MetaFood Training-Free Cultural Generalization in Food Recognition via Knowledge-Grounded Prompt Expansion

János Horváth
Visionary Tech & Event Solutions
USA, California, Sacramento
horvath_janos@visionarytecheventsolutions.com

Abstract

*Food recognition systems for dietary logging and recipe retrieval can fail in zero-shot settings on underrepresented cuisines, in part because dish names include transliterations and aliases that are weakly represented in pretraining. We study a **training-free** pipeline for food recognition under cultural naming variation with a frozen contrastive vision–language backbone, combining prompt expansion, alias matching, and optional transductive refinement. We evaluate on two cuisine-focused datasets. Our transductive variant assumes access to the unlabeled test batch.*

MetaFood builds a per-class text bank *from food-aware prompts, conservative lexical variants, and optional alias expansion from a public dish knowledge base (FoodKB), whose coverage is partial, then pools variant logits and optionally refines frozen image embeddings. Across both benchmarks, it improves accuracy or calibration without fine-tuning or per-dataset tuning, and we provide per-class diagnostics showing that failures concentrate in KB-unmatched classes.*

1. Introduction

Food is a hard case for zero-shot recognition because dish names vary across regions and communities: visually similar dishes can have different names, and the same dish can be written with multiple transliterations or aliases. When class names fall outside the vocabulary emphasized during web-scale pretraining, frozen vision–language models can appear "almost right" visually yet fail at culturally specific labels. Cuisine-focused food recognition is therefore a useful stress test for open-vocabulary recognition and vision–language grounding.

Contrastive vision–language models (VLMs) enable zero-shot classification by comparing an image embedding with text embeddings of class prompts [8, 36, 43]. On common categories, careful prompting can be competitive with supervised models, but food benchmarks expose two coupled distribution shifts: (i) **visual shift** (ingredients, plating conventions, photography styles) and (ii) **linguistic shift** in *label surface forms* (regional names, spelling variants, aliases). The second shift is often treated as an implementation detail, yet small edits to a class name can materially change results [69, 70]. In downstream applications such as dietary logging, over-confident cross-cuisine mistakes are particularly undesirable [49].

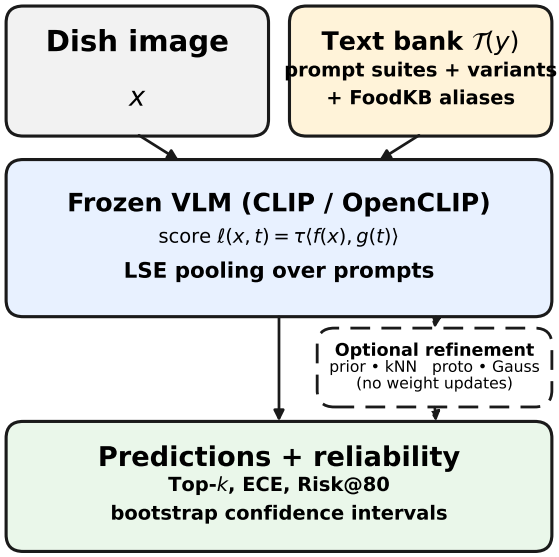


Figure 1. **Teaser.** MetaFood is *strictly training-free*: a frozen vision–language model scores a dish image against a richer *text bank* (food prompts, lexical variants, and optional FoodKB aliases), then applies lightweight post-hoc refinement on frozen embeddings (prior correction, kNN smoothing, prototype/Gaussian alignment) to improve accuracy and reliability under cultural shift.

Training-free setting. We consider a practical scenario aligned with MetaFood: a researcher has a public cuisine dataset and a frozen VLM, but cannot collect new labels or fine-tune. The goal is to improve and audit zero-shot recognition using only dataset metadata (class strings) and public text resources. Because cuisine datasets can be small and ambiguous, we emphasize *reliability*—calibration and selective prediction—in addition to Top-k accuracy [15, 17, 34]. Our method is evaluated on two cuisine-focused datasets, and the optional transductive variant assumes access to the unlabeled test set.

Overview. We propose MetaFood, a training-free pipeline with two practical components. First, we treat class-name design as a first-class component: for each label we construct a *text bank* combining food-specific prompt templates, conservative lexical variants, and (when possible) aliases retrieved from a lightweight dish knowledge base (FoodKB). We pool the resulting logits with log-sum-exp to reduce sensitivity to any single phrasing. Second, when unlabeled test images are available, we apply deterministic, gradient-free transductive refinements on frozen embeddings (prior correction, kNN smoothing, and prototype/Gaussian alignment), without updating backbone weights. Fig. 1 provides a compact schematic.

Why this matters. MetaFood-style systems often chain modules (dish naming $\rightarrow$ recipe retrieval $\rightarrow$ nutrition estimates), so brittle dish naming under cultural shift can propagate downstream. Cuisine datasets are also a natural place to evaluate uncertainty: multiple names may be acceptable, labels can be noisy, and a safe system should *know when it does not know*. By focusing on strictly training-free interventions, we avoid confounding gains from dataset-specific fine-tuning and enable lightweight, reproducible stress tests across cuisines.

Contributions.

- **Pipeline:** a training-free food-specific prompt expansion pipeline for linguistic variation.
- **Evaluation:** a diagnostic evaluation framework emphasizing calibration, selective risk, KB coverage, and long-tail behavior.
- **Analysis:** an analysis of when training-free prompt expansion helps and when it does not.

The remainder of the paper reviews related work (Sec. 2), describes our method (Sec. 3), experimental setup (Sec. 4), and results (Sec. 5), and discusses ethics and limitations (Sec. 6).

2. Related work

Food recognition and dataset bias. Food understanding spans fine-grained dish classification, ingredient estimation, recipe retrieval, nutrition/portion estimation, dietary assessment, wearable sensing, and egocentric food understanding [1–3, 6, 16, 19, 24, 31, 38, 39, 41, 42, 60, 61, 72]. Representative datasets include UECFOOD and FOOD-101 [5], larger label spaces such as VIREOFOOD-172 and FOOD2K [68], and cross-modal resources linking images to instructions (RECIPE1M) [40]. More recent benchmarks target nutrition (NUTRITION5K) and egocentric cooking video (EPIC-KITCHENS, YOUCOOK2) [11, 49, 71]. Recent food benchmarks also motivate broader cross-cultural and multimodal evaluation, including METAFOOD3D, FOODVIDEOQA, food degradation analysis, colorful food assessment, and WORLDCUISINES [7, 23, 44, 53**?**]. Despite this progress, dataset bias remains pervasive: label vocabularies, image styles, and geographic coverage are uneven [50]. Cuisine-centric datasets are therefore informative not only as "harder food" but as a controlled probe of cultural distribution shift.

Zero-shot VLMs and open-vocabulary recognition. Contrastive pretraining on image–text pairs enables open-vocabulary recognition by comparing images to prompt-text embeddings. CLIP [36] popularized this paradigm, with follow-ups scaling data and compute (ALIGN, LiT, SigLIP) [18, 63, 64]. Open implementations and scaling studies (OpenCLIP, LAION) show that pretraining corpus and filtering strongly influence downstream robustness [8, 43]. Related multimodal and systems work further underscores the value of scalable training-free evaluation across model families and deployment settings [10, 27, 45, 51, 52, 56, 58, 59, 62, 67]. We focus on frozen contrastive VLMs because they support clean training-free attribution: improvements come from text design and inference, not gradient updates.

Prompting, adapters, and the label phrasing problem. Prompt engineering and prompt ensembling can substantially change zero-shot accuracy, and prompt learning methods (CoOp/CoCoOp) further improve performance by optimizing prompts from labeled data [69, 70]. Lightweight adapters (CLIP-Adapter, Tip-Adapter) adapt representations using cached features or small modules [13, 66]. These approaches typically assume labeled supervision or an explicit optimization step. In contrast, we study a stricter regime where the only control is the *string* used for each class. Our text-bank construction can be viewed as a training-free alternative that is especially relevant when label surface forms are themselves shifted (aliases, transliterations). Recent work also studies training-free prompt weighting for zero-shot prompt ensembles [**?**]. Relatedly,

description-based zero-shot classification uses richer textual descriptions rather than class names alone [**?**]. Our use of log-sum-exp pooling is a simple training-free aggregation choice over prompt variants, not a new prompt weighting method, and we use curated food-specific prompts and aliases rather than generated free-form descriptions.

Retrieval and knowledge grounding. External text resources such as WordNet, Wikidata, and ConceptNet provide aliases and relations [28, 47, 54]. Retrieval-augmented pipelines (RAG/DPR/ColBERT) operationalize access to large corpora [21, 22, 25], but naive retrieval can introduce noise for short or ambiguous dish names. We use conservative string matching to ground dataset labels to a lightweight public dish KB (FoodKB), and treat KB coverage as an explicit diagnostic rather than an implicit assumption. Relatedly, SuS-X studies training-free name-only transfer with external textual descriptors [**?**]. Our KB alias expansion is closest in spirit to descriptor-based transfer, but specialized to the food domain with conservative matching and explicit KB coverage diagnostics.

Transductive inference, test-time adaptation, and uncertainty. Test-time adaptation (TTA) updates parameters using unlabeled target data (TENT, MEMO, CoTTA, EATA, SAR) [32, 33, 55, 57, 65]. Such methods can be powerful but may be sensitive to optimization hyperparameters and can drift under label noise. We instead explore *gradient-free* transductive refinements on frozen embeddings (prior correction, kNN smoothing, prototype/Gaussian alignment) [9, 37, 46, 48]. Martin et al. study transductive zero-shot and few-shot CLIP inference over unlabeled batches [**?**]. Our transductive refinement should be understood as an application of this idea in the food setting using simple gradient-free operators, rather than a brand-new transductive method. Because cultural benchmarks can be ambiguous, we emphasize calibration and selective prediction [15, 17, 30, 34].

Responsible evaluation. Finally, our framing follows recommendations to document dataset/model limitations and avoid essentialist claims about cultures [4, 14, 29]. We treat cuisine datasets as empirical probes of distribution shift in both vision and language, and we report diagnostics that explain errors in terms of coverage and ambiguity rather than stereotypes.

3. Method

3.1. Setting

We assume a fixed label set $\mathcal{Y} = \{y_1, \dots, y_C\}$, a test set of images $\mathcal{D} = \{x_i\}_{i=1}^N$, and a frozen contrastive VLM with image encoder $f(\cdot)$ and text encoder $g(\cdot)$. Following CLIP-style inference [36], we normalize embeddings and score an image against a text prompt t via $\ell(x,t) = \tau\langle f(x), g(t)\rangle$, where τ is the logit scale. **No training or fine-tuning is allowed:** we never update backbone parameters and operate only through (i) text-bank design and (ii) optional post-hoc transformations on frozen embeddings/logits. The non-transductive (inductive) pipeline in this paper consists of text-bank design and prompt pooling only; the optional stage below adds transductive refinement when unlabeled test batches are available.

3.2. Text bank construction

A core source of brittleness is the *string* used to represent each class. For culturally specific dishes, a dataset label may be a transliteration, a colloquial alias, or a compound phrase. We therefore construct a **text bank** $\mathcal{T}(y)$ for each class y. At inference, we score an image against all prompts in $\mathcal{T}(y)$ and then aggregate to a single class score. Concretely, let $\mathcal{S}$ be a set of prompt templates and $\mathcal{V}(y)$ a set of name variants for class y (including optional KB aliases). We instantiate prompts as $t = s(v)$ for $s \in \mathcal{S}$ and $v \in \mathcal{V}(y)$, and define $\mathcal{T}(y) = \{\, s(v) \mid s \in \mathcal{S}, v \in \mathcal{V}(y) \,\}$. This simple factorization makes it easy to ablate where gains come from: templates vs. variants vs. KB aliases.

Given a test image x, we obtain a per-class logit by pooling prompt logits:

$$s_c(x) = \mathrm{LSE}_{t \in \mathcal{T}(y_c)}\ \ell(x,t), \qquad p_c(x) = \mathrm{softmax}(s(x))_c. \tag{1}$$

We then predict $\hat{y}(x) = \arg\max_c s_c(x)$ and use $\max_c p_c(x)$ as confidence.

What is new here? Our contribution is the food-specific composition of prompt templates, conservative lexical variants, optional KB aliases, and a unified evaluation/diagnostic framework around these choices. It is an application-driven synthesis for food recognition, not a new CLIP training method.

