# REPICL: REUSABLE IN-CONTEXT PREDICTION ACROSS HETEROGENEOUS REPRESENTATION SPACES

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

## ABSTRACT

Frozen representations are widely reused for downstream classification, yet each new task typically requires fitting a new predictor. We ask whether the few-shot prediction procedure itself can instead be learned once and reused across datasets and representation spaces. To study this question, we introduce RepShiftBench, comprising 1,218 encoder–dataset tasks across text, image, and audio, with separate evaluation of generalization to unseen datasets, unseen encoders, jointly unseen datasets and encoders, and unseen modalities. The benchmark exposes a substantial gap: Logistic Regression fitted independently on each episode outperforms every evaluated in-context learner across all settings. We introduce RepICL, a meta-trained in-context learner that canonicalizes each episode through episodic whitening before prediction. Its inductive variant, RepICL-I, surpasses Logistic Regression in all 12 benchmark settings, while RepICL-T substantially outperforms existing transductive methods. Ablations identify episodic whitening as the primary source of these gains, while showing that it is not a universally beneficial preprocessing step. Across both variants, the gains concentrate on queries for which simple support prototypes favor the wrong class or provide little separation between the true class and competing classes. Transduction provides its largest additional gains when limited support coverage gives a misleading view of class separation. Together, these results demonstrate that a shared few-shot prediction procedure can generalize beyond the representation spaces observed during training.

## 1 INTRODUCTION

Pretrained encoders have made representation reuse a standard paradigm across modern machine learning (Devlin et al., 2019; Radford et al., 2021; Oquab et al., 2024). A frozen encoder can support many downstream datasets, with only a lightweight classifier fitted on top of its representations. This *embed-then-fit* paradigm is simple and remarkably effective: strong frozen representations often support competitive downstream performance even with basic classifiers (Chen et al., 2019; Kornblith et al., 2019; Tian et al., 2020; El Assadi et al., 2026b). Yet this reuse largely stops at the representation: the downstream prediction rule is still fitted separately for each dataset and refitted when the encoder changes (Kornblith et al., 2019; Tian et al., 2020).

In-context and model-based meta-learning (Santoro et al., 2016; Hospedales et al., 2022) suggest a more ambitious possibility: rather than repeatedly fitting a task-specific classifier, can we learn the *prediction procedure itself* once and reuse it across tasks? Recent in-context learners instantiate this idea over vector representations and can transfer across datasets and tasks (Kirsch et al., 2022; Fifty et al., 2024; Woerner et al., 2026). However, existing evaluations either keep the encoder fixed or study cross-encoder and cross-modality transfer only in limited settings (Woerner et al., 2026), leaving encoder identity largely unexplored as an independent axis of generalization. Varying the dataset changes the downstream prediction problem, whereas varying the encoder changes the representation space itself, as different encoders can induce spaces with different dimensionalities, scales, covariance structures, and geometries. This raises a distinct question: ***can a learned prediction procedure still transfer to an unseen representation space?***

To study this question directly, we introduce **RepShiftBench**, a benchmark for few-shot classification across heterogeneous frozen representations. Each task is defined by an *encoder–dataset*



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.

*pair*, allowing the downstream dataset and representation space to vary independently, from which we sample few-shot episodes consisting of support and query examples. Across 59 datasets and 62 pretrained encoders spanning text, image, and audio, RepShiftBench comprises 1,218 tasks and evaluates generalization to unseen datasets, unseen encoders, jointly unseen datasets and encoders, and entirely unseen modalities. This systematic evaluation exposes a substantial gap between reusable prediction procedures and per-episode fitting: ***Logistic Regression fitted independently on each episode outperforms every evaluated in-context learner across all benchmark settings***. The challenge is therefore not simply few-shot prediction, but learning a shared predictor that remains reliable as the representation space changes.

We address this gap with **RepICL**, a meta-trained in-context learner that canonicalizes each episode's representations through *episodic whitening*—episode-wise PCA followed by standardization—before in-context prediction. We study an inductive variant, RepICL-I, which predicts each query from the labeled support set alone, and a transductive variant, RepICL-T, which additionally exploits the unlabeled query set. RepICL-I surpasses per-episode Logistic Regression in all 12 benchmark settings while consistently outperforming existing in-context learners, and RepICL-T achieves substantially larger gains over existing transductive methods. Together, these results demonstrate that ***a single learned prediction procedure can generalize beyond the representation spaces observed during training, including to entirely unseen modalities.***

Our analyses further clarify what enables this transfer and where its gains arise. Ablations identify episodic whitening as the primary source of improvement, yet applying the same transformation to existing learners does not improve their performance, indicating that its benefit comes from meta-training the predictor on whitened episodes rather than from better preprocessing alone. Across both variants, the gains concentrate on queries for which simple support prototypes either favor the wrong class or provide little separation between the true class and competing classes. Among these difficult queries, transduction provides its largest additional gains when the few-shot support set is an unrepresentative sample of its classes, suggesting that unlabeled queries are especially useful when support coverage is limited.

## 2 Related Work

**Few-shot prediction over pretrained representations.** Few-shot classification has been studied through metric-based, optimization-based, and model-based approaches (Vinyals et al., 2016; Snell et al., 2017; Finn et al., 2017; Santoro et al., 2016; Hospedales et al., 2022). With strong pretrained representations, however, simple non-meta-learned rules can be highly competitive: nearest-centroid methods provide strong baselines (Wang et al., 2019), while linear classifiers fitted on frozen support representations can rival more elaborate meta-learning methods (Tian et al., 2020). Transductive methods additionally exploit unlabeled queries at inference (Liu et al., 2019; Boudiaf et al., 2020; Ziko et al., 2020; Hu et al., 2021), while cross-domain few-shot learning studies transfer across data distributions (Tseng et al., 2020; Li et al., 2022). We instead ask whether a single reusable predictor can retain the strength of per-task fitting as the underlying representation space changes.

**In-context prediction across representation spaces.** Model-based meta-learners can learn reusable prediction procedures over labeled examples and queries without test-time parameter updates (Garnelo et al., 2018; Kirsch et al., 2022; Fifty et al., 2024). Prior-fitted networks and their tabular extensions similarly learn predictors from distributions of synthetic tasks (Müller et al., 2022; Hollmann et al., 2023; 2025; Qu et al., 2025). Closest to our setting, TAIL targets heterogeneous feature spaces and demonstrates transfer to unseen modalities (Woerner et al., 2026). However, its cross-modal evaluation uses only one dataset–encoder pair for each unseen modality, and it does not systematically disentangle dataset and encoder novelty. Thus, while TAIL establishes the feasibility of transfer across feature spaces and modalities, it remains unclear how a shared predictor behaves across broad combinations of unseen datasets and pretrained encoders. RepShiftBench is designed to study this question systematically.

Methods also differ in how they reduce dependence on feature coordinates. GPICL meta-trains over randomly projected tasks (Kirsch et al., 2022), while TAIL maps encoder outputs through randomized projections into a common latent space (Woerner et al., 2026). More broadly, prior work on representation normalization and alignment has normalized episode statis-

tics (Bronskill et al., 2020; Hollmann et al., 2023), whitened using broader reference statistics (Su et al., 2021), or aligned representations through shared anchors (Moschella et al., 2023). RepICL instead uses episode-wise PCA and standardization to canonicalize heterogeneous frozen representations for a shared prediction procedure.

**Evaluating representation-space generalization.** Few-shot benchmarks such as Meta-Dataset evaluate learners across diverse classes, datasets, and domains (Vinyals et al., 2016; Triantafillou et al., 2020; Ullah et al., 2022), while embedding benchmarks such as MTEB, MIEB, X-ARES, and MAEB evaluate pretrained representations across broad collections of downstream tasks (Muennighoff et al., 2023; Xiao et al., 2025b; Zhang et al., 2025a; El Assadi et al., 2026a). The former primarily vary the downstream task while evaluating few-shot learners, while the latter primarily vary the encoder while evaluating representation quality. Neither is designed to jointly vary dataset and encoder while evaluating a single shared few-shot predictor. RepShiftBench fills this gap by treating the encoder–dataset pair as the unit of evaluation and separately evaluating dataset, encoder, joint, and modality generalization.

## 3 Few-Shot Prediction Setup

**Few-shot episodes.** In this paper, we consider standard $N$-way $K$-shot classification episodes (Vinyals et al., 2016; Snell et al., 2017). Each episode contains $N$ classes, with $K$ labeled support examples and $Q$ query examples per class. Let $\mathcal{S} = \{(\mathbf{z}_i^s, y_i^s)\}_{i=1}^{NK}$ and $\mathcal{Q} = \{(\mathbf{z}_j^q, y_j^q)\}_{j=1}^{NQ}$ denote the support and query sets, respectively, where query labels are withheld from the predictor. Here, $\mathbf{z} \in \mathbb{R}^d$ denotes an input representation and $y \in \{1, \dots, N\}$ an episode-local class label, with labels remapped independently within each episode. The goal is to predict the labels of the query representations from the information available under the corresponding inference setting.

**Inductive and transductive inference.** Let $f_\theta$ denote a few-shot prediction procedure parameterized by $\theta$. Under inductive inference, each query is predicted independently from the support set and its own representation, $\hat{y}_j^q = f_\theta(\mathcal{S}, \mathbf{z}_j^q)$, where $\hat{y}_j^q$ is the predicted label for query $j$. Under transductive inference (Vapnik, 1998; Liu et al., 2019), the learner may additionally condition on the full unlabeled query set $\mathcal{U} = \{\mathbf{z}_j^q\}_{j=1}^{NQ}$ and jointly predict $\hat{\mathbf{y}}^q = f_\theta(\mathcal{S}, \mathcal{U})$, where $\hat{\mathbf{y}}^q$ collects the predictions for all queries. Because transductive methods access query-set information unavailable to inductive methods, we evaluate the two inference settings separately.

## 4 RepShiftBench

The episode formulation in Section 3 treats representations as given, without specifying how their representation space varies across tasks. To study whether a single prediction procedure can generalize when that space itself changes, RepShiftBench treats the pretrained encoder as an explicit task axis alongside the downstream dataset. Figure 1 summarizes the resulting benchmark construction and evaluation protocol.

### 4.1 Encoder–Dataset Task Formulation

**Encoder–dataset tasks.** Let $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^n$ denote a labeled dataset and $E$ a frozen pretrained encoder with embedding function $\phi_E : \mathcal{X} \to \mathbb{R}^{d_E}$. We define a task as a compatible encoder–dataset pair $(E, \mathcal{D})$. For each episode sampled from this task, the representations in Section 3 are obtained as $\mathbf{z} = \phi_E(x) \in \mathbb{R}^{d_E}$. Thus, $\mathcal{D}$ determines the downstream examples and label space, whereas $E$ determines the representation space in which prediction is performed. Methods receive only these frozen representations and the support labels, rather than the raw inputs.

**Generalization settings.** Let $\mathcal{T}_{\text{train}}$ denote the encoder–dataset tasks used for meta-training. We evaluate four forms of generalization according to which components of $(E, \mathcal{D})$ are unseen at test time: **Dataset Adaptation**, where $\mathcal{D}$ is unseen but $E$ has appeared during meta-training; **Encoder Adaptation**, where $E$ is unseen but $\mathcal{D}$ is seen; **Joint Adaptation**, where both $E$ and $\mathcal{D}$ are unseen; and **Modality Adaptation**, where both are unseen and belong to a modality absent entirely

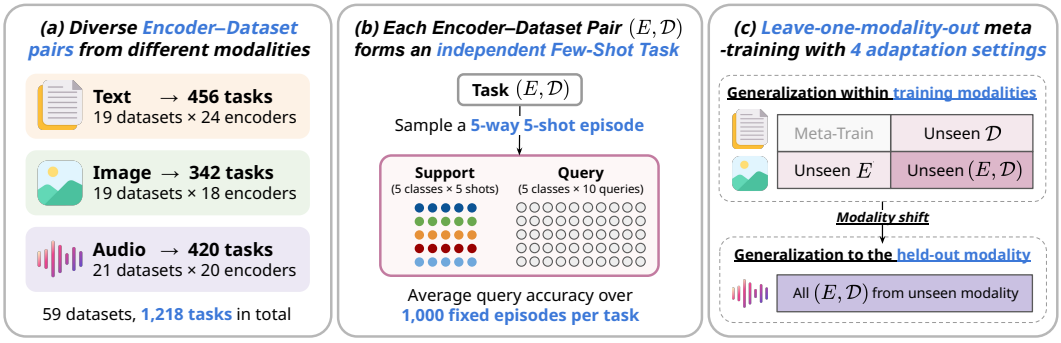



Figure 1: RepShiftBench: encoder–dataset tasks, episodic evaluation, and adaptation settings.

from meta-training. Together, these settings disentangle changes in the downstream problem from changes in the representation space and extend the evaluation to unseen modalities.

### 4.2 Benchmark Construction

**Datasets and encoders.** RepShiftBench spans text, image, and audio, drawing classification datasets from MTEB for text (Muennighoff et al., 2023), MIEB for images (Xiao et al., 2025b), and X-ARES and MAEB for audio (Zhang et al., 2025a; El Assadi et al., 2026a). We retain datasets with at least five classes and pair each dataset with every compatible encoder of the same modality. The encoder pool spans diverse architectures, training objectives, model sizes, and embedding dimensions, providing broad variation in the representation spaces encountered by the predictor. In total, RepShiftBench contains 1,218 compatible encoder–dataset tasks formed from 59 datasets and 62 pretrained encoders. Complete dataset and encoder inventories and representation extraction details are provided in Appendices A.1 and A.2.

**Folds and task splits.** We construct three folds, each holding out one modality entirely from meta-training. Within the remaining two modalities, datasets and encoders are partitioned separately into seen and held-out sets. Pairs in which both are seen form the meta-training pool, while combinations involving held-out datasets or encoders instantiate the Dataset, Encoder, and Joint Adaptation settings defined above. The entirely held-out modality defines Modality Adaptation. Notably, held-out datasets span both unfamiliar domains and familiar task types, while held-out encoders are selected from more recently released models, testing generalization to new datasets and domains and transfer to newer encoders. Full task assignments and split details are provided in Appendices A.3 and A.4.

### 4.3 Evaluation Protocol

**Episode sampling and scoring.** Each encoder–dataset task is evaluated on the same 1,000 fixed, class-balanced 5-way 5-shot episodes for all methods, with 10 queries per class. For each evaluation task, supports and queries are drawn according to the source dataset's predefined train–test split whenever available, with supports from the training split and queries from the test split. Exceptions and split-construction details are provided in Appendix A.4. For each task, we average query accuracy across its 1,000 episodes. An adaptation-setting score is computed as the unweighted mean over its encoder–dataset tasks. When reporting an aggregate score for a modality-held-out fold, we average the four adaptation-setting scores equally, regardless of the number of tasks in each setting.

