ILLINOIS INSTITUTE OF TECHNOLOGY

Department of Computer Science

CSP-577 Deep Learning

Image classification using hierarchical based shifted window Vision Transformers.

Naveen Raju Sreerama Raju Govinda Raju

nsreeramarajugovinda@hawk.iit.edu

A20516868

Raghunath Babu

rbabu@hawk.iit.edu

A20511598

Table of Contents

1. Abstract

This project presents a new variant of Vision Transformer with Shifted Window phenomenon and several optimizations, that will capably address all tasks in computer vision like classification, object detection and segmentation. The predominant traditional vision transformers adapted from Natural Language Processing tasks had a lot of differences in adjusting for high resolution pixels of images compared to the language domain. To tackle this problem in the vision domain, here a hierarchical transformer is implemented bringing in the Shifted window phenomenon. This shifted window has a great advantage in computing self attention to non overlapping local windows and in deeper layers a global context is established by cross window connection. This gives a flexibility to the model to handle input images at various scales thereby decreasing to linear computational complexity from quadratic.

2. Problem Statement

The motivation of this project is to handle a predominant adversary in the field of computer vision in handling high dimensional images with poor image quality and thereby preserving the computational efficiency. Prominent approaches with Convolutional Neural Networks(CNNs), Deep learning networks, RNNs, LSTMs and state of the art Vision Transformers were leveraged for this use case. The problem they faced with CNNs, DNNs, RNNS etc were, there was a need to add several layers to support the dimension of the input. This increased the parameters of the model, resulting in a burden to computation which were considered unnecessary. Also in applications with huge datasets CNNs failed to capture long range dependencies and global context. When Vision Transformers were used, it helped in capturing global dependencies within an image, but with huge computation efforts.

During the initial processing of the high dimensional images, where converting the images to image patches for tokenization, this vision transformer resulted in a lot of key, value and queries and performing self attention demanded a lot of computation with huge efforts in training. The model resulting would be too complex and would be practically difficult to host and perform api calls. This boils down to our approach where we want to get the underlying global dependencies in an image and minimize the computation.

In this project we are focused on implementing a hierarchy vision transformer where we define windows as a collection of image patches(similar to the regular vision transformer) and this window size will be hierarchically decreased. Another advantage in the proposed approach is that, We can leverage several architectures like Feature pyramid network(FPN), U-Net etc. The self attention would be performed within the local window and the window would be shifted within the images subsequently. A cyclic shifted window technique(resulting in non overlapping windows) is implemented to capture the global context. Analyzing this, we could infer that the properties of the Vision transformers are preserved and quadratically decreasing computational complexity.

3. Dataset Description

Dataset named Food -101 is considered for our project
Source - https://data.vision.ee.ethz.ch/cvl/datasets_extra/food-101/

Data set size : -
Training images per class : 960
Testing images per class : 40

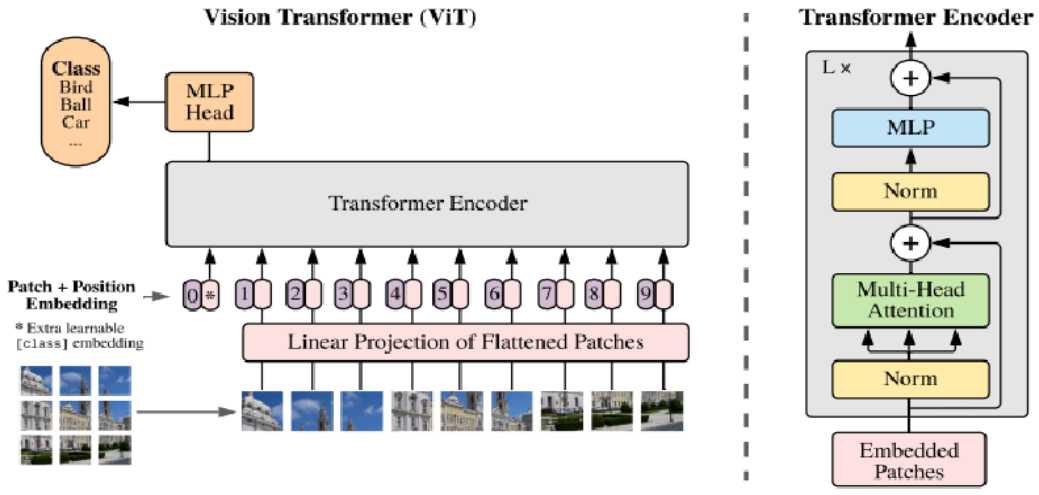
Data set classes : -
Garlic bread, Hot Dog, Ice Cream, Omelette, Pizza

