GLiT: Neural Architecture Search for Global and Local Image Transformer

Boyu Chen[1]*, Peixia Li[1]*, Chuming Li[1,2], Baopu Li[3]†, Lei Bai[1], Chen Lin[4],
Ming Sun[2], Junjie Yan[2], Wanli Ouyang[1]†
[1] The University of Sydney, [2] SenseTime Group Limited,
[3] BAIDU USA LLC, [4] University of Oxford

Abstract

We introduce the first Neural Architecture Search (NAS) method to find a better transformer architecture for image recognition. Recently, transformers without CNN-based backbones are found to achieve impressive performance for image recognition. However, the transformer is designed for NLP tasks and thus could be sub-optimal when directly used for image recognition. In order to improve the visual representation ability for transformers, we propose a new search space and searching algorithm. Specifically, we introduce a locality module that models the local correlations in images explicitly with fewer computational cost. With the locality module, our search space is defined to let the search algorithm freely trade off between global and local information as well as optimizing the low-level design choice in each module. To tackle the problem caused by huge search space, a hierarchical neural architecture search method is proposed to search the optimal vision transformer from two levels separately with the evolutionary algorithm. Extensive experiments on the ImageNet dataset demonstrate that our method can find more discriminative and efficient transformer variants than the ResNet family (e.g., ResNet101) and the baseline ViT for image classification. The source codes are available at https://github.com/bychen515/GLiT.

1. Introduction

Convolutional Neural Networks (CNN) -based architecture (e.g., ResNet [14]) contributes to the great success of deep learning in computer vision tasks [26, 6, 19] for past several years. By stacking a set of CNN layers, CNN-based models can achieve larger receptive filed and perceive more contextual information on scarifies of the efficiency. Driven by the great success of transformer [30] in Natural Language Processing(NLP) tasks, there are increasing interests

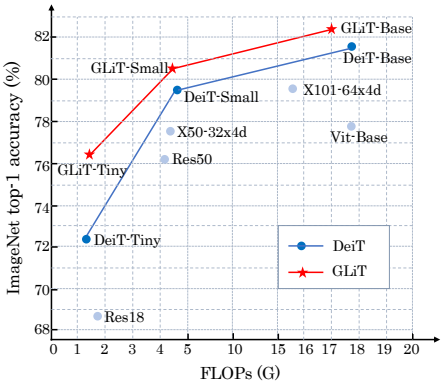

Figure 1. Top-1 accuracy (y-axis) and FLOPs (x-aixs) for different backbones on ImageNet. GLiT is our method.

in the computer vision community to develop more efficient architectures based on the transformer [11, 28, 4, 37] which can manipulate the global correlations directly. Among these works, vision transformer (ViT) is a representative one [11] as it does not rely on the CNN-based backbone to extract features and solely relies on self-attention modules in transformer to establish global correlations among all input image patches. While ViT achieves impressive performance, if extra training data is not used, ViT still has lower accuracy than the well-designed CNN models such as ResNet-101 [14]. To further exploit the potential of transformer in image recognition tasks, DeiT [28] uses teacher-student strategy for distilling knowledge to the transformer token. These two methods rely on the original transformer architecture but neglect potential gap between NLP tasks and image recognition tasks in architecture.

In this work, we argue that there are unignorable gaps between different kinds of data modalities (e.g., image and text), leading to the disparities between different tasks. Thus, directly applying the vanilla transformer architecture to other tasks may be sub-optimal. It is natural that there exists better transformer architectures for image recognition. However, hand-designing such an architecture is time consuming since there are too many influential factors to be considered. On one hand, Neural Architecture Search (NAS) has achieved great progress in computer vi-

*Equal contribution
†Corresponding author

sion tasks [13, 22, 2]. It can automatically discover an optimal network architecture without manual try-and-error. On the other hand, computer vision community still has not investigated NAS for transformers.

Based on the above observations, we intend to discover a better transformer architecture by NAS for specific tasks, e.g., the image classification task in this work.

There are two key factors when designing NAS for transformer in vision tasks, a well-designed search space that contains candidates with good performance and an efficient searching algorithm to explore the search space.

A naïve search space would only contain the architecture parameters in the transformer architecture, such as the feature dimension for query and value, the number of attention heads in the Mutli-Head Attention (MHA) mechanism, and the number of MHA blocks. However, the search space does not consider two factors. First, the self-attention mechanism in transformer cost quadratic memory and computational burden w.r.t the number of input tokens [37] during the inference stage. Second, local recurrence in human visual system [16, 17] is not realized in the transformers like ViT and DeiT. Inspiration from the local recurrence in human visual system leads to the success of convolutional layer, locally connected layers for computer vision tasks [18]. Although theoretically possible, it is hard to model sparse local correlations by the vaninlla self-attention mechanism (e.g., a fixed-size neighbor tokens) in practice.

Considering the two factors mentioned above, we expand the search space of the vanilla transformer by introducing a locality module to the MHA. The locality module only operates on the nearby tokens, requiring fewer parameters and computation. The locality module and the self-attention module are alternative, which is searched by NAS to decide which one is used. We rename the expanded MHA block as the global-local block as it can capture both global and local correlations among the input tokens. According to our experiments (Table 1), the flexibility of the transformer in capturing global and local information is an important factor for the final performance.

