Masked LoGoNet: Fast and Accurate 3D Image Analysis for Medical Domain

Amin Karimi Monsefi
The Ohio State University
Columbus, Ohio, USA
karimimonsefi.1@osu.edu

Payam Karisani
University of Illinois at Urbana-Champaign
USA
karisani@illinos.edu

Mengxi Zhou
The Ohio State University
Columbus, Ohio, USA
zhou.2656@osu.edu

Stacey Choi
The Ohio State University
Columbus, Ohio, USA
choi.1080@osu.edu

Nathan Doble
The Ohio State University
Columbus, Ohio, USA
doble.2@osu.edu

Heng Ji
University of Illinois at Urbana-Champaign
USA
hengji@illinois.edu

Srinivasan Parthasarathy
The Ohio State University
Columbus, Ohio, USA
srini@cse.ohio-state.edu

Rajiv Ramnath
The Ohio State University
Columbus, Ohio, USA
ramnath.6@osu.edu

ABSTRACT

Standard modern machine-learning-based imaging techniques have faced challenges in medical applications due to the high cost of dataset construction and, thereby, the limited labeled training data available. In this paper, we introduce a new neural network architecture, termed LoGoNet, with a tailored self-supervised learning (SSL) method to mitigate such challenges. LoGoNet integrates a novel feature extractor within a U-shaped architecture, leveraging Large Kernel Attention (LKA) and a dual encoding strategy to capture both long-range and short-range feature dependencies adeptly. This combination of strategies is especially beneficial in medical image segmentation, given the difficulty of learning intricate and often irregular body organ shapes. Complementarily, we propose an SSL method tailored for 3D images to compensate for the lack of large labeled datasets. The method combines masking and contrastive learning techniques within a multi-task learning framework and is compatible with both Vision Transformer (ViT) and CNN-based models. We demonstrate the efficacy of our methods in two tasks across two datasets (i.e., BTCV and MSD). Benchmark comparisons with eight state-of-the-art models highlight LoGoNet's superior performance in both accuracy and latency. Code is available at ... (we removed it due to the anonymity)

CCS CONCEPTS

• **Do Not Use This Code → Generate the Correct Terms for Your Paper**; *Generate the Correct Terms for Your Paper*; Generate the Correct Terms for Your Paper; Generate the Correct Terms for Your Paper.

KEYWORDS

Medical Imaging, Image Segmentation, Dual-Encoder, Self-Supervised Learning, Multi-task learning

ACM Reference Format:
Amin Karimi Monsefi, Payam Karisani, Mengxi Zhou, Stacey Choi, Nathan Doble, Heng Ji, Srinivasan Parthasarathy, and Rajiv Ramnath. 2024. Masked LoGoNet: Fast and Accurate 3D Image Analysis for Medical Domain. In *Proceedings of Make sure to enter the correct conference title from your rights confirmation emai (ACM KDD 2024).* ACM, New York, NY, USA, 17 pages. https://doi.org/XXXXXXX.XXXXXXX

1 INTRODUCTION

Accurate medical image segmentation can facilitate disease diagnosis and treatment planning [19, 50]. One of the fundamental difficulties in this task is the presence of organs or structures that span a large receptive field. These structures may have irregular shapes, complex boundaries, or significant variations in appearance, making the segmentation task particularly demanding. Additionally, the high cost of expert annotation in this domain restricts the availability of large-scale labeled datasets. Consequently, it limits the applicability of general domain computer vision techniques [4, 14, 17, 47]. Additionally, processing a large volume of images on a daily basis demands a substantial computational resources and can leave a large carbon footprint [9, 26, 31].

In the present work, we propose a fast and accurate architecture for image segmentation in the medical domain. We also propose a pretraining algorithm to exploit unlabeled images, and therefore, alleviate the demand for human annotation.

Our model is based on the widely adopted U-shaped architecture. We particularly employ two strategies to enhance the inference speed and maintain the prediction accuracy.

ACM KDD 2024, August 25–29, Barcelona, Spain

ACM ISBN 978-1-4503-XXXX-X/18/06
https://doi.org/XXXXXXX.XXXXXXX

————-

To overcome these challenges, we propose three novel techniques in this work. First, we enhance feature learning by proposing a U-shaped architecture that employs the large-kernel attention model (LKA) [16] to encode input images, which we termed **ULKANet** (**U**net **L**arge **K**ernel **A**ttention **Net**work). In contrast to ULKANet, existing methods mostly rely on Convolutional Neural Networks (CNNs) and Vision Transformers (ViTs) as encoders [12, 29]. As we discuss in the next section, these existing models suffer from a high memory complexity, slow, and lack a proper strategy to process image sequences.[1] Our strategy is demonstrably more effective and efficient due to the presence of LKA in our encoder, enabling enhanced encoding capabilities.

Simultaneously, learning short-range and long-range dependencies is essential in medical image segmentation due to the large receptive field of organs. As our second novelty, we further enhance feature learning by incorporating ULKANet into a novel dual encoding approach to learn local (short-range) as well as global (long-range) features. We term this model **LoGoNet** (**Lo**cal and **Glo**bal **Net**work). LoGoNet is able to capture intricate patterns and contextual information at various scales. In contrast, existing studies employ U-Net with the attention mechanism to handle feature dependencies [5, 35, 38]. Our technique is particularly advantageous for segmenting organs such as the spleen, which has an elongated shape and irregular corners. Such body organs demand the extraction of global and local features for segmentation.

Finally, we propose a novel self-supervision technique for 3D images to address the lack of labeled training data. Our self-supervision method combines masking and multi-task learning. Using a multi-clustering algorithm, we generate a list of pseudo-labels for each unlabeled image. We then methodically mask selected parts of these images to implicitly feed the structural information of the unlabeled data into our model. An outstanding feature of our proposed SSL technique lies in its versatility, as it seamlessly supports both CNN and ViT-based models. This flexibility sets our strategy apart from conventional SSL approaches, which often cater to a specific architecture [22, 30, 53]. Furthermore, our innovative strategy leverages the inherent characteristics of 3D medical images, specifically embracing the concept of sequential images and neighborhood information of voxels in 3D images. This unique utilization enhances pre-trained models, representing a pioneering approach that has not been explored previously.

We evaluate our techniques on two tasks across two datasets, i.e., the BTCV dataset [15] for segmenting body organs, and the MSD [37] encompasses diverse tasks in medical imaging, ranging from liver tumors to cardiac and lung segmentation. It consists of meticulously labeled data across ten semantic segmentation challenges, providing a comprehensive benchmark for evaluating the generalizability of machine learning algorithms in the field of medical imaging. Additionally, we benchmark our method against eight state-of-the-art baseline models. The results demonstrate the effectiveness and efficiency of our techniques. To offer a thorough insight into the unique attributes of our approach, we undertook extensive experiments, meticulously showcasing our model's distinguishing features and capabilities. To summarize, our contributions are threefold:

- We propose a resource-efficient model based on the commonly used U-shaped architecture. Our model achieves the least average inference time and, at the same time, outperforms state-of-the-art methods. We achieve this by employing two strategies: first, instead of relying on CNN or ViT-based techniques, we utilize the large-kernel attention method to reduce computational complexity. Second, instead of vertically scaling up our network to improve feature extraction, we use a dual encoding algorithm to facilitate the task. We empirically demonstrate that our strategies combined achieve the best inference time and the highest precision.
- We propose a multi-task self-supervision technique to overcome the lack of labeled data by employing a new masking approach specifically designed for 3D images.
- We evaluate the efficacy of our model in two tasks across two datasets, and show that it outperforms eight state-of-the-art baseline models.

2 RELATED WORK

To model long-range dependencies in images, existing studies mostly use vision transformers [2, 7, 13, 16, 18, 19, 23, 42, 44], and draw ideas from sequence modeling in natural language processing (NLP) areas. A limitation of these approaches is their treatment of images as 1D sequences, thereby overlooking the input's inherent 2D or 3D structure. They struggle to grasp the spatial relationships between pixels, leading to poor performance in tumor detection or organ segmentation tasks. Additionally, they suffer from quadratic memory complexity, leading to high processing costs and slowness for high-resolution images, especially in the 3D context. In contrast, our proposed model, ULKANet, adopts an attention mechanism with LKA[2] to handle long-range dependencies while preserving the spatial structure of the images. This distinctive property enables our model to capture spatial patterns of the input more effectively, resulting in more informative representations. This is particularly advantageous in detecting tumors, where the conditions may extend over a considerable area, and models that rely solely on local features often fail to detect such cases [43].

In addressing dependencies within data, various techniques are employed based on the range of the dependencies. CNN-based models have proven effective for short-range dependencies, leveraging convolutional operations to identify relevant spatial patterns efficiently. Through this approach, hierarchical representations are learned, enhancing the understanding of the intrinsic structure of the data [29, 34, 45, 54]. However, our methodology takes a comprehensive approach, recognizing the importance of long and short-range dependencies. We adopt a dual encoding strategy to achieve this, incorporating an attention mechanism in parallel mode. This dual encoding technique enables the simultaneous capture and encoding of both types of dependencies, providing a more holistic representation of the underlying relationships in the data.

[1] The term "sequence" in 3D medical imaging refers to a series of volumetric data that can be either a temporal sequence, capturing changes over time in a specific anatomical region, or a spatial sequence, consisting of different slices from a 3D volume to provide a comprehensive view of the anatomy from various angles.

[2] LKA [16] is a method for computer vision tasks that effectively captures long-range relationships from input features. LKA reduces computational costs while generating attention maps highlighting essential features without additional normalization functions by decomposing large kernel convolutions into spatial local, long-range, and channel convolutions.

Next, the lack of labeled training data is a primary challenge in medical image analysis. To address this challenge, some studies have focused on domain-specific pretext tasks, as seen in Ahn et al. [1], Cao et al. [6], He et al. [24], Xu and Adalsteinsson [48], Zhao et al. [51], Zhu et al. [56]. Others, such as Zhou et al. [52], adapt contrastive learning techniques to suit medical data by focusing on feature level contrast, creating homogeneous and heterogeneous data pairs by mixing image and feature batches, and utilizing a momentum-based teacher-student architecture. A comprehensive evaluation of various SSL strategies for 3D medical imaging was conducted by Taleb et al. [39]. Azizi et al. [3] demonstrated the benefits of pre-training a model on ImageNet for dermatology image classification, showcasing the potential of transfer learning in the medical imaging domain.

Additionally, Tang et al. [40] combined inpainting with contrastive learning to improve medical segmentation. Recently, Chen et al. [11] introduced a masked approach as a pretext task for 3D medical images. Their method centers around the task of reconstructing the masked regions of images, essentially treating it as a single-task objective. However, in contrast to this approach, our work builds upon the concept of masking but focuses on predicting (pseudo-)labels for masked images. This constitutes a multi-label learning framework where each masked image is associated with more than one label.

Our work notes that capturing information in 3D medical images not only relies on individual images but also involves considering the sequence of images. To utilize this additional source of information, we propose a multi-learning approach employing a batch of clustering algorithms. These algorithms aid in establishing multiple labels for each image, enabling the model to learn the data characteristics from various aspects.