Prompt suites. We use three prompt suites: **minimal** ("y"), **base** (generic photo prompts such as "a photo of y"), and **plate** (food-aware templates such as "a plate of y" or "y served as a dish"). Food-aware templates are motivated by common caption patterns for meals and dishes in web data. In our release we log the exact prompt strings to avoid hidden prompt hacking.

Optional cuisine hint. When a dataset is explicitly cuisine-specific, we also test a minimal hint that conditions the prompt on the cuisine name (e.g., "a Turkish dish called y"). This sometimes improves accuracy by nudging the text embedding toward the relevant region of the language

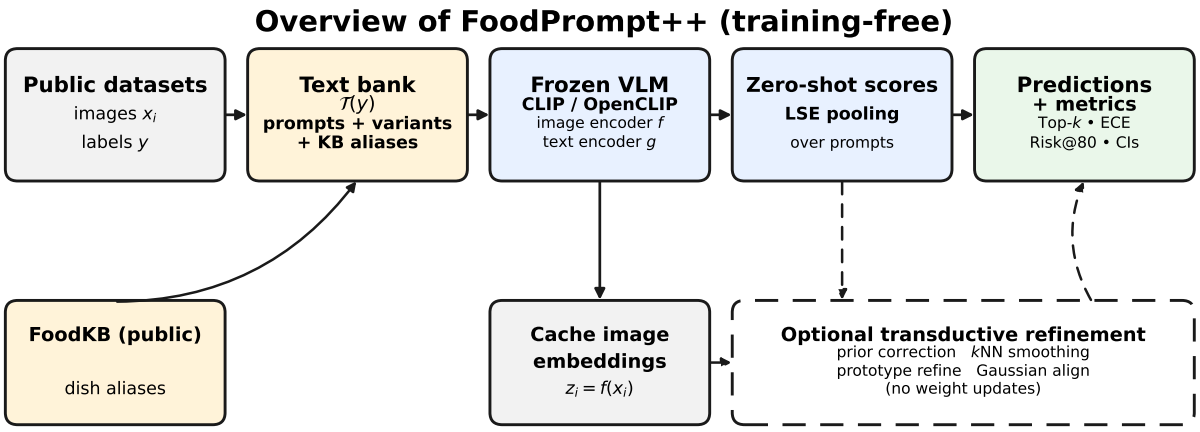


Figure 2. Overview of MetaFood (training-free). We build a per-class text bank (prompt suites + lexical variants + optional FoodKB aliases), pool variant logits with log-sum-exp, and optionally apply deterministic transductive refinements on cached frozen image embeddings (prior correction, kNN smoothing, prototype/Gaussian alignment) without updating model weights. Dashed box/arrows denote the optional refinement path.

space. We treat it as a baseline rather than a core component, because it is dataset-dependent and could be misused if interpreted as a claim about what a cuisine "should" contain.

Conservative lexical variants. Within a suite, small lexical edits can move the text embedding. We generate conservative variants that preserve semantics: lowercasing, punctuation stripping, hyphen/space swaps, collapsing repeated whitespace, removing bracketed qualifiers, and light template alternations ("dish of y" vs. "y"). We avoid aggressive paraphrasing or translation to reduce semantic drift and to keep the method auditable. These variants yield **Prompt+Variants** baselines.

3.3. Knowledge-grounded alias expansion

Prompt variants do not address cases where a dataset label is simply not well aligned with the pretrained text space. We therefore optionally expand class names using a public dish knowledge base, FoodKB. Each KB entry provides a canonical dish name and alternate aliases.

Conservative KB matching. We match each dataset label string to at most one KB entry by normalized string similarity. In practice we normalize strings by case-folding, stripping punctuation and diacritics, and collapsing repeated whitespace, then compute a token-based similarity (RapidFuzz token-set ratio) against KB canonical names. A class is marked *KB-matched* only if a unique best match exceeds a global threshold δ; ties at the top score are treated as ambiguous and rejected. If no entry passes δ, or if the label is short, generic, or only partially overlapping with KB names, we keep the class *KB-unmatched* and use only prompt variants. In our current datasets, conservative alias matching applies to only a subset of classes. We therefore treat alias matching as a high-precision, low-recall design choice rather than a high-coverage retrieval scheme. Lowering δ increases accepted matches but also increases the risk of mis-grounding, whereas raising δ rejects more classes and reduces coverage. We report accepted/rejected counts and threshold sensitivity in Sec. 5.4. KB grounding uses only public KB text and dataset label strings; it never uses images, annotations, or class exemplars.

Alias filtering. To reduce noise, we discard aliases that are extremely long, contain obvious boilerplate, or are overly generic (e.g., "food", "dish"). We also deduplicate aliases after normalization. The resulting text bank is denoted **+FoodKB aliases**. We log all matches and a matched/unmatched flag, enabling the coverage diagnostics in Sec. 5.4.

3.4. Logit pooling with log-sum-exp

Following prior work on training-free prompt aggregation and weighting [**?**], we aggregate logits with log-sum-exp (LSE) pooling:

$$m = \max_k \ell_k, \qquad \mathrm{LSE}(\{\ell_k\}) = m + \frac{1}{\alpha} \log \sum_k \exp\big(\alpha(\ell_k - m)\big). \tag{2}$$

where $\alpha > 0$ controls sharpness (we use a fixed α across datasets). LSE interpolates between averaging (robust) and

max pooling (select the best phrasing), and is computed stably via `logsumexp`. We keep it because it is simple, stable, training-free, and reproducible. We therefore use **Prompt+Variants (LSE)** as a robust prompting baseline rather than as a novel contribution.

Fixed hyperparameters. We use a small set of global hyperparameters (e.g., LSE sharpness α, maximum aliases K, and match threshold δ) and keep them fixed across datasets and backbones. This "no per-dataset tuning" choice is important for a credible cultural generalization claim.

3.5. Optional gradient-free transductive refinement

The inductive / non-transductive pipeline in this paper ends with text-bank construction and LSE pooling. **This stage is transductive and requires access to the full unlabeled test set.** It is therefore suitable for offline batch evaluation and less suitable for single-image online deployment. When unlabeled test images are available, we additionally apply lightweight refinements that use the test distribution *without* updating model weights. Following prior transductive CLIP inference [**?**], we call such methods *transductive*. Unlike parameter-updating TTA methods (e.g., TENT [55]), our refinements are deterministic and easy to audit.

Prior correction. Let $p_i = \mathrm{softmax}(s(x_i))$ be the zero-shot posterior for test image x_i. We estimate a smoothed test-set class prior

$$\hat{\pi}_c = \frac{1}{N}\sum_{i=1}^{N} p_{i,c}, \qquad \tilde{\pi}_c = \frac{\hat{\pi}_c + \epsilon}{\sum_{c'}(\hat{\pi}_{c'} + \epsilon)}. \tag{3}$$

We then apply a log-prior correction

$$s^{\mathrm{pc}}_{i,c} = s_{i,c} - \lambda \, \log \tilde{\pi}_c, \tag{4}$$

with fixed λ and small ϵ for stability on small test sets. This yields **prior_corr**, inspired by label-shift and balanced-softmax ideas [37].

kNN smoothing. We compute frozen image embeddings $z_i = f(x_i)$ and build a cosine kNN graph (FAISS [20]). For each test point i, let $\mathcal{N}_k(i)$ denote its k nearest neighbors in embedding space. We form a smoothed posterior by mixing the local neighborhood average with the original prediction:

$$\tilde{p}_i = \mathrm{Normalize}\left((1-\beta)\, p_i + \frac{\beta}{k}\sum_{j\in\mathcal{N}_k(i)} p_j\right), \tag{5}$$

where $\beta \in [0,1]$ is fixed globally. This classic transductive trick exploits representation geometry without learning [9]. We treat k as a small constant (e.g., k=20) and run in chunks to limit memory.

Prototype/Gaussian alignment. Given (possibly smoothed) pseudo-posteriors $\tilde{p}_{i,c}$ and embeddings z_i, we form class prototypes

$$\mu_c = \frac{\sum_{i=1}^{N} \tilde{p}_{i,c}\, z_i}{\left\|\sum_{i=1}^{N} \tilde{p}_{i,c}\, z_i\right\|_2}, \qquad s^{\mathrm{proto}}_{i,c} = \tau\langle z_i, \mu_c\rangle. \tag{6}$$

This yields **proto_refine** [46]. We further apply a lightweight moment-matching alignment (diagonal CORAL-style) between the test embedding distribution and the prototype distribution. Let μ_z, σ_z be the per-dimension mean and standard deviation of $\{z_i\}$, and let μ_μ, σ_μ be those of $\{\mu_c\}$. We align embeddings as

$$\bar{z}_i = \mathrm{diag}(\sigma_\mu)\,\mathrm{diag}(\sigma_z)^{-1}(z_i - \mu_z) + \mu_\mu, \qquad s^{\mathrm{ga}}_{i,c} = \tau\langle \bar{z}_i, \mu_c\rangle. \tag{7}$$

This yields **gauss_align**, inspired by CORAL [48] but implemented without gradients. Both operations can improve calibration even when Top-1 changes little, which is valuable for abstention-style systems.

3.6. Method variants and compute

We denote by **MetaFood (pure)** the strongest non-transductive text-bank configuration (prompt pooling + KB aliases) without embedding refinement. **MetaFood** refers to the combined pipeline when the optional transductive refinement is enabled. All variants reuse cached image features: we extract $z_i = f(x_i)$ once per dataset and cache to disk, then sweep text banks and refinements cheaply. This design is important for workshop-style ablation density, and it also reduces peak memory compared to storing logits for every prompt variant.

Memory-efficient implementation. To limit peak memory, we cache normalized image embeddings to disk and stream batches through text-bank scoring. Text embeddings are also cached per backbone, and kNN queries are computed in chunks. This design was motivated by practical workshop constraints (single GPU / limited RAM) and makes it feasible to run many ablations and bootstrap resamples.

4. Experimental setup

4.1. Datasets

We evaluate on two cuisine-focused, publicly available image classification datasets. This is a deliberately narrow stress-test setting rather than a claim of broad benchmark coverage.

TURK contains 15 Turkish dish classes with an official test split of 742 images. CAFD contains 42 Central Asian dish classes; for rapid iteration we report results on a fixed random subset of 1000 test images from the standard test

Dataset	classes	test images	KB coverage	Eval. split
TURK	15	742	6/15	official test split
CAFD	42	1000	8/42	fixed test subset

Table 1. Dataset summary. KB coverage reports the number of classes with accepted conservative FoodKB matches in our current runs.

split, while the benchmark also supports full-split evaluation. Both datasets emphasize culturally specific dish names and expose both visual and linguistic distribution shift.

We do not yet include a standard non-shift food-recognition benchmark or a broader multilingual benchmark such as WorldCuisines [**?**]; this is an important limitation. Within the current paper, the main control for naming variation is the comparison between minimal prompts, food-aware prompts, lexical variants, and KB aliases on the same images: any gains from these text-bank changes isolate the effect of label surface form without changing the visual data.

Knowledge base. For alias expansion we use FoodKB (`worldcuisines/food-kb`), a public dish knowledge base containing 2414 entries (split `main`). We treat it purely as auxiliary text: no images or additional labels are introduced.

4.2. Backbones

All experiments use frozen contrastive VLMs from OpenCLIP. Our default backbone is ViT-B/32 with LAION-2B pretraining (`laion2b_s34b_b79k`) [8, 43]. We additionally verify key trends with the OpenAI-pretrained ViT-B/32 checkpoint. Across backbones, we keep all method hyperparameters fixed to avoid per-dataset tuning.

4.3. Methods compared

We evaluate both non-transductive text-bank variants and optional transductive refinements. Table 2 summarizes which components are enabled and whether the unlabeled test batch is used.

All methods are training-free; none update model weights.

