Advances in Artificial Intelligence and Machine Learning; Research 5 (1) 3216-3235 Received 27-11-2025; Accepted 4-01-2025; Published 11-01-2025

Flexible Transformer: A Simple Novel Transformer-based Network for Image Classification in Variant Input Image Sizes

Zijiang Yang z-yang-4w3@eagle.sophia.ac.jp
Department of Information and Computer Sciences
Sophia University
Chiyoda-ku, Tokyo 102-8854, Japan

Tad Gonsalves t-gonsal@sophia.ac.jp
Department of Information and Computer Sciences
Sophia University
Chiyoda-ku, Tokyo 102-8854, Japan

Corresponding Author: Zijiang Yang

Abstract

Convolutional neural networks (CNNs), the most important deep learning networks for computer vision, have undergone a series of developments and improvements for image-related tasks such as object recognition, image classification, semantic segmentation, etc. However, in the field of natural language processing (NLP), the novel attention-based network Transformer had a profound impact on machine translation, which subsequently led to a boom in attention-based models for computer vision. State-of-the-art models with attention have already shown good performance for computer vision tasks due to the sophisticated design of network architectures and advanced computational efficiency techniques. For example, self-attention learns relationships between segments or words in different positions compared to the current performance of convolutional neural networks. Inspired by Vision Transformer (ViT), we propose a simple novel transformer architecture model, called Flexible Transformer, which inherits the properties of attention-based architectures and is flexible for inputs of arbitrary size. Besides self-attention, the inputs in ViT are not pre-processed, such as resizing or cropping, but kept intact without altering them, which could lead to distortion or loss of information. In this paper, we want to present a novel and simple architecture that meets these requirements. Compared to state-of-arts, our model processes inputs with arbitrary image sizes without any pre-processing and pretraining costs. Also, the results of the experiments show that the model can potentially provide good results with high accuracy despite limited resources. Even though the results of Flexible Transformer are not as accurate as those of Vision Transformer, they show the potential of a model with high performance in image classification tasks with variable size images. The significance of the research opens up possibilities for dealing with primitive images in deep learning tasks. Based on the original inputs, reliable results with good accuracy could be obtained if the proposed model is optimized and further trained on large datasets.

Citation: Zijiang Yang and Tad Gonsalves. Flexible Transformer: A Simple Novel Transformer-based Network for Image Classification in Variant Input Image Sizes. Advances in Artificial Intelligence and Machine Learning. 2025;5(1):185.

Keywords: Convolutional Neural Network (CNN), Vision transformer, Self-Attention, Spatial Pyramid Pooling (SPP), Flexible transformer.

1. INTRODUCTION

Inspired by the BERT model [1], for machine translation in NLP, Vision Transformer [2], was introduced for image-related tasks by using self-attention techniques as a paradigm and foundation for further research on self-attention models in computer vision. Inspired by Vision Transformer, various transformer-based self-attention models such as Swin Transformer [3], CrossFormer [4], and Pyramid Vision Transformer [5], have achieved good performance on image tasks with derived architectures. Compared to the CNN architecture and its variants, self-attention models attempt to learn relationships among pixels in combination with additional position information. Like NLP tasks such as machine translation and text recognition [6], the relative positional relationship is also a critical factor for models to understand images, as semantically different positions of objects can influence the decision of models in computer vision tasks, which is not considered or highlighted in classical CNN models due to their inherent convolutional architecture.

Compared to classical CNN models, instead of combining convolution with pooling algorithms for feature extraction, Transformer-based models proposed a novel idea that performs embedding [1, 6], for patches after pre-processing and subsequently self-attention for the embedded patches in the transformer block. During self-attention, each patch pays attention to the others by following a certain rule [2–5, 7] (e.g., shifted window [2]). On the other hand, each patch should pay more attention to the related patches, revealing the importance of those related patches to it. Adding positional encodings such as Transformer [1, 2, 6], also learns the impact of positions for each element of an input on performance. Transformer with self-attention brings substantial benefits to computer vision and novel feature extraction strategies that are excluded from CNN counterparts.

To circumvent the limitations of CNN models, researchers are devoted to proposing decent transformer or attention-based architectures [1–6], to make a breakthrough in the field of computer vision. However, designing fantastic models is complicated and requires thoughtful concepts. Since it is a new concept in the field of computer vision, despite the release of some delicate transformer models with good performance, there is still much room for improvement in terms of accuracy, computational efficiency, power, learning performance, etc.

Research into Transformer-based architectures for deep learning is still in the early stages. Existing models are equipped with some more exquisite designs, such as shifted windows and spatial reduction attention [3], which are more efficient for self-attention and allows some tolerance in accuracy compared to the state-of-the-art. Nevertheless, these sophisticated models may sacrifice some accuracy when it comes to variable-sized inputs due to their reliance on fixed positional encoding strategies (either global [2], or local [3–5]). Hence, the adaptability to variable size input images not only allows the models to avoid laborious pre-processing steps, but also to keep the original input images intact without resizing, which may lead to the loss or distortion of some key features. With the ability to handle input images of different sizes, it takes less or even no effort in image pre-processing steps such as resizing and scaling. In this paper, we aim to present a novel and simple architecture that meets these requirements.

Transformer-based architectures attained great success in recent years with fantastic state-of-arts. However, prerequisites for success of these models are sufficiently reliable data with identical size. To meet this requirement, raw data needs to be pre-processed prior to learning procedure. Regardless of any pre-processing fashions, information in data may potentially be lost or be corrupted in order to fix the size, and if a titanic dataset is required, pre-processing steps would be laborious and expensive. Thus, such dilemma motivates the design of novel architectures that take advantage of transformer. With sufficient datasets in arbitrary size for pre-training, the performance of model is expected to be further improved and adapt to different downstream tasks.

The purpose of the research is to explore a transformer-based model for images in different sizes for image classification tasks with competitive accuracy in performance within a certain tolerance. In other words, the main goal is to explore a flexible transformer-based model that adapts to the variable size of inputs without losing too much accuracy compared to other de-facto transformer-based models instead of pursuing the most accurate results in performance. The following contributions are expected from this research:

- It proposes a simple, but novel Transformer-based model for image classification tasks, that can be applied to other image tasks.
- It proposes a model with a novel self-attention strategy for addressing problems with inputs of different sizes of images in image classification.
- It conveys a signal for considering problems with variable sizes in image tasks as a reference for future related topics.
- The ideas underlying the research encourage researchers to develop new advanced models for image-related tasks.
- It improves models with flexibility in processing arbitrary-size inputs and no cost for input preprocessing and pretraining.

The rest of the paper is organized as follows: We give a description of the research background and related architectures in the next section. Then, the architecture for models in the research is elaborated, which is followed by the section for experiments and results. Following the experimental results, we show the analysis and summary of research experiments. Finally, we end the paper with the research conclusions and outlooks for future works.

2. RELATED WORK

Convolutional neural networks have dominated in the realm of computer vision in the past decades, with considerable research focusing on advanced CNN-based models [8–11]. GoogleNet[12, 13], ResNet[14], AlexNet[15], are designed with a large number of convolutional layers and residual networks, optimizing learning progress and improving performance tasks. For instance, Deep Residual Network gains accuracy with increase in depth by using residual networks for optimization. Optimization techniques like Batch Normalization [16–18], Layer Normalization [19, 20], speeds up performance for achieving high accuracy in image tasks. Meanwhile, Transformer-based architectures like BERT has become a state-of-the-art standard for natural language processing [1, 6].

The idea for self-attention mechanism learns relationships among all components or segments in the inputs to extract features, reflecting how related each component is to the others.

