Searching for BurgerFormer with Micro-Meso-Macro Space Design

Longxing Yang [1 2 3] **Yu Hu** [1 2 3] **Shun Lu** [1 2 3] **Zihao Sun** [1 2 3] **Jilin Mei** [1 2 3] **Yinhe Han** [1 2 3] **Xiaowei Li** [2 3]

Abstract

With the success of Transformers in the computer vision field, the automated design of vision Transformers has attracted significant attention. Recently, MetaFormer found that simple average pooling can achieve impressive performance, which naturally raises the question of how to design a search space to search diverse and high-performance Transformer-like architectures. By revisiting typical search spaces, we design micro-meso-macro space to search for Transformer-like architectures, namely BurgerFormer. Micro, meso, and macro correspond to the granularity levels of operation, block and stage, respectively. At the microscopic level, we enrich the atomic operations to include various normalizations, activation functions, and basic operations (e.g., multi-head self attention, average pooling). At the mesoscopic level, a hamburger structure is searched out as the basic BurgerFormer block. At the macroscopic level, we search for the depth, width, and expansion ratio of the network based on the multi-stage architecture. Meanwhile, we propose a hybrid sampling method for effectively training the supernet. Experimental results demonstrate that the searched BurgerFormer architectures achieve comparable even superior performance compared with current state-of-the-art Transformers on the ImageNet and COCO datasets. The codes can be available at https://github.com/xingxing-123/BurgerFormer.

[1]Research Center for Intelligent Computing Systems, Institute of Computing Technology, University of Chinese Academy of Sciences, Beijing, China [2]State Key Laboratory of Computer Architecture, Institute of Computing Technology, University of Chinese Academy of Sciences, Beijing, China [3]School of Computer Science and Technology, University of Chinese Academy of Sciences, Beijing, China. Correspondence to: Yu Hu <huyu@ict.ac.cn>.

Proceedings of the 39^{th} International Conference on Machine Learning, Baltimore, Maryland, USA, PMLR 162, 2022.

1. Introduction

Since the emergence of landmark works such as Vision Transformer (ViT) (Dosovitskiy et al., 2021), DeiT (Touvron et al., 2021b) and Swin (Liu et al., 2021), Transformers have made encouraging performance advances in the Computer Vision (CV) field such as recognition, detection, and segmentation. As a result, combining Transformer and Neural Architecture Search (NAS) (Chen et al., 2021b;a) has also gained increasing attention. Recently, MetaFormer (Yu et al., 2022) leverages simple average pooling to achieve impressive performance, which naturally raises the question of how to design a search space to search high-performance Transformer-like architectures.

Recalling the development of NAS, the design of search space plays a key role. In the early stage, the NAS-RL (Zoph & Le, 2017) search space is proposed to find out optimal hyper-parameters of the entire network. Afterwards, NASNet (Zoph et al., 2018) proposes a search space containing 13 operations for searching cells and then stacks cells into a network. The search space is further compacted to 8 operations in DARTS (Liu et al., 2019). ProxylessNAS (Cai et al., 2019) and FBNet (Wu et al., 2019) search spaces borrow from the block design of MobileNet (Sandler et al., 2018), which means the search spaces contain various MBConv blocks. Similarly, in Transformer NAS, the search space of AutoFormer (Chen et al., 2021b) is borrowed from the block design of Transformer.

As shown in Fig. 1, we classify the typical search space design according to the granularity of operations, blocks, and stages from the micro, meso, and macro perspectives. In the NASNet and DARTS search spaces, diverse and rich atomic operations belong to micro design. The ProxylessNAS, FBNet, AutoFormer, and ViT-ResNAS (Liao et al., 2021) search spaces contain MobileNet or Transformer blocks which belong to meso design. In the RegNet (Ilija et al., 2020), FBNet, AutoFormer and ViT-ResNAS search spaces, the search for hyper-parameters such as depth and width of the stage belongs to macro design.

In this paper, we aim to design a search space at the micro-meso-macro level to search for more efficient Transformer-like architectures. At the microscopic level, we use richer atomic operations, including various normalizations, activation functions, and basic operations (e.g., multi-head self

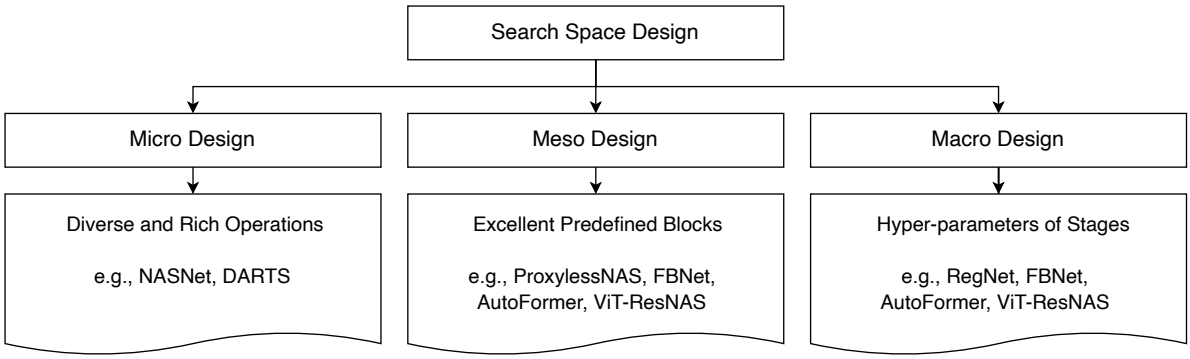


Figure 1. A taxonomy of search space designs. Micro, Meso, and Macro correspond to operation, block, and stage granularity, respectively.

attention, average pooling). At the mesoscopic level, we design a combined Norm-Op-Norm-Act operation structure and propose the hamburger structure to search for diverse Transformer-like blocks. At the macroscopic level, we search for the depth, width, and expansion ratio based on the multi-stage architecture.

Our major contributions are as follows:

1. We propose a micro-meso-macro search space for the first time and propose a Norm-Op-Norm-Act and Hamburger structure to offer more diverse operations and block design. Based on the proposed search space, high-performance Transformer-like architecture BurgerFormer can be searched.

2. We utilized One-Shot NAS for searching and propose a hybrid sampling method to effectively train the supernet.

3. Experiments on the ImageNet and COCO datasets show that the searched BurgerFormers match even outperform state-of-the-art Transformers.

2. Related Work

Neural Architecture Search. NAS is designed to automatically search for efficient neural architectures to reduce the cost of manual trial and error. Early NAS utilized reinforcement learning (Zoph & Le, 2017) and evolutionary algorithms (Real et al., 2019) to search for convolutional neural network and recurrent neural network. These methods had an immense resource overhead because hundreds of individual networks needed to be trained from scratch, resulting in thousands of GPU days even on small-scale datasets. Subsequently, ENAS (Pham et al., 2018) proposed the weight sharing strategy, which reduced the search time to a few GPU days as only one supernet needs to be trained. Based on the strategy, differentiable NAS methods (Liu et al., 2019; Chu et al., 2020; Xu et al., 2020; Chen et al., 2019) make the search space continuous and adopt architecture parameters to select operations. Single-Path methods (Guo et al., 2020; Chu et al., 2021; Yu et al., 2020) further reduce the resource consumption by sampling and training single-path sub-networks from the supernet, and then searching sub-networks based on validation set accuracy.

