Exploring Vision Transformers through XAI methods

Deep Learning - Final Project - 2024

Nejc Ločičnik, 63180183, nl4952@student.uni-lj.si

I. Introduction

This project is exploratory in nature. My goal was more to gain familiarity with how the attention mechanism works, specifically its application in computer vision tasks. I thought that simply training some models wouldn't be sufficient for developing an intuition about how everything works, so I focused on explainable AI (XAI) methods, primarily visualizations of attention matrices. This approach required me to dissect a model using PyTorch's (forward) hook feature and to rebuild exactly what happens at specific layers in the model. While I included some fine-tuning at the end, it should be considered a bonus (to at least engage in some training) rather than a major part of this project.

II. Related Work

The model architectures explored in this project are the Vision Transformer and the Swin Transformer, both of which adapt the base Transformer model for computer vision tasks [1]. This section provides an overview of these architectures and the technique used for visualizing attention.

A. Vision Transformer

The Vision Transformer (ViT) adapts the Transformer model, originally designed for sequential data, to handle images. It does so by dividing the input image into fixed-size patches (typically 16x16 pixels), which are then linearly projected into embeddings. Additionally, ViT incorporates a CLS token (introduced with BERT) to aggregate image representations throughout the model [2].

B. Swin Transformer

The Swin Transformer (Shifted Window Transformer) enhances the Transformer architecture's efficiency for computer vision tasks. It divides images into non-overlapping patches but introduces a hierarchical structure that progressively merges these patches, enabling the model to capture both local and global features. A key innovation is the shifted window mechanism, which applies self-attention within small, shifting windows across layers. This approach improves the model's ability to handle long-range dependencies while maintaining computational efficiency, making Swin Transformer scalable across various vision tasks [3].

C. Attention Rollout

Attention Rollout is a technique for visualizing and understanding how Transformers, such as ViT, make decisions by aggregating attention weights across all layers. This method rolls out attention weights from each layer, revealing the influence of different parts of the input (e.g., image patches) on the final output. It provides insights into which regions of an image are most influential in the decision-making process, helping to interpret the inner workings of Transformer-based models [4].

III. Experiments & Evaluation

This section explores the attention mechanisms of Vision Transformers (ViT) and Swin Transformers, focusing on the distribution of attention across different layers and heads. We use techniques such as Attention Rollout to quantify information flow and analyze how these attention patterns contribute to feature extraction and overall model understanding.

A. Attention Mechanism in Transformers

Vision Transformers (ViT) and Swin Transformers are composed of multiple encoding blocks. Each block in these models includes:

1) Several attention heads that are responsible for fusing information from different patches in the image.
2) A Multi-Layer Perceptron (MLP) that transforms each patch representation into a higher-level feature representation.
3) Residual connections that facilitate learning by allowing gradients to flow through the network.

In ViT, each of the 12 encoder blocks contains 12 attention heads. In contrast, Swin Transformers have varying numbers of attention heads depending on the hierarchical layer: [4, 8, 16, 32], with each of the 24 Swin Blocks paired (windowed self-attention followed by shifted-windowed self-attention), effectively resulting in 12 encoder blocks.

Attention heads in both models are calculated using the Query (Q), Key (K), and Value (V) matrices. Each of these matrices has a shape of $N_{TOKENS} \times d_{HIDDEN}$. The attention matrix is derived using the formula $softmax(\frac{QK^T}{\sqrt{d_{HIDDEN}}})$, which results in an $N_{TOKENS} \times N_{TOKENS}$ matrix indicating how each token (row) should attend to other tokens. The final token representation is obtained by multiplying the attention matrix with the V (value) matrix.

The dot-product QK^T measures the similarity between the query q_i of token i and the key k_j of token j. If the dot-product is positive, it means the token at location j contributes information to the token at location i. Conversely, if it is negative, the information flow is reduced. Summing across all channels provides the similarity score in the attention matrix A_{ij}.

Scaling the dot-product by $\sqrt{d_{HIDDEN}}$ helps to mitigate issues with softmax skewing caused by large values, but I see it more as a band-aid solution. Newer models, such as Swin Transformer V2, use alternative approaches like cosine similarity to address this issue more effectively.

