Token Sparsification for Faster Medical Image Segmentation

Lei Zhou[1(✉)], Huidong Liu[1,3], Joseph Bae[2], Junjun He[4], Dimitris Samaras[1], and Prateek Prasanna[2]

[1] Department of Computer Science, Stony Brook University, Stony Brook, NY, USA
lezzhou@cs.stonybrook.edu
[2] Department of Biomedical Informatics, Stony Brook University, Stony Brook, NY, USA
[3] Amazon, Seattle, WA, USA
[4] Shanghai Artificial Intelligence Laboratory, Shanghai, China

Abstract. *Can we use sparse tokens for dense prediction, e.g., segmentation?* Although token sparsification has been applied to Vision Transformers (ViT) to accelerate classification, it is still unknown how to perform segmentation from sparse tokens. To this end, we reformulate segmentation as a *sparse encoding* → *token completion* → *dense decoding* (SCD) pipeline. We first empirically show that naïvely applying existing approaches from classification token pruning and masked image modeling (MIM) leads to failure and inefficient training caused by inappropriate sampling algorithms and the low quality of the restored dense features. In this paper, we propose *Soft-topK Token Pruning (STP)* and *Multi-layer Token Assembly (MTA)* to address these problems. In *sparse encoding*, *STP* predicts token importance scores with a lightweight subnetwork and samples the topK tokens. The intractable topK gradients are approximated through a continuous perturbed score distribution. In *token completion*, *MTA* restores a full token sequence by assembling both sparse output tokens and pruned multi-layer intermediate ones. The last *dense decoding* stage is compatible with existing segmentation decoders, e.g., UNETR. Experiments show SCD pipelines equipped with *STP* and *MTA* are much faster than baselines without token pruning in both training (up to 120% higher throughput) and inference (up to 60.6% higher throughput) while maintaining segmentation quality. Code is available here: https://github.com/cvlab-stonybrook/TokenSparse-for-MedSeg.

Keywords: Token Pruning · Multi-layer Token Assembly · Medical Image Segmentation

1 Introduction

Vision Transformers (ViT) [6] for dense prediction [20,29] have achieved impressive results in tasks including medical image segmentation [8]. In general, high-resolution features [26] preserving details are always desirable for precise seg-

A. Frangi et al. (Eds.): IPMI 2023, LNCS 13939, pp. 743–754, 2023.
https://doi.org/10.1007/978-3-031-34048-2_57

mentation. However, because of the quadratic computation complexity in self-attention [25], doubling the resolution per dimension in a 3D volume can lead to an 8× longer sequence and hence 64× more computation. This growing computing burden can quickly surpass limited computation budgets. Considering ViT's flexibility and great potential in masked image modeling [9,14], we explore acceleration algorithms based on the standard ViT. Recently, token sparsification [15,16,21] has been proposed to accelerate inference in ViT for classification by dropping less important tokens. However, *to the best of our knowledge, there are no ViT token sparsification approaches for segmentation.* This leads us to ask the question: *Can we use sparse tokens for dense prediction, e.g., segmentation?*

To answer the question, we reformulate segmentation as a *sparse encoding* → *token completion* → *dense decoding* (SCD) pipeline. Unlike a standard *dense encoding* → *dense decoding* (DD) pipeline, *sparse encoding* and *token completion* are required in SCD. *Sparse encoding* requires learning a sparse token representation for speed and *token completion* is needed to restore the full set of tokens for dense prediction. We first examine a naïve realization of *sparse encoding* and *token completion* by applying existing approaches. Specifically, we adapt sampling methods in classification, e.g., EViT [15] and DynamicViT [21], to *sparse encoding*, and masked image modeling (MIM) [2,9] to *token completion*. However, we observe significantly inferior results in this SCD pipeline (See Table 1). Next, we provide more insight into the problems of existing methods.

Problems in *Sparse Encoding*. There are two steps in this step, i.e., token score estimation and token sampling. We show that EViT's token score estimation is inappropriate for segmentation and DynamicViT's token sampling leads to training inefficiency: *i)* **EViT** [15] uses the attention weights between spatial tokens and the `[CLS]` token to estimate scores. While this is sound for classification since `[CLS]` is used for prediction, it is sub-optimal for segmentation because `[CLS]` is deprecated in the segmentation decoder. *ii)* **DynamicViT** [21] estimates token scores with a sub-network. DynamicViT frames token sampling as a series of independent binary decisions to keep or drop tokens. This does not guarantee a fixed number of sampled tokens for each training input. To fit in batch training, DynamicViT keeps all tokens in memory and masks self-attention entries, leading to training inefficiency.

Problems in *Token Completion*. Previous sparse token classification models [15,21] do not require *token completion*. Thus, we borrow the design from MIM. MIM reconstructs full tokens from a partial token sequence by padding it to full length with learnable mask tokens and then hallucinating the masked regions from their context. While MIM is useful for pre-training, it cannot accurately restore detailed information, resulting in inferior segmentation results.

