FBTA: Enabling Single-GPU End-to-End Gigapixel WSI Classification with Feature Bridging and Translation Alignment

Supplementary Material

6. Theoretical Analysis

In this section, we investigate the feasibility of using translated features F_i^{TB} as a substitute for end-to-end features F_i^{EB} during inference. Consider first a deterministic ABMIL network without Dropout, denoted as f_{MIL}. In this setting, ReLU, Linear, and Tanh layers are globally Lipschitz continuous [4]. Although Softmax is not globally Lipschitz, its inputs are bounded by the preceding Tanh layer. Consequently, within this relevant input domain, Softmax is effectively Lipschitz continuous with a finite constant, implying that the MIL network exhibits locally Lipschitz-like behavior. The network's overall Lipschitz constant L can be theoretically upper-bounded by the product of the constants of individual layers, i.e., $L \leq \prod_{l=1}^{n} L_l$, which is a loose theoretical estimate. Under this assumption, the MIL prediction discrepancy satisfies

$$\|f_{\text{MIL}}(F_i^{TB}) - f_{\text{MIL}}(F_i^{EB})\| \leq L \cdot \|F_i^{TB} - F_i^{EB}\|, \quad (5)$$

indicating that the output error is linearly bounded by the feature alignment error.

In practice, f_{MIL} may be affected by additional factors. Nevertheless, minimizing the representation distance between translated and end-to-end features can effectively control the resulting prediction discrepancy. Indeed, as shown in Table 8, the instance-wise ℓ_2 distance between translated and end-to-end features is on the order of 10^{-4}, confirming their close alignment. Due to the local Lipschitz continuity of f_{MIL}, well-aligned translated features F_i^{TB} can yield prediction closely approximating those from the end-to-end bag F_i^{EB}. Furthermore, since F_i^{TB} is also used during training of f_{MIL}, inference using this translated bag does not introduce potential distribution shifts, thereby preserving predictive performance while substantially reducing computational cost.

7. More Implementation Details

7.1. Comparative Methods Implementation

Details of Vanilla MIL Methods. All vanilla methods (indicated by the gray row in Table 1) are trained with Adam on frozen features for up to 200 epochs. The ResNet-50 feature extractor used in this paper is identical to the version in CLAM [6], where the fourth residual block is removed, resulting in a feature dimension of 1024. Its parameters are taken from the official PyTorch library. Although this differs from the original, full ResNet-50 implementation, we still refer to it as ResNet-50 for consistency with prior works [3, 6, 7]. The ViT-B/32 feature extractor is adopted from the official PyTorch library, yielding a feature dimension of 768. An early-stopping strategy, similar to that used in DBPAug [3], is employed, and models are selected on the validation set based on the mean of AUC and accuracy. The batch size is set to 1, a common practice in multi-instance learning [5, 7, 8], because WSIs contain variable numbers of instances, which renders batch-wise parallelization infeasible. For all experiments, the learning rate is 3e-5.

Details of SSL-based MIL Methods. We pretrain the ResNet-50 and ViT-B/32 feature extractors using SimSiam [2] and DINO [1] (indicated by the gray row in Table 2), respectively. For each dataset, the feature extractor is pretrained separately, and all pretraining data come exclusively from the WSIs in the training split to avoid information leakage. To eliminate the influence of differing numbers of pretraining steps, we ensure that the extractors across datasets are trained for comparable numbers of iterations. Concretely, we use 100, 40, and 20 epochs for NSCLC-Shot20, NSCLC-Shot50, and NSCLC-Shot100, respectively, with a fixed batch size of 128. Unless otherwise specified, all remaining hyperparameters follow the official SimSiam and DINO implementations. After pretraining is completed, the feature extractors are frozen and used to generate instance-level features, which are then fed into the MIL models. The subsequent MIL training follows exactly the same configuration as described in the "Details of Vanilla MIL Methods" section.

Details of Augmentation-based MIL Methods. We employ DTFD-AFS [9] as the data augmentation method. The feature extractor used in this part is ResNet-50. Following the hyperparameters specified in the DTFD [9] paper, each WSI is divided into four pseudo-bags during training. We adapt the ABMIL, TransMIL, and MambaMIL architectures to the dual-tier MIL structure of DTFD, following the implementation of DPBAug [3]. The subsequent MIL training follows exactly the same configuration as described in the "Details of Vanilla MIL Methods".

7.2. FBTA Details

In all comparative experiments, FBTA employs exactly the same frozen feature extractor f_{FFE}, MIL network f_{MIL}, maximum training steps, optimizer, batch size, and early-stopping strategy as the corresponding baselines. To accelerate training, the trainable feature extractor f_{TFE} is initialized from f_{FFE}, and the MIL network f_{MIL} is initialized with the weights of the corresponding vanilla MIL models.