Sample Images :

Garlic bread	Hot Dog	Ice Cream	Omelette	Pizza
				

4. Proposed Solution

a) Vision Transformer



Vision Transformer architecture general overview:

Patch Embedding : First it divides the image into fixed sized non overlapping patches, following which it linearly embeds it individually.

Position Embedding : To the patch embedding it will add position embedding to it to maintain spatial relationship in an image. This is done because the vision transformer processes all patches parallelly instead of sequentially.

Class token : class token is appended to embeddings, this is used to represent global information of the entire image.

Transformer encoder Block : Vision transformer will have stacks of encoder layers. Each encoder will have a normalization layer followed by Muti-Head Attention, Normalisation and Multi Layer Perceptron.

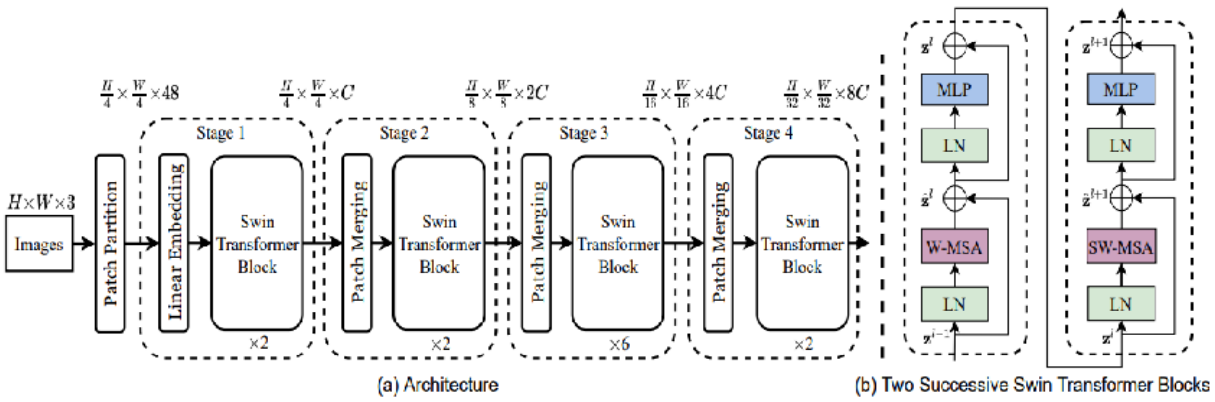
Multi Layer Perceptron : Also called as feed forward layer it is used to refine the features obtained from multi head self attention networks. It introduces non linearity to the model which allows the model to learn complex patterns and relationship of features.

Classification Head : A fully connected layer is added at last. This layer will transform features into class probabilities.

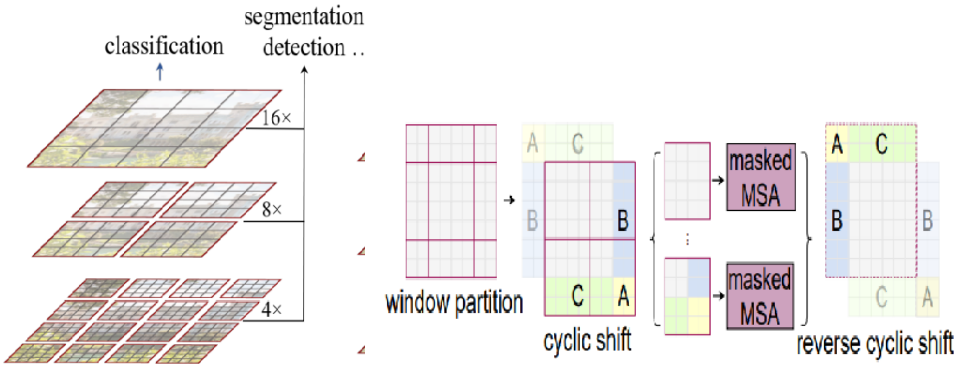
Self Attention : It is the mechanism which will capture relations between different patches in image patches. Since Vision transformer takes all patches in parallel for processing, self attention is required for capturing interaction between these patches, learning long and short range dependency. Self attention is performed using query, key, value vectors dot products and their weighted sum of values.