**Reference methods.** Per-episode methods such as Logistic Regression and ProtoHead are fitted or constructed using only the support set of the current episode and have no meta-training history. Accordingly, notions such as *seen encoder* or *seen dataset* describe the meta-training history of learned predictors; for per-episode methods, the same adaptation labels simply identify the corresponding evaluation subsets. All methods are nevertheless evaluated on exactly the same tasks and episodes: per-episode methods quantify what can be achieved from the current support set alone, whereas meta-trained methods test whether knowledge learned across previous tasks and representation spaces provides an additional benefit.

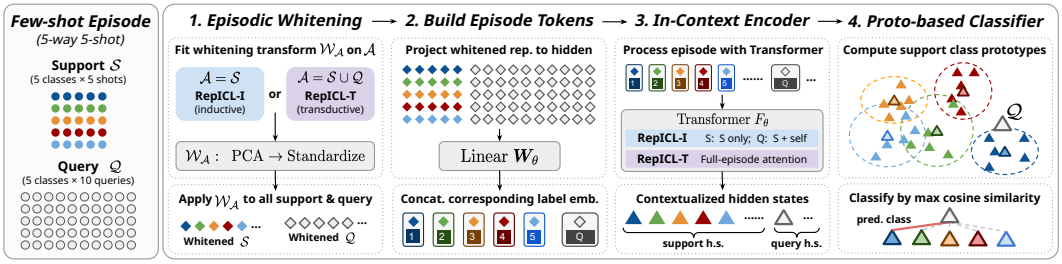



Figure 2: RepICL: episodic whitening enables shared in-context prediction across heterogeneous representation spaces. RepICL-I estimates whitening from the support set and uses inductive attention, where supports attend only to supports and each query attends to supports and itself. RepICL-T instead uses unlabeled queries for whitening and permits full-episode attention.

# 5 REPICL

A shared predictor in RepShiftBench must operate across representations produced by different encoders, whose dimensionalities, scales, and covariance structures can vary substantially. RepICL addresses this heterogeneity by first canonicalizing each episode through *episodic whitening* and then applying a shared Transformer-based in-context predictor in the resulting space, without test-time parameter updates. We study two inference variants. RepICL-I is inductive: the labeled support set alone determines both the episode transformation and each query prediction. RepICL-T is transductive: it additionally uses the unlabeled query set when constructing the episode transformation and during in-context prediction. Figure 2 summarizes the overall procedure.

## 5.1 EPISODIC WHITENING

Rather than learning an encoder-specific alignment, RepICL constructs an episode-local representation through *episodic whitening*. For an episode, let $\mathcal{Z}_S = \{\mathbf{z}_i^s\}_{i=1}^{NK}$ and $\mathcal{Z}_Q = \{\mathbf{z}_j^q\}_{j=1}^{NQ}$ denote the support and query representations. We estimate the episode transformation from $\mathcal{A} = \mathcal{Z}_S$ for RepICL-I and $\mathcal{A} = \mathcal{Z}_S \cup \mathcal{Z}_Q$ for RepICL-T. We first perform PCA on $\mathcal{A}$ and then standardize the retained coordinates. Let $\boldsymbol{\mu}_\mathcal{A}$ be the mean of $\mathcal{A}$ and $\mathbf{W}_\mathcal{A} \in \mathbb{R}^{d_E \times r_\mathcal{A}}$ the retained principal directions, and $\mathbf{m}_\mathcal{A}$ and $\mathbf{v}_\mathcal{A}$ the coordinate-wise mean and variance of the PCA projections of $\mathcal{A}$. The episode-specific whitening transform is

$$\mathcal{W}_\mathcal{A}(\mathbf{z}) = \frac{\mathbf{W}_\mathcal{A}^\top(\mathbf{z} - \boldsymbol{\mu}_\mathcal{A}) - \mathbf{m}_\mathcal{A}}{\sqrt{\mathbf{v}_\mathcal{A} + \varepsilon}} \in \mathbb{R}^{r_\mathcal{A}}, \tag{1}$$

where division is elementwise and $\varepsilon > 0$ is a small constant for numerical stability. We set $r_\mathcal{A} = \min(d_E, |\mathcal{A}|-1, r_{\max})$, with $r_{\max} = 24$ for RepICL-I and 74 for RepICL-T, matching the maximum PCA ranks under the 5-way 5-shot meta-training episode configuration of RepShiftBench. We then apply $\mathcal{W}_\mathcal{A}$ to all support and query representations, yielding the whitened representations $\mathbf{u}_i^s$ and $\mathbf{u}_j^q$ for the shared predictor.

## 5.2 IN-CONTEXT PREDICTION

**Episode tokens.** Each whitened representation is projected and concatenated with a label embedding to form a Transformer token:

$$\mathbf{t}_i^s = [\,\mathbf{W}_\theta \mathbf{u}_i^s \,\|\, \mathbf{e}_{y_i^s}\,], \qquad \mathbf{t}_j^q = [\,\mathbf{W}_\theta \mathbf{u}_j^q \,\|\, \mathbf{e}_q\,]. \tag{2}$$

Here, $\mathbf{W}_\theta$ is a learned projection. Support labels use ELMES embeddings (Fifty et al., 2024), while $\mathbf{e}_q$ is a single learnable embedding shared by all queries.

**Contextualization.** The episode tokens are processed by a shared Transformer $F_\theta$ to obtain contextualized support and query representations $\{\mathbf{h}_i^s\}$ and $\{\mathbf{h}_j^q\}$. For RepICL-I, support tokens attend only to support tokens, while each query attends to all supports and itself but not to other queries. For RepICL-T, all support and query tokens can attend to one another, allowing information to propagate across the entire episode.

**Prototype-based classification.** The contextualized support representations define an episode-specific classifier. For each class $c$, let $\mathcal{I}_c = \{i : y_i^s = c\}$ denote its support indices. We form a class prototype by averaging the corresponding contextualized representations and classify each query by cosine similarity:

$$\mathbf{p}_c = \frac{1}{|\mathcal{I}_c|} \sum_{i \in \mathcal{I}_c} \mathbf{h}_i^s, \qquad \ell_{j,c} = \frac{\cos(\mathbf{h}_j^q, \mathbf{p}_c)}{\tau}, \qquad \hat{y}_j^q = \arg\max_c \ell_{j,c}, \tag{3}$$

where $\tau$ is a fixed temperature and $\hat{y}_j^q$ is the predicted label for query $j$. As in Prototypical Networks (Snell et al., 2017) and FEAT (Ye et al., 2020), the classifier is constructed from the current support set rather than tied to fixed class identities.

### 5.3 Meta-Training Across Representation Spaces

We meta-train a single $\theta$ across tasks on 5-way 5-shot episodes. We optimize it on episodes sampled from $\mathcal{T}_{\text{train}}$, which contains multiple encoder–dataset pairs, by minimizing query-set cross-entropy:

$$\mathcal{L}(\theta) = -\frac{1}{NQ} \sum_{j=1}^{NQ} \log \frac{\exp(\ell_{j,y_j^q})}{\sum_{c=1}^{N} \exp(\ell_{j,c})}. \tag{4}$$

During meta-training, tasks are sampled to prevent very large datasets from dominating optimization, while class labels are randomly permuted within each episode so that episode-local label embeddings cannot acquire fixed semantic meanings. By default, we initialize the Transformer and label embeddings from the released CAML checkpoint (Fifty et al., 2024), initialize the post-whitening projection from scratch, and replace CAML's fixed output head with the episode-specific prototype classifier described above. We select the checkpoint with the highest accuracy on the validation tasks. Full optimization details are provided in Appendix B.

## 6 Experiments

### 6.1 Experimental Setup

We evaluate RepICL on RepShiftBench against three groups of baselines: **(1) classical classifiers** fitted on each episode's support set, including Logistic Regression and an RBF-kernel SVM; **(2) few-shot methods**, including the inductive ProtoHead and SimpleShot (Wang et al., 2019), and the transductive LaplacianShot (Ziko et al., 2020), $\alpha$-TIM (Veilleux et al., 2021), and PT-MAP (Hu et al., 2021); and **(3) in-context learners**, including TAIL (Woerner et al., 2026), TabPFN (Hollmann et al., 2025), and TabICL (Qu et al., 2025). Methods trained on RepShiftBench use 5-way 5-shot episodes, and all methods are evaluated on the same fixed episodes. Hyperparameters and training details are provided in Appendix C.

### 6.2 Main Results

**Results on RepShiftBench.** Table 1 reports accuracy across three modality-held-out folds and four adaptation settings. We report inductive and transductive methods separately because only the latter access the unlabeled query set. The benchmark exposes a consistent gap between reusable predictors and per-episode fitting: Logistic Regression is the strongest inductive baseline in all 12 settings, while TabPFN and TabICL remain below it throughout. This does not imply universal dominance: conventional few-shot methods can outperform Logistic Regression on CNN representations, but this advantage does not persist across broader encoder families or other modalities (Appendix E.1).

Under inductive inference, RepICL-I surpasses Logistic Regression in all 12 settings by 0.30–1.26 points, with significant gains in 10 of them. Under transductive inference, RepICL-T leads in all 12 settings, exceeding the strongest transductive baseline by 2.65–6.66 points with significant gains throughout. Both variants have per-episode runtimes comparable to Logistic Regression (Appendix E.3). As an additional comparison, RepICL-I is comparable to LoRA on a subset of RepShiftBench without test-time updates, while RepICL-T further improves with unlabeled queries (Appendix E.2). Together, these results show that a shared prediction procedure can generalize across heterogeneous representation spaces and different shifts under both inference settings.

Table 1: RepShiftBench 5-way 5-shot accuracy (%) under dataset (D), encoder (E), joint (J), and modality (M) adaptation across three modality-held-out folds. Bold marks the best result within each block. TAIL$^{\dagger}$ denotes our reproduction of the released recipe with scale calibration (Appendix C.4). Stars indicate a significant improvement over the strongest baseline within the same block and column (two-sided paired test clustered by dataset; $^{*}p < 0.05$, $^{**}p < 0.01$, $^{***}p < 0.001$).

| **Adaptation Setting →** | **Text-held-out fold** | | | | **Audio-held-out fold** | | | | **Image-held-out fold** | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Method ↓** | D | E | J | M | D | E | J | M | D | E | J | M |
| ***Inductive*** | | | | | | | | | | | | |
| ProtoHead | 56.11 | 72.55 | 61.05 | 64.41 | 68.64 | 71.72 | 71.28 | 48.41 | 54.08 | 65.83 | 56.22 | 80.88 |
| SimpleShot | 56.14 | 73.06 | 61.65 | 64.90 | 68.92 | 72.22 | 72.01 | 49.04 | 54.26 | 66.64 | 57.01 | 81.12 |
| SVM (RBF kernel) | 55.52 | 72.68 | 62.32 | 64.62 | 67.03 | 72.13 | 71.81 | 50.49 | 55.00 | 66.70 | 57.81 | 76.64 |
| Logistic regression | 60.19 | 75.33 | 65.74 | 69.62 | 72.37 | 76.05 | 75.56 | 54.46 | 58.35 | 71.56 | 61.90 | 82.38 |
| TabPFN | 59.26 | 73.32 | 64.36 | 63.38 | 67.96 | 72.46 | 72.71 | 52.55 | 55.30 | 67.31 | 59.19 | 80.65 |
| TabICL | 59.08 | 74.04 | 63.90 | 67.29 | 70.69 | 74.17 | 73.66 | 52.72 | 56.81 | 69.02 | 59.92 | 81.90 |
| **RepICL-I (Ours)** | **61.41**$^{**}$ | **75.85**$^{**}$ | **66.22** | **70.10**$^{***}$ | **73.52**$^{**}$ | **76.79**$^{**}$ | **75.86** | **55.14**$^{***}$ | **59.19**$^{**}$ | **72.82**$^{***}$ | **62.56**$^{**}$ | **83.26**$^{***}$ |
| ***Transductive*** | | | | | | | | | | | | |
| LaplacianShot | 55.21 | 72.78 | 60.55 | 64.42 | 68.77 | 71.39 | 71.22 | 47.53 | 53.48 | 65.71 | 55.61 | 81.15 |
| $\alpha$-TIM | 59.04 | 74.31 | 64.62 | 66.39 | 70.27 | 73.77 | 73.99 | 52.62 | 56.31 | 68.85 | 59.90 | 82.24 |
| PT-MAP | 56.82 | 74.60 | 63.00 | 68.65 | 72.04 | 74.47 | 74.82 | 49.46 | 55.73 | 69.92 | 59.29 | 83.01 |
| TAIL$^{\dagger}$ | 52.79 | 71.96 | 61.05 | 62.23 | 66.06 | 71.13 | 71.07 | 45.67 | 50.33 | 64.93 | 55.55 | 79.31 |
| **RepICL-T (Ours)** | **63.55**$^{**}$ | **77.99**$^{***}$ | **68.29**$^{***}$ | **73.26**$^{***}$ | **75.27**$^{**}$ | **78.13**$^{***}$ | **77.47**$^{**}$ | **55.62**$^{***}$ | **61.79**$^{**}$ | **76.58**$^{***}$ | **65.02**$^{***}$ | **85.69**$^{***}$ |

**External validation under the TAIL protocol.** To test whether these gains depend on the design of RepShiftBench, we additionally evaluate RepICL-T under TAIL's original protocol of 11 visual datasets represented by CLIP features. We use the image-held-out checkpoint, meta-trained only on text and audio tasks, so all 11 visual datasets and the CLIP image encoder are unseen during meta-training. Despite this cross-modality transfer, RepICL-T achieves the highest mean accuracy, improving over our TAIL reproduction from 90.47% to 92.31%. Full per-dataset results and comparisons with CAML and GPICL are provided in Appendix D.1, and TAIL reproduction details in Appendix C.4.

### 6.3 Ablation Study

Table 2 ablates episodic whitening, model-structure choices, and meta-training diversity, with each variant retrained under the same budget across all three folds. Table 3 further tests whether episodic whitening benefits learners that were not trained on whitened episodes.

**Episodic whitening drives the gains.** Removing either component of episodic whitening causes the largest drops in Table 2. The two components show distinct patterns across adaptation settings: removing coordinate-wise standardization is particularly harmful under dataset adaptation, whereas removing PCA is more damaging when the test encoder is unseen (encoder, joint, and modality adaptation). Replacing PCA with either a fixed or trainable projection produces similar degradation, suggesting that its benefit comes from the episode-dependent PCA transformation rather than merely matching dimensionality or adding projection capacity. In contrast, CAML initialization has negligible effect, while replacing the prototype-based classifier with a fixed five-class head is nearly neutral for RepICL-I and slightly improves RepICL-T. These results identify episodic whitening, rather than initialization or output-head design, as the primary source of RepICL's gains.

