MECFormer: Multi-task Whole Slide Image Classification with Expert Consultation Network (Supplementary Material)

Doanh C. Bui and Jin Tae Kwak(✉)

School of Electrical Engineering, Korea University, Seoul, Republic of Korea
{doanhbc,jkwak}@korea.ac.kr

A Overview

In these Supplementary Materials, we provide more details regarding the design of the generative transformer of MECFormer in terms of building vocabulary (B.1), self-attention mechanisms in the encoder $\mathcal{E}(\cdot)$ (B.2), and the decoder $\mathcal{D}(\cdot)$ (B.3). Then, we report ablation results per task/dataset for three designs of projection layers: $\mathcal{P}_1$, $\mathcal{P}_T$ and $\mathcal{P}_{\texttt{ECN}}$ (C.1) and the effectiveness of language decoder $\mathcal{D}(\cdot)$ (C.2). We also take a closer look at how the process of *Preliminary Consultation* works in MECFormer.

B MECFormer

B.1 Vocabulary

As mentioned in the main manuscript, we aim to adopt a language decoder to predict the textual diagnostic term for an input WSI. First of all, a vocabulary is built. Given that the t^{th} task has n_t diagnostic categories $C_t = \{c_i^t\}_{i=1}^{n_t}$, we map each category c_i^t to the natural term S_i^t. The term S_i^t is then tokenized into n_i^t words, i.e., $S_i^t = \{s_j^t\}_{j=1}^{n_i^t}$. Then, for each t^{th} task, we form the t^{th} vocabulary, which has $n_{\text{voc}}^t = \sum_{i=1}^{n_t} n_i^t$ words, i.e., $V_t = \{s_j^t\}_{j=1}^{n_{\text{voc}}^t}$. Overall, for T tasks, we build the vocabulary to include $n_{\text{voc}} = \sum_{t=1}^{T} n_{\text{voc}}^t$, i.e., $V = \{s_i\}_{i=1}^{n_{\text{voc}}}$.

B.2 Visual Encoder $\mathcal{E}(\cdot)$

For the design of the visual encoder $\mathcal{E}(\cdot)$, we use Nyström Self-Attention [2] to handle thousands of tokens, which we denote as $\text{NA}(\cdot)$. Given queries $\mathbf{Q} \in \mathbb{R}^{N \times d}$, keys $\mathbf{K} \in \mathbb{R}^{N \times d}$, and values $\mathbf{V} \in \mathbb{R}^{N \times d}$, where N is large, $\mathbf{K}$ and $\mathbf{V}$ are first divided into k segments, each segment having N' tokens, where $N' \ll N$. These tokens are computed as the mean per channel dimension, forming *landmarks*, denoted as $\tilde{\mathbf{K}}$ and $\tilde{\mathbf{V}}$. Then, $\text{NA}(\cdot)$ is formulated as follows:

$$\begin{gathered}\tilde{F} = \texttt{softmax}\Big(\frac{\mathbf{Q}\tilde{\mathbf{K}}^T}{\sqrt{d_q}}\Big), \tilde{A} = \texttt{softmax}\Big(\frac{\tilde{\mathbf{Q}}\tilde{\mathbf{K}}^T}{\sqrt{d_q}}\Big)^+, \tilde{B} = \texttt{softmax}\Big(\frac{\tilde{\mathbf{Q}}\mathbf{K}^T}{\sqrt{d_q}}\Big), \\ \tilde{F} \in \mathbb{R}^{N\times N'}, \tilde{A} \in \mathbb{R}^{N'\times N'}, \tilde{B} \in \mathbb{R}^{N'\times N}, \\ \mathbf{H} = (\tilde{F} \times \tilde{A}) \times (\tilde{B} \times \mathbf{V}),\end{gathered} \tag{1}$$

where $(\cdot)^+$ denotes Moore–Penrose inverse operation.