Vision Transformer. Dosovitskiy et al (Dosovitskiy et al., 2021) introduced Transformer to the computer vision field, which achieved competitive performance compared to Convolutional Neural Networks (CNNs). Subsequently, DeiT (Touvron et al., 2021b) proposed data-efficient ViT and a teacher-student distillation strategy. Swin-Transformer (Liu et al., 2021) and PVT (Wang et al., 2021) used multi-stage strategy to successfully apply Transformer to different vision tasks such as detection and segmentation. TNT (Han et al., 2021) proposed that attention to finer-grained patches can effectively improve performance. ConViT (d'Ascoli et al., 2021) introduced a soft inductive bias of CNN into ViT to bridge the gap between CNN and Transformer. CeiT (Yuan et al., 2021) combined the benefits of CNN in extracting low-level features and locality with the advantage of Transformer in extracting long-range dependencies. MLP-Mixer (Tolstikhin et al., 2021) and ResMLP (Touvron et al., 2021a) used MLP instead of the self-attention module, still achieving comparable results. Moreover, MetaFormer (Yu et al., 2022) discovered that using the average pooling operation instead of the self-attention module also achieved promising performance.

Vision Transformer NAS. AutoFormer (Chen et al., 2021b) is the first algorithm to automatically search ViT for embedding dimension, depth, MLP ratio, and the number of heads. ViTAS (Su et al., 2021) searched ViT models based on the weight-sharing mechanism. GLiT (Chen et al., 2021a) introduced locality modules into the search space, proposed a hierarchical search strategy, and searched ViT from both global and local levels. ViT-ResNAS (Liao et al., 2021) proposed residual spatial reduction and multi-architectural sampling techniques for searching a multi-stage ViT architecture. (Minghao et al., 2021) proposed S3 space based on ViT and Swin, and leveraged weight sharing and E-T

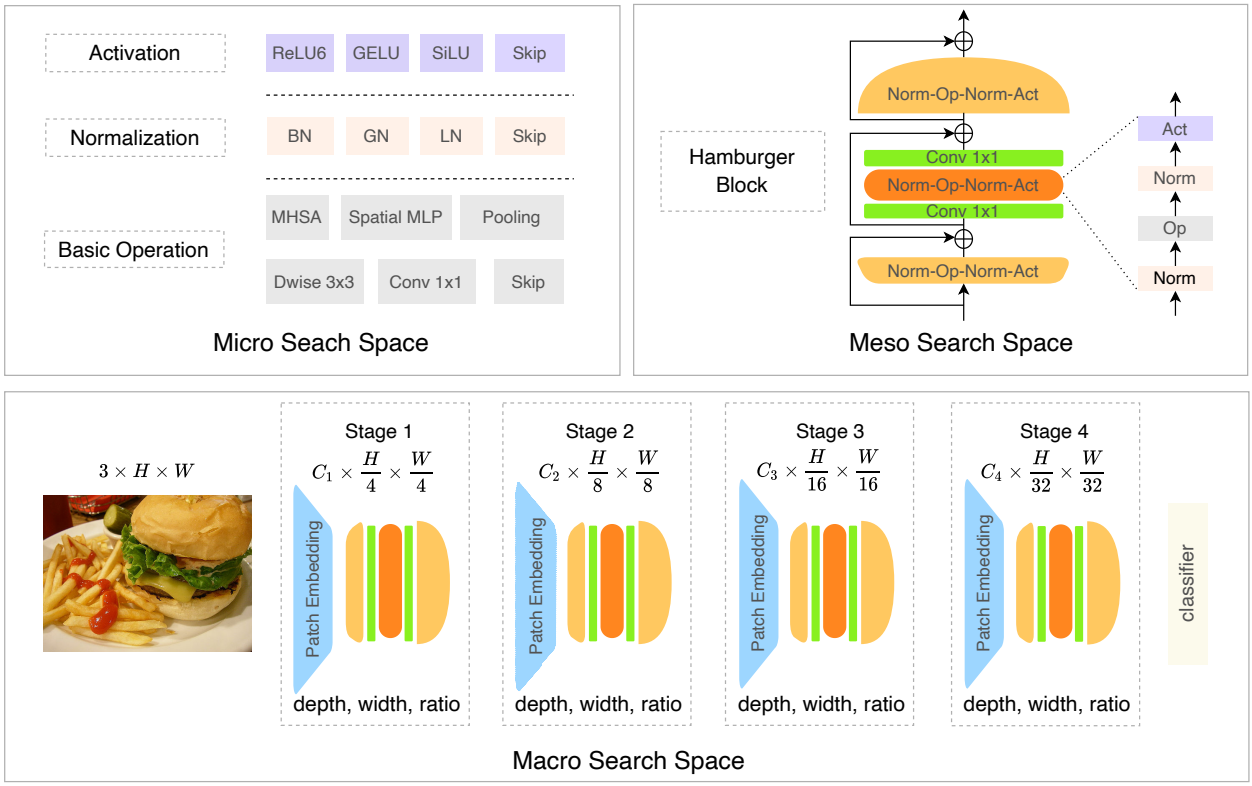


Figure 2. The micro-meso-macro search space. The top left is the micro search space, which contains candidate activation functions, normalizations and basic operations. Skip indicates no data processing. The top right is the meso search space, which includes the Hamburger and the Norm-Op-Norm-Act structure. Norm, Op, and Act represent normalization, basic operation, and activation function, respectively. The lower part is the macro search space, which produces a multi-stage architecture.

Error to search the architecture and search space. NASViT (Chengyue et al., 2022) designed a hybrid space based on MBConv and Transformer blocks and developed ViTs from 200M to 800M FLOPs. These works focus on the design of the block or stage granularity, lacking the diversity of block structures and a comprehensive consideration of all three granularities.

3. Search Space Design

In this section, we will first introduce the micro-meso-macro search space design of BurgerFormer in detail. Then we will briefly discuss the size of the proposed search space.

3.1. Micro Search Space Design

The micro search space corresponds to the operation granularity. Rich candidate operations facilitate the search for different mesoscopic structures. For example, eight operations can build about 10^9 normal or reduction cells in the DARTS search space. To improve the richness of operations, our micro search space contains varied normalizations, activation functions, and basic operations.

Normalization and Activation Function. Normalization and activation are two types of atomic operations. Normalization can accelerate training and improve the generalization, while activation functions can enhance the nonlinear fitting ability of the network. Proper selection of the two types of atomic operations can improve performance. Previous NAS methods directly utilize fixed combinations of normalization and activation, e.g., Conv-BN-ReLU. However, existing Transformers and CNNs already employ different normalization and activation functions, which inspires us to search varied normalization and activation functions. As shown in Fig. 2 , our micro search space includes layer normalization (LN) (Ba et al., 2016), batch normalization (BN) (Ioffe & Szegedy, 2015), group normalization (GN) (Wu & He, 2018), activation functions of ReLU6 (Agarap, 2018), GELU (Hendrycks & Gimpel, 2016), and SiLU (Ramachandran et al., 2018).

Basic Operation. Basic operations refer to micro operations other than normalizations and activation functions. Following Transformer-like architectures, the basic operations of BurgerFormer contain Multi-Head Self Attention (MHSA) (Dosovitskiy et al., 2021), Spatial MLP (Touvron et al., 2021a), and Pooling (Yu et al., 2022), where the pool-

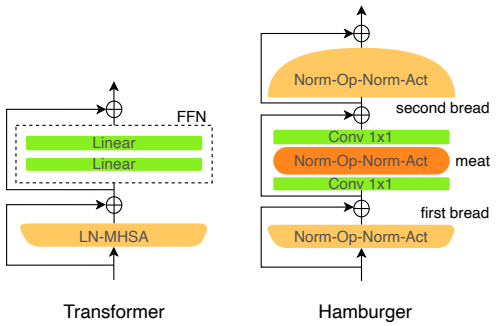


Figure 3. Transformer block vs. Hamburger block.

