SPARSE TOKENS FOR DENSE PREDICTION- THE MEDICAL IMAGE SEGMENTATION CASE

Anonymous authors
Paper under double-blind review

ABSTRACT

Can we use sparse tokens for dense prediction, e.g., segmentation? Although token sparsification has been applied to Vision Transformers (ViT) for acceleration on classification tasks, it is still unknown how to perform segmentation from sparse tokens. To this end, we reformulate segmentation as a *sparse encoding* $\rightarrow$ *token completion* $\rightarrow$ *dense decoding* (SCD) pipeline. We first show empirically that naïvely applying existing approaches from classification token pruning and masked image modeling (MIM) leads to failure and training inefficiency. This is 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 the above problems. Particularly, in the *sparse encoding* stage, *STP* predicts token-wise importance scores with a lightweight sub-network and samples topK-scored tokens. The intractable gradients of topK are approximated through a continuous perturbed score distribution. In the *token completion* stage, *MTA* restores a full token sequence by assembling both sparse output tokens and pruned intermediate tokens from multiple layers. Compared to MIM which fills the pruned positions with mask tokens, *MTA* produces more informative representations allowing more accurate segmentation. The last *dense decoding* stage is compatible with decoders of existing segmentation frameworks, e.g., UNETR. Experiments show SCD pipelines equipped with our *STP* and *MTA* are much faster than baselines without token sparsification in both training (up to 120% higher throughput) and inference (up to 60.6% higher throughput) while maintaining segmentation quality.

1 INTRODUCTION

Vision Transformers (ViT) (Dosovitskiy et al., 2020) for dense prediction (Zheng et al., 2021; Ranftl et al., 2021; Strudel et al., 2021) have achieved impressive results in tasks including medical image segmentation (Hatamizadeh et al., 2022). In general, high-resolution representations (Wang et al., 2020) preserving detailed structures are always desirable for precise segmentation. However, because of the quadratic computation complexity in self-attention (Vaswani et al., 2017), doubling the feature resolution per dimension in a 3D medical volume can lead to an 8 times longer sequence and hence 64 times more computation. This growing computation burden can quickly surpass limited computation budgets. Considering the ViT's flexibility and great potential in mask image modeling (He et al., 2021; Li et al., 2021b), we explore acceleration algorithms based on the standard ViT. Recently, token sparsification (Rao et al., 2021; Liang et al., 2021; Meng et al., 2021; Fayyaz et al., 2021; Yin et al., 2021) has been proposed to accelerate inference in ViT for classification tasks 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* $\rightarrow$ *token completion* $\rightarrow$ *dense decoding* (SCD) pipeline. In constrast to a common *dense encoding* $\rightarrow$ *dense decoding* (DD) segmentation pipeline, two new components, *sparse encoding* and *token completion*, are required in SCD. Specifically, *sparse encoding* needs to learn a sparse token representation for fast inference and *token completion* needs to restore the full set of tokens from the sparse representation for dense prediction. We first examine a naïve realization of *sparse encoding* and *token completion* by applying existing token sampling and completion approaches. Particularly, we adapt sampling methods

used in classification, e.g., EViT (Liang et al., 2021) and DynamicViT (Rao et al., 2021), to *sparse encoding*, and masked image modeling (MIM) (He et al., 2021; Bao et al., 2021; Fang et al., 2022) to *token completion*. However, compared with the DD baseline, we observe significantly inferior segmentation performance in an SCD pipeline (See Table 1). Next, we provide more insight into the problems of existing methods.

Problems in *Sparse Encoding*. There are two necessary steps in this component, 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:

- EViT (Liang et al., 2021) uses the attention weights between spatial tokens and the `[CLS]` token to estimate token importance scores. While this is sound for classification tasks since the `[CLS]` token is used to predict the class label, it is sub-optimal for segmentation because the `[CLS]` token is deprecated in the segmentation decoder.
- DynamicViT (Rao et al., 2021) estimates token importance scores with a lightweight sub-network, thus avoiding the dependence on the `[CLS]` token. In order to use the off-the-shelf Gumbel Softmax trick for differentiability, DynamicViT considers token sampling as a series of independent binary decisions on keeping versus dropping. However, this does not guarantee a fixed number of sampled tokens for each input during training. To fit in batch training, DynamicViT keeps all the tokens in memory and masks the entries in the self-attention equivalently, leading to training inefficiency.

Problems in *Token Completion*. Previous sparse token classification models (Rao et al., 2021; Liang et al., 2021) do not require *token completion*. Thus, we borrow the design from a similar scenario, Masked Image Modeling (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 the partial context. While MIM-style token hallucination is useful for pre-training, it does not accurately restore fine-detailed information, resulting in inferior segmentation results.

In this paper, we propose two modules: *Soft-topK Token Pruning (STP)* and *Multi-layer Token Assembly (MTA)* to fulfill *sparse encoding* and *token completion* respectively. *i)* In *sparse encoding*, *STP* predicts token-wise importance scores with a lightweight sub-network, thus avoiding the limitation of `[CLS]` token attention weights in segmentation. Then, STP always samples topK-scored tokens instead of making binary decisions per token separately, accelerating training because only the sampled tokens are involved in memory and computing. Motivated by subset sampling (Xie & Ermon, 2019; Li et al., 2021a; Kool et al., 2020; Cordonnier et al., 2021), intractable gradients of the topK operation are approximated through a perturbed (e.g., by Gumbel noise) 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 allowing more accurate segmentation. For *dense decoding*, the SCD pipeline is compatible with existing segmentation framework decoders, such as UNETR.

