# Final Report

## IND5005B Industry consulting and application project

## Sustainable AI

*Submitted by*

Du Zhouyang, A0254065H
Qiu Baiyuan, A0254076A
Teo Kok Yong, A0254252L

*Supervised by*

A/Prof Prahlad Vadakkepat
Electrical and Computer Engineering
College of Design and Engineering

*Project Sponsor*

Dr Joey Zhou
Institute of High Performance Computing,
A*STAR

*Submitted on*

8 June 2023

**Table of Contents**

# 1. Introduction

## 1.1. Problem Statement

In recent years, state-of-the-art transformer-based AI models have proven successful in applications of natural language processing. This has spurred interest in its use in other fields such as computer vision. However, such models, like GPT-3, tend to be memory intensive, composing 96 layers, 175 billion parameters, and requiring an estimate of over 700 GB of memory (Brown et al., 2020). Training such a model on a V100 GPU server with 28 TFLOPS capacity would take 35 years and cost approximately $4.6 million. This intensive process also has a significant environmental cost, with carbon emissions from training a single transformer network equivalent to more than 300 round-trip air flights from New York to San Francisco.

As their deployment becomes increasingly pervasive, we can expect successor models to be larger and more demanding. It is critical to reduce their storage and computational costs, especially for some real-time on-device applications. Model compression techniques, including network pruning, quantization, and weight sharing, have proven effective for significantly reducing the model size of state-of-the-art convolutional deep neural networks while maintaining their predictive capabilities. However, these techniques may not be as readily applicable to transformer-based vision models. (Nagel et al., 2021) This presents a unique challenge and a potential area for exploration in the field of AI model optimization.

## 1.2. Project Scope

As vision transformer models are a recent development (Dosovitskiy et al., 2020) and research into applicable quantization techniques is ongoing, this project aims to summarize the state-of-the-art in post-training quantization (PTQ) techniques for Vision Transformer models and propose a unique module to quantize Vision Transformers. We hope that this work will serve potentially as the foundation for a white paper, which to the best of our knowledge, that represents the first work of this study.

The primary focus of this project is to design an efficient inference mechanism for transformer-based vision models using model quantization. The project can be broken down into three main phases:

1. Exploring and experimenting with existing quantization techniques: This phase involves a detailed study and hands-on experimentation with currently available quantization methods.
2. Developing a novel quantization technique: Based on the knowledge and experience gathered in the first phase, the second phase will focus on devising an improved quantization method that optimizes transformer-based vision models more efficiently.
3. Building an implementation pipeline: The final phase of the project will involve constructing a robust pipeline to implement and test the proposed quantization technique.

### 1.3. Company Overview

A*STAR Institute of High Performance Computing (IHPC) was established in August 1998 to provide leadership in computational modelling, simulation, and AI to solve major scientific, industrial, and societal challenges. It seeks to promote and spearhead scientific advances and technological innovations through multidisciplinary R&D, and to develop impactful applications to further economic growth and improve lives.

Their research focuses on computing science and AI; large scale complex systems modelling; social and cognitive computing; computational engineering mechanics, fluidic dynamics, electronics and photonics, materials science, and chemistry. These core capabilities enable IHPC to tackle real-world challenges in physical and human systems, such as in manufacturing, energy, transportation and urban systems, environmental sustainability, and healthcare.

## 2. Background

### 2.1. Motivation of Study

There have been considerable efforts to research the issues of deploying deep neural networks (NNs) more efficiently while considering for their size, latency, and portability. These have been broadly categorised into techniques like pruning and knowledge distillation (Gholami et al., 2020). However, quantization has emerged as one of the more effective and popular methods (Nagel et al., 2021).

Given the growing popularity of transformer models, our study aims to consolidate and reveal insights into the post-training quantization (PTQ) of state-of-the-art (SOTA) vision transformer models. By doing so, we hope to contribute valuable knowledge to this rapidly developing field.

### 2.2. Overview of Transformer Structure

The attention mechanism (Bahdanau et al., 2014; Kim et al., 2017) was originally used to complement Recurrent Neural Networks (RNNs). In theory, it allows the modelling of dependencies with an infinite reference window, addressing the limitations of standalone RNNs that struggled with longer input sequences. *Attention is all you need* (Vaswani et al., 2017) revolutionized this landscape by creating a transformer model that relies entirely on self-attention and does not use sequence aligned RNNs or convolution. This important shift alleviates the constraints of RNNs related to parallelization due to their sequential nature, enabling faster training times.

The architecture of the Transformer model consists of an encoder stack and a decoder stack. (Fig. 1 left) Each layer in the stack is composed of sub-layers, which are either a multi-head attention layer or a point-wise feed-forward network (FFN). Learned embeddings are used to convert the input and output tokens into vectors. These embeddings are then supplemented with positional encodings, providing some positional context of the tokens in the sequence. These embeddings are then transformed into query (Q), key (K), and value (V) matrices.

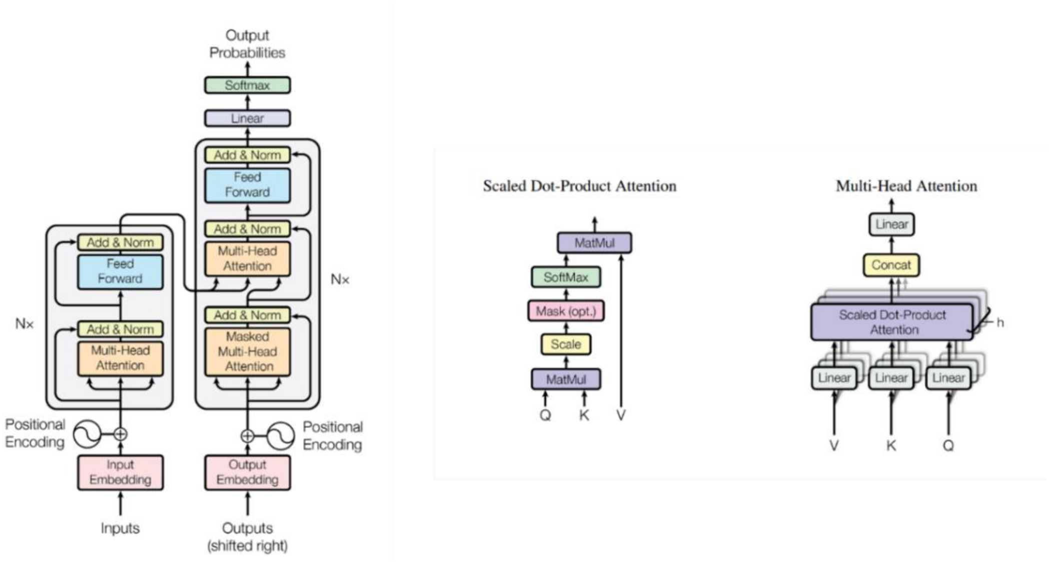


*Fig. 1 The Transformer –model architecture, left half encoder, right half decoder (left). Scaled Dot-Product Attention (mid). Multi-Head Attention of several attention layers in parallel (right).*