B.3 Language Decoder $\mathcal{D}(\cdot)$

Learnable positional encoding. For positional encoding (PE), we implement the version of learnable positional encoding. Herein, PE is an $L \times d_{model}$ matrix, where L is the length of the sequence of tokens (i.e., word embeddings) to be fed into the language decoder $\mathcal{D}(\cdot)$. PE is initialized the same as in the study by [1], which adopts sine and cosine functions of different frequencies:

$$\begin{aligned}\text{PE}_{(i,2j)} &= \sin\Big(i \cdot (10000^{2j/d_{model}})^{-1}\Big), \\ \text{PE}_{(i,2j+1)} &= \cos\Big(i \cdot (10000^{2j/d_{model}})^{-1}\Big),\end{aligned} \tag{2}$$

where $\text{PE}_{(i,j)}$ indicates the element at the j^{th} dimension of the i^{th} token. Hence, each dimension in PE is represented as a sinusoid, with the interval of wavelengths ranging as a geometric progression from 2π to $2\pi \cdot 10000$. After being initialized by 2, PE is then learnable, i.e., it is trained along with the whole network.

Multi-head self- and cross-attention (MHSA and MHCA). For the design of the language decoder $\mathcal{D}(\cdot)$, we utilize vanilla Multi-head Self-attention (Masked_MHSA) to compute self-attention between hidden tokens, where Multi-head Cross-attention (MHCA) to compute visual-text correspondences between hidden tokens and visual encoded tokens from the encoder $\mathcal{E}(\cdot)$. Given hidden states $\mathbf{h}^{(l-1)}$ at l^{th} decoder layer, multi-head self-attention are computed as follows:

$$\begin{aligned}&\mathbf{Q}^{(h)} = W^{Q,h}\mathbf{h}^{(l-1),(h)}, \mathbf{K}^{(h)} = W^{K,h}\mathbf{h}^{(l-1),(h)}, \mathbf{V}^{(h)} = W^{V,h}\mathbf{h}^{(l-1),(h)} \\ &\mathbf{h}^{(l),(h)} = \texttt{Softmax}\Big(\frac{\mathbf{Q}^{(h)}\mathbf{K}^{(h)^T}}{\sqrt{d/N_h}} + \mathcal{M}\Big) \times \mathbf{V}^{(h)}, \\ &\mathbf{h}^{(l)} = \texttt{Concatenate}\Big(\Big\{\mathbf{h}^{(l),(h)}\Big\}_{h=1}^{N_h}\Big),\end{aligned} \tag{3}$$

where N_h is the number of heads. $W^{Q,h}$, $W^{K,h}$, and $W^{V,h}$ are learnable matrices that project $\mathbf{h}^{(l-1)}$ to $\mathbf{Q}^{(h)}$, $\mathbf{K}^{(h)}$, and $\mathbf{V}^{(h)}$ at the h^{th} head. $\mathcal{M}$ is an $N \times N$ matrix. To mask the future words, the elements in $\mathcal{M}$ are formed as below:

$$M_{ij} = \begin{cases}0, & \text{if } i \le j \\ -\inf, & \text{otherwise,}\end{cases} \tag{4}$$

where M_{ij} is an element that indicates the masking status between the i^{th} and j^{th} tokens.

Then, given visual encoded tokens $\mathbf{v}$ from the encoder, along with hidden states $\mathbf{h}^{(l)}$, MHCA is computed as follows:

$$\begin{aligned}
&\mathbf{Q}^{(h)} = W^{Q,h}\mathbf{h}^{(l),(h)}, \mathbf{K}^{(h)} = W^{K,h}\mathbf{v}^{(h)}, \mathbf{V}^{(h)} = W^{V,h}\mathbf{v}^{(h)} \\
&\mathbf{h}^{(l),(h)} = \texttt{Softmax}\left(\frac{\mathbf{Q}^{(h)}\mathbf{K}^{(h)^T}}{\sqrt{d/N_h}}\right) \times \mathbf{V}^{(h)}, \\
&\mathbf{h}^{(l)} := \texttt{Concatenate}\left(\left\{\mathbf{h}^{(l),(h)}\right\}_{h=1}^{N_h}\right),
\end{aligned} \tag{5}$$

For simplication, we use the same notations regarding to $W^{Q,h}$, $W^{K,h}$ and $W^{V,h}$ in Eq. 3 and Eq. 5. However, they are not shared in practice. N_h is set to 8 for both MHSA and MHCA.

C Extensive Ablation Results

Ablation results are reported in overall metrics in the main manuscript. Herein, we report the results per task, to further ensure the effectiveness of MECFormer in multi-task WSI classification.

C.1 Effectiveness of ECN

Table 1 reports the results of three projection designs ($\mathcal{P}_1$, $\mathcal{P}_T$, and $\mathcal{P}_{\texttt{ECN}}$) per task. The first observation is that using either a single projection for all tasks or a single projection per task results in some invalid terms. Consequently, the F1, Recall, and Precision metrics are significantly lower compared to MECFormer.