3 PROPOSED MODEL

Figure 1a illustrates the architecture of our model LoGoNet, which stands for Local and Global Networks. The forward pass begins by processing the input data in parallel. We have two modules in this stage, the global and the local modules. In the global module, the original data cube[3] is fed into our feature extractor (ULKANet). In the local module, the same data cube is partitioned into smaller cubes, and then, each cube is processed by a separate feature extractor. Thereafter, the resulting feature tensors are concatenated to reconstruct the input. Then the outputs of the global and the local modules are aggregated by an element-wise summation operator– note that they have the same dimensions. Finally, the resulting tensor is passed through a convolution kernel followed by a 3D batch normalization operator and a GELU activation function to shape the input to our classifier. Our final classifier is a convolution kernel.

In the next section, we discuss our $3D$ encoder-decoder architecture (ULKANet), which is armed with a $3D$ adaptation of LKA in the encoding phase. We then explain our local-global dual encoding strategy, which enables our model to extract feature dependencies at various scales. After describing our model in detail in Sections 3.1 and 3.2, we then explain our novel pre-training method in Section 3.3. We use this pre-training algorithm to initialize the parameters of our model before beginning to fine-tune the network on labeled data.

3.1 U-Shaped Feature Extractor

Figure 1b illustrates an overview of our feature extractor (ULKANet), which has an encoder and a decoder. The encoder consists of a sequence of blocks. Each block consists of a repeating sequence of three components: a patch embedding component, a chain of transformer-like modules that employ LKA (L_i modules for i^{th} block of the encoder), and a layer normalization component. For conciseness, Figure 1b only shows the top-level blocks, while a detailed illustration of the model architecture and inner components is provided in the appendix section 7.

The Patch Embedding component plays a crucial role in the processing of input data within the encoder block, transforming the input into a tensor that is subsequently passed to the next component in the sequence. Throughout the current encoder block, the dimension of the embedding vectors remains constant, denoted as dim.

The mathematical representation of the projection operation is defined as follows:

$$Patch = Norm(Conv3D(X, dim, k, padding = \frac{k}{2})).flatten(2), \tag{1}$$

where X represents the input with five dimensions (b, C, seq, H, W), and b is the batch size, C is the channel size, k is the size of the 3D convolution kernel, dim is the number of channels for the output of Conv3D, and Norm represents the batch normalization operator. (seq, H, W) denotes the size of the 3D input, and the flatten operation results in a tensor with dimensions $(b, dim, \text{seq} \times H \times W)$. The Patch Embedding process serves to efficiently capture and represent the relevant features of the input data, facilitating the subsequent stages of the network architecture.

To enable our model to efficiently extract complex feature dependencies that are often present in medical images, we opt for using transformer modules. However, instead of using the regular transformers with self-attention that is slow and needs more memory, we use LKA [16] in the attention layer. This type of attention mechanism decomposes large convolution kernels into spatial dependencies and channel convolutions. It enables our model to go deeper and remain memory efficient. The attention module is implemented as follows:

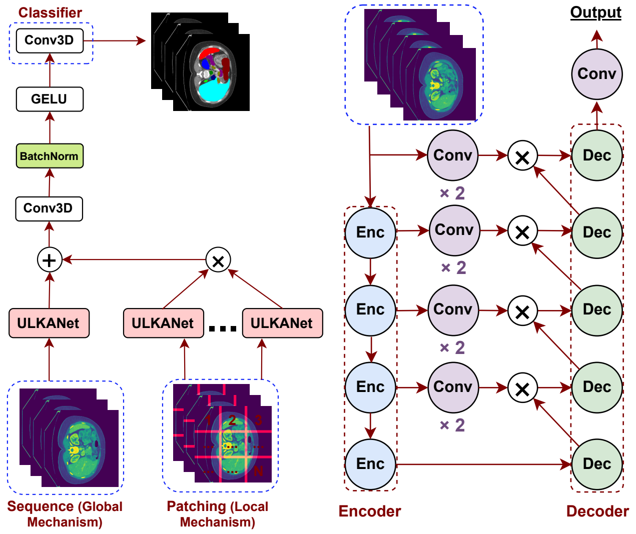
$$Atts = Conv3D_{1\times1}\left(DiConv3D\left(ChConv3D(X)\right)\right), \tag{2}$$

where X is the input tensor and *ChConv3D* is a depth-wise convolution operating on a single channel. *DiConv3D* is a dilated depth-wise convolution to broaden the receptive field and to enable the extraction of long-range dependencies. The point-wise convolution $Conv3D_{1\times1}$ is applied to aggregate the information across the channels. The final activations are obtained as follows:

$$Attention\ Value = Atts \odot X, \tag{3}$$

where $\odot$ is the element-wise product. The remaining components of the transformer block follow the conventional structure of typical transformers.

[3] "Cube" typically refers to a three-dimensional (3D) region of interest (ROI) within the volumetric medical image. Medical images, such as those obtained from MRI or CT scans, are often represented as 3D volumes, where each voxel (3D pixel) contains intensity or other information about the tissue or structures being imaged. A cube in this scenario is a 3D subset of the entire image volume.



(a) Proposed Model **(b) ULKANet**

Figure 1: 1a) Overview of our model LoGoNet. In order to take into account the local and global feature dependencies in images, they are fed into the model in parallel. In the local mechanism, the input data is partitioned into small parts, and each part is separately fed into our feature extractor (ULKANet). 1b) Overview of the ULKANet Architecture. A U-shaped network with the encoder-decoder design. Blue circles represent encoder blocks, and green circles represent the decoder clocks. The + sign represents element-wise summation, and the × sign represents the concatenation operator.

The decoder in our model aims to restore the spatial resolution of the input using a sequence of blocks (green circles in Figure 1b). Each decoder block consists of a chain of three convolution modules followed by an upsampling operation. The convolution modules are responsible for volumetric convolution operation. They consist of a Conv3D layer and a batch normalization layer, followed by a LeakyReLU activation function. The upsampling operation scales the resolution by a factor of two. As we stated earlier, a second larger illustration of our architecture that shows the inner modules can be found in the appendix section 7.

For each individual block in the encoder, the decoder has one corresponding block. There is also an additional decoder block in the bottleneck layer, as shown in Figure 1b. The input to each decoder block is supplied by the block in the previous layer and also the corresponding encoder block through a skip connection. In order to enhance the reconstruction of input, we use the skip connections to facilitate the transfer of high-level features [49] to the layers that are responsible for the reconstruction task.

In the next section, we discuss our dual encoder strategy for capturing long- and short-range dependencies.

3.2 Dual Encoding Strategy

One of the difficulties in medical image segmentation is the presence of organs that have complex shapes. For instance, the human gallbladder and spleen have an elongated structure. Hence, to achieve satisfactory performance in the segmentation task, the model should be able to detect and extract relevant features in multiple regions of the input images, heavily relying on global features. On the other hand, this organ has irregular corners. This characteristic requires the model to be able to detect local features in multiple regions of the input, which needs a deep and slow model. While having a large amount of labeled data will likely enable the model to automatically understand these characteristics, due to the high cost of dataset construction, this is generally not feasible in the medical domain.

To reduce the burden of automatic feature mining and, consequently, to reduce the need for labeled data, we propose to impose an inductive bias [33] on the feature extraction process. We propose to have two feature extractors in parallel, one focusing on the global scale and another one focusing on the local scale–as shown in Figure 1a. The global module is able to extract long-range dependencies due to access to the original data cube. On the other hand, the local module focuses on short-range dependencies. This is accomplished by partitioning the input cube into smaller ones, allowing for a more focused analysis and resulting in finer-grained features.

To implement our idea, we use one instantiation of ULKANet in the global module, and a sequence of N instantiations of ULKANet in the local module. To prepare the data for the local module, the input 3D image is split into N smaller cubes of size $B \times B \times B$. Given an image of size $S \times H \times W$, the value of B is obtained by $B = \sqrt[3]{\frac{S\times H\times W}{N}}$.

To reconstruct the input data cube, the outputs of the local module are concatenated, as shown in Figure 1a. In order to aggregate the outputs of the global and local modules, we use an element-wise summation operator. The resulting tensor is expected to represent both global and local range dependencies.

3.3 Pre-Training Method

Before fine-tuning our model on labeled data, we utilize a multitask pre-training technique to put the model weights in a favorable state. This self-supervised approach allows the model to learn general information from 3D medical images, without the necessity of ground-truth labels.

Pre-training of our model is done in three stages. First, we methodically mask certain regions of the input images. In this stage, the goal is to capture long-range and short-range feature dependencies. Second, we generate pseudo-labels for the masked images. The model later learns to generalize to unseen cases by predicting the pseudo-labels of the masked data. Finally, the masked images, along with their pseudo-labels, are used to pre-train the model. Below we explain each step.

3.3.1 Masking Algorithm.

In 3D imaging, objects are depicted across multiple 2D surfaces. Therefore, we argue that an effective masking strategy should step beyond 2D inputs.

In order to help the model explore not only the dependencies between pixels in 2D images but also the connections among pixels that form 3D masses, we propose an algorithm to mask chains of

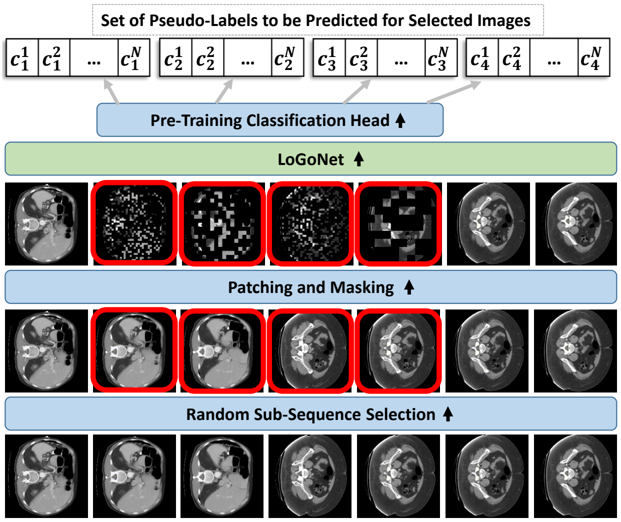

Set of Pseudo-Labels to be Predicted for Selected Images
c_1^1 c_1^2 ... c_1^N c_2^1 c_2^2 ... c_2^N c_3^1 c_3^2 ... c_3^N c_4^1 c_4^2 ... c_4^N
Pre-Training Classification Head
LoGoNet
Patching and Masking
Random Sub-Sequence Selection

Figure 2: Illustration of our pre-training pipeline. We begin by randomly selecting a set of m sequential images (here m is four), on which we apply patching and masking. Then LoGoNet (or ULKANet) is used to predict the set of pseudo-labels that we generated for each distorted image (see Section 3.3 for details). During the pre-training stage, a classification head (a feed-forward network) is used on top of the model for prediction. This head is replaced with a convolution head (see Figure 1a) for fine-tuning on the segmentation task with labeled data.

patches in an image sequence.[4] We begin by randomly selecting an image from the set of unlabeled data, with probability ϕ_1 for selecting an individual image. Along the selected image, we also retrieve the $m-1$ preceding images in the same sequence. Then, we apply a masking technique to the images in the chain. Various masking techniques can be used in this stage [32, 36], and we employ the method introduced by Xie et al. [46]. Therefore, for each image in the chain, we randomly select a patch size P, and partition it into $\frac{H \times W}{(P_j)^2}$ patches, where H and W are the height and width of the image. Finally, with the probability ϕ_2 we mask out each patch of the image. Appendix 8.1 shows details about the masking algorithm and how we find the hyperparameter.