ing size is 3. In addition, we extend the 3x3 depthwise convolution (Dwise 3x3) and 1x1 convolution (Conv 1x1) to introduce more locality to the network. Specifically, MHSA, Spatial MLP, and Pooling are formulated as follows:

$$MHSA(Y) = Concat(head_1, \dots, head_h)W^o,$$
$$head_i = Softmax(\frac{(YW_i^Q)(YW_i^K)^T}{\sqrt{d_i}})YW_i^V, \quad (1)$$

$$Spatial\ MLP(Y) = (Y^T W)^T + b, \quad (2)$$

$$Pooling(X_{i,j,k}) = \frac{1}{9}\sum_{p,q=1}^{3} X_{i,j+p-2,k+q-2} - X_{i,j,k}, \quad (3)$$

where $W^Q, W^K, W^V, W^o \in R^{c\times c}$, and $W \in R^{wh\times wh}$ are the weights, $X \in R^{c\times w\times h}$ are 2D feature maps, and $Y \in R^{n\times d}$ are 1D tokens, respectively. c, w, h, n, d represent channel, width, height, number of tokens, and embedding dimension, respectively. Following DeiT (Touvron et al., 2021b) source code, the token dimension in each head is $R^{(c/head)}$, so $W^o \in R^{head*(c/head)\times c} = R^{c\times c}$. Please note that the Pooling operator is from MetaFormer, where the subtraction is only used as a preprocessing method for the subsequent residual connection. In practice, pooling and convolutions require 2D feature maps, while MHSA and Spatial MLP require 1D tokens, so two data formats need to be converted. For the conversion of a 2D feature map to a 1D token, we first flatten the spatial dimension of $X \in R^{c\times w\times h}$ to obtain $X \in R^{c\times(wh)}$, then transpose X to get $Y \in R^{(wh)\times c} = R^{n\times d}$. Conversion of a 1D token to a 2D feature map is an inverse process.

3.2. Meso Search Space Design

Good blocks in the meso search space are halfway to successfully search for high-performance macro architecture. As shown in Fig. 3, we design the hamburger block based on three modifications to the classical Transformer block. Firstly, the combination of LN and MHSA is abstracted as a Norm-Op-Norm-Act structure for search, where Norm, Act, and Op represent normalization, activation function, and basic operation, respectively. Please note that both the pre-Op normalization and the post-Op normalization are determined by searching, so the two normalizations can be different. Secondly, on the one hand, adding convolution to feed-forward network (FFN) can improve performance (Yuan et al., 2021), on the other hand, we observe that both of the MobileNet and Transformer blocks have inverse bottleneck structures. Therefore, we split the FFN into two 1x1 convolutions and add Norm-Op-Norm-Act between them. Finally, we symmetrically add Norm-Op-Norm-Act before and after the FFN to search for more diverse meso blocks. Since the block structure resembles a hamburger, we call it hamburger block. Meanwhile, we vividly call the three Norm-Op-Norm-Act structures from input to output as the first bread, the meat, and the second bread.

Stage	Depth	Width	Ratio
1	$\{1,2,3,4\}$	$\{32,64,96\}$	$\{1,2,4,6\}$
2	$\{1,2,3,4\}$	$\{64,96,128\}$	$\{1,2,4,6\}$
3	$\{1,2,3,4,5,6,7,8\}$	$\{128,192,256,320\}$	$\{1,2,4,6\}$
4	$\{1,2,3,4\}$	$\{128,256,384,512\}$	$\{1,2,4,6\}$

Table 1. Macro Search Space. Depth is the number of blocks per stage. Width refers to the channels of 2D feature maps or the embedding dimension of 1D tokens. Ratio is the expansion ratio between the two 1x1 convolutions.

3.3. Macro Search Space Design

The macro search space corresponds to the design of the stage granularity. Referring from PVT (Wang et al., 2021) and Swin (Liu et al., 2021), we design the multi-stage search space for searching optimal depth, width, and expansion ratio, which mean the number of hamburger blocks, the channels of 2D feature map or the embedding dimension of 1D tokens, and the expansion ratio between the two 1x1 convolutions, respectively. The search ranges of these hyper-parameters are shown in Table 1.

3.4. Search Space Size

In order to reduce resource consumption, we have adjusted the search space as follows. First, we did not search MHSA and Spatial MLP in the first and the second stages, because the complexity of these two operations is quadratic to hw, which is not suitable for use in large resolution. Second, the basic operations in the meat part of the burger contain only skip and 3x3 depthwise convolution. Third, we search for hamburger blocks for each stage. There are about 4.5×10^{28} architectures in the search space. The range of parameters is from $0.5M$ to $41.6M$, and the range of FLOPs is from $0.2G$ to $7.4G$.

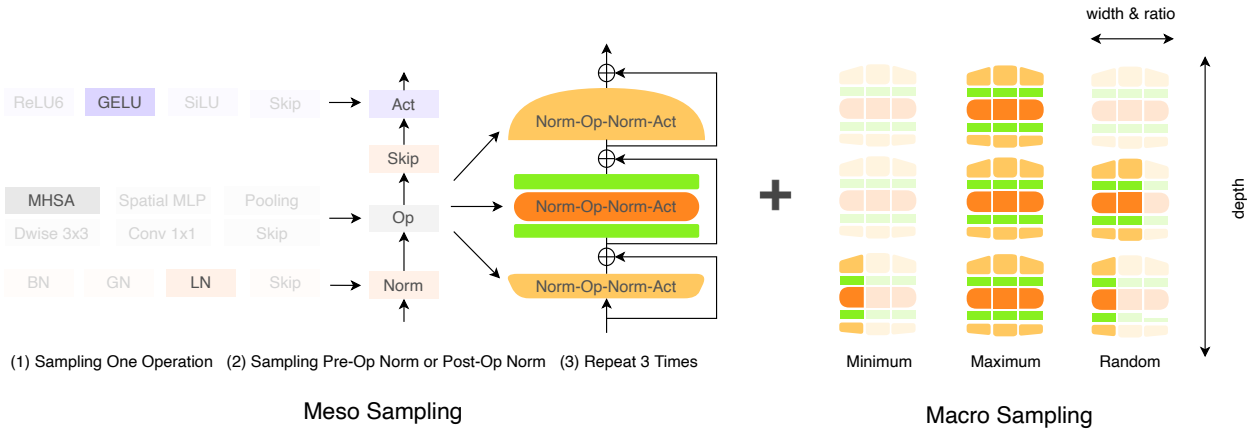


Figure 4. Hybrid sampling method. Meso sampling is sampling one hamburger block for each stage. Macro sampling is sampling the minimum, maximum, and random sub-networks in the macro search space, whose blocks are determined by the meso sampling. The supernet weights are updated with the accumulated gradients from the sampled three sub-networks.

4. One-Shot NAS

One-Shot NAS is a search strategy based on a weight sharing mechanism. The strategy only needs to train one supernet N and then evaluates the accuracy of the sub-networks A that inherit supernet weights, which greatly reduces the search cost compared with training thousands of stand-alone networks. For keeping consistency and reducing memory usage, the supernet is usually optimized by uniformly sampling the sub-networks:

$$\begin{aligned} W_A^* &= \arg\min_A L_{train}(N(A, W)), \\ \alpha^* &= \arg\max_{a \in A} Acc_{val}(N(\alpha, W^*)), \\ s.t.\ &Resource(N(\alpha, W_{\alpha^*})) \leq C, \end{aligned} \tag{4}$$

where L_{train} is the training loss, Acc_{val} is the validation accuracy, and C is the resource constriant (e.g., FLOPs).

4.1. Supernet Training with Hybrid Sampling

In One-Shot NAS, it is common practice to sample a sub-network (SPOS) (Guo et al., 2020) in each iteration and train the supernet. However, SPOS suffers from low-accuracy sub-networks in our search space. Sandwich sampling (Yu et al., 2020) can improve the accuracy of sub-networks, but we observe that the maximum model in the search space has poor performance and slow convergence speed. The convergence ability of the maximum model is usually worse than the minimum model, so it is not suitable to use the sandwich sampling directly. More details can be referred to Appendix A. In order to better train the supernet, we propose a hybrid sampling method that combines SPOS and sandwich sampling.