Introducing global-local block should be effective, but poses challenge to the searching algorithm. The NAS algorithm for our search space should 1) discover the optimal distribution of locality modules and self-attention modules in each global-local block, and 2) find the detailed settings of both locality modules and self-attention modules by searching the module parameters. Such a search space is extremely huge (10^{18} times of the possible choices in [13] and 10^{12} times of the possible choices in [22]), which makes it challenging for existing NAS methods like SPOS [13] in geting an ideal result. To deal with the problem mentioned above, we propose a Hierarchical Neural Architecture Search method to find the optimal networks. Specifically, we first train a supernet that contains both locality modules and self-attention modules, and determine the high-level global and local sub-modules distribution with evolutionary algorithm. Then, the detailed architecture within each module are searched in a similar manner. Compared with traditional searching strategies, the proposed hierarchical searching can stabilize the searching process and improve the searching performance.

Fig. 1 shows that our searched Global Local image Transfomer (GLiT) achieves up to 4% absolute accuracy increase when compared with the state-of-the-art tranformer backbone DeiT on ImageNet.

To summarize, the main contributions are as follows:

- So far as we know, concurrent with [8], we are the first to explore better transformer architecture by NAS for image classification. Our work finds a new transformer variant that achieves better performance than ResNet101 and ResNeXt101 using the same training setting without pre-training on extra data.
- We introduce locality modules to the search space of vision transformer model, which not only decreases the computation cost but also enables explicitly local correlation modeling.
- We propose a Hierarchical Neural Architecture Search strategy, which can handle the huge searching space in the vision transformer efficiently and improves the searching results.

2. Related work

Transformers in Vision. The vision community has witnessed bloom of interest and attention in combining transformers with CNN, including DETR [4] and Deformable DETR [37] for object detection, ViT [11] and DeiT [28] for image classification, and IPT [7] for multi low-level tasks. Different from DETR [4] and Deformable DETR [37], our method does not rely on CNNs for feature extraction. Instead, the whole model is totally based on transformer architecture. Deformable DETR [37] introduces local mechanism to reduce computation by only attending to small set of key sampling points around a reference. The new local mechanism is not well optimized on GPUs, so training Deformable DETR still needs quadratic memory costs. Differently, our proposed locality module helps to reduce not only the computation but also the memory resources. It is more efficient than the local attention in Deformable DETR.

Global and Local Attention in NLP. Transformers based on self-attention technique were proposed in [30] to replace RNN for sequence learning on machine translation and become state-of-the-art since then. We are inspired by the use of global and local attention in [12, 3] for NLP. Longformer [3] splits the original global attention with mask global attention and masked local attention for long sequence learning. Our introduction of local attention is inspired by Conformer [12], which combines convolutions

with self-attention to build the global and local interactions in Automatic Speech Recognition (ASR) model. However, it is unknown whether the Conformer for NLP is effective for image recognition. Different from Conformer and Longformer for NLP tasks, we introduce the convolution as the Local Attention in the transformers for the image classification task. Besides, our exploration on searching the distribution of global and local sub-modules in a network by NAS is not investigated in [3, 12].

Global and Local Attention in Vision. Similar as NLP field, global and local attention mechanism is also proved effective in computer vision tasks. SAN [34] proposes pairwise and patchwise attention mechanism for image recognition. [15, 31] achieve the performance gain from both global and local information. SENet [15] introduces the channel-wise attention in the local connected convolution network. [31] utilizes Non-local blocks to capture long-range dependencies in CNNs. Recently, BotNET [27] replaces the last residual blocks of ResNet through transformer blocks to extract global information. All the above methods manually design attention mechanism to CNNs, while our focus is to introduce the local attention to vision transformers and automatically search for the optimal global-local setting.

Neural Architecture Search. Recently, NAS methods make great progress on the vision tasks [5, 9, 23, 35, 20, 21]. Early NAS mothods apply RL [2, 38] or EA [25] to train multiple models and update the controller to generate model architectures. To reduce the searching cost, weight-sharing methods are proposed. These methods construct and train the supernet which includes all architecture candidates. Darts [22] proposes a differentiable method to jointly optimize the network parameters and architecture parameters. SPOS [13] proposes a single-path one-shot method, which trains only one subnet from the supernet in each training iteration. After supernet training, the optimal architecture is found through Evolutionary Algorithm (EA). However, due to the memory restriction (Darts [22]) or low correlation problems (SPOS [13]), these two methods cannot handle our search space with too many candidate architectures. We propose the Hierarchical Neural Architecture Search to solve problem caused by huge search space.

NAS has been used to search an optimal architecture for NLP models. AutoTrans [36] designs a special parameter sharing mechanism for RL-based NAS to reduce the searching cost. [29] proposes a sampling-based one-shot architecture search method to get a faster model. NAS-BERT [32] constructs a big supernet including multiple architectures and find optimal compressed models with different sizes in the supernet. Different from the above methods, we focus on NAS for transformer on image classification instead of NLP tasks. Concurrent with our work, [8] proposes Weight Entanglement method to search the optimal architecture for original ViT model. Different from [8], we introduce locality into vision transformer models and propose Hierarchical Neural Architecture Search to handle huge search space.

3. Method

We propose Global-Local (GL) transformer and search its optimal architecture. The *GLiT* consists of multiple global-local blocks (Sec. 3.1) which are constructed by introducing local sub-modules to the original global blocks, as shown in Fig. 2. Based on the global-local blocks, we design the *specific search space* for vision transformer (Sec. 3.2), as described in Table 2. Accordingly, the *hierarchical neural architecture search* method (Sec. 3.3) is proposed for better searching results, with which we first search the high-level global-local distribution and then detailed architecture for all modules, as shown in Fig 4.

Similary with other vision transformers [11, 28], the GLiT receives a 1D sequence of token embeddings as input. To handle 2D images, we split each image into several patches and flatten each patch to a 1D token. The features of an image is represented by $F \in \mathbb{R}^{w\times h\times c}$, where c, w and h are the channel size, width and height of the image. We split the image features F into patches of size $m \times m$ and flatten each patch to a 1D variable. Then, $F \in \mathbb{R}^{w\times h\times c}$ is reshaped to $\tilde{F} \in \mathbb{R}^{m^2\times \frac{cwh}{m^2}}$, which consists of m^2 input tokens. We combine the m^2 input tokens with a learnable class token (shown in green at the 'Input' of Fig. 2) and send all m^2+1 tokens to the GLiT. Finally, the output class token from the last block is sent to a classification head to get the final output.