**Meta-training diversity primarily benefits transductive inference.** Restricting meta-training to one dataset or one encoder per modality has only a modest effect on RepICL-I (0.1–0.9 points), but causes substantially larger drops for RepICL-T (1.5–2.4 points). Restricting encoder diversity is also generally more harmful than restricting dataset diversity, suggesting that exposure to varied representation spaces is particularly important for RepICL-T. Overall, RepICL-I is relatively insensitive to these restrictions, whereas RepICL-T benefits more from diversity across tasks and encoders.

**Whitening is not a drop-in preprocessing step.** To test whether episodic whitening simply produces better inputs, we apply the same transformation to learners not trained on whitened episodes. Logistic Regression is nearly unchanged (0.2–0.3 point drops), whereas TabICL loses up to 1.4 points and TabPFN more than 6 points. Thus, whitening does not universally improve representations themselves. The benefit instead comes from training the shared predictor on whitened episodes.

Table 2: Ablation of RepICL components and meta-training diversity. The full model reports accuracy (%) averaged across the three folds. All ablations report the change in percentage points relative to the full model, with positive changes bolded. Full fold-level results and the meta-training diversity setup are provided in Appendix D.2.

| | RepICL-I | | | | RepICL-T | | | |
|---|---|---|---|---|---|---|---|---|
| | Dataset | Encoder | Joint | Modality | Dataset | Encoder | Joint | Modality |
| **Full model** | **64.71** | **75.15** | **68.21** | **69.50** | **66.87** | **77.57** | **70.26** | **71.52** |
| ***Episodic whitening*** | | | | | | | | |
| w/o Coord. Standardize | −7.06 | −4.77 | −4.81 | −6.84 | −11.41 | −9.37 | −9.78 | −10.24 |
| w/o PCA, trainable proj. | −6.38 | −9.90 | −9.42 | −9.52 | −7.55 | −11.61 | −10.80 | −11.22 |
| w/o PCA, fixed proj. | −6.37 | −9.51 | −9.46 | −9.13 | −8.02 | −11.98 | −11.03 | −11.24 |
| ***Model structure*** | | | | | | | | |
| w/o CAML initialization | −0.07 | −0.12 | **+0.11** | −0.02 | **+0.34** | **+0.35** | **+0.21** | **+0.47** |
| fixed five-class head | −0.21 | −0.32 | **+0.13** | −0.08 | **+1.01** | **+0.91** | **+0.90** | **+1.18** |
| ***Meta-training diversity*** | | | | | | | | |
| one dataset per modality | −0.44 | −0.47 | −0.12 | −0.42 | −1.91 | −1.97 | −1.76 | −1.69 |
| one encoder per modality | −0.87 | −0.70 | −0.30 | −0.49 | −2.40 | −2.35 | −2.00 | −1.52 |

Table 3: Accuracy (%) on raw representations and the change after applying episodic whitening, averaged across the four adaptation settings within each modality-held-out fold. Results use 30 episodes per task due to compute constraints, so raw accuracies differ slightly from Table 1.

| | Logistic Regression | | | TabICL | | | TabPFN | | |
|---|---|---|---|---|---|---|---|---|---|
| | Text | Audio | Image | Text | Audio | Image | Text | Audio | Image |
| Raw representations | 67.44 | 69.59 | 68.36 | 65.70 | 67.81 | 66.59 | 64.66 | 66.42 | 65.37 |
| + Episodic whitening | -0.32 | -0.34 | -0.22 | −1.05 | −1.42 | −1.11 | −5.91 | −6.27 | −6.22 |

### 6.4 ANALYSES

We use support-prototype geometry to answer two questions raised by the main results: where does inductive RepICL improve over per-episode Logistic Regression, and among these difficult cases, when does transductive inference provide additional benefit?

**Where does RepICL-I improve over Logistic Regression?** For each query, we form class prototypes from the 5-shot support set in the unit-normalized representation space. Let $D_{\text{own}}$ denote the squared distance to the prototype of the ground-truth class and $D_{\text{other}}$ the distance to the nearest competing prototype. We define the prototype margin

$$m = \frac{D_{\text{other}} - D_{\text{own}}}{D_{\text{other}} + D_{\text{own}}}. \tag{5}$$

A positive margin indicates that the ground-truth prototype is nearest, whereas $m < 0$ indicates that a competing prototype is closer; we group $m \leq 0$ as queries not correctly resolved by the support prototypes.

Table 4 reveals a sharp difference between these regimes. For queries with $m \leq 0$, RepICL-I improves over Logistic Regression by 2.89 points and outperforms it across all negative-margin bins. In contrast, Logistic Regression is slightly stronger when $m > 0$, with its advantage becoming more consistent as the margin increases. The same pattern is amplified by RepICL-T. Thus, across both variants, the gains concentrate on queries for which simple support prototypes either favor the wrong class or provide little separation between the true class and competing classes.

**What does transductive inference add?** A non-positive margin can arise for different reasons: the 5-shot support set may provide an unrepresentative sample of its classes, or the query may remain ambiguous even when more labeled examples are available. To distinguish these cases, we recompute Equation 5 using leave-one-query-out prototypes formed from all other labeled examples in the episode, including the remaining queries, and denote the resulting *oracle prototype margin* by $m^\star$. Because $m^\star$ uses query labels, it is unavailable to all evaluated methods and serves only as an analysis diagnostic. Relative to the 5-shot margin $m$, it provides a less support-sensitive reference for the episode geometry.

Table 4: Accuracy (%) by prototype margin $m$, pooled across the three modality-held-out folds (query-weighted). The first two rows aggregate queries by the sign of $m$, while the remaining rows provide finer margin bins. Bold marks the best method in each row.

| Margin $m$ | Queries (%) | Logistic regression | TabICL | TAIL | PT-MAP | RepICL-I | RepICL-T |
|---|---|---|---|---|---|---|---|
| $m \leq 0$ | 36.3 | 24.66 | 22.19 | 18.22 | 22.34 | 27.55 | **34.27** |
| $m > 0$ | 63.7 | **92.78** | 91.59 | 86.39 | 91.15 | 92.33 | 91.86 |
| $< -0.20$ | 10.5 | 17.84 | 13.54 | 7.46 | 9.65 | 22.95 | **25.61** |
| $[-0.20, -0.10)$ | 8.2 | 19.92 | 16.74 | 12.25 | 15.45 | 23.50 | **28.72** |
| $[-0.10, -0.05)$ | 7.2 | 23.27 | 21.32 | 18.78 | 22.75 | 25.54 | **33.47** |
| $[-0.05, -0.02)$ | 5.9 | 29.80 | 29.82 | 28.37 | 33.50 | 30.51 | **42.08** |
| $[-0.02, 0]$ | 4.6 | 44.42 | 43.40 | 39.71 | 48.81 | 44.73 | **55.27** |
| $(0, 0.02)$ | 4.8 | 66.56 | 59.70 | 51.71 | 64.99 | 68.59 | **69.10** |
| $[0.02, 0.05)$ | 6.9 | 83.35 | 77.24 | 66.28 | 78.72 | **83.40** | 81.12 |
| $[0.05, 0.10)$ | 10.0 | **91.26** | 89.93 | 81.26 | 87.87 | 90.24 | 89.18 |
| $[0.10, 0.20)$ | 14.1 | 95.57 | **95.88** | 90.69 | 93.93 | 94.50 | 94.14 |
| $\geq 0.20$ | 27.9 | 98.67 | **98.95** | 96.88 | 98.42 | 98.20 | 98.15 |

Table 5: Accuracy (%) stratified by the signs of the 5-shot prototype margin $m$ and oracle prototype margin $m^\star$, pooled across the three modality-held-out folds (query-weighted). Bold marks the best method in each row. T − I reports the accuracy difference between RepICL-T and RepICL-I.

| $m$ | $m^\star$ | Queries (%) | Logistic regression | TabICL | TAIL | PT-MAP | RepICL-I | RepICL-T | T − I |
|---|---|---|---|---|---|---|---|---|---|
| $\leq 0$ | $\leq 0$ | 22.9 | 21.24 | 18.61 | 14.53 | 14.79 | 23.89 | **27.18** | $+\mathbf{3.29}$ |
| $\leq 0$ | $> 0$ | 13.5 | 30.48 | 28.29 | 24.50 | 35.17 | 33.75 | **46.32** | $+\mathbf{12.56}$ |
| $> 0$ | $\leq 0$ | 10.0 | **78.47** | 75.81 | 66.17 | 68.38 | 76.19 | 72.13 | $-\mathbf{4.06}$ |
| $> 0$ | $> 0$ | 53.7 | 95.44 | 94.53 | 90.15 | 95.39 | 95.34 | **95.53** | $+0.19$ |

Table 5 shows that the benefit of transduction depends strongly on the relationship between the 5-shot and oracle margins. When $m \leq 0$ but $m^\star > 0$, the 5-shot support prototypes favor a competing class, whereas prototypes estimated from the larger labeled reference set favor the ground-truth class. In this regime, RepICL-T improves over RepICL-I by 12.56 points. The gain is much smaller when both margins remain non-positive, becomes negligible when both are positive, and reverses when the support prototypes favor the correct class but the oracle margin does not. This asymmetry suggests that unlabeled query context is especially useful when limited support coverage gives a misleading view of class separation, rather than uniformly improving prediction across queries.

## 7 Conclusion, Limitations and Future Work

**Conclusion.** In this paper, we studied whether a few-shot prediction procedure, like pretrained representations, can be learned once and reused across heterogeneous frozen representation spaces. RepShiftBench exposes a substantial gap in this setting: existing in-context learners consistently trail simple per-episode Logistic Regression. RepICL shows that a shared predictor can nevertheless surpass the strongest per-episode baseline across changing datasets and representation spaces. Its inductive variant surpasses Logistic Regression across all benchmark settings, while RepICL-T achieves larger gains under transductive inference. Our analyses identify episodic whitening as the primary source of these improvements and show that RepICL's gains concentrate on queries that are poorly resolved by simple support prototypes, with transduction providing its largest additional benefit when support coverage is limited.

**Limitations and future work.** Our evaluation focuses on 5-way 5-shot classification, leaving generalization to other episode shapes, including different numbers of ways or shots, for future work. Moreover, while RepShiftBench varies representation spaces across tasks, each episode is still drawn from a single encoder–dataset pair. These limitations motivate two natural extensions: improving generalization across episode shapes, and moving from transfer *across* representation spaces to joint reasoning *over multiple* representation spaces within an episode. These directions point toward a broader goal: a reusable prediction procedure that could serve as a common adaptation layer between pretrained encoders and downstream tasks. Rather than fitting a new predictor whenever the dataset, encoder, or modality changes, such a procedure could amortize adaptation across them while exploiting richer episode-level structure than simple per-task fitting.

## AI USE STATEMENT

We used generative AI tools to discuss research ideas and experiments, write code, draft and polish the paper, and find related work, and did not use them for any other task that requires disclosure under the ICLR 2027 AI policy. We have reviewed all AI-assisted work: code, results, and claims were checked against the recorded experiments, and literature suggestions were verified against the cited sources. We take responsibility for the final content of this work, including text, claims, code, and analyses produced with the aid of generative AI.

## REPRODUCIBILITY STATEMENT

To support reproducibility, we describe our setup in the main paper and give further details in the appendices. Section 3 introduces the few-shot setup, and Section 4 introduces RepShiftBench, whose construction is detailed in Appendix A. Section 5 describes RepICL, and Appendix B gives its training details. The compared methods are introduced in Section 6.1 and detailed in Appendix C, and our reproduction of TAIL is described in Appendix C.4.

## REFERENCES

Arun Babu, Changhan Wang, Andros Tjandra, Kushal Lakhotia, Qiantong Xu, Naman Goyal, Kritika Singh, Patrick von Platen, Yatharth Saraf, Juan Pino, Alexei Baevski, Alexis Conneau, and Michael Auli. XLS-R: Self-supervised cross-lingual speech representation learning at scale. In *Interspeech*, pp. 2278–2282, 2022.

Alexei Baevski, Steffen Schneider, and Michael Auli. vq-wav2vec: Self-supervised learning of discrete speech representations. In *International Conference on Learning Representations*, 2020a.

Alexei Baevski, Yuhao Zhou, Abdelrahman Mohamed, and Michael Auli. wav2vec 2.0: A framework for self-supervised learning of speech representations. In *Advances in Neural Information Processing Systems*, volume 33, pp. 12449–12460, 2020b.

Alexei Baevski, Wei-Ning Hsu, Qiantong Xu, Arun Babu, Jiatao Gu, and Michael Auli. data2vec: A general framework for self-supervised learning in speech, vision and language. In *International Conference on Machine Learning*, pp. 1298–1312, 2022.

Samyadeep Basu, Shell Xu Hu, Daniela Massiceti, and Soheil Feizi. Strong baselines for parameter-efficient few-shot fine-tuning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 11024–11031, 2024.

Parishad BehnamGhader, Vaibhav Adlakha, Marius Mosbach, Dzmitry Bahdanau, Nicolas Chapados, and Siva Reddy. LLM2Vec: Large language models are secretly powerful text encoders. In *Conference on Language Modeling (COLM)*, 2024.

Etienne Bennequin. EasyFSL: Ready-to-use code and tutorial notebooks for few-shot learning. `https://github.com/sicara/easy-few-shot-learning`, 2021.

Daniel Bolya, Po-Yao Huang, Peize Sun, Jang Hyun Cho, Andrea Madotto, Chen Wei, Tengyu Ma, Jiale Zhi, Jathushan Rajasegaran, Hanoona Rasheed, Junke Wang, Marco Monteiro, Hu Xu, Shiyu Dong, Nikhila Ravi, Daniel Li, Piotr Dollár, and Christoph Feichtenhofer. Perception encoder: The best visual embeddings are not at the output of the network. In *Advances in Neural Information Processing Systems*, 2025.

Malik Boudiaf, Imtiaz Masud Ziko, Jérôme Rony, Jose Dolz, Pablo Piantanida, and Ismail Ben Ayed. Transductive information maximization for few-shot learning. In *Advances in Neural Information Processing Systems*, 2020.

John Bronskill, Jonathan Gordon, James Requeima, Sebastian Nowozin, and Richard E. Turner. TaskNorm: Rethinking batch normalization for meta-learning. In *International Conference on Machine Learning*, 2020.

Heng-Jui Chang, Alexander H. Liu, Saurabhchand Bhati, Mrudula Athi, Anton Ratnarajah, Amit Chhetri, and James Glass. USAD 2.0: Scaling representation distillation for universal audio understanding. In *Interspeech*, 2026.