In contrast to algorithms such as SimMIM [46], our proposed approach distinguishes itself by selecting a sequence of images and subsequently applying masking to that sequence. This method facilitates the encoder in gathering information by focusing on the interdependence of voxels within the sequence of images. Notably, our algorithm operates independently of the specific model structure, diverging from approaches seen in studies by Kakogeorgiou et al. [30], He et al. [22], and Zhou et al. [53], all of which exhibit a reliance on model structure. Furthermore, our approach is compatible with Vision Transformer (ViT)-based [12] and CNN-Based models.

3.3.2 Pseudo-Label Generation.

Our pseudo-label generation algorithm assigns labels to all the images in the unlabeled set. Later in the pre-training pipeline, our model is asked to predict the pseudo-labels of the masked out images in each sequence. The information conveyed by the distorted images is insufficient for label prediction. Therefore, the model must explore the associations between pixels across multiple 2D images in the sequence to correctly predict the pseudo-labels of the target images. In the experiments section, we empirically show that this exploration task helps the model to learn the properties of the domain and to generalize better.

A clustering algorithm is employed for the pseudo-label generation. For simplicity, we use the k-means clustering method, although other types of clustering methods, such as hierarchical or spectral methods, can be utilized. Given a random number k as the predefined number of clusters, we train a k-means clusterer on a random subset (e.g. 10% in our experiments) of the unlabeled data. Then we use the clusterer to label the entire unlabeled set. Note that masking is not applied in any of these stages, and the clusterer has access to the unmasked images. The obtained labels are used as pseudo-labels to pre-train the model by predicting the corresponding labels for every masked image.

The k-means clusterer is able to use all the properties of the images to form the clusters. For instance, a cluster may constitute images that illustrate elongated organs, while another cluster may constitute images that depict organs that have particular corners. During pre-training, the model is asked to recover the pseudo-labels of a sequence of images that are distorted by masking. In order to predict their correct labels, the model must discover the associations between neighboring pixels. This pretext task enables the model to learn long-range and short-range spatial dependencies effectively.

Assuming that a clustering method exploits a finite set of characteristics in data to form the clusters, our model needs to learn these characteristics to correctly assign each image to the associated clusters. We conjecture that having N different clusterers labeling the data and then using our model to simultaneously predict these multiple labels can further help the model gain broader knowledge from the data. From a different perspective, we can assume that recovering the characteristics of each clusterer is a separate pre-training task, and then, concurrently recovering the characteristics of multiple clusterers is a multi-task training. The efficacy of multi-tasking is well-documented in the machine learning literature [8]. Figure 2 shows our pre-training pipeline. In this figure, N denotes the total number of clusterers, and c_i^j denotes the pseudo-label generated by j-th clusterer for the i-th masked image in the sequence.

3.3.3 Pre-Training Loss Function.

To pre-train our model, we use a cumulative negative log-likelihood function on the model predictions for the masked images as follows:

$$\mathcal{L} = -\sum_{i=1}^{N} \sum_{j=1}^{S} \log(p^i(e|x_j)), \tag{4}$$

where N is the number of clusterers, S is the number of masked images that can calculated by $S = M \times Q$, M is the length of sequenced images for masking, and because it is possible to have more than one sequence of masked images we consider Q as a number of having this sequenced image for masking, $x_1, x_2, x_3, ..., x_S$ are all masked images, and $p^i(e|x_j)$ is the probability that the j-th masked image in the sequence (i.e., x_j) is correctly assigned to the pseudo-label e generated by the i-th clusterer. The value of $p^i(e|x_j)$ is calculated

[4] Note that in speech processing, where data is naturally sequential, applying this technique seems to be the default method [28]. However, to our knowledge, we are the first to propose this technique in the computer vision domain.

Models	SegResNetVAE	SwinUNETR	UNETR	UNet++	nnUNet
# Param	3.9 M	62.2 M	101.7 M	84.6 M	30.7 M
FLOPs (G)	15.50	329.84	264.59	4229.20	1250.65
Models	**DiNTS Search**	**DiNTS Instance**	**Attention U-Net**	***LoGoNet***	
# Param	74.1 M	74.1 M	64.1 M	67.5 M	
FLOPs (G)	743.88	743.88	7984.21	246.96	

Table 1: Comparison of Our Proposed Models and Baselines in Terms of Trainable Parameters and FLOPs

by a softmax function on top of the pre-training classification head, which is a simple feed-forward network. This simplicity helps the effect of training just move to the primary model and not to the projection layer. Therefore, given a clusterer, we have:

$$p^{\bullet}(e|x) = \frac{exp(f_e(x)/\tau)}{\sum_{s=1}^{K} exp(f_s(x)/\tau)}, \qquad (5)$$

where $exp(\bullet)$ is the exponential function, K is the number of clusters generated by the clusterer, e is the cluster that the input image x belongs to, and $f_s(x)$ is the s-th logit of the pre-training classification head. The hyper-parameter τ is called the softmax temperature. The value of τ determines the strength of the gradients backpropagated through the network. Lower temperature values increase the magnitude of gradients [27]. This, in turn, reduces the standard deviation of output probabilities–also known as sharpening the posterior probabilities.

Our loss function (Equation 4), iterates over all the predictions that our model makes during the pre-training stage and penalizes for the mistakes. As we discussed earlier, our pretraining framework enables LoGoNet or ULKANet to become familiar with the properties of the domain to generalize better by exploiting unlabeled data. Appendix 8.1 shows details and some results for pre-training. We empirically support this argument in our experiments and analyses (see section 5).

4 EXPERIMENTAL SETUP

In this section, we briefly describe the datasets used in the experiments, provide a list of baseline models we compare to, and also provide an overview of our setup. However, the following items can be found in the appendix section 8 and 9: **1)** a comprehensive description of the datasets; **2)** information of the hardware used; **3)** a detailed explanation about implementation and hyper-parameter tuning.

Datasets. We use two common datasets previously studied by many related works for the validity of our experiments. As the first dataset, we use the BTCV dataset[5] introduced by Gibson et al. [15]. This dataset contains 13 segmentation tasks, and 40 examples are obtained via abdominal CT scans for each task. As the second dataset, we use the MSD dataset[6] introduced by Simpson et al. [37]. This dataset contains a variety of tasks obtained via magnetic resonance imaging (MRI), computed tomography (CT), and positron emission tomography (PET). We use six different tasks from this dataset that contain a total of 900 examples. As the unlabeled data, we use the meta-dataset collected by Tang et al. [40], which consists of 4,500 examples. The images in this dataset (which are all unlabeled) are 3D scans covering a variety of organs. Detailed information about these datasets can be found in the appendix 9.

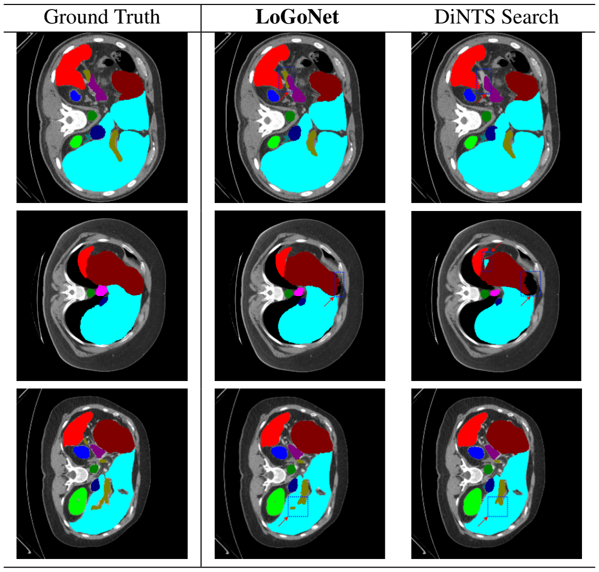


Figure 3: Output of LoGoNet compared to the best performing baseline model in BTCV dataset, i.e., DiNTS Search. We see that our model tangibly outperforms the mentioned model in detecting organ boundaries.

Baselines. We compare LoGoNet to a suite of baseline models, including those that use Visual Transformers or Convolutional Neural Networks. We compare to nnUNet [29], Attention U-Net [35], SegResNetVAE [34], UNet++ [55], DiNTS (two variations of Search and Instance) [25], SwinUNETR (feature size 48) [20], and UNETR (feature size 32) [21].

Table 1 reports a comparison between our models and the baseline models in terms of the number of trainable parameters, FLOPS (the floating point operations per second). This table shows that our model is considered an average-sized network regarding the number of trainable parameters. One noteworthy observation is that in some cases, e.g., nnUNet or DiNTS Instance, even though the number of trainable parameters is on a par or smaller than ours, their FLOPS are substantially higher, indicating that our model is relatively resource efficient. See Appendix 8 for more information.

Setup. We follow standard practices to carry out the experiments (detailed in the appendix 8.2). We use the Dice metric, a common metric for the image segmentation task, to report the performance results. We conduct the experiments in each dataset task separately

[5] Available at https://www.synapse.org/#!Synapse:syn3193805/wiki/217789
[6] Available at http://medicaldecathlon.com/

Models	Spl	RKid	Lkid	Gall	Eso	Liv	Sto	Aor	IVC	Veins	Pan	Rad	Lad	AVG
UNETR	.912	.940	.938	.693	.690	.954	.754	.891	.830	.703	.734	.660	.577	.790
SegResNetVAE	.941	.938	.933	.670	.718	.955	.745	.892	.848	.695	.783	.633	.528	.791
nnUNet	.859	.944	.924	.796	.755	.960	.781	.894	.849	.756	.776	.675	.663	.818
Attention U-Net	.955	.936	.930	.735	.739	.964	.770	.898	.852	.753	.763	.695	.688	.821
DiNTS Instance	.935	.942	.938	.770	.769	.962	.743	.909	.857	.759	.782	.641	.691	.823
UNet++	.934	.931	.925	.810	.715	.961	.786	.900	.846	.747	.829	.685	.679	.827
SwinUNETR	.952	.947	*.945*	.790	.770	.963	.755	.901	.850	.771	.760	.702	.659	.828
DiNTS Search	.937	.934	.930	.788	.770	.960	.774	.904	.866	.751	.813	.670	.711	.831
LoGoNet	.958	**.949**	**.947**	.818	.786	.969	.880	.912	.865	.769	.821	.726	.698	.854
LoGoNet + PRE	**.961**	.947	.944	**.866**	**.845**	**.970**	**.898**	**.936**	**.885**	**.791**	**.838**	**.738**	**.757**	**.875**

Table 2: Performance of our model (in terms of Dice metric) compared to the baseline models in BTCV dataset. All experiments were conducted using identical data splits, computing resources, and testing conditions to ensure a fair comparison. Additionally, to ensure faithfulness to the original implementation of the baseline methods, we used their publicly available implementations available at MONAI network repository. Spl: Spleen, RKid: Right Kidney, LKid: Left Kidney, Gall: Gallbladder, Eso: Esophagus, Liv: Liver, Sto: Stomach, Aor: Aorta, IVC: Inferior Vena Cava, Veins: Portal and Splenic Venis, Pan: Pancreas, Rad: Right Adrenal Glands, Lad: Left Adrenal Glands

and report the average results for five runs in the BTCV dataset and two runs in the MSD dataset.

We begin by reporting the performance of a second variation of our model with more transformer modules in the encoder blocks and larger embedding dimensions that we named large (shows with L postfix). Our default LoGoNet and ULKANet models have four encoder blocks with 3, 4, 6, and 3 transformer modules in each block, respectively. The dimensions of the embedding vectors in these models are 64, 128, 256, and 512, respectively. Our larger variant uses four encoder blocks with 3, 3, 24, and 3 transformer modules, respectively. The dimensions of the embedding vectors in this model are 96, 192, 384, and 768, respectively. Please see appendix 8 for more information.