As shown in Fig. 4, hybrid sampling consists of two steps, i.e., meso sampling and macro sampling. In the meso sampling step, for each Norm-Op-Norm-Act structure, we first randomly select normalization, activation, and basic operation from the candidates, then randomly select pre-Op normalization and post-Op normalization. Note that normalization and activation will be set to skip when the basic operation is skip, and the residual connections next to the Norm-Op-Norm-Act structure will be removed in first and second bread. After three independent samplings, a specific hamburger block is determined per stage. In the macro sampling step, we randomly select one hyper-parameter from the candidate depth, width, and expansion ratio for each stage. In order to better train the supernet, we additionally sample the maximum macro architecture and the minimum macro architecture of each stage after confirming the hamburger block. The three sampled paths are trained separately, and then the gradients are accumulated to update the supernet weights.

4.2. Evolutionary Search

After training supernet, we utilize an evolutionary algorithm to search sub-networks by evaluating performance with the inherited weights from the supernet. During evolutionary iteration, we maintain the population of sub-networks. In the first iteration, P_0 sub-networks are randomly selected. In subsequent iterations, parent sub-networks are randomly selected from the sub-network population to generate child sub-networks by crossover and mutation. Crossover is the exchange of architecture parameters with a certain probability P_c, and mutation randomly changes the architecture parameters with a certain probability P_m. During search, sub-networks that satisfy the resource constraints will be selected. The whole process will be repeated T iterations.

Model	Params. (M)	FLOPs (G)	Top-1 acc. (%)	Top-5 acc. (%)	Design Type
DeiT-Ti (Touvron et al., 2021b)	6	1.3	72.2	91.1	Manual
TNT-Ti (Han et al., 2021)	6	1.4	73.9	-	Manual
CeiT-T (Yuan et al., 2021)	6	1.2	76.4	-	Manual
AutoFormer-tiny (Chen et al., 2021b)	6	1.3	74.7	92.6	Auto
GLiT-Tiny (Chen et al., 2021a)	7	1.4	76.3	91.1	Auto
ViTAS-DeiT-A (Su et al., 2021)	7	1.4	75.6	92.5	Auto
BurgerFormer-Tiny	10	1.0	78.0	93.7	Auto
ConVi-Ti+ (d'Ascoli et al., 2021)	10	2.0	76.7	93.6	Manual
PVT-Tiny (Wang et al., 2021)	13	1.9	75.1	-	Manual
PoolFormer-S12 (Yu et al., 2022)	12	2.0	77.2		Manual
BurgerFormer-Small	14	2.1	80.4	95.0	Auto
Deit-S (Touvron et al., 2021b)	22	4.7	79.9	-	Manual
Swin-T (Liu et al., 2021)	29	4.5	81.3	-	Manual
CvT-13 (Haiping et al., 2021)	20	4.5	81.6	-	Manual
TNT-S (Han et al., 2021)	24	5.2	81.5	95.7	Manual
PVT-Small (Wang et al., 2021)	25	3.8	79.8	-	Manual
ViL-Small (Pengchuan et al., 2021)	25	4.9	82.0	-	Manual
ResMLP-S12 (Touvron et al., 2021a)	31	6.0	79.4	-	Manual
Twins-PCPVT-S (Xiangxiang et al., 2021)	24	3.8	81.2	-	Manual
PoolFormer-S36 (Yu et al., 2022)	31	5.2	81.4	-	Manual
RegNetY-4G (Ilija et al., 2020)	21	4.0	80.0	-	Auto
AutoFormer-small (Chen et al., 2021b)	23	5.1	81.7	-	Auto
GLiT-Small (Chen et al., 2021a)	25	4.4	80.5	-	Auto
ViTAS-DeiT-B (Su et al., 2021)	23	4.9	80.2	95.1	Auto
S3-T (Minghao et al., 2021)	28	4.7	82.1	95.8	Auto
ViT-ResNAS-Medium (Liao et al., 2021)	97	4.5	82.4	-	Auto
BurgerFormer-Base	26	3.9	82.7	96.2	Auto
Swin-S (Liu et al., 2021)	50	8.7	83.0	-	Manual
Twins-PCPVT-B (Xiangxiang et al., 2021)	44	6.4	82.7	-	Manual
CvT-21 (Haiping et al., 2021)	32	7.1	82.5	-	Manual
BoTNet-S1-59 (A. et al., 2021)	34	7.3	81.7	-	Manual
RegNetY-8G (Ilija et al., 2020)	39	8.0	81.7	-	Auto
BossNet-T1 (Changlin et al., 2021)	-	7.9	82.2	95.8	Auto
BurgerFormer-Large	36	6.5	83.0	96.8	Auto

Table 2. Comparison with state-of-the-art models on ImageNet. "Param." is the volume of parameters. "acc." is accuracy. The gray color highlights the searched BurgerFormer architectures. We group different models according to their FLOPs.

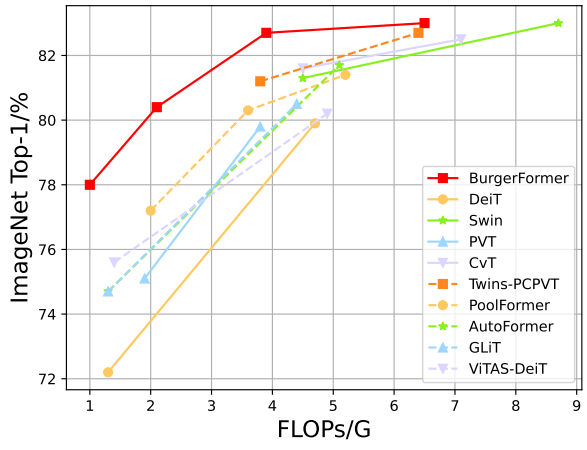


Figure 5. Comparison between BurgerFormer and efficient vision transformers, in terms of ImageNet Top-1 over FLOPs.

5. Experiments

In this section, we first show the searched architectures and compare them with state-of-the-art transformers. In addition, we transfer the searched Burgerformer to the COCO dataset to verify its transferability. We then conduct an ablation study of search methods and hamburger blocks. Finally, the searched architectures and visualization are discussed.

5.1. Results on ImageNet

Searching Settings. ImageNet (Olga et al., 2015) contains $1.28M$ training images and $50,000$ validation images. We split $25,000$ images from the training set as the validation set for searching. In the supernet training phase, we trained the supernet using AdamW (Ilya & Frank, 2019) optimizer with learning rate $1e-3$ and weight decay 0.05. we turned off the normalization statistics because of varying sampled architectures. The data augmentation and other

BackBone	RetinaNet 1x						
	Param. (M)	AP^b	AP^b_{50}	AP^b_{75}	AP_S	AP_M	AP_L
ResNet50 (He et al., 2016)	37.7	36.3	55.3	38.6	19.3	40.0	48.8
PVT-Small (Wang et al., 2021)	34.2	40.4	61.3	43.0	25.0	42.9	55.7
PoolFormer-S24 (Yu et al., 2022)	31.1	38.9	59.7	41.3	23.3	42.1	51.8
BurgerFormer-Base	35.9	41.2	61.3	43.9	24.2	44.4	55.4
BackBone	**Mask R-CNN 1x**						
	Param. (M)	AP^b	AP^b_{50}	AP^b_{75}	AP^M	AP^M_{50}	AP^M_{75}
ResNet50 (He et al., 2016)	44.2	38.0	58.6	41.4	34.4	55.1	36.7
PVT-Small (Wang et al., 2021)	44.1	40.4	62.9	43.8	37.8	60.1	40.3
PoolFormer-S24 (Yu et al., 2022)	41.0	40.1	62.2	43.4	37.0	59.1	36.9
Swin-T (Liu et al., 2021)	48.0	43.7	66.6	47.7	39.8	63.3	42.7
BurgerFormer-Base	45.9	44.0	65.6	48.1	40.2	62.7	43.4

Table 3. Comparison with state-of-the-art models on COCO.

techniques are essentially the same as for retraining, except that stochastic depth is not used. The epochs are 120 and the warmup epochs are 10. Experiments are performed on eight V100s with a batch size of 32 per GPU. In the searching phase, our settings follow (Liao et al., 2021). The initial population size P_0 is 500 and the number of iterations T is 20, The number of parents and child networks are 75 and 150, respectively. The crossover and mutation probability is 0.3. We select the Top-5 architectures and validate them on ImageNet-100 (Yonglong et al., 2020), and then take the optimal architecture and retrain it on ImageNet.