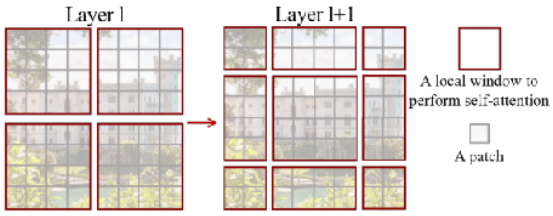
Multi head attention : is extension of self attention. Instead of using single self attention, here multiple self attention is used in parallel. Where each self attention focuses on different aspects of image. Advantages of multi head self attention are: capturing different representations, increased model capacity, robustness for model generalisation, increased parallelisation and efficiency.

b) Swin transformer



In a high level architecture of the swin transformer, it constructs a hierarchical representation by starting from small patches and gradually merging neighbor patches in deeper transformer layers. In this project, a 4 block Swin transformer with successive transformers with regular Masked Self Attention(MSA) and Shifted Window Masked Self Attention is implemented.
The two major phenomena of Swin transformer is patch merging and shifted window.
Patch merging is merging neighboring patches as we go deeper into the transformer layer thereby creating a hierarchical feature map. In this module before every swin transformer block(totally 4), there is a patch merging operation taking place.





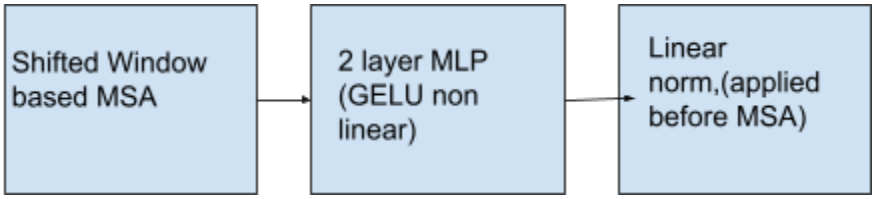
Initially after breaking down the image into patches a chunk of patches in square fashion is considered as a window. Unlike the VIT here Self Attention is performed within the window and not in the entire image. As moving through the subsequent transformer block the windows are shuffled within the image.
Two major shuffling approaches were experimented, the shifted window and the sliding window. Sliding window is just the displacement of windows and in a shifted window, the displacement happens along with the cyclic shift, leading to lower latency as shown in the above figure(when patch A comes out of the window's dimension it is diagonally appended). In experimentation it is found that Shifted Window has an improved performance over sliding window.
Upon this a Masked Self Attention is performed within the window. This results in using the same key for all the query patches in the window.

Through this modeling approach, the global and the local context is achieved and making its performance equivalent to the prominent Vision Transformers. Secondly it reduces the quadratic time complexity of ViT to linear time complexity.

$$\Omega(\text{MSA}) = 4hwC^2 + 2(hw)^2C,$$
$$\Omega(\text{W-MSA}) = 4hwC^2 + 2M^2hwC,$$

After the Shifted Window based Masked Self Attention, a 2 layer Multi Layer perceptron is followed with a non linear GELU Activation function. The reason for GELU is to address smoothness and continuity to ease algorithms like stochastic gradient descent. Secondly it also takes care of the normalization property to mitigate the problem of vanishing gradient.

For the initial positional embedding, relative positional embedding is used. It is learnt by the distance between the tokens where the bias in it tries to capture the feature relevance based on the distance parameter.



5. Training parameters

a) **Vision Transforme and Swin Transformer**:

Input size : 224*224*3

Number of transformer layers/blocks = 12

Number of head = 12
This is the count of multi attention heads in each encode block

Patch size = 16
Size of each non overlapping patch of an image.