Sanyuan Chen, Chengyi Wang, Zhengyang Chen, Yu Wu, Shujie Liu, Zhuo Chen, Jinyu Li, Naoyuki Kanda, Takuya Yoshioka, Xiong Xiao, Jian Wu, Long Zhou, Shuo Ren, Yanmin Qian, Yao Qian, Jian Wu, Michael Zeng, Xiangzhan Yu, and Furu Wei. WavLM: Large-scale self-supervised pre-training for full stack speech processing. *IEEE Journal of Selected Topics in Signal Processing*, 16(6):1505–1518, 2022.

Sanyuan Chen, Yu Wu, Chengyi Wang, Shujie Liu, Daniel Tompkins, Zhuo Chen, Wanxiang Che, Xiangzhan Yu, and Furu Wei. BEATs: Audio pre-training with acoustic tokenizers. In *International Conference on Machine Learning*, pp. 5178–5193, 2023.

Wei-Yu Chen, Yen-Cheng Liu, Zsolt Kira, Yu-Chiang Frank Wang, and Jia-Bin Huang. A closer look at few-shot classification. In *International Conference on Learning Representations*, 2019.

Wenxi Chen, Yuzhe Liang, Ziyang Ma, Zhisheng Zheng, and Xie Chen. EAT: Self-supervised pre-training with efficient audio transformer. In *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence (IJCAI)*, 2024.

DeepSeek-AI. DeepSeek-R1 incentivizes reasoning in LLMs through reinforcement learning. *Nature*, 645:633–638, 2025.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics*, pp. 4171–4186, 2019.

Heinrich Dinkel, Zhiyong Yan, Yongqing Wang, Junbo Zhang, Yujun Wang, and Bin Wang. Scaling up masked audio encoder learning for general audio classification. In *Interspeech*, pp. 547–551, 2024.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021.

Adnan El Assadi, Isaac Chung, Chenghao Xiao, Roman Solomatin, Animesh Jha, Rahul Chand, Silky Singh, Kaitlyn Wang, Ali Sartaz Khan, Marc Moussa Nasser, Sufen Fong, Pengfei He, Alan Xiao, Ayush Sunil Munot, Aditya Shrivastava, Artem Gazizov, Niklas Muennighoff, and Kenneth Enevoldsen. MAEB: Massive audio embedding benchmark. *arXiv preprint arXiv:2602.16008*, 2026a.

Adnan El Assadi, Niklas Muennighoff, and Jinhyuk Lee. The embedder's dilemma: LLMs are better, but at what cost? In *Conference on Language Modeling (COLM)*, 2026b.

Christopher Fifty, Dennis Duan, Ronald G. Junkins, Ehsan Amid, Jure Leskovec, Christopher Ré, and Sebastian Thrun. Context-aware meta-learning. In *International Conference on Learning Representations*, 2024.

Enrico Fini, Mustafa Shukor, Xiujun Li, Philipp Dufter, Michal Klein, David Haldimann, Sai Aitharaju, Victor Guilherme Turrisi da Costa, Louis Béthune, Zhe Gan, Alexander T. Toshev, Marcin Eichner, Moin Nabi, Yinfei Yang, Joshua M. Susskind, and Alaaeldin El-Nouby. Multimodal autoregressive pre-training of large vision encoders. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *International Conference on Machine Learning*, pp. 1126–1135, 2017.

Marta Garnelo, Dan Rosenbaum, Christopher J. Maddison, Tiago Ramalho, David Saxton, Murray Shanahan, Yee Whye Teh, Danilo Jimenez Rezende, and S. M. Ali Eslami. Conditional neural processes. In *International Conference on Machine Learning*, pp. 1704–1713, 2018.

Gemma Team. Gemma 3 technical report. *arXiv preprint arXiv:2503.19786*, 2025.

Yuan Gong, Yu-An Chung, and James Glass. AST: Audio spectrogram transformer. In *Interspeech*, pp. 571–575, 2021.

Yuan Gong, Cheng-I Jeff Lai, Yu-An Chung, and James Glass. SSAST: Self-supervised audio spectrogram transformer. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pp. 10699–10709, 2022.

Aaron Grattafiori et al. The Llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Edouard Grave, Piotr Bojanowski, Prakhar Gupta, Armand Joulin, and Tomas Mikolov. Learning word vectors for 157 languages. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC)*, 2018.

Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 16000–16009, 2022.

Pengcheng He, Jianfeng Gao, and Weizhu Chen. DeBERTaV3: Improving DeBERTa using ELECTRA-style pre-training with gradient-disentangled embedding sharing. In *International Conference on Learning Representations*, 2023.

Noah Hollmann, Samuel Müller, Katharina Eggensperger, and Frank Hutter. TabPFN: A transformer that solves small tabular classification problems in a second. In *International Conference on Learning Representations*, 2023.

Noah Hollmann, Samuel Müller, Lennart Purucker, Arjun Krishnakumar, Max Körfer, Shi Bin Hoo, Robin Tibor Schirrmeister, and Frank Hutter. Accurate predictions on small data with a tabular foundation model. *Nature*, 637(8045):319–326, 2025.

Florian Hönicke, Michael Günther, Andreas Koukounas, Mohammad Kalim Akram, Saba Sturua, and Han Xiao. jina-embeddings-v5-omni: Geometry-preserving embeddings via locked aligned towers. *arXiv preprint arXiv:2605.08384*, 2026.

Timothy M. Hospedales, Antreas Antoniou, Paul Micaelli, and Amos Storkey. Meta-learning in neural networks: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44 (9):5149–5169, 2022.

Andrew Howard, Mark Sandler, Grace Chu, Liang-Chieh Chen, Bo Chen, Mingxing Tan, Weijun Wang, Yukun Zhu, Ruoming Pang, Vijay Vasudevan, Quoc V. Le, and Hartwig Adam. Searching for MobileNetV3. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 1314–1324, 2019.

Wei-Ning Hsu, Benjamin Bolte, Yao-Hung Hubert Tsai, Kushal Lakhotia, Ruslan Salakhutdinov, and Abdelrahman Mohamed. HuBERT: Self-supervised speech representation learning by masked prediction of hidden units. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 29:3451–3460, 2021.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022a.

Shell Xu Hu, Da Li, Jan Stühmer, Minyoung Kim, and Timothy M. Hospedales. Pushing the limits of simple pipelines for few-shot learning: External data and fine-tuning make a difference. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9068–9077, 2022b.

Yuqing Hu, Vincent Gripon, and Stéphane Pateux. Leveraging the feature distribution in transfer-based few-shot learning. In *International Conference on Artificial Neural Networks (ICANN)*, 2021.

Gao Huang, Zhuang Liu, Laurens van der Maaten, and Kilian Q. Weinberger. Densely connected convolutional networks. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4700–4708, 2017.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. Mistral 7B. *arXiv preprint arXiv:2310.06825*, 2023.

Louis Kirsch, James Harrison, Jascha Narain Sohl-Dickstein, and Luke Metz. General-purpose in-context learning by meta-learning transformers. In *NeurIPS Workshop on Meta-Learning*, 2022.

Simon Kornblith, Jonathon Shlens, and Quoc V. Le. Do better imagenet models transfer better? In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2661–2671, 2019.

Chankyu Lee, Rajarshi Roy, Mengyao Xu, Jonathan Raiman, Mohammad Shoeybi, Bryan Catanzaro, and Wei Ping. NV-Embed: Improved techniques for training LLMs as generalist embedding models. In *International Conference on Learning Representations*, 2025.

Chaofan Li, Minghao Qin, Shitao Xiao, Jianlyu Chen, Kun Luo, Defu Lian, Yingxia Shao, and Zheng Liu. Making text embedders few-shot learners. In *International Conference on Learning Representations*, 2025.

Mingxin Li, Yanzhao Zhang, Dingkun Long, Keqin Chen, Sibo Song, Shuai Bai, Zhibo Yang, Pengjun Xie, An Yang, Dayiheng Liu, Jingren Zhou, and Junyang Lin. Qwen3-VL-Embedding and Qwen3-VL-Reranker: A unified framework for state-of-the-art multimodal retrieval and ranking. *arXiv preprint arXiv:2601.04720*, 2026.

Wei-Hong Li, Xialei Liu, and Hakan Bilen. Cross-domain few-shot learning with task-specific adapters. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022.

Andy T. Liu, Shang-Wen Li, and Hung-yi Lee. TERA: Self-supervised learning of transformer encoder representation for speech. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 29:2351–2366, 2021.

Yanbin Liu, Juho Lee, Minseop Park, Saehoon Kim, Eunho Yang, Sung Ju Hwang, and Yi Yang. Learning to propagate labels: Transductive propagation network for few-shot learning. In *International Conference on Learning Representations*, 2019.

Ze Liu, Han Hu, Yutong Lin, Zhuliang Yao, Zhenda Xie, Yixuan Wei, Jia Ning, Yue Cao, Zheng Zhang, Li Dong, Furu Wei, and Baining Guo. Swin Transformer V2: Scaling up capacity and resolution. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 12009–12019, 2022a.

Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A ConvNet for the 2020s. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11976–11986, 2022b.

Daniel Morales-Brotons, Thijs Vogels, and Hadrien Hendrikx. Exponential moving average of weights in deep learning: Dynamics and benefits. *Transactions on Machine Learning Research*, 2024.

Luca Moschella, Valentino Maiorca, Marco Fumero, Antonio Norelli, Francesco Locatello, and Emanuele Rodolà. Relative representations enable zero-shot latent space communication. In *International Conference on Learning Representations*, 2023.

Niklas Muennighoff, Nouamane Tazi, Loïc Magne, and Nils Reimers. MTEB: Massive text embedding benchmark. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pp. 2014–2037, 2023.

Niklas Muennighoff, Hongjin Su, Liang Wang, Nan Yang, Furu Wei, Tao Yu, Amanpreet Singh, and Douwe Kiela. Generative representational instruction tuning. In *International Conference on Learning Representations*, 2025.

Samuel Müller, Noah Hollmann, Sebastian Pineda Arango, Josif Grabocka, and Frank Hutter. Transformers can do Bayesian inference. In *International Conference on Learning Representations*, 2022.

Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, Mahmoud Assran, Nicolas Ballas, Wojciech Galuba, Russell Howes, Po-Yao Huang, Shang-Wen Li, Ishan Misra, Michael Rabbat, Vasu Sharma, Gabriel Synnaeve, Hu Xu, Herve Jegou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. DINOv2: Learning robust visual features without supervision. *Transactions on Machine Learning Research*, 2024.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Édouard Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12: 2825–2830, 2011.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1532–1543, 2014.

Boris T. Polyak and Anatoli B. Juditsky. Acceleration of stochastic approximation by averaging. *SIAM Journal on Control and Optimization*, 30(4):838–855, 1992.

Jingang Qu, David Holzmüller, Gaël Varoquaux, and Marine Le Morvan. TabICL: A tabular foundation model for in-context learning on large data. In *International Conference on Machine Learning*, 2025.

Jingang Qu, David Holzmüller, Gaël Varoquaux, and Marine Le Morvan. TabICLv2: A better, faster, scalable, and open tabular foundation model. In *International Conference on Machine Learning*, 2026.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. Technical report, OpenAI, 2019.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pp. 8748–8763, 2021.

Alec Radford, Jong Wook Kim, Tao Xu, Greg Brockman, Christine McLeavey, and Ilya Sutskever. Robust speech recognition via large-scale weak supervision. In *International Conference on Machine Learning*, pp. 28492–28518, 2023.

Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*, pp. 3982–3992, 2019.

Adam Santoro, Sergey Bartunov, Matthew Botvinick, Daan Wierstra, and Timothy Lillicrap. Meta-learning with memory-augmented neural networks. In *International Conference on Machine Learning*, 2016.

Oriane Siméoni, Huy V. Vo, Maximilian Seitzer, Federico Baldassarre, Maxime Oquab, Cijo Jose, Vasil Khalidov, Marc Szafraniec, Seungeun Yi, Michaël Ramamonjisoa, Francisco Massa, Daniel Haziza, Luca Wehrstedt, Jianyuan Wang, Timothée Darcet, Théo Moutakanni, Leonel Sentana, Claire Roberts, Andrea Vedaldi, Jamie Tolan, John Brandt, Camille Couprie, Julien Mairal, Hervé Jégou, Patrick Labatut, and Piotr Bojanowski. DINOv3. *arXiv preprint arXiv:2508.10104*, 2025.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations*, 2015.

Jake Snell, Kevin Swersky, and Richard S. Zemel. Prototypical networks for few-shot learning. In *Advances in Neural Information Processing Systems*, 2017.

Jianlin Su, Jiarun Cao, Weijie Liu, and Yangyiwen Ou. Whitening sentence representations for better semantics and faster retrieval. *arXiv preprint arXiv:2103.15316*, 2021.

Quan Sun, Yuxin Fang, Ledell Wu, Xinlong Wang, and Yue Cao. EVA-CLIP: Improved training techniques for CLIP at scale. *arXiv preprint arXiv:2303.15389*, 2023.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2818–2826, 2016.

Mingxing Tan and Quoc V. Le. EfficientNetV2: Smaller models and faster training. In *International Conference on Machine Learning*, pp. 10096–10106, 2021.

Changli Tang, Qinfan Xiao, Ke Mei, Tianyi Wang, Fengyun Rao, and Chao Zhang. WAVE: Learning unified & versatile audio-visual embeddings with multimodal LLM. In *International Conference on Learning Representations*, 2026.

Yonglong Tian, Yue Wang, Dilip Krishnan, Joshua B. Tenenbaum, and Phillip Isola. Rethinking few-shot image classification: a good embedding is all you need? In *European Conference on Computer Vision*, 2020.

Eleni Triantafillou, Tyler Zhu, Vincent Dumoulin, Pascal Lamblin, Utku Evci, Kelvin Xu, Ross Goroshin, Carles Gelada, Kevin Swersky, Pierre-Antoine Manzagol, and Hugo Larochelle. Meta-dataset: A dataset of datasets for learning to learn from few examples. In *International Conference on Learning Representations*, 2020.

Michael Tschannen, Alexey Gritsenko, Xiao Wang, Muhammad Ferjad Naeem, Ibrahim Alabdulmohsin, Nikhil Parthasarathy, Talfan Evans, Lucas Beyer, Ye Xia, Basil Mustafa, Olivier Hénaff, Jeremiah Harmsen, Andreas Steiner, and Xiaohua Zhai. SigLIP 2: Multilingual vision-language encoders with improved semantic understanding, localization, and dense features. *arXiv preprint arXiv:2502.14786*, 2025.

Hung-Yu Tseng, Hsin-Ying Lee, Jia-Bin Huang, and Ming-Hsuan Yang. Cross-domain few-shot classification via learned feature-wise transformation. In *International Conference on Learning Representations*, 2020.

Ihsan Ullah, Dustin Carrión-Ojeda, Sergio Escalera, Isabelle Guyon, Mike Huisman, Felix Mohr, Jan N. van Rijn, Haozhe Sun, Joaquin Vanschoren, and Phan Anh Vu. Meta-Album: Multi-domain meta-dataset for few-shot image classification. In *Advances in Neural Information Processing Systems, Datasets and Benchmarks Track*, 2022.

