Cleaning the Pool: Progressive Filtering of Unlabeled Pools in Deep Active Learning

Supplementary Material

A. Proofs of Theorems

Here, we provide detailed proofs for the theorems from the main paper. Furthermore, to complement the theoretical analysis of Theorems 1 and 2, we also include numerical examples illustrating the probability of preserving valuable instances and filtering out uninformative ones.

Proof of Theorem 1. Let $P_r(\boldsymbol{x}) = \Pr(\boldsymbol{x} \in \mathcal{C}_r \mid \boldsymbol{x} \in \mathcal{C}_{r-1})$ be the probability of instance $\boldsymbol{x}$ surviving round r. Given $\boldsymbol{x} \in \mathcal{C}_{r-1}$, let $S_{m,r}$ be the *event* that AL strategy s_m selects $\boldsymbol{x}$ in round r at least once in its J trials. The total survival event is the union of these individual events, *i.e.*, $\boldsymbol{x}$ survives if at least one strategy selects it:

$$P_r(\boldsymbol{x}) = \Pr\left(\bigcup_{m=1}^{M} S_{m,r}\right).$$

We first derive $\Pr(S_{m,r})$ and then use this to find the lower bound for the union.

1. For a single draw $j \in \{1, \ldots, J\}$ by strategy s_m, the event of selecting $\boldsymbol{x}$ requires two *independent conditions* to be met:
   (a) Instance $\boldsymbol{x}$ must be present in the random subsample $\mathcal{U}_{\text{sample}}$. The probability of this is
   $$\Pr(\boldsymbol{x} \in \mathcal{U}_{\text{sample}}) = \alpha.$$
   (b) Strategy s_m must select $\boldsymbol{x}$ from that subsample. By definition, this probability is $p_{m,r}(\boldsymbol{x})$.
2. The joint probability of $\boldsymbol{x}$ being selected in a single draw j is the product of these probabilities:
   $$\Pr(\text{select } \boldsymbol{x} \text{ in draw } j) = \alpha \cdot p_{m,r}(\boldsymbol{x}).$$
3. The probability of $\boldsymbol{x}$ *not being selected* in this single draw is $1 - \alpha \cdot p_{m,r}(\boldsymbol{x})$.
4. As per Algorithm 1, strategy s_m makes J independent draws (each with a new subsample). The probability that $\boldsymbol{x}$ is *never selected* by s_m across all J draws is the product of their individual probabilities:
   $$\Pr(\bar{S}_{m,r}) = \prod_{j=1}^{J} (1 - \alpha \cdot p_{m,r}(\boldsymbol{x})) = (1 - \alpha \cdot p_{m,r}(\boldsymbol{x}))^J.$$
5. Therefore, the probability that $\boldsymbol{x}$ is selected *at least once* by strategy s_m is the complement:
   $$\Pr(S_{m,r}) = 1 - \Pr(\bar{S}_{m,r}) = 1 - (1 - \alpha \cdot p_{m,r}(\boldsymbol{x}))^J.$$

Having found $\Pr(S_{m,r})$, we now derive the lower bound.

1. The probability of a union of events is always greater than or equal to the probability of any single event in that union. Therefore, for any strategy s_i:
   $$P_r(\boldsymbol{x}) = \Pr\left(\bigcup_{m=1}^{M} S_{m,r}\right) \geq \Pr(S_{i,r}).$$
2. Since this holds for all i, it must also hold for the maximum probability among them:
   $$P_r(\boldsymbol{x}) \geq \max_{m \in \{1,\ldots,M\}} \Pr(S_{m,r}).$$
3. Substituting the expression for $\Pr(S_{m,r})$ from Step 5:
   $$P_r(\boldsymbol{x}) \geq \max_{m \in \{1,\ldots,M\}} \left[1 - (1 - \alpha \cdot p_{m,r}(\boldsymbol{x}))^J\right].$$
4. The function $f(p) = 1 - (1 - \alpha p)^J$ is monotonically increasing with p. Therefore, the maximum value of the function is achieved when p is at its maximum. Let $p_{\max} = \max_{m \in \{1,\ldots,M\}} p_{m,r}(\boldsymbol{x})$. Then:
   $$\max_m \left[1 - (1 - \alpha \cdot p_{m,r}(\boldsymbol{x}))^J\right] = 1 - (1 - \alpha \cdot p_{\max})^J.$$
5. This proves the final bound:
   $$P_r(\boldsymbol{x}) \geq 1 - \left(1 - \alpha \cdot \max_{m \in \{1,\ldots,M\}} p_{m,r}(\boldsymbol{x})\right)^J. \quad \blacksquare$$

Numerical Example. We provide a numerical example to illustrate the probability of preserving a valuable instance. We consider $\alpha = 0.4$, $J = 10$, and an instance $\boldsymbol{x}$ that receives a high score from Margin ($p_{\text{margin}} = 0.9$) but a low score from TypiClust ($p_{\text{typiclust}} = 0.2$). We assume an ensemble of two strategies, where the instance's true value is correlated with the uncertainty heuristic, which is accounted for by the Margin. Since the bound uses the maximum score, we have $p_{\max} = 0.9$. This yields $P_r(\boldsymbol{x}) \geq 1 - (1 - 0.4 \cdot 0.9)^{10} = 1 - (0.64)^{10} \approx 0.9885$, demonstrating that the instance has a high lower-bound probability because at least one strategy strongly prioritizes it.

Proof of Theorem 2. Let $\boldsymbol{x}$ be an ϵ-uninformative instance and let $P_r(\boldsymbol{x}) = \Pr(\boldsymbol{x} \in \mathcal{C}_r \mid \boldsymbol{x} \in \mathcal{C}_{r-1})$ be the probability that an ϵ-uninformative instance $\boldsymbol{x}$ survives round r. We first derive an upper bound for $P_r(\boldsymbol{x})$ and then use this to find the bound for surviving all R rounds.

1. For a single draw $j \in \{1, \ldots, J\}$ by strategy s_m, the event of selecting $\boldsymbol{x}$ requires *two conditions* to be met:

(a) Instance $\boldsymbol{x}$ must be present in the random subsample $\mathcal{U}_{\text{sample}}$. The probability of this is

$$\Pr(\boldsymbol{x} \in \mathcal{U}_{\text{sample}}) = \alpha.$$

(b) Strategy s_m must select $\boldsymbol{x}$ from that subsample. By Definition 1, this probability is bounded: $p_{m,r}(\boldsymbol{x}) \leq \epsilon$.