We propose *Soft-topK Token Pruning (STP)* and *Multi-layer Token Assembly (MTA)* to implement *sparse encoding* and *token completion*. *i)* In *sparse encoding*, *STP* predicts token importance scores with a sub-network, avoiding the limitation of `[CLS]` in segmentation. STP then samples topK-scored tokens instead of making binary decisions per token separately, accelerating training by retaining only the sampled tokens in memory and computing. Motivated by

subset sampling [5,12,28], the intractable gradients of the topK operation are approximated through a perturbed continuous score distribution. *ii)* In *token completion*, the *MTA* restores a full token sequence by assembling both sparse output tokens and pruned intermediate tokens from multiple layers. Compared to MIM that fills the pruned positions with identical mask tokens, *MTA* produces more informative, position-specific representations. For *dense decoding*, the SCD pipeline is compatible with existing segmentation decoders, such as UNETR.

We evaluate our method on two relatively sparse 3D medical image segmentation datasets, the CT Abdomen Multi-organ Segmentation (BTCV [11], N = 30) dataset and the MRI Brain Tumor Segmentation (MSD BraTS [1], N = 484) dataset. On both tasks, STP+MTA+UNETR matches the UNETR baseline while providing significant computing savings with large token pruning ratios. On BraTS, STP+MTA+UNETR accelerates segmentation inference/-training throughput by 60.6%/120% and achieves the same segmentation accuracy. On BTCV, STP+MTA+UNETR increases inference/training throughput by 24.1%/97.36% while maintaining performance. In summary, our contributions are:

- To the best of our knowledge, we are the first to use token pruning/dropping for ViT-based medical image segmentation.
- Based on subset sampling, our proposed *Soft-topK Token Pruning (STP)* module can be flexibly incorporated into a standard ViT to prune tokens with greater efficiency while maintaining accuracy.
- We propose *Multi-layer Token Assembly (MTA)* to recover a full set of tokens, i.e., a dense representation, from a sparse set. *MTA* preserves high-detail information for accurate segmentation.
- We show that STP+MTA+UNETR maintains performance compared with UNETR with much less computation on two 3D medical image datasets.

2 Methodology

Generally, a segmentation model consists of an encoder and a decoder. Our goal is to accelerate the ViT segmentation encoder. To this end, we reformulate segmentation as a *sparse encoding* → *token completion* → *dense decoding* (SCD) pipeline. *Sparse encoding* learns a sparse token representation for acceleration; *token completion* restores the full tokens for dense prediction; *dense decoding* predicts the segmentation mask from dense features. We first recap Vision Transformers and then illustrate the three components in the SCD pipeline.

Preliminary: Vision Transformers. Vision Transformers treat an image/volume as a sequence of tokens. In the case of 3D medical images, a 3D volume $\mathbf{x} \in \mathbb{R}^{H\times W\times D\times C_{in}}$ is first reshaped to a sequence of flattened patches $\mathbf{x}_p \in \mathbb{R}^{N\times(P^3\times C_{in})}$ where $H\times W\times D$ is the spatial size, C_{in} is the input channel, $P\times P\times P$ is the patch size, and $N = HWD/P^3$ is the sequence length, i.e., the number of patches. All the patches are then projected linearly to a C-dimensional token space, with position embeddings added to the projected patches. These

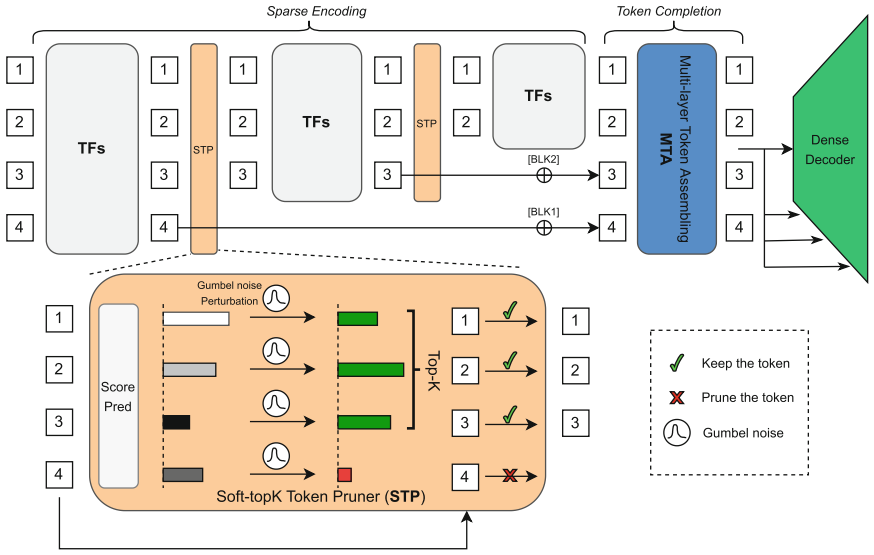

Fig. 1. Sparse Token Segmentation Pipeline. We reformulate segmentation as a *sparse encoding* → *token completion* → *dense decoding* pipeline. In *sparse encoding*, we design a *Soft-topK Token Pruning (STP)* module. In the forward pass, *STP* performs topK sampling on perturbed scores. In the backward pass, *STP* approximates the intractable gradient with a continuous Gumbel Softmax estimation. In *token completion*, we propose *Multi-layer Token Assembly (MTA)* to assemble both the output sparse tokens and the pruned intermediate ones to restore the complete tokens. In *dense decoding*, we avoid the intermediate sparse tokens by taking all inputs from the output of *MTA*. In this simplified figure, we visualize token pruning as dropping the last token. However, in practice pruned tokens are selected according to predicted scores.