Inspired by the attention [6], in machine translation [1], Vision Transformer [2], was introduced for large-scale image classification tasks, pre-training with gigantic image datasets JFT-300M [21–23]. It simply applies the standard Transformer Encoder architecture with as few modifications as possible. In the scenario for input images, input words are substituted with image patches fed into the Transformer Encoder after the sequence of linear embeddings combined with positional encoding that contains information about the positions. However, the highly accurate results produced by the Vision Transformer is at the price of large amount of pre-training and high computational costs owing to the immense number of parameters in the training models. To mitigate those problems, some algorithms were developed to speed up the performance or save computational resources. Instead of global self-attention for all patches in ViT, [3] proposed a new Vision Transformer architecture with hierarchical structure for feature representation by a shifted window for local self-attention, which only produces linear computational complexity. Alternatively, DeiT [24], applies self-attention in knowledge-distillation strategy [24–27], for effective training with smaller image datasets. Other variants such as Pyramid Vision Transformer [5], CrossFormer [4], etc. proposed unique architectures for feature representation to replace attention layers in the Transformer Encoder block. Compared to the original Vision Transformer, subsequent variants optimized models by modification of self-attention layers (e.g., from global to local and long-short distance attention [4]) as well as patch embedding architectures [2–5], for feature representations.

Success in attention-based architectures improves models for image-related tasks in many aspects such as computational efficiency and quality of output result. It even extends the discussion of image data intro video frames [28], that consist of images in continuous time and add time dimension based on image data. Models with high quality of input data would give even better results in specific tasks. After introductions to self-attention architectures [1, 2, 6], in images, to deal with demand in high-quality image data, explorations in image tasks with transformers also expand to focus on super-resolution images [29, 30], and video data [31]. Based on vision transformer, [28] proposes video vision transformer for video classifications based on success in image classification and attains state-of-art results with multiple video data. Subsequently, with the possibility in video tasks, much research succeeds in generating high-quality images and videos. For instance, in the field of video inpainting, [31] introduces a transformer-based network Flow-Guided Global Local Aggregation Transformer (FVIFormer) modules based on self-attention and optical features. Similarly, it is worthwhile in producing and restoring super-resolution images for different purposes. For example, in the field of medical diseases, [29] proposes a novel Reinforced Transformer Network that attempts to learn local part of coronary in Coronary CT Angiography and improves the diagnosis of cardiovascular diseases based on progressive reinforcement learning and transformer-based architecture. To generate high-resolution images, [30] proposes multi-scale attention network that considers spatial relationship and optimizes computation based on multi-scale attention and gated spatial attention.

As mentioned above, successful Transformer architectures were built upon sufficiently large datasets for a huge set of training parameters (millions or even billions like JFT-300M). However, when the massive amount of data is unavailable or the data is insufficient for large models for good performance, one solution is to pre-train the model with larger datasets and then conduct a regular learning process with the target dataset. However, it is not always available for large pre-training datasets like

JFT-300M. To address the problem of limited size of datasets, various data augmentation techniques [26, 32, 33], or other policies [34–37], for optimization such regularization [38], were proposed for deep learning models in computer vision. Specifically, [24] proposed a novel knowledge-distillation technique to avoid noise or distortion by data augmentation, allowing tolerance for class-level similarities with semantically similar labels with the ground truth as a result of teacher model's supervision or guide for student model. AutoAugment [32], introduced a simple procedure for searching for proper data aug-mentation policies from the given datasets.

3. MODEL DESCRIPTION

Our model, similar to other attention-based architectures [6], is inspired from BERT and ViT models [2]. Rather than altering the internal layers and self-attention algorithm in the transformation block, we maintain a similar structure and layers to ViT model but make a small modification to handle varying input image sizes. The new model, benefiting from self-attention from the ViT model, can process input images of varying sizes. Additionally, in order to handle varying sequence lengths for classification, an additional layer is added before the classification layer. This section delves into the intricacies of our model, providing a thorough examination of each layer and the accompanying algorithms. The overall algorithm for flexible transformer is shown below, with definition of each key component demonstrated later in this section.

Algorithm 1 Flexible Transformer

procedure FlexTrans($d, p, head, dim, hidden_dim, num_class, [b_1, b_2, b_3], SP\_dim$)
  $features \leftarrow PatchEmbedding(d, p, dim)$
  for $i \leftarrow 1, head$ **do**
    $features \leftarrow TransEncoderBlock(features, dim, hidden\_dim)$
  end for
  $features \leftarrow SpatialPyramidPooling(features, [b_1, b_2, b_3], SP\_dim)$
  $outputs \leftarrow MLP\_Head(features, num\_class)$
  return $outputs$
end procedure

procedure MLP_Head($inputs, num\_class$)
  $outputs \leftarrow LayerNorm(inputs)$
  $outputs \leftarrow flatten(outputs)$
  $outputs \leftarrow Dense(outputs, num_class)$
  $outputs \leftarrow softmax(outputs)$
  return $outputs$
end procedure

3.1 Flexible Transformer

As mentioned earlier, our approach involves adopting the structure and architecture of the ViT model, but with modifications for variable-sized images. We named our new model Flexible Transformer because it can adapt to varying input sizes in image-related datasets.