At the heart of the Transformer model is the attention function. This function uses a query and a set of key-value pairs to compute a weighted sum of values. The model applies a process known as "Scaled Dot-Product Attention" (Fig. 1 mid) to calculate the dot products of the query with all keys, scales these by a function of the model dimension, and then applies a Softmax function to obtain the weights for the values. The Transformer model also employs a "multi-headed" approach (Fig. 1 right), projecting the queries, keys, and values multiple times with different, learned linear projections. This strategy allows the model to attend to information from different representation subspaces at different positions.

The outputs from the multi-head attention process are then normalized and passed through a fully connected feed-forward neural network. This network applies two linear transformations, with a Rectified Linear Unit (ReLU) activation function in between. Normalization is a crucial step that stabilizes the network and reduces training time.

## 2.3. Overview of Vision Transformer Models

### 2.3.1. Vision Transformer

Inspired by the successful adoption of Transformers in natural language processing tasks, Dosovitskiy, *A*. et. al. (2020) proposed using a standard Transformer model, with minimal modification, for computer vision tasks. This is done by first splitting an image into fixed-size patches. Thereafter, these patches are linearly embedded and supplemented with positional encoding before being fed into the Transformer's encoder stack (Fig. 2).

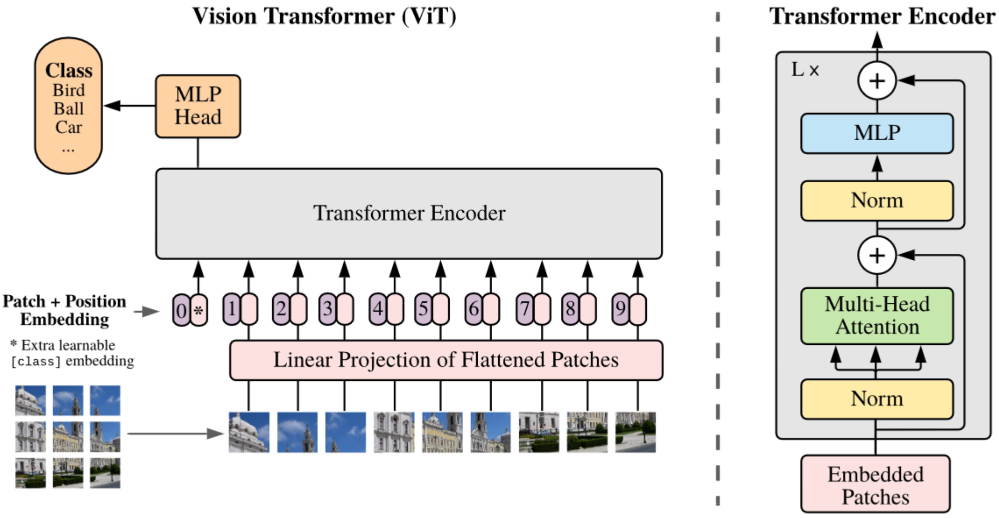


*Fig. 2: Vision Transformer – model architecture*

A key modification in the ViT model is the inclusion of an 'extra learnable class embedding', which is inspired by the BERT model (Devlin et al., 2019). This class embedding is prepended to the sequence of embedded patches, allowing the self-attention mechanism to be computed between the patch tokens and the class token within the Transformer's architecture. At the end of the process, the state of this class token is passed to a classification head.

Another significant difference between the ViT and the original Transformer model is the activation function used in the multi-layer perceptron (MLP) block. Instead of the ReLU activation function used in the original Transformer, the ViT employs a Gaussian Error Linear Unit (GELU) activation function.

The ViT yielded modest results when trained on mid-sized datasets such as ImageNet. However, when trained with larger datasets, proprietary JFT-300M dataset, the ViT closely matches or outperforms SOTA models on many image recognition benchmarks. This outcome was expected as the Transformer’s generalization capacity is known to be limited when trained with insufficient data volumes.

### 2.3.2. Data efficient image Transformer

While the convolution-free ViT closed the gap with the state of the art on ImageNet, it did so through training with a large privately-owned image dataset (JFT-300M, 300 million images). This motivated the proposal of Data-efficient image Transformer (DeiT), a transformer-only model that attains competitive results, despite being trained solely on the publicly available ImageNet dataset (Touvron et al., 2020).

A key enhancement that they introduced was the distillation token, employing a RegNetY-16GF architecture (Radosavovic et al., 2020) as the default teacher model (Fig. 3). Empirically, they found a convolutional neural network (CNN) to be a more effective teacher than a transformer, likely due to the inherent inductive bias present in the CNN architecture. The attention heads compute the scores for the distillation token together with the other embeddings and use the objective function given by the distillation component of the loss for optimization.

Additionally, extensive data augmentation was added to satisfy the transformer’s requirement for a larger amount of data. Rand-Augment (Cubuk et al., 2019) was preferred over Auto-Augment (Cubuk et al., 2018) after ablation studies. Overall, it was confirmed that strong data augmentation improves the model in all data-augmentation methods evaluated. The DeiT was found to be competitive against state of the art on ImageNet. The pre-learned models were also competitive on fine-grained classification on a few popular public benchmarks.