patch tokens, together with a learnable prepended `[CLS]` token, are denoted as $\mathbf{z}_0 \in \mathbb{R}^{(1+N)\times C}$. $\mathbf{z}_0$ are further processed by L Transformer blocks sequentially. Each block consists of a multi-head self-attention (MSA) module and an MLP. We denote the tokens output from the ith Transformer block as $\mathbf{z}_i \in \mathbb{R}^{(1+N)\times C}$. For the segmentation task, before feeding the output $\mathbf{z}_L$ of the encoder to the decoder, we drop the `[CLS]` token and project the non-`[CLS]` token sequence $\mathbf{z}_L^{[1:N]} \in \mathbb{R}^{N\times C}$ back to the original 3D feature map $\mathbf{x}_L \in \mathbb{R}^{H/P\times W/P\times D/P\times C}$.

2.1 Sparse Encoding: Soft-topK Token Pruning (STP)

We build our sparse encoder on a ViT without modifying the self-attention module. Instead, we propose a learnable plug-and-play *Soft-topK Token Pruning (STP)* module. Compared to EViT & DynamicViT, our *STP*, as shown in the lower half of Fig. 1, estimates token scores more effectively and can be trained efficiently. *STP* can be inserted between two Transformer blocks $\texttt{TF}_i$ and $\texttt{TF}_{i+1}$. Receiving as input the token sequence $\mathbf{z}_i \in \mathbb{R}^{N_i\times C}$ from $\texttt{TF}_i$, *STP* prunes tokens

with a ratio r and passes the remaining tokens $\mathbf{z}'_i \in \mathbb{R}^{\lfloor (1-r)N_i \rceil \times C}$ to $\texttt{TF}_{i+1}$. In particular, *STP* consists of token-wise score estimation and token sampling. To be concise, we change the notation of number of tokens from N_i to n.

Token Score Estimation. To decide which tokens to keep or prune, we introduce a lightweight sub-network $s_\theta : \mathbb{R}^{n \times C} \rightarrow \mathbb{R}^n$ to predict the token importance scores $\mathbf{s}$, where θ are the network parameters. The architecture of s_θ is designed to aggregate both the local and global features, similarly to [21]. The global feature is simply obtained by average pooling over all the tokens.

$$\mathbf{s} = s_\theta(\mathbf{z}) = \texttt{Sigmoid}\Big(\texttt{MLP}_2\Big([\mathbf{z}, \texttt{AvgPool}\big(\texttt{MLP}_1(\mathbf{z})\big)]\Big)\Big) \tag{1}$$

Straight-Through Gumbel Soft TopK Sampling. Given a token pruning ratio r, *STP* needs to select $K = \lfloor (1-r)n \rceil$ tokens out of n to keep. After predicting the scores $\mathbf{s}$, we re-interpret each score value $\mathbf{s}_i$ as the probability of the i-th token ranking in the topK. We formulate this process as sampling a binary policy mask $\mathbf{M} \in \{0,1\}^n$ from the predicted probabilities where $\mathbf{M}$ is subject to $\texttt{sum}(\mathbf{M}) = K$. $\mathbf{M}_i = 1$ indicates keeping the i-th token while $\mathbf{M}_i = 0$ indicates pruning. However, such discrete sampling is non-differentiable. To overcome the problem, we relax the sampling of discrete topK masks to a continuous approximation, the Gumbel-Softmax distribution:

$$\underbrace{\mathbf{M}_i = \mathbb{1}_{\text{topK}}(\log(s_i) + g_i)}_{\text{forward}} \xleftarrow{\text{approx}} \underbrace{\tilde{\mathbf{M}}_i = \frac{\exp((\log(s_i) + g_i)/\tau)}{\sum_{j=1}^{n} \exp((\log(s_j) + g_j)/\tau)}}_{\text{backward}} \tag{2}$$

where $\mathbb{1}_{\text{topK}}$ is an indicator function of whether the input perturbed score is among the topK of all n perturbed scores, $\{g\}_n$ are i.i.d samples from the Gumbel$(0, 1)$ distribution[1]. While training, we forward *STP* to sample the topK tokens based on the discrete $\mathbf{M}$ but backward with the gradient approximated by the continuous $\tilde{\mathbf{M}}$. We call this Straight-through (ST) Gumbel Soft TopK Sampling. During inference, we perform normal topK selection based on predicted scores without Gumbel noise perturbation for deterministic inference.