4.4. Metrics and uncertainty

Let $p_i = \mathrm{softmax}(s(x_i))$ be predicted class probabilities and let confidence be $c_i = \max_c p_{i,c}$. We report Top-k accuracy, calibration via expected calibration error (ECE; 15 bins) and NLL [17, 30], and selective prediction Risk@80 [12, 15]. Concretely,

$$\mathrm{Top1} = \frac{1}{N}\sum_{i=1}^{N} \mathbf{1}[\hat{y}_i = y_i], \qquad \mathrm{NLL} = -\frac{1}{N}\sum_{i=1}^{N} \log p_{i,y_i}. \tag{8}$$

For calibration, we bin predictions by confidence into M equal-width bins $\{B_m\}_{m=1}^{M}$ and compute

$$\mathrm{ECE} = \sum_{m=1}^{M} \frac{|B_m|}{N} \left|\, \mathrm{acc}(B_m) - \mathrm{conf}(B_m) \right|, \tag{9}$$

where $\mathrm{acc}(B_m)$ is the average correctness and $\mathrm{conf}(B_m)$ is the average confidence in bin m. For selective prediction, let $S_{0.8}$ be the indices of the top 80% highest-confidence samples; then

$$\mathrm{Risk@80\%} = 1 - \frac{1}{|S_{0.8}|}\sum_{i \in S_{0.8}} \mathbf{1}[\hat{y}_i = y_i]. \tag{10}$$

As an ambiguity proxy we also report mean predictive entropy

$$H(p_i) = -\sum_c p_{i,c} \log p_{i,c}, \qquad \bar{H} = \frac{1}{N}\sum_i H(p_i). \tag{11}$$

For Top-1, we report 95% confidence intervals from a nonparametric bootstrap over test samples: we sample N indices with replacement, compute $\mathrm{Top1}^{(b)}$ for $b \in \{1, \ldots, B\}$, and report the percentile interval $[Q_{0.025}, Q_{0.975}]$. We also save per-sample predictions to enable auditing.

4.5. Implementation and reproducibility

We use HuggingFace `datasets` for data loading [26] and PyTorch for inference [35]. To enable many ablations efficiently, we cache image embeddings once per dataset and reuse them across methods. Nearest-neighbor queries are implemented with FAISS [20]. Each run logs (i) the exact prompt strings, (ii) KB match decisions, and (iii) all random seeds. The benchmark exports a self-contained HTML report with tables, figures, and optional saved example predictions.

5. Results and analysis

5.1. Non-transductive vs. transductive results

Table 3 separates non-transductive and transductive rows. Non-transductive rows use only text-bank design; transductive rows additionally use the full unlabeled test batch.

Non-transductive results. On TURK, most of the gain is already visible before any transductive refinement: Prompt+Variants (LSE) improves to +FoodKB aliases in

Variant family	Prompt suites variants	KB aliases KB aliases	Uses unlabeled test batch	Setting
zs_minimal, **zs_base**, **zs_plate**	Yes	No	No	non-transductive
zs_plate_variants_lse	Yes	No	No	non-transductive
zs_plate_kb_alias, **MetaFood _pure**	Yes	Yes	No	non-transductive
prior_corr, **knn_smooth**, **proto_refine**, **gauss_align**, **MetaFood**	Yes	optional	Yes	transductive

Table 2. Method summary. Any method that uses the unlabeled test batch is reported as transductive.

Dataset	Setting	Method	Top-1 (%)↑	95% CI	Top-5 (%)↑	ECE (%)↓	Risk@80 (%)↓
TURK	non-transductive	Prompt+Variants (LSE)	41.5	[38.0, 45.1]	81.5	19.6	54.2
	non-transductive	+FoodKB aliases	44.6	[41.0, 48.2]	81.8	**13.3**	50.0
	transductive	kNN smoothing	44.5	[40.9, 48.1]	**85.7**	14.4	51.0
	transductive	**MetaFood**	**48.7**	[45.1, 52.3]	83.4	15.4	**44.3**
CAFD	non-transductive	Prompt+Variants (LSE)	12.3	[10.4, 14.4]	32.1	23.6	86.0
	non-transductive	+FoodKB aliases	13.2	[11.2, 15.4]	32.4	20.7	84.4
	transductive	kNN smoothing	**14.0**	[11.9, 16.3]	**34.4**	18.1	**83.9**
	transductive	**MetaFood**	13.4	[11.3, 15.7]	31.2	**5.8**	84.7

Table 3. Main results with bootstrap 95% confidence intervals for Top-1. ECE and Risk@80 measure calibration and selective reliability (lower is better). Non-transductive rows use only text-bank design; transductive rows use the full unlabeled test batch. **Bold** marks the best value per metric within each dataset block.

Table 3, and Table 4 shows MetaFood (pure) reaching 48.4 Top-1 without using the test batch. On CAFD, non-transductive text-bank changes help modestly, but absolute accuracy remains low, indicating that naming fixes alone do not solve the harder shift.

Optional transductive results. When the full unlabeled test batch is available, kNN smoothing and MetaFood can further adjust predictions or calibration. These rows should be read separately from the non-transductive results because they assume batch access to test images. On TURK, the full pipeline reaches 48.7% Top-1 and reduces Risk@80 to 44.3. On CAFD, transductive refinement affects calibration more than Top-1.

5.2. Calibration and selective prediction

TURK: accuracy and selective reliability improve. On TURK, MetaFood improves Top-1 over the strongest text baseline (+FoodKB aliases) and substantially reduces Risk@80, indicating fewer high-confidence mistakes [15]. Notably, the gains come without any weight updates: improving the *language side* (aliases + prompt pooling) is enough to recover a meaningful fraction of errors.

CAFD: reliability gains matter even when accuracy is low. On CAFD, absolute accuracy is low across methods, but MetaFood sharply improves calibration (ECE 5.8% vs. 18–24%). This matters because in ambiguous cultural settings, a system that knows when it is uncertain can abstain or ask for clarification rather than outputting a confident wrong name [17, 34].

What do the low CAFD numbers mean? The ∼13–14% Top-1 results on CAFD should be read as evidence that severe cultural shift remains hard for training-free methods. This may reflect a real ceiling for the current pipeline: when label strings are weakly aligned with pretraining, KB coverage is sparse, and classes are visually overlapping, prompt expansion and frozen-embedding refinement are not a full solution. In this regime, gains in calibration and selective prediction still matter, but the method should be interpreted as a practical baseline and diagnostic framework rather than a complete solution for multicultural food recognition.

Confidence intervals. Bootstrap intervals in Table 3 are relatively tight on TURK (hundreds of test images) and wider on CAFD, reflecting both the smaller evaluation subset and higher variance under long-tail class difficulty. We recommend reporting CIs for cultural benchmarks because small changes in sampling or label strings can otherwise lead to over-interpreting marginal gains.

Backbone sensitivity. We additionally ran our benchmark with an OpenAI-pretrained ViT-B/32 checkpoint. While absolute scores differ across checkpoints (consistent with prior scaling studies [8]), the qualitative trends were stable: food-aware prompts outperform minimal prompts, KB alias expansion helps when matches are available, and

Variant	Top-1	ECE	Top-1	ECE
	TURK		CAFD	
Prompt+Variants (LSE)	41.5	19.6	12.3	23.6
+FoodKB aliases	44.6	13.3	13.2	20.7
+ kNN smoothing	44.5	14.4	14.0	18.1
MetaFood (pure text-bank)	48.4	13.7	13.2	21.8
MetaFood + prototype refine	**48.7**	15.4	**13.4**	**5.8**
MetaFood + Gaussian align	**48.7**	15.2	**13.4**	**5.8**

Table 4. Ablations for Top-1 accuracy (%) and ECE (%). The Setting column makes explicit which rows use the unlabeled test batch. Text-bank design drives accuracy on TURK; prototype/Gaussian refinement improves calibration on CAFD.

embedding-space refinement affects calibration more than Top-1 on the hardest dataset.

Positioning vs. test-time adaptation (TTA). Recent TTA methods can improve robustness under distribution shift by updating model parameters at inference time [32, 33, 55, 57]. Our goal here is complementary: to establish a strong *training-free* baseline family whose behavior is easy to audit. In particular, our main gains on TURK come from the language side (aliases + prompt pooling), which is orthogonal to parameter-updating TTA and could be combined with it in future work.

5.3. Ablations: text bank vs. embedding refinement

Table 4 separates text-bank components (prompt pooling, KB aliases) from embedding-space refinements (kNN, prototype, Gaussian). On TURK, most gains come from better text banks (aliases + pooling). On CAFD, prototype/Gaussian refinement mainly affects calibration: Top-1 changes little, but ECE drops dramatically. We interpret this as follows: when the model's top-1 class is often wrong due to ambiguity, refinement cannot magically fix semantics, but it can still reshape confidence so that the model becomes less over-confident.

5.4. Knowledge-base coverage diagnostics

A key question is *when* retrieval helps. Our conservative FoodKB matching covers only a subset of classes (6/15 for TURK and 8/42 for CAFD in our runs), leaving 9/15 and 34/42 classes unmatched, respectively. Figure 3 plots per-class accuracy split by KB-matched vs. unmatched classes. Matched classes tend to have higher accuracy, but the dominant error mass lies in KB-unmatched classes, explaining why gains can saturate when KB coverage is sparse. This also suggests a practical direction: improving name grounding (better KB coverage, multilingual aliases) may yield larger gains than more elaborate ensembling.

When can retrieval hurt? If a label is mis-matched to a KB entry, added aliases can shift the text embedding toward an incorrect concept, harming both accuracy and calibration. For this reason we prefer precision over recall in matching and treat unmatched classes separately. In small sweeps, lowering the match threshold δ increases coverage but can reduce Top-1 when dish names are short and ambiguous. We therefore recommend reporting coverage and match thresholds, and manually spot-checking a sample of matches for each dataset.

5.5. Per-class behavior and long-tail errors

Figure 4 shows sorted per-class accuracies for MetaFood (pure). Both datasets exhibit long-tail behavior: a few classes are recognized well while many are near-zero, especially on CAFD. This pattern is consistent with a combination of (i) visually similar dishes across classes, (ii) short and ambiguous label strings, and (iii) limited KB grounding for rare or regional names.

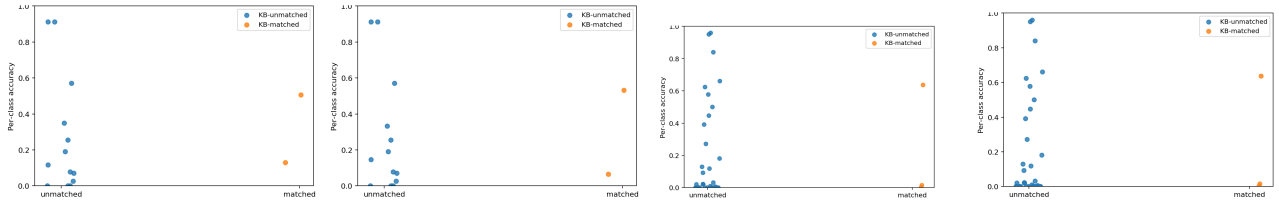
Ambiguity and label noise. Some dish categories overlap visually (e.g., stews, dumplings, or stuffed pastries) or differ primarily in ingredients that may not be visible. Such ambiguity implies an inherent error floor, so emphasizing calibration and abstention can be more realistic than chasing marginal Top-1 gains on the long tail.

Why is CAFD harder? The diagnostics suggest two concrete factors. First, KB coverage is low: most classes are unmatched, so alias expansion cannot help, and the model falls back to brittle label strings. Second, per-class performance is extremely heavy-tailed, implying that average accuracy is dominated by a few recognizable classes while many are effectively unrecognized. In such regimes, calibration and selective prediction become more meaningful than incremental Top-1 gains: a model that is calibrated can abstain on the long tail while still being useful on the head.

Takeaway. On easier cuisine sets (TURK), accuracy improvements and better selective reliability can be achieved largely from text-bank design, with optional transductive refinement giving a smaller additional gain. On harder sets (CAFD), the most defensible improvements are in *trustworthiness* (calibration), which is essential for systems that can abstain or ask users for clarification rather than returning over-confident wrong dish names.

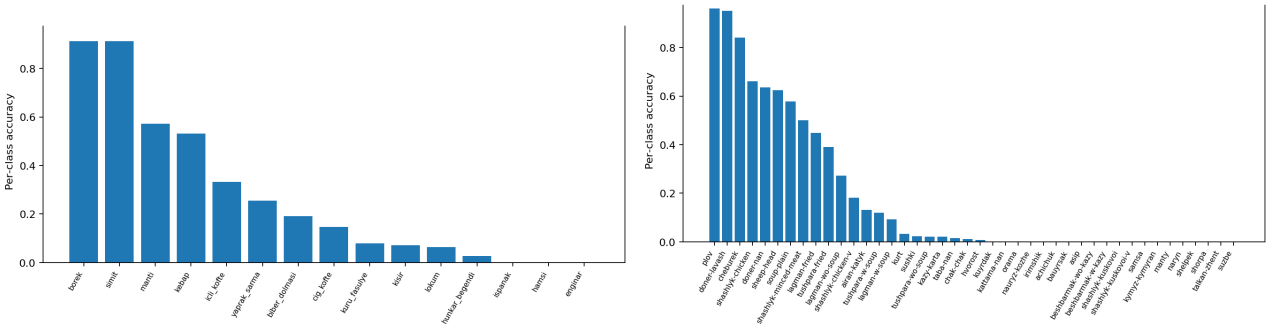
6. Ethics, limitations, and responsible framing

Empirical framing. We evaluate VLMs under cultural distribution shift and avoid claims about cuisines or communities. Observed differences reflect dataset design,



(a) TURK: Prompt+Variants (LSE) (b) TURK: MetaFood (pure) (c) CAFD: Prompt+Variants (LSE) (d) CAFD: MetaFood (pure)

Figure 3. KB coverage vs. per-class accuracy. Each point is a class; orange points are KB-matched and blue points are unmatched. Sparse KB coverage is a primary driver of remaining errors.