5 RESULTS

5.1 Main Results

The performance of our proposed LoGoNet is demonstrated through the results presented in Tables 2 and 3. Across both datasets, our model exhibits superior performance compared to baseline methods. Specifically, when evaluating our proposed model without pretraining, it outperforms baselines across 13 out of 19 tasks. Furthermore, incorporating our pre-training strategy into the LoGoNet enhances its performance even further, surpassing baselines in 18 out of 19 tasks. These findings underscore the effectiveness and versatility of our approach in tackling a diverse range of tasks with notable efficacy.

In the BTCV dataset, our LoGoNet variant outperforms the top three baseline models by 2.7%, 3.0%, and 3.2%, respectively. Regarding speed-up, this variant outperforms the top three models by 17.6%, 14.8%, and 118.2%, respectively. A complete analysis of the speed and memory consumption of the models can be found in the appendix 8.

Models	**Col**	**Spl**	**Hep**	**Pan**	**Lun**	**Car**	**AVG**
UNETR	.677	.969	.715	.699	.730	.953	.790
SegResNetVAE	.742	.968	.745	.740	.765	.951	.818
SwinUNETR	.695	.967	.737	.738	.763	.957	.810
nnUNet	.736	.977	.742	.742	.816	.958	.829
Attention U-Net	-	-	-	-	-	-	-
UNet++	.553	.975	.752	.760	.753	.961	.792
DiNTS Search	.776	**.980**	.749	.749	.768	.960	.830
DiNTS Instance	.768	.979	.731	.742	.790	**.963**	.829
LoGoNet	.786	**.980**	.757	.798	.802	.951	.846
LoGoNet + PRE	**.801**	**.980**	**.779**	**.833**	**.828**	.958	**.863**

Table 3: Performance of our model (in terms of Dice metric) compared to the baselines in MSD dataset. The baseline model “Attention U-Net” was not runnable on regular chipsets which each has 35 Gigabyte of memory in MSD dataset. Col: Colon Cancer Primaries, Spl: Spleen, Hep: Hepatic vessels and tumor, Pan: Pancreas Tumour, Lun: Lung Tumours, Car: Cardiac

5.2 Analysis

In this section, we demonstrate the properties of our model from multiple aspects. Specifically, we show the impact of increased model size on performance, demonstrate the efficacy of our pre-training approach, report a segmentation visualization, analyze the hyperparameter sensitivity of our model, and finally, report an ablation study on the steps in our pre-training method. The experiments in this section are carried out in the BTCV dataset unless stated otherwise.

SSL Approach	**Gall**	**Eso**	**Veins**	**Lad**	**AVG**
SimMIM [46]	.837	.829	.785	.733	.864
Rubik’s Cube [41]	.815	.820	.780	.725	.859
SimCLR [10]	.829	.803	.780	.720	.859
Our SSL Approach	**.866**	**.845**	**.791**	**.757**	**.875**

Table 4: Performance Comparison of Our Proposed Multi-Task Self-Supervised Learning Approaches. (Number of Clusterer is N=80)

We conduct a comprehensive comparison of our LoGoNet models with established baselines across various tasks on both labeled datasets. The performance metrics are summarized in tables 2 and 3, revealing that our LoGoNet model consistently outperforms all baselines on average. This superior performance can be attributed to the unique capability of our model to leverage both long-range and short-range dependencies, demonstrating its effectiveness across a range of tasks.

Next, we take a different approach and qualitatively inspect our model. Figure 3 compares the output of LoGoNet to the best performing baseline model in BTCV dataset, i.e., DiNTS Search (and more qualitative comparisons can be found in the appendix 10). We see that our model particularly excels in segmenting organ boundaries. This can be attributed to our effective strategy for extracting local-range dependencies, which plays a crucial role in extracting details from input data.

To elaborate further, our model's adeptness in capturing long-range dependencies allows it to grasp contextual information that extends over significant distances within the data. Simultaneously, its proficiency in handling short-range dependencies ensures precision in capturing localized patterns. This dual advantage positions our LoGoNet model as a robust solution, showcasing its versatility and adaptability in comparison to the considered baselines.

In the second experiment, we report the efficacy of our pre-training method. To carry out this experiment, we use the algorithm proposed in Section 3.3 to initialize the weights of our model, and then, we follow the regular fine-tuning steps. In Tables 2 and 3, We report the results of this model for both datasets, indicated by postfix PRE. We see that the improvements achieved by pre-training are consistent across both datasets.

Models	Gall	Eso	Veins	Lad	AVG
LoGoNet	.818	.786	.769	.698	.854
LoGoNet L	.847	.781	.768	.710	.855
LoGoNet + PRE	.866	.845	.791	.757	.875
LoGoNet L + PRE	**.921**	**.859**	**.805**	**.784**	**.891**

Table 5: Performance Comparison of Our Proposed Model with Different Sizes with and without Pre-training (Number of Clusters N=80, L means Large Size of Model and BTCV dataset)

Upon increasing the dimensions of our model, we observed an improvement in results, though it fell short of our initial expectations. We attribute this to the limited number of labeled data available. However, upon integrating our pre-training methodology into our standard and larger variants of LoGoNet, we noted a significant enhancement in performance, particularly noticeable in the larger LoGoNet. For comprehensive insights, please refer to Table 5, which shows the results for the BTCV dataset(for complete details of the structure of the large model, see Appendix and for complete results, see Appendix 10).

One inquiry worth exploring is the comparative impact of our dual encoder model, LoGoNet, against our feature extractor, ULKANet. Table 6 provides insights into this matter for BTCV dataset. Our dual encoder, LoGoNet, adeptly captures both long and short-range dependencies, offering a comprehensive understanding of the data.

Models	Gall	Eso	Veins	Lad	AVG
ULKANet	.761	.782	.690	.684	.824
LoGoNet	**.818**	**.786**	**.769**	**.698**	**.854**

Table 6: Performance Comparison of LoGoNet and ULKANet on BTCV dataset.

For a comprehensive overview of the results, please refer to the appendix in Section 10.

In the evaluation of our novel multi-task self-supervised learning approach, we conduct a comparative analysis against several baseline self-supervised methods, as outlined in Table 4. This table presents the results obtained from experiments conducted with the LoGoNet model, employing SimMIM [46], Rubuk's Cube [41], and SimCLR [10] strategies. Notably, our proposed model exhibits superior performance in three out of four experiments, showcasing its effectiveness in a diverse set of tasks. The complete results are available in the appendix 10.

The comprehensive comparison in Table 4 highlights the competitive edge of our model, with performance metrics closely aligned with the SimMIM approach. This indicates the robustness and efficacy of our multi-task self-supervised learning methodology in capturing meaningful representations across various domains.

An inherent advantage of our approach lies in its versatility, as it is designed to be compatible with both CNN and ViT-based models. This flexibility broadens the applicability of our approach, allowing it to seamlessly integrate with different architectural paradigms commonly used in computer vision tasks.

Model	**N = 1**		**N = 40**		**N = 80**	
	Gall	**Eso**	**Gall**	**Eso**	**Gall**	**Eso**
LoGoNet + PRE	.830	.819	.843	.860	.866	.845

Table 7: Performance of our models at varying number of clusterers for pre-training. As the number of clusterers increases, the contribution of multi-tasking becomes more noticeable.

In Section 3.3, we claimed that having multiple clusterers serves as a multi-task training approach. In order to demonstrate the benefit of having various clusterers, and also show the sensitivity of our model to the number of these learners in our algorithm, we report the results of our model with varying numbers of clusterers in Table 7. We see that as the number of clusterers increases, the performance improves. The results support our hypothesis regarding the ability of our model to extract broader knowledge from the unlabeled data in the presence of multi-tasking.

Finally, we report an ablation study on the effectiveness of our masking approach during the pre-training stage. In Section 3.3, we argued that by distorting input images, the model must learn the properties of neighboring pixels in order to predict the correct labels. We then argued that this exploration task enables the model to faster learn the domain and to generalize better. The results reported in Table 8 supports our claim. We see that by incorporating the masking step, the performance noticeably improves signifying a better generalizablity of our method.

Model	w/ M		wo/ M		w/ M + wo/ M	
	Gall	Eso	Gall	Eso	Gall	Eso
LoGoNet + PRE	.866	.845	.845	.802	.851	.820

Table 8: Ablation study on the effectiveness of our masking algorithm for 3D inputs. "w/ M" refers to pretraining with masking, and "wo/ M" refers to pretraining without masking. (BTCV Dataset)

In summary, we demonstrated the efficacy of our model in two datasets across 19 segmentation tasks. We also compared our method to eight recent baseline models, including those that use Visual Transformers. Our results testify to the effectiveness of our novel feature extraction techniques. Our analysis shows that our pre-training method is successfully able to exploit unlabeled data to improve parameter initialization. We also showed that our method significantly speeds up inference time compared to the best-performing models.

The computer vision domain is a rapidly evolving research field. It seems unrealistic to expect long-term plans. However, with the existing challenges in the medical domain, this community will invest more in developing methods for mitigating the lack of large labeled sets. Therefore, in the next step, we plan to explore Domain Adaptation, which is one of the well-known methods for addressing this challenge.

6 CONCLUSIONS

In this paper, we proposed a novel approach for 3D medical image segmentation using the LoGoNet architecture, which combines global and local attention mechanisms. The localized mechanism in LoGoNet significantly improves segmentation, especially for small organ sections, while the incorporation of both global and local scales captures effective long-range dependencies. Additionally, we proposed a pre-training method to exploit unlabeled data to enhance model generalization. Experiments in the BTCV and MSD datasets demonstrated that LoGoNet surpasses the baselines, achieving superior segmentation accuracy. The combination of LoGoNet with pretraining further enhances performance. The utilization of masked data in pretraining framework significantly boosts the model performance to capture long-range dependencies, leading to a deeper understanding of structural relationships within 3D images, thereby improving segmentation accuracy.

REFERENCES

[1] Euijoon Ahn, Dagan Feng, and Jinman Kim. 2021. A spatial guided self-supervised clustering network for medical image segmentation. In *Medical Image Computing and Computer Assisted Intervention–MICCAI*. Springer.

[2] Bobby Azad, Reza Azad, Sania Eskandari, Afshin Bozorgpour, Amirhossein Kazerouni, Islem Rekik, and Dorit Merhof. 2023. Foundational models in medical imaging: A comprehensive survey and future vision. *arXiv preprint arXiv:2310.18689* (2023).

[3] Shekoofeh Azizi, Basil Mustafa, Fiona Ryan, Zachary Beaver, et al. 2021. Big self-supervised models advance medical image classification. In *Proceedings of the IEEE/CVF international conference on computer vision*.

[4] Yu Cai, Hao Chen, Xin Yang, Yu Zhou, and Kwang-Ting Cheng. 2023. Dual-distribution discrepancy with self-supervised refinement for anomaly detection in medical images. *Medical Image Analysis* 86 (2023), 102794.

[5] Yutong Cai and Yong Wang. 2022. Ma-unet: An improved version of unet based on multi-scale and attention mechanism for medical image segmentation. In *Third International Conference on Electronics and Communication; Network and Computer Technology (ECNCT 2021)*, Vol. 12167. SPIE, 205–211.