B. Obtaining attention

To extract attention information, a forward hook must be registered for each encoder block to capture the input and output of specific modules, such as self-attention. The approach for retrieving this information varies depending on the model's implementation. In this instance, I extracted the input from each encoder block and manually computed the Query (Q), Key (K), and Value (V) matrices using the module's weights and biases. Subsequently, I calculated the attention matrix by applying the scaled dot-product followed by the softmax operation. These components of each encoder block were saved and computed during the forward pass of the model on an input image.

C. *Visualizing attention in ViT*

Figure 1 visualizes the attention maps of each head within each Encoder block for the Vision Transformer (ViT) using the same sample image as in Figure 2.

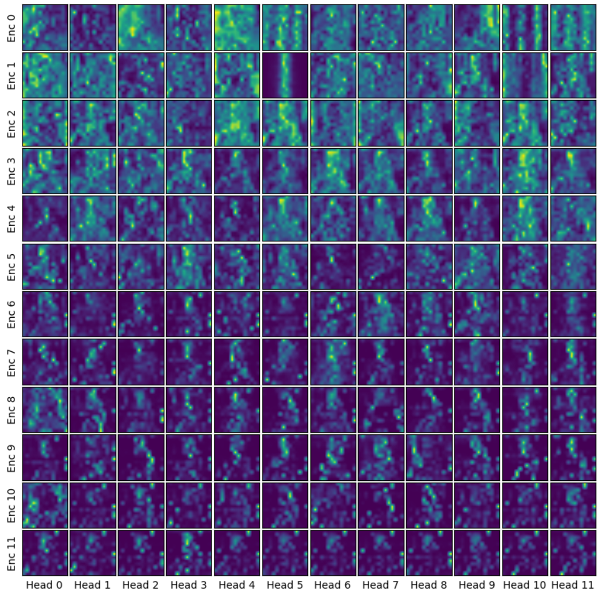

Figure 1. Visualization of attention maps for individual heads and encoders in the base ViT.

The attention heads in the initial encoder blocks exhibit interpretable patterns. For instance, in encoder 0, attention heads 2 and 9 split the image into distinct regions, with head 2 focusing on the left side and head 9 on the right side. Encoder 0 generally features "column-like" attention maps. Encoder 1 shows more refined attention patterns; head 5 focuses on the central column of the image, while head 10 emphasizes the left and right edges. This suggests that early encoders primarily perform spatial attention, helping the model to determine the importance of different image areas, unlike CNNs that capture low-level features such as edges and textures due to their smaller kernel sizes (e.g., 3x3, 5x5). ViT's attention operates on 16x16 patches, offering a coarser level of detail.

As we progress to later encoders, attention maps begin to focus more on specific features. For example, in encoder 3, head 4 attends to the heads of both the dog and the cat, with a stronger focus on the dog's head. However, attention maps in subsequent layers become increasingly sparse and harder to interpret due to the scaling and softmax operations. Tokens with extreme values in the attention matrix are squeezed towards 0 and 1 by the softmax, leading to dark rows with only a few highlighted patches. This results in later layers exhibiting similar attention patterns.

Additionally, residual connections are evident in the attention maps. For instance, in head 0 from encoder 7 onwards, the dog's head is attended to, then partially lost in the subsequent encoder, before reappearing in later layers and repeating this pattern through to the final encoder block.

D. *Visualizing Information Flow in ViT*

The images in Figure 1 illustrate individual attention activations, but they don't reveal how attention flows throughout the Transformer from start to finish. To quantify this flow, we use a technique called Attention Rollout, as described by S. Abnar and W. Zuidema in Quantifying Attention Flow [4].

At each Transformer block, we obtain an attention matrix A_{ij}, which indicates how much attention flows from token j in the previous layer to token i in the next layer. By multiplying these matrices between successive layers, we can calculate the total attention flow across the network. However, we must also account for residual connections by adding the identity matrix I to each layer's attention matrix: $A_{ij} + I$. The challenge lies in handling multiple attention heads; the authors suggest averaging them, though other approaches like taking the minimum or maximum could also be explored.

The Attention Rollout matrix at layer L can be recursively computed using the following formula:

$$AttentionR_L = (A_L + I)AttentionR_{L-1}$$

While normalizing the rows to maintain a total attention flow of 1 might seem logical, I found that skipping normalization produced better, less noisy results. Additionally, I excluded the first encoder block from the recursive computation, as it introduced significant noise. Another crucial improvement was discarding the lowest attention values, with the best results achieved by discarding the bottom 90% of attention values after head fusion.