Retraining Settings. Our implementations follow DeiT (Touvron et al., 2021b) and MetaFormer (Yu et al., 2022). Models are optimized using AdamW with learning rate $1e-3$ and weight decay 0.05 and batch size $1,024$. Data augmentations include MixUp (Hongyi et al., 2018), CutMix (Sangdoo et al., 2019), CutOut (Zhun et al., 2020) and RandAugment (Ekin Dogus et al., 2020). We alse use stochasic depth (Gao et al., 2016) and layerscale (Hugo et al., 2021). Label Smoothing (Szegedy et al., 2016) is set to 0.1. The training epochs are 300 and the warmup epochs are 10. Retraining is also conducted on eight V100s.

Results. The experimental results are shown in Table 2. We searched BurgerFormer under four different levels of FLOPs and compared it with state-of-the-art Transformers. BurgerFormers are competitive even superior to either manually designed or automatically searched Transformers under similar FLOPs. For example, BurgerFormer-Tiny has only $1.0G$ FLOPs, but the accuracy is higher than the manually designed DeiT-T and the automatically searched AutoFormer-Tiny by 5.8% and 3.3%. Meanwhile, BurgerFormer-Base outperforms Swin-T by 1.4% points with even fewer FLOPs ($3.9G$ vs. $4.5G$) and parameters ($26M$ vs. $29M$). In addition, although BurgerFormer-Large has the same Top-1 accuracy as Swin-S, it has 25% and 28% fewer FLOPs and parameters, respectively. BurgerFormer-Large also outperforms the pure CNN model RegNetY-8G and the hybrid CNN-Transformer model BossNAS-T1. Furthermore, BurgerFormer consistently outperforms the automatically searched Transformer-like model, indicating the effectiveness of our micro-meso-macro designed search space.

5.2. Results on COCO

Setting. We conduct experiments on COCO benchmark (Lin et al., 2014) to verify the transferability of BurgerFormer architectures. COCO benchmark contains 118K training images ($train2017$) and 5K validation images ($val2017$). We employ BurgerFormer as backbone for two standard detectors, i.e., RetinaNet (TsungYi et al., 2017) and Mask R-CNN (Kaiming et al., 2017). The backbone is initialized with the weights pre-trained on ImageNet, while the added layers are initialized by Xavier (Xavier & Yoshua, 2010). The detectors are trained with AdamW (Ilya & Frank, 2019) optimizer and an initial learning rate of $1e{-}4$. Following (TsungYi et al., 2017; Kaiming et al., 2017), we utilized $1\times$ training schedule where the detectors are trained with 12 epochs. The training images are resized to 800 pixels (short side) by no more than 1,333 pixels (long side). The testing images have a short side of 800 pixels.

Results. As shown in Table 3, under a similar volume of parameters, BurgerFormer significantly outperforms ResNet50 and PoolFormer-S24. When compared with powerful handcrafted baselines such as PVT-Small and Swin-T, BurgerFormer achieves competitive results, which validates the transferability of the searched model.

5.3. Ablation Study

Search Method. In Table 4, we compared random search, SPOS, Sandwich, and our hybrid sampling search method on ImageNet. For a fair comparison, all models are constrained to $1G$ FLOPs and are trained under the same settings. The experimental results demonstrate that our hybrid sampling search method outperforms random search,

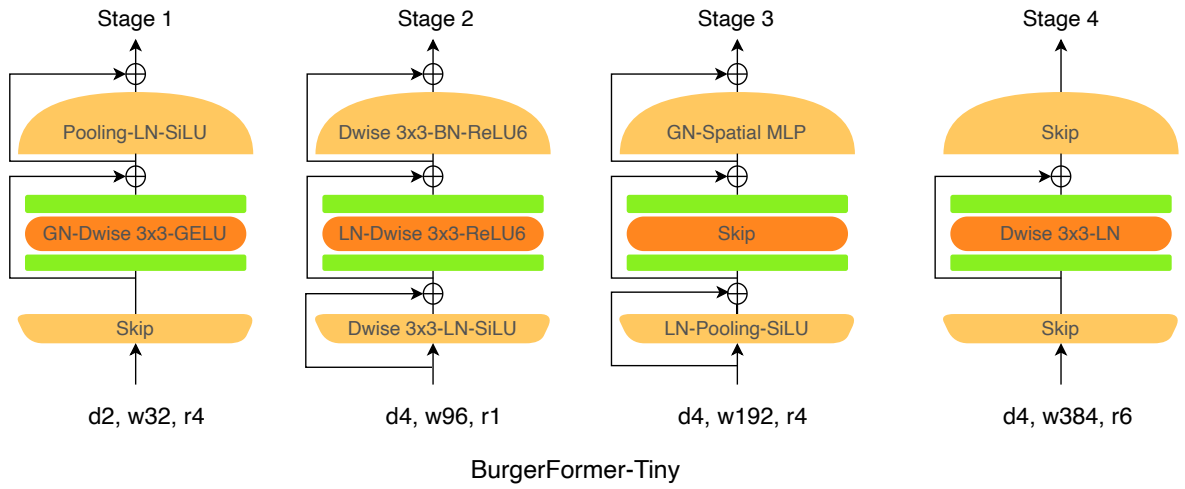


Figure 6. The architecture of BurgerFormer-Tiny. "d2, w32, r4" means that depth, width, and expansion ratio are 2, 32, and 4, respectively.

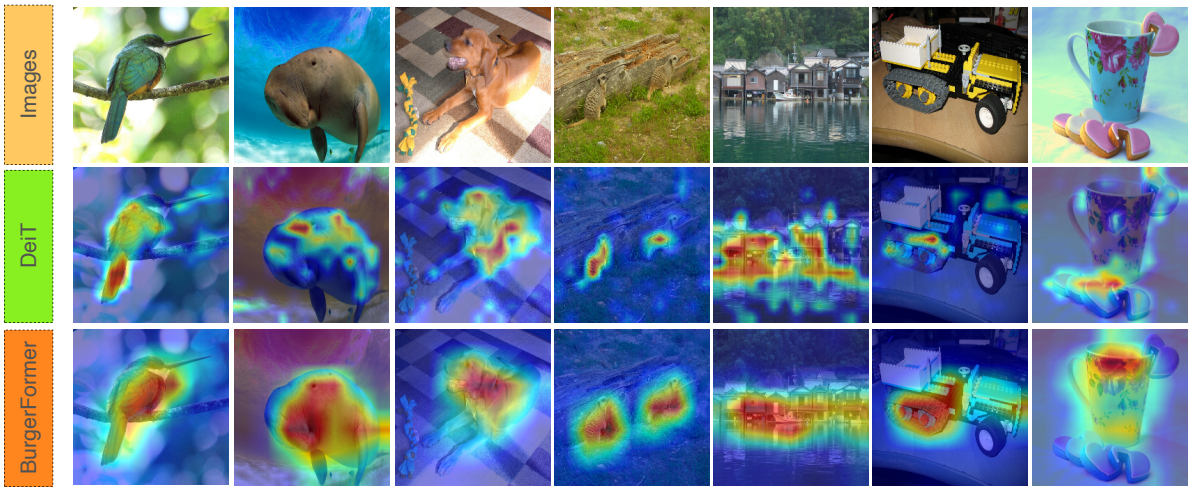


Figure 7. Visualization of features for Deit and BurgerFormer on ImageNet.