3.1. Global-local block

There are two kinds of modules in the global-local block, including global-local module (in the green dotted box) and Feed Forward Module (FFN), as shown in the Fig 2.

3.1.1 Global-local module

Self-Attention as the Global Sub-module. All m^2+1 input tokens are linearly transformed to queries $Q \in \mathbb{R}^{(m^2+1)\times d_k}$, keys $K \in \mathbb{R}^{(m^2+1)\times d_k}$ and values $V \in \mathbb{R}^{(m^2+1)\times d_v}$, where d_k and d_v are the dimension of features for each token in the query (keys) and value. $d_k = d_v$ in the design of transformer. The global attention is calculated by:

$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}})V. \quad (1)$$

This module calculates the attention results by considering the relationship among all input tokens, so we named this self-attention head as the global sub-module in this paper.

The formulation in Equation (1) is further modified to Multi-Head Attention (MHA) mechanism, where

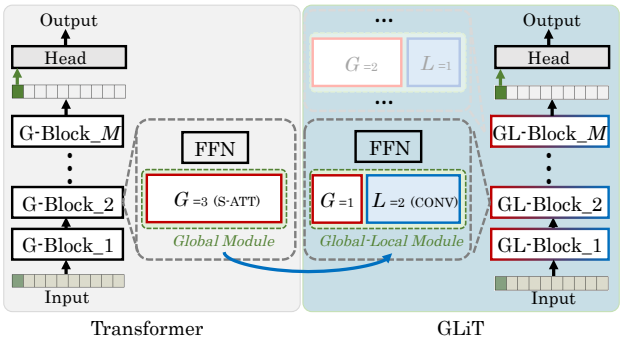

Figure 2. The construction of GLiT. 'S-ATT' is self-attention head, 'CONV' denotes convolution head, G and L are the numbers of self-attention and convolution heads. The original transformer consists of only global module and Feed Forward module, *i.e.* the 'FFN' in the figure. We further introduce local sub-module to the global module and get the Global-Local module. GLiT is constructed by M GL blocks. The distribution of global and local sub-modules may be different in different GL blocks. For example, GL-Block_2 in this figure has $G = 1$ global sub-module and $L = 2$ local sub-modules.

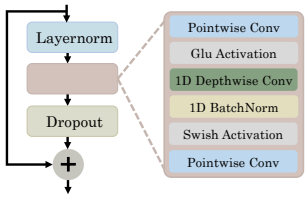

Figure 3. Convolution layers in the local sub-module.

the queries, keys and values are split into N parts along the dimensions, whose outputs are denoted as $head_0, head_1, ..., head_i, ..., head_N$,

$$head_i(Q_i, K_i, V_i) = softmax(\frac{Q_i K_i^T}{\sqrt{d_{head_i}}})V_i, \quad (2)$$

where Q_i, K_i and V_i are the i_{th} part of Q, K and V, d_{head_i} is the dimension of each head and is equal to $\frac{d_k}{N}$. The output values of N heads are concatenated and linearly projected to construct the final output.

Convolution heads as Local Sub-module. 1D convolution has been used in NLP tasks [33, 12] to model local information. Inspired by the Conformer blocks in [12], we apply 1D convolution to establish local connections, which is named local sub-module in the following description. As shown in Fig. 3, one convolution head consists of three convolutional layers, including two point-wise convolutional layers with one 1D depth-wise convolutional layer between them. Each convolutional layer is followed by normalization, activation (such as GLU [10]) and dropout layers. The first point-wise convolutional layer followed by Glu activation has an expansion ratio E to expand the feature dimension to E times. After that, the 1D depth-wise convolutional layer with kernel size K does not change the feature dimension. Finally, the last point-wise convolutional layer projects the feature dimension back to the input dimension.

We utilize 1D convolution layer instead of 2D convolution layer in local sub-modules, as it is more suitable for 1D sequence of input tokens. Besides, the $m^2 + 1$ input tokens in GLiT can not be directly reshaped to a 2D array.

Constructing Multi-head Global-Local Module. Given global and local sub-modules, next question is how to combine them. We construct the global-local module by replacing several heads in the MHA with local sub-modules. For example, if there are $N = 3$ heads in the MHA, we can keep one MHA head ($head_0$) unchanged and replace two heads ($head_1$ and $head_2$) with our local sub-module. If all heads in MHA are global sub-modules, then the global-local block degenerates to the transformer block used in ViT [11] and DeiT [28]. In the global-local module, queries, keys and values are only calculated for the heads implemented by global sub-module. For the heads implemented by local sub-module, inputs are directly sent to convolution layers.

Table 1 shows the experimental results of evaluating the GLiT with different ratios of the global and local sub-modules in the global-local block. As we can see, the ratio of global and local sub-modules has obvious influence on the performance. Simply replacing all self-attention heads by convolution heads will cause a huge performance drop due to the lack of global information. On the other hand, the network with 1 self-attention head and 2 convolution heads in every global-local module performs the best among all models, improving 1.8% Top-1 accuracy compared with the baseline model. The performance variation with different ratios between self-attention and convolution heads demonstrates that introducing local information brings more performance gains only with proper global-local ratio.