2.2 Token Completion: Multi-layer Token Assembly (MTA)

The output of the STP-ViT encoder is sparse. Thus, before passing the output to the decoder, we need to first restore the complete tokens. A straightforward solution can be obtained from Masked Image Modeling (MIM) [2,9]. MIM reconstructs an image from random partial image patches. It first pads the sparse token set with learnable `[MASK]` tokens up to its full length. Then the padded tokens are forwarded through Transformer blocks to reconstruct the masked regions. However, MIM is mostly utilized for pre-training which focuses more on semantic hallucination rather than accurate detail restoration. Thus,

[1] Gumbel$(0, 1)$ samples are drawn by sampling $-\log(-\log u)$ where $u \sim \text{Uniform}(0, 1)$.

it is sub-optimal for segmentation tasks that require assigning labels to pixels accurately.

We propose *Multi-layer Token Assembly (MTA)* to restore dense features by assembling both the outputted sparse tokens and the pruned intermediate tokens from multiple layers. Suppose we insert three *STP*s, $\{STP_1, STP_2, STP_3\}$, after different Transformer blocks in a ViT. We denote the token sets pruned by the three *STP*s as $\{\bar{\mathbf{z}}_1, \bar{\mathbf{z}}_2, \bar{\mathbf{z}}_3\}$. We concatenate these pruned tokens with the final output $\mathbf{z}_L$ and rearrange them to their original spatial order. Then, we add three learnable block tokens $\{[\texttt{BLK}_1], [\texttt{BLK}_2], [\texttt{BLK}_3]\}$ to the corresponding pruned tokens to indicate which block each token is pruned from. Finally, we introduce sin-cos position embeddings $\mathbf{E}_{pos}$ to all the tokens and forward them through Transformer blocks. The completion process can be summarized as follows:

$$\mathbf{z}_{\text{compl}} = \texttt{TF}(\texttt{rearrange}(\left[\bar{\mathbf{z}}_1 + [\texttt{BLK}_1], \bar{\mathbf{z}}_2 + [\texttt{BLK}_2], \bar{\mathbf{z}}_3 + [\texttt{BLK}_3], \mathbf{z}_L\right]) + \mathbf{E}_{pos}) \quad (3)$$

2.3 Dense Decoding and Optimization

As our goal is to design an acceleration method that is agnostic to decoder designs, designing a new segmentation decoder is beyond the scope of this paper. Thus, we couple the SCD pipeline with existing segmentation decoders. However, certain segmentation decoders, e.g., UNETR, require inputs from multiple layer outputs from the encoder, which causes problems because intermediate features are still sparse. Motivated by recent research on the non-hierarchical feature pyramid [13], we use the output $\mathbf{z}_{\text{compl}}$ of the completion network to replace all the intermediate features required by the segmentation head, as shown in Fig. 1.

Unlike DynamicViT, we do not introduce additional loss functions for token pruning. We optimize all segmentation models by segmentation loss. We adopt a combination of cross entropy and Dice loss. Both loss weights are set to 1.

3 Experiments

3.1 Dataset Description

We evaluate on two benchmark 3D medical segmentation datasets with sparse targets. The tasks are CT multi-organ and MRI Brain tumor segmentation.

CT Multi-organ Segmentation (BTCV). The BTCV [11] (Multi Atlas Labeling Beyond The Cranial Vault) dataset consists of 30 subjects with abdominal CT scans where 13 organs were annotated under the supervision of board-certified radiologists. Each CT volume has 85–198 slices of 512×512 pixels, with a voxel spatial resolution of ($0.54 \times 0.98 \times$ [2.5–5.0] mm^3). For comparison convenience, we follow [3,4] to split the 30 cases into 18 for training and 12 for validation. Hyper-parameters are selected via 3-fold cross validation in the training set. We report the average DSC (Dice Similarity Coefficient) and 95% Hausdorff Distance (HD95) on 8 abdominal organs (aorta, gallbladder, spleen, left kidney, right kidney, liver, pancreas, spleen, stomach) to align with [4].

Table 1. Performance of existing approaches on BTCV. We first examine the performance of the naïve combination of existing approaches. For a large pruning ratio $r = 0.9$ on BTCV, MIM fails to perform segmentation effectively. Even with our proposed MTA instead of MIM, EViT and DynamicViT still perform worse than our STP. We report the `mean` and `std` on three random runs unless otherwise stated. Please see Sec. 3 for more analysis.

