Appendix: Bayesian Test-Time Adaptation for Vision-Language Models

Lihua Zhou[1], Mao Ye[2], Shuaifeng Li[2], Nianxin Li[2], Xiatian Zhu[3], Lei Deng[4],
Hongbin Liu[1,5], Zhen Lei[1,5,6,7*]
[1] CAIR, HKSIS, CAS, [2] UESTC, [3] University of Surrey, [4] Shenzhen University,
[5] MAIS, Institute of Automation, CAS, [6] SAI, UCAS, [7] M.U.S.T

1. Algorithm complexity analysis

In this section, we analyze and compare the complexity of our method BCA with the baseline method CLIP. Assume that BCA uses M class embeddings $\boldsymbol{U} = [\boldsymbol{\mu}_1^T; \boldsymbol{\mu}_2^T; \cdots; \boldsymbol{\mu}_M^T] \in R^{d*M}$, where d is the is the dimension of the embedding and T represents transpose, and $\boldsymbol{V} = [P(Y|\boldsymbol{\mu}_1), P(Y|\boldsymbol{\mu}_2), \cdots, P(Y|\boldsymbol{\mu}_M)] \in R^{M*K}$ for test time adaptation. When a sample $\boldsymbol{x}_i$ arrives, BCA needs to complete the following steps to get the prediction for the sample $\boldsymbol{x}_i$:

1. Map sample $\boldsymbol{x}_i$ to the visual embedding $\boldsymbol{f}_i^v$.
2. Calculate the probability of belonging to each class embedding $P(\boldsymbol{U}|\boldsymbol{x}_i) = \texttt{Softmax}(\boldsymbol{f}_i^v * \boldsymbol{U})$.
3. Compute the final prediction by integrating the prior $P(Y|\boldsymbol{x}_i) = P(\boldsymbol{U}|\boldsymbol{x}_i) * \boldsymbol{V}$.
4. Update $\boldsymbol{U}[s]$ and $\boldsymbol{V}[s]$ when $P(\boldsymbol{U}[s]|\boldsymbol{x}_i) > \tau$, where $s = \texttt{argmax}_m P(\boldsymbol{U}[m]|\boldsymbol{x}_i)$.

For the first step, the required time t_1 is mainly affected by the visual backbone. For the second step, the required time t_2 can be divided into the calculation of $\boldsymbol{f}_i^v * \boldsymbol{U}$, with a complexity of $\boldsymbol{O}(d*M)$, and the softmax operation, with a complexity of $\boldsymbol{O}(M)$, so the overall complexity is $\boldsymbol{O}(d*M)$. For the third step, the required time t_3 is used to calculate of $P(\boldsymbol{U}|\boldsymbol{x}_i) * \boldsymbol{V}$, and its complexity is $\boldsymbol{O}(K*M)$. Compared with the first three steps, the the required time t_4 for the fourth step involves not only the completion of the calculation t_{4-cal}, but also the reading and writing of memory t_{4-rw}. The first part t_{4-cal} is used to calculate updated class embeddings and priors, which is first used to determine whether it exceeds the threshold τ, with a complexity of $\boldsymbol{O}(M)$, and then used to update $\boldsymbol{U}[s]$ and $\boldsymbol{V}[s]$, with a complexity of $\boldsymbol{O}(d+K)$. Please note that the update in the fourth step will only be executed if the required conditions are met, so its complexity is sometimes only $\boldsymbol{O}(M)$.

Since current deep networks usually require a lot of matrix operations, the first step t_1 usually takes the longest time. Generally speaking, $d > K$, which means $\boldsymbol{O}(d*M) > \boldsymbol{O}(K*M)$, that is, the time required for the second step t_2 is also greater than that for the third step t_3. In the fourth step, the update is just some very simple operations, so it takes the shortest time t_{4-cal}. Therefore, theoretically, $t_1 > t_2 > t_3 > t_{4-cal}$. To verify this, we verify it on the Cross Domain benchmark, as shown in the Table 1. The results show that while t_1 remains the longest step, consistent with our theoretical analysis, the actual running time for t_{4-cal} is significantly longer than both t_2 and t_3. Specifically, for both BCA-RN50 and BCA-ViT-B/16, t_{4-cal} takes approximately 10 seconds, which is much longer than t_3 (around 0.64s for BCA-RN50 and 0.63s for BCA-ViT-B/16) and even longer than t_2 (4.11s for BCA-RN50 and 3.18s for BCA-ViT-B/16). This discrepancy can be attributed to the following reasons:

1. **Serial Operations**: The update operations in $t_4 - cal$ involve multiple serial steps, such as condition checks and sequential updates of multiple variables. These serial operations cannot be fully parallelized, leading to increased time overhead. In contrast, t_2 and t_3 primarily involve matrix operations, which can be efficiently parallelized on the GPU. Matrix operations are highly optimized in modern deep learning frameworks, allowing them to run much faster despite their higher theoretical complexity.
2. **Kernel Launch Overhead**: Frequent kernel launches and synchronization operations in t_4 can add significant overhead. While t_2 and t_3 do not involve read and write operations to external variables; they operate on local variables, further reducing memory access overhead.

Therefore, while the theoretical complexity suggests t_{4-cal} should be the shortest, practical considerations such as serial operations and kernel launch overhead, can significantly impact the actual running time. These factors explain

Table 1. Analysis of the required time for each step on *Cross Domain* benchmark.

	t_1	t_2	t_3	t_{4-cal}	t_{4-rw}
BCA-RN50	217.50s	4.11s	0.64s	10.22s	4.19s
BCA-ViT-B/16	208.86s	3.18s	0.63s	10.25s	4.32s

*Corresponding author

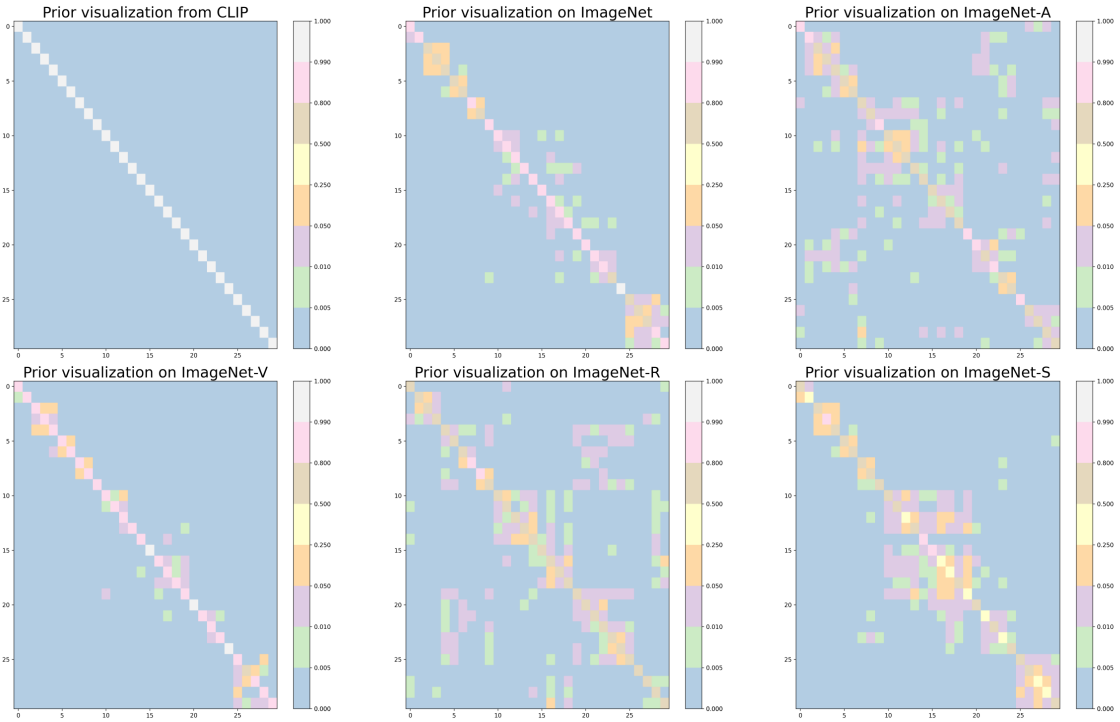

Figure 1. Prior visualization on OOD benchmark.

why t_{4-cal} is observed to be longer than both t_2 and t_3 in practice.