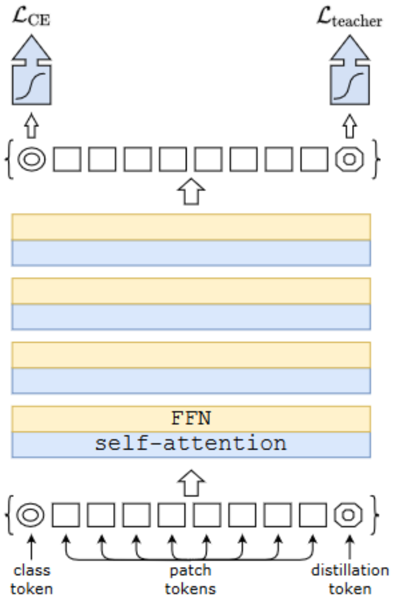


*Fig. 3: Distillation enhancement to ViT in DeiT*

### 2.4. Overview of Quantization

Quantization is the division of a quantity into a discrete number of small parts, often assumed to be integral multiples of a common quantity (Gray & Neuhoff, 1998). This can be applied in the post training quantization (PTQ) of pre-trained NN without any need for the original training pipeline. Weights and computations can be quantized from FP32 to INT8 representations to realize 4x efficiency in storage and 16x efficiency in computational costs (Nagel et al., 2021).

### 2.4.1. Uniform Quantization

Uniform quantization is a common approach in quantization that maps real values to uniformly spaced discrete values. In simple terms, this is done by assigning each real-valued input (r) a corresponding integer value within a specific range. The scaling (S) and the integer zero point (z) are critical parameters in this process.

$$\mathrm{Quant}(r) = \mathrm{Int}\left(\frac{r}{s}\right) - z$$

Quantization essentially compresses the range of values, which can then be dequantized back into real values when needed. However, due to the rounding operation involved in quantization, the dequantized values may not always match the original values exactly.

The scaling factor (s) is a key component in this process and can be chosen using several methods. A popular method involves defining a clipping range, denoted by [α, β], which is also known as calibration.

$$s = \frac{\beta - \alpha}{2^b - 1}$$

Asymmetric quantization uses a min-max approach to determine the clipping range, with α and β being the minimum and maximum values of r respectively. This method is particularly useful when the distribution of real values is skewed, and the range of values isn't symmetrically distributed around the origin.

Symmetric quantization, on the other hand, is more widely used. Here, the clipping range is also chosen based on the minimum and maximum values but centred around zero, allowing the zero point (z) to be zero. This results in computational savings and ease of implementation.

There are also other calibration methods that can be used, including the percentile approach, optimizing a loss function (Mean Square Error or Cross Entropy), batch normalization (Nagel et al., 2021; Wu et al., 2020), channel splitting (Zhao et al., 2019), and a technique called AdaRound (Nagel et al., 2020).

### 2.4.2. Non-Uniform Quantization

The concept behind non-uniform quantization essentially maps a real-valued input (r) to a corresponding discrete quantization level ($X_{\mathrm{i}}$,) if r falls within a certain range (between $\Delta_i$ and

$\Delta_{i+1}$) The ranges and levels are not uniformly spaced, thus allowing for more flexibility in capturing the distribution of real values in the model.

$$\mathrm{Quant}(r) = X_{\mathrm{i}}, ifr \in [\Delta_i, \Delta_{i+1})$$

One variation of this method is to restrict the scale factor (s) to a power-of-two. This makes use of a base-2 logarithmic representation which has the advantage of simplifying multiplications and divisions to bit-shifting operations, thereby providing hardware efficiencies (Przewlocka-Rus et al., 2022). The trade-off for the improved accuracy however is that that these methods can be challenging to deploy efficiently on general computation hardware due to their complexity.

### 2.4.3. Zero-Shot Quantization

Maintaining the accuracy of Neural Network models after quantization is a critical goal. One method to mitigate the accuracy degradation often seen after quantization is to apply fine-tuning with a subset of the original training data (Gholami et al., 2021). However, practical challenges often arise with this approach due to the large size of the dataset, its proprietary status, or the sensitive nature of the information it contains.

As an alternative, several studies have explored a generative approach, aiming to create synthetic data for the purpose of fine-tuning the quantized model. This technique, which falls under the category of 'data-free' or 'zero-shot quantization' (ZSQ) (Chen et al., 2019), operates without the need to access the original training or validation data.

Zero-shot quantization provides a promising avenue for efficient model optimization while navigating constraints around data accessibility and sensitivity. The synthetic data generated in this process serve as a proxy for fine-tuning, allowing models to maintain accuracy post-quantization without the need for the original training dataset.

## 2.5. Application to Vision Transformers

### 2.5.1. Differences between CNN and ViT PTQ

When performing PTQ on ViTs, the process is more complicated compared CNNs due to their distinct structures. Firstly, there is the attention block unique to ViTs. Though ViTs have shown the potential to outperform many SOTA CNN models in computer vision (CV) tasks, they are very much more heavy-weight when compared to CNNs which have been powering many mobile vision tasks (Mehta & Rastegari, 2021). We can see this when comparing ViT-B/16's 86 million parameters vs. MobileNetv3's 7.5 million. This has often been attributed to the fact that ViTs lack image-specific inductive bias, which are inherent in CNNs (Xiao et al., 2021). We identify the quantizing of the attention map to be a bottleneck for computational resources and energy to be saved on the hardware level.

Secondly, ViTs uses layer normalization (LayerNorm) rather than batch normalization (BatchNorm) in CNNs, which impacts the quantization process. BatchNorm is common in CV tasks as it maintains the size relationship that might be important for analyzing different pictures. Comparatively, LayerNorm is used in NLP tasks, for which the transformer was originally designed for, to preserve word embeddings. Given that BatchNorm narrows the difference between features, performing quantization on this step does not cause a significant degradation on accuracy. This difference in features remains for the output of LayerNorm in ViTs.

Additionally, it is common to keep the activation of the Softmax, which exists only in the final layer of CNNs, in full precision during PTQ. On the other hand, there is a Softmax layer for each of the 12 attention blocks in ViT. Keeping this layer in full precision will require de-quantizing the attention map before passing through Softmax and re-quantize again prior to computation with the value matrix. This can result in higher computational cost and hence longer inference time compared to the original full precision ViT model.

Finally, while many CNN models use the ReLU activation function, ViT-based models employ the GELU function, a non-linear activation function, which causes asymmetrical distribution on both sides of the zero point. This affects the quantization strategy for multi-layer perceptron (MLP) component, making it more complex than in the case of CNNs.

### 2.5.2. Challenges

The application of PTQ on ViTs presents a considerably diverse set of challenges from CNNs, primarily due to their structural differences. Notable works by Lin et al. (2021) and Li et al. (2022) have demonstrated that quantization on LayerNorm in ViT leads to significant performance degradation (Li, Xiao, et al., 2022; Lin et al., 2021). This is attributed to the fact that LayerNorm doesn't mitigate the heterogeneity of feature distribution within a sample, unlike BatchNorm in CNNs.