Vladimir N. Vapnik. *Statistical Learning Theory*. Wiley, 1998.

Olivier Veilleux, Malik Boudiaf, Pablo Piantanida, and Ismail Ben Ayed. Realistic evaluation of transductive few-shot learning. In *Advances in Neural Information Processing Systems*, 2021.

Henrique Schechter Vera et al. EmbeddingGemma: Powerful and lightweight text representations. *arXiv preprint arXiv:2509.20354*, 2025.

Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Koray Kavukcuoglu, and Daan Wierstra. Matching networks for one shot learning. In *Advances in Neural Information Processing Systems*, 2016.

Changhan Wang, Yun Tang, Xutai Ma, Anne Wu, Sravya Popuri, Dmytro Okhonko, and Juan Pino. Fairseq S2T: Fast speech-to-text modeling with fairseq. *arXiv preprint arXiv:2010.05171*, 2020. Version 2, June 2022; post-conference update of the AACL 2020 demo paper.

Liang Wang, Nan Yang, Xiaolong Huang, Binxing Jiao, Linjun Yang, Daxin Jiang, Rangan Majumder, and Furu Wei. Text embeddings by weakly-supervised contrastive pre-training. *arXiv preprint arXiv:2212.03533*, 2022.

Liang Wang, Nan Yang, Xiaolong Huang, Linjun Yang, Rangan Majumder, and Furu Wei. Improving text embeddings with large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 11897–11916, 2024.

Yan Wang, Wei-Lun Chao, Kilian Q. Weinberger, and Laurens van der Maaten. SimpleShot: Revisiting nearest-neighbor classification for few-shot learning. *arXiv preprint arXiv:1911.04623*, 2019.

Benjamin Warner, Antoine Chaffin, Benjamin Clavié, Orion Weller, Oskar Hallström, Said Taghadouini, Alexis Gallagher, Raja Biswas, Faisal Ladhak, Tom Aarsen, Griffin Thomas Adams, Jeremy Howard, and Iacopo Poli. Smarter, better, faster, longer: A modern bidirectional encoder for fast, memory efficient, and long context finetuning and inference. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 2526–2547, 2025.

Stefano Woerner, Seong Joon Oh, and Christian F. Baumgartner. Universal algorithm-implicit learning. In *International Conference on Machine Learning*, 2026.

Yusong Wu, Ke Chen, Tianyu Zhang, Yuchen Hui, Taylor Berg-Kirkpatrick, and Shlomo Dubnov. Large-scale contrastive language-audio pretraining with feature fusion and keyword-to-caption augmentation. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 1–5, 2023.

Chenghao Xiao, Hou Pong Chan, Hao Zhang, Weiwen Xu, Mahani Aljunied, and Yu Rong. Scaling language-centric omnimodal representation learning. In *Advances in Neural Information Processing Systems*, 2025a.

Chenghao Xiao, Isaac Chung, Imene Kerboua, Jamie Stirling, Xin Zhang, Márton Kardos, Roman Solomatin, Noura Al Moubayed, Kenneth Enevoldsen, and Niklas Muennighoff. MIEB: Massive image embedding benchmark. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 22187–22198, 2025b.

Shitao Xiao, Zheng Liu, Peitian Zhang, Niklas Muennighoff, Defu Lian, and Jian-Yun Nie. C-Pack: Packed resources for general chinese embeddings. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 641–649, 2024.

Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1492–1500, 2017.

Mengyao Xu, Wenfei Zhou, Yauhen Babakhin, Gabriel Moreira, Ronay Ak, Radek Osmulski, Bo Liu, Even Oldridge, and Benedikt Schifferer. Omni-Embed-Nemotron: A unified multimodal retrieval model for text, image, audio, and video. *arXiv preprint arXiv:2510.03458*, 2025.

An Yang et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Han-Jia Ye, Hexiang Hu, De-Chuan Zhan, and Fei Sha. Few-shot learning via embedding adaptation with set-to-set functions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 8808–8817, 2020.

Maxime Zanella and Ismail Ben Ayed. Low-rank few-shot adaptation of vision-language models. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2024.

Xiaohua Zhai, Basil Mustafa, Alexander Kolesnikov, and Lucas Beyer. Sigmoid loss for language image pre-training. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 11975–11986, 2023.

Dun Zhang, Jiacheng Li, Ziyang Zeng, and Fulong Wang. Jasper and stella: distillation of SOTA embedding models. *arXiv preprint arXiv:2412.19048*, 2024.

Junbo Zhang, Heinrich Dinkel, Yadong Niu, Chenyu Liu, Si Cheng, Anbei Zhao, and Jian Luan. X-ARES: A comprehensive framework for assessing audio encoder performance. In *Interspeech*, pp. 4868–4872, 2025a.

Yanzhao Zhang, Mingxin Li, Dingkun Long, Xin Zhang, Huan Lin, Baosong Yang, Pengjun Xie, An Yang, Dayiheng Liu, Junyang Lin, Fei Huang, and Jingren Zhou. Qwen3 Embedding: Advancing text embedding and reranking through foundation models. *arXiv preprint arXiv:2506.05176*, 2025b.

Imtiaz Masud Ziko, Jose Dolz, Eric Granger, and Ismail Ben Ayed. Laplacian regularized few-shot learning. In *International Conference on Machine Learning*, 2020.

# A REPSHIFTBENCH CONSTRUCTION

## A.1 DATASETS AND ENCODERS

**Datasets.** RepShiftBench covers three modalities: text, image, and audio. For each modality, we draw datasets from established embedding benchmarks that span diverse application domains and classification tasks: MTEB for text, MIEB for images, and X-ARES and MAEB for audio. We select single-label, example-level classification tasks that support our five-way episodic protocol; this excludes, for example, multi-label and frame-level annotation tasks. In addition, we adapt six MTEB clustering datasets for classification by treating their cluster labels as classes.

**Encoders.** We select encoders to capture diversity across model families, training objectives, and release periods. The inventory includes models commonly used for classification as well as models designed to produce general-purpose embeddings. Their public release dates range from 2014 to 2026. When a model family has several variants, we include one variant to limit redundancy. Tables 6 and 7 list the full inventory.

## A.2 REPRESENTATION EXTRACTION

**Preprocessing.** Every encoder receives inputs preprocessed as it requires, such as image resizing and normalization or audio resampling. Beyond this, we apply a few dataset-specific steps. For text, the MTEB clustering datasets store each record as a list of texts with their cluster labels, so we expand each list into one example per text and remove exact duplicates. We also exclude text examples longer than 2,048 tokens, measured with a single fixed tokenizer (GPT-2), so that every text encoder sees the same examples. Due to limited GPU resources, we further truncate the inputs of decoder-only LLMs used through their hidden states, GritLM, and LLM2Vec to 512 tokens, and keep only the first 30 seconds of each VoxCeleb1 and BirdCLEF clip.

**Embedding extraction.** Each frozen encoder then maps the preprocessed inputs to one embedding per example, which we compute once and cache. The embedding is obtained by selecting a designated token or averaging the encoder's outputs over tokens, patches, or frames, depending on the modality, and is then L2-normalized. We generally follow each encoder's released embedding pipeline where one exists.

## A.3 FOLDS, SPLITS, AND ADAPTATION SETTINGS

**Folds.** The three folds of Section 4.2 let us compare meta-trained methods fairly against few-shot methods that receive no meta-training. Each fold holds out one of the three modalities (text, image, or audio) and draws its meta-training tasks only from the other two. Together with the dataset and encoder splits below, this keeps every test task out of meta-training, so a meta-trained method is never tested on a task it was trained on. Rotating the held-out modality across the three folds tests generalization on every modality, not just one.

**Dataset and encoder splits.** Within each modality, datasets and encoders are separately partitioned into seen and held-out sets, and this partition is shared by all folds. In the fold where a modality is held out, none of its datasets or encoders is used for meta-training, including those in its seen set. For encoders, the held-out set contains the most recently released encoders of the modality, so that Encoder Adaptation tests transfer to newer models. For datasets, the held-out set includes both out-of-domain datasets, such as remote-sensing images and speaker counting, and datasets whose task type also appears in the seen set. This tests generalization to unfamiliar domains as well as to new datasets of familiar task types. Tables 6 and 7 give the split of every dataset and encoder.

**Task sets and adaptation settings.** In each fold, every task belongs to the meta-training, validation, or test set. In the two meta-training modalities, the tasks whose dataset and encoder are both

seen, which form the meta-training pool of Section 4.2, are divided into a meta-training set $\mathcal{T}_{\text{train}}$ (Section 4.1) and a validation set, with about 20% of them in the latter for model and hyperparameter selection. All other tasks form the test set, and each belongs to one of the four adaptation settings of Section 4.1. A task of the held-out modality is a Modality Adaptation task. A task of a meta-training modality is a Dataset, Encoder, or Joint Adaptation task, depending on whether its dataset, its encoder, or both are held out. The four adaptation settings represent different levels of generalization: an unseen dataset, an unseen encoder, both, or an unseen modality.

The same task can fall into different adaptation settings in different folds. A text task with a seen dataset and a held-out encoder, for example, is an Encoder Adaptation task when image is held out, but a Modality Adaptation task when text is held out. The adaptation settings describe what a meta-trained method has seen before testing, so in Table 1 they apply only to RepICL. For the other methods, the folds and adaptation settings only determine which tasks they are evaluated on and how these tasks are grouped. Evaluating RepICL in each adaptation setting shows whether its ability is limited to particular adaptation settings or is consistent across all of them.

### A.4 Episode Construction

**Episode sampling.** RepShiftBench defines all evaluation episodes as 5-way 5-shot with 10 queries per class, and we use the same shape for meta-training and validation. Each episode is sampled from a single task: we sample 5 classes, then 5 support and 10 query examples per class, all without replacement, and randomly permute the class labels in each episode. Episodes therefore use only classes with enough examples for this shape, and a task needs at least five such classes. Table 6 reports, for every dataset, the number of classes that meet this requirement and the number of all classes.

**Example pools.** Each dataset has a support pool and a query pool of examples, shared by all encoders. Both follow the splits of the source benchmark: the support pool is its original training split and the query pool its test split. If the source lacks such a split, we create it by class-stratified splitting; for a few sources that provide a single labeled split, both pools are that split and support and queries are sampled without overlap. Episodes of tasks in the test set sample 5 support examples per class from the support pool and 10 queries per class from the query pool. Meta-training and validation episodes sample all 15 examples per class from the support pool, so the query pool is never used for meta-training or validation.

**Fixed evaluation episodes.** We sample the 1,000 evaluation episodes of each task in the test set once and use them, including their label assignment, for all methods. A task also keeps the same episodes when it appears in different folds and adaptation settings, so differences across settings do not come from different episodes.

Table 6: Datasets of RepShiftBench, grouped by modality and by seen/held-out set (Appendix A.3). #Classes gives available and all classes (Appendix A.4).

| Dataset | Category | #Classes (avail./all) | #Examples | Source |
|---|---|---|---|---|
| *Text* (seen) | | | | |
| ArXivHierarchicalClusteringS2S | Academic-paper clustering | 169 / 169 | 627,949 | MTEB |
| Banking77Classification | Intent classification | 77 / 77 | 13,069 | MTEB |
| EmotionClassification | Emotion classification | 6 / 6 | 19,930 | MTEB |
| HWU64 | Intent classification | 57 / 64 | 10,030 | MTEB |
| MassiveIntentClassification | Intent classification | 54 / 60 | 16,521 | MTEB |
| MassiveScenarioClassification | Scenario classification | 18 / 18 | 16,521 | MTEB |
| MTOPDomainClassification | Domain classification | 11 / 11 | 19,723 | MTEB |
| MTOPIntentClassification | Intent classification | 32 / 113 | 19,723 | MTEB |
| RedditClustering | Topic clustering | 50 / 50 | 64,337 | MTEB |
| StackExchangeClustering | Topic clustering | 171 / 171 | 514,513 | MTEB |
| TREC (coarse) | Question-type classification | 5 / 6 | 5,952 | MTEB |
| TweetTopicSingleClassification | Topic classification | 6 / 6 | 3,209 | MTEB |
| TwentyNewsgroupsClustering | Topic clustering | 20 / 20 | 5,242 | MTEB |
| YahooAnswersTopicsClassification | Topic classification | 10 / 10 | 1,341,064 | MTEB |
| *held-out* | | | | |
| AmazonReviewsClassification | Review sentiment classification | 5 / 5 | 210,000 | MTEB |
| BiorxivClusteringS2S | Academic-paper clustering | 25 / 26 | 53,773 | MTEB |
| CLINC150 (no OOS) | Intent classification | 150 / 150 | 22,500 | MTEB |
| DBpediaClassification | Topic classification | 14 / 14 | 4,096 | MTEB |
| MedrxivClusteringS2S | Academic-paper clustering | 49 / 51 | 17,646 | MTEB |
| *Image* (seen) | | | | |
| BirdsnapClassification | Fine-grained bird classification | 500 / 500 | 17,849 | MIEB |
| Caltech101Classification | Generic object classification | 87 / 102 | 9,144 | MIEB |
| CIFAR10Classification | Generic object classification | 10 / 10 | 60,000 | MIEB |
| CIFAR100Classification | Generic object classification | 100 / 100 | 60,000 | MIEB |
| Country211Classification | Geolocation classification | 211 / 211 | 52,750 | MIEB |
| DTDClassification | Texture classification | 47 / 47 | 5,640 | MIEB |
| FER2013Classification | Facial-expression classification | 7 / 7 | 35,887 | MIEB |
| Food101Classification | Fine-grained food classification | 101 / 101 | 101,000 | MIEB |
| ImageNet1KClassification | Generic object classification | 1000 / 1000 | 1,331,167 | MIEB |
| MNISTClassification | Handwritten-digit classification | 10 / 10 | 70,000 | MIEB |
| OxfordFlowersClassification | Fine-grained flower classification | 83 / 102 | 8,189 | MIEB |
| OxfordPetsClassification | Fine-grained pet classification | 37 / 37 | 7,349 | MIEB |
| StanfordCarsClassification | Fine-grained car classification | 196 / 196 | 16,185 | MIEB |
| STL10Classification | Generic object classification | 10 / 10 | 13,000 | MIEB |
| SUN397Classification | Scene classification | 397 / 397 | 108,752 | MIEB |
| *held-out* | | | | |
| EuroSATClassification | Remote-sensing classification | 10 / 10 | 27,000 | MIEB |
| FGVCAircraftClassification | Fine-grained aircraft classification | 100 / 100 | 10,000 | MIEB |
| GTSRBClassification | Traffic-sign classification | 43 / 43 | 39,270 | MIEB |
| RESISC45Classification | Remote-sensing classification | 45 / 45 | 31,500 | MIEB |
| *Audio* (seen) | | | | |
| CREMA-D | Speech-emotion classification | 6 / 6 | 7,442 | X-ARES |
| ESC-50 | Environmental-sound classification | 50 / 50 | 2,000 | X-ARES |
| Fluent Speech Commands | Spoken-intent classification | 247 / 248 | 30,043 | X-ARES |
| Free Music Archive Small | Music-genre classification | 8 / 8 | 7,994 | X-ARES |
| FSD18-Kaggle | Sound-event classification | 41 / 41 | 11,073 | X-ARES |
| GTZAN Genre | Music-genre classification | 9 / 10 | 999 | X-ARES |
| MInDS-14 | Spoken-intent classification | 14 / 14 | 8,168 | MAEB |
| NSynth-Instruments | Musical-instrument classification | 10 / 11 | 71,774 | X-ARES |
| SIB-FLEURS | Spoken-topic classification | 7 / 7 | 11,424 | MAEB |
| Speech Commands V1 | Keyword spotting | 30 / 30 | 64,721 | X-ARES |
| VocalSound | Vocal-sound classification | 6 / 6 | 20,985 | X-ARES |
| VoxCeleb1 | Speaker identification | 94 / 1251 | 153,412 | X-ARES |
| VoxLingua33 | Language identification | 24 / 33 | 67,609 | X-ARES |
| VoxPopuliLanguageID | Language identification | 5 / 5 | 500 | MAEB |
| *held-out* | | | | |
| BirdCLEF | Bioacoustic classification | 50 / 50 | 1,000 | MAEB |
| CommonLanguageAgeDetection | Speaker-age classification | 5 / 8 | 6,000 | MAEB |
| LibriCount | Speaker-count classification | 11 / 11 | 5,720 | X-ARES |
| MridinghamTonic | Music-tonic classification | 6 / 6 | 6,977 | MAEB |
| RAVDESS | Speech-emotion classification | 8 / 8 | 1,440 | X-ARES |
| UrbanSound 8K | Environmental-sound classification | 10 / 10 | 8,732 | X-ARES |
| Vocal Imitation | Vocal-imitation classification | 9 / 302 | 5,601 | X-ARES |