Using CTransPath as the feature extractor $\mathcal{G}(\cdot)$, across all tasks, a single projection layer ($\mathcal{P}_1$) results in drops of 0.005%∼7.493% in Acc, 0.161∼0.374 in F1, 0.162∼0.383 in Recall, and 0.159∼0.363 in Precision. For the design of a single projection per task ($\mathcal{P}_T$), there are also drops of 1.261∼7.229 in Acc, 0.038∼0.372 in F1, 0.031∼0.379 in Recall, and 0.045∼0.366 in Precision.

Employing UNI as the feature extractor $\mathcal{G}(\cdot)$, performance drops are still observed. With a single projection layer ($\mathcal{P}_1$), there are drops of 1.180%∼3.563% in Acc, 0.184∼0.289 in F1, 0.187∼0.289 in Recall, and 0.178∼0.287 in Precision. The setting of a projection per task ($\mathcal{P}_T$) shows larger drop margins: 1.630%∼6.623% in Acc, 0.029∼0.379 in F1, 0.032∼0.385 in Recall, and 0.020∼0.374 in Precision.

C.2 Effectiveness of the language decoder $\mathcal{D}$

Table 2 reports the results with and without using the language decoder $\mathcal{D}$ for each task. As shown, even without the language decoder $\mathcal{D}$, there are no invalid predictions when evaluating per task. These results are surprising and further confirm the effectiveness of ECN. However, performance drops are still observed when the decoder $\mathcal{D}$ is removed, such as a decrease from 0.396∼3.061

Table 1: Ablation results per task on the three projection layer: $\mathcal{P}_1$, $\mathcal{P}_T$, and $\mathcal{P}_{\texttt{ECN}}$.

Dataset	Setting	Acc	F1	Recall	Precision
CTransPath					
CAMELYON-16	$\mathcal{P}_1$	86.822 (±5.590)	0.770 (±0.203)	0.764 (±0.188)	0.796 (±0.220)
	$\mathcal{P}_T$	90.698 (±1.343)	0.900 (±0.013)	0.895 (±0.014)	0.911 (±0.028)
	$\mathcal{P}_{\texttt{ECN}}$	**94.315 (±1.614)**	**0.938 (±0.018)**	**0.926 (±0.020)**	**0.956 (±0.014)**
TCGA-BRCA	$\mathcal{P}_1$	92.818 (±3.326)	0.554 (±0.103)	0.545 (±0.113)	0.570 (±0.106)
	$\mathcal{P}_T$	93.251 (±2.731)	0.699 (±0.204)	0.707 (±0.237)	0.730 (±0.152)
	$\mathcal{P}_{\texttt{ECN}}$	**94.512 (±2.712)**	**0.903 (±0.047)**	**0.898 (±0.077)**	**0.916 (±0.026)**
TCGA-ESCA	$\mathcal{P}_1$	87.998 (±7.303)	0.649 (±0.281)	0.644 (±0.286)	0.657 (±0.274)