2. The joint probability of $\boldsymbol{x}$ being selected in a single draw j by strategy s_m is bounded:

$$\Pr(\text{select } \boldsymbol{x} \text{ in draw } (m, j)) = \alpha \cdot p_{m,r}(\boldsymbol{x}) \leq \alpha\epsilon.$$

3. The probability of $\boldsymbol{x}$ *not* being selected in this single draw is therefore lower-bounded:

$$\Pr(\text{not select } \boldsymbol{x} \text{ in draw } (m, j)) \geq 1 - \alpha\epsilon.$$

4. As per Algorithm 1, we make a total of $M \cdot J$ independent draws in round r. The event that $\boldsymbol{x}$ is *never selected* requires it to be missed by all $M \cdot J$ draws. Using the lower bound from the previous step, we get a lower bound for $\boldsymbol{x}$ not surviving the round:

$$\Pr(\boldsymbol{x} \notin \mathcal{C}_r \mid \boldsymbol{x} \in \mathcal{C}_{r-1}) \geq \prod_{m=1}^{M} \prod_{j=1}^{J} (1-\alpha\epsilon) = (1-\alpha\epsilon)^{MJ}.$$

5. Consequently, the event that $\boldsymbol{x}$ is selected at least once is bounded by:

$$\begin{aligned} P_r(\boldsymbol{x}) &= \Pr(\boldsymbol{x} \in \mathcal{C}_r \mid \boldsymbol{x} \in \mathcal{C}_{r-1}) \\ &= 1 - \Pr(\boldsymbol{x} \notin \mathcal{C}_r \mid \boldsymbol{x} \in \mathcal{C}_{r-1}) \\ &\leq 1 - (1 - \alpha\epsilon)^{MJ}. \end{aligned}$$

Having found the upper bound for $P_r(\boldsymbol{x})$, we now derive the bound for surviving all R rounds.

1. For $\boldsymbol{x}$ to survive all R rounds, it must survive each round. The total survival probability is the product of the per-round survival probabilities:

$$\Pr(\boldsymbol{x} \in \mathcal{C}_R) = \prod_{r=1}^{R} \Pr(\boldsymbol{x} \in \mathcal{C}_r \mid \boldsymbol{x} \in \mathcal{C}_{r-1}) = \prod_{r=1}^{R} P_r(\boldsymbol{x}).$$

2. Since, by definition, instances $\boldsymbol{x}$ remain ϵ-uninformative across rounds (for all r):

$$\begin{aligned} \Pr(\boldsymbol{x} \in \mathcal{C}_R) &\leq \prod_{r=1}^{R} \left(1 - (1 - \alpha\epsilon)^{MJ}\right) \\ &= \left(1 - (1 - \alpha\epsilon)^{MJ}\right)^R, \end{aligned}$$

proving the first part of Theorem 2.

3. For the approximation, we assume $\alpha\epsilon \ll \frac{1}{MJ}$. Using the approximation $(1 - x)^n \approx 1 - nx$ for small x:

$$(1 - \alpha\epsilon)^{MJ} \approx 1 - MJ\alpha\epsilon.$$

4. Substituting this approximation back into the bound, we obtain that

$$\Pr(\boldsymbol{x} \in \mathcal{C}_R) \lesssim (1 - (1 - MJ\alpha\epsilon))^R = (MJ\alpha\epsilon)^R,$$

showing that the probability decreases exponentially with R. ■

Numerical Example. We provide a numerical example to illustrate the survival probability of an uninformative instance. We set $R = 5$, $M = 3$, $J = 10$, and $\alpha = 0.4$. Furthermore, we consider an ϵ-uninformative instance $\boldsymbol{x}$ with $\epsilon = 0.05$. The probability that this instance survives a single round is

$$\begin{aligned} P_r(\boldsymbol{x}) &\leq 1 - (1 - \alpha\epsilon)^{MJ} \\ &= 1 - (1 - 0.4 \cdot 0.05)^{30} \\ &= 1 - (0.98)^{30} \approx 0.4545. \end{aligned}$$

Extending this over all $R = 5$ rounds, the bound becomes $\Pr(\boldsymbol{x} \in \mathcal{C}_5) \leq (1 - (0.98)^{30})^5 \approx 0.0194$. After 5 rounds, the probability of this uninformative instance remaining in the candidate pool is bounded below 2%.

Proof of Theorem 3. Let $V(\boldsymbol{x})$ be the unknown true value of instance $\boldsymbol{x}$, and let $\mathbb{E}[V|\mathcal{C}] = \frac{1}{|\mathcal{C}|} \sum_{\boldsymbol{x} \in \mathcal{C}} V(\boldsymbol{x})$ be the expected average value of pool $\mathcal{C}$. We show that the expected value is non-decreasing in any single round r.

1. Let $\mathbb{E}[V \mid \mathcal{C}_{r-1}] = \frac{1}{|\mathcal{C}_{r-1}|} \sum_{\boldsymbol{x} \in \mathcal{C}_{r-1}} V(\boldsymbol{x})$ be the average value of the pool at the start of the round.
2. The pool $\mathcal{C}_r$ is formed by sampling from $\mathcal{C}_{r-1}$. As shown in the proof of Theorem 1, the probability that an instance $\boldsymbol{x} \in \mathcal{C}_{r-1}$ survives to $\mathcal{C}_r$ is

$$P_r(\boldsymbol{x}) = \Pr(\boldsymbol{x} \in \mathcal{C}_r \mid \boldsymbol{x} \in \mathcal{C}_{r-1}).$$

3. The expected value of the new pool $\mathcal{C}_r$ is the weighted average of the values from the previous pool, where the weights are these survival probabilities[1]:

$$\mathbb{E}[V \mid \mathcal{C}_r] = \frac{\sum_{\boldsymbol{x} \in \mathcal{C}_{r-1}} V(\boldsymbol{x}) P_r(\boldsymbol{x})}{\sum_{\boldsymbol{x} \in \mathcal{C}_{r-1}} P_r(\boldsymbol{x})}.$$

4. From the theorem's assumption, an instance's value $V(\boldsymbol{x})$ is positively correlated with its selection probability $p_{m,r}(\boldsymbol{x})$ for all strategies:

$$V(\boldsymbol{x}) > V(\boldsymbol{y}) \implies p_{m,r}(\boldsymbol{x}) \geq p_{m,r}(\boldsymbol{y}) \quad \forall m, r.$$

5. From this and the fact that $P_r(\boldsymbol{x})$ is a monotonically increasing function of each $p_{m,r}(\boldsymbol{x})$, it follows that $V(\boldsymbol{x})$ is *also positively correlated* with $P_r(\boldsymbol{x})$:

$$V(\boldsymbol{x}) > V(\boldsymbol{y}) \implies P_r(\boldsymbol{x}) \geq P_r(\boldsymbol{y}).$$

[1] Since we iterate over the specific instances fixed in the previous pool $\mathcal{C}_{r-1}$, the value $V(\boldsymbol{x})$ is treated as a constant.