Figure 2 presents the Attention Rollout for the CLS token's representation, comparing the original image with results from mean head fusion and max head fusion.

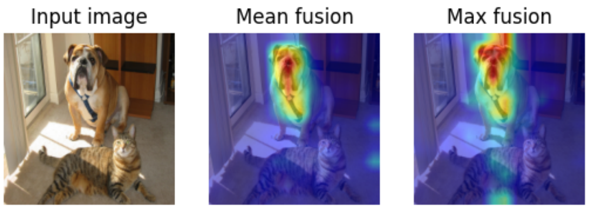

Figure 2. Visualizing attention rollout with mean and max head fusions.

Mean fusion of heads tends to produce the most accurate results, so let's examine how Attention Rollout evolves across each encoder block (excluding the first) as shown in Figure 3.

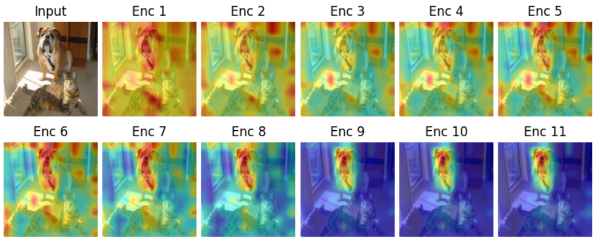

Figure 3. Visualizing attention rollout at different encoder blocks with mean head fusion.

E. *Visualizing attention in Swin Transformer*

Visualizing attention in the Swin Transformer presents unique challenges since it lacks a CLS token to aggregate attention scores across all patches. Additionally, the shifted-window self-attention complicates matters by rolling the input and then unrolling it after the attention scores are multiplied with the value matrix (dimension: $B \times H \times W \times C$). Although I initially aimed to unroll the attention matrix (dimension: $Windows \times Heads \times N_{patches} \times N_{patches}$). I encountered difficulties due to my limited experience with tensor transformations. Instead, I focused on visualizing the attention maps of windowed self-attention, which is relatively straightforward. Without a CLS token, I examined the attention matrices of specific patches within a chosen window before recombining them into a complete image.

The Swin Transformer is organized into four stages, with the following number of Swin Blocks per stage: [2, 2, 18, 2]. These blocks are paired, with the first block utilizing normal windowed self-attention and the second using shifted-window self-attention. Stages 1, 2, and 4 contain a single pair, while stage 3, responsible for most of the image modeling, contains nine sequential block pairs.

Each stage is followed by patch merging, so each stage represents a different level in the spatial hierarchy. I found that the first windowed self-attention block in the 3rd stage the most interesting. Figure 4 shows some samples, where stage 3 has four windows (separated by red lines) and 16 heads. The small red squares indicate the patch whose attention is visualized.

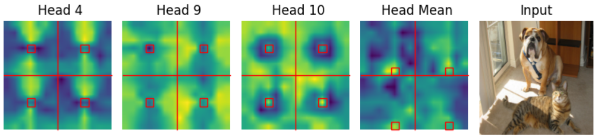

Figure 4. Visualization of a few heads and their mean for the first Swin Block in 3rd stage of Swin Transformer.

Head 4 seems to direct patches to attend to whatever is above and below them, forming a cone-like pattern. In contrast, Head 9 appears to focus attention horizontally, making patches attend to what is on either side of them. Head 10 exhibits a different behavior, with patches attending to their distant surroundings while suppressing attention to their immediate vicinity. The final image shows the average attention across all 16 heads, where the outline of the dog becomes discernible in the upper two windows.

The focus of attention mechanisms on spatial relations versus actual features can vary depending on the size of the patches used. For instance, visualizing the attention from the first Swin block with patches of size 4x4 revealed patterns resembling edge detectors, emphasizing the model's sensitivity to fine spatial details.

F. Model Interpretation with LIME

In this section, we employ LIME (Local Interpretable Model-Agnostic Explanations) to gain insights into model predictions by analyzing the impact of different image regions on classification outcomes. We explore how variations in image location and size influence predictions, revealing insights into model biases and the effects of positional embeddings.

Since I couldn't fully visualize attention for the Swin Transformer or produce an Attention Rollout, I turned to a more universal and model-agnostic method: LIME (Local Interpretable Model-Agnostic Explanations) [5]. LIME treats the model as a black box, attempting to understand its properties by manipulating the input and analyzing the output.