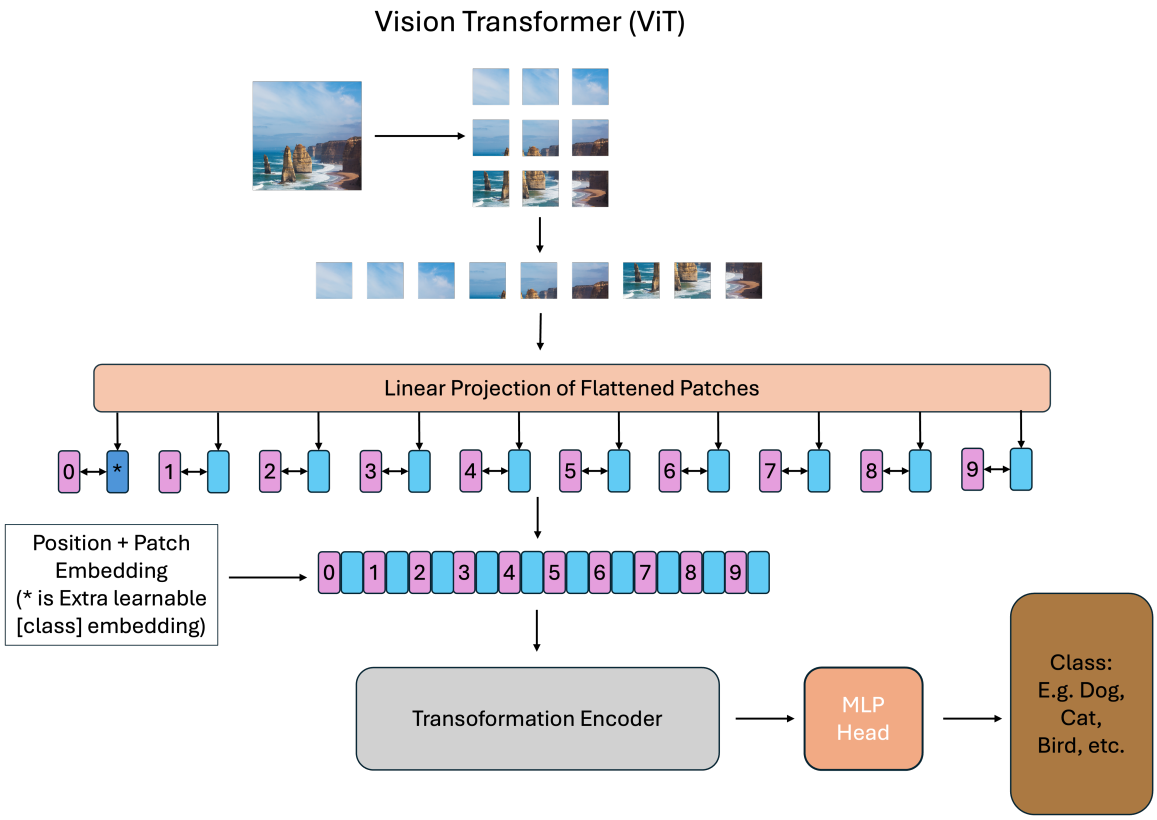

Figure 1: Vision Transformer Model Architecture

An overview of the ViT model is shown in FIGURE 1. Input image is first split into small patches based on pre-defined patch size. Then patches are aligned and fed into linear projection layer with specific dimensions. Afterwards, each embedded patch denoted by the blue box is attached with positional information in the pink box that is embedded into the same dimension based on the position of the input patch. Specifically, the first positional information (represented as 0) with embedded patch represents class information for input image. Combination of positional information and embedded patches is further processed in transformation encoder with multi-head self-attention. Similar to the Vision Transformer, our model consists of layers for patch extraction, linear projection, transformer encoder, MLP head as well as the output layer, except that we remove the position embedding component after the linear projection of the flattened patches. FIGURE 2 shows the architecture of our Flexible Transformer model. Besides, prior to patch extraction, the model takes a 2D input image $x \in \mathbb{R}^{H\times W\times C}$, where H and W are the height and width of the original image in pixels, and C is the number of channels. To extract patches, the input images are reshaped into 2D image patches $x_p \in \mathbb{R}^{N\times(p^2\cdot C)}$ where (p, p) is the size of each patch and $N = HW/p^2$ is the number of patches after extraction. However, instead of reshaping an input image into a patch by a given patch size p, the input images are divided into groups of patches with different patch sizes. For example, given a 2D input image $x \in \mathbb{R}^{H\times W\times C}$ and a set of n patch sizes $P = \{p_1, p_2, p_3, ..., p_n\}$, the input image is reshaped into n groups of patches $x_{p_1} \in \mathbb{R}^{N_1\times(p_1^2\times C)}$, $x_{p_2} \in \mathbb{R}^{N_2\times(p_2^2\times C)}$, $x_{p_3} \in \mathbb{R}^{N_3\times(p_3^2\times C)}$,..., and $x_{p_n} \in \mathbb{R}^{N_n\times(p_n^2\times C)}$, where $N_1 = HW/p_1^2$, $N_2 = HW/p_2^2$, $N_3 = HW/p_3^2$, ..., and $N_n = HW/p_n^2$ are the number of patches after extraction in patch

sizes (p_1, p_1), (p_2, p_2), (p_3, p_3), ..., (p_n, p_n). All patches are aligned separately in 1D sequences of lengths $N_1, N_2, N_3, ..., N_n$ which serve as input length sequences to the Transformer Encoder. FIGURE 1 shows two patch sizes in set of patches $P = \{p_1, p_2, ...\}$ where $p_1 = 2, p_2 = 3$ such that $N_1 = HW/p_1^2 = HW/4$ and $N_2 = HW/p_2^2 = HW/9$. On the other hand, the input image is split into $4, 9, ...,$ and p_n^2 patches with patch size equal to $2, 4, ...,$ and p_n.

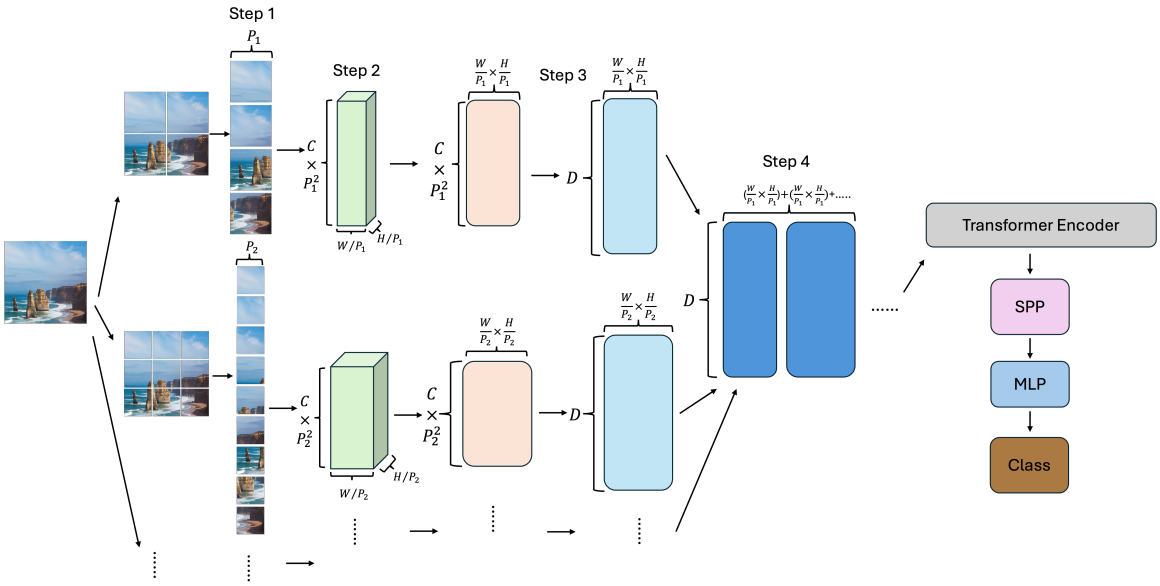

Figure 2: Flexible Transformer Model Architecture

Before feeding into the Transformer Encoder block, rather than a regular linear projection that maps input sequences into D dimensions, we apply linear projection to all separate patch sequences. In ViT, a sequence of image patches $x_p \in \mathbb{R}^{N\times(p^2\times C)}$ is mapped into the sequence with given patch size (p, p) as shown in Equation 1:

$$z_0 = [x_{class}; x_p^1 E; x_p^2 E; x_p^3 E; ...; x_p^N E] + E_{pos}, E \in \mathbb{R}^{(P^2\cdot C)\times D}, E_{pos} \in \mathbb{R}^{(N+1)\times D} \quad (1)$$

where D is the dimension of the linear projection. In contrast, both the number of patches and the corresponding patch size are not fixed in our model, so that in the linear projection, given an input image of size (H, W, C), and the set of patch size $P = \{p_1, p_2, p_3, ..., p_n\}$, we map the input image to n different patch sequences of sizes (p_1, p_1), (p_2, p_2), (p_3, p_3), ..., (p_n, p_n) so that each sequence can be represented as $x_{p_n} \in \mathbb{R}^{N_n\times(p_n^2\times C)}$, where the number of patches is $N_n = (HW/p_n^2)$. Then, we map each sequence into D dimensions via linear projection, and the output is $z_{l_n} \in \mathbb{R}^{N_n\times D}$. The linear projection of each sequence is given by Equation 2.

$$z_{l_n} = [x_{p_n}^1 E; x_{p_n}^2 E, ..., x_{p_n}^n E], E \in \mathbb{R}^{(P^2\times C)\times D} \quad (2)$$

Compared to the linear projection in ViT, we remove the class embedding and the position embedding. Instead, we expand the number of patch sequences by using multiple patch sizes, so the number of patches after linear projection is not fixed. Instead of performing position embedding,

we simply merge the sequences, generating a large sequence block with D dimensions. Equation 3 shows the result of the merged sequence:

$$z_l = [z_{l_1}; z_{l_2}; ...; z_{l_n}], z_{l_n} \in \mathbb{R}^{N_t \times D}, N_t = N_1 + N_2 + \cdots + N_n \tag{3}$$

In FIGURE 2, the input image is split into groups of patches, which are represented as multiple sequences depending on the patch size, shown in step 1. Then, in step 2 and 3 each sequence of patches is mapped to the same dimension D, in which they can be merged into a large sequence block for the next layer with self-attention, shown in step 4.

Once the merging of the sequences of linear projection is completed, the large sequence is fed into the transformation block. From this step onwards, we follow the same procedure as in Vision Transformer [2]. Similarly, the linear projection sequence is fed into the transformer encoder, followed by an additional layer to define the size of the features and the MLP head for the final output. Algorithm for process of position and patch embedding after linear projection of input data is shown in Algorithm 2.