6. Because the values $V(\boldsymbol{x})$ and the weights $P_r(\boldsymbol{x})$ are positively correlated, *Chebyshev's sum inequality* shows that the weighted average must be greater than or equal to the unweighted average:

$$\underbrace{\frac{\sum_{\boldsymbol{x}\in\mathcal{C}_{r-1}} V(\boldsymbol{x})P_r(\boldsymbol{x})}{\sum_{\boldsymbol{x}\in\mathcal{C}_{r-1}} P_r(\boldsymbol{x})}}_{\mathbb{E}[V|\mathcal{C}_r]} \geq \underbrace{\frac{\sum_{\boldsymbol{x}\in\mathcal{C}_{r-1}} V(\boldsymbol{x})}{|\mathcal{C}_{r-1}|}}_{\mathbb{E}[V|\mathcal{C}_{r-1}]}$$

7. Since $\mathbb{E}[V \mid \mathcal{C}_r] \geq \mathbb{E}[V \mid \mathcal{C}_{r-1}]$ holds for any round r, applying this result iteratively proves:

$$\mathbb{E}[V \mid \mathcal{C}_R] \geq \mathbb{E}[V \mid \mathcal{C}_{R-1}] \geq \cdots \geq \mathbb{E}[V \mid \mathcal{C}_0]. \quad \blacksquare$$

B. Use Case: Audio Spectrogram Classification

This use case demonstrates the practical utility of progressive filtering as a preprocessing step in AL. Specifically, we demonstrate that practitioners can achieve substantial accuracy gains, even transforming a previously underperforming strategy into a viable one. They can accomplish this without discarding their existing, trusted AL workflows by employing progressive filtering as a workflow-agnostic method to refine the unlabeled pool. This approach enhances any AL workflow with minimal engineering effort and risk. An additional benefit can also be computational feasibility. By reducing the unlabeled pool size, progressive filtering enables the practical application of complex or computationally expensive AL strategies that would otherwise be prohibitively costly on large (unfiltered) unlabeled pools.

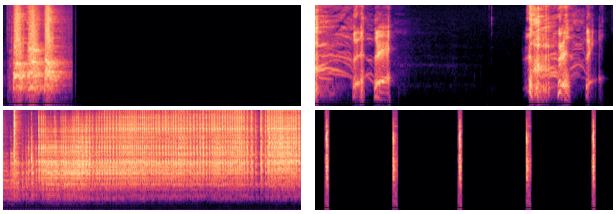

Figure 7. Example spectrograms from ESC50.

To illustrate this, we conducted a use case study on the ESC50 audio dataset [28]. As ESC50 does not have a dedicated test split, we randomly sampled 20% of the data for evaluation. We transformed audio segments into spectrograms following [31] (*cf*. Fig. 7) and employed the EAT base backbone to extract features [7]. Progressive filtering was applied exactly as described in the main paper, using the same hyperparameters and ensemble $\mathcal{S}$. This use case simulates a practitioner whose trusted AL strategy is Coreset [33], which is highly cited and well-known. Critically, Coreset is not part of the ensemble $\mathcal{S}$ used for filtering, which is a scenario where progressive filtering enhances an external strategy.

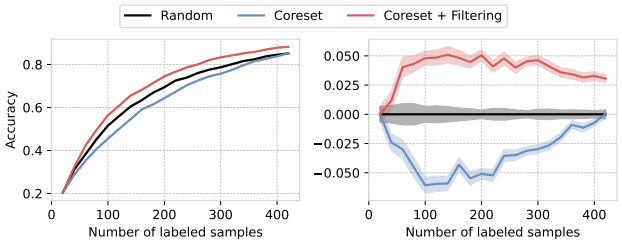

Figure 8. Absolute and relative learning curves comparing random sampling on $\mathcal{U}_t$, Coreset on $\mathcal{U}_t$, Coreset on $\mathcal{C}_R$ on ESC50.

As shown in Figure 8, applying the original Coreset strategy to the entire unlabeled pool $\mathcal{U}_t$ yields accuracy considerably below the random sampling baseline. In this instance, the standard strategy failed, meaning the practitioner would have obtained better results using simple random sampling. In contrast, by applying progressive filtering as a preprocessing step, we observe an accuracy improvement of up to 10%. This demonstrates that filtering can even effectively "fix" a failing strategy, allowing it to outperform baselines (*e.g.*, random) without requiring practitioners to abandon their preferred workflow.

C. Additional Ablations

Filtering as Pool Refinement. In Fig. 5a, we reported improvements in AULC [%], which showed that all strategies applied to the progressively filtered pool $\mathcal{C}_R$ lead to performance improvement. For these strategies, Fig. 9 presents the corresponding accuracy learning curves. The figure compares strategies applied to the filtered pool versus the unfiltered pool.

Figure 9 demonstrates that progressive filtering acts as a powerful preprocessing step, consistently improving the accuracies of individual AL strategies. This behavior is consistent across both DINOv2 and DINOv3 backbones. While the impact is most substantial for poorly performing AL strategies such as AlfaMix, progressive filtering still enhances strong performers such as UHerding, even if the improvement is less pronounced. Moreover, filtering significantly improves even random sampling, demonstrating its ability to clean uninformative instances from the unlabeled pool. Furthermore, progressive filtering can act as a crucial stabilizer for failing AL strategies. For example, while standard BAIT performs worse than random sampling on DINOv3, applying filtering fixes this issue and substantially improves the strategy's effectiveness.

Weak Ensemble Members. We investigate a potential failure case in REFINE by considering an ensemble where the majority of strategies perform worse than random sampling. Specifically, we run REFINE on Dopanim (CLIP) using an ensemble of three weak and one strong strategy.