Compared with CLIP, which only has the first two steps and $M = K$, BCA further includes the third and fourth steps. The total time required for the third and fourth steps is less than 10% of the entire method. This also shows that our method is very efficient.

Furthermore, we analyzed the memory usage of BCA compared to CLIP. When $M = K$, BCA only requires additional storage $\boldsymbol{V}$ on the basis of CLIP. Essentially, this means storing an additional $K * K$ elements. For example, on Imagenet, which has 1000 categories, we only need to store an additional $1000 * 1000$ matrix, which amounts to $1000 * 1000 * 4bytes \approx 4M$ of extra memory. This additional memory requirement is minimal compared to the overall model size.

2. More Ablation Studies

Prior visualization. In this experiment, we visualize the prior of the model after performing TTA on ImageNet and its four variant datasets, and compared them with the fixed prior of the CLIP model. Considering that ImageNet contains a large number of categories, we selected the top 30 categories for visualization, as shown in Figure 1. Through this experiment, it can be observed that even when classifying the same categories, distribution discrepancies can lead to significant differences in the model's prior. This highlights the importance of dataset characteristics and distributions on model learning and prediction, and underscores the importance of performing prior adaptation.

Table 2. Analysis of the number of hand-crafted prompts M on *ImageNet*.

Method	$M = K$	$M = 2K$	$M = 3K$	$M = 4K$
CLIP-RN50	58.15	57.15	57.71	58.95
BCA-RN50	60.54	60.75	61.12	61.33
CLIP-ViT-B/16	66.74	66.18	67.39	67.48
BCA-ViT-B/16	69.24	69.63	69.95	70.05

Analysis of the number of hand-crafted prompts M. In this experiment, we investigate the impact of using different numbers of hand-crafted prompts on the experimental results, as described in the method section. Notably, in this experiment, we did not perform prompt ensemble as in Section 4.2. The experimental results are shown in Table 2. Specifically, when $M = k$, our hand-crafted template is $\{\texttt{a photo of a [Class } k\texttt{]}\}_{k=1}^{K}$. When $M = 2k$, our hand-crafted template is $\{\texttt{a photo of a [Class } k\texttt{]}\}_{k=1}^{K} + \{\texttt{a origami [Class } k\texttt{]}\}_{k=1}^{K}$. When $M = 3k$, our hand-crafted template is $\{\texttt{a photo of a [Class } k\texttt{]}\}_{k=1}^{K} + \{\texttt{a origami [Class } k\texttt{]}\}_{k=1}^{K} + \{\texttt{art of the [Class } k\texttt{]}\}_{k=1}^{K}$. When $M = 4k$,

our hand-crafted template is {a photo of a [Class k]}$_{k=1}^{K}$ + {a origami [Class k]}$_{k=1}^{K}$ + {art of the [Class k]}$_{k=1}^{K}$ + {itap of a [Class k]}$_{k=1}^{K}$. From the experimental results, we can observe that as M increases, the performance of BCA continues to improve, while this is not the case for CLIP [2]. The reason is that CLIP does not update class embedding to perform likelihood adaptation and prior adaptation in the new environment, so it is greatly affected by the hand-crafted prompts, while BCA is relatively less affected.

Table 3. Prior Adaptaion Analysis Combined with TDA. ImageNet: accuacy on ImageNet dataset. OOD Average: mean accuracy across four Out-of-Distribution datasets. CD Average: mean accuracy across ten Cross Domain datasets.

Method	ImageNet	OOD Average	CD Average
TDA-RN50	61.35	46.63	61.03
TDA+PA	62.09	47.52	62.23
TDA-ViT-B/16	69.51	63.89	67.53
TDA+PA	70.19	64.58	68.36

Prior Adaptaion Analysis Combined with TDA. In this work, our core strategy is the introduction of prior adaptation. To validate its generalization capability combined with other likelihood adaptation, we integrate this strategy with the existing likelihood adaptation method TDA [1], which optimizes the model's likelihood adaptation by continuously adding visual embeddings as new class embeddings. The experimental results are shown in Table 3. These results demonstrate that prior adaptation effectively enhances the generalization and adaptability of the TDA method, leading to improved performance across multiple datasets and evaluation metrics. This also proves prior adaptation has a good ability to integrate with other likelihood adaptation.