We expect that token pruning is most useful when segmentation targets are sparse, such as in 3D medical images (See Tables 6& 7 in the Appendix). We evaluate our proposed framework on two relatively sparse 3D medical image segmentation datasets, the CT Abdomen Multi-organ Segmentation dataset (BTCV (Landman et al., 2015), N=30) and the MRI Brain Tumor Segmentation dataset (MSD BraTS (Antonelli et al., 2021), N=484). On both tasks, STP+MTA+UNETR matches the UNETR baseline while enjoying significant computational savings due to a large token pruning ratio. In particular, on BraTS, STP+MTA+UNETR accelerates total segmentation throughput by 60.6% and still achieves the same segmentation accuracy. On BTCV, STP+MTA+UNETR increases throughput by 24.1% while maintaining performance at the same time.

In summary, our contributions are:

- To the best of our knowledge, we are the first to use token pruning/dropping for ViT-based 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 the UNETR baseline while significantly reducing computation on two 3D medical image datasets.

2 Related Work

Adaptive Networks. To pursue improvements in computation efficiency, researchers have proposed many adaptive computation frameworks on both CNNs (Figurnov et al., 2017) and ViTs (Rao et al., 2021; Liang et al., 2021; Meng et al., 2021; Fayyaz et al., 2021; Yin et al., 2021). Figurnov et al. (2017) extends the idea of Adaptive Computation Time (Graves, 2016) to the spatial dimension as Spatial Adaptive Computation Time (SACT). Due to the grid constraints on convolution, SACT is implemented with perforated convolutions, which are slower. The ViT shows increased flexibility in handling adaptive computations. A-ViT (Yin et al., 2021) follows the ACT idea and learns to halt the forwarding of some tokens in ViT when the accumulated score reaches a threshold. DynamicViT (Rao et al., 2021) uses Gumbel Softmax to sample tokens. However, DynamicViT cannot guarantee sampling a fixed number of tokens and thus sacrifices training efficiency. EViT (Liang et al., 2021) samples tokens according to the `[CLS]` token attention weights. Notably, each of these ViT-based methods focuses on classification tasks.

Gumbel Softmax Trick. The Gumbel Softmax trick (Jang et al., 2016) relaxes the Gumbel-Max trick for categorical sampling. The Gumbel-Max trick (Gumbel, 1954) draws samples z from a categorical distribution with class probabilities $s = [s_1, s_2, \cdots, s_n]$. i.e., $z = \text{one\_hot}(\text{argmax}_k[g_k + \log s_k])$ where $\{g\}_n$ are i.i.d samples drawn from the $\text{Gumbel}(0, 1)$ distribution. The Gumbel Softmax trick uses the softmax function as a continuous and differentiable approximation to argmax and thus generates Gumbel Softmax samples y where $y_k = \exp((\log s_k + g_k)/\tau)/\sum_{j=1}^{n} \exp((\log s_j + g_j)/\tau)$. Recently, researchers have extended the Gumbel Softmax trick to perform subset (topK) sampling (Xie & Ermon, 2019; Li et al., 2021a; Kool et al., 2020), a.k.a., Gumbel Soft topK sampling in the literature. When used in a Straight-Through manner, Gumbel Soft topK sampling approximates the backward gradients of discrete topK sampling instead of the one-hot (top-1) sampling in the original Gumbel Softmax trick. Please refer to Sec. 3.2 for more illustrations of the Gumbel Soft topK sampling.

3 Methodology

Generally speaking, a segmentation model consists of an encoder and a decoder. The goal of the proposed method is to alleviate the compute burden of the ViT encoder in segmentation models. To this end, we reformulate segmentation as a *sparse encoding* $\rightarrow$ *token completion* $\rightarrow$ *dense decoding* (SCD) pipeline, as shown in Fig. 1. Specifically, *sparse encoding* needs to learn a sparse token representation for fast inference; *token completion* needs to restore the full tokens from the sparse representation for dense prediction; *dense decoding* needs to predict the segmentation mask from the dense representation. As we build the whole model on a standard ViT, we first recap Vision Transformers and then illustrate the three components in the SCD pipeline.