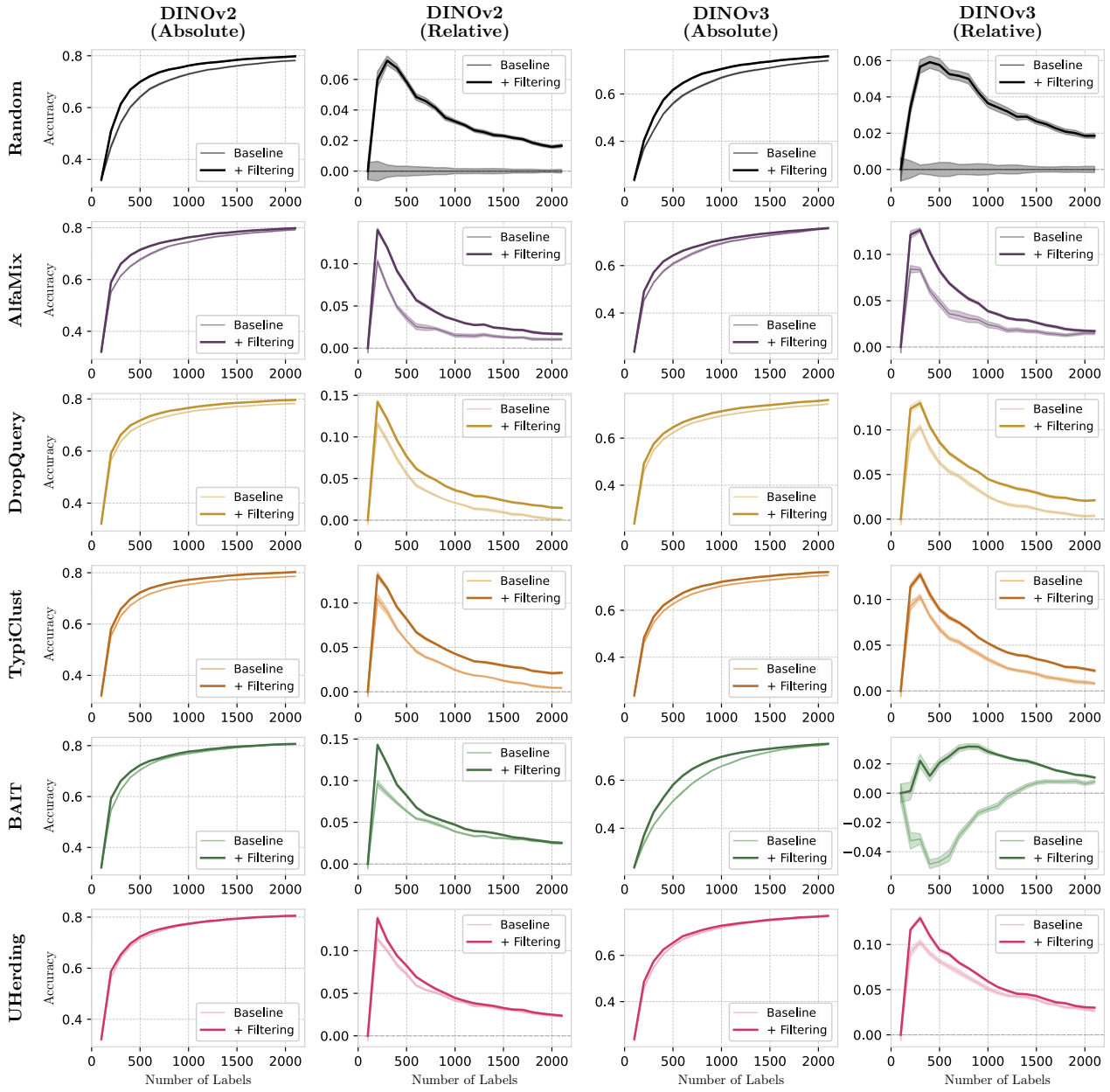

Figure 9. Relative and absolute learning curves on CIFAR-100 comparing selection strategies applied to the filtered pool (bold lines) versus the unfiltered pool (normal lines) using DINOv2 and DINOv3 backbones.

All ensemble members and REFINE are shown in Fig. 10 (left). Despite this unfavorable composition, REFINE performs well, closely following the strong ensemble member. A promising future direction may be to measure the similarity of ensemble selections and reduce the influence of non-complementary strategies over the AL cycles. This would encourage more independent selection within the ensemble, increasing the likelihood of retaining valuable instances.

Varying Batch Size. We investigate REFINE's behavior when changing the batch size b. Hence, we evaluate REFINE on CIFAR-100 (DINOv2) with batch sizes $b \in \{10, 25, 50, 100\}$ under a fixed total budget of $B \approx 2500$. Interestingly, results in Fig. 10 (right) show that larger batch sizes yield slightly better performance. We attribute this to the filtering process benefiting from a larger candidate set: each strategy contributes more instances, increasing the diversity of the filtered pool. Over multiple rounds, this effectively exposes the filtering process to a larger portion of the unlabeled pool, making it more likely that valuable instances are preserved.

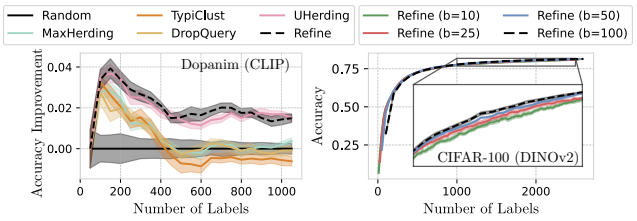

Figure 10. (Left) Relative learning curves of REFINE and all ensemble members on Dopanim. (Right) Absolute learning curves of REFINE on CIFAR-100 for different batch sizes b.

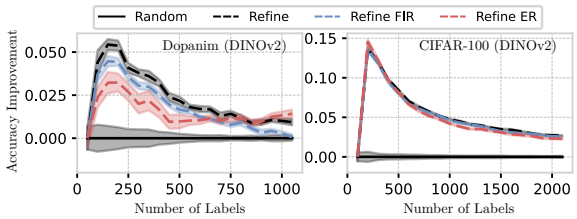

Figure 11. Relative learning curves of REFINE on Dopanim and CIFAR-100 (DINOv2) using coverage-based, FIR-based, and EER-based batch selection in the second stage.