Table 4. Performance Comparison for the last 50% samples. Visual backbone: ViT-B/16. I: ImageNet; A: ImageNet-A; V: ImageNet-V2; R: ImageNet-R; S: ImageNet-S.

Method	I	A	V	R	S
CLIP	67.89	49.78	61.36	77.27	48.44
CLIP+LA	69.17	59.06	63.75	80.08	50.07
CLIP+PA	69.34	59.75	64.11	79.78	50.28
BCA	**70.68**	**61.99**	**65.48**	**81.14**	**51.36**

Performance Comparison for the last 50% samples. In this experiment, we aim to evaluate the long-term performance and adaptability of different methods by focusing on the last 50% of samples in OOD benchmark. This setup helps us understand how well these methods can adapt to new environments over time. We compared our method, BCA, with CLIP on these later samples. The results show that BCA consistently outperformed CLIP, achieving the best performance across the last 50% of samples. This indicates that BCA not only maintains its initial effectiveness but also demonstrates strong adaptability to new and evolving environments. This robust performance over time highlights BCA's potential for real-world applications where data distributions can change dynamically. In conclusion, the experimental results confirm that BCA is highly effective in adapting to new conditions, making it a promising approach for long-term deployment in vision-language modeling tasks.

Table 5. Performance Comparison on OOD benchmark. Visual backbone: ViT-L/14. I: ImageNet; A: ImageNet-A; V: ImageNet-V2; R: ImageNet-R; S: ImageNet-S.

Method	I	A	V	R	S
CLIP-ViT-L/14	74.04	53.88	67.69	87.42	63.18
TDA	76.28	61.27	68.42	**88.41**	64.67
BCA	**77.09**	**61.62**	**69.93**	88.27	**65.41**

Performance Comparisons on Larger-Scale VLMs. In this experiment, we use ViT-L/14 as the visual backbone to evaluate the performance of our proposed method, BCA, on OOD benchmark. The goal was to assess BCA's effectiveness with a lager visual backbone, which is a common requirement in real-world applications where handling complex and diverse data is essential. The experimental results, shown in Table 5, demonstrate that BCA consistently outperformed other state-of-the-art methods, such as CLIP and TDA, across four of the five tested datasets. Specifically, BCA achieved the highest accuracy on ImageNet (I), ImageNet-A (A), ImageNet-V2 (V), and ImageNet-S (S). While TDA slightly outperformed BCA on ImageNet-R (R) with an accuracy of 88.41%, BCA still maintained a high accuracy of 88.27%. These results highlight BCA's robustness and adaptability in handling OOD data. The consistent performance gains across multiple datasets indicate that BCA can effectively leverage the rich feature representations provided by large-scale pre-trained models like ViT-L/14. This is particularly important as the use of such models becomes increasingly prevalent in both research and industry.

Table 6. Ablation Study on Update Strategies for BCA. Visual backbone: ViT-B/16. Dataset: ImageNet.

Update Strategy	Accuracy (%)
Count-based	70.22
Momentum-based	70.08
Decay-based	69.92

Ablation Study on Update Strategies. We evaluated BCA's robustness across update strategies using ViT-B/16 on ImageNet. Table 6 shows that Count-based (70.22%), Momentum-based (70.08%), and Decay-based (69.92%) strategies yield comparable accuracies, with a variance of less than 0.3%. This consistency highlights BCA's robustness to different update mechanisms, reinforcing the effectiveness of its core design.

References

[1] Adilbek Karmanov, Dayan Guan, Shijian Lu, Abdulmotaleb El Saddik, and Eric Xing. Efficient test-time adaptation of vision-language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14162–14171, 2024. 3

[2] 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*, pages 8748–8763. PMLR, 2021. 3