[6] Bing Cao, Han Zhang, Nannan Wang, Xinbo Gao, and Dinggang Shen. 2020. Auto-GAN: self-supervised collaborative learning for medical image synthesis. In *Proceedings of the AAAI conference on artificial intelligence*.

[7] Hu Cao, Yueyue Wang, Joy Chen, et al. 2023. Swin-unet: Unet-like pure transformer for medical image segmentation. In *Computer Vision–ECCV*.

[8] Rich Caruana. 1997. Multitask Learning. *Mach. Learn.* (1997). https://doi.org/10.1023/A:1007379606734

[9] Chen Chen, Xiaopeng Liu, Meng Ding, Junfeng Zheng, and Jiangyun Li. 2019. 3D dilated multi-fiber network for real-time brain tumor segmentation in MRI. In *Medical Image Computing and Computer Assisted Intervention–MICCAI 2019: 22nd International Conference, Shenzhen, China, October 13–17, 2019, Proceedings, Part III 22*. Springer, 184–192.

[10] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*.

[11] Zekai Chen, Devansh Agarwal, Kshitij Aggarwal, Wiem Safta, Mariann Micsinai Balan, and Kevin Brown. 2023. Masked image modeling advances 3d medical image analysis. In *Proceedings of the IEEE/CVF Conference on WACV*.

[12] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, et al. 2020. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929* (2020).

[13] Sania Eskandari, Janet Lumpp, and Luis Sanchez Giraldo. 2023. Skin Lesion Segmentation Improved by Transformer-Based Networks with Inter-scale Dependency Modeling. In *International Workshop on Machine Learning in Medical Imaging*. Springer, 351–360.

[14] Sania Eskandari, Saeed Meshgini, and Ali Farzamnia. 2022. Using a novel algorithm in ultrasound images to detect renal stones. In *Proceedings of the 12th National Technical Seminar on Unmanned System Technology 2020: NUSYS'20*. Springer, 755–767.

[15] Eli Gibson, Francesco Giganti, and et al Hu. 2018. Automatic multi-organ segmentation on abdominal CT with dense V-networks. *IEEE transactions on medical imaging* (2018).

[16] Meng-Hao Guo, Cheng-Ze Lu, Zheng-Ning Liu, et al. 2022. Visual attention network. *arXiv preprint arXiv:2202.09741* (2022).

[17] Fatemeh Haghighi, Mohammad Reza Hosseinzadeh Taher, Michael B Gotway, and Jianming Liang. 2022. DiRA: Discriminative, restorative, and adversarial learning for self-supervised medical image analysis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 20824–20834.

[18] Kai Han, An Xiao, Enhua Wu, et al. 2021. Transformer in transformer. *Advances in Neural Information Processing Systems* (2021).

[19] Ali Hatamizadeh, Vishwesh Nath, Yucheng Tang, et al. 2021. Swin unetr: Swin transformers for semantic segmentation of brain tumors in mri images. In *International MICCAI Brainlesion Workshop*.

[20] Ali Hatamizadeh, Vishwesh Nath, Yucheng Tang, et al. 2021. Swin unetr: Swin transformers for semantic segmentation of brain tumors in mri images. In *International MICCAI Brainlesion Workshop*.

[21] Ali Hatamizadeh, Yucheng Tang, Vishwesh Nath, et al. 2022. Unetr: Transformers for 3d medical image segmentation. In *Proceedings of the IEEE/CVF Conference on WACV*.

[22] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. 2022. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 16000–16009.

[23] Sheng He, P Ellen Grant, and Yangming Ou. 2021. Global-local transformer for brain age estimation. *IEEE transactions on medical imaging* (2021).

[24] Yufan He, Aaron Carass, Lianrui Zuo, Blake E Dewey, and Jerry L Prince. 2021. Autoencoder based self-supervised test-time adaptation for medical image analysis. *Medical image analysis* 72 (2021), 102136.

[25] Yufan He, Dong Yang, Holger Roth, et al. 2021. Dints: Differentiable neural network topology search for 3d medical image segmentation. In *Proceedings of the IEEE/CVF Conference on CVPR*.

[26] Leonie Henschel, Sailesh Conjeti, Santiago Estrada, Kersten Diers, Bruce Fischl, and Martin Reuter. 2020. Fastsurfer-a fast and accurate deep learning based neuroimaging pipeline. *NeuroImage* 219 (2020), 117012.

[27] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531* (2015).

[28] Wei-Ning Hsu, Benjamin Bolte, et al. 2021. Hubert: Self-supervised speech representation learning by masked prediction of hidden units. *IEEE/ACM Transactions on Audio, Speech, and Language Processing* (2021).

[29] Fabian Isensee, Jens Petersen, Andre Klein, et al. 2018. nnu-net: Self-adapting framework for u-net-based medical image segmentation. *arXiv preprint arXiv:1809.10486* (2018).

[30] Ioannis Kakogeorgiou, Spyros Gidaris, Bill Psomas, Yannis Avrithis, Andrei Bursuc, Konstantinos Karantzalos, and Nikos Komodakis. 2022. What to hide from your students: Attention-guided masked image modeling. In *European Conference on Computer Vision*. Springer, 300–318.

[31] Jiangyun Li, Junfeng Zheng, Meng Ding, and Hong Yu. 2021. Multi-branch sharing network for real-time 3D brain tumor segmentation. *Journal of Real-Time*

Image Processing (2021), 1–11.

[32] Zhaowen Li, Zhiyang Chen, Fan Yang, et al. 2021. Mst: Masked self-supervised transformer for visual representation. *Advances in Neural Information Processing Systems* (2021).

[33] Thomas M. Mitchell. 1997. *Machine Learning* (1 ed.). McGraw-Hill, Inc., USA.

[34] Andriy Myronenko. 2019. 3D MRI brain tumor segmentation using autoencoder regularization. In *Brainlesion: Glioma, Multiple Sclerosis, Stroke and Traumatic Brain Injuries: 4th International Workshop, Held in Conjunction with MICCAI*.

[35] Ozan Oktay, Jo Schlemper, et al. 2018. Attention u-net: Learning where to look for the pancreas. *arXiv preprint arXiv:1804.03999* (2018).

[36] Yuge Shi, N Siddharth, et al. 2022. Adversarial masking for self-supervised learning. In *International Conference on Machine Learning*.

[37] Amber L Simpson, Michela Antonelli, Spyridon Bakas, Michel Bilello, Keyvan Farahani, Bram Van Ginneken, Annette Kopp-Schneider, Bennett A Landman, Geert Litjens, Bjoern Menze, et al. 2019. A large annotated medical image dataset for the development and evaluation of segmentation algorithms. *arXiv preprint arXiv:1902.09063* (2019).

[38] Ashish Sinha and Jose Dolz. 2020. Multi-scale self-guided attention for medical image segmentation. *IEEE journal of biomedical and health informatics* (2020).

[39] Aiham Taleb, Winfried Loetzsch, Noel Danz, Julius Severin, et al. 2020. 3d self-supervised methods for medical imaging. *Advances in neural information processing systems* (2020).

[40] Yucheng Tang, Dong Yang, et al. 2022. Self-supervised pre-training of swin transformers for 3d medical image analysis. In *Proceedings of the IEEE/CVF Conference on CVPR*.

[41] Xing Tao, Yuexiang Li, Wenhui Zhou, Kai Ma, and Yefeng Zheng. 2020. Revisiting Rubik's cube: self-supervised learning with volume-wise transformation for 3D medical image segmentation. In *Medical Image Computing and Computer Assisted Intervention–MICCAI 2020: 23rd International Conference, Lima, Peru, October 4–8, 2020, Proceedings, Part IV 23*. Springer, 238–248.

[42] Jeya Maria Jose Valanarasu, Poojan Oza, et al. 2021. Medical transformer: Gated axial-attention for medical image segmentation. In *Medical Image Computing and Computer Assisted Intervention*.

[43] Risheng Wang, Tao Lei, et al. 2022. Medical image segmentation using deep learning: A survey. *IET Image Processing* (2022).

[44] Huisi Wu, Shihuai Chen, et al. 2022. FAT-Net: Feature adaptive transformers for automated skin lesion segmentation. *Medical image analysis* (2022).

[45] Yingda Xia, Fengze Liu, Dong Yang, et al. 2020. 3d semi-supervised learning with uncertainty-aware multi-view co-training. In *Proceedings of the IEEE/CVF Conference on WACV*.

[46] Zhenda Xie, Zheng Zhang, Yue Cao, et al. 2022. Simmim: A simple framework for masked image modeling. In *Proceedings of the IEEE/CVF Conference on CVPR*.

[47] Jiashu Xu. 2021. A review of self-supervised learning methods in the field of medical image analysis. *International Journal of Image, Graphics and Signal Processing (IJIGSP)* 13, 4 (2021), 33–46.

[48] Junshen Xu and Elfar Adalsteinsson. 2021. Deformed2self: Self-supervised denoising for dynamic medical imaging. In *Medical Image Computing and Computer Assisted Intervention–MICCAI 2021: 24th International Conference, Strasbourg, France, September 27–October 1, 2021, Proceedings, Part II 24*. Springer, 25–35.

[49] Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. 2014. How transferable are features in deep neural networks?. In *Annual Conference on Neural Information Processing Systems*.

[50] Boxiang Yun, Yan Wang, et al. 2021. Spectr: Spectral transformer for hyperspectral pathology image segmentation. *arXiv preprint arXiv:2103.03604* (2021).

[51] He Zhao, Yuexiang Li, Nanjun He, Kai Ma, Leyuan Fang, Huiqi Li, and Yefeng Zheng. 2021. Anomaly detection for medical images using self-supervised and translation-consistent features. *IEEE Transactions on Medical Imaging* 40, 12 (2021), 3641–3651.

[52] Hong-Yu Zhou, Shuang Yu, Cheng Bian, et al. 2020. Comparing to learn: Surpassing imagenet pretraining on radiographs by comparing image representations. In *Medical Image Computing and Computer Assisted Intervention*.

[53] Jinghao Zhou, Chen Wei, Huiyu Wang, Wei Shen, Cihang Xie, Alan Yuille, and Tao Kong. 2021. ibot: Image bert pre-training with online tokenizer. *arXiv preprint arXiv:2111.07832* (2021).

[54] Mengxi Zhou, Nathan Doble, Stacey S Choi, , et al. 2022. Using deep learning for the automated identification of cone and rod photoreceptors from adaptive optics imaging of the human retina. *Biomedical Optics Express* (2022).

[55] Zongwei Zhou, Md Mahfuzur Rahman Siddiquee, et al. 2019. Unet++: Redesigning skip connections to exploit multiscale features in image segmentation. *IEEE transactions on medical imaging* (2019).

[56] Jiuwen Zhu, Yuexiang Li, Yifan Hu, Kai Ma, S Kevin Zhou, and Yefeng Zheng. 2020. Rubik's cube+: A self-supervised feature learning framework for 3d medical image analysis. *Medical image analysis* (2020).

APPENDIX

In the next section, we present a more detailed description about our feature extractor ULKANet. Then, we provide the details of our experiments, including the configurations of the baseline models, our pre-training algorithm, and our model architecture. We continue by a description about each used dataset, and finally, we conclude the article by reporting an additional qualitative experiment.