Algorithm 2 Patch Embedding and Position encoding

```
procedure PositionPatchEmbedding(d, p, dim)
    features ← extractPatches(d, p)
    features ← reshapePatches(features)
    features ← Embedding(dim)
    positions ← random(d.shape)
    outputs ← Concentate(positions, features)
    return outputs
end procedure
```

3.2 Transformer Encoder

After merging sequences of the linear projection, the feature maps are shifted to the transformer encoder blocks with multi-head attention. Similar to [2], we retain transformer design as close as possible. As the core of the model, we follow the multi-head self-attention as follows:

$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}})V \tag{4}$$

$$MultiHead(Q, K, V) = Concat(head_1, head_2, ..., head_n)W^O \tag{5}$$

$$head_i = Attention(QW_i^Q, KW_i^K, VW_i^V), i = 1, 2, .., n \tag{6}$$

where $W_i^Q \in \mathbb{R}^{d_{model} \times d_k}$, $W_i^K \in \mathbb{R}^{d_{model} \times d_k}$, $W_i^V \in \mathbb{R}^{d_{model} \times d_v}$ and $W^O \in \mathbb{R}^{hd_v \times d_{model}}$. However, in our models, we set up different number of heads corresponding to the ViT counterparts. Therefore, the number of heads is $head_i = d_v = d_{model}/h$. Similar to ViT model, as shown in FIGURE 3, Layer Normalization (LN)[19], is applied as normalization layer in each transformation block, represented as Norm Layer in orange box. Depending on configuration for each model variant, the number of transformation block is different, represented as L. Algorithm for each trasformer encoder block is shown in Algorithm 3

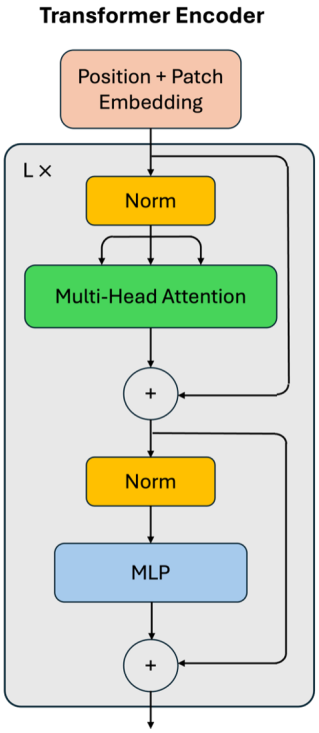

Figure 3: Transformer Encoder

Algorithm 3 Transformer Encoder Block

procedure TransEncoderBlock($d, dim, hidden_dim$)
    $output1 \leftarrow LayerNorm(d)$
    $output2 \leftarrow MultiHeadAttention(output1)$
    $output3 \leftarrow d + output1$
    $output4 \leftarrow LayerNorm(output3)$
    $output5 \leftarrow MLP(output4, dim, hidden_dim)$
    $outputs \leftarrow output3 + output5$
    return $outputs$
end procedure

procedure MLP($inputs, dim, hidden_dim$)
    $output \leftarrow Dense(dim)$
    $output2 \leftarrow gelu(output1)$
    $output3 \leftarrow Dropout(output2)$
    $output4 \leftarrow Dense(dim)$
    $outputs \leftarrow Dropout(output4)$
    return $outputs$
end procedure

3.3 Spatial Pyramid Pooling

After the iterations of transformer encoder blocks, the size of the output features is different due to different size of the input images. However, the MLP head for the classification stage requires features with identical size for the final classification due to its fully connected layer. To solve this issue, we choose to add an extra layer – spatial pyramid pooling (SPP) layer to fix the size of the output features after the Transformation Encoder block.

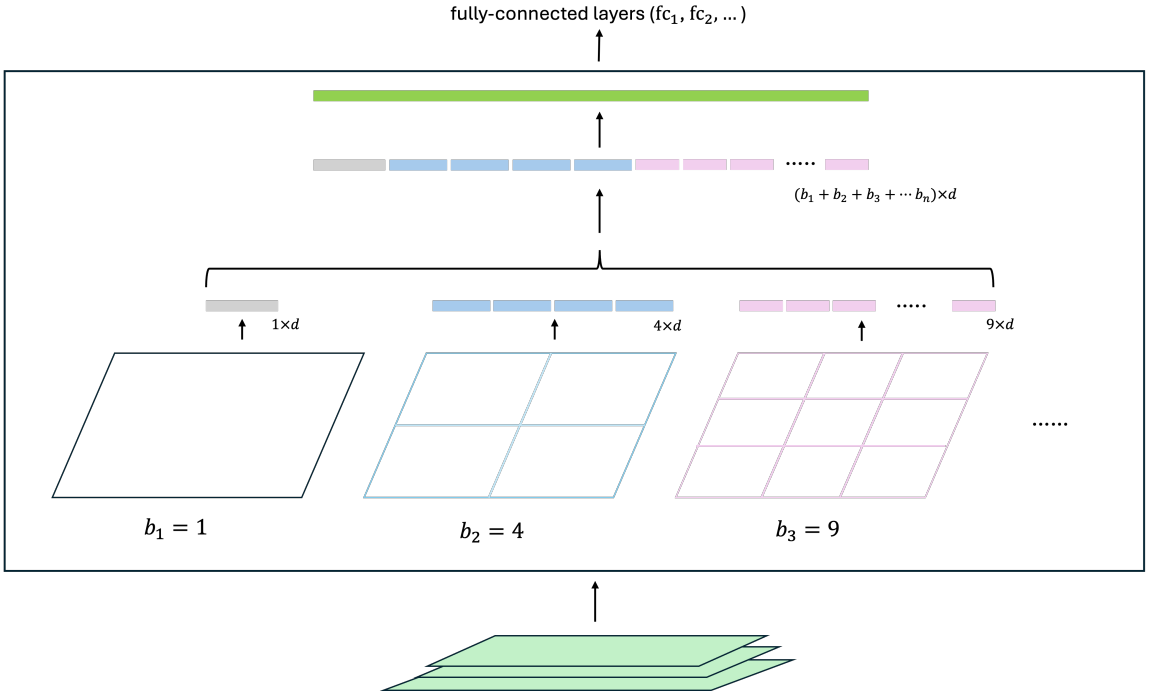

Figure 4: Spatial Pyramid Pooling

Similar to traditional CNN, our model takes various sizes of inputs and generates outputs with variable sizes, which are not valid as inputs to the MLP head. In CNN and its variants, inputs of arbitrary size are accepted by the convolutional layer, which generates feature maps of arbitrary size. In fact, only the fully connected layer requires a fixed input size. In our classification model, we also use an MLP head for the final classification. Input features with arbitrary size are fed into the SPP layer, generating fixed-sized outputs, pooled by spatial bins of size $n \times n$ in m different levels, each of which is represented with a different size of bin. In each level, the size of bin specifies the total number of segments split. Given an input feature x with size (m, n) from transformation blocks and suppose the SPP layer contains spatial bins B_q of size (b_q, b_q) in l levels (where q is between 1 and l), each level generates feature maps based on the size of the corresponding spatial bin during the pooling process. FIGURE 4 shows an example of three different levels with bin size $b_1 = 1$, $b_2 = 4$ and $b_3 = 9$ in different colors. In this case, we follow the SPP layer in [35], by using max pooling for feature map in each bin. For each level, the output feature is calculated as:

$$Y_q = maxPool(x, B_q) \tag{7}$$

where the size is (b_q, b_q). Specifically, for each entry $y_{(i,j)}$ in the vector Y_q, the size of the pooling filter $F_{(i,j)}$ is $(\lceil \frac{m}{b_q} \rceil, \lceil \frac{n}{b_q} \rceil)$ where i and j are between 1 and $(b_q - 1)$, $(\lfloor \frac{m}{b_q} \rfloor, \lceil \frac{n}{b_q} \rceil)$ where $i = b_q$ and j is between 1 and $(b_q - 1)$, $(\lceil \frac{m}{b_q} \rceil, \lfloor \frac{n}{b_q} \rfloor)$ where i is between 1 and $b_q - 1$ and $j = b_q$, and $(\lfloor \frac{m}{b_q} \rfloor, \lfloor \frac{n}{b_q} \rfloor)$ where $i = b_q$ and $j = b_q$. Therefore, due to the pooling for each entry, the size of the output feature of the spatial bin in one level is (b_q, b_q). Then the output features from each level of the spatial bin are linearly merged so that the final output of the SPP layer is as follows:

$$Y = [Y_1, Y_2, ..., Y_q], Y_q \in \mathbb{R}^{b_q \times b_q}, Y \in \mathbb{R}^{b \times b} \tag{8}$$

where $b = b_1 + b_2 + \cdots + b_n$, which is finally used by the MLP for final classification.