Moreover, ViTs have been observed to exhibit a drastic increase in feature variance as layers deepen  (Ze Liu et al., 2021), leading to serious inter-channel variation for LayerNorm inputs. When applying linear PTQ, we typically use a single pair of quantization parameters for each encoder's LayerNorm, which could potentially introduce errors impacting model performance. This is mainly because the scaling factor is an essential parameter for uniform quantization, and a layerwise quantization strategy does not consider the substantial distribution differences between all channels.

Another challenger is the impact of quantization on the relative order of the attention map in ViTs  (Zhenhua Liu et al., 2021). The attention map serves to indicate the correlation between the different tokens. However, this relative order is changed after quantization, and the representational capacity of transformer blocks becomes negatively affected.

Additionally, the storage and computation of attention maps present bottlenecks for inference throughput and latency. The original distributions of attention maps can be highly imbalanced original distributions, with most values centered around a very small float number (~0.001) and a few between 0.1 and 1, or close to 1. Thus, direct application of uniform quantization leads to most values being assigned to a single quantization bin, resulting in substantial information loss. Also, the values close to 1 will be sensitive to perturbations of calibration data.

The activations after Softmax and the asymmetrical distribution after the GELU activation function present similar challenges. While an asymmetric quantizer might potentially address these issues and maintain the quantized model's accuracy, we must consider other vital metrics such as inference time and energy consumption. Since the motivation of quantization is efficient and pervasive deployment on edge devices, the implementation feasibility on hardware and potential impact on inference time need to be considered alongside model size and accuracy. Whilst taking such an approach may maintain performance, the hardware implementation can be complicated with a longer inference time.

## 2.6. Existing Approaches to Vision Transformer Quantization

This section dives into the SOTA approaches to ViT quantization, offering potential solutions to the challenges outlined earlier. Through a comprehensive review of relevant literature, we summarize the prominent strategies that aim to balance model performance, computational efficiency and implementational feasibility in the context of ViT quantization.

### 2.6.1. Mixed Precision (Ranking Loss) for ViT

The approach proposed relies on an innovative strategy known as "ranking loss and mixed-precision quantization" (Zhenhua Liu et al., 2021). Essentially, this technique aims to optimize the assignment of lower-bit quantization intervals for weights and inputs, while also ensuring the functionality of the self-attention mechanism remains intact. The concept of ranking loss was introduced to recognise the critical role that the relative order of self-attention features plays. The ranking loss mechanism is designed to maintain the relative order of self-attention results, with a method to correct bias also utilized. This correction comes into play after the optimal parameters are identified, helping to reduce any biased errors that might arise due to the quantization process.

In addition, the paper introduced a technique called "bias correction" aimed at curbing the potential drop in accuracy due to changes in distribution that occur during quantization. Essentially, this method seeks to adjust the parameters to account for the shifts in distribution, thereby minimizing the impact on accuracy.

To further optimise the quantization process, the paper suggested a mixed-precision quantization approach. The key idea is to allocate more bits to the layers that are more sensitive, which ensures the model's overall performance remains competitive.

### 2.6.2. PTQ4ViT

This study explored an innovative approach to address the significant deviation observed in the distribution of activation values following the application of Softmax and GELU functions, a deviation from the expected Gaussian distribution. (Yuan et al., 2022). Furthermore, they observed that commonly used quantization metrics, like MSE and cosine distance, are not adequate in identifying the optimal scaling factor.

To confront these issues, they put forward a "twin uniform quantization" method to minimize quantization error on activation values through the utilization of two distinct quantizers. Moreover, they introduced a Hessian guided metric, which was used to assess different scaling factors, thereby improving the calibration accuracy with minimal additional cost.

This high-efficiency framework, known as PTQ4ViT, expedites the quantization process for vision transformers. Their experiments showed that the quantized vision transformers maintained almost the same level of prediction accuracy, with only a minimal drop (less than 0.5%) at 8-bit quantization on the ImageNet classification task. This achievement underlines the potential value of their method in progressing the realm of post-training quantization for vision transformers. As a result, this framework offers a valuable tool in streamlining the process of quantization.

### 2.6.3. FQ-ViT

Lin, Y. et. al.  identified gaps in previous research, which did not propose fully quantized models, by addressing the serious inter-channel variation problem in LayerNorm and Softmax operations (Lin et al., 2021). They introduced the power-of-two factor which assigns unique factors to different channels. The most optimal factor for each channel is determined by how similar the quantized feature map is to the original full-precision version. A key advantage is the hardware-level efficiency as having the scaling factor divisible by a power of two allows execution with a simple bit-shift operation.

In addition, FQ-ViT proposed non-linear Log-Int-Softmax for the quantization of attention map and Softmax. This mitigates the need for data moving between CPU and GPU when performing dequantization and requantization before and after Softmax. As a result, the method is more hardware-friendly. The researchers have done a wide range of experiments and find that FQ-ViT outperforms the previous classical quantization method (MinMax, EMA, Percentile, OMSE, Bit-Split etc) in all DeiT, ViT, and Swin-Transformer models. When compared to the original full-precision model, top-1 accuracy only degrades by 1-2 percent.

$$X_Q = Q(X|b) = clip(\lfloor\frac{X}{2^{\alpha}s}\rceil + zp, 0, 2^b - 1)$$

### 2.6.4. NoisyQuant

Liu et al. proposed an unorthodox solution that by adding noise that follows a uniform distribution to the activations before quantization, quantization error can be significantly reduced (Liu et al., 2022). This method is focused on improving non-linear quantization algorithms that deal with the imbalanced distribution of parameters. It is, however, ill-suited for the quantization of attention maps, Softmax and GELU operations.

The key idea behind the adding of noise is to modify the original imbalanced distribution to a new one that is suitable for implementing the hardware-friendly uniform quantization. This noisy quantizer also refines the distributions to reduce quantization error.

Additionally, since the quantization of activations is a key contributor to performance degradation, the researchers proposed introducing noise that is randomly sampled from a predetermined uniform distribution to improve the quantization process. The output of the quantizer then undergoes a de-noising phase. This method has been shown to improve the SOTA top-1 accuracy on ImageNet by up to 1.7% for ViT, 1.1% for DeiT and 0.5% for Swin Transformer (Liu et al., 2022).