LIME can be seen as a more structured approach to testing occlusion sensitivity. It works by progressively occluding different areas of an image and recording how this affects the output classification. This process generates an "explanation," which clearly indicates which areas of the image positively or negatively influence the predicted class.

The goal of using LIME is to gain a deeper understanding of the features the model has learned, which features it considers discriminative, and how these contribute to the final prediction.

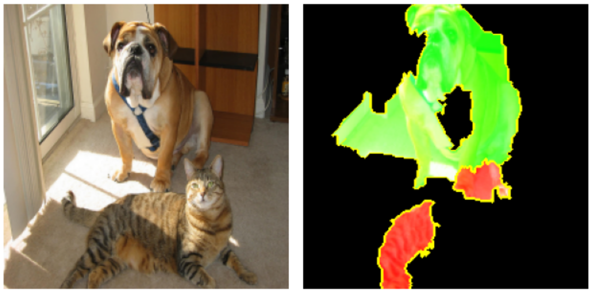

Figure 5. Example of positive and negative regions from LIME.

Figure 5 illustrates an example of the positive (areas contributing to the input image's top prediction) and negative regions (areas contributing to the input image's second-best prediction) as identified by LIME. In this instance, the most likely class is "Bullmastiff," while the second-highest class is "Tabby cat." Interestingly, the model seems to weigh the cat's fur stripes more heavily than its head when distinguishing it as a "Tabby cat."

I tested LIME on Swin Transformers of different sizes (tiny and base) and on ViT (base), and found that all models tended to focus on similar areas.

G. Impact of Image Location and Size

I observed that the quantity of features, specifically the area they cover in the image, might significantly influence the predicted classes. Additionally, the location of these features could also play an important role, particularly considering the earlier visualization of attention, where the first few encoder blocks focus heavily on spatial attention rather than feature attention. This suggests that central placement of features in an image might boost the prediction probability for certain classes. To further investigate this, I modified the image with rough cutouts (as shown in Figure 6) and analyzed the top 3 predicted classes (see Table I).

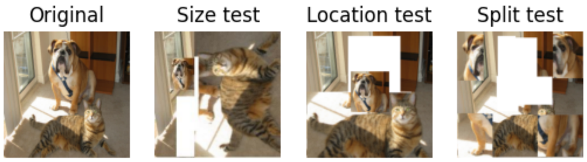

Figure 6. The tested modified images.

Location and size significantly affect the predicted classes. The "Location Test" image was particularly interesting, as the "Bullmastiff" class was suppressed to 3.3% (5th most likely class), with its position replaced by the "Boxer" class. Initially,

Image	1st class	2nd class	3rd class
Original	Bullmastif 39.2	Tabby cat 10.5	Boxer 7.3
Size Test	Tabby cat 24.6	Bullmastif 15.2	Tiger cat 12.6
Loc. Test	Tabby cat 23.0	Boxer 20.8	Tiger cat 13.1
Split Test	Bullmastif 20.1	Tabby cat 11.6	Tiger cat 9.6

Table I

Predicted classes of the modified images.

I thought the change was due to size, as Boxers are generally smaller than Bullmastiffs, but this was not supported by the "Size Test" image results.

This outcome was surprising, as I did not expect the model to be misled so easily. This might be partly due to biases in ImageNet, the dataset used for pre-training and fine-tuning, where the predicted class often appears near the center of the image. Another possible explanation could be a characteristic of the self-attention mechanism, specifically the positional embeddings. In Swin Transformers, these are implemented as positional biases added directly to the attention matrix. This could lead attention to naturally converge towards the center of the image, as the central patch has the shortest distances to other patches.

H. *Fine-tuning and Model Comparison*

This section might be slightly unrelated to the primary focus of the project and can be considered as bonus content. Initially, I intended to emphasize understanding and comparing Vision Transformers through fine-tuning for specific computer vision tasks. However, I later pivoted towards explainable AI methods.

The goal was to determine whether the Swin Transformer benefits more or less from similar fine-tuning techniques (mainly different adapter layers, location-wise) compared to the Vision Transformer (ViT). Specifically, does Swin Transformer's hierarchical structure and shifted window attention mechanism enhance the effectiveness of adapter layers?

I focused on image classification tasks since this allows both ViT and Swin Transformer models to be used as standalone models, without integrating additional architectures like R-CNN for detection. I used the base pre-trained versions of both models, as they have a comparable number of parameters (around 85M). The baseline comparison involved basic fine-tuning, replacing and retraining the classification head while freezing the rest of the weights. All runs were done using 10 epochs with 0.001 learning rate and 0.0001 weight decay. Using a learning schedule seemed to produce worse results.