Table 7: Encoders of RepShiftBench, grouped by modality and by seen/held-out set (Appendix A.3), and ordered within each group by the release date of the cited paper. Checkpoints are from Hugging Face unless marked *, which come from other sources. †SigLIP 2 (2025-02) takes the release date and split of the original SigLIP.

| Checkpoint | Release | Dim. | Ref. |
|---|---|---|---|
| *Text* (seen) | | | |
| stanfordnlp/glove.6B.300d* | 2014-10 | 300 | Pennington et al. (2014) |
| facebook/fasttext-en-vectors | 2018-02 | 300 | Grave et al. (2018) |
| google-bert/bert-base-uncased | 2018-10 | 768 | Devlin et al. (2019) |
| openai-community/gpt2 | 2019-02 | 768 | Radford et al. (2019) |
| sentence-transformers/all-mpnet-base-v2 | 2019-08 | 768 | Reimers & Gurevych (2019) |
| openai/clip-vit-large-patch14 | 2021-02 | 768 | Radford et al. (2021) |
| microsoft/deberta-v3-large | 2021-11 | 1024 | He et al. (2023) |
| intfloat/e5-large-v2 | 2022-12 | 1024 | Wang et al. (2022) |
| google/siglip-so400m-patch14-224 | 2023-03 | 1152 | Zhai et al. (2023) |
| BAAI/bge-large-en-v1.5 | 2023-09 | 1024 | Xiao et al. (2024) |
| mistralai/Mistral-7B-Instruct-v0.3 | 2023-10 | 4096 | Jiang et al. (2023) |
| intfloat/e5-mistral-7b-instruct | 2023-12 | 4096 | Wang et al. (2024) |
| GritLM/GritLM-7B | 2024-02 | 4096 | Muennighoff et al. (2025) |
| McGill-NLP/LLM2Vec-Mistral-7B-Instruct-v2-mntp-supervised | 2024-04 | 4096 | BehnamGhader et al. (2024) |
| nvidia/NV-Embed-v2 | 2024-05 | 4096 | Lee et al. (2025) |
| meta-llama/Meta-Llama-3.1-8B-Instruct | 2024-07 | 4096 | Grattafiori et al. (2024) |
| BAAI/bge-en-icl | 2024-09 | 4096 | Li et al. (2025) |
| NovaSearch/stella_en_1.5B_v5 | 2024-12 | 1024 | Zhang et al. (2024) |
| answerdotai/ModernBERT-base | 2024-12 | 768 | Warner et al. (2025) |
| *held-out* | | | |
| deepseek-ai/DeepSeek-R1-Distill-Qwen-7B | 2025-01 | 3584 | DeepSeek-AI (2025) |
| google/gemma-3-4b-it | 2025-03 | 2560 | Gemma Team (2025) |
| Qwen/Qwen3-8B | 2025-05 | 4096 | Yang et al. (2025) |
| Qwen/Qwen3-Embedding-8B | 2025-06 | 4096 | Zhang et al. (2025b) |
| google/embeddinggemma-300m | 2025-09 | 768 | Vera et al. (2025) |
| *Image* (seen) | | | |
| timm/vgg16.tv_in1k | 2014-09 | 4096 | Simonyan & Zisserman (2015) |
| timm/inception_v3.tv_in1k | 2015-12 | 2048 | Szegedy et al. (2016) |
| timm/densenet121.tv_in1k | 2016-08 | 1024 | Huang et al. (2017) |
| timm/resnext101_32x8d.tv_in1k | 2016-11 | 2048 | Xie et al. (2017) |
| timm/mobilenetv3_large_100.miil_in21k_ft_in1k | 2019-05 | 960 | Howard et al. (2019) |
| google/vit-large-patch16-224-in21k | 2020-10 | 1024 | Dosovitskiy et al. (2021) |
| openai/clip-vit-large-patch14 | 2021-02 | 768 | Radford et al. (2021) |
| timm/tf_efficientnetv2_l.in21k_ft_in1k | 2021-04 | 1280 | Tan & Le (2021) |
| microsoft/swinv2-large-patch4-window12to16-192to256-22kto1k-ft | 2021-11 | 1536 | Liu et al. (2022a) |
| timm/vit_large_patch16_224.mae | 2021-11 | 1024 | He et al. (2022) |
| facebook/convnext-large-224-22k-1k | 2022-01 | 1536 | Liu et al. (2022b) |
| timm/eva02_large_patch14_clip_224.merged2b | 2023-03 | 1024 | Sun et al. (2023) |
| google/siglip2-so400m-patch14-384 | 2023-03† | 1152 | Tschannen et al. (2025) |
| *held-out* | | | |
| apple/aimv2-huge-patch14-224 | 2024-11 | 1536 | Fini et al. (2025) |
| timm/vit_pe_core_large_patch14_336.fb | 2025-04 | 1024 | Bolya et al. (2025) |
| facebook/dinov3-vitl16-pretrain-lvd1689m | 2025-08 | 1024 | Siméoni et al. (2025) |
| Qwen/Qwen3-VL-Embedding-8B | 2026-01 | 4096 | Li et al. (2026) |
| jinaai/jina-embeddings-v5-omni-small | 2026-05 | 1024 | Hönicke et al. (2026) |
| *Audio* (seen) | | | |
| s3prl/vq_wav2vec* | 2019-10 | 512 | Baevski et al. (2020a) |
| facebook/wav2vec2-large | 2020-06 | 1024 | Baevski et al. (2020b) |
| s3prl/tera* | 2020-07 | 768 | Liu et al. (2021) |
| MIT/ast-finetuned-audioset-10-10-0.4593 | 2021-04 | 768 | Gong et al. (2021) |
| facebook/hubert-large-ll60k | 2021-06 | 1024 | Hsu et al. (2021) |
| Simon-Kotchou/ssast-base-patch-audioset-16-16 | 2021-10 | 768 | Gong et al. (2022) |
| microsoft/wavlm-large | 2021-10 | 1024 | Chen et al. (2022) |
| facebook/wav2vec2-xls-r-300m | 2021-11 | 1024 | Babu et al. (2022) |
| facebook/data2vec-audio-large | 2022-02 | 1024 | Baevski et al. (2022) |
| facebook/wav2vec2-conformer-rel-pos-large | 2022-06 | 1024 | Wang et al. (2020) |
| laion/larger_clap_general | 2022-11 | 512 | Wu et al. (2023) |
| microsoft/unilm/BEATs_iter3_plus_AS2M* | 2022-12 | 768 | Chen et al. (2023) |
| openai/whisper-large-v3 | 2022-12 | 1280 | Radford et al. (2023) |
| worstchan/EAT-base_epoch30_pretrain | 2024-01 | 768 | Chen et al. (2024) |
| mispeech/dasheng-0.6B | 2024-06 | 1280 | Dinkel et al. (2024) |
| *held-out* | | | |
| tsinghua-ee/WAVE-7B | 2025-09 | 3584 | Tang et al. (2026) |
| LCO-Embedding/LCO-Embedding-Omni-7B | 2025-10 | 3584 | Xiao et al. (2025a) |
| nvidia/omni-embed-nemotron-3b | 2025-10 | 2048 | Xu et al. (2025) |
| jinaai/jina-embeddings-v5-omni-small | 2026-05 | 1024 | Hönicke et al. (2026) |
| MIT-SLS/USAD2-Large | 2026-06 | 1024 | Chang et al. (2026) |

# B RepICL Meta-Training Details

Table 8 lists the hyperparameters of RepICL; the whitening and architecture follow Section 5. Each fold is meta-trained once on a single NVIDIA RTX 4090; for the audio fold this takes about 31 hours for RepICL-I and about 45 hours for RepICL-T.

Table 8: Hyperparameters of RepICL-I and RepICL-T.

| | RepICL-I | RepICL-T |
|---|---|---|
| *Episode and whitening* | | |
| Way / shot / queries per class | 5 / 5 / 10 | |
| Fit set $\mathcal{A}$ | $\mathcal{Z}_\mathcal{S}$ | $\mathcal{Z}_\mathcal{S} \cup \mathcal{Z}_\mathcal{Q}$ |
| PCA rank $r_\mathcal{A}$ | 24 | 74 |
| Standardization $\varepsilon$ | $10^{-5}$ | |
| *Architecture* | | |
| Layers / heads | 24 / 16 | |
| Token dimension (feature + label) | 1024 (768 + 256) | |
| MLP dimension | 4096 | |
| Attention mask $\boldsymbol{M}$ | queries masked | none |
| Readout temperature $\tau$ | 0.07 | |
| Parameters | ≈303M | |
| Initialization | CAML-CLIP | |
| *Meta-training* | | |
| Optimizer | AdamW | |
| Learning rate / warmup steps | $10^{-4}$ / 2,500 | |
| Weight decay | 0.01 | |
| Optimizer steps | 50,000 | |
| Episodes per step / microbatch | 1,024 / 128 | |
| Input noise $\sigma$ | 0.07 | |
| EMA decay | 0.999 | |
| Precision | bf16 | |
| Seed | 42 | |

**Task sampling.** Each fold meta-trains on the training tasks of its two non-held-out modalities. Each optimizer step draws 1,024 training tasks. The two modalities receive equal probability; within a modality, a dataset $\mathcal{D}$ is drawn with probability proportional to $\sqrt{\tilde{n}_\mathcal{D}}$, where $\tilde{n}_\mathcal{D}$ is its training-pool size clipped to $[500,\ 200{,}000]$; and the encoders paired with $\mathcal{D}$ are equally likely. A task $(\mathcal{D}, E)$ of modality $m$ is thus drawn with probability

$$p(\mathcal{D}, E) = \frac{1}{2} \cdot \frac{\sqrt{\tilde{n}_\mathcal{D}}}{\sum_{\mathcal{D}' \in \mathcal{D}_m} \sqrt{\tilde{n}_{\mathcal{D}'}}} \cdot \frac{1}{|\mathcal{E}_D|},$$

where $\mathcal{D}_m$ is the set of training datasets of $m$ and $\mathcal{E}_\mathcal{D}$ the set of encoders paired with $\mathcal{D}$. The square root favors larger datasets without letting them dominate. Each drawn task yields one episode, whose class labels are randomly permuted.

**Optimization.** We minimize $\mathcal{L}(\theta)$ of Section 5.3 with AdamW (learning rate $10^{-4}$, weight decay 0.01) for 50,000 steps; the learning rate warms up linearly over the first 2,500 steps and then stays constant. The 1,024 episodes of a step are processed as 8 accumulated microbatches of 128. As a simple input augmentation against overfitting to the training tasks, Gaussian noise with standard deviation $\sigma = 0.07$ is added to the $r_\mathcal{A}$ active coordinates of each whitened token during training. Model weights are kept in bf16, while the whitening statistics are computed in fp32.

**Weight averaging and checkpoint selection.** To smooth the step-to-step noise of episodic updates, we keep an exponential moving average $\bar{\theta}$ of the weights (Polyak & Juditsky, 1992; Morales-Brotons et al., 2024) with decay 0.999. Over the first 1,000 steps $t$, the decay is lowered to $(1 + t)/(10 + t)$ so that the initialization does not dominate the average. We evaluate $\bar{\theta}$ on the validation tasks every 2,500 steps and keep the checkpoint with the highest validation accuracy, which is used for every reported result.

# C Comparison Methods

For all methods other than RepICL, we use the raw frozen representation without episodic whitening.

## C.1 Classical Classifiers Fitted on the Support Set

Logistic regression and an RBF-kernel SVM serve as a linear and a nonlinear classifier fitted on the support set of each episode. Both use scikit-learn 1.6.1 (Pedregosa et al., 2011) at its defaults after StandardScaler normalization fitted on the support set, except that logistic regression runs for up to 1,000 iterations.

## C.2 Few-Shot Methods on Frozen Representations

**ProtoHead.** ProtoHead is the variant of prototypical networks (Snell et al., 2017) on a frozen backbone used by Woerner et al. (2026): each query is assigned to the class whose mean $L_2$-normalized support representation is nearest.

**SimpleShot.** SimpleShot (Wang et al., 2019) applies nearest-centroid classification after a feature transformation. Its strongest variant, CL2N, centers with the mean of the encoder's base-class training features, which are unavailable for our encoders; it therefore centers with the support mean, $L_2$-normalizes, and scores by cosine similarity to the class means.

**LaplacianShot and $\alpha$-TIM.** LaplacianShot (Ziko et al., 2020) refines nearest-prototype predictions with a Laplacian term that encourages nearby queries to share a label, and $\alpha$-TIM (Veilleux et al., 2021) maximizes the mutual information between queries and their predictions with an $\alpha$-divergence suited to class-imbalanced query sets. LaplacianShot uses easyfsl 1.5.0 (Bennequin, 2021) at its defaults, and $\alpha$-TIM uses $\alpha = 7$, temperature 15, and 1,000 gradient steps with learning rate $10^{-4}$. Both take $L_2$-normalized inputs.