Table 1. Performance comparisons of different head distributions in DeiT-Tiny model [28] on ImageNet dataset. All blocks utilize the same distribution of heads. Here, the total head number in each transformer block is 3. The first row with 3 self-attention heads and 0 Conv1d head is the baseline model corresponding the the ViT in [11]. In the 2nd, 3rd and 4th rows, we gradually replace self-attention heads (global sub-modules) with more Convolution heads (local sub-modules).

Self-attention head number	Conv1d head number	Acc (%)
3	0	72.20
2	1	72.89
1	2	73.98
0	3	71.02

3.1.2 Feed Forward Module

Apart from the global-local module, there is a Feed Forward Module (FFN) in each GL-block to further transform input features. The FFN consists of a Layer Normalization and two fully-connected layers with a Swish activation

and dropout layer between them. Mathematically, the FFN $f(X)$ for its input $X \in \mathbb{R}^{m^2 \times d}$ can be represented as:

$$f(X) = \sigma(LN(X)W_1 + b_1)W_2 + b_2, \tag{3}$$

where $LN(\cdot)$ denotes Layer Normalization [1], $\sigma(.)$ is the Swish activation function, $W_1 \in \mathbb{R}^{d \times d_m}$ and $W_2 \in \mathbb{R}^{d_m \times d}$ are weights of fully-connected layers, $b_1 \in \mathbb{R}^{d_m}$ and $b_2 \in \mathbb{R}^d$ are the bias terms, d and d_m are respectively the feature dimension of input for the first and the second FC layers. We denote $d_z = \frac{d_m}{d}$ as the expansion ratio of FFN.

3.2. Search space of the global-local block

The search space of proposed global-local block includes the high-level global-local sub-module distribution and low-level detailed architecture of each sub-module. At the high-level, we aim to search the distribution of convolution and self-attention heads over all global-local blocks. At the low-level, we search the detailed architecture of all sub-modules. Table 2 summarizes the high-level and low-level search space implemented in this paper.

High-level global-local distribution. Suppose there are N_m heads in the mth transformer block $m \in \{1, \ldots, M\}$, we can keep $G \in \{0, \ldots, N_m\}$ self-attention heads unchanged and replace L self-attention heads with convolution heads ($L = N_m - G$), so there are $N_m + 1$ different variations of the global-local distribution in the mth block. The candidate high-level designs for the mth block is $\mathcal{N}_m = [(0, N_m), (1, N_m - 1), ..., (j, N_m - j), ..., (N_m, 0)]$, where $(j, N_m - j)$ denotes $G = j$ self-attention heads and $L = N_m - j$ convolution heads in the global-local module. The high-level search space contains the candidate high-level designs for all M blocks , *i.e.* $\mathcal{N} = \mathcal{N}_0 \times \mathcal{N}_1 \times ... \times \mathcal{N}_m \times ... \times \mathcal{N}_M$, where $\times$ denotes Cartesian product.

Low-level Detailed architecture. The search space of detailed architecture focuses on four items: the feature dimension d_k of queries (keys, values) in self-attention heads, the expansion ratio d_z of FFN, the expansion ratio E of the first point-wise convolution layer and the kernel size K of the 1D depth-wise convolutional layer in the convolution heads. Table 2 lists all the possible choices for d_k, d_z, E, and K. Suppose the total candidate numbers of d_k, d_z, E, K are V_1,V_2,V_3,V_4, we can get the search operation sets $\mathcal{D}_k = [d_k^1, d_k^2, ..., d_k^{V_1}]$, $\mathcal{D}_z = [d_z^1, d_z^2, ..., d_z^{V_2}]$, $\mathcal{E} = [E_p^1, E_p^2, ..., E_p^{V_3}]$ and $\mathcal{K} = [K_d^1, K_d^2, ..., K_d^{V_4}]$. Random selecting one operation from each set can form a candidate global-local block, so there are totally $V_1V_2V_3V_4$ candidates of one global-local block on the low-level.

It should be noted that all convolution heads in one block share the same architecture, similarly for self-attention heads. For block $m \in \{1, \ldots, M\}$, the inner search operation sets for convolution, self-attention and FFN are $\mathcal{E}_m \times \mathcal{K}_m$, $\mathcal{D}_{km}$ and $\mathcal{D}_{zm}$, where $\times$ denotes Cartesian product. The overall search space for block m is $\mathcal{S}_m = \mathcal{D}_{km} \times \mathcal{D}_{zm} \times \mathcal{E}_m \times \mathcal{K}_m$ and the final search space in the low-level is $\mathcal{S} = \mathcal{S}_0 \times \mathcal{S}_1 \times ... \times \mathcal{S}_m \times ... \times \mathcal{S}_M$.

Table 2. Search Space for GLiT. 'Local' is the local sub-module, 'Global' is the global sub-module and 'FFN' is the Feed Forward Module. (G, L) denotes the number of global and local sub-modules in each block. K is the kernel size of local sub-module, E is the expansion ratio of local sub-module, d_k is the feature dimension of global sub-module, d_z is the expansion ratio in FFN.

High-Level	(G,L)		(0,3), (1,2), (2,1), (3,0)
Low-Level	Local	K	17, 31, 45
		E	1, 2, 4
	Global	d_k	96, 192, 384
	FFN	d_z	2, 4, 6

Search Space Size. Considering both the high-level distribution and detailed architectures on the low-level, the candidate number of each block is about $(N + 1)V_1V_2V_3V_4$. In our search space, different blocks have different high-level distributions and detailed architectures. If a transformer has M blocks, the final search space contains $((N + 1)V_1V_2V_3V_4)^M$ candidate networks, which is an extremely huge search space. For our implementation in Table 2 for $M = 12$ blocks, $((N+1)V_1V_2V_3V_4)^M \approx 1.3 \times 10^{30}$, which is about 10^{12} times the search space of DARTS [22] and 10^{18} times the search space of SPOS [13]. The main-stream fast Neural Architecture Search methods, such as differential [22] and one-shot [13] method, can not work well on this huge search space. For DARTS, the parameters of all candidate networks are trained for each iteration, leading to an unacceptable memory requirements. One-shot NAS method does not have the above problem, because they only select one candidate network during each training iteration. However, the correlation between the retrained subnet and the subnet sampled from the supernet is lower under the huge search space, so the architectures searched using supernet become unreliable. To solve the searching problem on the huge search space, we propose the Hierarchical Neural Architecture Search method to get the optimal network architecture with suitable memory requirements.