DSC(%) on BTCV (pruning ratio $r = 0.9$)		*sparse encoding*		
		DynamicViT [21]	EViT [15]	STP (ours)
token completion	MIM [2,9]	24.35 (single run)	18.64 (single run)	44.71 (single run)
	MTA (ours)	80.24 ± 0.34	78.62 ± 0.10	82.18 ± 0.12

MRI Brain Tumor Segmentation (BraTS). The Medical Segmentation Decathlon (MSD) [1] BraTS dataset has 484 multi-modal (FLAIR, T1w, T1-Gd and T2w) MRI scans. The ground-truth segmentation labels include peritumoral edema, GD-enhancing tumor and the necrotic/non-enhancing tumor core. The performance is measured on three recombined regions, i.e., tumor core, whole tumor and enhancing tumor. We randomly split the dataset into training (80%), validation (15%), and test (5%) sets. We report average DSC and HD95.

3.2 Implementation Details

Our method is implemented in PyTorch [19] and MONAI [18] on a single NVIDIA A100. Our encoder is based on a ViT-Base model. Three *STP* modules are inserted after the 3rd, 6th, and 9th Transformer blocks in ViT-B. We follow UNETR [8] on data processing. For BTCV, we clip the raw values between -958 and 326, and re-scale the range between -1 and 1. For BraTS, we perform an instance-wise normalization over the non-zero region per channel. For training, we set the batch size to 2 and the initial learning rate to 1.3e$-$4. We use AdamW as the optimizer and adopt layer-wise learning rate decay (ratio = 0.75) to improve training. For inference, we use a sliding window with an overlap of 50%.

3.3 Results

Naïve Combination of EViT/DynamicViT+MIM. We first test the straightforward approach of applying EViT/DynamicViT to *sparse encoding* and MIM to *token completion*. We use UNETR as the segmentation decoder. In Table 1, EViT/DynamicViT + MIM fails to perform dense prediction for a very high pruning ratio $r = 0.9$ on BTCV. This justifies our efforts in this paper to accelerate sparse token segmentation models while maintaining performance.

Our Approach: STP+MTA. We evaluate the efficiency of our *Soft-topK Token Pruning (STP)* and *Multi-layer Token Assembly (MTA)* on the BTCV

Table 2. STP+MTA+UNETR vs. UNETR performance comparison. Based on the same ViT scale and patch size, our proposed STP+MTA+UNETR can maintain performance while significantly reducing computation by a large margin. We report the `mean` and `std` of three random runs on BTCV. Please refer to Sect. 3.3 for more details on the experimental setting and analysis.

Method	MSD BraTS		Encoder Throughput(img/s)	Throughput (img/s)	MACs(G)
	DSC↑	HD95↓			
UNETR	75.44	8.89	7.10	4.85	824.38
STP+MTA+UNETR	**75.79**	**8.31**	20.04	7.79 (+60.6%)	428.28

Method	BTCV		Encoder Throughput(img/s)	Throughput (img/s)	MACs(G)
	DSC↑	HD95↓			
UNETR	80.78 ± 0.34	**15.90** ± 1.01	30.30	16.18	273.45
STP+MTA+UNETR	**82.18** ± 0.12	19.85 ± 1.12	57.31	20.08 (+24.1%)	146.63

and BraTS datasets based on UNETR. We measure the efficiency by profiling the throughput(image/s) and MAC number (Multiply-accumulate operations) for each model variant. The throughput is measured on a NVIDIA A100 GPU with batch size 1. MACs are computed by measuring the forward complexity of a single image. We present the results in Table 2. On BraTS, with an input size of $(128 \times 128 \times 128)$, our STP+MTA+UNETR ($r = 0.75$) maintains performance while significantly increasing inference throughput by 60.8%. On BTCV, with an input size of $(96 \times 96 \times 96)$, STP+MTA+UNETR ($r = 0.9$) can maintain performance while the corresponding inference throughput increases by 24.1%. Our method also increases training efficiency. The training throughput on BTCV increases from 2.65 imgs/s to 5.23 imgs/s by 97.36%. The training throughput on BraTS increases from 0.75 imgs/s to 1.65 imgs/s by 120%.

Sparse Encoding: STP vs. EViT/DynamicViT. EViT [15] and DynamicViT [21] were initially designed for classification. Thus, we need to adapt EViT/DynamicViT for comparison. To constrain the pruning ratio in DynamicViT, we add the ratio loss function $\mathcal{L}_{ratio}$ with a weight of $\lambda_{ratio} = 2$ following [21]. In EViT, we take the `[CLS]` attention weights from the Transformer block as the token scores and use topK for sampling. As shown in Table 4a, our STP-ViT performs the best. The inferiority of DynamicViT could be caused by *i*) mismatch between the training (variable number of pruned tokens) and testing phases (fixed number of pruned tokens) and *ii*) more hyper-parameters (e.g., λ_{ratio}). The performance drop in EViT indicates that the `[CLS]` attention scores are not suitable for representing the true token importance in segmentation.

Token Completion: MTA vs. MIM. We implement a baseline inspired by MIM [2,9]. As Table 4b shows, MIM-style completion fails (44.71%) with a high pruning ratio $r = 0.9$. Our results suggest that pruned token reuse in *MTA* plays an important role in a highly sparse token segmentation framework.