Embedding dimension = 768
Given patch dimension height = 16 and width =16, it is of 3 dimensions. Therefore, 16*16*3 = 768

Multi layer perceptron size = 3072
Here we have chosen 3072 because in the MLP network we expand the embedding dimension and bring it back to the original dimension. W increase embedding dimension by 4 times i.e 768*3 = 3072

MLP dropout = 0.1, 0.2
This dropout is used in multi layer perceptrons. Here 0.1 is used initially and later on 0,2 is used to avoid overfitting and to generalize well.
attn_dropout = 0, 0.2
This is used to self attention layer usually 0 is used according to paper but in later stages of training 0.2 was used to mitigate overfitting

Embedding dropout = 0.1, 0.2

This is used after creating embedding of each patch of image. This is used to make models give equal importance to overall embedding rather than strongly depending on certain embedding vectors.

Optimiser: Adam, AdamW
Adam W was used after experimenting with Adam.

In Adam optimiser weight decay is often applied to both parameter updating as well as parameters themselves, which might lead to suboptimal behavior. But AdamW will separate these two components, which will result in more accurate weight decay.

Weight decay = 0.3, 0.001,0.03
Various weight decay parameters were used while training based on underfitting and overfitting situations. Weight decay is used for regularization in optimisers. This is used to prevent overfitting during training to penalize large weights.

Initial learning rate = 0.001,0.0001
Initially higher learning rate was used and later stages smaller learning rates were used so as to gradually reach global minimum.

Cosine Annealing learning rate : min=0.0001

Cosine annealing learning rate is a learning rate scheduler used during training models. It smoothly varies the learning rate between a maximum and minimum value over the course of training, following a cosine-shaped curve.

Step learning rate : Where at a certain frequency of epochs the learning rate is decayed by se factor, Which was not as efficient as Cosine annealing learning rate.

Label smoothening = 0.3
This helps in overcoming overfitting. This prevents model by preventing model in becoming too confident on training data. Instead of using hard labels like 0 and 1 probability to classes, label smoothening introduces a small amount of uncertainty by using a smoothened label distribution.

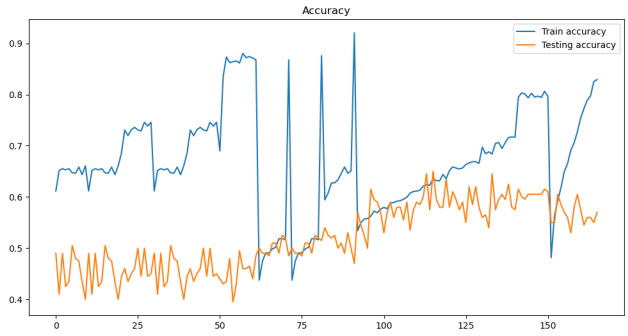
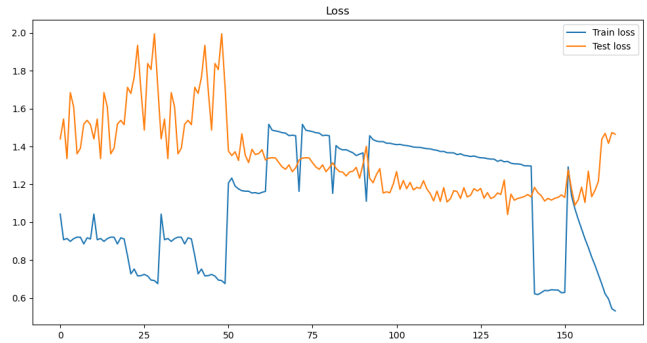
Normalize (mean=[0.556, 0.447, 0.335], std=[0.231, 0.242, 0.238])
Normalizing all color channels in images is used for balancing scale, enhancing convergence speed, enabling numerical stability, and also increasing generalization during training.

Augmentation:
Augmentation is used to improve the robustness, generalization, and performance of during training by exposing them to a broader range of variations.