(a) TURK: MetaFood (pure) (b) CAFD: MetaFood (pure)

Figure 4. Per-class accuracy (sorted) for MetaFood (pure). Cultural food recognition exhibits heavy tails: a small head of easy classes and a large tail of near-zero classes, amplified when KB coverage is sparse.

model pretraining, and evaluation protocols rather than inherent cultural properties. We focus on measurable factors such as label ambiguity, KB coverage, and long-tail behavior.

Key limitations. Our pipeline is built from known ingredients rather than a new training algorithm. Current evaluation is limited to two cuisine-focused datasets, so the empirical scope is narrow. KB alias coverage is partial, especially on the harder dataset. The optional transductive refinement assumes access to the full unlabeled test set. Harder datasets such as CAFD remain challenging, so the method should be read as a practical training-free baseline and diagnostic framework rather than a full solution for multicultural food recognition.

Dataset and KB limits. Cuisine datasets may contain collection bias and label noise. Knowledge bases can be incomplete or skewed toward well-documented foods. Our conservative matching reduces mis-grounding but results in sparse coverage, which we report as a diagnostic.

7. Conclusion

We introduced MetaFood, a training-free framework for culturally diverse food recognition using frozen vision–language models. It combines robust prompting and knowledge-base aliases with optional gradient-free transductive refinement. On TURK, it improves Top-1 accuracy and selective risk; on the harder CAFD dataset, it substantially improves calibration.

We emphasize *diagnostic evaluation* in cultural domains: reporting KB coverage, long-tail behavior, and reliability metrics helps interpret failures without stereotyping. These diagnostics generalize to other culturally grounded tasks where naming variation matters.

Future work includes multilingual name matching (including non-Latin scripts), expanding lightweight KBs, and scaling to more cuisines and backbones. Interactive systems that abstain or return multiple plausible dish names can turn uncertainty into a usability feature.

More broadly, we advocate reporting coverage diagnostics and confidence intervals alongside accuracy to clarify dataset difficulty under cultural shift.

References

[1] Md. Toukir Ahmed, Md Wadud Ahmed, and Mohammed Kamruzzaman. A systematic review of explainable artificial intelligence for spectroscopic agricultural quality assessment. *Computers and Electronics in Agriculture*, 235: 110354, 2025. 2

[2] Md Wadud Ahmed, Sahir Junaid Hossainy, Alin Khaliduzzaman, Jason Lee Emmert, and Mohammed Kamruzzaman. Non-destructive optical sensing technologies for advancing the egg industry toward industry 4.0: A review. *Comprehensive Reviews in Food Science and Food Safety*, 2023.

[3] Saeed S. Alahmari and Tawfiq Salem. Food state recognition using deep learning. *IEEE Access*, 10:131061–131073, 2022. 2

[4] Emily M Bender, Timnit Gebru, Angelina McMillan-Major, and Shmargaret Shmitchell. On the dangers of stochastic parrots: Can language models be too big? In *ACM Conference on Fairness, Accountability, and Transparency (FAccT)*, 2021. 3

[5] Lukas Bossard, Matthieu Guillaumin, and Luc Van Gool. Food-101 – mining discriminative components with random forests. In *European Conference on Computer Vision (ECCV)*, 2014. 2

[6] Jingjing Chen, Chong-Wah Ngo, et al. Deep-based ingredient recognition for cooking recipe retrieval. In *ACM International Conference on Multimedia (ACM MM)*, 2016. 2

[7] Yuhao Chen, Jiangpeng He, Chris Czarnecki, Gautham Vinod, Talha Ibn Mahmud, Siddeshwar Raghavan, Jinge Ma, Dayou Mao, Saeejith Nair, Pengcheng Xi, Alexander Wong, Edward Delp, and Fengqing Zhu. Metafood3d: Large 3d food object dataset with nutrition values. *arXiv preprint arXiv:2409.01966*, 2024. 2

[8] Mehdi Cherti, Rémi Beaumont, Ross Wightman, Maya Coombes, Alexander Mullis, Mitchell Wortsman, Ludwig Schmidt, Jenia Jitsev, and Daniel Haziza. Scaling laws for contrastive language–image learning. In *International Conference on Learning Representations (ICLR)*, 2023. 1, 2, 6, 7

[9] Thomas Cover and Peter Hart. Nearest neighbor pattern classification. pages 21–27, 1967. 3, 5

[10] Yuchen Cui et al. No, to the right: Online language correction for robotic navigation. In *ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, 2023. 2

[11] Dima Damen et al. EPIC-KITCHENS-100. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 2

[12] Ran El-Yaniv and Yair Wiener. On the foundations of noise-free selective classification. *Journal of Machine Learning Research*, 11(53):1605–1641, 2010. 6

[13] Peng Gao, Shijie Geng, Renrui Zhang, Teli Ma, Rongyao Fang, Yongfeng Zhang, Hongsheng Li, and Yu Qiao. CLIP-adapter: Better vision-language models with feature adapters. *arXiv preprint arXiv:2110.04544*, 2021. 2

[14] Timnit Gebru, Jamie Morgenstern, Briana Vecchione, Jennifer Wortman Vaughan, Hanna Wallach, Hal Daumé III, and Kate Crawford. Datasheets for datasets. In *Communications of the ACM*, 2021. 3

[15] Yonatan Geifman and Ran El-Yaniv. Selective classification for deep neural networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017. 2, 3, 6, 7

[16] Kristen Grauman, Andrew Westbury, Eugene Byrne, Zachary Chavis, Antonino Furnari, Rohit Girdhar, Jackson Hamburger, Hao Jiang, Miao Liu, Xingyu Liu, Miguel Martin, Tushar Nagarajan, Ilija Radosavovic, Santhosh Kumar Ramakrishnan, Fiona Ryan, Jayant Sharma, Michael Wray, Mengmeng Xu, Eric Zhongcong Xu, Chen Zhao, Siddhant Bansal, Dhruv Batra, Vincent Cartillier, Sean Crane, Tien Do, Morrie Doulaty, Akshay Erapalli, Christoph Feichtenhofer, Adriano Fragomeni, Qichen Fu, Abrham Gebreselasie, Cristina Gonzalez, James Hillis, Xuhua Huang, Yifei Huang, Wenqi Jia, Weslie Khoo, Jachym Kolar, Satwik Kottur, Anurag Kumar, Federico Landini, Chao Li, Yanghao Li, Zhenqiang Li, Karttikeya Mangalam, Raghava Modhugu, Jonathan Munro, Tullie Murrell, Takumi Nishiyasu, Will Price, Paola Ruiz Puentes, Merey Ramazanova, Leda Sari, Kiran Somasundaram, Audrey Southerland, Yusuke Sugano, Ruijie Tao, Minh Vo, Yuchen Wang, Xindi Wu, Takuma Yagi, Ziwei Zhao, Yunyi Zhu, Pablo Arbelaez, David Crandall, Dima Damen, Giovanni Maria Farinella, Christian Fuegen, Bernard Ghanem, Vamsi Krishna Ithapu, C. V. Jawahar, Hanbyul Joo, Kris Kitani, Haizhou Li, Richard Newcombe, Aude Oliva, Hyun Soo Park, James M. Rehg, Yoichi Sato, Jianbo Shi, Mike Zheng Shou, Antonio Torralba, Lorenzo Torresani, Mingfei Yan, and Jitendra Malik. Ego4d: Around the world in 3,000 hours of egocentric video. 2022. 2

[17] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger. On calibration of modern neural networks. In *International Conference on Machine Learning (ICML)*, 2017. 2, 3, 6, 7

[18] Chao Jia, Yinfei Yang, Ye Xia, Yi-Ting Chen, Zarana Parekh, Hieu Pham, Quoc V Le, et al. Scaling up visual and vision-language representation learning with noisy text supervision. In *International Conference on Machine Learning (ICML)*, 2021. 2

[19] Wenyan Jia et al. Automatic food detection in egocentric images using artificial intelligence technology. *Public Health Nutrition*, 2018. 2

[20] Jeff Johnson, Matthijs Douze, and Hervé Jégou. FAISS: A library for efficient similarity search. In *International Conference on Big Data*, 2019. 5, 6

[21] Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wentau Yih. Dense passage retrieval for open-domain question answering. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2020. 3

[22] Omar Khattab and Matei Zaharia. ColBERT: Efficient and effective passage search via contextualized late interaction over BERT. In *International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, 2020. 3

[23] Weslie Khoo, David Crandall, Jiangpeng He, et al. Bridging human intuition and AI in colorful food assessment. In *CHI Conference on Human Factors in Computing Systems (CHI)*, 2025. 2

[24] Zhengyi Kwan et al. Nutrition estimation for dietary man-

agement: A transformer approach with depth sensing. *IEEE Transactions on Multimedia*, 2025. arXiv:2406.01938. 2

[25] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, et al. Retrieval-augmented generation for knowledge-intensive NLP tasks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. 3

[26] Quentin Lhoest, Albert Villanova del Moral, Yacine Jernite, Abhishek Thakur, Patrick von Platen, Suraj Patil, Julien Chaumond, Mariama Drame, Julien Plu, Lewis Tunstall, et al. Datasets: A community library for natural language processing. In *Conference on Empirical Methods in Natural Language Processing (EMNLP): System Demonstrations*, 2021. 6

[27] Joshua Li, Fernando Jose Pena Cantu, Emily Yu, and Alexander Wong. Samjam: Zero-shot video scene graph generation for egocentric kitchen videos, 2025. 2

[28] George A Miller. WordNet: A lexical database for english. *Communications of the ACM*, 38(11):39–41, 1995. 3

[29] Margaret Mitchell, Simone Wu, Andrew Zaldivar, Parker Barnes, Lucy Vasserman, Ben Hutchinson, Elena Spitzer, Inioluwa Raji, and Timnit Gebru. Model cards for model reporting. In *Conference on Fairness, Accountability, and Transparency (FAccT)*, 2019. 3

[30] Mahdi Pakdaman Naeini, Gregory F Cooper, and Milos Hauskrecht. Obtaining well calibrated probabilities using Bayesian binning into quantiles. In *AAAI Conference on Artificial Intelligence*, 2015. 3, 6

[31] Bhalaji Nagarajan, Petia Radeva, et al. Bayesian DivideMix++: Robust learning under noisy labels by combining bayesian inference and label correction. *Neural Networks*, 2024. 2

[32] Shuaicheng Niu, Yizhou Wu, Yifan Zhang, Yao Chen, Xiaoxiao Li, and Mingkui Tan. EATA: Efficient test-time adaptation without forgetting. In *International Conference on Machine Learning (ICML)*, 2022. 3, 8

[33] Shuaicheng Niu, Jiaxiang Wu, Yifan Zhang, Zhiquan Wen, Yaofo Chen, Peilin Zhao, and Mingkui Tan. Towards stable test-time adaptation in dynamic wild world. In *International Conference on Learning Representations (ICLR)*, 2023. 3, 8

[34] Yaniv Ovadia et al. Can you trust your model's uncertainty? evaluating predictive uncertainty under dataset shift. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019. 2, 3, 7

[35] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. PyTorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019. 6

[36] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning (ICML)*, 2021. 1, 2, 3

[37] Jiawei Ren, Mengye Yu, Chen Wang, Quanquan Sun, and Trevor Darrell. Balanced meta-softmax for long-tailed visual recognition. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. 3, 5

[38] Javier Ródenas, Bhalaji Nagarajan, Marc Bolaños, and Petia Radeva. Learning multi-subset of classes for fine-grained food recognition, 2022. 2