3.1 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 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

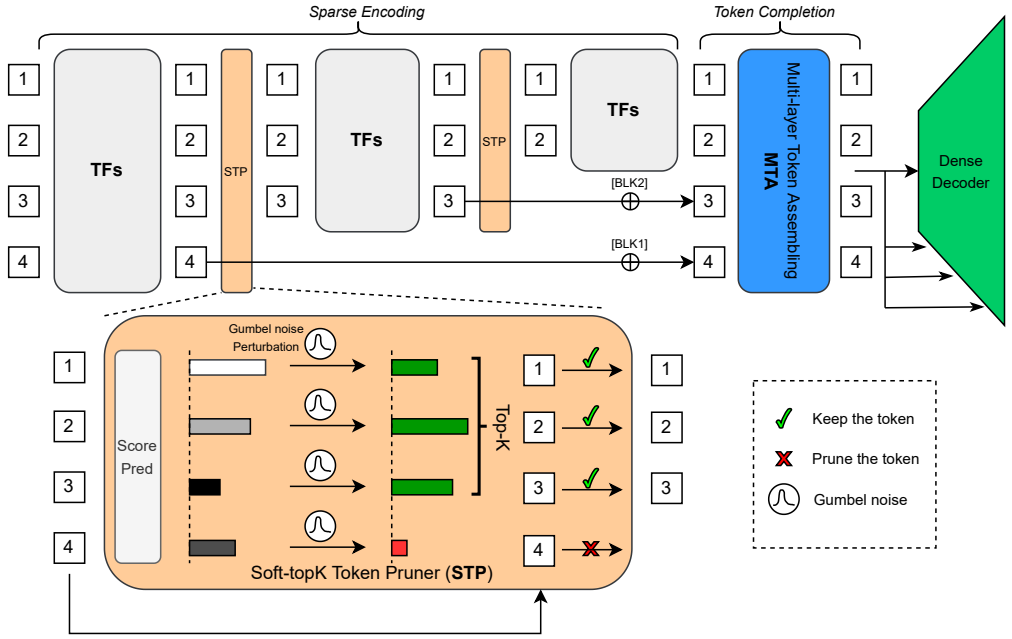

Figure 1: **Sparse Token Segmentation Pipeline**. We reformulate segmentation as a *sparse encoding* $\rightarrow$ *token completion* $\rightarrow$ *dense decoding* pipeline. In *sparse encoding*, we design a *Soft-topK Token Pruning (STP)* module that relaxes the discrete topK sampling by perturbing the token distribution. In the forward pass, *STP* performs topK sampling on the Gumbel-perturbed scores. In the backward pass, *STP* approximates the intractable sampling gradient with a continuous perturbed 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 full set of tokens. To identify which block the pruned tokens are from, we augment them with block-wise learnable `[BLK]` tokens. In *dense decoding*, we avoid the intermediate sparse tokens but take all the inputs from the output of the completion network.

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 $\mathbf{x}_L \in \mathbb{R}^{H/P \times W/P \times D/P \times C}$.

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

We build our sparse encoder on a standard ViT without modifying the self-attention module. Instead, we propose a learnable plug-and-play *Soft-topK Token Pruning (STP)* module, to prune tokens such that the subsequent Transformer blocks have fewer tokens as input and are thus sped up. Compared to EViT and DynamicViT, our proposed *STP*, as shown in the lower half of Fig. 1, estimates token scores more effectively and can be trained efficiently. *STP* can be inserted in between any consecutive Transformer blocks $\texttt{TF}_i$ and $\texttt{TF}_{i+1}$. Taking in 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 two steps: token-wise score estimation and token sampling, illustrated next. 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 importance scores $\mathbf{s}$ of all the tokens, where θ are the network parameters. In this work, the architecture of s_θ is designed to aggregate both the local and global features of tokens, similarly to DynamicViT. 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 importance 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 $\texttt{sum}(\mathbf{M}) = K$. Particularly, $\mathbf{M}_i = 1$ 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 (Please refer to Paragraph Gumbel Softmax 2 in Related Work for more details). This approximation can be summarized as,

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

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]. In the training phase, we forward *STP* to sample the topK tokens based on the discrete $\mathbf{M}$ but backward with the gradient approximated from the continuous $\tilde{\mathbf{M}}$. We call this Straight-through (ST) Gumbel Soft TopK Sampling. In the inference phase, we perform normal topK selection based on predicted scores without Gumbel noise perturbation for deterministic inference.

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

Segmentation tasks require dense prediction. However, the output of the STP-ViT encoder is sparse, as less informative tokens are pruned. Therefore, before passing the sparse tokens to the decoder, we need to first restore the full set of tokens. A straightforward solution can be obtained from recent advances in Masked Image Modeling (MIM) (Bao et al., 2021; He et al., 2021; Fang et al., 2022). In general, MIM reconstructs an image from random partial image patches. It first pads the sparse token set with identical learnable `[MASK]` tokens up to its full length. Then the padded tokens are forwarded through Transformer blocks to infer the missing information in masked regions from the non-masked ones. However, MIM-style token completion is mostly utilized for pre-training which focuses more on semantic hallucination rather than accurate restoration of true details. Thus, it is sub-optimal for segmentation tasks that require assigning labels to each pixel accurately.

We propose *Multi-layer Token Assembly (MTA)* to restore the dense features by assembling both the output sparse tokens and the pruned intermediate tokens from multiple layers. Suppose we insert three *STP* modules, $\{STP_1, STP_2, STP_3\}$, to prune tokens after three 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:

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

3.4 Dense Decoding

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 representations in the intermediate layers are still sparse. Motivated by recent research on the non-hierarchical feature pyramid (Li et al., 2022), 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. In our experiments, this approach has performed well.

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

3.5 OPTIMIZATION

We define the loss function for our method. Unlike DynamicViT, we do not introduce additional loss functions for token pruning. All segmentation models are optimized purely by segmentation loss. We adopt a combination of cross entropy and Dice loss. Both loss weights are set to 1.