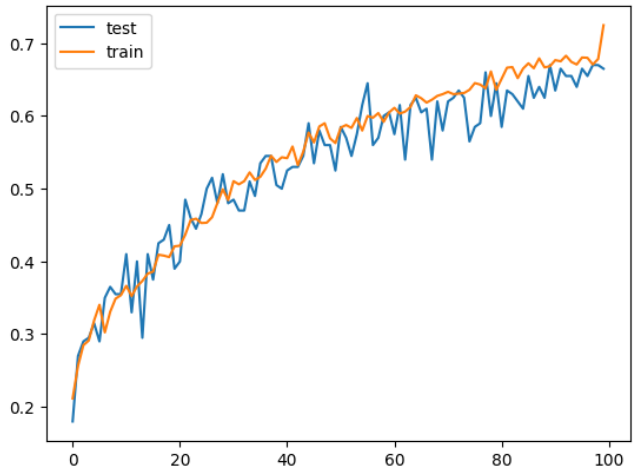
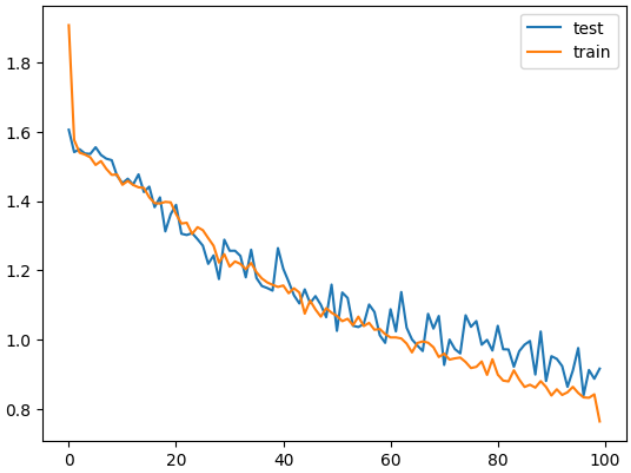
Augmentation used are Random Horizontal Flip, Random Vertical Flip, ColorJitter (brightness=0.2, contrast=0.15, saturation=0.1, hue=0.1), Random Affine (degrees=0, translate=(0.1, 0.1), scale=(0.8, 1.2), Random Perspective (distortion_scale=0.2, p=0.5), Gaussian Blur (kernel =5), Gaussian Noise

6. Results and Evaluation Metrics

Vision transformer

Train and Test accuracy	Train and Test Loss
	

Swin Transformer

Train and Test accuracy	Train and Test Loss
	

Train Data Metrics(in percentage)

	Metric	Garlic bread	Hot Dog	Ice Cream	Omelette	Pizza
Vision transformer	Precision	78.63	76.8	79.65	80.54	88.22
	Recall	77.08	76.77	86.87	80.625	81.97
	F1-Score	77.85	76.81	83.10	80.58	84.98
Swin	Precision	100	82.3	77.2	81.8	82.6

Transformer	Recall	80	70	85	90	95
	F1-Score	88.88	75.6	80.9	85.7	88.3

Vision Transformer overall accuracy - 80.66 %
Swin Transformer overall accuracy - 84 %

Test Data Metrics(in percentage)

	Metric	**Garlic bread**	**Hot Dog**	**Ice Cream**	**Omelette**	**Pizza**
Vision transformer	Precision	60.97	60.6	66.6	50	67.64
	Recall	62.5	50	70	62.5	57.5
	F1-Score	61.72	57.7	68.2	55.5	62.1
Swin Transformer	Precision	70.3	68.7	86.6	73.3	74
	Recall	95	55	65	55	1
	F1-Score	80.8	61.1	74.2	62.8	85.1

Vision Transformer overall accuracy - 60.5 %
Swin Transformer overall accuracy - 84 %

Model Parameters comparison

	Total Parameters	**Trainable parameters**
Vision transformer	85,802,501	27,499,877
Swin Transformer	85,802,501	27,499,877

Therefore, it is evident that Swin Transformer, with 32 percent of the total parameters of ViT, was able to outperform efficiently

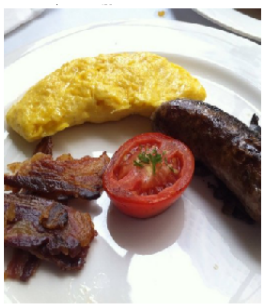
Sample result images:

a) **Correct Predictions**

Ground Truth : Garlic Bread **Prediction : Garlic**	**Ground Truth : Hot Dog** **Prediction : Hot**	**Ground Truth : Ice cream** **Prediction : Ice**	**Ground Truth : Omelette** **Prediction :**	**Ground Truth : Pizza** **Prediction : Pizza**

Bread	**Dog**	**cream**	**Omelette**	
				

b) **Incorrect predictions:**

Ground Truth : Garlic Bread **Prediction : Garlic Bread**	**Ground Truth : Hot Dog** **Prediction : Hot Dog**	**Ground Truth : Ice cream** **Prediction : Ice cream**	**Ground Truth : Omelet** **Prediction : Omelet**	**Ground Truth : Pizza** **Prediction : Pizza**
				