The model inherits typical characteristics of vision transformer such as global self-attention. However, by splitting images into sequences with different sized patches, the model can perform self-attention between regions in a single patch, since area region in a large single patch in one sequence can be split into two or even more small patches in other sequences, allowing the model to learn more information from the input dataset. In addition, our model is flexible with respect to arbitrary image sizes without the need to reshape or crop to the same image size, which potentially loses information or distorts objects and ends up altering image semantics. By retaining the original image, all information remains intact and the most reliable results are achieved as long as the model is optimized for performance. Algorithm for SPP is shown in Algorithm 4

Algorithm 4 Spatial Pyramid Pooling

procedure SpatialPyramidPooling(d, $[b_1, b_2, b_3]$, SP_dim)
  $output1 \leftarrow Maxpooling2D(d, b_1)$
  $output1 \leftarrow Reshape(output1)$
  $output1 \leftarrow Dense(output1, SP\_dim)$
  $output2 \leftarrow Maxpooling2D(d, b_2)$
  $output2 \leftarrow Reshape(output2)$
  $output2 \leftarrow Dense(output2, SP\_dim)$
  $output3 \leftarrow Maxpooling2D(d, b_3)$
  $output3 \leftarrow Reshape(output3)$
  $output3 \leftarrow Dense(output3, SP\_dim)$
  $outputs \leftarrow Contentate(output1, output2)$
  $outputs \leftarrow Contentate(output, output3)$
  return $outputs$
end procedure

4. DATASETS

We use two open-source and integrated datasets to investigate the performance of our model after examining publicly available datasets. To evaluate model variants for our model, we choose two datasets – CIFAR-10 and Street View House Numbers (SVHN). The former dataset is rather small, the second is huge.

CIFAR-10 dataset consists of 50,000 images for training and 10,000 images for testing, and each image has the same size (32,32,3), which represents the height, width, and number of channels respectively. As the name indicates, there are 10 categories with 6000 images for each class.

In contrast, the Street View House Numbers dataset consists of over 600,000 images in 10 classes designed for image classification and object recognition algorithms in machine learning. Unlike CIFAR-10, it contains two formats – the original format with raw images and the cropped format where the images have been cropped and reformatted into (32,32). Even though it is better to work with raw images in our research, for simplicity and to be consistent with other datasets and models, we choose the cropped format and randomly resize the images to simulate raw images in different sizes in our model.

Performance of transformer-based architecture is greatly influenced by size of datasets, so in this research we choose two datasets with different number of inputs and make comparisons for size of datasets in models. Results in section 6 prove that performances of models are improved with larger datasets. Based on time and resources in the research, time spent in loading datasets and training models is acceptable in addition to sufficient amount of inputs for each label versus the number of labels in both datasets.

For each Vision Transformer model variant, the size of the images in both datasets is fixed, so one simply takes the dataset as input to the models after batch processing, whereas in our model, as mentioned earlier, we resize each image randomly into different sizes for simulation and then batch them. The existing fixed image size dataset also provides useful features for data processing, so we keep the dataset and process it as simply as possible to reduce the complexity of data processing and save time in experimentation.

5. EXPERIMENT

We evaluate our Flexible Transformer model with two datasets and compare it with the corresponding vision transformer model. To build the model, we use similar configurations as in the ViT models. Vision Transformer [2], includes three main variants – ViT-Base, ViT-Large and ViT-Huge based on the configuration in BERT [1]. TABLE 1 and TABLE 2, show details of the model variants for Vision Transformer and our model. As mentioned in the previous section, for each model variant, we adopt the same configuration as in the ViT counterparts but replace the layers for patches and position embed-ding with our algorithms for merging sequences of patches after linear projections. Therefore, we integrate the nature of the vision transformer with our arbitrary size image patches processing strategy for each input image as well as spatial pyramid pooling to determine the size of the output features after Transformation Encoder block.

Table 1: Configurations for Vision Transformer Model Variants

Model	Layers	Hidden Size	MLP Size	# of Heads
ViT-Base	12	768	3072	12
ViT-Large	24	1024	4096	16
ViT-Huge	32	1280	5120	16

Table 2: Configurations for Flexible Transformer Model Variants

Model	Layers	Hidden Size	MLP Size	# of Heads
Flex-Base	12	768	3072	12
Flex-Large	24	1024	4096	16
Flex-Huge	32	1280	5120	16

Tables above show all parameters for both models in detail. As mentioned before, compared to its ViT counterpart, we use multiple patch sizes for the input images in our model and add a spatial pyramid pooling layer before the MLP head.

As far as the data sets are concerned, all input images have the same size $(32, 32, 3)$. Since the fixed size of the input data is not sufficient, and Flexible Transformer is trained with input with unequal size, each input image is randomly resized by a size between 0 and 32 so that our model can process inputs with variable size in the range between 32 and 64 for width and height. Afterwards, for batch processing, each input image is padded into the same size so that all inputs are formed with the same size within the batch but different from input images in other groups of batches. On the other hand, suppose in one dataset with N inputs, given each input image with size $(w_i, h_i, 3)$, where $i = 1, 2, 3, ..., N$ and batch size B, after each input image is resized into $(w_i + w_R, h_i + h_R, 3)$ where w_R and h_R are random integers and $w_R, h_R \in [0, 32]$, then size of input images in one batch is $(w_{max}, h_{max}, 3)$ where $w_{max} \in \{w_1, w_2, w_3, ..., w_B\}$ and $h_{max} \in \{h_1, h_2, h_3, ..., h_B\}$. Furthermore, models are trained with datasets processed in different batch size and epochs.

6. RESULTS

We show typical results of models for both input datasets in this section. Details of results for experiments is shown in the table below. In this section, we choose to show the best two accuracy results for derivations in each model. As the tables shown below (TABLE 3 and TABLE 4), accuracy for each model is recorded, which is the percentage of correct predictions for each input over the total number of inputs.

Table 3: Results for Models with CIFAR-10

Model	Patch Size (Set) P	Accuracy
ViT-Base	4	64.530%
ViT-Base	8	59.560%
ViT-Large	4	64.780%
ViT-Large	8	60.320%
Flex-Base	[4, 16]	42.090%
Flex-Base	[4, 8]	41.330%
Flex-Large	[4, 8]	44.130%
Flex-Large	[4, 32]	42.990%

Table 4: Results for Models with SVHN

Model	Patch Size (Set) P	Accuracy
ViT-Base	8	90.646%
ViT-Base	4	90.104%
ViT-Large	8	90.189%
ViT-Large	4	89.417%
Flex-Base	[16]	84.104%
Flex-Base	[8]	81.961%
Flex-Large	[8]	74.804%
Flex-Large	[16]	73.110%

Due to space limitations, only best two results for model variants are shown on tables for each dataset above.