4 EXPERIMENTS

4.1 DATASET DESCRIPTION

We evaluate our method on two benchmark 3D medical segmentation datasets which have sparse targets (See Tables 6 & 7 in Appendix). The tasks are CT multi-organ segmentation and MRI Brain tumor segmentation.

CT Multi-organ Segmentation (BTCV). The BTCV (Landman et al., 2015) (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 $85 \sim 198$ slices of 512×512 pixels, with a voxel spatial resolution of $(0.54 \times 0.98 \times [2.5 \sim 5.0]$ $mm^3)$. For comparison convenience, we follow (Chen et al., 2021; Chen) 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 Chen et al. (2021).

MRI Brain Tumor Segmentation (BraTS). Brain tumor segmentation is one of the 10 tasks in Medical Segmentation Decathlon (MSD) Challenge (Antonelli et al., 2021). The entire dataset has 484 multi-modal (FLAIR, T1w, T1-Gd and T2w) MRI brain 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. Reported metrics include average DSC and HD95.

4.2 IMPLEMENTATION DETAILS

Our method is implemented in PyTorch (Paszke et al., 2019) and MONAI (MONAI Consortium, 2020) (Apache 2.0 license). 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 (Hatamizadeh et al., 2022) on data processing. Please refer to Sec. G for data processing in the Appendix. 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 overlap of 50%. Experiments were run on a single NVIDIA A100.

4.3 RESULTS

Naïve Combination of EViT/DynamicViT + MIM. We first test the straighforward approach of applying EViT/DynamicViT to *sparse encoding* and MIM to *token completion*. We use UNETR as the segmentation decoder. As shown in Table 1, combining EViT/DynamicViT and 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. First, we evaluate the efficiency of our proposed *Soft-topK Token Pruning (STP)* and *Multi-layer Token Assembly (MTA)* on the BTCV 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 single 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. For the brain tumor segmentation task, with an input volume size of $(128 \times 128 \times 128)$, our STP+MTA+UNETR ($r = 0.75$) maintains the performance while significantly increasing the inference throughput by 60.8%. For the multi-organ segmentation task, with an input volume size of $(96 \times 96 \times 96)$, STP+MTA+UNETR ($r = 0.9$) can maintain the performance while the corresponding inference throughput increases by 24.1%. Our

DSC(%) on BTCV (pruning ratio $r = 0.9$)		*sparse encoding*		
		DynamicViT	EViT	STP (ours)
token completion	MIM	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

Table 1: **Performance of existing approach combination on BTCV**. We first examine the performance of the naïve combination of existing approaches, EViT/DynamicViT for token sampling, and mask image modeling (MIM) for token completion. For a large pruning ratio $r = 0.9$ on BTCV, MIM fails to perform segmentation effectively. Even with our later 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. 4 for more 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

Table 2: **STP+MTA+UNETR vs. UNETR performance comparison**. Based on the same ViT scale and patch size ($8 \times 8 \times 8$), our proposed STP+MTA+UNETR can maintain the performance while significantly reducing the computation by a large margin. We report the `mean` and `std` of three random runs on BTCV. Please refer to Sec 4.3 for more details on experimental setting and analysis.

method also increases training efficiency. Specifically, the training throughput on BTCV is increased from 2.65 imgs/s to 5.23 imgs/s by 97.36%. The training throughput on BraTS is increased from 0.75 imgs/s to 1.65 imgs/s by 120%.

Sparse Encoding: STP vs. EViT/DynamicViT. EViT (Liang et al., 2021) and DynamicViT (Rao et al., 2021) were initially designed for classification. Therefore, there are no official implementations for segmentation tasks. Thus we must adapt EViT/DynamicViT for segmentation. Specifically, to constrain the pruning ratio in DynamicViT, we add the ratio loss function $\mathcal{L}_{ratio}$ with a weight of $\lambda_{ratio} = 2$ following Rao et al. (2021). In EViT, we take the `[CLS]` attention weights from the Transformer block as the token scores and use topK for sampling. We show the comparison results in Table 4a. Our STP-ViT achieves the best performance. For DynamicViT, the inferiority 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 weight of the pruning ratio constraint loss) to be tuned. For EViT, the performance drop may indicate that the `[CLS]` attention score map is not suitable for representing the true token importance in the segmentation framework. Please refer to Table E in Appendix for more implementation details.

Token Completion: MTA vs. MIM. Token completion is a unique and important component for sparse token segmentation. We implement a baseline strategy inspired by masked image modeling (MIM) (Bao et al., 2021; He et al., 2021) for completion. 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 of *STP* in Table 3. Generally, *STP* is robust to a wide range of pruning ratios from 0.25 to 0.9. Therefore, our STP+MTA+UNETR can adopt a high pruning ratio (e.g., 90% for BTCV) to cut down the computation costs by a large margin. Although our method achieves better performance on BTCV than UNETR over DSC, the HD95 is worse. We speculate that HD95 is more sensitive to the boundary segmentation results while token pruning may lead to sub-optimal boundary prediction.