	$\mathcal{P}_T$	86.775 (±7.653)	0.779 (±0.227)	0.778 (±0.235)	0.787 (±0.217)
	$\mathcal{P}_{\texttt{ECN}}$	**94.004 (±3.739)**	**0.939 (±0.038)**	**0.940 (±0.041)**	**0.941 (±0.034)**
TCGA-NSCLC	$\mathcal{P}_1$	89.428 (±1.827)	0.555 (±0.078)	0.547 (±0.085)	0.569 (±0.072)
	$\mathcal{P}_T$	89.726 (±3.342)	0.557 (±0.099)	0.551 (±0.103)	0.566 (±0.097)
	$\mathcal{P}_{\texttt{ECN}}$	**92.962 (±1.880)**	**0.929 (±0.019)**	**0.930 (±0.017)**	**0.932 (±0.020**
TCGA-RCC	$\mathcal{P}_1$	96.155 (±0.653)	0.796 (±0.126)	0.796 (±0.145)	0.798 (±0.109)
	$\mathcal{P}_T$	94.627 (±2.878)	0.702 (±0.026)	0.693 (±0.022)	0.713 (±0.029)
	$\mathcal{P}_{\texttt{ECN}}$	**96.160 (±1.744)**	**0.957 (±0.032)**	**0.960 (±0.030)**	**0.957 (±0.035)**
UNI					
CAMELYON-16	$\mathcal{P}_1$	95.090 (±3.227)	0.692 (±0.261)	0.687 (±0.260)	0.699 (±0.263)
	$\mathcal{P}_T$	95.607 (±3.133)	0.952 (±0.035)	0.944 (±0.042)	0.966 (±0.022)
	$\mathcal{P}_{\texttt{ECN}}$	**98.191 (±1.630)**	**0.981 (±0.024)**	**0.976 (±0.037)**	**0.986 (±0.055)**
TCGA-BRCA	$\mathcal{P}_1$	92.855 (±1.905)	0.647 (±0.217)	0.657 (±0.236)	0.642 (±0.200)
	$\mathcal{P}_T$	92.460 (±2.027)	0.594 (±0.020)	0.603 (±0.040)	0.590 (±0.040)
	$\mathcal{P}_{\texttt{ECN}}$	**94.090 (±0.448)**	**0.897 (±0.005)**	**0.893 (±0.006)**	**0.908 (±0.003)**
TCGA-ESCA	$\mathcal{P}_1$	92.218 (±4.486)	0.727 (±0.206)	0.722 (±0.211)	0.737 (±0.196)
	$\mathcal{P}_T$	86.775 (±8.254)	0.768 (±0.177)	0.762 (±0.183)	0.783 (±0.172)
	$\mathcal{P}_{\texttt{ECN}}$	**93.398 (±2.799)**	**0.934 (±0.028)**	**0.939 (±0.026)**	**0.934 (±0.025)**
TCGA-NSCLC	$\mathcal{P}_1$	90.222 (±3.585)	0.657 (±0.244)	0.654 (±0.247)	0.661 (±0.243)
	$\mathcal{P}_T$	90.549 (±1.548)	0.559 (±0.092)	0.554 (±0.095)	0.566 (±0.090)
	$\mathcal{P}_{\texttt{ECN}}$	**93.785 (±2.776)**	**0.938 (±0.028)**	**0.939 (±0.026)**	**0.940 (±0.025)**
TCGA-RCC	$\mathcal{P}_1$	93.857 (±2.874)	0.775 (±0.131)	0.783 (±0.159)	0.773 (±0.110)
	$\mathcal{P}_T$	93.099 (±6.394)	0.707 (±0.245)	0.714 (±0.266)	0.703 (±0.229)
	$\mathcal{P}_{\texttt{ECN}}$	**96.930 (±1.748)**	**0.959 (±0.027)**	**0.970 (±0.029)**	**0.951 (±0.032)**