Methods	FLOPs (G)	Top-1 acc. (%)
Random Search	1.0	75.6
SPOS	1.0	77.0
Sandwich	1.0	77.5
Hybrid Sampling	1.0	78.0

Table 4. Comparisons of Random Search, SPOS, Sandwich and our hybrid sampling search method on ImageNet.

Model	FLOPs (G)	Top-1 acc. (%)
BurgerFormer-Tiny	1.0	78.0
w/o the first bread	0.85	77.7
w/o the second bread	0.82	77.2
w/o the meat	0.84	76.9

Table 5. Ablation study of Hamburger blocks.

SPOS and sandwich by 2.4 points, 1.0 points and 0.5 points, respectively, validating the effectiveness of the proposed method.

Hamburger Blocks. Table 5 analyzes the effect of each Norm-OP-Norm-Act in hamburger blocks. We choose the Transformer-Tiny model as the baseline. We remove the part corresponding to burger, i.e., replace the part with skip and remove the residual connections, to check its impact on performance. Table 5 shows that each part contributes to the performance with similar FLOPs, but the impact of the meat part is somewhat larger, which we attribute to that this part locates in the inverse bottleneck structure thus has a greater impact on the performance. We can see the hamburger structure offers more diversity to find a better architecture.

5.4. Discussion

Searched Architecture. Fig. 6 visualizes the structure of BurgeFormer-Tiny, where d, w, and r denote depth, width, and expansion ratio, respectively. Overall, local operations such as pooling and convolution dominate in BurgerFormer-Tiny, while there is only one global operation (i.e., Spatial MLP) in the third stage. Interestingly, the hamburger block of the fourth stage resembles a depthwise separable convolution but does not use an activation function with a different normalization. Meanwhile, the second stage uses three depthwise convolutions to obtain a larger perceptual field. More discussions can be referred to Appendix B.

Visualization. Fig. 7 visualizes learned features of DeiT-Small and BurgerFormer-Base on ImageNet. For a fair comparison, both methods use the same visualization technique (Ramprasaath R. et al., 2017). The visualized heat map is calculated by multiplying the output feature map by its gradient and then scaling it to the size of the input image. As shown in Fig. 7, BurgerFormer locates objects more accurately than DeiT, which indicates that the searched architectures have a better ability to extract features.

Micro-Meso-Macro Design. In this work, we present a micro-meso-macro schema for designing search spaces. An intuitive question is whether three granularities are needed. We believe that the co-design of the three granularities is beneficial and necessary. Micro multivariate operations can constitute diverse local and global operators, and the appropriate pairing of these is helpful to improve performance; Meso hamburger block covers the common structure of Mobilenet and Transformer block (e.g., inverse bottleneck, residuals); macro-level design can significantly affect FLOPs and parameters, and thus the model capacity. Recently, ConvNeXt (Liu et al., 2022) manually improved ResNet from three granularities to outperform Swin (Liu et al., 2021), so the automatic search for three granularities should be a direction worth thinking about and exploring.

Limitation. Limited by hardware resources, the ranges of FLOPs and parameters of the searched architectures are relatively small (e.g., maximum FLOPs $<$ 8G), and we conjecture that BurgerFormer searched under larger FLOPs or higher volume of parameters will have stronger representation capability. In addition, because all of the three granularities are involved in the micro-meso-macro search space, the search cost is high, e.g., training of the supernet takes 11 days on eight V100s. Therefore, reducing the search cost is a direction for our future work.

6. Conclusion

In this work, we design a new search space for searching BurgerFormer architectures from a micro-meso-macro perspective. Meanwhile, we propose a hybrid search strategy to train supernet effectively. Experiments on ImageNet and COCO datasets verify the effectiveness of the BurgerFormer architecture. While demonstrating better performance than state-of-the-art Transformers, this work indicates that good search space is very critical for finding out powerful visual Transformer-like architectures. In the future, we will further explore the Transformer-like search space design and experiment with other computer vision tasks.

Acknowledgement

This work is supported in part by the National Key R&D Program of China under Grant No. 2018AAA0102701, and in part by the National Natural Science Foundation of China under Grant No. 62176250.

References

A., S., Tsung-Yi, L., Niki, P., Jonathon, S., P., A., and Ashish, V. Bottleneck transformers for visual recognition. In *Conference on Computer Vision and Pattern Recognition*, 2021.

Agarap, A. F. Deep learning using rectified linear units (relu). *arXiv preprint arXiv:1803.08375*, 2018.

Ba, J. L., Kiros, J. R., and Hinton, G. E. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

Cai, H., Zhu, L., and Han, S. Proxylessnas: Direct neural architecture search on target task and hardware. In *International Conference on Learning Representations*, 2019.

Changlin, L., Tao, T., Guangrun, W., Jiefeng, P., Bing, W., Xiaodan, L., and Xiaojun, C. Bossnas: Exploring hybrid cnn-transformers with block-wisely self-supervised neural architecture search. In *International Conference on Computer Vision*, 2021.

Chen, B., Li, P., Li, C., Li, B., Bai, L., Lin, C., Sun, M., Yan, J., and Ouyang, W. Glit: Neural architecture search for global and local image transformer. In *International Conference on Computer Vision*, 2021a.

Chen, M., Peng, H., Fu, J., and Ling, H. Autoformer: Searching transformers for visual recognition. In *International Conference on Computer Vision*, 2021b.

Chen, X., Xie, L., Wu, J., and Tian, Q. Progressive differentiable architecture search: Bridging the depth gap between search and evaluation. In *International Conference on Computer Vision*, 2019.

Chengyue, G., Dilin, W., Meng, L., Xinlei, C., Zhicheng, Y., Yuandong, T., Qiang, L., and Vikas, C. Nasvit: Neural architecture search for efficient vison transformers with

gradient conflict-aware supernet training. In *International Conference on Learning Representations*, 2022.

Chu, X., Zhou, T., Zhang, B., and Li, J. Fair darts: Eliminating unfair advantages in differentiable architecture search. In *European Conference on Computer Vision*, 2020.

Chu, X., Zhang, B., and Xu, R. Fairnas: Rethinking evaluation fairness of weight sharing neural architecture search. In *International Conference on Computer Vision*, 2021.

d'Ascoli, S., Touvron, H., Leavitt, M., Morcos, A., Biroli, G., and Sagun, L. Convit: Improving vision transformers with soft convolutional inductive biases. In *International Conference on Machine Learning*, 2021.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021.

Ekin Dogus, C., Barret, Z., Jonathon, S., and Quoc V., L. Randaugment: Practical automated data augmentation with a reduced search space. In *Conference on Computer Vision and Pattern Recognition Workshops*, 2020.

Gao, H., Yu, S., Zhuang, L., Daniel, S., and Kilian Q., W. Deep networks with stochastic depth. In *European Conference on Computer Vision*, 2016.

Guo, Z., Zhang, X., Mu, H., Heng, W., Liu, Z., Wei, Y., and Sun, J. Single path one-shot neural architecture search with uniform sampling. In *European Conference on Computer Vision*, 2020.

Haiping, W., Bin, X., Noel C. F., C., Mengchen, L., Xiyang, D., Lu, Y., and Lei, Z. Cvt: Introducing convolutions to vision transformers. In *International Conference on Computer Vision*, 2021.

Han, K., Xiao, A., Wu, E., Guo, J., Xu, C., and Wang, Y. Transformer in transformer. In *Advances in Neural Information Processing Systems*, 2021.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *Conference on Computer Vision and Pattern Recognition*, 2016.

Hendrycks, D. and Gimpel, K. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016.

Hongyi, Z., Moustapha, C., Yann, D., and David, L.-P. mixup: Beyond empirical risk minimization. In *International Conference on Learning Representations*, 2018.

Hugo, T., Matthieu, C., Alexandre, S., Gabriel, S., and Herv'e, J. Going deeper with image transformers. In *International Conference on Computer Vision*, 2021.