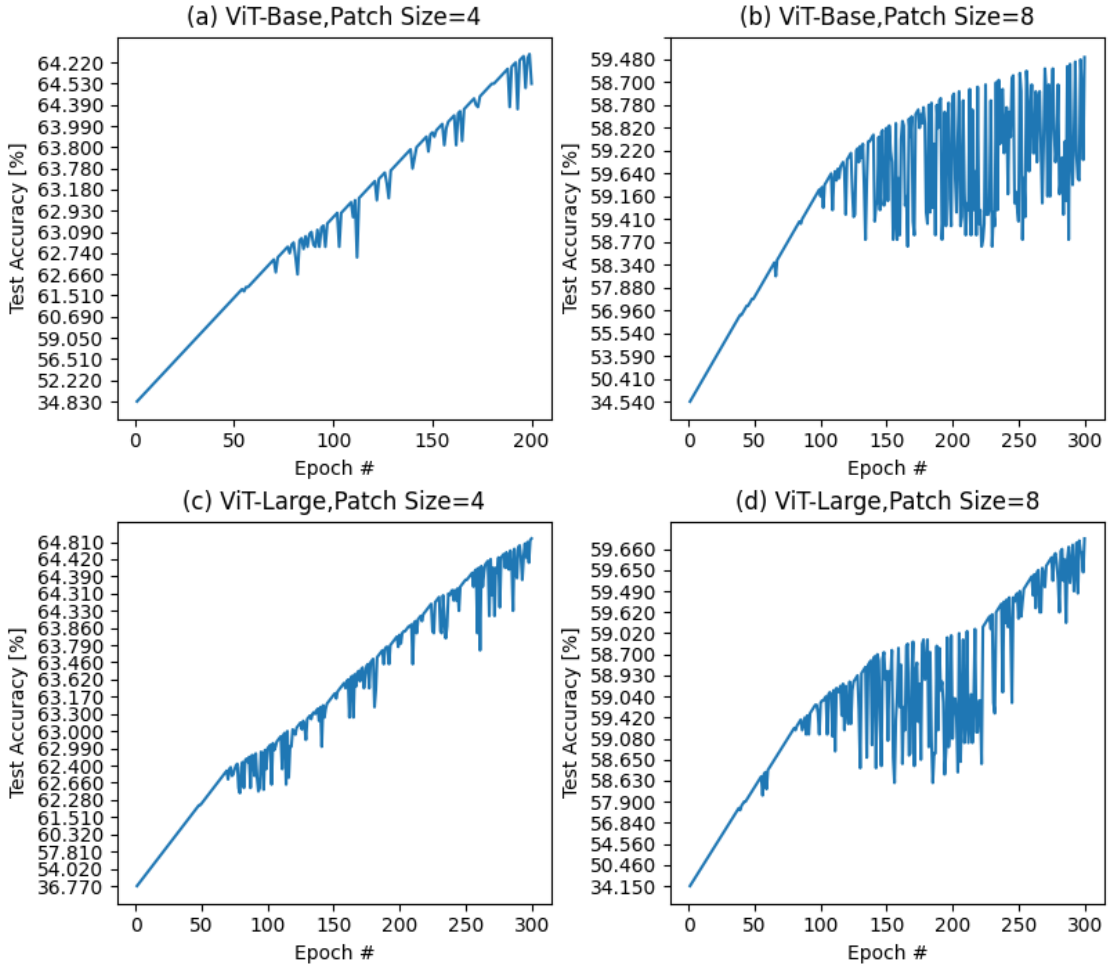

Figure 5: Vision Transformer Models with Cifar-10

6.1 Visualization

In addition to results of each model variants, performance of each epoch is also visualized for analysis of the research. In this case, we show the visualization for results implemented in experiments.

FIGURE 5-FIGURE 8 show results of test accuracy for each model variants with two datasets used in the research. Implementations of models are visualized and patch size set for model configurations are labeled in graphs. FIGURE 9 shows visualization of results for ablation study in which the number of layers is set to be 6 so that each model variant contains 6 transformer encoder blocks. Based on the results for epochs, accuracy for each model increases with number of epochs in general shown by ascending curves, even though some of them show fluctuations in large epochs. Visualizations of performances are helpful for further analysis and display which shows factors that affect performance of models.