in Acc, 0.009∼0.037 in F1 score, and 0.024∼0.036 in Recall. Regarding Precision, MECFormer shows a slight drop of 0.005 on TCGA-BRCA dataset. When using UNI as a feature extractor, removing $\mathcal{D}$ results in drops across all metrics, such as a decrease from 1.261%∼1.809% in Acc, 0.013∼0.022 in F1 score, 0.014∼0.022 in Recall, and 0.009∼0.027 in Precision.

Table 2: Ablation results per task on the language decoder $\mathcal{D}$.

Dataset	Setting	Acc	F1	Recall	Precision
CTransPath					
CAMELYON-16	MECFormer w/o $\mathcal{D}$	92.248 (±1.550)	0.915 (±0.018)	0.901 (±0.018)	0.939 (±0.015)
	MECFormer	**94.315 (±1.614)**	**0.938 (±0.018)**	**0.926 (±0.020)**	**0.956 (±0.014)**
TCGA-BRCA	MECFormer w/o $\mathcal{D}$	94.116 (±1.903)	0.894 (±0.027)	0.874 (±0.037)	**0.921 (±0.020)**
	MECFormer	**94.512 (±2.712)**	**0.903 (±0.047)**	**0.898 (±0.077)**	0.916 (±0.026)
TCGA-ESCA	MECFormer w/o $\mathcal{D}$	90.411 (±5.478)	0.902 (±0.056)	0.903 (±0.060)	0.903 (±0.053)
	MECFormer	**94.004 (±3.739)**	**0.939 (±0.038)**	**0.940 (±0.041)**	**0.941 (±0.034)**
TCGA-NSCLC	MECFormer w/o $\mathcal{D}$	90.606 (±0.869)	0.906 (±0.009)	0.906 (±0.008)	0.908 (±0.010)
	MECFormer	**92.962 (±1.880)**	**0.929 (±0.019)**	**0.930 (±0.017)**	**0.932 (±0.020**
TCGA-RCC	MECFormer w/o $\mathcal{D}$	93.099 (±5.002)	0.925 (±0.050)	0.946 (±0.021)	0.918 (±0.066)
	MECFormer	**96.160 (±1.744)**	**0.957 (±0.032)**	**0.960 (±0.030)**	**0.957 (±0.035)**
UNI					
CAMELYON-16	MECFormer w/o $\mathcal{D}$	96.382 (±1.613)	0.961 (±0.018)	0.956 (±0.021)	0.967 (±0.012)
	MECFormer	**98.191 (±1.630)**	**0.981 (±0.024)**	**0.976 (±0.037)**	**0.986 (±0.055)**
TCGA-BRCA	MECFormer w/o $\mathcal{D}$	92.829 (±1.663)	0.875 (±0.028)	0.876 (±0.049)	0.881 (±0.044)
	MECFormer	**94.090 (±0.448)**	**0.897 (±0.005)**	**0.893 (±0.006)**	**0.908 (±0.003)**
TCGA-ESCA	MECFormer w/o $\mathcal{D}$	91.613 (±2.747)	0.915 (±0.028)	0.917 (±0.033)	0.916 (±0.025)
	MECFormer	**93.398 (±2.799)**	**0.934 (±0.028)**	**0.939 (±0.026)**	**0.934 (±0.025)**
TCGA-NSCLC	MECFormer w/o $\mathcal{D}$	92.493 (±2.680)	0.925 (±0.027)	0.925 (±0.027)	0.926 (±0.027)
	MECFormer	**93.785 (±2.776)**	**0.938 (±0.028)**	**0.939 (±0.026)**	**0.940 (±0.025)**
TCGA-RCC	MECFormer w/o $\mathcal{D}$	95.398 (±4.138)	0.946 (±0.050)	0.953 (±0.053)	0.942 (±0.046)
	MECFormer	**96.930 (±1.748)**	**0.959 (±0.027)**	**0.970 (±0.029)**	**0.951 (±0.032)**

C.3 Analysis of Preliminary Consultation

To provide a closer look at the process of *Preliminary Consultation*, i.e., how the router $\mathcal{R}(\cdot)$ specifies the contributions of experts to a given task, we present a visualization to show how the weights are produced per task. As we perform the *Expert Assignment* process, we also show how the weights change after multiplying by the scale hyperparameter γ in Fig. 1. As shown, before *Expert Assignment*, the router works well at recognizing tasks, i.e., specifying which expert should provide the most knowledge for the CAMELYON16, TCGA-BRCA, and TCGA-ESCA datasets. In these cases, the router assigns the highest weights to the expert corresponding to the input WSI. However, regarding TCGA-RCC and TCGA-NSCLC, the TCGA-BRCA expert seems to be specified to provide the most knowledge, as its weight appears to be the highest. From our perspective, this is understandable. As observed in Fig. 4 in the main manuscript, the samples of TCGA-BRCA, TCGA-NSCLC, and TCGA-RCC are the most overlapped in terms of raw features, while CAMELYON16 and TCGA-ESCA appear more distinct. Based on this observation, we suppose that the BRCA, NSCLC, and

RCC WSI samples share some characteristics that confuse the router. To address this phenomenon, the *Expert Assignment* process scales up the weights that are to be the highest (based on the assumption that the target task is known) by multiplying them by a scale factor γ and then applying the `softmax` function. This operation not only does not negatively impact CAMELYON16, BRCA, or ESCA—since if they are the highest, they remain the highest—but also encourages the weights of RCC and NSCLC to become the highest, thereby promoting their corresponding experts to make the most significant contributions.

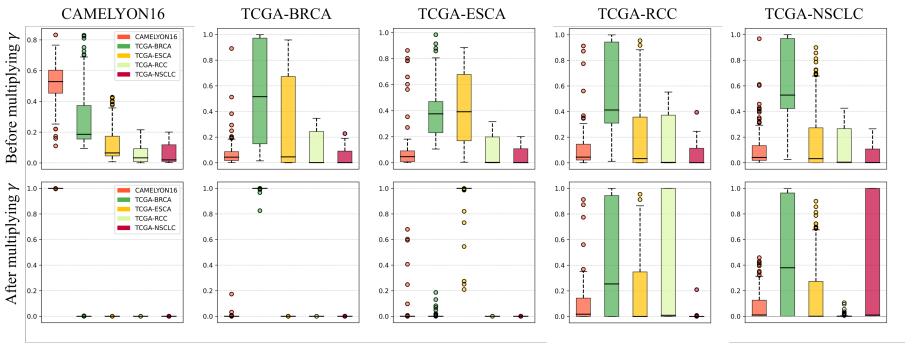

Fig. 1: Visualization of weights produced by the router $\mathcal{R}(\cdot)$ before and after multiplying scale hyperparameter γ. Each subplot corresponds to a specific task.