3.3. Hierarchical Neural Architecture Search.

The Hierarchical Neural Architecture Search method consists of two main stages, as shown in Fig. 4. First, we search the optimal distribution $\mathcal{N}^*$ of the global and local sub-modules in each block. Then, we fixed the distribution $\mathcal{N}^*$ and search the detailed architecture $\mathcal{S}^*$ of both the global and local sub-modules.

First Stage. At the first stage, we fixed the low-level detailed architecture parameters d_k, d_z, E and K for global and local sub-modules. The one-shot NAS method SPOS

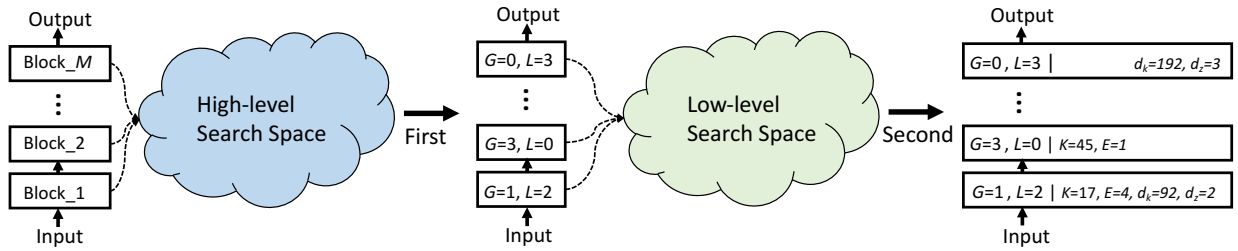

Figure 4. The framework of Hierarchical Neural Architecture Search. First, we find the optimal distribution of local (L) and global (G) sub-modules in the high-level search space. For example, $L = 1, G = 2$ means 1 local sub-module and 2 global sub-modules in the global-local module. Then, the detailed architecture for all sub-modules is searched in the low-level search space (detailed in Table 2).

in [13] is applied to search the optimal distribution of global and local sub-modules from the search space $\mathcal{N}$. There are three main steps when the SPOS is applied: supernet training, subnet searching and subnet retraining. In each iteration of supernet training, we 1) randomly sample indices $[j_0, j_1, ..., j_M]$; 2) use these indices to sample a subnet from the supernet, where the number of global and local sub-modules in the M blocks is $[(j_0, N_0 - j_0), (j_1, N_1 - j_1)..., (j_m, N_m - j_m), ..., (j_M, N_M - j_M)]$; and 3) train the subnet. After supernet training, we utilize Evolutionary Algorithm [13] at the subnet searching step to find the top-5 optimal architectures according to validation accuracy. Finally, at the subnet retraining step, we retrain the five networks and choose the architecture with the highest validation accuracy as the output model, where the distribution of global and local sub-modules is $\mathcal{N}^* = [(j_0^*, N_0 - j_0^*), (j_1^*, N_1 - j_1^*)..., (j_m^*, N_m - j_m^*), ..., (j_M^*, N_M - j_M^*)]$.

Second Stage. After obtaining the optimal the distribution of global-local modules in all blocks at the first stage, we fix this distribution and search the detailed architecture of all modules. Similar to the first stage, we adopt SPOS [13] to find the optimal architecture $\mathcal{S}^*$ in the search space. The main difference is changed search space and correspondingly the random index of a block is an array with four elements, instead of a single number j_m. The random index of block m is $(j_m^1, j_m^2, j_m^3, j_m^4)$, which corresponds to the index of $(\mathcal{D}_{km}, \mathcal{D}_{zm}, \mathcal{E}_m, \mathcal{K}_m)$ respectively.

The proposed search method has two main advantages compared with existing NAS methods [13, 22]. First, the proposed method divides the huge search space into two smaller search spaces. As mention above, the size of original search space is $((N+1)V_1V_2V_3V_4)^M$. With our proposed search method, the total size of two smaller search space is $(N+1)^M + (V_1V_2V_3V_4)^M$, which is reduced to less than 10^{-7} times the original search space for our implementation in Table 2. Second, the size of low-level search space $(V_1V_2V_3V_4)^M$ can be further reduced with a fixed global-local distribution. As shown in Fig. 5, after the first searching stage, most blocks in the searched architecture include either global or local sub-modules and only two blocks have both global and local sub-modules. For most blocks, the size of low-level search space is V_1V_2 or V_3V_4, instead of $V_1V_2V_3V_4$. To fix the size of search space for each block, we reduce the search space for the blocks with both global and local sub-modules, by only searching d_z and E for these blocks. With the hierarchical search method, the final search space falls into the effective search space range of existing NAS methods. The significantly reduced search space makes it easier for SPOS in obtaining better model.

4. Experiments

We evaluate our GLiT on image classification task. In Section 4.2, we compare our searched transformer architectures with DeiT [28], which is a recently published algorithm on vision transformer. In Section 4.3, we design more experiments to show the necessity of our search space and search method. All experiments are tested on NVIDIA GTX 1080Ti GPU with the Pytorch framework.