7 DETAILED ARCHITECTURE OF ULKANET

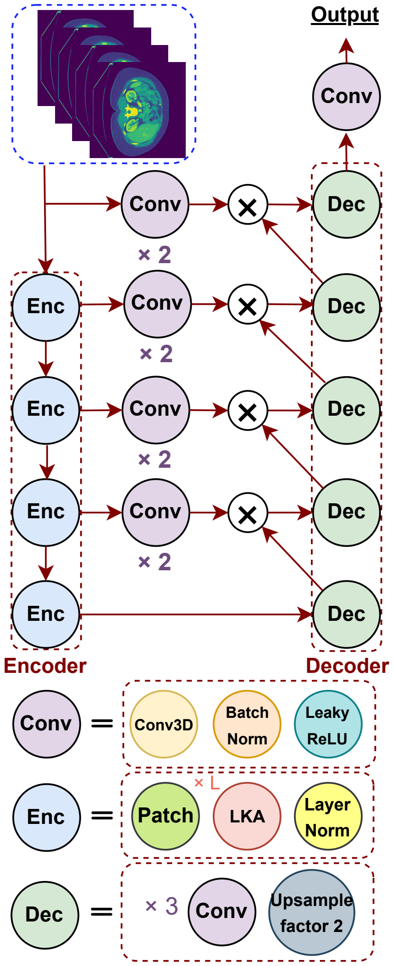


Figure 4: Architecture of our feature extractor (ULKANet). The numbers next to some of the components indicate a sequence of the depicted component with the specified length.

Figure 4 illustrates our feature extractor. This feature extractor is structured into two main components: an encoder and a decoder. The encoder is comprised of a series of blocks, each consisting of a recurring sequence of three essential elements: a patch embedding component, which you can find the algorithm of this component in the algorithm 1, a set of transformer-like modules employing the LKA technique (The number of these modules in the sequence is represented as L), and a layer normalization component. The

LKA component contains two crucial parts, first attention, which we describe in part 3.1, and the MLP part, which you can find in the algorithm 3; also, the algorithm of LKA part is available in the algorithm 2. This architecture has been meticulously designed to process and extract crucial input data features effectively. The patch embedding operation transforms the input into a feature vector with a dimension of *dim*. Additionally, we incorporate a *Conv* block, which encompasses three layers: a *Conv3D* layer, batch normalization, and the *LeakyRelu* activation function.

Furthermore, the presence of a decoder block denoted as *Dec* in Figure 4 is a crucial element. This block consists of three *Conv* blocks and an upsampling layer, which upscales the input by a factor of 2. This comprehensive structure enables our model to efficiently handle the input data and extract meaningful features for further processing.

Algorithm 1 Patch Embedding Pseudo Code

```
procedure PatchEmbed3D($X$, $dim$, $patchSize$, $inputChannel$, $stride$)  ▷ Input: $X$ is the input tensor, and $dim$ is embed dimension
    $projection \leftarrow Conv3D(inputChannel, dim, kernel = patchSize, stride = stride, padding = patchSize//2)$
    $X \leftarrow projection(X)$
    $B, C, D, H, W \leftarrow X.Shape$
    $X \leftarrow BatchNorm(X)$
    $X \leftarrow X.flatten(2).transpose(1, 2)$
    Return X, D, H, W
end procedure
```

Algorithm 2 Pseudo Code of LKA Block

```
procedure LKA($X$, $dim$, $H$, $W$, $mlpRatio$)  ▷ Input: $X$ is the input tensor. $dim$, $H$, and $W$ are the dimensions of the input tensor.
    $B, N, C \leftarrow X.shape$
    $X \leftarrow X.permute(0, 2, 1).view(B, C, dim, H, W)$
    $X \leftarrow BatchNorm(X)$
    $attentionValue \leftarrow attentionFunction(X)$  ▷ The attention function is described before in the part 3.1
    $X \leftarrow X + attentionValue$
    $X \leftarrow BatchNorm(X)$
    $mlpValue = MLP(X, dim, mlpRatio \times dim)$
    $X \leftarrow X + mlpValue$
    $X \leftarrow X.view(B, C, N).permute(0, 2, 1)$
    Return X
end procedure
```

Algorithm 3 Pseudo Code of MLP Block

```
procedure MLP($X$, $inSize$, $hiddenSize$, $outSize$)  ▷ Input: $X$ is the input tensor.
    $fc1 \leftarrow Conv3d(inSize, hiddenSize, kernel = 1)$
    $X \leftarrow fc1(x)$
    $X \leftarrow GELU((X))$
    $dwconv3d \leftarrow Conv3d(inSize, inSize, kernel = 3)$
    $X \leftarrow dwconv3d(X)$
    $X \leftarrow GELU((X))$
    $fc2 =\leftarrow Conv3d(hiddenSize, outSize, kernel = 1)$
    Return X
end procedure
```

8 IMPLEMENTATION DETAILS

In our model architecture, we have incorporated four encoder blocks, a feature present in both the standard and the larger variants. However, it's important to note that our model is flexible and can seamlessly adapt to the use of varying numbers of encoder layers. The primary distinction between the regular and large models lies in the quantity of transformer modules within each block and the dimensions of the internal embedding vectors.

To provide a comprehensive understanding, Table 9 presents a detailed comparison between our standard model and its larger counterpart. It's noteworthy that, despite any variations, the size of the embedding vectors for each patch module and the mlpRatio remains consistent across all encoder blocks.

This structural consistency ensures that the essential characteristics of the model components are preserved, facilitating ease of integration and adaptability. Whether opting for the standard or larger version, users have the freedom to fine-tune the model's performance by adjusting the number of encoder layers to suit their specific requirements. This flexibility is a key advantage of our model, allowing for versatility in handling diverse applications and tasks.

Table 10 presents a comprehensive comparison between our model and various baseline approaches. The reported inference time in this table represents the average across five test runs conducted on the BTCV dataset. This meticulous evaluation allows for a thorough understanding of the performance and efficiency of each method.

In the implementation of the local strategy within LoGoNet, a pivotal decision was made to partition each image tensor into $N = 8$ segments. While this approach offers advantages in enhancing local processing capabilities, it concurrently introduces a significant surge in the number of trainable parameters. In addressing this challenge, a thoughtful strategy has been employed within the local section of LoGoNet.

Specifically, in the local processing segment of LoGoNet, a judicious selection has been made to utilize only two encoder blocks, in contrast to the four blocks employed in the global section, as previously mentioned. This intentional divergence in the number of encoder blocks between the local and global sections serves to strike a balance between computational complexity and model expressiveness.

By limiting the local section to two encoder blocks, we manage to mitigate the potential escalation in trainable parameters, thereby optimizing the trade-off between computational efficiency and model performance. This strategic choice is rooted in a nuanced understanding of the interplay between local and global processing within the overall architecture of LoGoNet.

In essence, our design rationale carefully tailors the number of encoder blocks in each section to the specific demands of local and

Layer Number	**1**			**2**			**3**			**4**		
	L	**dim**	**mlpRatio**	**L**	**dim**	**mlpRatio**	**L**	**dim**	**mlpRatio**	**L**	**dim**	**mlpRatio**
Normal	3	64	8	4	128	8	6	256	4	3	512	4
Large	3	96	8	3	192	8	24	384	4	3	768	4

Table 9: The number of LKA modules in each encoder block and mlpRatio for each encoder layer, as well as the embedding dimensions of the Patch Embedding module for the regular and the large variants of our model.

Models	**SegResNetVAE**	**SwinUNETR**	**UNETR**	**UNet++**	**Attention U-Net**	**nnUNet**
# Param	3.9 M	62.2 M	101.7 M	84.6 M	64.1 M	30.7 M
FLOPs (G)	15.50	329.84	264.59	4229.20	7984.21	1250.65
Time (S)	1.73	6.16	5.89	11.74	19.74	2.87
Models	**DiNTS Search**	**DiNTS Instance**	***ULKANet***	***LoGoNet***	***ULKANet L***	***LoGoNet L***
# Param	74.1 M	74.1 M	40.2 M	67.5 M	121.20 M	172.54 M
FLOPs (G)	743.88	743.88	109.94	246.96	218.70	487.65
Time (S)	6.33	6.28	3.45	5.38	5.06	14.49

Table 10: Comparison of Our Proposed Models and Baselines in Terms of Trainable Parameters and FLOPs

global processing, ensuring a harmonious integration that optimally leverages the strengths of both approaches. This meticulous consideration of architectural choices reflects our commitment to achieving a well-balanced and efficient model in LoGoNet.

8.1 Pre-Training Details

We used the scikit-learn implementation[7] of the Mini Batch KMeans algorithm as the clusterers in our pre-training pipeline. During the training phase of the k-means models, we adopted a transformation process that converted the input image from a $Channel \times X \times Y \times Z$ format to a vector representation of dimensions $Z \times T$, where T is equivalent to $Channel \times X \times Y$. This transformation enabled us to generate a label for each cluster per image slice, resulting in a sequence of labels for a sequence of images. Subsequently, the model underwent 350 iterations of training, with each iteration utilizing a randomly selected 10% subset of the unlabeled data. To introduce diversity and enhance robustness, we employed a stochastic approach in determining the value of K, randomly sampling from a range spanning 80 to 500.

The information pertaining to pre-training is encapsulated in Table 12. To train the pre-trained model, we leveraged the *AdamW* optimizer and fine-tuned the process by configuring specific parameters. In particular, we assigned values of 0.1 and 0.7 to ϕ_1 and ϕ_2 respectively. Additionally, the sequence of distorted images, denoted as M, was set to 5.

Table 11 presents the outcome of selecting hyperparameter values, with results obtained from the BTCV dataset using the ULKANet model. This tabulated information sheds light on the meticulous decision-making process involved in determining specific values for key hyperparameters, providing valuable insights into our experimental configuration.

Hyperparameter	**M = 3**	**M = 5**	**M = 7**
ϕ_1 **= 0.1**	.835	**.850**	.847
ϕ_1 **= 0.2**	.838	.847	.840
ϕ_1 **= 0.3**	.841	.843	.838

Table 11: Hyperparameter Tuning for Sequenced Mask Image Length (M) and Rate of Sampled Images (ϕ_1): A detailed exploration of hyperparameter variations to optimize key aspects of our experimental setup. Result is for BTCV dataset and ULKANet model.

Our observations reveal that augmenting both the values of M (length of sequenced mask images) and ϕ_1 (rate of sampled images) results in an increased rate of masked images. However, this heightened rate poses challenges for our model, making it more intricate to exploit dependencies between successive slices for effectively capturing information related to missing voxels. This delicate interplay between hyperparameters emphasizes the necessity of finding an optimal balance to enhance model performance, as an excessive increase in masked images may impede the model's ability to leverage contextual dependencies within the data.

Furthermore, we introduced randomness in the selection of patch sizes, choosing from the set $(1, 2, 4, 8, 16, 32, 96)$.

Our innovative pre-training approach involves the incorporation of a header designed to adapt the model output to align with the requirements of our pseudo-labeling. Figure 2 shows the structure of our proposed pre-training. The structure of this header can be found in algorithm 4. This header modification serves as a crucial step in optimizing the model's output for seamless integration with our pseudo-labeling methodology during the training process.

8.2 Configuration Setup

Table 14 provides a comprehensive overview of the specifics pertaining to our training or fine-tuning procedures. This tabulated information encapsulates crucial details, offering insights into the intricacies of our training regimen. By examining the contents of this table, readers can gain a nuanced understanding of the parameters, configurations, and methodologies employed during the training or fine-tuning phase of our experiments.

