ETTA: Efficient Test-Time Adaptation for Vision-Language Models through Dynamic Embedding Updates

Hamidreza Dastmalchi[1], Aijun An[1], Ali Cheraghian[2]

[1]Dept. of Electrical Engineering and Computer Science, York University

[2]The Australian National University, Canberra, Australia

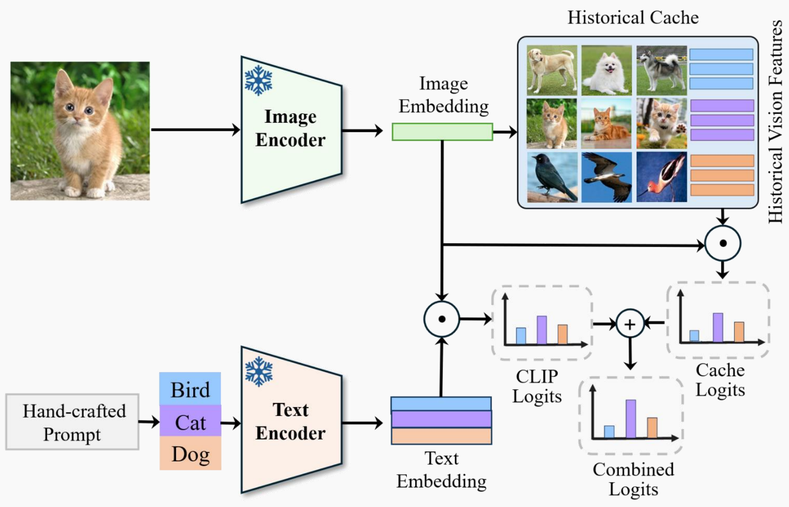
[2]Data61-CSIRO, Australia

Abstract

Pretrained vision-language models (VLMs) like CLIP perform well in zero-shot settings but struggle under distribution shifts. We introduce **Efficient Test-Time Adaptation (ETTA)** — a training-free, lightweight framework that continually refines model predictions from unlabeled test data. ETTA features a **Recursive Updating** module that integrates all incoming samples to mimic an unbounded cache and a **Dynamic Ensemble** that adaptively selects prompts and combines scores based on confidence. With minimal memory and compute cost, ETTA achieves superior accuracy and efficiency over state-of-the-art TTA methods on multiple benchmarks.

Motivation

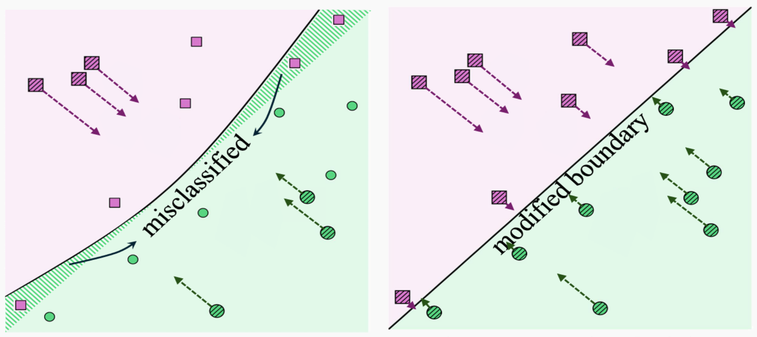
➢ **How cache-base adaptation methods work:**



➢ **Drawbacks:**

➢ Different prompt templates yield varying accuracies across images and datasets.

➢ (Left): Conventional cache-based adaptation relies on a few high-confidence prototypes, causing boundary misclassifications.



(Conventional Cache) (Recursive Update Cache)

➢ (Right): Our Recursive Update cache integrates all samples to refine the boundary and improve accuracy.

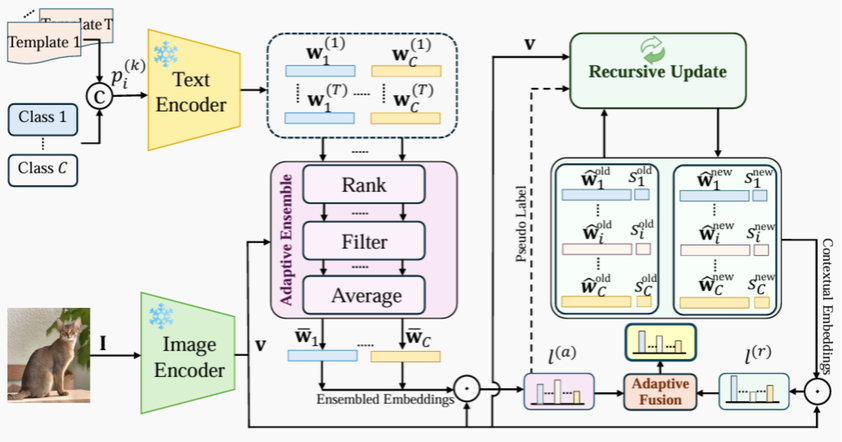
Method

Our model consists of three modules:

➢ **Adaptive Ensemble Module:** Generates and selectively aggregates diverse text embeddings from multiple prompt candidates.