Ilija, R., Raj Prateek, K., Ross B., G., Kaiming, H., and Piotr, D. Designing network design spaces. In *Conference on Computer Vision and Pattern Recognition*, 2020.

Ilya, L. and Frank, H. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019.

Ioffe, S. and Szegedy, C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, 2015.

Kaiming, H., Georgia, G., Piotr, D., and Ross B., G. Mask R-CNN. In *International Conference on Computer Vision*, 2017.

Langley, P. Crafting papers on machine learning. In Langley, P. (ed.), *Proceedings of the 17th International Conference on Machine Learning (ICML 2000)*, pp. 1207–1216, Stanford, CA, 2000. Morgan Kaufmann.

Liao, Y.-L., Karaman, S., and Sze, V. Searching for efficient multi-stage vision transformers. *arXiv preprint arXiv:2109.00642*, 2021.

Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., and Zitnick, C. L. Microsoft coco: Common objects in context. In *European Conference on Computer Vision*, 2014.

Liu, H., Simonyan, K., and Yang, Y. Darts: Differentiable architecture search. In *International Conference on Learning Representations*, 2019.

Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., and Guo, B. Swin transformer: Hierarchical vision transformer using shifted windows. In *International Conference on Computer Vision*, 2021.

Liu, Z., Mao, H., Wu, C.-Y., Feichtenhofer, C., Darrell, T., and Xie, S. A convnet for the 2020s. 2022.

Minghao, C., Kan, W., Bolin, N., Houwen, P., Bei, L., Jianlong, F., Hongyang, C., and Haibin, L. Searching the search space of vision transformer. In *Advances in Neural Information Processing Systems*, 2021.

Olga, R., Jia, D., Hao, S., Jonathan, K., Sanjeev, S., Sean, M., Zhiheng, H., Andrej, K., Khosla, A., Michael, B., Alexander C., B., and Li, F.-F. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 2015.

Pengchuan, Z., Xiyang, D., Jianwei, Y., Bin, X., Lu, Y., Lei, Z., and Jianfeng, G. Multi-scale vision longformer: A new vision transformer for high-resolution image encoding. In *International Conference on Computer Vision*, 2021.

Pham, H., Guan, M., Zoph, B., Le, Q., and Dean, J. Efficient neural architecture search via parameters sharing. In *International Conference on Machine Learning*, 2018.

Ramachandran, P., Zoph, B., and Le, Q. V. Searching for activation functions. In *International Conference on Learning Representations*, 2018.

Ramprasaath R., S., Michael, C., Abhishek, D., Ramakrishna, V., Devi, P., and Dhruv, B. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *International Conference on Computer Vision*, 2017.

Real, E., Aggarwal, A., Huang, Y., and Le, Q. V. Regularized evolution for image classifier architecture search. In *AAAI Con-ference on Artificial Intelligence*, 2019.

Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., and Chen, L.-C. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Conference on Computer Vision and Pattern Recognition*, 2018.

Sangdoo, Y., Dongyoon, H., Seong Joon, O., Sanghyuk, C., Junsuk, C., and Young Joon, Y. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *International Conference on Computer Vision*, 2019.

Su, X., You, S., Xie, J., Zheng, M., Wang, F., Qian, C., Zhang, C., Wang, X., and Xu, C. Vision transformer architecture search. *arXiv preprint arXiv:2106.13700*, 2021.

Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., and Wojna, Z. Rethinking the inception architecture for computer vision. In *Conference on Computer Vision and Pattern Recognition*, 2016.

Tolstikhin, I., Houlsby, N., Kolesnikov, A., Beyer, L., Zhai, X., Unterthiner, T., Yung, J., Keysers, D., Uszkoreit, J., Lucic, M., et al. Mlp-mixer: An all-mlp architecture for vision. In *Advances in Neural Information Processing Systems*, 2021.

Touvron, H., Bojanowski, P., Caron, M., Cord, M., El-Nouby, A., Grave, E., Izacard, G., Joulin, A., Synnaeve, G., Verbeek, J., et al. Resmlp: Feedforward networks for image classification with data-efficient training. *arXiv preprint arXiv:2105.03404*, 2021a.

Touvron, H., Cord, M., Douze, M., Massa, F., Sablayrolles, A., and Jégou, H. Training data-efficient image transformers & distillation through attention. In *International Conference on Machine Learning*, 2021b.

TsungYi, L., Priya, G., Ross B., G., Kaiming, H., and Piotr, D. Focal loss for dense object detection. In *International Conference on Computer Vision*, 2017.

Wang, W., Xie, E., Li, X., Fan, D.-P., Song, K., Liang, D., Lu, T., Luo, P., and Shao, L. Pyramid vision transformer: A versatile backbone for dense prediction without convolutions. In *International Conference on Computer Vision*, 2021.

Wu, B., Dai, X., Zhang, P., Wang, Y., Sun, F., Wu, Y., Tian, Y., Vajda, P., Jia, Y., and Keutzer, K. Fbnet: Hardware-aware efficient convnet design via differentiable neural architecture search. In *Conference on Computer Vision and Pattern Recognition*, 2019.

Wu, Y. and He, K. Group normalization. In *European Conference on Computer Vision*, 2018.

Xavier, G. and Yoshua, B. Understanding the difficulty of training deep feedforward neural networks. In *Artificial Intelligence and Statistics*, 2010.

Xiangxiang, C., Zhi, T., Yuqing, W., Bo, Z., Haibing, R., Xiaolin, W., Huaxia, X., and Chunhua, S. Twins: Revisiting the design of spatial attention in vision transformers. In *Advances in Neural Information Processing Systems*, 2021.

Xu, Y., Xie, L., Zhang, X., Chen, X., Qi, G.-J., Tian, Q., and Xiong, H. Pc-darts: Partial channel connections for memory-efficient architecture search. In *International Conference on Learning Representations*, 2020.

Yonglong, T., Dilip, K., and Phillip, I. Contrastive multiview coding. In *European Conference on Computer Vision*, 2020.

Yu, J., Jin, P., Liu, H., Bender, G., Kindermans, P.-J., Tan, M., Huang, T., Song, X., Pang, R., and Le, Q. Bignas: Scaling up neural architecture search with big single-stage models. In *European Conference on Computer Vision*, 2020.

Yu, W., Luo, M., Zhou, P., Si, C., Zhou, Y., Wang, X., Feng, J., and Yan, S. Metaformer is actually what you need for vision. In *Conference on Computer Cision and Pattern Recognition*, 2022.

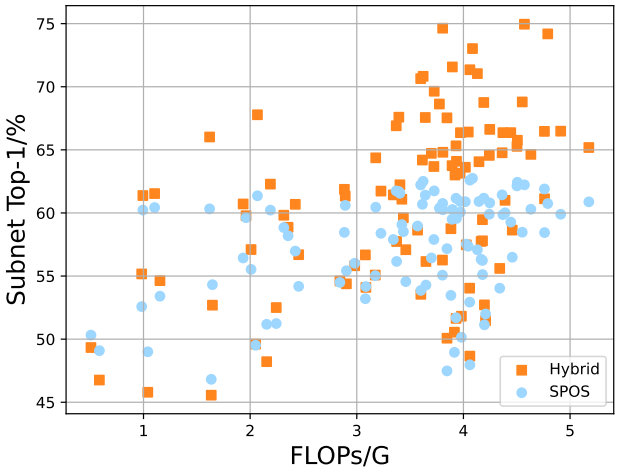
Yuan, K., Guo, S., Liu, Z., Zhou, A., Yu, F., and Wu, W. Incorporating convolution designs into visual transformers. In *International Conference on Computer Vision*, 2021.

Zhun, Z., Liang, Z., Guoliang, K., Shaozi, L., and Yi, Y. Random erasing data augmentation. In *Association for the Advancement of Artificial Intelligence*, 2020.