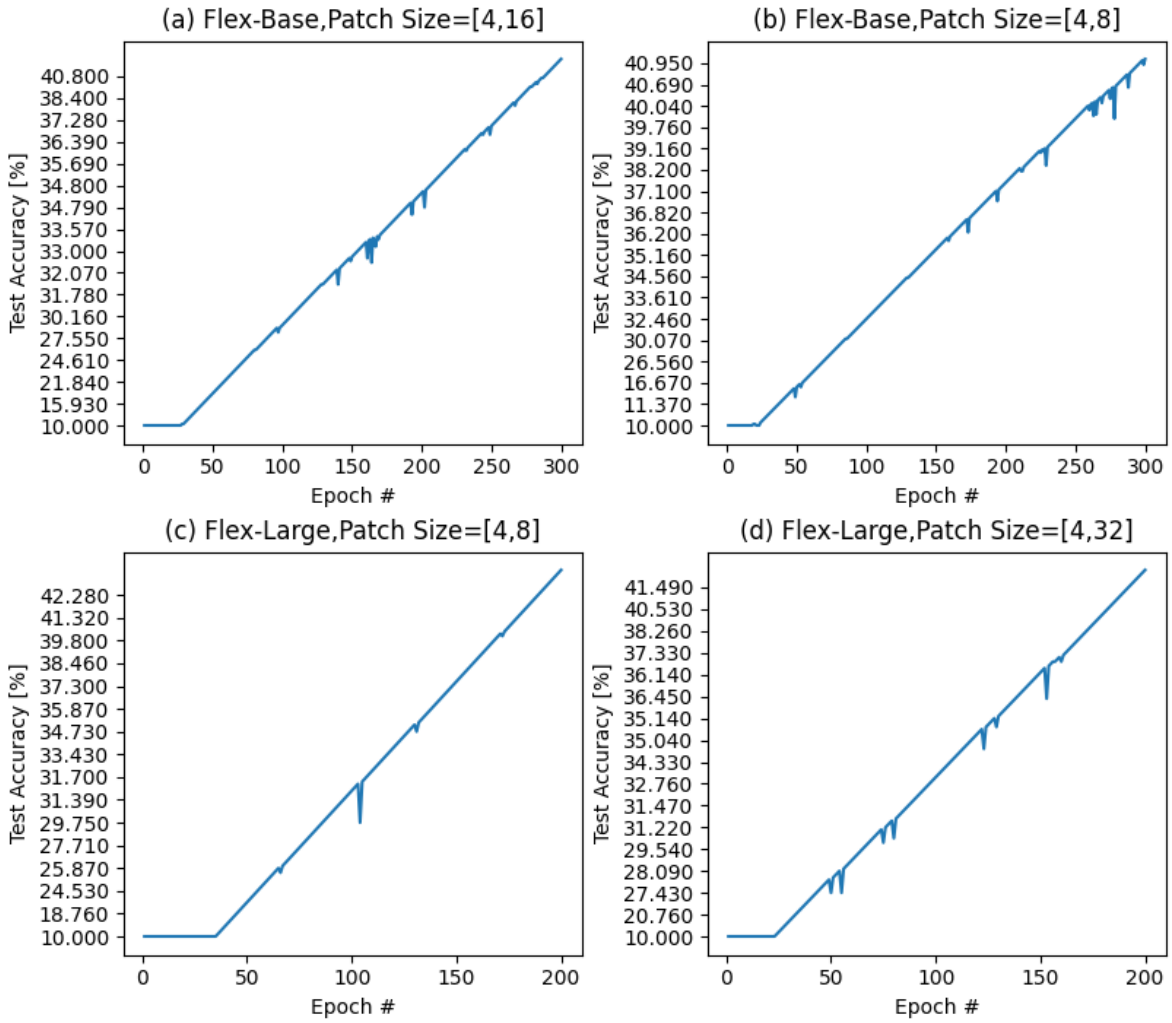

Figure 6: Flexible Transformer Models with Cifar-10

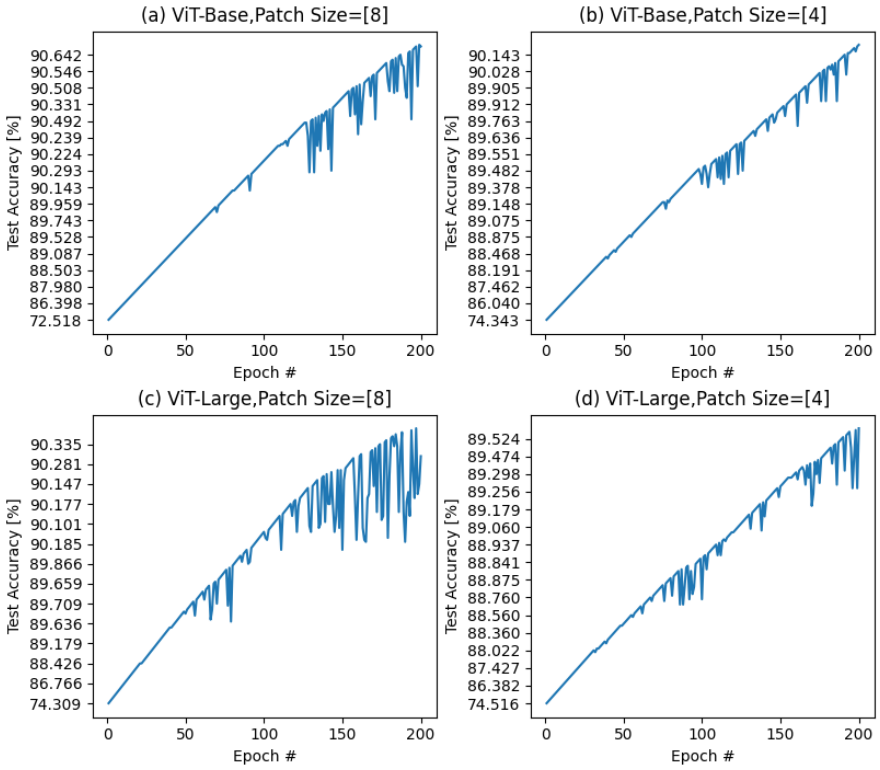

Figure 7: Vision Transformer Models with SHVN

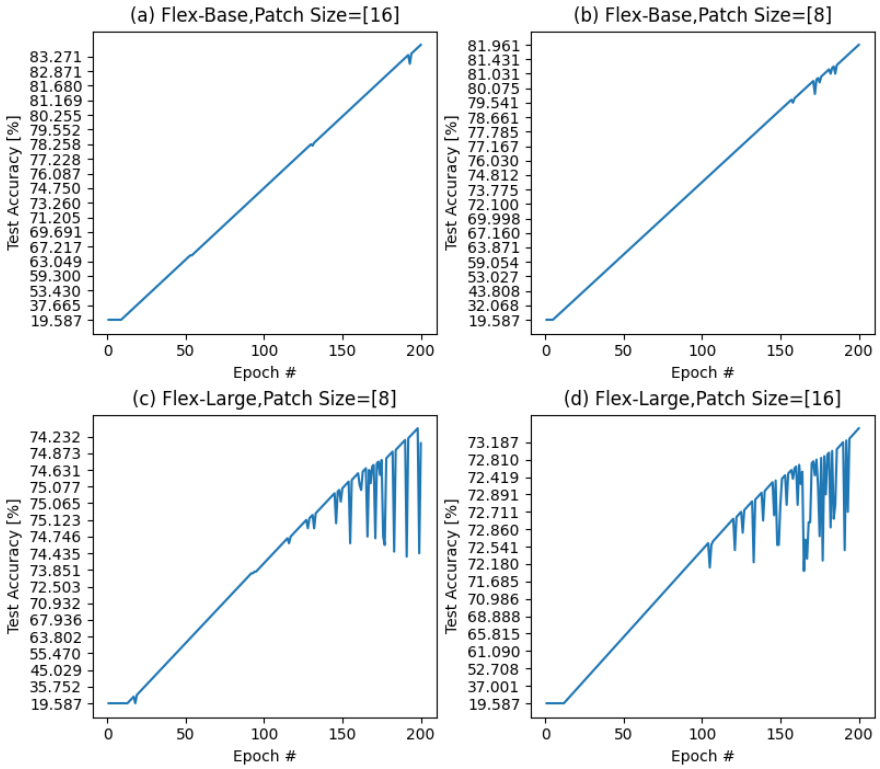

Figure 8: Flexible Transformer Models with SHVN

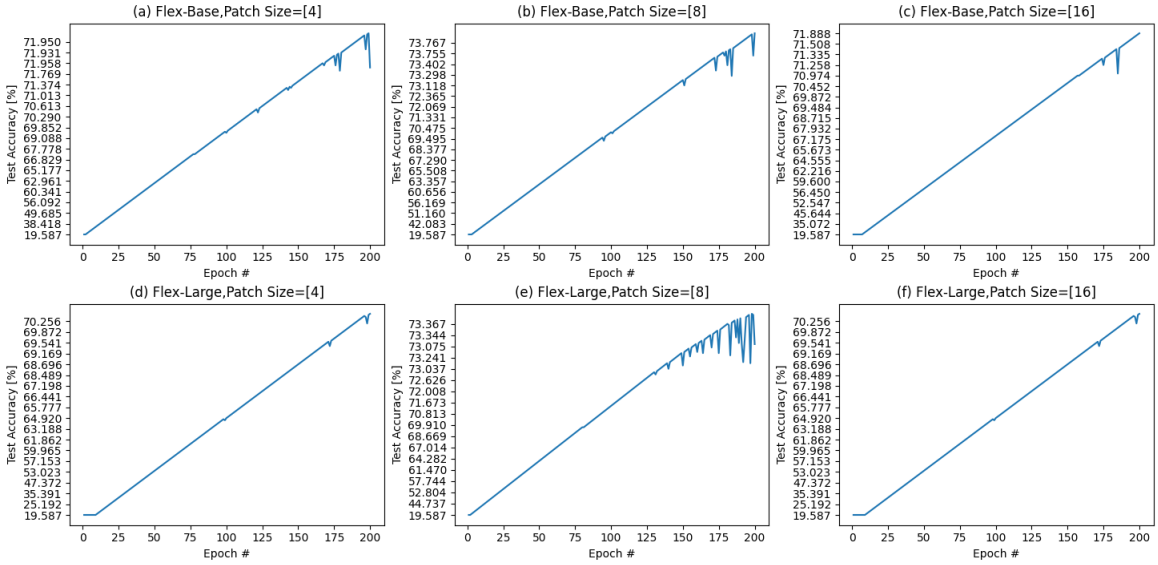

Figure 9: Ablation Study: Flexible Transformer Models with SHVN and 6 Layers

7. CONCLUSION

In conclusion, we propose a novel attention-based model for image-related tasks based on Vision Transformer. Unlike previous attention-based and CNN models for classification tasks, we instead apply a multi-spot size strategy and incorporate additional layers to deal with raw input images of arbitrary size in two different datasets without further additional modifications. The result of the research shows that our model can be as good as the current self-observation models, but it can uniquely handle tasks with arbitrary size inputs.

The research contribution provides a simple way to deal with all raw image inputs of different sizes without the need for preprocessing, such as resizing, in image-related tasks. Unlike convolutional neural networks and attention-based models, the input in our model remains intact without any preprocessing before execution, and we try to process the data with the original information. To solve the problem of limitation in the MLP layer, we apply the SPP layer with pooling so that the original input is used until the final classification. The replacement with multiple patch sizes is to ensure self-attention by better splitting the patches so that smaller patches consider each other with the features contained in a single larger patch. As mentioned earlier, the success of our model should serve as a reference for other tasks dealing with input signals of different sizes.

Architecture for our model plays a key role in the performance of image task. However, influences for other factors should not be overlooked. For instance, as stated in the section 4, selection for datasets is critical for performance of models especially for transformer-based models. In addition, deployment for pre-training is also an important factor for performance of our model, as well as important for further fine-tuning and transfer learning. In future, we would invest more in pre-training with larger datasets to discover potential for our model based on current work. Other factors such as distribution for input data and normalization methods for latent features also potentially affect eventual results of tasks.

Our model provides good and encouraging results in the research. However, there is still much room for improvement in our model, both in the datasets and in the configuration of the layers. By ablation study in the research, there will be more challenges to overcome in the future. We already conducted ablation study to explore effect of number of layers on model variants. Therefore, modification for parameters in standard architecture is also worth being explored. Due to the limited research capabilities, the performance of our model is not optimal, so it is necessary to conduct further experiments with different configurations, such as the patch group size and bin size for SPP based on datasets, to eliminate losses in training for better accuracy. Meanwhile, we try to improve the model with larger datasets of any size that better fit our model and further optimize the model by adjusting the parameters in the model and even changing the layers for better performance.