➢ **Recursive Update Module:** Incrementally updates class representations using incoming test samples.

➢ **Adaptive Fusion Module:** Merges static and dynamic logits through confidence-based weighting for final prediction.



Blockdiagram of ETTA

Adaptive Ensemble Module

➢ Given T templates, we generate class-specific embeddings:

$$p_i^{(k)} = \{t^{(k)}; \text{name}_i\} \qquad \mathbf{w}_i^{(k)} = \frac{\mathrm{E_t}(p_i^{(k)})}{\|\mathrm{E_t}(p_i^{(k)})\|},$$
$$k = 1,\ldots,T, \quad i = 1,\ldots,C$$

➢ Similarly, the normalized image embedding is obtained:

$$\mathbf{v} = \frac{\mathrm{E_v}(\mathbf{I})}{\|\mathrm{E_v}(\mathbf{I})\|}$$

➢ Next, class text embeddings are ranked by cosine similarity to the image embedding, retaining only the top percentile and discarding outliers.

$$\mathcal{W}_i^{\alpha} = \left\{ \mathbf{w}_i^{(k)} \mid \langle \mathbf{w}_i^{(k)}, \mathbf{v} \rangle \geq \gamma_{\alpha} \right\}$$

➢ The ensembled class embedding is obtained by averaging and normalizing the retained text embeddings.

$$\overline{\mathbf{w}}_i = \frac{\sum_{\mathbf{w}_i^{(k)} \in \mathcal{W}_i^{\alpha}} \mathbf{w}_i^{(k)}}{\left\| \sum_{\mathbf{w}_i^{(k)} \in \mathcal{W}_i^{\alpha}} \mathbf{w}_i^{(k)} \right\|} \quad \text{for } i = 1,\ldots,C$$

➢ The ensembled is then used to compute the adaptive logit.

$$l_i^{(\mathrm{a})} = \langle \overline{\mathbf{w}}_i, \mathbf{v} \rangle, \quad \text{for } i = 1,\ldots,C$$

Recursive Update Module

➢ From Bounded Cache to Recursive Update module:

$$\sigma\left(\overline{\mathbf{w}}_i \times \mathbf{V}^T \right) \times \mathbf{V}$$

➢ We begin with a bounded cache ($\mathbf{V}$) storing the most confident prototypes for each class (i). We contextualize embeddings via cross-attention:

$$\widehat{\mathbf{w}}_i = \sigma\left(\overline{\mathbf{w}}_i \mathbf{V}^T\right) \mathbf{V}, \quad \text{for } i = 1,\ldots,C$$

➢ Reformulating the equation in vector form:

$$\widehat{\mathbf{w}}_i = \frac{\sum_{k=1}^{n} \mathbf{v}_k \exp\left(\langle \overline{\mathbf{w}}_i, \mathbf{v}_k \rangle\right)}{\sum_{k=1}^{n} \exp\left(\langle \overline{\mathbf{w}}_i, \mathbf{v}_k \rangle\right)}$$

➢ Let $\widehat{\mathbf{w}}_i^{old}$ be the contextual embedding. When a new image with pseudo-label i arrives, it updates the cache and produces $\widehat{\mathbf{w}}_i^{new}$:

$$\widehat{\mathbf{w}}_i^{\text{new}} = \frac{s_i^{\text{old}} \widehat{\mathbf{w}}_i^{\text{old}} + \exp\left(\langle \overline{\mathbf{w}}_i, \mathbf{v} \rangle\right) \mathbf{v}}{s_i^{old} + \exp\left(\langle \overline{\mathbf{w}}_j, \mathbf{v} \rangle\right)}$$

➢ where:

$$s_i^{\text{old}} = \textstyle\sum_{k=1}^{n} \exp\left(\langle \overline{\mathbf{w}}_i, \mathbf{v}_k \rangle\right)$$

➢ and:

$$s_i^{\text{new}} = s_i^{\text{old}} + \exp\left(\langle \overline{\mathbf{w}}_i, \mathbf{v} \rangle\right)$$