### 2.6.5. CPT-ViT

Building on previous research of using linear search strategies to find the optimal clipping range for quantization, Frumkin, N. et. al. a block-wise evolutionary search strategy to reduce the search complexity (Frumkin et al., 2022). This search algorithm comes from the genetic algorithm, which is in turn inspired by the phenomenon of replication, crossover, and mutation in natural selection. It starts with an initial population and, through a series of random selection, crossover, and mutation operations, generates a group of individuals better adapted to the environment. This process leads to continuous evolution over generations, resulting in a group of individuals optimally suited to their surroundings.

In the context of ViT quantization, the individuals represent different quantization parameters. The performance of each parameter set is evaluated using contrastive loss, which measures how well each set preserves the original model's performance post-quantization. More specifically, the contrastive loss is used to maximize the similarity between the outputs of the quantized model and the original full-precision model for a given image while simultaneously maximizing dissimilarity with other images in the same batch. This approach has been shown to outperform both DeiT and ViT on 8-bit and 4-bit quantization, emphasizing the potential of altering the search strategy to find superior quantization parameters.

### 2.6.6. RepQ-Vit

Liu, Z. et. al. proposed a framework that notably decouples the quantization and inference processes (Li, Xiao, et al., 2022). They accomplish this by applying channel-wise quantization for LayerNorm and log root2 quantization for the Softmax layer during the calibration process. Then, a scale reparameterization is performed so that the layer-wise quantization and log2 quantization are used during inference process. This approach cleverly ensures the model benefits from the high performance of the former method and the efficiency of the latter.

Recognizing the challenge posed by severe inter-channel variation before LayerNorm, the researchers proposed applying channel-wise quantization to tackle this, although this method requires more energy and time for inference. During quantization, scale reparameterization uses all pairs of quantization parameters for each channel to adjust the gamma and beta parameter for normalization of each channel.

However, during inference, only one pair of quantization parameter (s and z) is used, changing the way the activations are normalized. Furthermore, the researchers find that log root2 quantization might be more suitable for quantizing Softmax layer, and log root2 quantizer can be easily implemented by multiplying the result of a log2 quantizer with 0.5, which can be easily done on a hardware level. Rep-Q ViT outperforms the PTQ4ViT on all ViT, DeiT, and Swin models, offering a new mode of optimizing the PTQ framework.

### 2.6.7. PSAQ-ViT (V1 and V2)

Li, Z. et. al. (Li, Ma, et al., 2022) proposed the Patch Similarity Aware data-free Quantization (PSAQ-ViT) framework in a generative approach to ZSQ. By comparing patch similarity and cosine similarity between Gaussian noise images and real images, the researchers defined optimization functions to encourage the generation of images to a desired category. These synthetic but “realistic” samples are then utilized in the calibration of the clipping range.

This was followed up with PSAQ-ViT V2 (Li, Chen, et al., 2022), which introduces an adaptive Teacher-Student Strategy to play a minimax game in image generation that uses the model discrepancy between the quantized model and full-precision model. Essentially, the framework aims to optimize the clipping range without any access to the original training data. Further, such images can be customised according to the computer vision tasks.

## 2.7. Structure of our work

Our literature review has uncovered some algorithms to help maintain the accuracy of quantized model and through our replication of FQ-ViT and twin-uniform quantization, we have understood some bottlenecks of post-training quantization for ViT (experiment results shown in Appendix 7.1). However, for the deployment of such models on hardware, we would expect to preserve the full precision models' performance while using simple quantization techniques that are friendly to hardware. Therefore, we build our own quantization pipeline with the help of SenseTime PPQ framework (Section 3.2.2). We aim to customize our own quantizer based on simple but effective quantization policy and discover more insights for the post-training quantization of ViT.

The remainder of the following content is as follows:

- Section 3.1 introduces the timeline of our project.

- Section 3.2 presents a brief introduction of the open-source tools which contribute to our research.
- Section 3.3 gives information of the image dataset used in our experiments.
- Section 3.4 shows our experiment setups, which forms the basis for all our follow-up experiments.
- Section 4 delivers the results of our quantization, as well as ablation study, and further discussions into the possible bottlenecks that affect the model's performance.
- Lastly, section 5 concludes with a summary of our findings and some future work that might be pursued based on existing results.

## 3. Methodology

### 3.1. Project Management and Schedule

Initially, our idea is to build a PTQ pipeline from scratch. However, as we found an open-source quantization framework, it facilitated our efforts to build a quantization pipeline. Nevertheless, it took some time before we became familiar with the framework. We were then able to implement some ideas from literature and come up with new ideas. Ultimately, the time schedule (Fig. 4) for building the pipeline remained the same.

**Jan 2023:** We started the project off by first understanding about transformer-based AI models and its applications in vision tasks.

**Feb 2023:** Next, we did a literature review on the quantization techniques and the state of the art approaches to PTQ for ViT.

**March 2023:** We performed some basic experiments on existing PTQ methods uncovered from the literature review.

**April 2023:** We started building the PTQ pipeline using open source tools where possible.

**May 2023:** Finally, we attempted to implement our own quantization method and analyzed its feasibility.

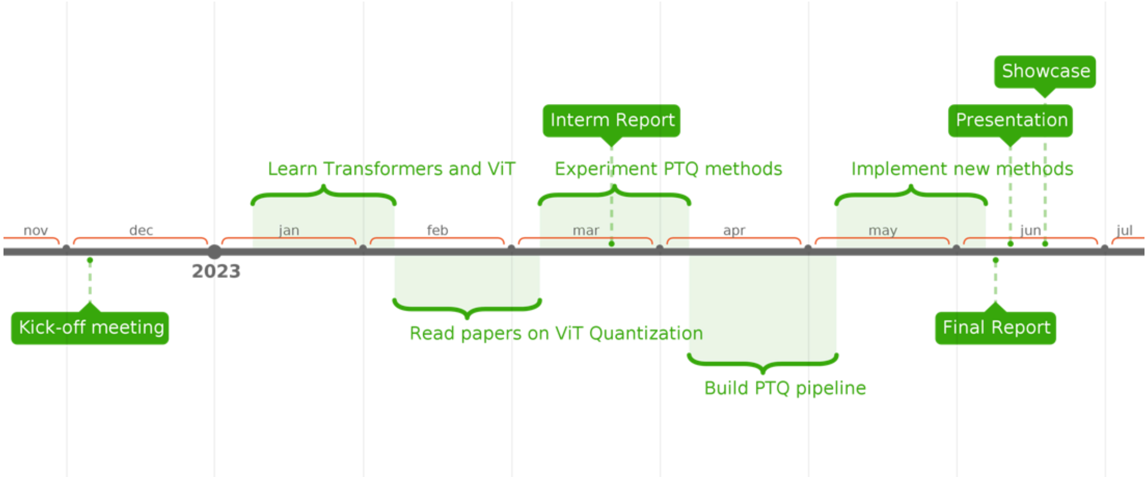


*Fig. 4: Timetable of our project*

## 3.2. Tools and Technologies Used

### 3.2.1. System Resources

The experiments conducted are performed on the following platforms:

- 11th Gen Intel® Core™ i7-11800H @ 2.30GHz, 16.0 GB RAM, GPU: NVIDIA GeForce RTX 3060 6GB
- Intel® Core™ i5-6600K CPU @ 3.50GHz 3.50GHz, 16.0 GB RAM, GPU: NVIDIA GeForce GTX 1060 3GB
- Intel® Core™ i3-10105F CPU @ 3.70GHz 3.70GHz, 16.0 GB RAM, GPU: NVIDIA GeForce RTX 960 2GB

### 3.2.2. Open-Source Tools

*PPQ* is a scalable, high-performance neural network quantization tool for industrial applications (SenseTime, 2023). This is a framework for dealing with complex quantization tasks - PPQ's execution engine is designed for quantization, as of PPQ 0.6.6 version, the software has a total of 99 common ONNX operator execution logic built in, and native support for quantization simulation operations during execution. The PPQ can be separated from the ONNX Runtime to complete the reasoning and quantification of the ONNX model.

To save computing power consumption of training the model, and to allow us to compare the quantization results with the initial model, we use the pre-trained model in *timm* library as the target quantization object (Wightman, 2019). Timm is a library containing SOTA computer vision models, layers, utilities, optimizers, schedulers, data-loaders, augmentations, and training/evaluation scripts. It comes packaged with >700 pretrained models and is designed to be flexible and easy to use.

## 3.3. Data Used

The data used in our experiments is a subset of Imagenet2012 ILSVRC2012, with 1000 classes.

- For calibration, 1000 images were randomly sampled from 1000 classes of 10 images each.
- For validation, 1000 classes of 50 images were sampled.
- To speed up experiments, we also use a tiny validation dataset, with 10 images per class, which makes 10000 images in total for our ViT model-based experiments.

## 3.4. Experiment Setup

### 3.4.1. Quantized operators

With the help of PPQ framework, we have successfully designed the structure of customized quantizer. We name all active operators as quantization operators, to make a fully quantized model. The base operators in ONNX graph include 'MatMul', 'Mul', 'Conv', 'Add', ‘Softmax’ etc. However, if we perform quantization on these operators individually, the actual inference time is much longer. This is because the quantization might only speed up the process of computation on GPU. However, during a whole process of the computation, it will go through the stage of E (task emission), R (read data), C (computing), and W (write data). For operators like bias, ReLU etc. Their R/W time is much more than their computing time, which means there will be a larger memory access consumption if doing quantization on each operator individually.

### 3.4.2. Graph fusion

Graph fusion refers to the merging of multiple computational operators into one larger computational operator. The main purpose of graph fusion is to reduce the overhead of computation and memory access to improve the inference speed and resource utilization of quantized models.

In our experiment we used three graph fusion operators, namely ‘LayerNorm’, ‘PPQBiasFusedMatMul ‘, and ‘GELU’. Fig. 5 shows the breakdown of these operators.

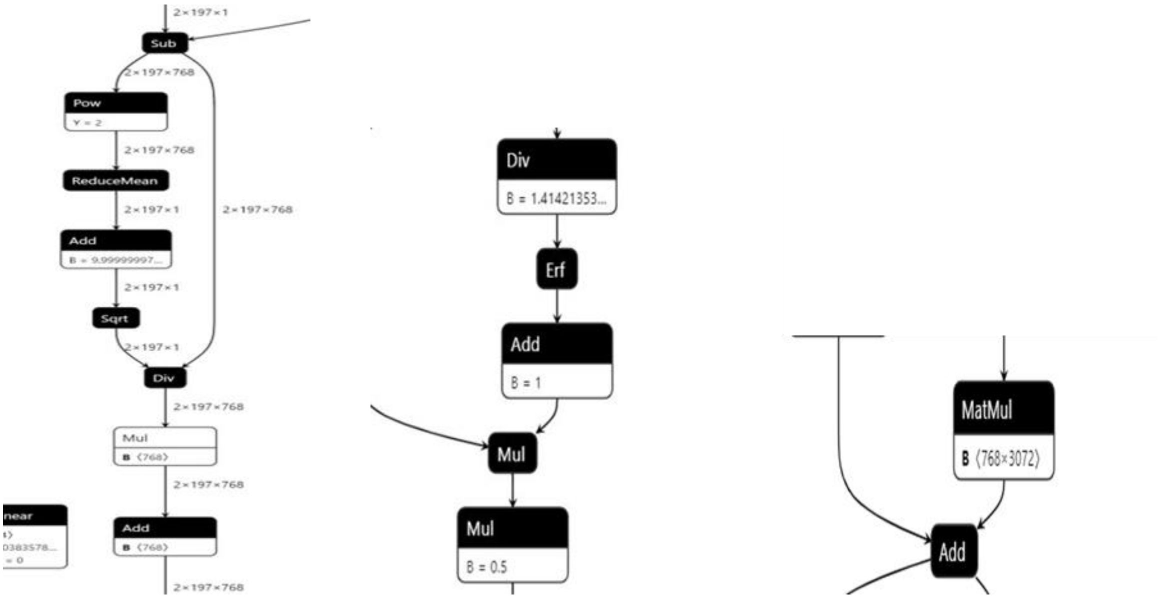


*Fig. 5: Operators to be merged for LayerNorm (left), GELU (mid), MatMulAdd (right)*

To give a more intuitive numeric comparison, Table 1 shows the average inference time \using different models with and without graph fusion on same device.

| **Avg inference time in quantized model** | **Without graph fusion (s/pic)** | **With graph fusion (s/pic)** |
|---|---|---|
| ViT-base | 0.288 | 0.180 |
| Deit-tiny | 0.149 | 0.163 |
| Deit-small | 0.223 | 0.181 |
| Deit-base | 0.313 | 0.190 |

*Table 1: Comparison of inference time with and without implementing graph fusion.*

### 3.4.3. Quantization initialization

For each of our operator, the initial setting for quantization is shown in the table below:

| | |
|---|---|
| Num of bits | 8 |
| Quant max value | 127 |
| Quant min value | -128 |
| Observer algorithm | ‘percentile’ for per-tensor |
| Policy | Per-tensor \| linear \| symmetric |
| Rounding principle | Round half even |

*Table 2: Initialized quantization parameter for each single operator to be quantized.*

In short, the 8-bit linear symmetric quantization is used as default setting, Observer algorithm is used for calibration stage, since the min-max value range is a key factor in quantization, ‘percentile’ means to drop certain percent of extreme range values, in case it harms the performance of quantized model. Our ablation study and other experiments are based on these setups.