[39] Tawfiq Salem et al. Zero-shot SAM for food image segmentation. *Electronics*, 2025. 2

[40] Amaia Salvador, Nicholas Hynes, Yusuf Aytar, Javier Marin, Ferda Ofli, and Ingmar Weber. Learning cross-modal embeddings for cooking recipes and food images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3020–3028, 2017. 2

[41] Edward Sazonov and Juan M. Fontana. A sensor system for automatic detection of food intake through non-invasive monitoring of chewing. *IEEE Sensors Journal*, 12(5):1340–1348, 2012. 2

[42] Edward Sazonov and Stephanie Schuckers. Wearable sensing for automatic eating detection and dietary monitoring. *IEEE Journal of Biomedical and Health Informatics*, 20(6): 1520–1530, 2016. 2

[43] Christoph Schuhmann, Richard Vencu, Rémi Beaumont, Robert Kaczmarczyk, Alexander Mullis, Akash Katta, Maya Coombes, Jenia Jitsev, and Aran Komatsuzaki. LAION-5b: An open large-scale dataset for training next generation image-text models. In *Advances in Neural Information Processing Systems (NeurIPS) Datasets and Benchmarks*, 2022. 1, 2, 6

[44] Krish Shah, Siddharth Viswanath, Pengcheng Xi, Alexander Wong, Yuhao Chen, et al. FoodVideoQA: A benchmark dataset for multimodal food video question answering. In *CVPR Workshops*, 2025. 2

[45] Kaleb E. Smith et al. On the impact of cross-domain data on German language models. Preprint, 2023. 2

[46] Jake Snell, Kevin Swersky, and Richard Zemel. Prototypical networks for few-shot learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017. 3, 5

[47] Robyn Speer, Joshua Chin, and Catherine Havasi. Conceptnet 5.5: An open multilingual graph of general knowledge. In *AAAI Conference on Artificial Intelligence*, 2017. 3

[48] Baochen Sun and Kate Saenko. Deep CORAL: Correlation alignment for deep domain adaptation. In *European Conference on Computer Vision Workshops (ECCVW)*, 2016. 3, 5

[49] Quin Thames, Arjun Karpur, Wade Norris, Fangting Xia, Liviu Panait, Tobias Weyand, and Jack Sim. Nutrition5k: Towards automatic nutritional understanding of generic food. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 1, 2

[50] Antonio Torralba and Alexei A Efros. Unbiased look at dataset bias. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2011. 2

[51] Zhengzhong Tu et al. MAXIM: Multi-axis MLP for image processing. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 2

[52] Zhengzhong Tu et al. MaxViT: Multi-axis vision transformer. In *European Conference on Computer Vision (ECCV)*, 2022. 2

[53] Julio Valdes, Stephie Liu, Shawn Yang, Yuhao Chen, Alexander Wong, Pengcheng Xi, et al. Food degradation analysis from OpenRoad. In *CVPR Workshops*, 2025. 2

[54] Denny Vrandečić and Markus Krötzsch. Wikidata: A free collaborative knowledgebase. In *International Conference on World Wide Web (WWW)*, 2014. 3

[55] Dequan Wang, Evan Shelhamer, Shaoteng Liu, Bruno Olshausen, and Trevor Darrell. Tent: Fully test-time adaptation by entropy minimization. In *International Conference on Learning Representations (ICLR)*, 2021. 3, 5, 8

[56] Ke-Lei Wang, Pin-Hsuan Chou, Young-Ching Chou, Chia-Jen Liu, Cheng-Kuan Lin, and Yu-Chee Tseng. Mp-polarmask: A faster and finer instance segmentation for concave images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2024. 2

[57] Qin Wang et al. CoTTA: Continual test-time adaptation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 3, 8

[58] Jinheng Xie, Weifeng Huang, Hongjiang Yu, Mike Shou, and Chi Zhang. BoxDiff: Text-to-image synthesis with training-free box-constrained diffusion. In *International Conference on Computer Vision (ICCV)*, 2023. 2

[59] Jinheng Xie, Mike Shou, et al. Show-o2: Improved native unified multimodal models. Preprint, 2025. 2

[60] Yoko Yamakata et al. Cooking recipe search by pairs of ingredient and action. In *International Conference on Multimedia Modeling*, 2017. 2

[61] Yoko Yamakata et al. A highly clean recipe dataset with ingredient states annotation. Preprint, 2025. 2

[62] Xi Yang, Aokun Chen, Nima PourNejatian, Hoo Chang Shin, Kaleb E. Smith, Christopher Parisien, Colin Compas, Cheryl Martin, Anthony B. Costa, Mona G. Flores, Ying Zhang, Tanja Magoc, Christopher A. Harle, Gloria Lipori, Duane A. Mitchell, William R. Hogan, Elizabeth A. Shenkman, Jiang Bian, and Yonghui Wu. A large language model for electronic health records. *npj Digital Medicine*, 5(1):194, 2022. 2

[63] Xiaohua Zhai, Alexander Kolesnikov, Lucas Beyer, et al. LiT: Zero-shot transfer with locked-image text tuning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 2

[64] Xiaohua Zhai, Basil Mustafa, Alexander Kolesnikov, and Lucas Beyer. SigLIP: Signal-improved language-image pre-training. *arXiv preprint arXiv:2303.15343*, 2023. 2

[65] Marvin Zhang, Sergey Levine, and Chelsea Finn. Memo: Test time robustness via adaptation and augmentation. In *Advances in Neural Information Processing Systems*, 2022. 3

[66] Renrui Zhang, Rongyao Fang, Peng Gao, Wei Zhang, Hailin Wang, Hongsheng Li, and Yu Qiao. Tip-adapter: Training-free adaption of CLIP for few-shot classification. In *European Conference on Computer Vision (ECCV)*, 2022. 2

[67] Xiaopei Zhang, Xingang Wang, and Xin Wang. A reinforcement learning-driven task scheduling algorithm for multi-tenant distributed systems, 2025. 2

[68] X. Z. Zhang and Z. Song. Dynamical preparation of a steady odlro state in the hubbard model with local non-hermitian impurity. page 174303, 2020. 2

[69] Kaiyang Zhou, Jingkang Yang, Chen Change Loy, and Ziwei Liu. Conditional prompt learning for vision-language models. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 1, 2

[70] Kaiyang Zhou, Jingkang Yang, Chen Change Loy, and Ziwei Liu. Learning to prompt for vision-language models. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 1, 2

[71] Luowei Zhou, Chenliang Xu, and Jason J. Corso. Towards automatic learning of procedures from web instructional videos. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2018. 2

[72] Fengqing Zhu, Mei Chen, and Edward Delp. An image analysis system for dietary assessment and evaluation. In *IEEE International Conference on Image Processing (ICIP)*, 2010. 2

A. Extended Method Details and Derivations

This appendix provides a more formal and self-contained specification of the training-free pipeline and its transductive (unlabeled-test) refinements. The goal is twofold: (i) to remove ambiguities in how prompts, variants, and knowledge-base aliases are instantiated and pooled, and (ii) to make the evaluation artifacts reproducible by fixing all operators in closed form.

Reminder (no training). Throughout the appendix, all backbone parameters are frozen. "Adaptation" refers only to deterministic post-hoc transforms on frozen embeddings/logits and to text-bank design; there are no gradient updates, no fine-tuning, and no learned fusion weights.

Equation count. We deliberately expand the derivations to make each step explicit. Many of these equations are alternative but equivalent views (scalar / vector / matrix) of the same operators, included for auditability.

A.1. Notation

We denote the test set by $\mathcal{D} = \{x_i\}_{i=1}^{N}$ with labels in $\mathcal{Y} = \{y_1, \dots, y_C\}$ (labels are used only for evaluation). A frozen VLM consists of an image encoder $f(\cdot)$ and a text encoder $g(\cdot)$. Embeddings are ℓ_2-normalized unless stated otherwise.

$$z_i = f(x_i) \in \mathbb{R}^d \tag{12}$$

$$u_k = g(t_k) \in \mathbb{R}^d \tag{13}$$

$$\bar{z}_i = \frac{z_i}{\|z_i\|_2} \tag{14}$$

$$\bar{u}_k = \frac{u_k}{\|u_k\|_2} \tag{15}$$

$$\ell_{i,k} = \tau\, \bar{z}_i^\top \bar{u}_k \tag{16}$$

$$\tau = \exp(\gamma) \quad \text{(frozen logit scale parameter)} \tag{17}$$

$$s_{i,c} = \mathrm{Pool}\left(\{\ell_{i,k} : t_k \in \mathcal{T}(y_c)\}\right) \tag{18}$$

$$p_{i,c} = \frac{\exp(s_{i,c})}{\sum_{c'=1}^{C} \exp(s_{i,c'})} \tag{19}$$

$$\hat{y}_i = \arg\max_c s_{i,c} \tag{20}$$

$$\mathrm{conf}(x_i) = \max_c p_{i,c} \tag{21}$$

A.2. Text bank construction as a factorized operator

For each class y_c, we form a set of surface forms (names / aliases / variants) and a set of prompt templates. Their cartesian product defines a *text bank*.

$$\mathcal{S} = \{s_m(\cdot)\}_{m=1}^{M} \quad \text{(prompt templates)} \tag{22}$$

$$\mathcal{V}(y_c) = \{v_{c,j}\}_{j=1}^{J_c} \quad \text{(name variants for class } c\text{)} \tag{23}$$

$$\mathcal{T}(y_c) = \{\, s_m(v_{c,j}) \mid 1 \le m \le M,\ 1 \le j \le J_c \,\} \tag{24}$$

$$K_c = |\mathcal{T}(y_c)| = M\, J_c \tag{25}$$

$$\mathbf{L}_i = [\ell_{i,1}, \ldots, \ell_{i,K}] \in \mathbb{R}^K \quad \text{(logits over all prompts)} \tag{26}$$

$$\mathbf{L}_{i,c} = [\ell_{i,k}]_{t_k \in \mathcal{T}(y_c)} \in \mathbb{R}^{K_c} \tag{27}$$

Prompt suites (explicit). The implementation instantiates three prompt suites: *minimal* ($s(v) = v$), *base* (generic photo prompts), and *plate* (food-aware prompts). For auditability, the exact template strings are logged; the equations below treat s_m abstractly.

A.3. Pooling over prompts: log-sum-exp family

We consider a family of pooling operators parameterized by α. This yields a continuous interpolation between mean pooling and max pooling.

$$\mathrm{LSE}_\alpha(\mathbf{a}) = \frac{1}{\alpha} \log \sum_{j=1}^{K_c} \exp(\alpha a_j) \tag{28}$$

$$m(\mathbf{a}) = \max_j a_j \tag{29}$$

$$\mathrm{LSE}_\alpha(\mathbf{a}) = m(\mathbf{a}) + \frac{1}{\alpha} \log \sum_{j=1}^{K_c} \exp(\alpha(a_j - m(\mathbf{a}))) \tag{30}$$

$$\lim_{\alpha \to 0} \mathrm{LSE}_\alpha(\mathbf{a}) = \frac{1}{K_c} \sum_{j=1}^{K_c} a_j \quad \text{(mean)} \tag{31}$$

$$\lim_{\alpha \to \infty} \mathrm{LSE}_\alpha(\mathbf{a}) = \max_j a_j \quad \text{(max)} \tag{32}$$

$$s_{i,c} = \mathrm{LSE}_\alpha(\mathbf{L}_{i,c}) \tag{33}$$

$$w_{i,c,j} = \frac{\exp(\alpha \ell_{i,c,j})}{\sum_{j'=1}^{K_c} \exp(\alpha \ell_{i,c,j'})} \quad \text{(implicit weights)} \tag{34}$$

$$\frac{\partial s_{i,c}}{\partial \ell_{i,c,j}} = w_{i,c,j} \quad \text{(sensitivity of pooled logit)} \tag{35}$$

A.4. Lexical variants as a controlled rewriting operator

Let $\nu(\cdot)$ be a canonicalization function that removes purely stylistic differences while preserving meaning (case, diacritics, punctuation, and whitespace). Unlike the previous unrolled presentation, we treat variant generation as a *set-valued* operator with an explicit budget. This avoids long chains of nearly identical equations while matching the implementation: we generate a candidate pool, canonicalize, deduplicate, then keep the top-$J_{\max}$ variants.