➢ The set of updated contextual is used to compute the corresponding scores:

$$l_i^{(\mathrm{r})} = \frac{\langle \widehat{\mathbf{w}}_i^{\text{new}}, \mathbf{v} \rangle}{\|\widehat{\mathbf{w}}_i^{\text{new}}\|} \quad \text{for } i = 1,\ldots,C$$

Adaptive Fusion Module

➢ Finally, scores from the adaptive and recursive components are combined inversely to their entropies:

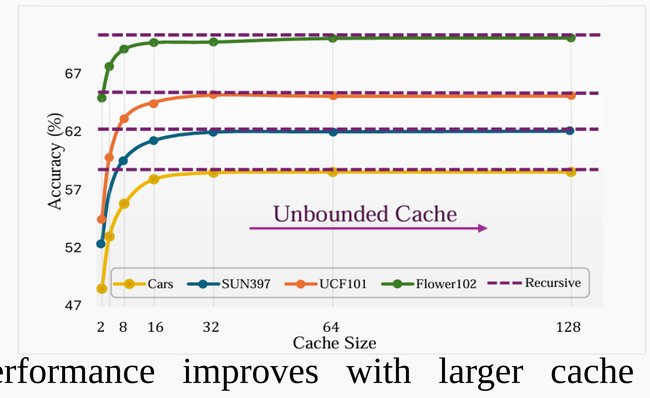
$$l_i^{(c)} = \frac{H^{(r)}}{H^{(a)} + H^{(r)}} l_i^{(a)} + \frac{H^{(a)}}{H^{(a)} + H^{(r)}} l_i^{(r)}, \text{for } i = 1,\ldots,C$$

➢ The combined logits are then used to make the final class prediction.

$$\hat{y} = \arg\max_i \left(l_i^{(c)} \right)$$

Experiments

➢ Why Mimicking an Unbounded Cache?



➢ Performance improves with larger cache sizes, approaching the accuracy of ETTA's Recursive Update cache.

Results on OOD Benchmark

	Method	ImageNet	ImageNet-A	ImageNet-V2	ImageNet-R	ImageNet-S	Average	OOD Avg
	ResNet-50 Backbone							
SRC	CLIP-ResNet-50 [51]	58.16	21.83	51.41	56.15	33.37	44.18	40.69
SRC	CLIP*-ResNet-50 [51]	59.81	23.24	52.91	60.72	35.48	46.43	43.09
FSA	CoOp [79]	63.33	23.06	55.40	56.60	34.67	46.61	42.43
FSA	CoCoOp [78]	62.81	23.32	55.72	57.74	34.48	46.81	42.82
FSA	Tip-Adapter [75]	62.03	23.13	53.97	60.35	35.74	47.04	43.30
GP	TPT [55]	60.74	26.67	54.70	59.11	35.09	47.26	43.89
GP	DiffTPT [12]	60.80	31.06	55.80	58.80	37.10	48.71	45.69
GP	TDA [31]	61.35	30.29	55.54	62.58	38.12	49.58	46.63
GP	ETTA (Ours)	61.80	31.17	55.87	62.67	38.33	49.97	47.01
CSP	DMN-ZS [76]	**63.87**	28.57	56.12	61.44	**39.84**	49.97	46.49
CSP	ETTA+ (Ours)	63.25	**31.26**	**57.05**	**62.91**	39.23	**50.74**	**47.61**
	ViT-B/16 Backbone							
SRC	CLIP-ViT-B/16 [51]	66.73	47.87	60.86	73.98	46.09	59.11	57.2
SRC	CLIP*-ViT-B/16 [51]	68.34	49.89	61.88	77.65	48.24	61.20	59.42
FSA	CoOp [79]	71.51	49.71	64.20	75.21	47.99	61.72	59.28
FSA	CoCoOp [78]	71.02	50.63	64.07	76.18	48.75	62.13	59.91
FSA	Tip-Adapter [75]	70.75	51.04	63.41	77.76	48.88	62.37	60.27
GP	TPT [55]	68.98	54.77	63.45	77.06	47.94	62.44	60.81
GP	DiffTPT [12]	70.30	55.68	65.10	75.00	46.80	62.28	60.52
GP	TDA [31]	69.51	60.11	64.67	80.24	50.54	65.01	63.89
GP	ETTA (Ours)	69.58	60.87	65.26	80.42	50.98	65.42	64.38
CSP	DMN-ZS [76]	72.25	58.28	65.17	78.55	53.20	65.49	63.80
CSP	ETTA+ (Ours)	**72.66**	**61.18**	**65.67**	**80.65**	**53.63**	**66.76**	**65.28**