[7] Available at: https://scikit-learn.org/stable/

Configuration	**Value**
Optimizer	$AdamW$
Epochs	100
Batch Size per GPU	1
Number of GPUs	16
Weight decay	$1e-5$
Optimizer momentum	$\beta_1, \beta_2 = 0.9, 0.999$
Peak learning rate	$1e-4$
Learning rate schedule	$LinearWarmupCosineAnnealingLR$
Warmup epochs	10
Dropout	0
Rand Spatial Crop Samples Data	$96 \times 96 \times 96$
MONAI Transforms: ScaleIntensityRanged	a_min = -1000 a_max = 1000 b_min = 0 b_max = 1 Clip = True
τ	0.1
ϕ_1	0.1
ϕ_2	0.7
M (Size of masked sequence)	5
P_j (Size of Patches)	1, 2, 4, 8, 16, 32, 96

Table 12: Pre-Training settings for our proposed approach

Algorithm 4 Pseudo Code of Pre-Training Classification Head

```
procedure PREHEAD($X$, $input\_dim$, $x\_dim$, $y\_dim$, $z\_dim$, $cluster\_num$, $class\_size$)    ▷ Input: $X$ is the input tensor.
    $X \leftarrow Conv3d(X, input\_dim, cluster\_num)$
    $X \leftarrow BatchNorm3d(X, cluster\_num).GELU(X)$
    $X \leftarrow Conv3d(X, cluster\_num, cluster\_num)$
    $X \leftarrow BatchNorm3d(X, cluster\_num).GELU(X)$
    $X \leftarrow X.permute(0, 3, 2, 1, 4)$

    $X \leftarrow Conv3d(X, y\_dim, class\_size)$
    $X \leftarrow BatchNorm3d(X, class\_size).GELU(X)$
    $X \leftarrow Conv3d(X, class\_size, class\_size)$
    $X \leftarrow BatchNorm3d(X, class\_size).GELU(X)$
    $X \leftarrow X.permute(0, 2, 1, 3, 4)$

    $X \leftarrow Conv3d(X, x\_dim, x\_dim//16)$
    $X \leftarrow BatchNorm3d(X, x\_dim//16).GELU(X)$
    $X \leftarrow Conv3d(X, x\_dim//16, 1)$
    $X \leftarrow BatchNorm3d(X, 1).GELU(X)$
    $X \leftarrow X.permute(0, 4, 3, 2, 1)$

    $X \leftarrow Conv3d(X, z\_dim, z\_dim)$
    $X \leftarrow BatchNorm3d(X, z\_dim).GELU(X)$
    $X \leftarrow Conv3d(X, z\_dim, z\_dim)$
    $X \leftarrow ReLU(X).squeeze()$
    Return X
end procedure
```

Our experimental setup involved the utilization of 8 nodes, each equipped with dual-GPU chipsets. Each GPU within these chipsets boasted a substantial 35GB of memory, ensuring ample resources for running our experiments efficiently. To adhere to established standards and foster equitable comparisons, we employed a comprehensive array of augmentation techniques to augment data variability. It's noteworthy that these augmentations were uniformly applied to all models, encompassing both our proposed models and the baseline models. This meticulous approach ensures a fair and unbiased comparative analysis.

For the implementation of our models and the baseline models, we leveraged the *MONAI* framework, which provided a robust and versatile foundation for our experimentation. This framework facilitated the seamless integration of our methodologies, streamlining the implementation process and contributing to the reliability of our experimental results. [8] In the course of each iteration, we strategically implemented a randomized cropping strategy, extracting four images for each case during the training phase. This deliberate approach was employed with the intent of diversifying the training dataset for each input case within every epoch, thereby enhancing the overall richness of the training process. The randomness introduced by the cropping strategy contributed to a more robust and varied training experience.

Furthermore, to fine-tune and optimize our training procedure, we incorporated the advanced *AdamW* optimizer. This optimizer played a pivotal role in adjusting the model's weights during training, ensuring an efficient convergence towards optimal performance. The synergistic combination of the cropping strategy and the utilization of the *AdamW* optimizer exemplifies our commitment to refining the training dynamics for improved model efficacy.

[8] Available at: https://monai.io/

To refine our model using labeled data, we employed the DiceCELoss as the objective function during the fine-tuning or training process. The DiceCELoss function serves as a crucial metric, enabling us to strike a balance between the Dice coefficient and Cross-Entropy, optimizing the model's performance on the labeled dataset. This choice of objective function reflects our commitment to a meticulous fine-tuning or training process, ensuring that the model is adept at capturing the nuanced patterns present in the labeled data. The DiceCELoss is articulated by the following formulation:

$$DiceCELoss = w_{dl} \times DiceLoss + w_{cl} \times CELoss, \tag{6}$$

where

$$DiceLoss = 1 - \frac{2 \times \sum_{i=1}^{N} p_i \times t_i + \epsilon}{\sum_{i=1}^{N} p_i + \sum_{i=1}^{N} t_i + \epsilon}, \tag{7}$$

and

$$CELoss = -\frac{1}{N} \sum_{i=1}^{N} t_i \times \log(p_i). \tag{8}$$

Thus, our fine-tuning and training loss term is the weighted summation between the regular dice loss term and the cross entropy term. p_i represents the predicted probability for the i-th class. t_i represents the ground truth label for the i-th class. N represents the number of classes. ϵ is a small constant (e.g., 1e-5) added to the denominator to avoid division by zero.

	w_{dl}	w_{cl}	w_{dl}	w_{cl}	w_{dl}	w_{cl}
LoGoNet	1.0	1.0	0.0	1.0	1.0	0.0
	.854		.841		.847	

Table 13: Performance Outcomes with Varied Weights for DiceCELoss: The presented results represent the average across all 13 organs in the BTCV dataset using the LoGoNet model.

Our experimentation has revealed that assigning equal weights to both CELoss and DiceLoss yields more favorable outcomes, surpassing the performance achieved with other weight ratios. The results of various weight configurations for losses are presented in Table 13. By according equal significance to both Cross-Entropy Loss (CELoss) and Dice Loss, we strike a balance that enhances the model's ability to effectively capture diverse patterns in the data. The results highlight the critical role of assigning equal weights in attaining optimal outcomes and underscore the robustness of this approach within the framework of our experimental setup. This observation emphasizes the significance of maintaining a balanced weighting scheme, showcasing its effectiveness in ensuring stability and reliability across various experimental conditions. The consistent performance across diverse scenarios further validates the efficacy of the equitable weighting strategy adopted in our study.

9 DATASET

This section is dedicated to providing a thorough exposition of the datasets meticulously chosen for inclusion in our experimental analyses. The selection of these datasets is pivotal in influencing the outcomes of our study, serving as the cornerstone for the evaluation and refinement of our methodologies. A detailed exploration of each dataset is undertaken to illuminate its characteristics and significance within the context of our experimental framework.

9.1 Unlabeled Datasets

As unlabeled data, we use a set of 4500 distinct data points republished as a meta-dataset by Tang et al. [40]. Each of these data points represents a volumetric 3D scan specifically focusing on key anatomical areas such as the chest, abdomen, head, and neck. This deliberate selection of regions ensures a thorough examination of the anatomical aspects relevant to our research. This meta-dataset consists of the following dataset:

- **HNSCC** [9]: imaging, radiation therapy, and clinical data of head and neck squamous cell carcinoma (HNSCC) patients at MD Anderson Cancer Center, the dataset comprises 954 samples.
- **LUNA16** [10]: is a publicly available data set and challenge designed to advance the development of computer-aided detection (CAD) algorithms for the accurate identification of pulmonary nodules in low-dose computed tomography (CT) scans, contributing to the improvement of lung cancer detection, and the data set consists of 842 samples.
- **CT Colonography**[11]: part of the National CT Colonography Trial, this dataset comprises 1532 cases of CT colonography imaging accompanied by polyp descriptions and their respective locations within colon segments. It serves as a valuable resource for validating the use of CT colonography in detecting colorectal neoplasia.
- **CT Images in COVID-19** [12]: this collection of medical imaging data includes CT scans and associated clinical information collected during the COVID-19 pandemic. The dataset includes 722 samples.
- **LIDC-IDRI** [13]: is a web-accessible collection of thoracic computed tomography (CT) scans with marked-up annotations of lung nodules, designed for the development and evaluation of computer-assisted diagnostic methods for lung cancer detection, the dataset comprises 450 samples.

9.2 BTCV Dataset

The provided dataset[14] consists of 40 abdominal CT scans that were collected under Institutional Review Board (IRB) supervision. These scans are sourced from a combination of an ongoing colorectal cancer chemotherapy trial and a retrospective ventral hernia study. The CT scans were obtained during the portal venous contrast phase, capturing the anatomical details of the abdominal region. The dataset showcases variability in terms of volume sizes, with the scans ranging from $512 \times 512 \times 85$ to $512 \times 512 \times 198$ voxels. Additionally, the field of view (FOV) varies, spanning approximately $280 \times 280 \times 280, \text{mm}^3$ to $500 \times 500 \times 650, \text{mm}^3$. In-plane resolution fluctuates between $0.54 \times 0.54, \text{mm}^2$ and $0.98 \times 0.98, \text{mm}^2$, while slice thickness varies from 2.5, mm to 5.0, mm. Standard registration data for the dataset has been generated using NiftyReg.

[9] Available at: https://wiki.cancerimagingarchive.net/display/Public/HNSCC
[10] Available at: https://luna16.grand-challenge.org/Data/
[11] Available at: https://www.cancerimagingarchive.net/collections/
[12] TCIA COVID-19 Datasets
[13] Available at: wiki.cancerimagingarchive.net/display/Public/LIDC-IDRI
[14] Available at: https://www.synapse.org/#!Synapse:syn3193805/wiki/217789

Configuration	BTCV	MSD
Optimizer	AdamW	
Epochs	5000	700
Batch Size per GPU	2	1
Number of GPUs	16	
Weight decay	$1e-5$	
Optimizer momentum	$\beta_1, \beta_2 = 0.9, 0.999$	
Peak learning rate	$1e-4$	
Learning rate schedule	LinearWarmupCosineAnnealingLR	
Warmup epochs	100	20
Dropout	0	
Rand Spatial Crop Samples Data	$96 \times 96 \times 96$	
MONAI Transforms: ScaleIntensityRanged	a_min = -175 a_max = 250 b_min = 0 b_max = 1 Clip = True	a_min = -100 a_max = 2000 b_min = 0 b_max = 1 Clip = True
MONAI Transforms: Spacingd	$space\_x$ = 1.5 $space\_y$ = 1.5 $space\_z$ = 2.0	$space\_x$ = 1.0 $space\_y$ = 1.0 $space\_z$ = 1.0
Data Augmentation: RandFlipd	prob for each axis = 0.2	
Data Augmentation: RandRotate90d	prob = 0.2	

Data Augmentation: RandScaleIntensityd	factors = 0.1
	prob = 0.1	

Data Augmentation: RandShiftIntensityd	offsets = 0.1
	prob = 0.1	

Table 14: Training and fine-tune settings for all proposed and baseline models

Task	Colon Cancer	Spleen	Hepatic Vessels	Pancreas Tumour	Lung Tumours	Cardiac
Train	119	35	241	210	51	14
Validation	13	3	26	23	5	2
Test	14	9	61	57	13	4

Table 15: Number of cases for each task of MSD dataset

For the preparation of the data set, we sliced at precise intervals of 1.5 mm in both the x and y directions and 2 mm for the isotropic resolution in the z-direction.