## 4. Results and Discussion

### 4.1. Ablation study

To study the effect of our proposed strategy, we first replicated the impact of interchannel variation on the performance of the quantized model. Whilst maintaining per-tensor quantization for all other layers, we observed that the model suffers extreme degradation in accuracy when the strategy for quantization at LayerNorm is per-tensor compared to per-channel. This confirms the importance of the choice of quantization strategy when a layer encounters serious inter-channel variation (Lin et al., 2021). Therefore, based on the discovery of previous research, we choose to perform the channel-wise quantization on our model through the rest of experiments.

We extended this idea to the ablation study of the other layers while maintaining per-channel quantization in LayerNorm. A few observations can be made from the results (Table 3).

- The largest gain (Fig. 6) in the Top-1 and Top-5 accuracy resulted from performing per-channel quantization on FC Layer 2 (FC2).
- The improvement was not significant when we adopted per-channel quantization on FC Layer 1 (FC1), QKV and Projection layers instead.
- Per-channel quantization for all layers gave the best top-1 and top-5 accuracy but is not significantly better compared to only on FC2. The added computational costs of per-channel quantization of all layers for such small gains may not be worth the tradeoff.

| **FC1** | **FC2** | **QKV** | **Projection** | **LN Layer** | **Top-1 Accuracy** | **Top-5 Accuracy** |
|---|---|---|---|---|---|---|
| per-tensor | per-tensor | per-tensor | per-tensor | per-tensor | 0.52 | 2.27 |
| per-tensor | per-tensor | per-tensor | per-tensor | per-channel | 64.06 | 84.96 |
| per-tensor | per-channel | per-tensor | per-tensor | per-channel | 71.5 | 90.23 |
| per-channel | per-tensor | per-tensor | per-tensor | per-channel | 64.26 | 85.09 |
| per-tensor | per-tensor | per-channel | per-tensor | per-channel | 64.29 | 84.93 |
| per-tensor | per-tensor | per-tensor | per-channel | per-channel | 64.42 | 85.09 |
| per-channel | per-channel | per-channel | per-channel | per-channel | 72 | 90.23 |
| per-channel | per-tensor | per-channel | per-channel | per-channel | 64.530 | 85.270 |
| full precision | full precision | full precision | full precision | full precision | 75.850 | 92.990 |

*Table 3: Top-1 and top-5 accuracy for performing per-tensor and per-channel quantization on different layers in ViT-Base compared to the full precision model.*

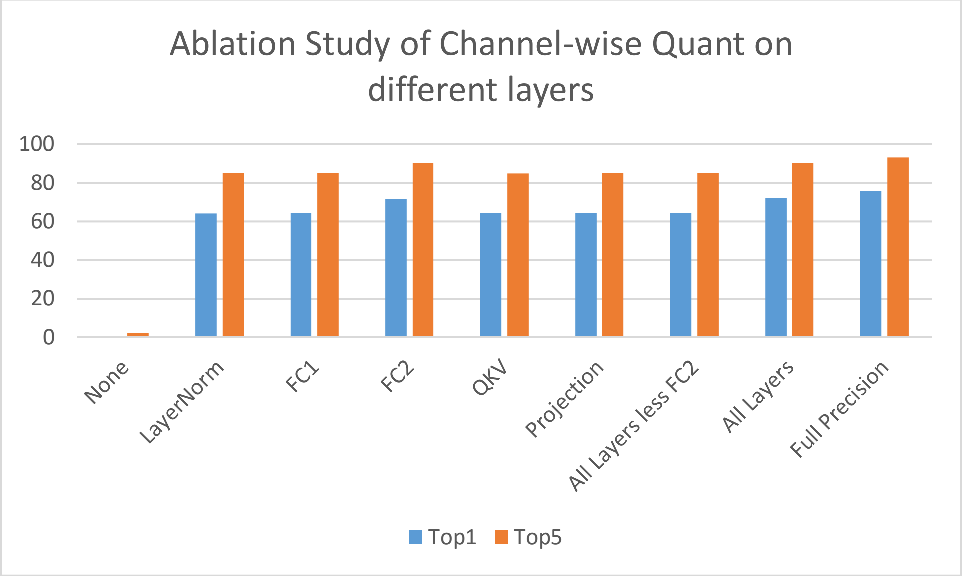


*Fig. 6: The result of ablation study where y-axis is the top-1 and top-5 accuracy; x-axis shows the operators in the block that will be quantized in a channel-wise way, except for the 'None' where all operators are quantized in tensor-wise way and 'Full Precision' is the result of original model.*

### 4.2. Results of our quantization strategy

| **Method** | **W/A/Attn** | **DeiT-Tiny** | **DeiT-Small** | **DeiT-Base** | **ViT-Base** |
|---|---|---|---|---|---|
| Full Precision | 32/32/32 | 72.016 | 79.724 | 81.74 | 75.664 |
| Ours | 8/8/8 | 70.812 | 78.21 | 80.486 | 72.000 |
| Degradation % |  | 1.672% | 1.9% | 1.534% | 3.566% |

*Table 4: Comparison of the top-1 accuracy with the full-precision model*

We compare the effectiveness of our proposed method against the full precision model of Deit-Tiny, -Small, -Base and ViT Base. The top-1 accuracy results are reported in Table 4. The percentage degradation is on average, less than 2% for our quantized method. This shows that despite the lower precision, our quantization strategy is still capable of showing competitive results.

Consistently, larger transformer based vision models are reflecting better scores for both full precision and quantized models. Interestingly, we observe that a quantized DeiT-Small model seems capable of outperforming the full precision ViT-Base model. The DeiT-Small model is approximately a quarter of the size of the ViT-Base model.