4.1. Implementation Details

Dataset. All experiments are conducted on ImageNet, which consists of 1.28M images in $train$ set and 50,000 validation images in $test$ set. We split 50K samples from $train$ set to construct val set. The val set is used for subnet evaluation during the searching process.

Hyper-parameters. We adopt mini-batch Nesterov SGD optimizer with a momentum of 0.9 during the supernet training. We utilize the learning rate 0.2 and adopt cosine annealing learning rate decay from 0.2 to 0. We train the network with a batch size of 1024 and L2 regularization with weight of 1e-4 for 100 epochs. Besides, the label smoothing is applied with a 0.1 smooth ratio. For subnet searching, we follow the EA setting in [13], which samples $N_s = 1000$ subnets under the FLOPs constraint in total. For the searched model retraining, we follow the training settings in DeiT [28].

4.2. Overall Results on ImageNet

We compare the searched transformer with two CNNs (ResNet and ResNeXt) and the state-of-the-art vision transformer DeiT [28]. Table 3 shows the results under different computational budgets. The results for the existing models, such as R18 (Resnet-18) and R50 (Resnet-50), in Table 3

Table 3. Classification accuracy of different models on ImageNet. 'Acc' denotes the Top-1 accuracy and '^{s}' denotes the models trained using the training configurations in DeiT [28]. R18 denotes Resnet-18, X50 denotes Resnext-50.

Models	Params(M)	FLOPS(G)	Acc(%)
R18	11.7	1.8	69.8
R18^s	11.7	1.8	68.5
DeiT-Tiny^s	5.7	1.3	72.2
GLiT-Tiny^s	**7.2**	**1.4**	**76.3**
R50	25.6	4.1	76.1
R50^s	25.6	4.1	78.5
X50-32x4d	25.0	4.3	77.6
X50-32x4d^s	25.0	4.3	79.1
DeiT-Small^s	22.1	4.6	79.6
GLiT-Small^s	**24.6**	**4.4**	**80.5**
X101-64x4d	83.5	15.6	79.6
X101-64x4d^s	83.5	15.6	81.5
Vit-Base	86.6	17.6	77.9
DeiT-Base^s	86.6	17.6	81.8
GLiT-Base^s	**96.1**	**17.0**	**82.3**

are copied from the results reported in [28]. We also use the training setting in [28] and report the results for R18, R50, X50-32x4d (Resnext-50), and X101-64x4d (Resnext-101), with ^{s} followed by these models in Table 3 for denoting the same training configurations. Our models achieve better accuracy than all compared networks under similar FLOPS restrictions. For example, our searched model with 1.3G FLOPS restriction achieves 76.3% accuracy score, which is higher than both DeiT-Tiny and ResNet18 (R18) by more than 4 points and 6 points respectively. Our searched models achieves obvious improvement in accuracy from the symphony our two designs: local information and architecture search. The local information brought by Conv1d without proper distribution has limited improvement according to Table 1. However, the searched global and local information distribution shows much better performance according to our ablation study and the detailed architecture search will further improve the performance of our GLiT model.

4.3. Ablation study

In this section, we conduct experiments to demonstrate the necessity of our searching space and effectiveness of Hierarchical Neural Architecture Searching method. All ablation studies are based on our GLiT-Tiny model on ImageNet using the same training setting as before.

Searching Space. The proposed searching space includes two levels, the global-local distribution and the detailed architecture of all modules. To verify the effectiveness of both levels, we investigate our model with the model searched only on global-local distribution ('Only distribution' in Table 4), and the baseline model DeiT-Tiny with human design. The model with only global and local distribution searched performs much better than the baseline model DeiT-Tiny without NAS. It also outperforms the best human-designed architecture with global-local information (73.98% in Table 1), improving the accuracy by about 1.5%. The performance gains come from the optimal transformer architecture with proper global-local information distribution searched by our method. After considering the detailed architecture of all modules, the final model ('Ours') further improves the accuracy about 1%, which is significant on ImageNet. Besides, we also compare our search space with that in NLP-NAS [29]. The search space in [29] includes the expansion ratio (query, key and value) and MLP ratio. The searched model from the NLP-NAS search space achieves a 73.4% top-1 accuracy on ImageNet, which is 1.2% higher than DeiT but 2.9% lower than ours. Directly using the search space in [29] limits performance improvements. Our search space and search method are more effective, which are our two main contributions. The experimental results in Table 4 demonstrate all components in our searching space is essential for an excellent vision transformer.

Table 4. Performance comparisons of models from different searching spaces on ImageNet. 'DeiT-Tiny' is the baseline model which totally relies on hand-designing. 'Only distribution' is the model searched only on global-local distribution. 'Ours' is the model searched from the complete searching space.

Method	Flops(G)	Acc
DeiT-Tiny	1.3	72.2
Only distribution	1.4	75.4
Ours	1.4	76.3
NLP-NAS	1.4	73.4

Searching Method. We compare the proposed searching method with the baseline NAS method (SPOS) and random search baseline. All searching methods are used in our proposed search space. For fair comparisons, we train the supernet of SPOS for 200 epochs, which is the same as the training epochs in ours. After supernet training, we select 5 architectures and retrain them for SPOS and ours. For the random baseline, we randomly sample and retrain 5 networks. The architecture with the highest retraining accuracy are chosen as the final model. In Table 5, our method achieve much better performance than the SPOS method and random search baseline. The performance improvement is from our hierarchical searching method, which reduce the searching space effectively in all stages. For SPOS method, since the supernet is constructed by the huge search space, the optimization of the supernet is different, even with the double training epochs compared with our supernet. Due to the insufficient training, the subnet with lower computation will converge faster and the searched result on the validation set tends to be a model with lower computation as shown in Table 5. Not only the model in Table 5, but also the other four models in the top-5 models identified by SPOS have small flops of 1.0G, 0.9G, 0.9G, and 0.9G. As a result, the searching result with vanilla SPOS method has