Token Pruning Ratio in STP. We ablate the pruning ratio in Table 3. *STP* is robust to a wide range of pruning ratios $[0.25, 0.9]$. Thus, our STP+MTA+UNETR can adopt a high pruning ratio to reduce computation by a large margin.

Table 3. Ablation on the Pruning Ratio r. STP shows robustness to a wide range of pruning ratios ($0.25 \rightarrow 0.9$) in terms of DSC. Different datasets have different optimal pruning ratios. Refer to Sect. 3.3 for more details. We report the `mean` and `std` of three random runs on BTCV unless otherwise stated.

Pruning Ratio r	BTCV		BraTS		Encoder Throughput	Throughput	MACs(G)
	DSC↑	HD95↓	DSC↑	HD95↓			
baseline	80.78 ± 0.34	15.90 ± 1.01	75.44	8.89	7.10	4.85	824.38
0.25	81.56 ± 0.16	19.65 ± 3.25	75.50	7.98	11.77	6.12	631.75
0.50	81.81 ± 0.59	$\mathbf{15.78} \pm 1.01$	75.02	**7.40**	17.34	7.35	497.97
0.75	81.95 ± 0.18	16.37 ± 5.41	**75.79**	8.31	20.04	7.79	428.28
0.9	$\mathbf{82.18} \pm 0.12$	19.85 ± 1.12	75.32	8.04	21.63	8.04	404.14

Table 4. Ablation studies on BTCV. In (a), we compare *STP* with DynamicViT and EViT. *STP* achieves better performance. In (b), we compare our proposed *MTA* with MIM where MIM performs much worse than MTA. In (c), we demonstrate that Gumbel perturbation is beneficial. In (d), we ablate different τ values. $\tau = 0.1$ and $\tau = 1$ perform similarly while $\tau = 0.01$ performs worse. We report the `mean` and `std` of three random runs unless otherwise stated.

Encoder	DSC
DynamicViT	80.24 ± 0.34
EViT	78.62 ± 0.10
STP-ViT (Ours)	$\mathbf{82.18} \pm 0.12$

(a) Comparison with DynamicViT&EViT

Token Completion	DSC
MIM	44.71 (single run)
MTA (ours)	$\mathbf{82.18} \pm 0.12$

(b) Token Completion Methods

Perturbation	DSC
No (ST TopK)	81.67 ± 0.21
Yes (ours)	$\mathbf{82.18} \pm 0.12$

(c) Gumbel Perturbation

τ	DSC
0.01	81.36 ± 0.15
0.1	82.06 ± 0.22
1 (ours)	$\mathbf{82.18} \pm 0.12$

(d) Temperature τ in *STP*

Although our method achieves higher DSC on BTCV than UNETR, the HD95 is worse. We speculate that HD95 is more sensitive to the boundary segmentation results and that token pruning may lead to sub-optimal boundary prediction.

Temperature τ in STP. We ablate temperature τ in Eq. 2 in Table 4d. According to [10], a small temperature leads to a large variance of gradients and vice versa. We tried three different τ values $\{0.01, 0.1, 1\}$. Experiments show $\tau = 0.1$ and $\tau = 1$ perform similarly while $\tau = 0.01$ performs worse.

Noise Perturbation in STP. In *Soft-topK Token Pruning (STP)*, we design a straight-through (ST) Gumbel soft topK algorithm for sampling. *STP* forward process can be split into three steps, i.e., score prediction, Gumbel perturbation, and topK sampling. In Table 4c, we ablate the Gumbel perturbation on BTCV by evaluating a straight-through (ST) topK variant. Note that we do not add Gumbel noise during inference, to ensure that the model performs deterministically for inference. For the ST topK variant, we also remove the Gumbel noise

Table 5. Comparison with other methods on BTCV.

Framework	DSC↑/HD95↓	Aorta	Gallbladder	Kidney(L)	Kidney(R)	Liver	Pancreas	Spleen	Stomach
V-Net [17]	68.81/–	75.34	51.87	77.10	80.75	87.84	40.05	80.56	56.98
DARR [7]	69.77/–	74.74	53.77	72.31	73.24	94.08	54.18	89.90	45.96
U-Net(R50) [22]	74.68/36.87	84.18	62.84	79.19	71.29	93.35	48.23	84.41	73.92
AttnUNet(R50) [23]	75.57/36.97	55.92	63.91	79.20	72.71	93.56	49.37	87.19	74.95
TransUNet [4]	77.48/31.69	87.23	63.13	81.87	77.02	94.08	55.86	85.08	75.62
UNETR (PatchSize = 16)	78.83/25.59	85.46	70.88	83.03	82.02	95.83	50.99	88.26	72.74
UNETR (PatchSize = 8)	80.78/**15.90**	88.59	70.97	83.38	83.76	95.52	59.76	88.53	74.30
STP+MTA+UNETR (PatchSize = 8)	**82.18**/19.85	89.23	73.60	85.66	83.65	95.59	62.17	88.84	77.37