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 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 Dice Similarity Coefficient. Different datasets have different preference to pruning ratios. Refer to Sec 4.3 for more details. We report `mean` and `std` of three random runs on BTCV 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*

Table 4: **Ablation studies on BTCV**. We explore different designs within the *STP* and the completion network, evaluated in terms of DSC. In (a), we compare *STP* with other token sampling methods: 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 the 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 `mean` and `std` of three random runs unless otherwise stated.

Temperature τ in STP. We ablate temperature τ in Eq. 2 in Table 4d. According to (Jang et al., 2016), 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 in any case, such that the model performs deterministically for inference. For the ST topK variant, we further remove the Gumbel noise perturbation from the training phase too. 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.

Framework	DSC↑/HD95↓	Aorta	Gallbladder	Kidney(L)	Kidney(R)	Liver	Pancreas	Spleen	Stomach
V-Net (Milletari et al., 2016)	68.81/-	75.34	51.87	77.10	80.75	87.84	40.05	80.56	56.98
DARR (Fu et al., 2020)	69.77/-	74.74	53.77	72.31	73.24	94.08	54.18	89.90	45.96
U-Net(R50) (Ronneberger et al., 2015)	74.68/36.87	84.18	62.84	79.19	71.29	93.35	48.23	84.41	73.92
AttnUNet(R50) (Schlemper et al., 2019)	75.57/36.97	55.92	63.91	79.20	72.71	93.56	49.37	87.19	74.95
TransUNet (Chen et al., 2021)	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

Table 5: Comparison with other methods on BTCV.

Pruning Policy Visualization. We visualize the pruning policy for both brain tumor (Fig. 2) and abdominal organs (Fig. 3 in the Appendix) under two extreme pruning ratios, the highest one at $r = 0.9$ and lowest at $r = 0.25$. We use shades of red to denote the depth at which tokens are

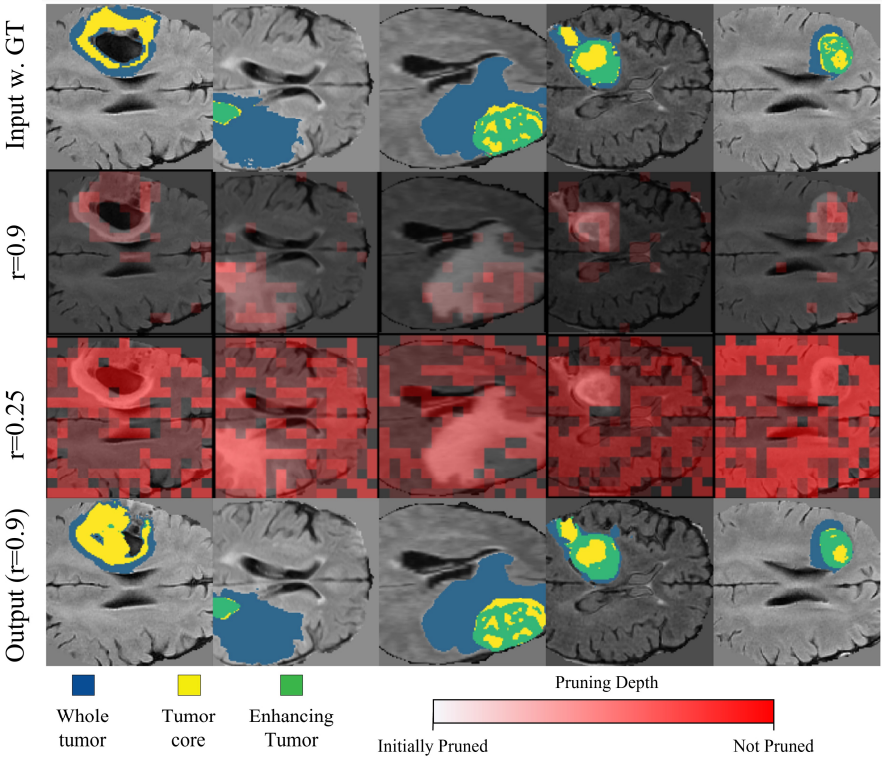

Figure 2: We show ground truth and model outputs for the brain tumor segmentation task. We also visualize the depth at which tokens are pruned under high (r=0.9) and low (r=0.25) pruning ratios (red shading in rows 2 and 3). Tokens that are immediately dropped are not shaded whereas darker red shading indicates pruning of tokens in later layers.

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 kept until the last. When the ratio decreases to $r = 0.25$, more patches are kept and still cluster around the target tumor region. In Fig. 3, we observe a similar phenomenon on the abdominal organ segmentation task.

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 (Zhou et al., 2021; Wu et al., 2022; Tang et al., 2022) rely on either stronger prior (window attention) or SSL pre-training. However, our goal is accelerating standard ViT-based segmentation instead of purely pursuing the performance.

5 Conclusion and Future Work

This paper 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. Our proposed STP can speed up ViTs in both the training and inference phases. In order 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 tokens and intermediate ones from multiple layers. Experiments on 3D medical image datasets show that 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, will be part of our future work.

REFERENCES

Michela Antonelli, Annika Reinke, Spyridon Bakas, Keyvan Farahani, Bennett A Landman, Geert Litjens, Bjoern Menze, Olaf Ronneberger, Ronald M Summers, Bram van Ginneken, et al. The medical segmentation decathlon. *arXiv preprint arXiv:2106.05735*, 2021.