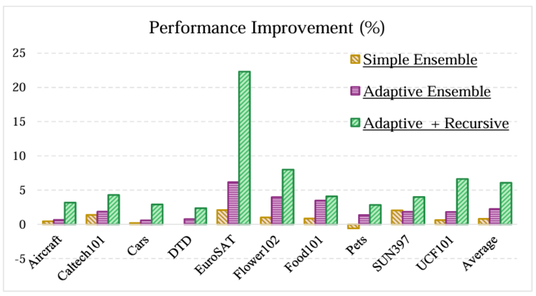
Results on Cross-Domain Benchmark

	Method	Aircraft	Caltech101	Cars	DTD	EuroSAT	Flower102	Food101	Pets	SUN397	UCF101	Average
	ResNet50 Backbone											
SRC	CLIP-ResNet50 [51]	15.66	85.88	55.70	40.37	23.69	61.75	73.97	83.57	58.80	58.84	55.82
SRC	CLIP*-ResNet50 [51]	16.11	87.26	55.89	40.37	25.79	62.77	74.82	82.97	60.85	59.48	56.63
FSA	CoOp [79]	15.12	86.53	55.32	37.29	26.20	61.55	75.59	87.00	58.15	59.05	56.18
FSA	CoCoOp [78]	14.61	87.38	56.22	38.53	28.73	65.57	76.20	88.39	59.61	57.10	57.23
GP	TPT [55]	17.58	87.02	58.46	40.84	28.33	62.69	74.88	84.49	61.46	60.82	57.66
GP	DiffTPT [12]	17.60	86.89	**60.71**	40.72	41.04	63.53	**79.21**	83.40	62.72	62.67	59.85
GP	TDA [31]	17.61	89.70	57.78	43.74	42.11	68.74	77.75	86.18	62.53	64.18	61.03
GP	ETTA (Ours)	18.84	90.18	58.60	42.73	46.01	69.75	78.05	86.42	62.79	65.48	61.88
CSP	CuPL [50]	19.59	89.29	57.28	48.64	38.38	65.44	76.94	84.84	62.55	58.97	60.31
CSP	SuS-X [59]	19.47	89.53	57.27	50.59	45.57	67.72	77.58	85.34	62.95	61.54	61.76
CSP	DMN-ZS [76]	22.77	90.14	60.02	50.41	48.72	67.93	76.70	86.78	**64.39**	65.34	63.71
CSP	ETTA+ (Ours)	**23.34**	**90.67**	60.24	**51.83**	**52.30**	**69.92**	78.17	**88.48**	63.93	**65.55**	**64.44**
	ViT-B/16 Backbone											
SRC	CLIP-ViT-B/16 [51]	23.67	93.35	65.48	44.27	42.01	67.44	83.65	88.25	62.59	65.13	63.58
SRC	CLIP*-ViT-B/16 [51]	23.22	93.55	66.11	45.04	50.42	66.99	82.86	86.92	65.63	65.16	64.59
FSA	CoOp [79]	18.47	93.70	64.51	41.92	46.39	68.71	85.30	89.14	64.15	66.55	63.88
FSA	CoCoOp [78]	22.29	93.79	64.90	45.45	39.23	70.85	83.97	90.46	66.89	68.44	64.63
GP	TPT [55]	24.78	94.16	66.87	47.75	42.44	68.98	84.67	87.79	65.50	68.04	65.10
GP	DiffTPT [12]	25.60	92.49	67.01	47.00	43.13	70.10	87.23	88.22	65.74	62.67	65.47
GP	TDA [31]	23.91	94.24	67.28	47.40	58.00	71.42	86.14	88.63	67.62	70.66	67.53
GP	ETTA (Ours)	25.65	94.74	68.40	49.64	**59.86**	73.86	86.61	90.08	68.47	71.95	68.93
CSP	DMN-ZS [76]	30.03	**95.38**	67.96	55.85	59.43	74.49	85.08	92.04	70.18	72.51	70.72
CSP	ETTA+ (Ours)	**30.18**	95.25	**68.51**	**56.20**	58.86	**77.43**	**87.38**	**92.15**	**71.11**	**73.78**	**71.09**

Time Efficiency

Method	CLIP*	TPT	DiffTPT	TDA	ETTA
Time	**18m**	19h 30m	51h 15m	25m	**19m**
Acc.	59.81	60.74	60.80	61.35	**61.80**

Contribution of Each Module



Github **Full Paper** **LinkedIn**