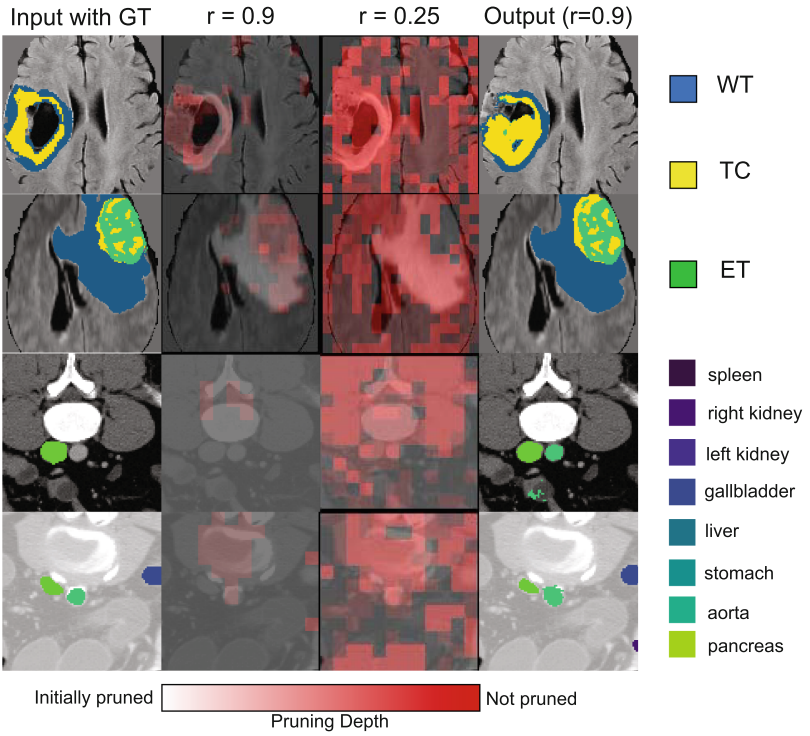

Fig. 2. Ground truth and model outputs on BraTS (first two rows) and BTCV (last two rows). We visualize the depth at which tokens are pruned under high (r = 0.9) and low (r = 0.25) pruning ratios (red shading in columns 2 and 3). Tokens that are immediately dropped are not shaded, whereas darker red shading indicates the pruning of tokens in later layers. (Color figure online)

perturbation from the training phase. With a pruning ratio $r = 0.9$, results show that the Gumbel perturbation is beneficial. It is worth noting that the ST topK variant without perturbation also achieves a competitive result.

Pruning Policy Visualization. We visualize the pruning policy for both brain tumors and abdominal organs in Fig. 2 under two extreme pruning ratios, the highest one at $r = 0.9$ and the lowest at $r = 0.25$. We use shades of red to denote the depth at which tokens are pruned. Patches (tokens in ViT) with no

red overlap are pruned by the very first *STP*, whereas patches with the deepest red color are kept in ViT until the last. In Fig. 2, with $r = 0.9$, most tokens are dropped at a very early stage. Some tokens around the brain tumor, especially at tumor boundaries, are never pruned. When the ratio decreases to $r = 0.25$, more patches are kept and still cluster around the target tumor region.

Class-Wise Comparison with Others on BTCV. We show class-wise results of UNETR, STP+MTA+UNETR, and other methods in Table 5.

STP+MTA+UNETR shows improvement over a series of methods on BTCV. Note that current SOTA methods [24,27,30] rely on either stronger priors (window attention) or SSL pre-training. However, our goal is accelerating standard ViT-based segmentation instead of purely pursuing increased performance.

4 Conclusion and Future Work

We introduced a ViT-based sparse token segmentation framework for medical images. First, we proposed a *Soft-topK Token Pruning* (STP) module to prune tokens in ViT. STP can speed up ViTs in both training and inference phases. To produce a full set of tokens for dense prediction, we proposed *Multi-layer Token Assembly (MTA)* that recovers a complete set of tokens by assembling both output and intermediate tokens from multiple layers. In our 3D medical image experiments STP+MTA+UNETR speeds up the UNETR baseline significantly while maintaining segmentation performance. Accelerating the decoder, which also plays a big role in the inference speed, is left for future work.

Acknowledgement. The reported research was partly supported by NIH award # 1R21CA258493-01A1, NSF awards IIS-2212046 and IIS-2123920, and Stony Brook OVPR seed grants. The content is solely the responsibility of the authors and does not necessarily represent the official views of the National Institutes of Health.