Base label. We start from the raw dataset label string.

$$v_{c,0} = y_c \tag{36}$$

Canonicalization. We define ν as a composition of simple idempotent string maps.

$$\nu(s) = \mathrm{ws}\big(\mathrm{punct}(\mathrm{diac}(\mathrm{lower}(s)))\big) \tag{37}$$

$$\mathrm{lower}(s) = \text{lowercase}(s) \tag{38}$$

$$\mathrm{diac}(s) = \text{strip\_diacritics}(s) \tag{39}$$

$$\mathrm{punct}(s) = \text{replace\_punctuation\_with\_space}(s) \tag{40}$$

$$\mathrm{ws}(s) = \text{collapse\_whitespace}(s) \tag{41}$$

Token view. Many matching and scoring heuristics operate on token sets.

$$\mathrm{tok}(s) = \text{set of whitespace-separated tokens of } \nu(s) \tag{42}$$

$$J(a,b) = \frac{|\mathrm{tok}(a) \cap \mathrm{tok}(b)|}{|\mathrm{tok}(a) \cup \mathrm{tok}(b)|} \tag{43}$$

Rewrite library. Let $\mathcal{R}$ be a small library of meaning-preserving rewrites (e.g., hyphen/underscore normalization, removal of bracketed descriptors, singularization rules). We consider bounded compositions of these rewrites.

$$\mathcal{R} = \{T_r\}_{r=1}^{R} \tag{44}$$

$$\mathcal{R}^{\le L_{\max}} = \bigcup_{\ell=0}^{L_{\max}} \{1,\dots,R\}^{\ell} \tag{45}$$

$$T_{\mathbf{r}} = T_{r_\ell} \circ \cdots \circ T_{r_1}, \qquad \mathbf{r} = (r_1,\dots,r_\ell) \in \{1,\dots,R\}^{\ell} \tag{46}$$

$$\mathcal{V}_{\mathrm{cand}}(y_c) = \big\{\nu(T_{\mathbf{r}}(y_c)) : \mathbf{r} \in \mathcal{R}^{\le L_{\max}}\big\} \tag{47}$$

$$|\mathcal{R}^{\le L_{\max}}| = \sum_{\ell=0}^{L_{\max}} R^{\ell} = \frac{R^{L_{\max}+1}-1}{R-1} \quad (R>1) \tag{48}$$

Deduplication. Canonicalization makes duplicate removal deterministic.

$$\mathcal{V}_{\mathrm{uniq}}(y_c) = \mathrm{Unique}\big(\mathcal{V}_{\mathrm{cand}}(y_c)\big) \tag{49}$$

Budgeted selection. We keep at most $J_{\max}$ variants per class.

$$J_c = \min\big(|\mathcal{V}_{\mathrm{uniq}}(y_c)|,\, J_{\max}\big) \tag{50}$$

To choose which J_c variants to keep, we define a lightweight lexical score that prefers close paraphrases while allowing small stylistic changes.

$$\mathrm{ED}(a,b) = \min_{\pi \in \mathcal{E}(a \to b)} |\pi| \quad \text{(edit distance via an edit script set } \mathcal{E}) \tag{51}$$

$$\Delta_{\mathrm{len}}(a,b) = \big||a| - |b|\big| \tag{52}$$

$$\mathrm{score}(v; y_c) = \lambda_J J(v, y_c) - \lambda_E \mathrm{ED}(v, y_c) - \lambda_L \Delta_{\mathrm{len}}(v, y_c) \tag{53}$$

$$\{v_{c,j}\}_{j=1}^{J_c} = \mathrm{TopJ}_{v \in \mathcal{V}_{\mathrm{uniq}}(y_c)}\ \mathrm{score}(v; y_c) \tag{54}$$

Variant-augmented class name set. The final lexical variant set used by a prompt suite is

$$\mathcal{V}(y_c) = \{y_c\} \cup \{v_{c,j}\}_{j=1}^{J_c} \tag{55}$$

$$\mathcal{T}(y_c) = \{\, s_m(v) \mid v \in \mathcal{V}(y_c),\ m = 1,\dots,M \,\} \tag{56}$$

This formulation matches the code paths for the `label_variants` options: the only difference across variants is the rewrite library $\mathcal{R}$ and the budget parameters $(L_{\max}, J_{\max})$.

A.5. Knowledge-grounded alias expansion (FoodKB)

We optionally expand a class label with aliases from a public knowledge base of dishes (FoodKB). Matching is performed using only strings (no images, no annotations), and a conservative threshold.

$$\mathrm{norm}(s) = \text{lowercase}(\text{strip\_punct}(\text{strip\_diacritics}(s))) \tag{57}$$

$$\mathrm{tok}(s) = \text{set of whitespace-separated tokens of } \mathrm{norm}(s) \tag{58}$$

$$\mathrm{sim}(a,b) = 100 \cdot \frac{|\mathrm{tok}(a) \cap \mathrm{tok}(b)|}{|\mathrm{tok}(a) \cup \mathrm{tok}(b)|} \quad \text{(token-set similarity)} \tag{59}$$

$$k^\star(c) = \arg\max_{k \in \mathcal{K}} \operatorname{sim}(y_c,\, \kappa_k) \quad \text{(best KB entry)} \tag{60}$$

$$\operatorname{match}(c) = \mathbb{I}\big[\operatorname{sim}(y_c, \kappa_{k^\star(c)}) \geq \delta\big] \tag{61}$$

$$\mathcal{A}(c) = \begin{cases} \text{top-}K \text{ aliases of } \kappa_{k^\star(c)} & \operatorname{match}(c) = 1 \\ \varnothing & \operatorname{match}(c) = 0 \end{cases} \tag{62}$$

$$\mathcal{V}(y_c) \leftarrow \mathcal{V}(y_c) \cup \mathcal{A}(c) \tag{63}$$

$$J_c \leftarrow |\mathcal{V}(y_c)| \tag{64}$$

A.6. Prompt-suite ensembles (no learned weights)

In addition to text-bank variants, we implement several deterministic ensemble rules across multiple prompt suites. Let $\mathcal{M}$ index a small set of base predictors (e.g., base/plate/cuisine-hint) producing logits $s_{i,c}^{(m)}$ and posteriors $p_{i,c}^{(m)}$.

$$s_{i,c}^{\text{mean}} = \frac{1}{|\mathcal{M}|} \sum_{m \in \mathcal{M}} s_{i,c}^{(m)} \tag{65}$$

$$s_{i,c}^{\text{logit-mean}} = \log\left(\frac{1}{|\mathcal{M}|} \sum_{m \in \mathcal{M}} \exp(s_{i,c}^{(m)})\right) \tag{66}$$

$$H_i^{(m)} = -\sum_{c=1}^{C} p_{i,c}^{(m)} \log p_{i,c}^{(m)} \quad \text{(entropy)} \tag{67}$$

$$w_i^{(m)} = \frac{\exp(-\eta H_i^{(m)})}{\sum_{m' \in \mathcal{M}} \exp(-\eta H_i^{(m')})} \quad \text{(entropy weights)} \tag{68}$$

$$s_{i,c}^{\text{ent-wt}} = \sum_{m \in \mathcal{M}} w_i^{(m)}\, s_{i,c}^{(m)} \tag{69}$$

$$m^\star(i) = \arg\max_{m \in \mathcal{M}} \max_c p_{i,c}^{(m)} \quad \text{(max-confidence selector)} \tag{70}$$

$$s_{i,c}^{\text{max-conf}} = s_{i,c}^{(m^\star(i))} \tag{71}$$

A.7. Prior correction as a label-shift style log adjustment

Prior correction adjusts logits to discourage over-predicting frequent classes under the (approximate) label-shift assumption. Although we do not assume label shift holds exactly, the correction is a simple and auditable post-hoc baseline.

$$\hat{\pi}_c = \frac{1}{N} \sum_{i=1}^{N} p_{i,c} \tag{72}$$

$$\tilde{\pi}_c = \frac{\hat{\pi}_c + \epsilon}{\sum_{c'=1}^{C} (\hat{\pi}_{c'} + \epsilon)} \tag{73}$$

$$s_{i,c}^{\text{pc}} = s_{i,c} - \lambda \log \tilde{\pi}_c \tag{74}$$

$$p_{i,c}^{\text{pc}} = \operatorname{softmax}(s_i^{\text{pc}})_c \tag{75}$$

$$\Delta s_{i,c} = s_{i,c}^{\text{pc}} - s_{i,c} = -\lambda \log \tilde{\pi}_c \tag{76}$$

$$\frac{p_{i,c}^{\text{pc}}}{p_{i,c}} = \frac{\exp(-\lambda \log \tilde{\pi}_c)}{\sum_{c'} p_{i,c'} \exp(-\lambda \log \tilde{\pi}_{c'})} \tag{77}$$

A.8. kNN smoothing as graph diffusion on frozen embeddings

We define a similarity graph over frozen embeddings and smooth posteriors by a single diffusion step or by an equivalent regularized objective. The main paper uses a single-step, chunked implementation; here we provide several equivalent forms for clarity.

$$w_{ij} = \max\{0,\, \bar{z}_i^\top \bar{z}_j\} \quad \text{(cosine affinity, clipped)} \tag{78}$$

$$\mathcal{N}_k(i) = \operatorname{TopK}_j\, w_{ij} \tag{79}$$

$$W_{ij} = \begin{cases} w_{ij} & j \in \mathcal{N}_k(i) \\ 0 & \text{otherwise} \end{cases} \tag{80}$$

$$D_{ii} = \sum_{j=1}^{N} W_{ij} \quad \text{(degree)} \tag{81}$$

$$P \in \mathbb{R}^{N \times C}, \quad P_{i,c} = p_{i,c} \tag{82}$$

$$\tilde{P} = (1 - \beta)P + \beta D^{-1} W P \tag{83}$$

$$\tilde{p}_i = \frac{\tilde{P}_{i,:}}{\sum_{c=1}^{C} \tilde{P}_{i,c}} \quad \text{(row-normalization)} \tag{84}$$

$$L = D - W \quad \text{(graph Laplacian)} \tag{85}$$

$$\mathcal{E}(Q) = \|Q - P\|_F^2 + \lambda \operatorname{Tr}(Q^\top L Q) \tag{86}$$

$$Q^\star = \arg\min_Q \mathcal{E}(Q) \tag{87}$$

$$\nabla_Q \mathcal{E}(Q) = 2(Q - P) + 2\lambda L Q \tag{88}$$

$$(I + \lambda L) Q^\star = P \tag{89}$$

$$Q^\star = (I + \lambda L)^{-1} P \tag{90}$$

Iterative view and closed-form diffusion. The single-step smoother $\tilde{P}$ can be extended to T diffusion iterations, but we avoid explicit unrolling $(P^{(1)}, P^{(2)}, \dots)$ since it yields long, repetitive series. Instead we provide (i) the recurrence, (ii) a closed form, and (iii) several standard normalization variants.

$$P^{(0)} = P \tag{91}$$

$$P^{(t+1)} = (1-\beta)P + \beta\, D^{-1} W P^{(t)} \qquad t = 0, 1, \dots \tag{92}$$

$$T_{\text{rw}} = D^{-1} W \quad \text{(random-walk transition)} \tag{93}$$

$$A = \beta T_{\text{rw}} \tag{94}$$

$$P^{(t)} = (1-\beta) \sum_{k=0}^{t-1} A^k P + A^t P^{(0)} \tag{95}$$

$$P^{(t)} = \left((1-\beta) \sum_{k=0}^{t-1} A^k + A^t \right) P \qquad \text{(since } P^{(0)} = P\text{)} \tag{96}$$

$$\sum_{k=0}^{t-1} A^k = (I - A^t)(I - A)^{-1} \qquad \text{if } I - A \text{ is invertible} \tag{97}$$

$$P^{(\infty)} = \lim_{t\to\infty} P^{(t)} = (1-\beta)(I - A)^{-1} P \tag{98}$$

$$(I - \beta T_{\text{rw}})\, P^{(\infty)} = (1-\beta) P \tag{99}$$

$$\rho(A) < 1 \implies A^t \to 0 \text{ and the limit in (98) exists} \tag{100}$$

$$\|P^{(t)} - P^{(\infty)}\|_F \le \|A^t\|_2 \, \|P^{(0)} - P^{(\infty)}\|_F \tag{101}$$

$$T_{\text{rw}} = V \Lambda V^{-1} \quad \Rightarrow \quad A^t = \beta^t V \Lambda^t V^{-1} \tag{102}$$

$$P^{(t)} = V \left((1-\beta) \sum_{k=0}^{t-1} \beta^k \Lambda^k + \beta^t \Lambda^t \right) V^{-1} P \tag{103}$$

Alternative normalizations. A common stable alternative uses symmetric normalization.

$$T_{\text{sym}} = D^{-1/2} W D^{-1/2} \tag{104}$$

$$P^{(t+1)} = (1-\beta)P + \beta\, T_{\text{sym}} P^{(t)} \tag{105}$$

$$L_{\text{norm}} = I - T_{\text{sym}} \tag{106}$$

$$P^{(t)} \approx \exp(-t\,\gamma L_{\text{norm}}) P \quad \text{(heat kernel view, small } \gamma\text{)} \tag{107}$$

Connection to Tikhonov regularization. The diffusion fixed point can be written as the solution of a linear system akin to Tikhonov smoothing.

$$Q^\star = \arg\min_Q \|Q - P\|_F^2 + \lambda \, \|Q - T_{\text{rw}} Q\|_F^2 \tag{108}$$

$$\left(I + \lambda (I - T_{\text{rw}})^\top (I - T_{\text{rw}})\right) Q^\star = P \tag{109}$$

$$Q^\star \approx (I + \lambda (I - T_{\text{rw}}))^{-1} P \quad \text{(first-order approximation)} \tag{110}$$

$$\beta = \frac{\lambda}{1+\lambda} \quad \Rightarrow \quad (I - \beta T_{\text{rw}})^{-1} \approx (1+\lambda)(I + \lambda(I - T_{\text{rw}}))^{-1} \tag{111}$$

Practical note. In the main experiments we use a small fixed T (often 1) and compute (83) efficiently in chunks; see the implementation notes below.