Alternative for second stage. While coverage-based selection in the second stage of REFINE performs well empirically, it is solely a single strategy applied to the refined pool $\mathcal{C}_R$ without focusing on the most valuable heuristic in the current cycle. To provide an outlook on how we can realize this focus, we additionally consider a more deliberate selection process. Rather than applying a single coverage-based strategy to $\mathcal{C}_R$, we use each strategy in the ensemble independently. With an ensemble of two strategies, this results in two proposed batches, each reflecting a different heuristic. The selection process then reduces to assessing which of the proposed batches is expected to yield the greatest improvement in model performance.

To rank the candidate batches, we investigate two established criteria from the literature: the Fisher information ratio (FIR) [36] and the expected error (ER) [32]. Both criteria estimate the impact on model performance of including a batch in the training dataset. Notably, these criteria cannot be straightforwardly applied in a deep batch AL setting [19, 20], as the search for a candidate batch is combinatorially infeasible given a large unlabeled pool. Letting each strategy in the ensemble construct a candidate batch effectively guides the search through this otherwise intractable space [21]. The results in Fig. 11 demonstrate that this selection process yields comparable results to coverage-based selection, highlighting that the second stage of REFINE can easily adapted with more sophisticated components. We consider this a promising direction for future work, where such an selection mechanism could, for example, incorporate cost-sensitive criteria or realize a truly adaptive selection. For further details, we refer to our implementation.

D. Additional Results and Experiments

Parallel Acquisition Times. To provide insights into the acquisition times of REFINE and demonstrate how they can be reduced through parallelization, we report runtimes (min:sec) on CIFAR-100 (DINOv2) in Tab. 3. Sequential acquisition times remain on the same order as training times, and at later cycles the latter actually dominates. Since the selections of individual ensemble members are independent, parallelization is straightforward. We compute projected parallel runtimes using Amdahl's Law, assuming a conservatively estimated 85% of the acquisition process is parallelizable, with a realistic scheduling overhead of 3% for 8 and 8% for 80 processes. With 8 processes, acquisition drops below training time, making it no longer the computational bottleneck. For reference, AutoAL [37], another computationally intensive ensemble AL method, requires 4:24 and 8:41 at cycles 10 and 20, respectively, on the same hardware. Unlike REFINE, AutoAL cannot be parallelized as easily, as an auxiliary model is updated after all strategies have produced their selections, limiting scalability.

Table 3. Acquisition times (min:sec) of REFINE on CIFAR-100 (DINOv2) at AL cycles 10 and 20.

Cycle	Training	Seq. Acq.	Par. Acq. (8×)	Par. Acq. (80×)
10	07:50	09:58	02:38	01:44
20	14:34	10:31	02:47	01:50

Aggressiveness of Progressive Filtering. To provide insights into how aggressively progressive filtering reduces the size of the candidate pools $\mathcal{C}_r$, we report the ratio $|\mathcal{C}_r|/|\mathcal{U}|$ across filtering rounds r for several datasets in Tab. 4. For datasets with small batch sizes such as CIFAR-10 ($b = 10$), filtering is highly aggressive: after five rounds, only approximately 5% of unlabeled instances remain in $\mathcal{C}_R$, meaning over 95% are discarded before final acquisition. For datasets with larger batch sizes such as CIFAR-100 and Tiny ImageNet, the reduction is notably less pronounced. This also connects to the batch size analysis in Fig. 10 (right), where larger batch sizes benefit REFINE by considering a larger portion of the unlabeled pool.

Table 4. Ratio $|\mathcal{C}_R|/|\mathcal{U}|$ across filtering rounds on several datasets.

Filtering Round	1	2	3	4	5
CIFAR-10 ($b = 10$)	0.163	0.087	0.067	0.058	0.052
Dopanim ($b = 50$)	0.453	0.293	0.232	0.199	0.179
CIFAR-100 ($b = 100$)	0.689	0.415	0.323	0.277	0.247
Tiny ImageNet ($b = 200$)	1.000	0.711	0.586	0.516	0.472

Robustness to Class Imbalance & Distribution Shift. To further investigate the robustness of REFINE in more re-

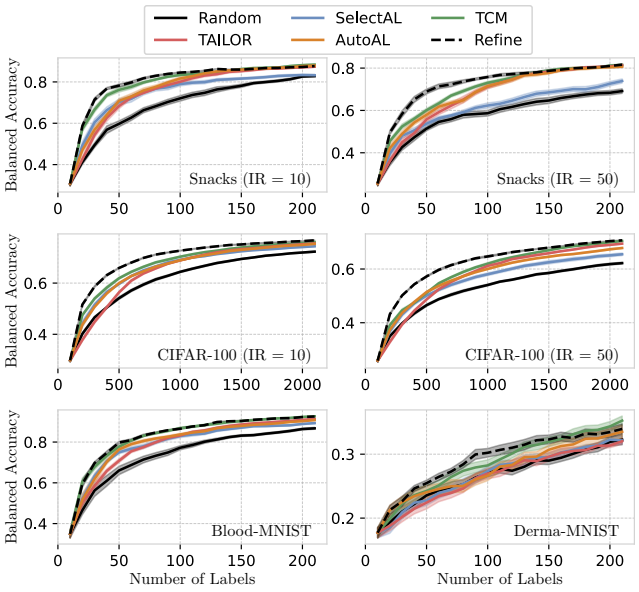

Figure 12. Absolute learning curves of ensemble AL methods on long-tailed and out-of-distribution datasets using DINOv2.

alistic settings, we conduct additional experiments considering both class imbalance and out-of-distribution data, *i.e.*, data that strongly differs from the foundation model's training distributions. For the class-imbalance setting, we run REFINE (DINOv2) on long-tailed versions of Snacks and CIFAR-100 using exponential class decay with imbalance ratios IR $\in \{10, 50\}$, denoting the ratio of the largest to the smallest class. For the out-of-distribution setting, we include BloodMNIST and DermaMNIST from the MedMNIST benchmark [39]. The results in Fig. 12 show that REFINE maintains strong performance, demonstrating robustness across both settings.

Fine-tuning Experiments. While the main experiments employ frozen features, reflecting current practice [13, 19, 20], REFINE is agnostic to the downstream model. To verify this, we investigate the performance of ensemble AL methods when fine-tuning the last two transformer blocks of a DINOv2 pretrained transformer. This setting is more challenging, as training hyperparameters, *i.e.*, learning rate and weight decay, had to be tuned on CIFAR-10 and transferred to the other datasets. Since the labeled pool grows in each AL cycle, these hyperparameters are highly likely to become suboptimal over the course of the AL process [18, 25]. The relative learning curves in Fig. 13 show that REFINE maintains strong performance.