References

1. Antonelli, M., et al.: The medical segmentation decathlon. arXiv preprint arXiv:2106.05735 (2021)
2. Bao, H., Dong, L., Wei, F.: BEiT: BERT pre-training of image transformers. arXiv preprint arXiv:2106.08254 (2021)
3. Chen, J.N.: Transunet. https://github.com/Beckschen/TransUNet
4. Chen, J., et al.: TransUNet: transformers make strong encoders for medical image segmentation. arXiv preprint arXiv:2102.04306 (2021)
5. Cordonnier, J.B., Mahendran, A., Dosovitskiy, A., Weissenborn, D., Uszkoreit, J., Unterthiner, T.: Differentiable patch selection for image recognition. In: CVPR, pp. 2351–2360 (2021)
6. Dosovitskiy, A., et al.: An image is worth 16×16 words: transformers for image recognition at scale. arXiv preprint arXiv:2010.11929 (2020)
7. Fu, S., et al.: Domain adaptive relational reasoning for 3D multi-organ segmentation. In: Martel, A.L., et al. (eds.) MICCAI 2020. LNCS, vol. 12261, pp. 656–666. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-59710-8_64

8. Hatamizadeh, A., et al.: UNETR. In: WACV (2022)
9. He, K., Chen, X., Xie, S., Li, Y., Dollár, P., Girshick, R.: Masked autoencoders are scalable vision learners. arXiv preprint arXiv:2111.06377 (2021)
10. Jang, E., Gu, S., Poole, B.: Categorical reparameterization with gumbel-softmax. arXiv preprint arXiv:1611.01144 (2016)
11. Landman, B., Xu, Z., Igelsias, J., Styner, M., Langerak, T., Klein, A.: MICCAI multi-atlas labeling beyond the cranial vault-workshop and challenge. In: Proceedings of the MICCAI Multi-Atlas Labeling Beyond Cranial Vault-Workshop Challenge (2015)
12. Li, J., Cotterell, R., Sachan, M.: Differentiable subset pruning of transformer heads. Trans. Assoc. Comput. Linguist. **9**, 1442–1459 (2021)
13. Li, Y., Mao, H., Girshick, R., He, K.: Exploring plain vision transformer backbones for object detection. arXiv preprint arXiv:2203.16527 (2022)
14. Li, Y., Xie, S., Chen, X., Dollar, P., He, K., Girshick, R.: Benchmarking detection transfer learning with vision transformers. arXiv preprint arXiv:2111.11429 (2021)
15. Liang, Y., Chongjian, G., Tong, Z., Song, Y., Wang, J., Xie, P.: Evit: expediting vision transformers via token reorganizations. In: ICLR (2021)
16. Meng, L., et al.: AdaViT: adaptive ViTs for efficient image recognition. arXiv preprint arXiv:2111.15668 (2021)
17. Milletari, F., Navab, N., Ahmadi, S.A.: V-net: fully convolutional neural networks for volumetric medical image segmentation. In: 3DV, pp. 565–571. IEEE (2016)
18. MONAI Consortium: MONAI: Medical Open Network for AI (2020). https://doi.org/10.5281/zenodo.4323058, https://github.com/Project-MONAI/MONAI
19. Paszke, A., et al.: PyTorch: an imperative style, high-performance deep learning library. In: NeurIPS, vol. 32 (2019)
20. Ranftl, R., Bochkovskiy, A., Koltun, V.: Vision transformers for dense prediction. In: ICCV (2021)
21. Rao, Y., Zhao, W., Liu, B., Lu, J., Zhou, J., Hsieh, C.J.: DynamicViT: efficient vision transformers with dynamic token sparsification. In: NeurIPS, vol. 34 (2021)
22. Ronneberger, O., Fischer, P., Brox, T.: U-net: convolutional networks for biomedical image segmentation. In: Navab, N., Hornegger, J., Wells, W.M., Frangi, A.F. (eds.) MICCAI 2015. LNCS, vol. 9351, pp. 234–241. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-24574-4_28
23. Schlemper, J., et al.: Attention gated networks: learning to leverage salient regions in medical images. Med. Image Anal. **53**, 197–207 (2019)
24. Tang, Y., et al.: Self-supervised pre-training of swin transformers for 3D medical image analysis. In: CVPR (2022)
25. Vaswani, A., et al.: Attention is all you need. In: NeurIPS, vol. 30 (2017)
26. Wang, J., Sun, K., Cheng, T., Jiang, B., Deng, C., Zhao, Y., Liu, D., Mu, Y., Tan, M., Wang, X., et al.: Deep high-resolution representation learning for visual recognition. IEEE Trans. PAMI **43**(10), 3349–3364 (2020)
27. Wu, Y., et al.: D-former: a U-shaped dilated transformer for 3D medical image segmentation. arXiv preprint arXiv:2201.00462 (2022)
28. Xie, S.M., Ermon, S.: Reparameterizable subset sampling via continuous relaxations. arXiv preprint arXiv:1901.10517 (2019)
29. Zheng, S., et al.: Rethinking semantic segmentation from a sequence-to-sequence perspective with transformers. In: CVPR (2021)
30. Zhou, H.Y., Guo, J., Zhang, Y., Yu, L., Wang, L., Yu, Y.: nnFormer: interleaved transformer for volumetric segmentation. arXiv preprint arXiv:2109.03201 (2021)