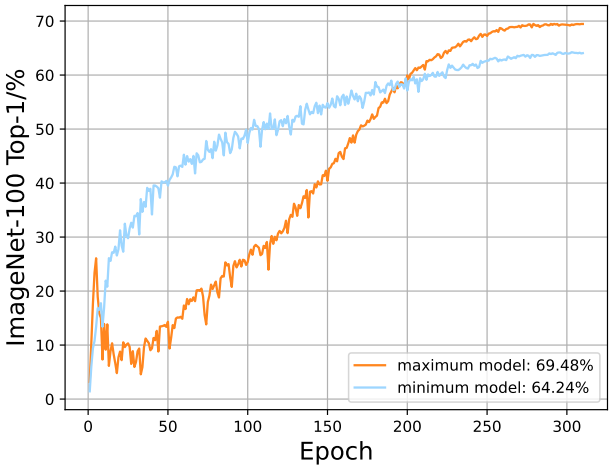
Zoph, B. and Le, Q. V. Neural architecture search with reinforcement learning. In *International Conference on Learning Representations*, 2017.

Zoph, B., Vasudevan, V., Shlens, J., and Le, Q. V. Learning transferable architectures for scalable image recognition. In *Conference on Computer Cision and Pattern Recognition*, 2018.

A. The Sampling Methods



(a) The Sub-networks Accuracy



(b) The Maximum and Minimum Models

Figure 8. (a) The sub-networks accuracy of SPOS and Hybrid Sampling Method on ImageNet. (b) The Top-1 accuracy along epochs of the maximum and minimum models on ImageNet-100.

Fig. 8(a) shows the FLOPs and accuracy of 100 random sampled sub-networks with weights inherited from the supernet. The figure indicates that the sub-networks accuracy of the SPOS method is generally lower than that of the hybrid sampling method, especially when the FLOPs are larger than 3G. Fig. 8 shows the retraining accuracy along training epochs of the maximum and minimum models on ImageNet-100. The training configuration is the same as that on ImageNet except that the batch size is 128. The experimental results demonstrate that the maximum model has much worse convergence ability than the minimum model before 200 epochs. Furthermore, the Top-1 accuracy of the maximum model is not the upper bound in the search space, so it is not suitable to train the supernet directly using the sandwich sampling method.

B. Searched BurgerFormers

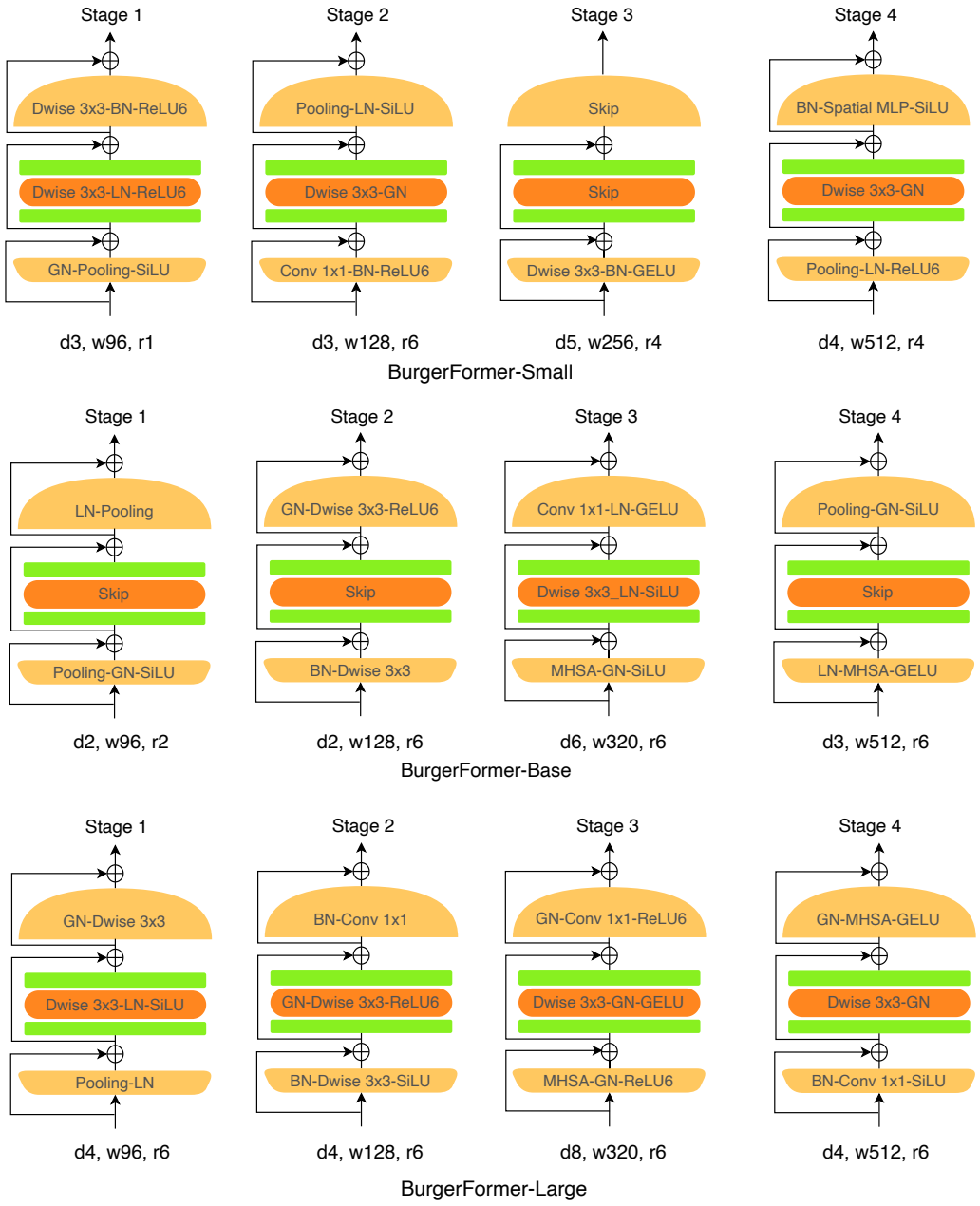


Figure 9. The architectures of BurgerFormer-Small, BurgerFormer-Base, and BurgerFormer-Large.

Fig. 9 visualizes the structure of BurgerFormer-Small, BurgerFormer-Base and BurgerFormer-Large. At the microscopic level, global operators (i.e., Spatial MLP, MHSA) always exist in stage three or stage four. And the searched architectures prefer MHSA to Spatial MLP when the model size grows. Besides, an interesting observation is that Pooling, Spatial MLP, and MHSA can be followed by an activation function, which is rarely considered in previous architecture designs. At the mesoscopic and macroscopic levels, all three parts of the hamburger block tend to be leveraged and the architectures become deeper and wider as the model size grows.

C. More details of Architectures

Stages	Tokens	Patch Embedding	MHSA Head Number
1	$\frac{H}{4} \times \frac{W}{4}$	$3 \times 3, stride\ 2, 24$ $\begin{bmatrix} 3 \times 3, stride\ 1, 24 \\ 3 \times 3, stride\ 1, 24 \end{bmatrix}$ $3 \times 3, stride\ 2, C_1$	-
2	$\frac{H}{8} \times \frac{W}{8}$	$3 \times 3, stride\ 2, C_2$	-
3	$\frac{H}{16} \times \frac{W}{16}$	$3 \times 3, stride\ 2, C_3$	4
4	$\frac{H}{32} \times \frac{W}{32}$	$3 \times 3, stride\ 2, C_4$	8

Table 6. The tokens, patch embedding and MHSA head number settings of each stage.

Table 6 shows the tokens, patch embedding, and MHSA head number settings of each stage. "$K \times K, stride\ S, C$" in the patch embedding column indicates the kernel size K, the stride S and the output channels C of convolution. Following (Liao et al., 2021), we add three additional 3x3 convolutions into the patch embedding of stage one. Since the three convolutions already have a large receptive field, we set the kernel size three in the first patch embedding.