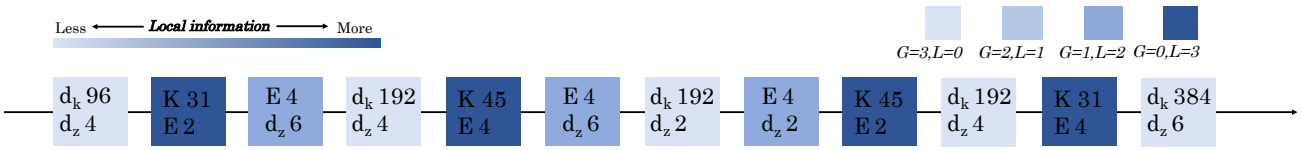

Figure 5. The architecture of GLiT-Tiny in Table 3. Each box represents a global-local block. The darker the color denotes the more local sub-modules in the block. L is the number of local sub-modules in each block. G is the number of global sub-modules.

performance even worse than random search.

Table 5. Performance comparisons between SPOS, Random searching baseline, and our searching method on ImageNet.

Method	Flops(G)	Acc
Ours	1.4	76.3
SPOS	0.9	72.7
Random search	1.3	73.3

Methods to introduce locality. We choose Conv1D to introduce the locality into Vision Transformer. However, there are other methods such as restricting self-attention in a local area or using Conv2D. We evaluate these two methods and compare them with ours. For restricting self-attention in a local area, we set the candidate local area sizes as (14, 28, 49). We keep the other settings (including the hierarchical search) fixed for fair comparison. In Table 6, the model with local self-attention has only 72.4% top-1 accuracy on ImageNet (much lower than ours 76.3%), possibly due to the lack of communication among different local areas. However, Conv1D in our GLiT model can solve this issue. To use Conv2D in our network, we remove the CLS token and add a global average pooling at the end of the network for classification. The candidate kernel sizes are set as $(3 \times 3, 5 \times 5, 7 \times 7)$. The searched model with Conv2D achieves 76.4% accuracy, which is similar with ours. For fair comparison with our baseline model ViT and DeiT, which utilize the CLS token, we adopt Conv1D in our final models.

Table 6. Performance comparisons between Self-attention in a local area, Conv2D and our searching method on ImageNet.

Method	Flops(G)	Acc
Local-area	1.4	72.4
Conv2d	1.4	76.4
Conv1d	1.4	76.3

4.4. Discussion.

Searched architecture. Fig. 5 shows the searched architecture of GLiT-Tiny (Table 3). There are only 25% blocks consists of both global and local sub-modules. Most blocks contains either global or local sub-modules. Sequential connection between global and local sub-modules may be more necessary than parallel connection. There is no 1D convolution layer with kernel size 17 in the searched architecture. 17 is the smallest value of kernel size in the search space. It shows that too small kernel size is not suitable for locality modules in vision transformers. The searched architecture has a trend of all-local, to local-global mixture, and then back to all-local blocks. This helps local and global features interact through the transformer blocks. This architecture looks like a mechanism similar to the stacked hourglass in [24], which has stacks local-global CNNs, where 'local' corresponds to CNN with high-resolution features and 3×3 convolution has smaller receptive fields, while 'global' corresponds to CNN features with lower resolution and a 3×3 convolution looks at more global region of the same image.

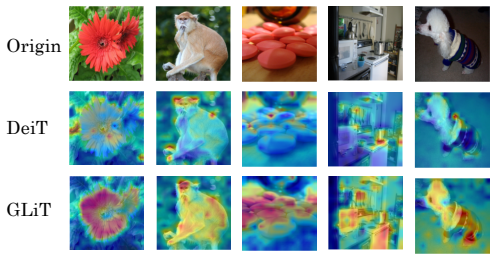

Figure 6. Visualization of features for DeiT [28] (second row) and our GLiT (third row). Images in the first row are from ImageNet.

Visualization. In Fig. 6, we show the visualization of learned features of both DeiT (the second row) and our GLiT (the last row). We calculate the heat maps by reshaping the output tokens to the same size as input images and averaging the reshaped tokens along channels. By reaching a good combination of local and global features, our GLiT focuses on more object regions than DeiT.

5. Conclusion

We exploit better architectures for vision transformers in this paper through carefully designing the searching space with local information and the hierarchical searching method. Transformer is applied to vision community not long ago. Its architecture is not well exploited for image recognition. Our method provides a feasible and automatic network design strategy. In addition to showing better performance compared with existing vision transformers, this work will inspire more researches on finding optimal transformer architecture for computer vision tasks.

Acknowledgements This work was supported by the Australian Research Council Grant DP200103223, FT210100228, and Australian Medical Research Future Fund MRFAI000085.

References

[1] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

[2] Irwan Bello, Barret Zoph, Vijay Vasudevan, and Quoc V. Le. Neural optimizer search with reinforcement learning. In *ICML*, 2017.

[3] Iz Beltagy, Matthew E Peters, and Arman Cohan. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*, 2020.

[4] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *ECCV*, 2020.

[5] Boyu Chen, Peixia Li, Baopu Li, Chen Lin, Chuming Li, Ming Sun, Junjie Yan, and Wanli Ouyang. Bn-nas: Neural architecture search with batch normalization. In *ICCV*, 2021.

[6] Boyu Chen, Dong Wang, Peixia Li, Shuang Wang, and Huchuan Lu. Real-time'actor-critic'tracking. In *ECCV*, 2018.