**PT-MAP.** PT-MAP (Hu et al., 2021) applies a power transform to the features and then estimates the class centers jointly from the support and query sets with an optimal-transport-based MAP procedure. It follows the reference implementation, whose power transform $x^{1/2}$ assumes the nonnegative features of a post-ReLU backbone; since our representations are signed, it uses $\mathrm{sign}(x)\,|x|^{1/2}$.

## C.3 In-Context Learners

**TAIL.** TAIL (Woerner et al., 2026) is a transformer-based algorithm-implicit meta-learner that operates across tasks with varying domains, modalities, and label configurations. Its checkpoint is not released, so TAIL uses our reproduction, with inputs rescaled to norm 13 (Appendix C.4).

**Tabular foundation models.** Tabular foundation models are pretrained on large collections of synthetic tabular datasets and predict the labels of an unseen table in context, without gradient updates. TabPFN v2.6 (Hollmann et al., 2025) from tabpfn 7.0.1 and TabICL v2 (Qu et al., 2026) from tabicl 2.0.2 take each support representation as a row of tabular features with its label as the target, and each query representation as a test row. TabPFN uses its `ignore_pretraining_limits` option to accept representations with more than 2,000 dimensions.

## C.4 TAIL Reproduction and Adaptation

**Reproduction.** TAIL (Woerner et al., 2026) releases code but no checkpoint, so we train the model for 500 epochs on the CLIP ViT-H features of ImageNet-1k, Meta-Album, and MedIMeta, following their released recipe. Table 9 compares our reproduction against the published numbers under TAIL's own task taxonomy; the two agree closely across all settings. One limitation: TAIL excludes a subset of classes from some datasets at evaluation time, but the released codebase does not implement this filtering, so our reproduction omits it and we cannot guarantee an identical evaluation protocol to theirs. As Table 9 shows, however, the resulting gap remains small across all settings. All TAIL results reported in this paper, including on TAIL's own evaluation datasets, use this reproduced checkpoint.

Table 9: TAIL paper-reported and reproduced macro accuracy (%).

| | 1-shot | | | 5-shot | | | |
|---|---|---|---|---|---|---|---|
| Setting | Paper | Reprod. | Gap | Paper | Reprod. | Gap | Per-dataset MAE |
| In-domain (4 datasets) | 90.10 | 89.73 | -0.37 | 95.66 | 95.10 | -0.56 | 1.03 |
| Cross-modal (2 datasets) | 62.75 | 64.31 | +1.56 | 72.48 | 72.53 | +0.05 | 1.00 |
| Cross-domain (8 datasets) | 72.46 | 73.25 | +0.79 | 82.32 | 82.93 | +0.61 | 3.87 |

**Transfer to RepShiftBench.** When we evaluate TAIL directly on RepShiftBench, it collapses to near-random accuracy on most individual tasks (Table 10). Feature scale accounts for a substantial part of this loss. TAIL is trained on unnormalized CLIP features whose $L_2$ norm is about 13, whereas our representations are preprocessed to unit norm. Rescaling each embedding to a shared target norm recovers most of the loss, and a grid search on the DINOv3, USAD2, and E5-large-v2 tasks shows that performance peaks at a target norm of 13 for each encoder, close to the norm of TAIL's own training features (Table 11). The table also reports each encoder's original pre-normalization norm and TAIL accuracy on freshly re-encoded raw features; the latter preserves row-to-row magnitude variation and is distinct from a fixed target norm. We call this rescaling the scale calibration. Another limitation is TAIL's model design, which accepts at most the 1,280 dimensions it was trained on, so we pass wider representations through a fixed Gaussian projection, one untrained matrix per representation dimension. Table 12 separates each setting into the tasks that fit TAIL natively and those that pass through the projection. The gap between the two bands is large, but it is the same gap for RepICL-T, which takes every dimension natively, and the two agree in sign in all adaptation settings. The similar pattern for RepICL-T suggests that the projection is unlikely to be the primary explanation for TAIL's performance gap. We therefore apply the scale calibration and the projection to every TAIL result on RepShiftBench.

Table 10: TAIL variants and RepICL-T on RepShiftBench. TAIL (unit-norm): raw unit-norm representations. TAIL (norm 13): scale-calibrated to norm 13. TAIL$^\star$: retrained on our folds' training tasks. RepICL-T is shown for reference. Bold marks the best TAIL variant per column.

| | **Text-held** | | | | **Audio-held** | | | | **Image-held** | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Method** | D | E | J | M | D | E | J | M | D | E | J | M |
| TAIL (unit-norm) | 26.86 | 34.48 | 29.01 | 28.54 | 30.34 | 34.67 | 32.42 | 24.27 | 25.82 | 29.15 | 26.14 | 38.22 |
| TAIL (norm 13) | **52.79** | **71.96** | **61.05** | **62.23** | **66.06** | **71.13** | **71.07** | **45.67** | **50.33** | **64.93** | **55.55** | **79.31** |
| TAIL$^\star$ | 36.96 | 48.48 | 38.86 | 33.07 | 55.43 | 62.33 | 60.21 | 36.75 | 21.81 | 20.16 | 20.08 | 20.25 |
| RepICL-T | 63.55 | 77.99 | 68.29 | 73.26 | 75.27 | 78.13 | 77.47 | 55.62 | 61.79 | 76.58 | 65.02 | 85.69 |

**Training TAIL on RepShiftBench.** The scale calibration and projection correct the representation mismatch, but TAIL's training tasks still differ from RepICL's. RepICL is meta-trained on the standard tasks of each fold, so for a fair comparison we also retrain TAIL using RepShiftBench's meta-training setup and write the model as TAIL$^\star$. The results in Table 10 show that matching the training setup does not help. TAIL$^\star$ loses to both the scale-calibrated TAIL and RepICL-T. The main results therefore use the scale-calibrated TAIL. One limit bounds this reading. We transfer TAIL's published hyperparameters rather than searching over them, so a tuned TAIL$^\star$ may score higher.

Table 11: TAIL accuracy (%) across input scales. "Raw" uses unnormalized features; numeric columns indicate target norms. "Raw norm" is the mean pre-$L_2$ embedding norm.

| | | | | TAIL input scale | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Encoder | Modality | # tasks | Raw norm | Raw | 1 | 3 | 6 | 13 | 26 | 52 | Ours |
| DINOv3 | image | 19 | 15.61 | 87.50 | 50.65 | 83.50 | 87.00 | **87.62** | 84.75 | 63.15 | 92.34 |
| USAD2 | audio | 21 | 163.86 | 6.59 | 21.98 | 31.21 | 44.85 | **51.00** | 40.25 | 23.77 | 63.94 |
| E5-large-v2 | text | 19 | 23.03 | 65.48 | 24.40 | 49.02 | 66.94 | **72.44** | 61.37 | 29.09 | 82.82 |

Table 12: Accuracy (%) split by input width: native is at most 1,280 dimensions, where TAIL fits without modification; proj. covers dimensions larger than 1,280, passed through the fixed Gaussian matrix.

| | **Text-held** | | | | **Audio-held** | | | | **Image-held** | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Width** | D | E | J | M | D | E | J | M | D | E | J | M |
| **TAIL (norm 13)** | | | | | | | | | | | | |
| native | 49.58 | 76.94 | 68.16 | 57.94 | 63.47 | 86.18 | 86.36 | 42.40 | 46.61 | 64.72 | 55.74 | 82.01 |
| proj. | 74.80 | 66.91 | 54.67 | 67.29 | 70.40 | 60.81 | 61.78 | 64.20 | 67.89 | 65.02 | 55.46 | 75.02 |
| $\Delta$ | +25.22 | −10.03 | −13.49 | +9.35 | +6.93 | −25.37 | −24.58 | +21.80 | +21.28 | +0.30 | −0.28 | −6.99 |
| **RepICL-T** | | | | | | | | | | | | |
| native | 60.61 | 83.72 | 75.92 | 68.64 | 73.48 | 90.21 | 91.14 | 53.11 | 58.23 | 75.55 | 65.07 | 89.34 |
| proj. | 83.62 | 72.18 | 61.45 | 78.71 | 78.28 | 69.84 | 69.18 | 69.84 | 78.51 | 77.01 | 64.98 | 79.89 |
| $\Delta$ | +23.01 | −11.54 | −14.47 | +10.07 | +4.80 | −20.37 | −21.96 | +16.73 | +20.28 | +1.46 | −0.09 | −9.45 |

# D Detailed Results of the Main Paper

## D.1 Per-Dataset Results under the TAIL Protocol

Table 13 gives the detailed results summarized in Section 6.2. The evaluation follows TAIL's setup, with CLIP features and 1,000 identical 5-way 5-shot episodes per dataset. Both RepICL variants use the image-held-out checkpoint, so the 11 datasets and the CLIP image encoder are unseen during meta-training, whereas the CLIP text encoder is among the training encoders. CAML and GPICL are included through their published results, since their released formulations do not support the heterogeneous input dimensions of RepShiftBench. Both TAIL and our main experiments include a linear baseline, but the two fit it differently. The table therefore reports three Linear columns: the numbers published by TAIL, our reproduction of the TAIL recipe, and our logistic regression. TAIL fits an unregularized linear layer per episode with 100 epochs of Adam, while ours fits a standardized, L2-regularized logistic regression. TAIL's recipe is therefore not necessarily the strongest possible per-episode linear classifier. RepICL-T outperforms every previously reported result on average, reaching 92.31% against TAIL's own best reported number of 90.05%.

## D.2 Full Ablation Results

Table 14 gives the detailed results of the ablation study in Section 6.3 for every fold and adaptation setting. For meta-training diversity, we ablate the set of meta-training tasks instead of a model component, to see how performance depends on the diversity of that set. Each fold meta-trains on two modalities, and we restrict the task pool $\mathcal{T}_{\text{train}}$ within each of them, keeping either one dataset with all its compatible encoders or one encoder with all its compatible datasets. Both settings are trained for the same 50,000 steps as the full model.

With limited task diversity, the choice of dataset or encoder matters. If the selected dataset is one on which few-shot classification is hard, its episodes may provide only a weak training signal, and the resulting drop would confound the effect of diversity with the difficulty of that dataset. The same applies to the selected encoder. We therefore select the source favorably. In each training modality, we choose, among the training datasets, the dataset with the highest Logistic Regression accuracy averaged over its compatible encoders, and, among the training encoders, the encoder

Table 13: Accuracy (%) on the visual datasets of TAIL. † and the published columns are values reported by Woerner et al. (2026). Bold marks the best result per row, excluding OCT (4-way).

| | Linear | | | | | TAIL | | Ours | |
|---|---|---|---|---|---|---|---|---|---|
| Dataset | published | TAIL recipe | ours | GPICL† | CAML† | published | reproduced | RepICL-I | RepICL-T |
| CIFAR-FS | 91.86 | 91.56 | 92.87 | 91.20 | 91.69 | 94.55 | 94.49 | 93.75 | **96.75** |
| miniImageNet | 97.65 | 97.66 | 98.56 | 99.44 | 99.29 | 99.63 | 99.40 | 98.71 | **99.77** |
| tieredImageNet | 95.30 | 95.12 | 96.50 | 98.18 | 97.98 | **98.67** | 98.05 | 96.72 | 98.43 |
| Pascal VOC | 83.57 | 83.49 | 85.70 | 87.46 | 87.87 | **89.78** | 88.33 | 86.13 | 89.55 |
| Aircraft | 92.12 | 91.82 | 93.65 | 90.83 | 93.09 | 95.01 | 94.31 | 94.12 | **96.66** |
| CUB | 95.17 | 94.77 | 96.81 | 96.02 | 97.55 | 98.51 | 97.52 | 97.41 | **98.93** |
| meta-iNat | 92.91 | 92.91 | 94.78 | 95.56 | 96.20 | **97.70** | 96.77 | 95.37 | 97.66 |
| tiered meta-iNat | 88.32 | 88.80 | 92.38 | 91.15 | 93.53 | 95.69 | 94.24 | 93.41 | **96.56** |
| Paintings | 66.21 | 66.69 | 68.85 | 54.29 | 58.72 | 63.03 | 73.39 | 69.93 | **73.82** |
| Pascal-Paintings | 71.62 | 71.10 | 74.78 | 65.79 | 70.13 | 73.04 | 79.82 | 76.34 | **81.81** |
| PBC | 68.39 | 68.01 | 76.67 | 53.28 | 80.19 | 84.96 | 78.90 | 79.13 | **85.41** |
| Mean | 85.74 | 85.63 | 88.32 | 83.93 | 87.84 | 90.05 | 90.47 | 89.18 | **92.31** |
| OCT (4-way) | – | 49.82 | 54.64 | – | – | – | 47.94 | 54.07 | 55.94 |

with the highest accuracy averaged over its compatible datasets. For text, image, and audio, the selected datasets are HWU64, Oxford Flowers, and ESC-50, and the selected encoders are BGE-ICL, EVA02-CLIP-L, and Whisper-large-v3.

This selection has two limitations. First, it uses Logistic Regression accuracy on the test split, which may favor the restricted settings. Second, because the restriction is applied within each modality, every restricted setting still contains two datasets or two encoders. The settings therefore reduce the diversity of the meta-training pool rather than remove it, and we do not test more restricted pools, such as a single dataset or encoder in total.