Hangbo Bao, Li Dong, and Furu Wei. Beit: Bert pre-training of image transformers. *arXiv preprint arXiv:2106.08254*, 2021.

Jie-Neng Chen. Transunet. URL `https://github.com/Beckschen/TransUNet`.

Jieneng Chen, Yongyi Lu, Qihang Yu, Xiangde Luo, Ehsan Adeli, Yan Wang, Le Lu, Alan L Yuille, and Yuyin Zhou. Transunet: Transformers make strong encoders for medical image segmentation. *arXiv preprint arXiv:2102.04306*, 2021.

Jean-Baptiste Cordonnier, Aravindh Mahendran, Alexey Dosovitskiy, Dirk Weissenborn, Jakob Uszkoreit, and Thomas Unterthiner. Differentiable patch selection for image recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2351–2360, 2021.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

Yuxin Fang, Shusheng Yang, Shijie Wang, Yixiao Ge, Ying Shan, and Xinggang Wang. Unleashing vanilla vision transformer with masked image modeling for object detection. *arXiv preprint arXiv:2204.02964*, 2022.

Mohsen Fayyaz, Soroush Abbasi Kouhpayegani, Farnoush Rezaei Jafari, Eric Sommerlade, Hamid Reza Vaezi Joze, Hamed Pirsiavash, and Juergen Gall. Ats: Adaptive token sampling for efficient vision transformers. *arXiv preprint arXiv:2111.15667*, 2021.

Michael Figurnov, Maxwell D Collins, Yukun Zhu, Li Zhang, Jonathan Huang, Dmitry Vetrov, and Ruslan Salakhutdinov. Spatially adaptive computation time for residual networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1039–1048, 2017.

Shuhao Fu, Yongyi Lu, Yan Wang, Yuyin Zhou, Wei Shen, Elliot Fishman, and Alan Yuille. Domain adaptive relational reasoning for 3d multi-organ segmentation. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pp. 656–666. Springer, 2020.

Alex Graves. Adaptive computation time for recurrent neural networks. *arXiv preprint arXiv:1603.08983*, 2016.

Emil Julius Gumbel. *Statistical theory of extreme values and some practical applications: a series of lectures*, volume 33. US Government Printing Office, 1954.

Ali Hatamizadeh, Yucheng Tang, Vishwesh Nath, Dong Yang, Andriy Myronenko, Bennett Landman, Holger R Roth, and Daguang Xu. Unetr: Transformers for 3d medical image segmentation. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 574–584, 2022.

Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. *arXiv preprint arXiv:2111.06377*, 2021.

Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. *arXiv preprint arXiv:1611.01144*, 2016.

Wouter Kool, Herke van Hoof, Max Welling, et al. Ancestral gumbel-top-k sampling for sampling without replacement. *J. Mach. Learn. Res.*, 21:47–1, 2020.

Bennett Landman, Zhoubing Xu, J Igelsias, Martin Styner, T Langerak, and Arno Klein. Miccai multi-atlas labeling beyond the cranial vault–workshop and challenge. In *Proc. MICCAI Multi-Atlas Labeling Beyond Cranial Vault—Workshop Challenge*, volume 5, pp. 12, 2015.

Jiaoda Li, Ryan Cotterell, and Mrinmaya Sachan. Differentiable subset pruning of transformer heads. *Transactions of the Association for Computational Linguistics*, 9:1442–1459, 2021a.

Yanghao Li, Saining Xie, Xinlei Chen, Piotr Dollar, Kaiming He, and Ross Girshick. Benchmarking detection transfer learning with vision transformers. *arXiv preprint arXiv:2111.11429*, 2021b.

Yanghao Li, Hanzi Mao, Ross Girshick, and Kaiming He. Exploring plain vision transformer backbones for object detection. *arXiv preprint arXiv:2203.16527*, 2022.

Youwei Liang, GE Chongjian, Zhan Tong, Yibing Song, Jue Wang, and Pengtao Xie. Evit: Expediting vision transformers via token reorganizations. In *International Conference on Learning Representations*, 2021.

Lingchen Meng, Hengduo Li, Bor-Chun Chen, Shiyi Lan, Zuxuan Wu, Yu-Gang Jiang, and Ser-Nam Lim. Adavit: Adaptive vision transformers for efficient image recognition. *arXiv preprint arXiv:2111.15668*, 2021.

Fausto Milletari, Nassir Navab, and Seyed-Ahmad Ahmadi. V-net: Fully convolutional neural networks for volumetric medical image segmentation. In *2016 fourth international conference on 3D vision (3DV)*, pp. 565–571. IEEE, 2016.

MONAI Consortium. MONAI: Medical Open Network for AI, 3 2020. URL `https://github.com/Project-MONAI/MONAI`.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

René Ranftl, Alexey Bochkovskiy, and Vladlen Koltun. Vision transformers for dense prediction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 12179–12188, 2021.

Yongming Rao, Wenliang Zhao, Benlin Liu, Jiwen Lu, Jie Zhou, and Cho-Jui Hsieh. Dynamicvit: Efficient vision transformers with dynamic token sparsification. *Advances in neural information processing systems*, 34, 2021.

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pp. 234–241. Springer, 2015.