D Extensive Model Analyses

D.1 Hyperparameters

Figure 2 shows the overall F1 results for scaling γ and shifting β hyperparameters, each ranging from 1 to 5 with 0.5 intervals. Higher γ generally yields better performance, to which MECFormer is more sensitive. Increasing γ from 1.0 to 1.5 improves the F1 score by $0.037 \sim 0.064$ across all β values. Meanwhile, increasing β by 0.5 results in slight variations of $-0.008 \sim 0.015$ across all γ values. Performance stabilizes with γ from 2.0 to 4.5 and peaks at $\gamma = 5.0$. The best results, obtained with $\gamma = 5.0$ and $\beta = 1.0$.

D.2 Model Complexity

For training and inference, we conduct all the experiments on a PC with Ubuntu 20.04.6 LTS, an Intel(R) Xeon(R) Silver 4210R CPU @ 2.40GHz, and a single NVIDIA A6000 GPU with 48GB of VRAM. All the models and experiments are implemented using the PyTorch library, version 1.8.1. To investigate model complexity, we provide Figure 3. As the number of patches per slide increases,

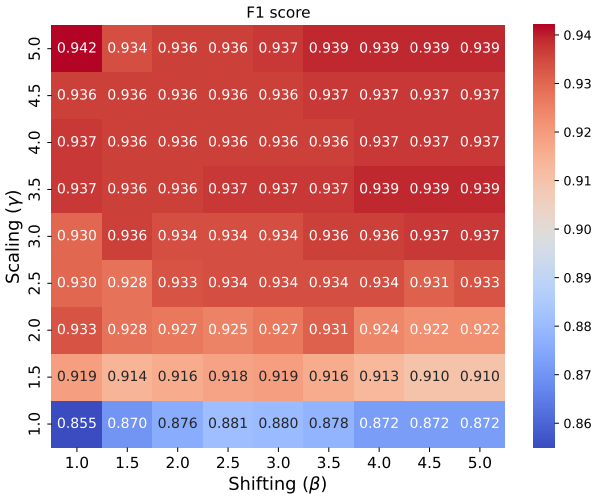

Fig. 2: Impact of scaling (γ) and shifting (β) hyperparameters on overall F1 score. Please zoom out for viewing ease.

MECFormer's FLOPs rise due to the introduction of the $\mathcal{P}_{\texttt{ECN}}$ and language decoder $\mathcal{D}$ (Fig. 3a). Despite higher FLOPs, MECFormer is the top performer in a *joint training* setting, which requires a single model to handle all five WSI classification tasks (Fig. 3b).

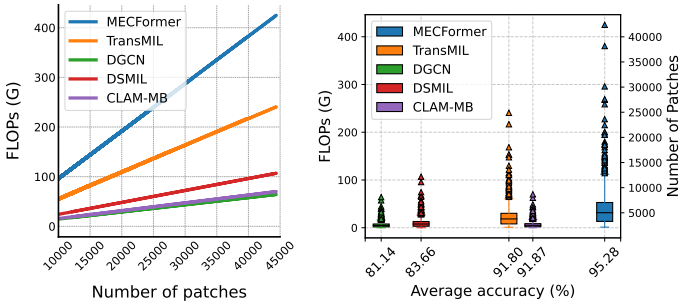

Fig. 3: Model complexity comparison between models: (a) FLOPs vs. number of patches trade-off (b) FLOPs vs. 5-task average accuracy (*joint training*) trade-off.

References

1. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. Advances in neural information processing systems **30** (2017) 2
2. Xiong, Y., Zeng, Z., Chakraborty, R., Tan, M., Fung, G., Li, Y., Singh, V.: Nyströmformer: A nyström-based algorithm for approximating self-attention. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 35, pp. 14138–14148 (2021) 1