Algorithm 1 FBTA: Training One Iteration (PyTorch-style)

Require: WSI X_i, label Y_i, frozen extractor f_{FFE}, trainable extractor f_{TFE}, translator f_T, MIL network f_{MIL}, EMA models $f^*_{\text{TFE}}, f^*_{\text{MIL}}$, optimizer, pseudo-bag size M, clusters C, EMA momentum α, current epoch t, total epoch T, cross-entropy loss $\mathcal{C}(\cdot)$, cosine similarity loss $\phi(\cdot)$

1. Multi-view bag formation
X_i^{FB} = split_into_patches(X_i)
$F_i^{FB} = f_{\text{FFE}}(X_i^{FB})$
clusters = K-Means(F_i^{FB}, C)
X_i^{SB}, F_i^{SB} = stratified_sample(X_i^{FB}, F_i^{FB}, clusters, M)
X_i^{AB} = horizontal_flip(X_i^{SB}, p=0.5)
$F_i^{EB} = f_{\text{TFE}}(X_i^{AB})$
$F_i^{TB} = \frac{1}{K}\sum_{k=1}^{K} f_T[k](F_i^{SB})$

2. Bag classification
$\tilde{Y}_i^{FB} = f_{\text{MIL}}(F_i^{FB})$
$\tilde{Y}_i^{EB} = f_{\text{MIL}}(F_i^{EB})$
$\tilde{Y}_i^{TB} = f_{\text{MIL}}(F_i^{TB})$

3. Teacher predictions (no grad)
$\hat{Y}_i^{FB} = f^*_{\text{MIL}}(F_i^{FB})$
$\hat{Y}_i^{EB} = f^*_{\text{MIL}}(F_i^{EB})$
$\hat{Y}_i^{TB} = f^*_{\text{MIL}}(F_i^{TB})$
$F_i^{SB,*} = f^*_{\text{TFE}}(X_i^{SB})$

4. Loss computation
$L_{i,\text{cls}} = \mathcal{C}(\tilde{Y}_i^{FB}, Y_i) + \mathcal{C}(\tilde{Y}_i^{EB}, Y_i) + \mathcal{C}(\tilde{Y}_i^{TB}, Y_i)$
$L_{i,\text{cls}^*} = \phi(\tilde{Y}_i^{FB}, \hat{Y}_i^{FB}) + \phi(\tilde{Y}_i^{EB}, \hat{Y}_i^{EB}) + \phi(\tilde{Y}_i^{TB}, \hat{Y}_i^{TB})$
$L_{i,\text{dist}}^{TB} = \frac{1}{K}\sum_{k=1}^{K} \mathcal{L}_2\big(f_T[k](F_i^{SB}),\ F_i^{SB,*}\big)$
$L_i = L_{i,\text{dist}}^{TB} + L_{i,\text{cls}} + \frac{t}{T} \cdot L_{i,\text{cls}^*}$

# 5. Optimization
optimizer.zero_grad()
L_i.backward()
optimizer.step()

6. EMA update
$f^*_{\text{TFE}} \leftarrow \alpha f^*_{\text{TFE}} + (1-\alpha) f_{\text{TFE}}$
$f^*_{\text{MIL}} \leftarrow \alpha f^*_{\text{MIL}} + (1-\alpha) f_{\text{MIL}}$

Ensure: Updated $f_{\text{TFE}}, f_T, f_{\text{MIL}}$ and EMA models ($f^*_{\text{TFE}}, f^*_{\text{MIL}}$)

f_T consists of $K = 10$ residual MLPs, each composed of a linear layer, a ReLU activation, a dropout layer, and another linear layer. The input and output dimensions of the first and second linear layers match the feature dimension d extracted by the feature extractor ($d = 1024$ for ResNet-50 and $d = 768$ for ViT-B/32), while the hidden dimension—i.e., the output of the first linear layer and the input of the second linear layer—is set to $d/4$. f_T is initialized using PyTorch's default initialization during training. For augmentation on X_i^{SB}, horizontal flipping is applied. During training, the EMA momentum for f^*_{MIL} and f^*_{TFE} is set to 0.6 to balance convergence speed and smoothing, allowing the EMA models to adapt more quickly from their pretrained initialization. We summarize the key steps of one FBTA training iteration in Algorithm 1. Additional implementation details will be provided in the code to be released.

8. More Results

8.1. Statistical Significance of FBTA Performance Improvements

Table 9 presents the statistical significance analysis of the performance improvements achieved by FBTA over the vanilla MIL methods (corresponding to Table 1). When using ResNet-50 as the feature extractor, FBTA achieves statistical significance ($p < 0.05$) in 29 out of 36 evaluation metrics. When using ViT-B/32, FBTA remains statistically significant in 17 out of 36 metrics. Moreover, even in the scenarios where the improvements do not reach statistical significance, FBTA still consistently outperforms the corresponding baselines in terms of the mean performance across 8 random seeds. These results collectively demonstrate that the performance gains introduced by FBTA are stable, reliable, and robust across different MIL architectures, datasets, and feature extractors.