E. Main Results: All Learning Curves

This section presents the complete set of learning curves used to derive Figs. 2 and 4, which were omitted from the main paper due to space limitations. For individual AL

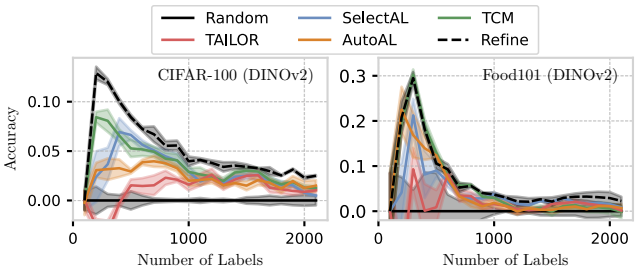

Figure 13. Relative learning curves of ensemble AL methods when fine-tuning the last two transformer blocks of DINOv2.

strategies, Fig. 14 displays the absolute learning curves (accuracy), while Fig. 15 shows the relative learning curves (accuracy relative to random sampling from $\mathcal{U}_t$). Similarly, for ensemble AL methods, Figs. 16 and 17 present the absolute and relative learning curves, respectively. These results provide a comprehensive overview supporting the conclusions drawn in the main text. Additionally, we include BoSS [21], an oracle strategy that approximates the optimal selection strategy by leveraging label information unavailable to standard AL strategies. By comparing against this oracle, we can assess how closely the AL strategies and REFINE approach the theoretical optimum.

Figures 14 and 15 demonstrate that REFINE outperforms all other individual AL strategies across nearly every dataset and backbone, including DINOv2, CLIP, and DINOv3. It excels in both low- and high-budget scenarios, consistently delivering high accuracies throughout the entire AL process. Furthermore, REFINE proves to be robust, maintaining a strong lead across diverse datasets while remaining invariant to the choice of the backbone. Similarly, Figures 16 and 17 demonstrate that REFINE outperforms all ensemble AL methods. These approaches generally struggle to effectively combine multiple strategies, often failing to exceed the performance of individual AL strategies. With REFINE, we propose the first ensemble AL method capable of effectively combining multiple AL strategies into a unified framework that consistently surpasses individual baselines. This improvement over existing ensemble AL methods is statistically significant ($\alpha = 0.01$), as confirmed by a Friedman test [10] on the AULC across 18 blocks (dataset $\times$ model) followed by post-hoc paired Wilcoxon signed-rank tests [38] with Holm correction [16].

When considering the optimal strategy approximated by the BoSS oracle, we observe that most strategies exhibit a considerable gap to this optimum. Notably, REFINE closes this gap more effectively than other strategies, yet a significant gap to the oracle remains, underscoring the need for continued research into more effective AL selection strategies.

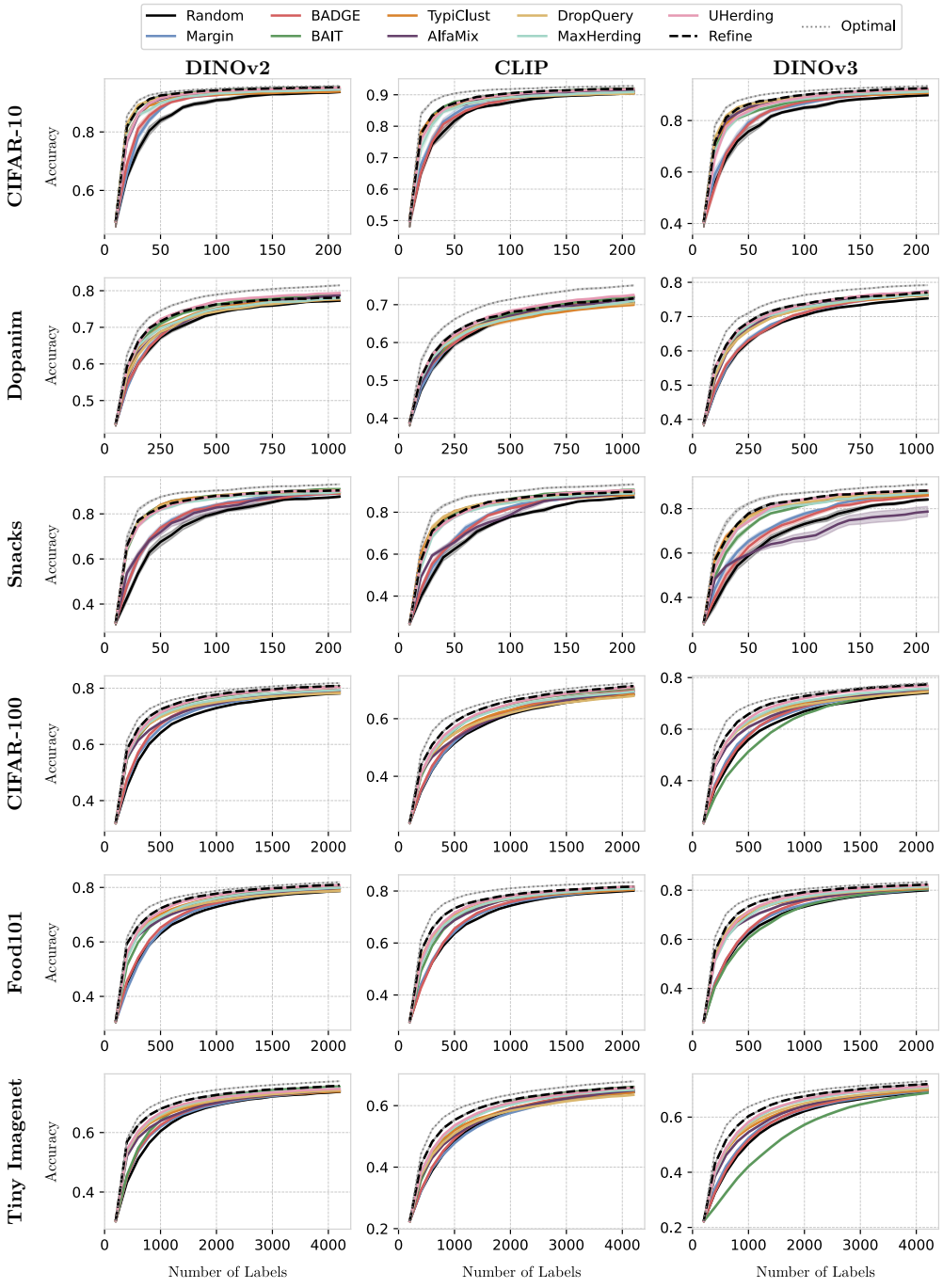

Figure 14. Complete set of absolute learning curves for individual AL strategies showing the accuracy for all datasets and backbone combinations. The optimal performance is approximated using BoSS [21].