[7] Hanting Chen, Yunhe Wang, Tianyu Guo, Chang Xu, Yiping Deng, Zhenhua Liu, Siwei Ma, Chunjing Xu, Chao Xu, and Wen Gao. Pre-trained image processing transformer. *arXiv preprint arXiv:2012.00364*, 2020.

[8] Minghao Chen, Houwen Peng, Jianlong Fu, and Haibin Ling. Autoformer: Searching transformers for visual recognition. *arXiv preprint arXiv:2107.00651*, 2021.

[9] Yuanzheng Ci, Chen Lin, Ming Sun, Boyu Chen, Hongwen Zhang, and Wanli Ouyang. Evolving search space for neural architecture search. In *ICCV*, 2021.

[10] Yann N. Dauphin, Angela Fan, Michael Auli, and David Grangier. Language modeling with gated convolutional networks. In Doina Precup and Yee Whye Teh, editors, *ICML*, 2017.

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

[12] Anmol Gulati, James Qin, Chung-Cheng Chiu, Niki Parmar, Yu Zhang, Jiahui Yu, Wei Han, Shibo Wang, Zhengdong Zhang, Yonghui Wu, and Ruoming Pang. Conformer: Convolution-augmented transformer for speech recognition. In Helen Meng, Bo Xu, and Thomas Fang Zheng, editors, *INTERSPEECH*, 2020.

[13] Zichao Guo, Xiangyu Zhang, Haoyuan Mu, Wen Heng, Zechun Liu, Yichen Wei, and Jian Sun. Single path one-shot neural architecture search with uniform sampling. In *ECCV*, 2020.

[14] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016.

[15] Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. In *CVPR*, 2018.

[16] Tim C Kietzmann, Courtney J Spoerer, Lynn KA Sörensen, Radoslaw M Cichy, Olaf Hauk, and Nikolaus Kriegeskorte. Recurrence is required to capture the representational dynamics of the human visual system. *Proceedings of the National Academy of Sciences*, 2019.

[17] Jonas Larsson and David J Heeger. Two retinotopic visual areas in human lateral occipital cortex. *Journal of neuroscience*, 2006.

[18] Yann LeCun, Yoshua Bengio, et al. Convolutional networks for images, speech, and time series. *The handbook of brain theory and neural networks*, 1995.

[19] Peixia Li, Boyu Chen, Wanli Ouyang, Dong Wang, Xiaoyun Yang, and Huchuan Lu. Gradnet: Gradient-guided network for visual object tracking. In *ICCV*, 2019.

[20] Xiang Li, Chen Lin, Chuming Li, Ming Sun, Wei Wu, Junjie Yan, and Wanli Ouyang. Improving one-shot nas by suppressing the posterior fading. In *CVPR*, 2020.

[21] Feng Liang, Chen Lin, Ronghao Guo, Ming Sun, Wei Wu, Junjie Yan, and Wanli Ouyang. Computation reallocation for object detection. *arXiv preprint arXiv:1912.11234*, 2019.

[22] Hanxiao Liu, Karen Simonyan, and Yiming Yang. DARTS: differentiable architecture search. In *ICLR*, 2019.

[23] Jie Liu, Chuming Li, Feng Liang, Chen Lin, Ming Sun, Junjie Yan, Wanli Ouyang, and Dong Xu. Inception convolution with efficient dilation search. In *CVPR*, 2021.

[24] Alejandro Newell, Kaiyu Yang, and Jia Deng. Stacked hourglass networks for human pose estimation. In *ECCV*, 2016.

[25] Esteban Real, Alok Aggarwal, Yanping Huang, and Quoc V. Le. Regularized evolution for image classifier architecture search. In *AAAI*, 2019.

[26] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28:91–99, 2015.

[27] Aravind Srinivas, Tsung-Yi Lin, Niki Parmar, Jonathon Shlens, Pieter Abbeel, and Ashish Vaswani. Bottleneck transformers for visual recognition. In *CVPR*, 2021.

[28] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In *ICML*, 2021.

[29] Henry Tsai, Jayden Ooi, Chun-Sung Ferng, Hyung Won Chung, and Jason Riesa. Finding fast transformers: One-shot neural architecture search by component composition. *arXiv preprint arXiv:2008.06808*, 2020.

[30] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NLP*, 2017.

[31] Xiaolong Wang, Ross B. Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks. In *CVPR*, 2018.

[32] Jin Xu, Xu Tan, Renqian Luo, Kaitao Song, Li Jian, Tao Qin, and Tie-Yan Liu. Task-agnostic and adaptive-size bert compression. 2020.

[33] Qian Zhang, Han Lu, Hasim Sak, Anshuman Tripathi, Erik McDermott, Stephen Koo, and Shankar Kumar. Transformer transducer: A streamable speech recognition model with transformer encoders and RNN-T loss. In *ICASSP*, 2020.

[34] Hengshuang Zhao, Jiaya Jia, and Vladlen Koltun. Exploring self-attention for image recognition. In *CVPR*, 2020.

[35] Dongzhan Zhou, Xinchi Zhou, Wenwei Zhang, Chen Change Loy, Shuai Yi, Xuesen Zhang, and Wanli Ouyang. Econas: Finding proxies for economical neural architecture search. In *CVPR*, 2020.

[36] Wei Zhu, Xiaoling Wang, Xipeng Qiu, Yuan Ni, and Guotong Xie. Autotrans: Automating transformer design via reinforced architecture search. *arXiv preprint arXiv:2009.02070*, 2020.

[37] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable detr: Deformable transformers for end-to-end object detection. *arXiv preprint arXiv:2010.04159*, 2020.

[38] Barret Zoph, Vijay Vasudevan, Jonathon Shlens, and Quoc V. Le. Learning transferable architectures for scalable image recognition. In *CVPR*, 2018.