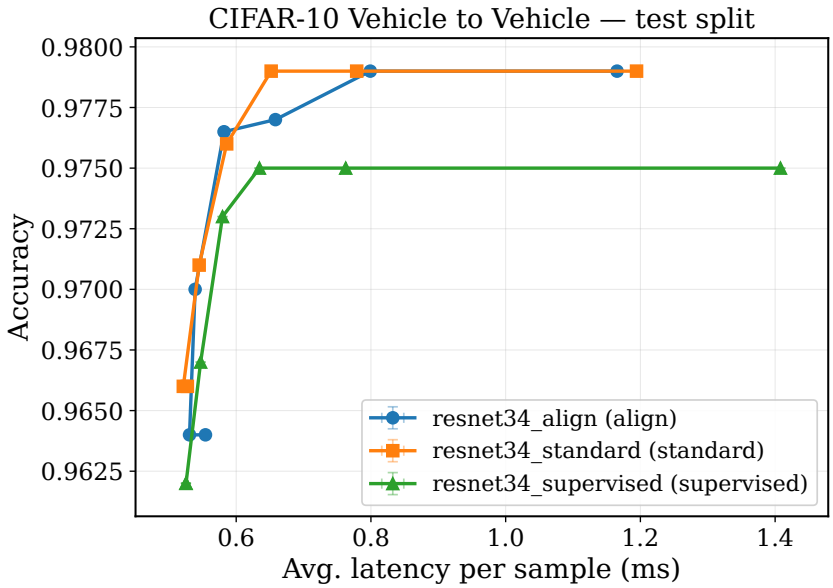


Figure 41: The Pareto frontier on the automobile and truck labels, indicates that both models are comparable in terms of efficiency, however the proposed encoder needs at most 50 epoch of training.

E.3 BACKGROUND

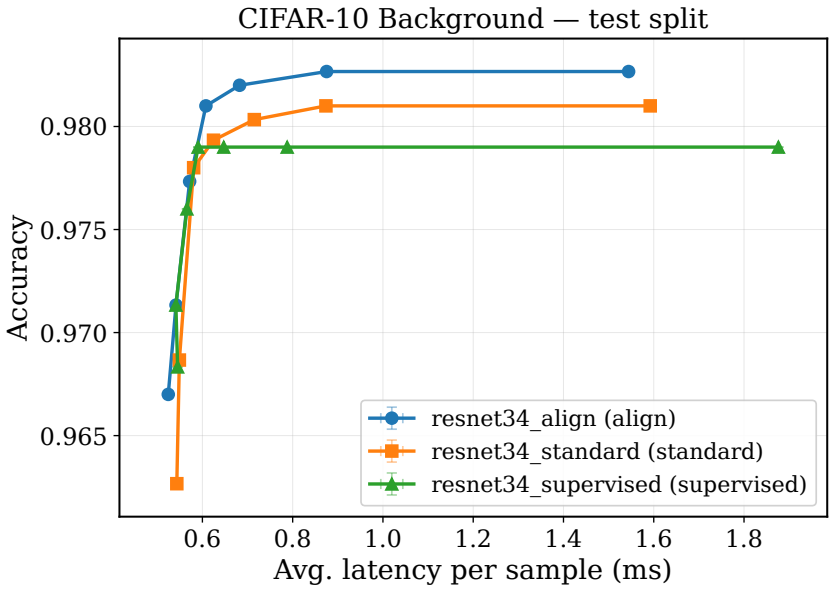


Figure 42: The Pareto frontier on the airplane, ship and frog labels, indicates that both models are comparable in terms of efficiency, however the proposed encoder needs at most 50 epoch of training.

The figures described above demonstrate that the representations can be used again across tasks that require similar representations and there is no need to train a supervised multi-exit neural network from scratch. The supervised models across the three experiments trained in total for 900 epochs whereas the models based on our encoders trained in total for 150 epochs or less based on convergence.