Dataset	ABMIL			TransMIL			MambaMIL		
	AUC	ACC	F1	AUC	ACC	F1	AUC	ACC	F1
STAD	✗	✓	✓	✗	✓	✓	✗	✓	✗
NSCLC-Shot20	✓	✓	✗	✓	✓	✗	✓	✓	✓
NSCLC-Shot50	✓	✓	✓	✓	✓	✓	✗	✓	✓
NSCLC-Shot100	✓	✓	✓	✓	✓	✓	✓	✓	✓

(a) Using ResNet-50 as the feature extractor.

Dataset	ABMIL			TransMIL			MambaMIL		
	AUC	ACC	F1	AUC	ACC	F1	AUC	ACC	F1
STAD	✗	✗	✓	✗	✗	✗	✗	✗	✗
NSCLC-Shot20	✓	✗	✗	✓	✓	✗	✗	✗	✗
NSCLC-Shot50	✓	✗	✗	✓	✓	✓	✓	✓	✗
NSCLC-Shot100	✓	✓	✓	✓	✓	✓	✓	✗	✗

(b) Using ViT-B/32 as the feature extractor.

Table 9. Statistical significance analysis of FBTA performance gains over vanilla MIL methods using t-tests, corresponding to the data in Table 1. A checkmark (✓) indicates $p < 0.05$, while a cross (✗) indicates $p \geq 0.05$.

8.2. More Ablation Studies

Figure 6 presents an ablation analysis of the impact of different feature views on the convergence of FBTA, using the same experimental setup as in Table 5. As observed, using only the end-to-end features results in limited performance

on the validation set and slow convergence. Incorporating the frozen features and translated features sequentially further improves both convergence speed and classification performance, which is consistent with the observations reported in Table 5.

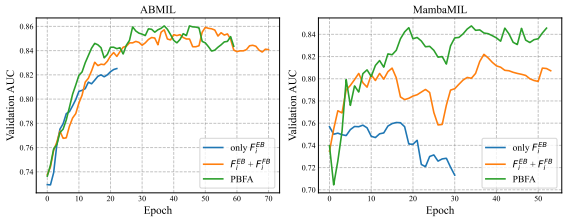

Figure 6. Convergence comparison across different feature views. Validation AUC on the NSCLC-Shot20 dataset (random seed = 0, using ImageNet-pretrained ResNet-50) is reported, with early stopping applied.

References

[1] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9650–9660, 2021. 4, 1

[2] Xinlei Chen and Kaiming He. Exploring simple siamese representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 15750–15758, 2021. 4, 1

[3] Jiuyang Dong, Junjun Jiang, Kui Jiang, Jiahan Li, Linghan Cai, and Yongbing Zhang. Disentangled pseudo-bag augmentation for whole slide image multiple instance learning. *IEEE Transactions on Medical Imaging*, 2025. 2, 1

[4] Allen A Goldstein. Optimization of lipschitz continuous functions. *Mathematical Programming*, 13(1):14–22, 1977. 4, 1

[5] Maximilian Ilse, Jakub Tomczak, and Max Welling. Attention-based deep multiple instance learning. In *International conference on machine learning*, pages 2127–2136. PMLR, 2018. 5, 1

[6] Ming Y Lu, Drew FK Williamson, Tiffany Y Chen, Richard J Chen, Matteo Barbieri, and Faisal Mahmood. Data-efficient and weakly supervised computational pathology on whole-slide images. *Nature biomedical engineering*, 5(6):555–570, 2021. 1, 4

[7] Zhuchen Shao, Hao Bian, Yang Chen, Yifeng Wang, Jian Zhang, Xiangyang Ji, et al. Transmil: Transformer based correlated multiple instance learning for whole slide image classification. *Advances in neural information processing systems*, 34:2136–2147, 2021. 1, 5

[8] Shu Yang, Yihui Wang, and Hao Chen. Mambamil: Enhancing long sequence modeling with sequence reordering in computational pathology. In *International conference on medical image computing and computer-assisted intervention*, pages 296–306. Springer, 2024. 5, 1

[9] Hongrun Zhang, Yanda Meng, Yitian Zhao, Yihong Qiao, Xiaoyun Yang, Sarah E Coupland, and Yalin Zheng. Dtfd-mil: Double-tier feature distillation multiple instance learning for histopathology whole slide image classification. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 18802–18812, 2022. 4, 6, 1