A.9. Prototype refinement as a soft EM procedure

Prototype refinement builds class prototypes from pseudo-posteriors on frozen embeddings, then re-scores images by prototype similarity. The process resembles one EM step in a mixture model with cosine similarity.

$$\tilde{P} \in \mathbb{R}^{N \times C} \quad \text{(pseudo-posteriors after optional smoothing)} \tag{112}$$

$$\mu_c = \frac{\sum_{i=1}^{N} \tilde{P}_{i,c}\, \bar{z}_i}{\left\| \sum_{i=1}^{N} \tilde{P}_{i,c}\, \bar{z}_i \right\|_2} \tag{113}$$

$$s_{i,c}^{\text{proto}} = \tau\, \bar{z}_i^\top \mu_c \tag{114}$$

$$p_{i,c}^{\text{proto}} = \text{softmax}(s_i^{\text{proto}})_c \tag{115}$$

EM view without long unrolling. Prototype refinement can be interpreted as one (or a few) iterations of soft EM on a mixture model over frozen embeddings. To avoid a long sequence of nearly identical iterates $(r^{(1)}, r^{(2)}, \dots)$, we provide the model, the EM objective, and the compact update rules.

vMF mixture on the unit sphere. Assume normalized embeddings $\bar{z}_i \in \mathbb{S}^{d-1}$ and unit-norm prototypes $\mu_c \in \mathbb{S}^{d-1}$. A von Mises–Fisher (vMF) component has density

$$p(\bar{z}_i \mid c) = C_d(\kappa) \exp(\kappa\, \mu_c^\top \bar{z}_i) \tag{116}$$

where $C_d(\kappa)$ is a dimension-dependent normalizer.

$$p(\bar{z}_i) = \sum_{c=1}^{C} \pi_c\, p(\bar{z}_i \mid c) \tag{117}$$

$$\mathcal{L}(\{\mu_c\}, \pi) = \sum_{i=1}^{N} \log \left(\sum_{c=1}^{C} \pi_c\, C_d(\kappa) \exp(\kappa\, \mu_c^\top \bar{z}_i) \right) \tag{118}$$

E-step responsibilities. The posterior responsibility is

$$r_{i,c} = \Pr(c \mid \bar{z}_i) = \frac{\pi_c \exp(\kappa\, \mu_c^\top \bar{z}_i)}{\sum_{c'=1}^{C} \pi_{c'} \exp(\kappa\, \mu_{c'}^\top \bar{z}_i)} \tag{119}$$

$$\pi_c = \frac{1}{N} \sum_{i=1}^{N} r_{i,c} \tag{120}$$

M-step prototypes. With fixed responsibilities, the expected complete log-likelihood for class c is proportional to

$$Q_c(\mu_c) = \sum_{i=1}^{N} r_{i,c}\, \kappa\, \mu_c^\top \bar{z}_i \quad \text{s.t. } \|\mu_c\|_2 = 1 \tag{121}$$

$$\mathcal{J}(\mu_c, \lambda) = -Q_c(\mu_c) + \lambda(\mu_c^\top \mu_c - 1) \tag{122}$$

$$\nabla_{\mu_c} \mathcal{J} = -\kappa \sum_{i=1}^{N} r_{i,c} \bar{z}_i + 2\lambda \mu_c = 0 \tag{123}$$

$$\mu_c = \frac{\sum_{i=1}^{N} r_{i,c} \bar{z}_i}{\left\| \sum_{i=1}^{N} r_{i,c} \bar{z}_i \right\|_2} \tag{124}$$

Connection to our implementation. We initialize responsibilities from the (optionally smoothed) pseudo-posteriors $\tilde{P}$.

$$r_{i,c}^{(0)} = \tilde{P}_{i,c} \tag{125}$$

The M-step is exactly the prototype computation already stated in the main appendix.

$$\mu_c^{(t)} = \frac{\sum_{i=1}^{N} r_{i,c}^{(t)} \bar{z}_i}{\left\| \sum_{i=1}^{N} r_{i,c}^{(t)} \bar{z}_i \right\|_2} \tag{126}$$

The E-step is a cosine-softmax.

$$r_{i,c}^{(t+1)} = \frac{\exp(\kappa\, \bar{z}_i^\top \mu_c^{(t)})}{\sum_{c'=1}^{C} \exp(\kappa\, \bar{z}_i^\top \mu_{c'}^{(t)})} \tag{127}$$

Regularized prototypes. To reduce drift on hard datasets, we optionally shrink image prototypes toward the original text prototype t_c.

$$t_c = \frac{g_{\text{text}}(y_c)}{\|g_{\text{text}}(y_c)\|_2} \tag{128}$$

$$\mu_c^{(t)} = \frac{\sum_{i=1}^{N} r_{i,c}^{(t)} \bar{z}_i + \lambda_0 t_c}{\left\| \sum_{i=1}^{N} r_{i,c}^{(t)} \bar{z}_i + \lambda_0 t_c \right\|_2} \tag{129}$$

Single-step "proto" predictor. After one M-step we produce proto-logits and proto-posteriors

$$s_{i,c}^{\text{proto}} = \tau\, \bar{z}_i^\top \mu_c^{(1)} \tag{130}$$

$$p_{i,c}^{\text{proto}} = \text{softmax}(s_i^{\text{proto}})_c \tag{131}$$

We can optionally fuse proto logits with the original CLIP logits $s_{i,c}$ (still training-free) via a convex combination.

$$s_{i,c}^{\text{fuse}} = (1-\alpha)s_{i,c} + \alpha s_{i,c}^{\text{proto}} \tag{132}$$

$$p_{i,c}^{\text{fuse}} = \text{softmax}(s_i^{\text{fuse}})_c \tag{133}$$

Hard-assignment limit. As $\kappa \to \infty$, responsibilities concentrate on the nearest prototype,

$$\lim_{\kappa\to\infty} r_{i,c} = \mathbb{I}\Big[c = \arg\max_{c'} \bar{z}_i^\top \mu_{c'}\Big] \tag{134}$$

recovering spherical k-means style updates.

Stability heuristics. We optionally temper the E-step and mix it with the initial pseudo-posterior to reduce drift.

$$r_{i,c}^{(t+1)} \propto \exp\Big(\frac{\kappa}{T}\,\bar{z}_i^\top \mu_c^{(t)}\Big) \tag{135}$$

$$\tilde{r}^{(t+1)} = (1-\gamma)r^{(0)} + \gamma r^{(t+1)} \tag{136}$$

$$t^\star = \min\{t : \|\mu^{(t)} - \mu^{(t-1)}\|_F \le \varepsilon\} \quad \text{(optional early stop)} \tag{137}$$

In practice, we use $t \in \{0, 1\}$ (one pass) for computational efficiency and to keep the method training-free and auditable.

A.10. Moment matching and diagonal CORAL-style alignment

Gaussian alignment whitens the test embedding distribution and re-colors it to match the prototype distribution (or vice versa). We implement a diagonal (per-dimension) version to keep the operator stable and memory-light.

$$\mu_z = \frac{1}{N}\sum_{i=1}^{N} \bar{z}_i \tag{138}$$

$$\sigma_z^2 = \frac{1}{N}\sum_{i=1}^{N} (\bar{z}_i - \mu_z) \odot (\bar{z}_i - \mu_z) \tag{139}$$

$$\mu_\mu = \frac{1}{C}\sum_{c=1}^{C} \mu_c \tag{140}$$

$$\sigma_\mu^2 = \frac{1}{C}\sum_{c=1}^{C} (\mu_c - \mu_\mu) \odot (\mu_c - \mu_\mu) \tag{141}$$

$$\text{whiten}(z) = \text{diag}(\sigma_z + \epsilon)^{-1}(z - \mu_z) \tag{142}$$

$$\text{recolor}(w) = \text{diag}(\sigma_\mu)(w) + \mu_\mu \tag{143}$$

$$\bar{z}_i^{\text{ga}} = \text{recolor}(\text{whiten}(\bar{z}_i)) \tag{144}$$

$$s_{i,c}^{\text{ga}} = \tau\,(\bar{z}_i^{\text{ga}})^\top \mu_c \tag{145}$$

A.11. Metrics: accuracy, calibration, and selective prediction

We report Top-1 and Top-5 accuracy, along with calibration (ECE) and selective prediction risk at fixed coverage. Below we define the metrics used in the benchmark script.

$$\text{Top1} = \frac{1}{N}\sum_{i=1}^{N} \mathbb{I}[\hat{y}_i = y_i] \tag{146}$$

$$\text{Top5} = \frac{1}{N}\sum_{i=1}^{N} \mathbb{I}[y_i \in \text{Top5}(\mathbf{s}_i)] \tag{147}$$

$$\text{NLL} = -\frac{1}{N}\sum_{i=1}^{N} \log p_{i,y_i} \tag{148}$$

$$\text{Brier} = \frac{1}{N}\sum_{i=1}^{N}\sum_{c=1}^{C} (p_{i,c} - \mathbb{I}[y_i = c])^2 \tag{149}$$

$$b(i) = \text{bin index of } \text{conf}(x_i) \text{ into } B \text{ bins} \tag{150}$$

$$\text{acc}(b) = \frac{1}{|I_b|}\sum_{i\in I_b} \mathbb{I}[\hat{y}_i = y_i] \tag{151}$$

$$\text{conf}(b) = \frac{1}{|I_b|}\sum_{i\in I_b} \text{conf}(x_i) \tag{152}$$

$$\text{ECE} = \sum_{b=1}^{B} \frac{|I_b|}{N}\,\big|\text{acc}(b) - \text{conf}(b)\big| \tag{153}$$

$$\pi(\alpha) = \frac{1}{N}\sum_{i=1}^{N} \mathbb{I}[\text{conf}(x_i) \ge \alpha] \quad \text{(coverage at threshold } \alpha) \tag{154}$$

$$R(\alpha) = \frac{\sum_{i=1}^{N} \mathbb{I}[\text{conf}(x_i) \ge \alpha]\,(1 - \mathbb{I}[\hat{y}_i = y_i])}{\sum_{i=1}^{N} \mathbb{I}[\text{conf}(x_i) \ge \alpha]} \quad \text{(risk)} \tag{155}$$

$$\alpha^\star = \max\{\alpha : \pi(\alpha) \ge 0.8\} \quad (\text{coverage } \ge 80\%) \tag{156}$$

$$\text{Risk@80} = R(\alpha^\star) \tag{157}$$

A.12. Bootstrap confidence intervals (Top-1)

To quantify uncertainty without additional training, we compute non-parametric bootstrap confidence intervals (CIs) for Top-1 accuracy. Let $\mathcal{I} = \{1, \dots, N\}$ index the evaluated test images.