Jo Schlemper, Ozan Oktay, Michiel Schaap, Mattias Heinrich, Bernhard Kainz, Ben Glocker, and Daniel Rueckert. Attention gated networks: Learning to leverage salient regions in medical images. *Medical image analysis*, 53:197–207, 2019.

Robin Strudel, Ricardo Garcia, Ivan Laptev, and Cordelia Schmid. Segmenter: Transformer for semantic segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 7262–7272, 2021.

Yucheng Tang, Dong Yang, Wenqi Li, Holger R Roth, Bennett Landman, Daguang Xu, Vishwesh Nath, and Ali Hatamizadeh. Self-supervised pre-training of swin transformers for 3d medical image analysis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 20730–20740, 2022.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Jingdong Wang, Ke Sun, Tianheng Cheng, Borui Jiang, Chaorui Deng, Yang Zhao, Dong Liu, Yadong Mu, Mingkui Tan, Xinggang Wang, et al. Deep high-resolution representation learning for visual recognition. *IEEE transactions on pattern analysis and machine intelligence*, 43 (10):3349–3364, 2020.

Yixuan Wu, Kuanlun Liao, Jintai Chen, Danny Z Chen, Jinhong Wang, Honghao Gao, and Jian Wu. D-former: A u-shaped dilated transformer for 3d medical image segmentation. *arXiv preprint arXiv:2201.00462*, 2022.

Sang Michael Xie and Stefano Ermon. Reparameterizable subset sampling via continuous relaxations. *arXiv preprint arXiv:1901.10517*, 2019.

Hongxu Yin, Arash Vahdat, Jose Alvarez, Arun Mallya, Jan Kautz, and Pavlo Molchanov. Adavit: Adaptive tokens for efficient vision transformer. *arXiv preprint arXiv:2112.07658*, 2021.

Sixiao Zheng, Jiachen Lu, Hengshuang Zhao, Xiatian Zhu, Zekun Luo, Yabiao Wang, Yanwei Fu, Jianfeng Feng, Tao Xiang, Philip HS Torr, et al. Rethinking semantic segmentation from a sequence-to-sequence perspective with transformers. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 6881–6890, 2021.

Hong-Yu Zhou, Jiansen Guo, Yinghao Zhang, Lequan Yu, Liansheng Wang, and Yizhou Yu. nnformer: Interleaved transformer for volumetric segmentation. *arXiv preprint arXiv:2109.03201*, 2021.

APPENDIX

A DATASET SPARSITY

We define sparsity in the following way:

$$\text{Sparsity} = \frac{\#\text{non\_target\_voxels}}{\#\text{non\_target\_voxels} + \#\text{target\_voxels}}$$

We obtain the dataset sparsity by averaging the sparsity of all the dataset samples. To estimate the benefit of our model STP+MTA+UNETR for different classes, we compare the performance improvement with the sparsity per class. In Table 6, we show the results of BTCV. The two classes for which performance improves the most using STP+MTA+UNETR (Pancreas, +4.03 and Gallbladder, +3.70) are also in the top-3 sparsest classes (Gallbladder 99.97%, Aorta 99.89%, and Pancreas 99.88%). For MSD BraTS, with respect to sparsity, ET $>$ TC $>$ WT. In Table 7, we observe that the order of performance improvement brought by STP+MTA+UNETR correlates positively with the sparsity order. Therefore, our method can exploit the sparsity in data to improve the model performance in both efficiency and effectiveness.

%	Aorta	Gallbladder	Kidney(L)	Kidney(R)	Liver	Pancreas	Spleen	Stomach	Total
Sparsity	**99.89**	**99.97**	99.79	99.79	97.64	**99.88**	99.66	99.35	95.76
UNETR DSC	88.59	70.97	83.38	83.76	95.52	59.76	88.53	74.30	80.78
STP+MTA+UNETR DSC	89.23	73.60	85.66	83.65	95.59	62.17	88.84	77.37	82.18
Perf Improvement	+0.72	+3.70	+1.23	-0.13	+0.07	+4.03	+0.35	+4.10	+1.73

Table 6: Class-wise sparsity and performance on BTCV. In the first row, we show sparsity of each class (8 classes) in BTCV. The second and third rows show class-wise performance of the baseline UNETR and our STP+MTA+UNETR with a pruning ratio of 90% respectively. We observe that the two most improved classes by STP+MTA+UNETR (Pancreas, +4.03 and Gallbladder, +3.70) are also in the top-3 sparsest classes (Gallbladder 99.97%, Aorta 99.89% and Pancreas 99.88%).

%	WT	TC	ET	Total
Sparsity	98.83	99.60	99.81	98.83
UNETR DSC	90.78	82.14	63.27	78.73
STP+MTA+UNETR DSC	90.29	81.94	64.02	78.75
Perf Improvement	-0.54	-0.24	+1.19	+0.03

Table 7: Class-wise sparsity and performance on MSD BraTS validation set. In the first row, we show the sparsity of each class on MSD BraTS (ET $>$ TC $>$ WT). In the second and third rows, we show class-wise performance of the baseline UNETR and our STP+MTA+UNETR with a pruning ratio of 75% respectively. The rank of performance improvement is the same as the rank of sparsity (ET $>$ TC $>$ WT).