I searched for datasets on platforms like Kaggle and HuggingFace, limiting myself to datasets with at most 100k images. I started with a simple classification task using the 525 Bird Species dataset. However, the samples seemed too similar to ImageNet, the dataset on which the models were pre-trained, leading to a classification accuracy of over 93% in the first epoch. This dataset was the largest used, with around 85k training samples.

Next, I tried the EuroSAT dataset, which consists of satellite images for classifying land types (e.g., industrial, residential, highway, lake/sea, forest). Since ImageNet doesn't include satellite or bird's-eye-view images, I hoped this would be a challenging task. However, the classes were too distinct, making classification too easy. ViT achieved an accuracy of 97.18%, while Swin Transformer reached 96.67%.

I then experimented with a dataset from a completely different domain: MRIs for Brain Tumors. Unfortunately, I again obtained high classification accuracy: 96.55% with ViT and 95.58% with Swin Transformer. I even tried the tiny version of Swin (around 30M parameters), hoping to reduce accuracy, but it surprisingly increased to 95.8%.

I aimed for lower classification accuracy to make the differences between fine-tuning methods more pronounced. However, after struggling to find an appropriate dataset, I decided to shift my focus back to explainable AI methods.

IV. Discussion

The most significant takeaway from this project, aside from gaining a better intuitive understanding of the attention mechanism, is the remarkable flexibility of attention itself. Fundamentally, attention operates as a similarity scoring matrix between different vectors (embeddings). This means that the form of the input becomes almost irrelevant—as long as it can be embedded (projected) into a common latent space, it can be effectively processed using attention. This flexibility is especially powerful in the context of multi-modality. In contrast, attempting something similar with Convolutional Neural Networks (CNNs) would require encoding the information into pixel form and adding it to the image (e.g., as additional rows of pixels). This approach is far less convenient and likely less effective in capturing relationships between different types of data.

Another interesting reflection is on the challenge of visualizing attention in the Swin Transformer. A potential solution to simplify this process could be to incorporate a CLS token into each window. By adding this token, the entire image could be reconstructed in a manner similar to the visualization approach I used. The tensor could take the shape: $Windows \times Heads \times (N_{tokens} + 1) \times (N_{tokens} + 1)$. To obtain the CLS token's representation, we would extract the first row and skip the first column (in the last two dimensions), then use the windows to reconstruct the image. The attention heads could be examined individually or fused (e.g., using a mean operation).

While adding a CLS token might not significantly improve the model's performance—since the model already summarizes information in the final hierarchical layer through average pooling—it would greatly simplify the visualization process. More importantly, it could provide a deeper understanding of how Swin Transformer's innovations, such as spatial hierarchy and shifted windows, contribute to its overall performance.

V. Conclusion

This project has provided valuable insights into the inner workings of Vision Transformers (ViT) and Swin Transformers, focusing on attention mechanisms and model interpretability. By exploring visualization techniques and employing methods such as Attention Rollout and LIME, we gained a deeper understanding of how attention is distributed and how it influences model predictions. We observed that while ViT's attention maps often focus on spatial patterns early in the network, Swin Transformer's hierarchical and shifted-window approach requires more complex visualization strategies but offers promising results. The fine-tuning experiments highlighted the robust performance of both models across various datasets, although significant differences in fine-tuning outcomes were not evident. Overall, the project underscores the versatility of attention mechanisms and opens avenues for further research, particularly in improving visualization techniques and understanding model innovations.

References

[1] A. Vaswani, “Attention is all you need,” *Advances in Neural Information Processing Systems*, 2017.

[2] A. Dosovitskiy, “An image is worth 16x16 words: Transformers for image recognition at scale,” *arXiv preprint arXiv:2010.11929*, 2020.

[3] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo, “Swin transformer: Hierarchical vision transformer using shifted windows,” in *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 10012–10022, 2021.

[4] S. Abnar and W. Zuidema, “Quantifying attention flow in transformers,” *arXiv preprint arXiv:2005.00928*, 2020.

[5] M. T. Ribeiro, S. Singh, and C. Guestrin, “" why should i trust you?" explaining the predictions of any classifier,” in *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pp. 1135–1144, 2016.