The dataset includes manual segmentations of thirteen abdominal organs. The segmentation was performed on a volumetric basis, providing accurate organ boundaries within the CT scans. The list of segmented organs includes the spleen, right kidney, left kidney, gallbladder, esophagus, liver, stomach, aorta, inferior vena cava, portal vein and splenic vein, pancreas, right adrenal gland, and left adrenal gland. It's important to note that some patients within the dataset may not have a right kidney or gallbladder, and as a result, these organs might not be labeled in those specific cases.

The dataset serves as a valuable resource for advancing automated segmentation approaches in the field of medical imaging. Researchers can utilize this dataset to develop and refine algorithms that accurately identify and segment abdominal organs in CT scans.

9.3 MSD Dataset

This dataset[15] contains ten different tasks. Within this data set, a wide variety of organs and structures are known to encompass vital anatomical regions of significance, such as the heart, liver, prostate, and pancreas.

To use the dataset, we divided it into exact 1.0 mm intervals along the x, y, and z directions, maintaining isotropic resolution for all tasks.

We evaluated our models using six tasks from the MSD dataset. Table 15 displays the case numbers for the various tasks. The initial task focuses on "Colon Cancer Primaries" and poses a challenge due to its varied and irregular appearance, referred to as "heterogeneous" (heterogeneous indicating dissimilar components or elements, causing irregular or variegated appearances; for instance, a dermoid cyst exhibits heterogeneous attenuation on CT scans).

The subsequent task revolves around "Spleen," involving a challenge related to the significant variation in foreground size. The following task targets "Hepatic vessels and tumor," presenting a

[15] Available at: http://medicaldecathlon.com/

Models	Spl	RKid	Lkid	Gall	Eso	Liv	Sto	Aor	IVC	Veins	Pan	Rad	Lad	AVG
UNETR	.912	.940	.938	.693	.690	.954	.754	.891	.830	.703	.734	.660	.577	.790
SegResNetVAE	.941	.938	.933	.670	.718	.955	.745	.892	.848	.695	.783	.633	.528	.791
nnUNet	.859	.944	.924	.796	.755	.960	.781	.894	.849	.756	.776	.675	.663	.818
Attention U-Net	.955	.936	.930	.735	.739	.964	.770	.898	.852	.753	.763	.695	.688	.821
DiNTS Instance	.935	.942	.938	.770	.769	.962	.743	.909	.857	.759	.782	.641	.691	.823
UNet++	.934	.931	.925	.810	.715	.961	.786	.900	.846	.747	.829	.685	.679	.827
SwinUNETR	.952	.947	.945	.790	.770	.963	.755	.901	.850	.771	.760	.702	.659	.828
DiNTS Search	.937	.934	.930	.788	.770	.960	.774	.904	.866	.751	.813	.670	.711	.831
ULKANet	.954	.936	.938	.761	.782	.964	.814	.885	.831	.690	.786	.690	.684	.824
LoGoNet	.958	.949	.947	.818	.786	.969	.880	.912	.865	.769	.821	.726	.698	.854
ULKANet + PRE	.954	.940	.933	.836	.794	.966	.853	.899	.854	.760	.823	.703	.734	.850
LoGoNet + PRE	.961	.947	.944	.866	.845	.970	.898	.936	.885	.791	.838	.738	.757	.875
ULKANet L	.948	.938	.933	.832	.789	.967	.832	.888	.862	.744	.784	.720	.690	.841
LoGoNet L	.960	.944	.922	.847	.781	**.971**	.840	.917	**.872**	.768	.841	.742	.710	.855
ULKANet L + PRE	.960	.943	.935	.869	.863	.970	**.914**	.891	.861	**.806**	**.850**	.734	.761	.874
LoGoNet L + PRE	**.986**	**.963**	**.954**	**.921**	**.859**	.969	.911	**.954**	.871	.805	.846	**.759**	**.784**	**.891**

Table 16: Performance of our model (in terms of Dice metric) compared to the baseline models in BTCV dataset. Spl: Spleen, RKid: Right Kidney, LKid: Left Kidney, Gall: Gallbladder, Eso: Esophagus, Liv: Liver, Sto: Stomach, Aor: Aorta, IVC: Inferior Vena Cava, Veins: Portal and Splenic Venis, Pan: Pancreas, Rad: Right Adrenal Glands, Lad: Left Adrenal Glands

challenge in dealing with small tubular structures adjacent to the heterogeneous tumor. Moving on, we have the "Pancreas Tumor" task, which aims to segment the liver and tumor, posing an unbalanced labeling challenge with large (background), medium (pancreas), and small (tumor) structures.

The "Lung Tumors" task comes next, presenting the challenge of segmenting a small target (cancer) within a large image. Lastly, we have the "Cardiac" task, which focuses on segmenting the Left Atrium and faces the challenge of a small training dataset with significant variability.

10 QUALITATIVE RESULTS

In this section we report additionall qualitative results of our model compared to the baseline models. These experiments we carried out in BTCV dataset. For all the experiments, to accommodate the constraints imposed by the limited RAM capacity of the GPUs employed for training, we implemented a cropping approach to diminish the size of the model. We opted for a cropping approach with dimensions of $96 \times 96 \times 96$. To derive the final results, we employed a sliding window inference methodology, in which the required data were systematically passed with a 0.5 mm overlapping mechanism.

In Tables 16, the results of our proposed model and pre-training approach are compared to the baselines. This analysis provides valuable insights into the effectiveness and performance gains achieved by our proposed methodology when compared against existing baseline methods.

The data presented in Table 17 illustrates the superiority of our innovative multi-task self-supervised learning strategy when compared to other baseline methods, as evidenced by the aggregated results on the BTCV dataset. Notably, our proposed approach showcases exceptional efficacy across three out of four experiments, highlighting its versatility and effectiveness across various tasks.

In Figure 5, the comprehensive results of our proposed model are presented alongside the outcomes of various baselines. Our model, equipped with a combined local and global attention mechanism, showcases superior performance in segmenting various body regions, including even the smaller anatomical parts. This enhanced segmentation capability is attributed to the ability of our model to effectively capture both local and long-range dependencies within the data, making it particularly proficient in delineating intricate structures within medical images. This mechanism provides a substantial advantage over the baselines, enabling our model to deliver more accurate and detailed segmentations across a wide range of body regions, which is essential for medical image analysis and diagnosis. The combination of global and local scales in our model, as well as its emphasis on meaningful feature extraction, makes it a powerful tool for improving the accuracy and precision of medical image segmentation tasks, particularly in scenarios where small anatomical details are crucial for accurate diagnosis.

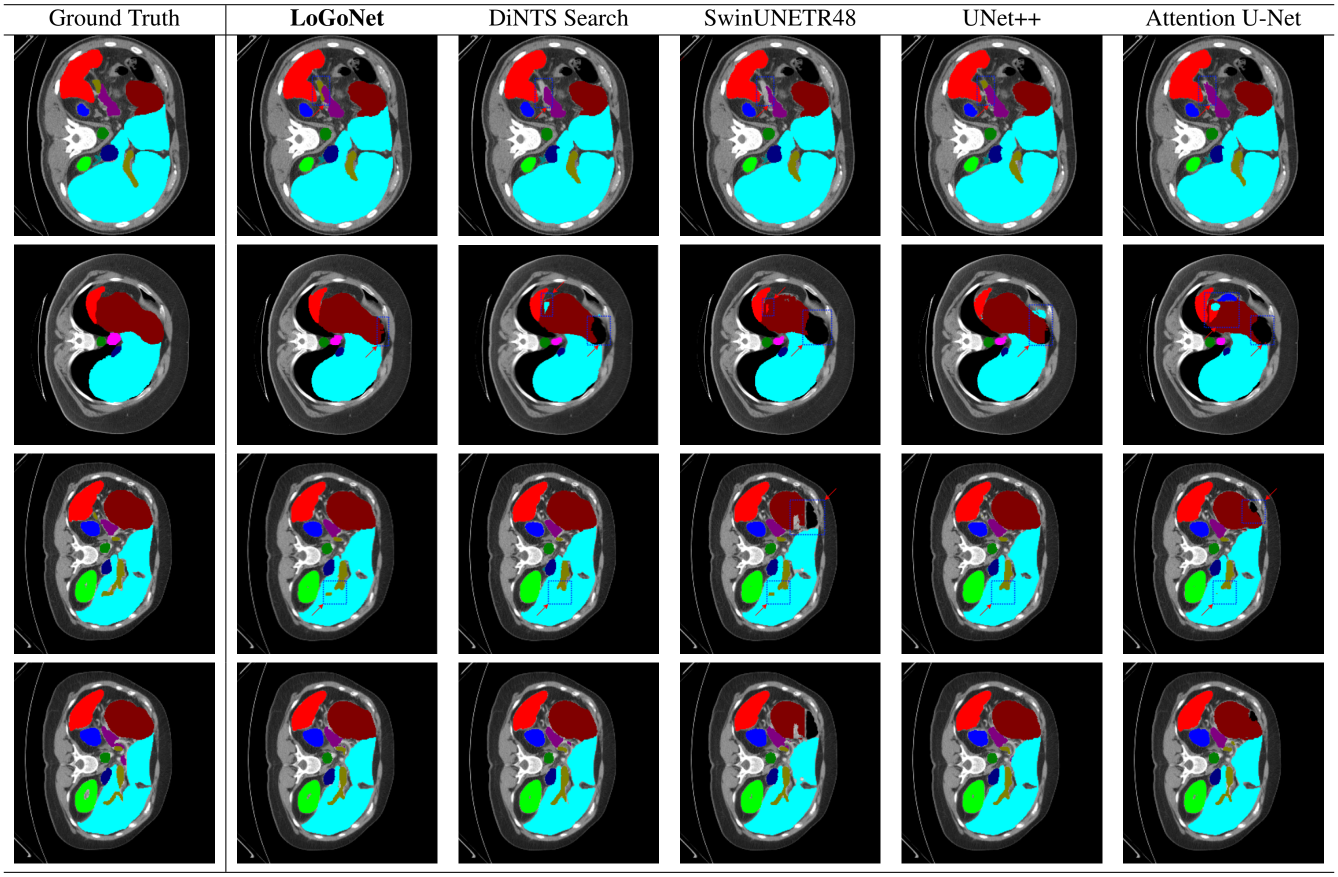


Figure 5: A visualization of LoGoNet outputs compared to the output of the baseline models. Our model outperforms the baselines, particularly in the segmentation of small organ sections.

SSL Approach	Spl	RKid	Lkid	Gall	Eso	Liv	Sto	Aor	IVC	Veins	Pan	Rad	Lad	AVG
SimMIM [46]	**.961**	**.949**	.940	.837	.829	.970	.879	.928	.880	.785	.827	.726	.733	.864
Rubik's Cube [41]	**.961**	.940	**.946**	.815	.820	**.971**	.878	.922	.872	.780	.825	.720	.725	.859
SimCLR [10]	.960	.944	.942	.829	.803	.969	.882	.920	.870	.780	.829	.726	.720	.859
Our SSL Approach	**.961**	.947	.944	**.866**	**.845**	.970	**.898**	**.936**	**.885**	**.791**	**.838**	**.738**	**.757**	**.875**

Table 17: Complete Result of Performance Comparison of Our Proposed Multi-Task Self-Supervised Learning Approaches. (Number of Clusterer is N=80)