B FULL 13 ORGANS PERFORMANCE ON BTCV

DSC%	Avg	Aorta	Gallbladder	Kidney(L)	Kidney(R)	Liver	Pancreas	Spleen	Stomach	Eso	IVC	PVC	RAG	LAG
STP+MTA+UNETR (PatchSize=8)	77.22	89.23	73.60	85.66	83.65	95.59	62.17	88.84	77.37	72.97	81.59	66.92	64.66	61.66

Table 8: Full 13 Organs Performance on BTCV.

C ADDITIONAL ABLATION STUDIES ON BTCV

As Gumbel Softmax trick is also defined on unnormalized probabilities, it could be unnecessary to apply LogSigmoid after score prediction. Thus, we provide additional ablation studies on the application of LogSigmoid after score prediction. Results are shown in Table 9.

LogSigmoid	DSC
No	82.23 ± 0.30
Yes (ours)	82.18 ± 0.12

Table 9: **Additional ablation studies on BTCV**. We ablate whether to apply LogSigmoid after the score prediction. Results show no significant difference.

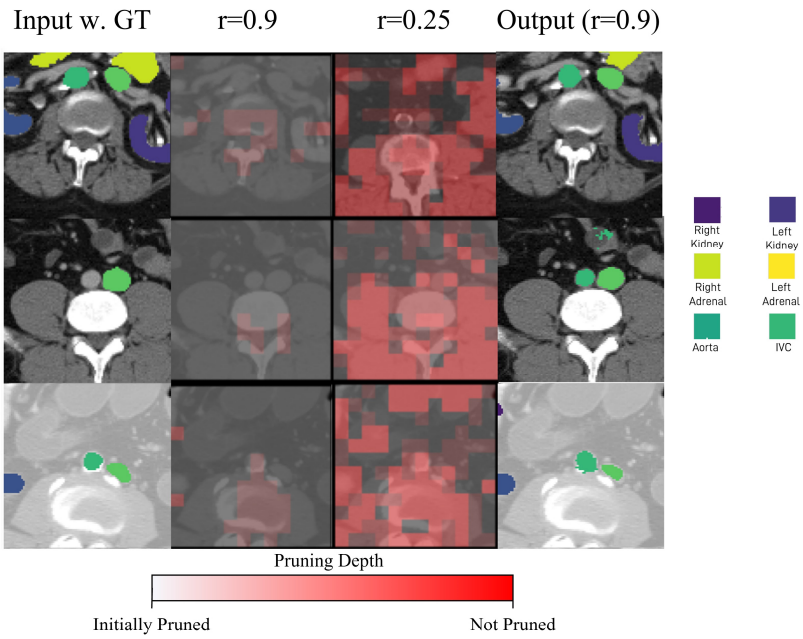

Figure 3: Multi-organ segmentation results are shown here in addition to pruning depth for tokens. Darker red indicates tokens that have been retained longer before being pruned.

D Pruning Policy Visualization on BTCV

Here, we visualize the pruning policy on BTCV in Fig. 3. We observe a similar phenomenon in that patches around target and boundaries are forwarded through more Transformer layers.

E Implementation of DynamicViT/EViT for Sparse Encoding

DynamicViT. As DynamicViT does not guarantee a fixed number of pruned tokens during training, we cannot reuse the pruned tokens in the completion stage by simply concatenating them because concatenation requires a fixed feature shape for batch training. Instead, we use a binary mask to indicate whether each token will be pruned or retained at each pruning location and then sum the tokens at all the pruning locations weighted by the binary masks. To constrain the pruning ratio, we add the ratio loss function $\mathcal{L}_{ratio}$ with a weight of $\lambda_{ratio} = 2$ following (Rao et al., 2021).

EViT. Compared with STP-ViT, EViT has the following differences: First, unlike STP-ViT which predicts the token scores with a learnable score predictor, EViT directly takes the `[CLS]` attention map from the Transformer block as the token scores. The sampling method in EViT is a vanilla topK with no perturbation.

F Gumbel Soft Top-K Pseudocode

To better illustrate the Gumbel Soft Top-K algorithm, we show the PyTorch-style pseudocode as follows. The code is based on the PyTorch official implementation of Gumbel Softmax.

```python
def gumbel_soft_topk(logits, k, tau=1., dim=-1):
    gumbels = (
        -torch.empty_like(logits).exponential_().log()
    )  # ~Gumbel(0,1)
    gumbels = (logits + gumbels) / tau  # ~Gumbel(logits,tau)
    y_soft = gumbels.softmax(dim)

    # Straight through.
    index = y_soft.topk(k, dim=dim)[1]
    y_hard = torch.zeros_like(logits).scatter_(dim, index, 1.0)
    ret = y_hard - y_soft.detach() + y_soft

    return ret
```

Listing 1: Gumbel Soft Top-K Pytorch Pseudocode

G Data Preprocessing and Augmentation

For BTCV, we clip the raw values between -958 and 326, and re-scale the range between -1 and 1. During training, we randomly flip and crop a $96 \times 96 \times 96$ 3D volume as the input. For MSD BraTS. we perform an instance-wise normalization over the non-zero region per channel. During training, we randomly flip and crop a $128 \times 128 \times 128$ 3D volume as the input.