$$\mathcal{I}^{(b)} \sim \text{UniformWithReplacement}(\mathcal{I}, N) \quad \text{for } b = 1, \dots, B \tag{158}$$

$$\text{Top1}^{(b)} = \frac{1}{N} \sum_{i \in \mathcal{I}^{(b)}} \mathbb{I}[\hat{y}_i = y_i] \tag{159}$$

$$\widehat{\text{Top1}} = \frac{1}{B} \sum_{b=1}^{B} \text{Top1}^{(b)} \tag{160}$$

$$\text{CI}_{95\%} = \left[Q_{0.025}(\{\text{Top1}^{(b)}\}),\ Q_{0.975}(\{\text{Top1}^{(b)}\}) \right] \tag{161}$$

We additionally report the bootstrap standard error (SE) and, when desired, the bias-corrected and accelerated (BCa) interval.

$$\text{SE}(\text{Top1}) = \sqrt{\frac{1}{B-1} \sum_{b=1}^{B} \left(\text{Top1}^{(b)} - \widehat{\text{Top1}} \right)^2} \tag{162}$$

$$\widehat{F}(x) = \frac{1}{B} \sum_{b=1}^{B} \mathbb{I}[\text{Top1}^{(b)} \le x] \tag{163}$$

$$z_0 = \Phi^{-1}\left(\widehat{F}(\text{Top1}_{\text{obs}}) \right), \qquad \text{Top1}_{\text{obs}} = \frac{1}{N} \sum_{i=1}^{N} \mathbb{I}[\hat{y}_i = y_i] \tag{164}$$

$$\text{Top1}_{(-i)} = \frac{1}{N-1} \sum_{j \in \mathcal{I} \setminus \{i\}} \mathbb{I}[\hat{y}_j = y_j] \qquad i = 1, \dots, N \tag{165}$$

$$\overline{\text{Top1}}_{(\cdot)} = \frac{1}{N} \sum_{i=1}^{N} \text{Top1}_{(-i)} \tag{166}$$

$$a = \frac{\sum_{i=1}^{N} \left(\overline{\text{Top1}}_{(\cdot)} - \text{Top1}_{(-i)} \right)^3}{6 \left[\sum_{i=1}^{N} \left(\overline{\text{Top1}}_{(\cdot)} - \text{Top1}_{(-i)} \right)^2 \right]^{3/2}} \tag{167}$$

$$\alpha' = \Phi\left(z_0 + \frac{z_0 + z_\alpha}{1 - a(z_0 + z_\alpha)} \right) \qquad \text{for } z_\alpha = \Phi^{-1}(\alpha) \tag{168}$$

$$\text{CI}_{95\%}^{\text{BCa}} = \left[Q_{\alpha_1'}(\{\text{Top1}^{(b)}\}),\ Q_{\alpha_2'}(\{\text{Top1}^{(b)}\}) \right], \quad (\alpha_1, \alpha_2) = (0.025, 0.9 \tag{169}$$

$$t^{(b)} = \frac{\text{Top1}^{(b)} - \text{Top1}_{\text{obs}}}{\text{SE}^{(b)}} \quad \Rightarrow \quad \text{CI from quantiles of } \{t^{(b)}\} \text{ (studentize} \tag{170}$$

$$\Delta^{(b)} = \text{Top1}^{(b)}(\text{method A}) - \text{Top1}^{(b)}(\text{method B}) \tag{171}$$

$$\text{CI}_{95\%}(\Delta) = \left[Q_{0.025}(\{\Delta^{(b)}\}), Q_{0.975}(\{\Delta^{(b)}\}) \right] \tag{172}$$

A.13. KB-match diagnostics (per-class analysis)

To understand why some culturally specific datasets are harder, we stratify per-class accuracy by whether a class is matched to FoodKB. Let $\mathcal{C}_{\text{m}}$ be the set of KB-matched classes and $\mathcal{C}_{\text{u}}$ the unmatched classes.

$$\mathcal{C}_{\text{m}} = \{c \in \{1, \dots, C\} : \text{match}(c) = 1\} \tag{173}$$

$$\mathcal{C}_{\text{u}} = \{1, \dots, C\} \setminus \mathcal{C}_{\text{m}} \tag{174}$$

$$\text{Acc}(c) = \frac{1}{N_c} \sum_{i : y_i = c} \mathbb{I}[\hat{y}_i = y_i] \tag{175}$$

$$\overline{\text{Acc}}_{\text{m}} = \frac{1}{|\mathcal{C}_{\text{m}}|} \sum_{c \in \mathcal{C}_{\text{m}}} \text{Acc}(c) \tag{176}$$

$$\overline{\text{Acc}}_{\text{u}} = \frac{1}{|\mathcal{C}_{\text{u}}|} \sum_{c \in \mathcal{C}_{\text{u}}} \text{Acc}(c) \tag{177}$$

$$\Delta_{\text{KB}} = \overline{\text{Acc}}_{\text{m}} - \overline{\text{Acc}}_{\text{u}} \tag{178}$$

A.14. Compute and memory complexity

We summarize the dominant costs for a single dataset with N images, C classes, and a total of $K = \sum_c K_c$ prompts in the text bank.

$$\text{Image feature extraction: } \mathcal{O}(N \cdot \text{Cost}(f)) \tag{179}$$

$$\text{Text feature extraction: } \mathcal{O}(K \cdot \text{Cost}(g)) \tag{180}$$

$$\text{Scoring (dense): } \mathcal{O}(NKd) \quad \text{(matrix multiply)} \tag{181}$$

$$\text{Pooling per class: } \mathcal{O}(N \sum_{c=1}^{C} K_c) \tag{182}$$

$$k\text{NN search (approx.): } \mathcal{O}(N \log N) \quad \text{(FAISS index build + query)} \tag{183}$$

$$\text{Memory for cached embeddings: } \mathcal{O}(Nd) \text{ floats} \tag{184}$$

$$\text{Memory for cached text bank: } \mathcal{O}(Kd) \text{ floats} \tag{185}$$

Practical note. The released code uses chunked computation for both (i) prompt scoring and (ii) kNN smoothing to bound peak memory.

A.15. Numerical stability, mixed precision, and chunked computation

Because the benchmark is designed to run on commodity GPUs/CPUs (including mixed precision on CUDA/MPS), we implement all reductions in numerically stable form and avoid materializing large intermediate tensors whenever possible. This subsection records the exact stable computations used in the code path so that independent re-implementations match bit-for-bit up to floating-point nondeterminism.

Stable softmax and log-softmax. Given a logit vector $\mathbf{s}_i \in \mathbb{R}^C$, define $m_i = \max_c s_{i,c}$ and compute probabilities without overflow.

$$m_i = \max_c s_{i,c} \tag{186}$$

$$\log Z_i = m_i + \log \sum_{c=1}^{C} \exp(s_{i,c} - m_i) \tag{187}$$

$$p_{i,c} = \exp(s_{i,c} - \log Z_i) \tag{188}$$

$$\log p_{i,c} = s_{i,c} - \log Z_i \tag{189}$$

$$H_i = -\sum_{c=1}^{C} p_{i,c} \log p_{i,c} \tag{190}$$

$$\mathrm{NLL}_i = -\log p_{i,y_i} \tag{191}$$

Stable log-sum-exp pooling. For each class c and image i, we pool prompt logits $\{\ell_{i,c,j}\}_{j=1}^{K_c}$ using a stable max-shift.

$$m_{i,c} = \max_{1 \le j \le K_c} \ell_{i,c,j} \tag{192}$$

$$s_{i,c} = m_{i,c} + \frac{1}{\alpha} \log \sum_{j=1}^{K_c} \exp\big(\alpha(\ell_{i,c,j} - m_{i,c})\big) \tag{193}$$

$$\text{If } \alpha = 1, \quad s_{i,c} = m_{i,c} + \log \sum_{j=1}^{K_c} \exp(\ell_{i,c,j} - m_{i,c}) \tag{194}$$

Mixed-precision casting (dtype alignment). On CUDA, the image encoder may emit `float16` embeddings while cached prototypes or text features may be stored as `float32`. All dot-products are therefore evaluated after explicit casting to a common dtype to avoid runtime errors and to keep numerical behavior predictable.

$$\mathrm{dtype}(A) = \mathrm{dtype}(B) \;\Rightarrow\; A^\top B \text{ is well-defined} \tag{195}$$

$$\bar{z}_i^{(\mathrm{fp32})} = \mathrm{cast}_{\mathrm{fp32}}(\bar{z}_i) \tag{196}$$

$$\mu_c^{(\mathrm{fp32})} = \mathrm{cast}_{\mathrm{fp32}}(\mu_c) \tag{197}$$

$$s_{i,c}^{\mathrm{proto}} = \tau\, (\bar{z}_i^{(\mathrm{fp32})})^\top \mu_c^{(\mathrm{fp32})} \tag{198}$$

$$\text{Optionally:} \quad s_{i,c}^{\mathrm{proto}} = \tau\, (\bar{z}_i^{(\mathrm{fp16})})^\top \mu_c^{(\mathrm{fp16})} \tag{199}$$

Chunked scoring to bound memory. Let $U \in \mathbb{R}^{K \times d}$ denote stacked text embeddings for all prompts (all classes and all variants) and $Z \in \mathbb{R}^{N \times d}$ stacked image embeddings. The dense score matrix $S = ZU^\top \in \mathbb{R}^{N \times K}$ can be too large to materialize when K is large. We therefore compute scores in blocks and *reduce on the fly* to class-level scores.

$$U = \begin{bmatrix} U^{(1)} \\ \vdots \\ U^{(B)} \end{bmatrix}, \qquad U^{(b)} \in \mathbb{R}^{K_b \times d}, \;\; \sum_{b=1}^{B} K_b = K \tag{200}$$

$$K_{<b} = \sum_{b'=1}^{b-1} K_{b'} \qquad \text{(prefix size, with } K_{<1} = 0) \tag{201}$$

$$S^{(b)} = Z(U^{(b)})^\top \in \mathbb{R}^{N \times K_b} \tag{202}$$

$$S_{i,K_{<b}+k} = S_{i,k}^{(b)} \qquad \forall\, i \in \{1, \dots, N\},\; k \in \{1, \dots, K_b\} \tag{203}$$

Prompt-to-class index set. Let $\phi : \{1, \dots, K\} \to \{1, \dots, C\}$ map each prompt index to its class. Then the prompt index set for class c is

$$\mathcal{I}_c = \{\, k \in \{1, \dots, K\} \mid \phi(k) = c \,\} \tag{204}$$

and the corresponding indices inside block b are

$$\mathcal{I}_c^{(b)} = \{\, k \in \{1, \dots, K_b\} \mid \phi(K_{<b} + k) = c \,\}. \tag{205}$$

Streaming class pooling. We compute class-level scores without storing S:

$$s_{i,c} = \mathrm{pool}\left(\{S_{i,k} : k \in \mathcal{I}_c\}\right) \tag{206}$$

$$s_{i,c}^{(b)} = \mathrm{pool}\left(\{s_{i,c}^{(b-1)}\} \cup \{S_{i,k}^{(b)} : k \in \mathcal{I}_c^{(b)}\}\right), \qquad s_{i,c}^{(0)} = -\infty \tag{207}$$

Peak memory (approximate). With dtype size B_{dtype} bytes, the dominant tensors are Z ($N \times d$), one text block $U^{(b)}$ ($K_b \times d$), and one score block $S^{(b)}$ ($N \times K_b$):

$$\mathrm{Mem}_{\mathrm{peak}} \approx B_{\mathrm{dtype}}\big(Nd + K_b d + NK_b\big) \tag{208}$$

Choosing K_b trades peak memory against kernel launch overhead; we pick the largest K_b that fits the device budget.

Memory accounting (bytes). If embeddings are stored in `float32`, a matrix of size $n \times d$ occupies approximately $4nd$ bytes. For mixed precision, replace 4 by 2 for `float16`.

$$\mathrm{bytes}(Z) \approx b_{\mathrm{fp}} \cdot Nd \tag{209}$$

$$\mathrm{bytes}(U) \approx b_{\mathrm{fp}} \cdot Kd \tag{210}$$

$$\mathrm{bytes}(S^{(b)}) \approx b_{\mathrm{fp}} \cdot NK_b \tag{211}$$

$$b_{\mathrm{fp}} = \begin{cases} 4 & \texttt{float32} \\ 2 & \texttt{float16} \end{cases} \tag{212}$$

$$K_b \approx \left\lceil \frac{K}{B} \right\rceil \quad \Rightarrow \max_b \mathrm{bytes}(S^{(b)}) \approx b_{\mathrm{fp}} \cdot N \left\lceil \frac{K}{B} \right\rceil \tag{213}$$