References

[1] Devlin J, Chang M, Lee K, Toutanova K. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. Minneapolis, USA. 2019:4171-4186.

[2] Dosovitskiy A, Beyer L, Kolesnikov A, Weissenborn D, Zhai X, et. al. An Image Is Worth 16×16 Words: Transformers for Image Recognition at Scale. 2021. Arxiv Preprint: https://arxiv.org/pdf/2010.11929

[3] Liu Z, Lin Y, Cao Y, Hu H, Wei Y, et. al. Swin Transformer: Hierarchical Vision Transformer Using Shifted Windows. 2021. Arxiv Preprint: https://arxiv.org/pdf/2103.14030

[4] Wang W, Chen W, Qiu Q, Chen L, Wu B, et. al. Crossformer++: A Versatile Vision Transformer Hinging on Cross-Scale Attention. 2022. Arxiv Preprint: https://arxiv.org/pdf/2303.06908

[5] Wang W, Xie E, Li X, Fan D P, Song K, et. al. Pyramid Vision Transformer: A Versatile Backbone for Dense Prediction Without Convolutions. 2021. Arxiv Preprint: https://arxiv.org/pdf/2102.12122

[6] Vaswani A, Shazeer N, Parmar N, et al. Attention Is All You Need. Proceedings of the 31st International Conference on Neural Information Processing Systems, Long Beach.2017:6000-6010

[7] Beal J, Kim E, Tzeng E, Park DH, Zhai A, et. al. Toward Transformer based Object Detection. 2021. Arxiv Preprint: https://arxiv.org/pdf/2012.09958

[8] LeCun Y, Boser B, Denker JS, Henderson D, Howard RE, et al. Back propagation Applied to Handwritten Zip Code Recognition. Neural Comput. 1989;1:541-551.

[9] LeCun Y, Bengio Y, Hinton G. Deep Learning Nature. Nature. 2015;521:436-444.

[10] LeCun Y, Bottou L, Bengio Y, Haffner P. Gradient-Based Learning Applied to Document Recognition. Proc IEEE. 1998;86:2278-2324.

[11] Gehring J, Auli M, Grangier D, Yarats D, Dauphin YN. Convolutional Sequence to Sequence Learning. presented at 34th Int. Conf. On Machine Learning (ICML). 2017;70:1243-1252.

[12] Szegedy C, Liu W, Jia Y, Sermanet P, Reed S, et al. Going deeper with convolutions. In Proceedings of the IEEE conference on computer vision and pattern recognition. 2015:1-9.

[13] Sa'idah S, Fany A, Suparta IP. Convolutional Neural Network Googlenet Architecture for Detecting the Defect Tire. International Conference on Computer Science and Software Engineering (CSASE). 2022:331-336.

[14] He K, Zhang X, Ren S, Sun J. Deep Residual Learning for Image Recognition. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR). USA. 2016:770-778.

[15] Krizhevsky A, Sutskever I, Hinton GE. ImageNet Classification With Deep Convolutional Neural Networks. Advances in neural information processing systems. 2012;25.

[16] Ioffe S, Szegedy C. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. Proceedings of the 32nd International Conference on Machine Learning, PMLR. 2015;37:448-456,

[17] Cooijmans T, Ballas N, Laurent C, Gülçehre Ç, Courville. Recurrent Batch Normalization. 5th International Conference on Learning Representations. Toulon, France. 2017:24-26.

[18] Laurent C, Pereyra G, Brakel P, Zhang Y, Bengio Y. Batch Normalized Recurrent Neural Networks. In: IEEE International Conference on Acoustics Speech and Signal Processing (ICASSP). Shanghai, China. 2016:2657-2661.

[19] Ba JL, Kiros JR, Hinton GE. Layer normalization. 2016. Available from: https://arxiv.org/pdf/1607.06450.pdf

[20] JXu J, Sun X, Zhang Z, Zhao G, Lin J. Understanding and Improving Layer Normalization. In: Wallach HM, Larochelle H, editors. Nips'19. Proceedings of the of the 33rd International Conference on Neural Information Processing Systems. 2019:32.

[21] Sun C, Shrivastava A, Singh S, Gupta A. Revisit Unreasonable Effectiveness of Data in Deep Learning Era. In: IEEE International Conference on Computer Vision (ICCV). 2017:843-852.

[22] Russakovsky O, Deng J, Su H, Krause J, Satheesh S, et. al. ImageNet Large Scale Visual Recognition Challenge. Int J Comput Vis. 2015;115:211-252.

[23] Deng J, Dong W, Socher R, Li LJ, Li K, et. al. ImageNet: A Large-Scale Hierarchical Image Database. In: IEEE Conference on Computer Vision and Pattern Recognition. Miami USA. 2009:248-255.

[24] Touvron H, Cord M, Douze M, Massa F, Sablayrolles A, et. al. Training Data-Efficient Image Transformers & Distillation Through Attention. In Proceedings of the 38th International Conference on Machine Learning. 2021:10347-10357.

[25] Abnar S, Dehghani M, Zuidema W. Transferring Inductive Biases Through Knowledge Distillation. Presented at International Conference on Learning Representations (ICLR). Vienna Austria [online]. 2021. Arxiv Preprint: https://arxiv.org/pdf/2006.00555

[26] Wei L, Xiao A, Xie L, Zhang X, Chen X, et al. Circumventing outliers of autoaugment with knowledge distillation. In European Conference on Computer Vision. Springer International Publishing. 2020:608-625.

[27] Cho JH, Hariharan B. On the Efficacy of Knowledge Distillation. In Proceedings of the IEEE/CVF International Conference on Computer Vision. Seoul, South Korea. 2019:4794-4802.

[28] Arnab A, Dehghani M, Heigold G, Sun C, Lucic M, et. al. Vivit: A Video Vision Transformer. In Proceeding of IEEE/CVF International Conference on Computer Vision (ICCV) [online].2021:6816-6826.

[29] Lu Y, Fu J, Li X, Zhou W, Liu S, et. al. Rtn: Reinforced Transformer Network for Coronary CT Angiography Vessel-level Image Quality Assessment. In International Conference on Medical Image Computing and Computer-Assisted Intervention Singapore. 2022:664-653.

[30] Wang Y, Li Y, Wang G, Liu X. Multi-Scale Attention Network for Single Image Super-resolution. In: Proceedings of the of 2024 IEEE/CVF conference on computer vision and pattern recognition. 2024:5950-5960.

[31] Yan W, Sun Y, Yue G, Zhou W, Liu H. Fviformer: Flow-Guided Global-Local Aggregation Transformer Network for Video Inpainting. IEEE J Emerg Sel Topics Circuits Syst. 2024;14:235-244.

[32] Cubuk ED, Zoph B, Mane D, Vasudevan V, Le QV. Autoaugment: Learning Augmentation Strategies From Data. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2019:113-123.

[33] Cubuk ED, Zoph B, Shlens J, Le QV. Randaugment: Practical Automated Data Augmentation With a Reduced Search Space. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops. 2020:3008-3017.

[34] Ulyanov D, Vedaldi A, Lempitsky V. Instance Normalization: The Missing Ingredient for Fast Stylization. ArXiv preprint: https://arxiv.org/pdf/1607.08022.pdf

[35] Huang X, Belongie S. Arbitrary Style Transfer in Real-Time With Adaptive Instance Normalization. In: IEEE International Conference on Computer Vision (ICCV). 2017:1510-1519.

[36] Salimans T, Kingma DP. Weight Normalization: A Simple Reparameterization to Accelerate Training of Deep Neural Networks. In: 30th Conference on Neural Information Processing Systems. Barcelona, Spain: NIPS. 2016:901-909.

[37] Zhang J, Karimireddy SP, Veit A, Kim S, Reddi S, et. al. Why Are Adaptive Methods Good for Attention Models? In 34th Conference of Neural Information Processing Systems. 2020:15383-15393.

[38] Gastaldi X. Shake-Shake Regularizations of 3 Branch Residual Networks. In: 5th International Conference on Learning Representations. 2017.