Table 14: Detailed results of the ablation study.

| | **Text-held-out fold** | | | | **Audio-held-out fold** | | | | **Image-held-out fold** | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | D | E | J | M | D | E | J | M | D | E | J | M |
| **RepICL-I** | 61.41 | 75.85 | 66.22 | 70.10 | 73.52 | 76.79 | 75.86 | 55.14 | 59.19 | 72.82 | 62.56 | 83.26 |
| *Episodic whitening* | | | | | | | | | | | | |
| w/o Coord. Standardize | −8.18 | −3.64 | −5.19 | −7.88 | −5.03 | −4.14 | −3.40 | −9.38 | −7.98 | −6.53 | −5.85 | −3.25 |
| w/o PCA, trainable proj. | −5.28 | −5.94 | −7.15 | −12.07 | −8.61 | −11.38 | −10.76 | −9.93 | −5.25 | −12.39 | −10.34 | −6.55 |
| w/o PCA, fixed proj. | −5.44 | −6.05 | −8.37 | −10.96 | −8.33 | −10.60 | −10.06 | −10.01 | −5.33 | −11.88 | −9.96 | −6.42 |
| *Model structure* | | | | | | | | | | | | |
| w/o CAML initialization | −0.08 | −0.01 | **+0.25** | −0.05 | −0.16 | −0.25 | −0.13 | −0.04 | **+0.02** | −0.10 | **+0.22** | **+0.02** |
| fixed five-class head | −0.12 | −0.07 | **+0.38** | −0.14 | −0.28 | −0.45 | −0.14 | −0.04 | −0.22 | −0.45 | **+0.14** | −0.05 |
| *Meta-training diversity* | | | | | | | | | | | | |
| one dataset per modality | −0.37 | −0.34 | **+0.03** | −0.74 | −0.40 | −0.32 | −0.14 | −0.32 | −0.54 | −0.75 | −0.26 | −0.21 |
| one encoder per modality | −0.79 | −0.27 | −0.03 | −0.41 | −0.99 | −1.03 | −0.54 | −0.76 | −0.83 | −0.80 | −0.33 | −0.29 |
| **RepICL-T** | 63.55 | 77.99 | 68.29 | 73.26 | 75.27 | 78.13 | 77.47 | 55.62 | 61.79 | 76.58 | 65.02 | 85.69 |
| *Episodic whitening* | | | | | | | | | | | | |
| w/o Coord. Standardize | −13.64 | −6.82 | −9.37 | −9.81 | −6.60 | −6.04 | −6.05 | −14.34 | −14.00 | −15.24 | −13.93 | −6.58 |
| w/o PCA, trainable proj. | −7.30 | −8.02 | −9.45 | −15.15 | −8.35 | −11.21 | −10.51 | −9.91 | −7.01 | −15.61 | −12.45 | −8.59 |
| w/o PCA, fixed proj. | −6.89 | −7.81 | −9.16 | −14.84 | −9.40 | −12.03 | −11.14 | −10.10 | −7.77 | −16.09 | −12.78 | −8.78 |
| *Model structure* | | | | | | | | | | | | |
| w/o CAML initialization | +0.00 | **+0.13** | −0.01 | **+0.04** | **+1.30** | **+1.04** | **+0.85** | **+1.53** | −0.28 | −0.11 | −0.20 | −0.17 |
| fixed five-class head | **+0.83** | **+0.53** | **+0.75** | **+0.76** | **+1.89** | **+1.43** | **+1.34** | **+2.44** | **+0.32** | **+0.78** | **+0.60** | **+0.35** |
| *Meta-training diversity* | | | | | | | | | | | | |
| one dataset per modality | −2.02 | −1.51 | −1.67 | −2.55 | −0.87 | −0.88 | −0.75 | −1.02 | −2.85 | −3.52 | −2.86 | −1.50 |
| one encoder per modality | −2.43 | −1.04 | −1.22 | −1.39 | −2.09 | −2.53 | −2.02 | −2.01 | −2.67 | −3.47 | −2.77 | −1.16 |

# E Additional Analyses and Experiments

## E.1 Linear vs. Previous Few-Shot Methods

In our Table 1 results on RepShiftBench, logistic regression outperforms almost every few-shot method, with PT-MAP the only method ahead of it, and only on the image-modality shift. Classical few-shot methods such as $\alpha$-TIM and PT-MAP were developed and tuned on CNN backbones (ResNet, WideResNet), and usually outperform linear methods there. We therefore ask whether the

gap we observe here reflects a mismatch between the encoders these methods were designed for and those evaluated on RepShiftBench, rather than evidence that the earlier methods are weaker than logistic regression.

**Matched CNN backbones.** On the exact ResNet-18, ResNet-34, and WRN-28-10 backbones used in the few-shot literature, evaluated under our benchmark's 5-way 5-shot image task, LaplacianShot, $\alpha$-TIM, and PT-MAP are all ahead of logistic regression on the same frozen features (Table 15). SimpleShot is the exception and is roughly on par, because our setting gives it no base-class-supervised training to draw its centering statistics from. On these matched backbones, previous few-shot methods do outperform logistic regression.

Table 15: Previous few-shot methods vs. logistic regression on CNN backbones. Accuracy (%).

| Backbone | Logistic regression | SimpleShot | LaplacianShot | $\alpha$-TIM | PT-MAP (signed-power) |
|---|---|---|---|---|---|
| ResNet-18 | 77.17 | 77.05 | 77.88 | 77.83 | **80.47** |
| ResNet-34 | 76.75 | 76.71 | 77.20 | 77.39 | **79.11** |
| WRN-28-10 | 58.68 | 59.20 | 60.30 | 62.12 | **65.18** |

**The benchmark's own encoder set.** We then run the same comparison over every encoder of RepShiftBench, in all three modalities (Table 16). Across the benchmark's encoder classes, the advantage is heterogeneous: previous few-shot methods lead on several classes, whereas Logistic Regression leads on many others and on text and audio overall. Thus, no previous few-shot method consistently dominates across the full range of encoder families and modalities. These methods were therefore not designed with transfer across encoder families in mind, and none of them holds up across the full set. That is the gap RepICL is trained to close.

Table 16: Logistic regression vs. previous few-shot methods on RepShiftBench, grouped by encoder representation class. Bold marks the best method in each row.

| Modality | Encoder class | Num. of Enc. | Logistic regression | SimpleShot | LaplacianShot | $\alpha$-TIM | PT-MAP |
|---|---|---|---|---|---|---|---|
| Image | CNN-like | 7 | 80.44 | 80.96 | 81.42 | 81.43 | **83.23** |
| | ViT / vision-SSL | 6 | **76.82** | 73.18 | 72.67 | 75.45 | 74.93 |
| | CLIP-like contrastive | 3 | 92.50 | 91.33 | 91.37 | 92.20 | **93.09** |
| | Multimodal embedding | 2 | 90.65 | 90.22 | 90.31 | 90.50 | **91.35** |
| | *All* | 18 | 82.38 | 81.12 | 81.15 | 82.24 | **83.01** |
| Text | Static word embeddings | 2 | **61.35** | 55.28 | 55.30 | 59.41 | 58.92 |
| | Pretrained LM hidden states | 4 | **50.62** | 38.50 | 35.45 | 45.86 | 44.07 |
| | Purpose-built text embeddings | 7 | 79.74 | 80.01 | 81.54 | 78.19 | **83.34** |
| | LLM/instruction embeddings | 4 | 79.95 | 80.51 | 81.48 | 76.98 | **83.15** |
| | LLM hidden-state features | 5 | **66.92** | 57.86 | 55.36 | 62.00 | 61.42 |
| | Vision-language contrastive | 2 | **66.56** | 60.79 | 60.17 | 62.92 | 65.18 |
| | *All* | 24 | **69.62** | 64.90 | 64.42 | 66.39 | 68.65 |
| Audio | Speech-pretrained / ASR | 9 | **49.53** | 42.92 | 40.74 | 47.33 | 42.81 |
| | General-audio SSL / event models | 6 | **53.39** | 47.00 | 45.39 | 51.52 | 47.57 |
| | Audio-text contrastive | 1 | 56.41 | 55.76 | 56.01 | 56.13 | **56.65** |
| | Audio/multimodal LLMs | 4 | **66.67** | 64.17 | 63.91 | 65.31 | 65.44 |
| | *All* | 20 | **54.46** | 49.04 | 47.53 | 52.62 | 49.46 |

### E.2 LoRA Comparison

RepICL adapts to a task in context, without updating any weights. An alternative is to update the encoder on the support set at test time, a paradigm studied for few-shot classification with adapters (Li et al., 2022), full-backbone fine-tuning (Hu et al., 2022b), cosine classifiers (Chen et al., 2019), and broader PEFT sweeps (Basu et al., 2024). We compare against LoRA (Hu et al., 2022a), one of the most widely used standard PEFT techniques, which is also a strong few-shot baseline (Basu et al., 2024; Zanella & Ben Ayed, 2024).

**Setup.** For each episode we re-initialize LoRA with rank 8, alpha 16 on the q/v attention projections and fit it on the support set with a prototype mean-cosine loss, similar to the LoRA + ProtoNCC

recipe of Basu et al. (2024). For whisper-large and SigLIP-2 we tune only the encoder used to extract features. All tasks use AdamW with learning rate $3 \times 10^{-4}$ and 50 steps, selected on held-out episodes of a few datasets per modality with one encoder per modality. We compare RepICL-I and RepICL-T with LoRA, with logistic regression on the frozen features, and with a frozen baseline that classifies queries by cosine similarity to class prototypes of the unadapted encoder output. Due to computation limits, we evaluate 9 encoders (3 per modality) from RepShiftBench, chosen to cover different encoder sizes, on 30 sealed episodes per task disjoint from the selection episodes. Average time is measured on a single RTX 4090 over two 5-way 5-shot episodes, on 6 of the 9 encoders.

Table 17: Performance compared to LoRA.

| Encoder | Modality | Params | Tasks | Frozen | LoRA | RepICL-I | RepICL-T | Logistic regression |
|---|---|---|---|---|---|---|---|---|
| AST audioset-base | audio | 86.2M | 20 | 46.6 | 55.7 | 55.8 | **56.5** | 54.8 |
| HuBERT-large | audio | 315.4M | 20 | 46.9 | 51.1 | 57.2 | **57.7** | 56.3 |
| whisper-large | audio | 637.0M | 20 | 40.6 | 61.5 | 64.4 | **64.8** | 63.5 |
| all-mpnet-base-v2 | text | 109.5M | 19 | 75.2 | 79.5 | 77.17 | **80.72** | 77.16 |
| embedding-gemma-300m | text | 302.9M | 19 | 78.2 | 82.3 | 80.95 | **84.25** | 80.99 |
| stella-1.5B-v5 | text | 1.54B | 19 | 80.1 | **85.4** | 81.17 | 84.01 | 80.19 |
| DINOv3 ViT-L/16 | image | 303.1M | 19 | 85.9 | 91.5 | 90.29 | **92.34** | 88.33 |
| SigLIP-2 so400m/384 | image | 428.2M | 19 | 88.7 | 90.9 | 93.08 | **94.73** | 91.13 |
| PE-core-L | image | 317.2M | 19 | 88.2 | 92.9 | 92.94 | **94.85** | 90.27 |
| Average over encoders | | | | 70.0 | 76.8 | 77.0 | **78.9** | 75.9 |
| Avg. time (ms/query) | | | | 7.67 | 418.90 | 7.81 | 7.92 | 7.75 |

**Results.** LoRA consistently improves over the frozen encoder on all nine encoders (Table 17), confirming the benefit of test-time adaptation. RepICL-I performs on par with LoRA, and RepICL-T outperforms it on almost every encoder. Since RepICL requires no test-time parameter update, it is also 33–68$\times$ faster per query. Overall, RepICL-I matches LoRA at a fraction of the test-time cost, and RepICL-T further outperforms it.

### E.3 Inference Cost

We compare the per-episode inference time of RepICL with the other methods to assess whether it is efficient enough for practical use. For each method, we time one 5-way 5-shot episode with 25 support and 50 query examples on frozen representations, from normalization to prediction. RepICL, TAIL, TabPFN, and TabICL run on one NVIDIA RTX 4090, and the remaining methods run on the CPU. All methods are timed under the same protocol and none is specifically optimized for speed.

Figure 3 shows that RepICL runs in a time comparable to TAIL and to Logistic Regression on the CPU, and is much faster than TabPFN and TabICL. Its median time per episode is 5.3 ms for RepICL-I and 5.9 ms for RepICL-T, against 6.7 ms for Logistic Regression, 0.7 s for TabPFN, and 1.5 s for TabICL. CPU and GPU times are not directly comparable, so these numbers indicate the order of magnitude of each method's inference time rather than a ranking.

The encoders in RepShiftBench produce representations of different widths, so we also examine how inference time depends on the representation width $D$. As Figure 4 shows, the inference time of RepICL is nearly independent of $D$, staying between 4.6 and 5.7 ms as $D$ grows from 512 to 4096, because only the episodic whitening operates in the $D$-dimensional space and the transformer always receives the same number of tokens.

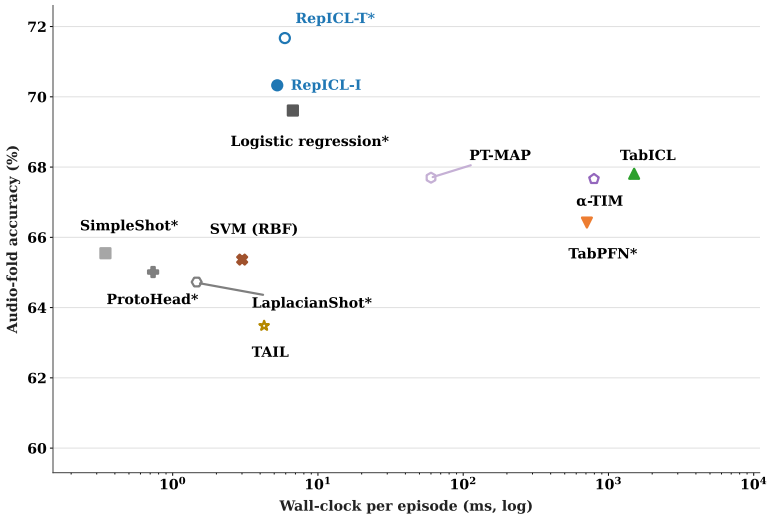



Figure 3: Accuracy against inference time per episode. Accuracy is the audio-fold mean from Table 1; hollow markers denote transductive methods. Time is the median over 22 episodes, one for each distinct representation width in the three folds. Methods marked with * show a run-to-run timing variation above 5% on some episodes.

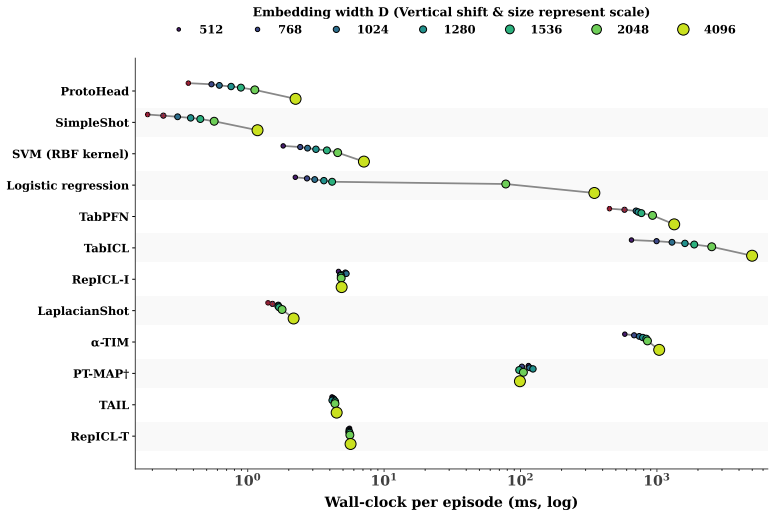



Figure 4: Inference time per episode as the representation width $D$ grows from 512 to 4096. PT-MAP$^{\dagger}$ is the median over 128 episodes, since its number of Sinkhorn iterations depends on the episode.