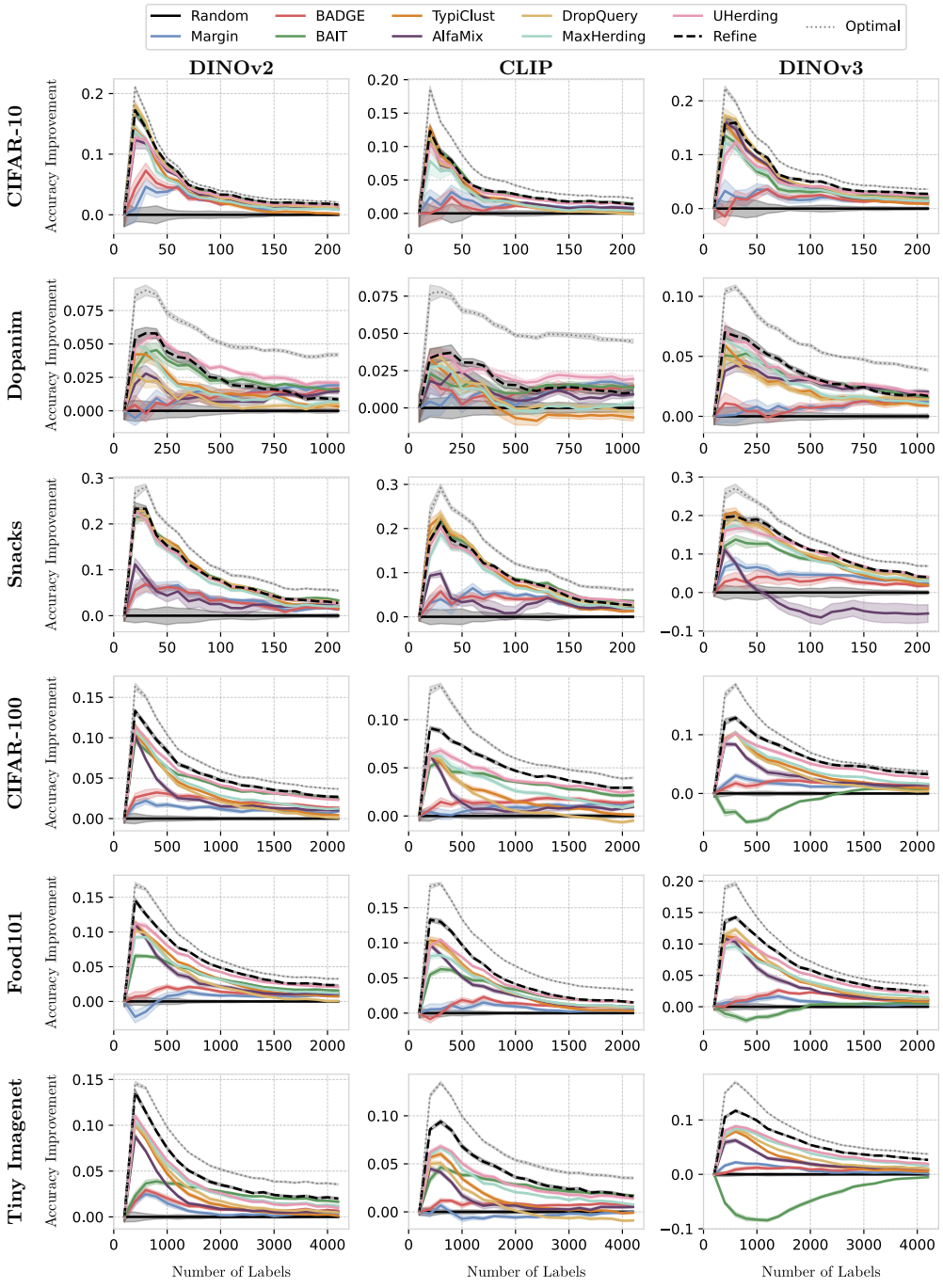

Figure 15. Complete set of relative learning curves for individual AL strategies showing the accuracy relative to random sampling from $\mathcal{U}_t$ for all datasets and backbone combinations. The optimal performance is approximated using BoSS [21].