7. Conclusion

- In a nutshell, the Swin transformer achieves a better performance than Vision Transformer(ViT) because of the following reasons: Hierarchical Processing with Shifted Windows, Improved Long-Range Dependency Handling.
- Swin transformer preserves computational and memory footprint with just 45 percent of parameters in ViT.
  Vision Transformer: Total params: 85,802,501, Trainable params: 85,802,501
  Swin Transformer: Total params: 27,499,877, Trainable params: 27,499,877
- The data set that we considered was the most challenging dataset for classification tasks as there is lot of intra class variability, as we can observe in above sample images of incorrect predictions that an omelet with tomato topping looks like a pizza, hot dog kind of food along with omelet gets predicted as hotdog instead of omelet, a zoomed image of garlic bread looks like a pizza, lot of brown onion topping upon small hot dog piece look like a garlic bread and so on. Hence the model was not able to get extremely high accuracies because of the above said reasons.

8. Future works

- We can use the incremental learning strategy to train the model to make the model generalize well by capturing fine grained features efficiently.
- Train multi label classification where the model predicts multiple class names if multiple foods category appears in image.
- Use hybrid architecture involving both CNN and SWIN Transformer.
- In addition to that, the swin transformer can efficiently accommodate architectures like UNET, Feature pyramid networks(FPN), making it suitable for all types of Computer Vision applications like Classification, Segmentation and Detection. The key highlights of Swin transformer is to handle images with high resolutions with various scales.

.

9. Lesson learnt from this project

- We did not use hugging face libraries instead we used code from scratch where we were able to understand every line of architecture code. Due to that we did not do transfer learning from imagenet weights as our architecture style did not match the pretrained imagenet weights.
- Due to that we face underfitting initially after solving that we face overfitting. Finally we were able to mitigate that as well.
- Learned how to set hyperparameters and parameters during the training process to handle underfitting and overfitting.
- Understood how to preprocess the images so that the model generalizes better during the training process.
- How Vision Transformer and Swin Transformer is better than traditional deep learning algorithms
- Advantages of transformer architecture : Addressing the issue of effectively capturing long-range dependency in images, global context understanding, flexibility in input images.
- Advantages of Swin transformer compared to Vision Transformer :Hierarchical Processing with Shifted Windows, Reduced computational complexity, Improved Long-Range Dependency Handling.

10. Contribution

Selecting the motivation for the project and choosing the data set was a shared responsibility.
After selecting this, a literature survey was conducted equally between both team members and spotted the scope in Vision Transformer and Swin Transformer. Both worked on inferring the results.

Naveen Raju - Dataset preparation, implemented custom Vision transformer, modeling, fine tuning of parameters and hyperparameters, metrics generation and also analyzed and decoded Swin Transformer code.

Raghunath - implemented ad hoc Swin transformer, modeling, fine tuned parameters, metrics generation and also analyzed and decoded Vision Transformer code.

11. Bibliography

1. D. Kang, P. Koniusz, M. Cho and N. Murray, "Distilling Self-Supervised Vision Transformers for Weakly-Supervised Few-Shot Classification & Segmentation," in 2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Vancouver, BC, Canada, 2023 pp. 19627-19638.

2. Z. Liu, et al., "Swin Transformer: Hierarchical Vision Transformer using Shifted Windows," in 2021 IEEE/CVF International Conference on Computer Vision (ICCV), Montreal, QC, Canada, 2021 pp. 9992-10002.
doi: 10.1109/ICCV48922.2021.00986

3. Agilandeeswari, L., Meena, S.D. SWIN transformer based contrastive self-supervised learning for animal detection and classification. Multimed Tools Appl 82, 10445–10470 (2023).

4. Alexander Kolesnikov and Alexey Dosovitskiy and Dirk Weissenborn and Georg Heigold and Jakob Uszkoreit and Lucas Beyer and Matthias Minderer and Mostafa Dehghani and Neil Houlsby and Sylvain Gelly and Thomas Unterthiner and Xiaohua Zhai.," An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale",

5. https://huggingface.co/docs/transformers/model_doc/vit (reference)

6. https://huggingface.co/docs/transformers/model_doc/swin (reference)