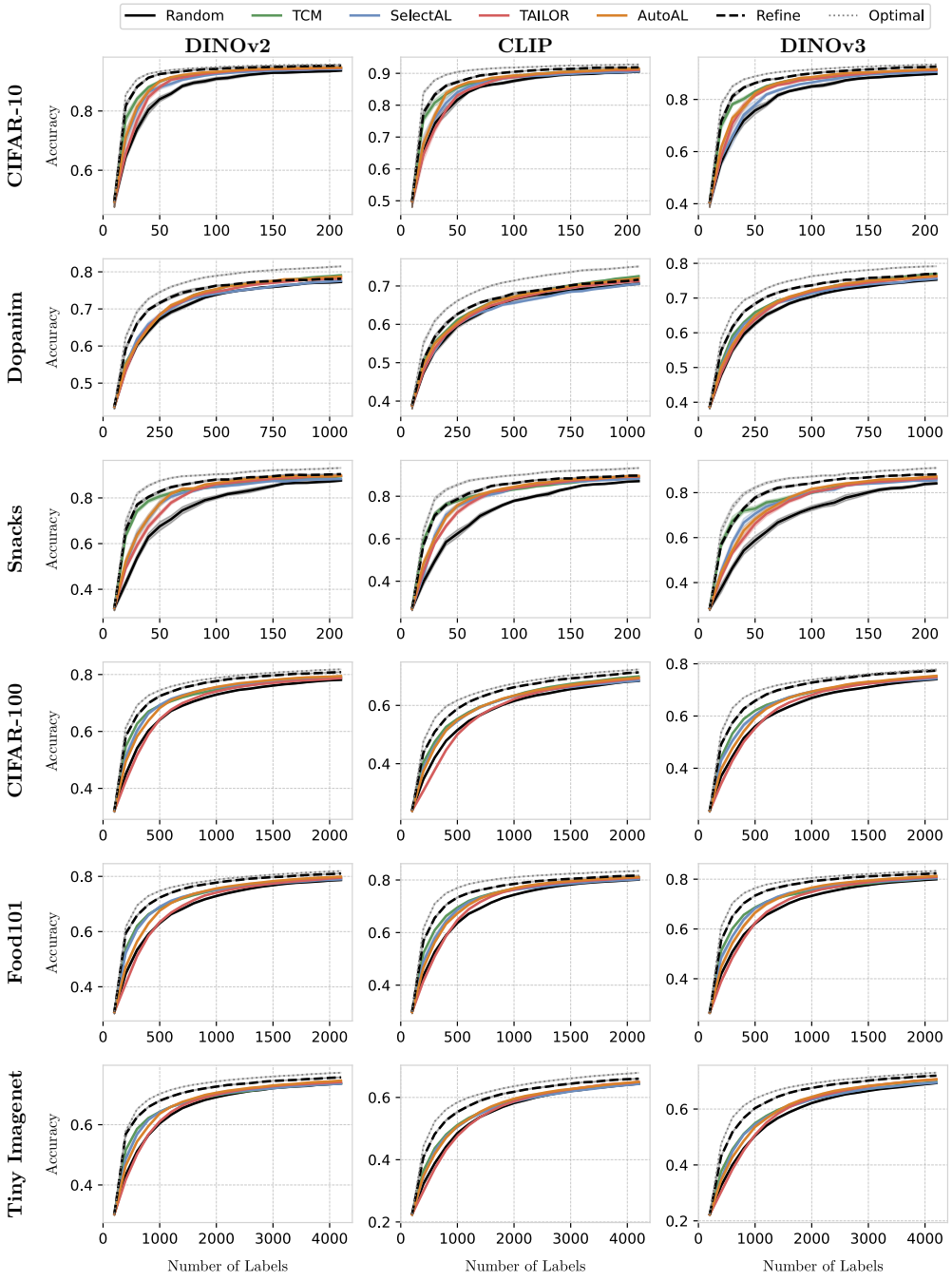

Figure 16. Complete set of absolute learning curves for ensemble AL methods showing the accuracy for all datasets and backbone combinations. The optimal performance is approximated using BoSS [21].

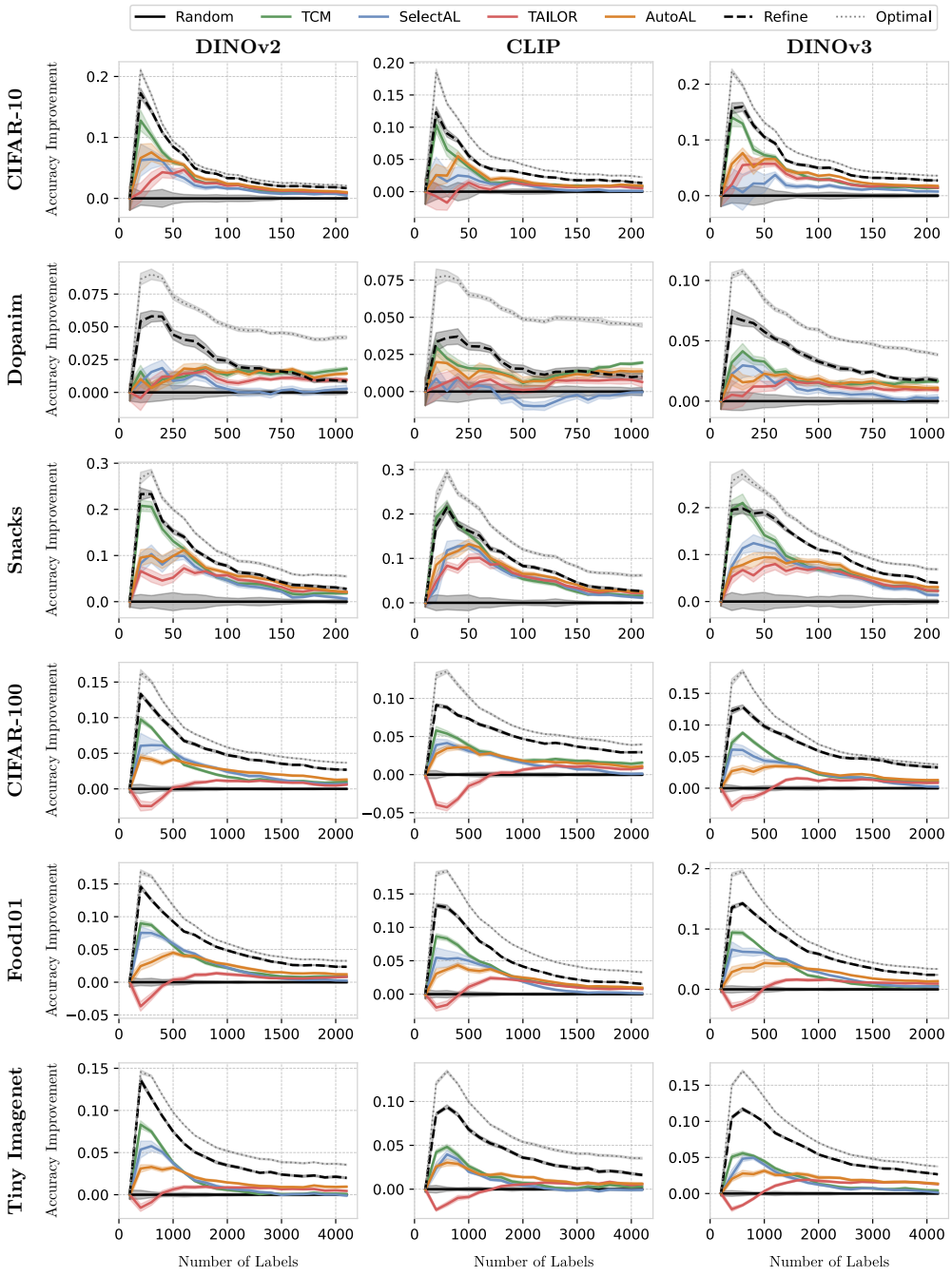

Figure 17. Complete set of relative learning curves for ensemble AL methods showing the accuracy relative to random sampling from $\mathcal{U}_t$ for all datasets and backbone combinations. The optimal performance is approximated using BoSS [21].