### 4.3. Analyzing activations

From the findings of our ablation study, we observed that MLP/FC2 layer of the transformer block seems to be a critical layer in the process of post-training quantization. Specifically, we tried to perform per-channel quantization on four different types of fused "*MatMul + Add*" operators, namely "*qkv MatMul + Add*", "*proj MatMul + Add*", "*fc1 MatMul + Add*" and "*fc2 MatMul + Add*". The details of the operators are as follow:

- **qkv MatMul + Add**: Linear projection from patch embedding tokens to query, key, and value matrices.
- **proj MatMul + Add**: Projection from the matrix multiplication between attention map and value matrix to the output of the Multihead Self-Attention block.
- **fc1 MatMul + Add** & **fc2 MatMul + Add**: The two fully connected layers in the multiple-layer perceptron of transformer block.

In this section, we present the insights derived by diving into the patterns of different activations of the inputs of these four operators.

#### 4.3.1. Channel Distribution Analysis

The researchers of FQ-ViT have shown the serious errors caused by the inter-channel variation before LayerNorm block. Therefore, we expect that the four fused MatMul operators mentioned may be affected by such variation. However, the effect of inter-channel variation on these four operators are not as obvious as that of LayerNorm, because the residual connection before the norm layer accumulates the original unnormalized activation values, resulting in a large difference between the maximum and minimum values of each channel. Nevertheless, this variation could still harm the model's performance.

In classification tasks using ViT, we believe the class token plays a significant role since all other tokens are ultimately abandoned. Only the class token is retained for the last MLP head. We can observe this effect when reviewing the activations of the fully connected layers in the transformer block. By breaking down the activations into the individual tokens, we can see the distribution of the minimum and maximum values of FC1 (Fig. 7 top) are heavily influenced by the first token (Fig. 7 bottom-left), which is the class token prepended to the patch tokens. However, this effect is no seen in FC2 layer (Fig. 7 bottom-right). This is likely attributed to the increase in number of channels and GELU operation which changes the distribution as each token has a different embedding dimension (1, 3072) to the original (1, 768). A full visualization of the tokens can be reviewed in Appendix 0 and 0.

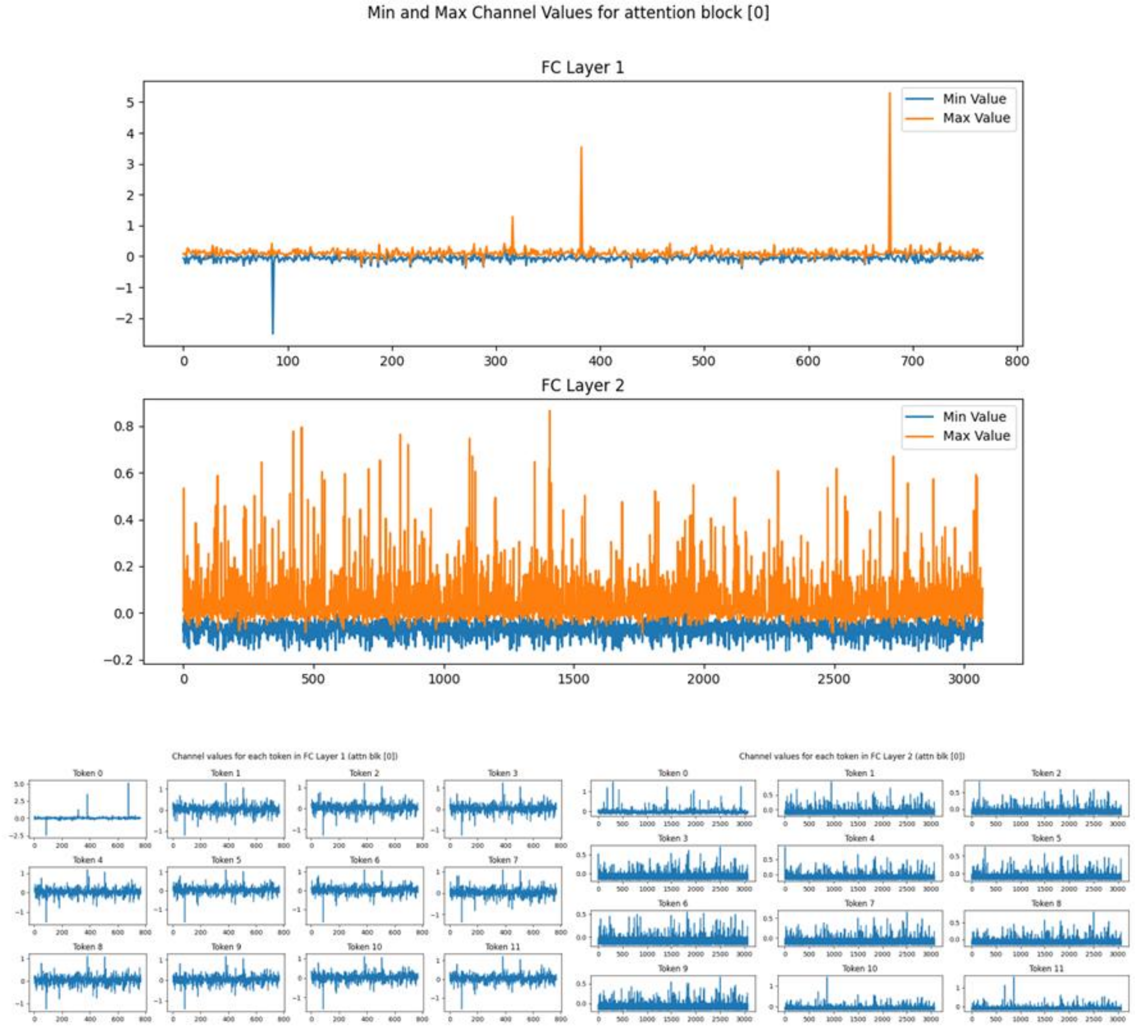


*Fig. 7: Min and max channel values of FC1 and FC2 (top), channel values of FC1 class token vs patch tokens (bottom-left), channel values of FC2 class token vs patch tokens (bottom-right).*

Therefore, we have included the error of class token as one of the metrics for measuring quantization error. We also use the ‘percentile’ strategy for per-tensor quantization to eliminate the effect of outlier values. Particularly for FC1, with the influence of the class token, the clipping range is extended over a large range of values. Hence, the remaining bits cannot be fully utilized for accurate representation of other values.

Based on this analysis, we visualize the channel-range of these four different operators, and the channel-range of tokens without class token, as well as the activation of class token.

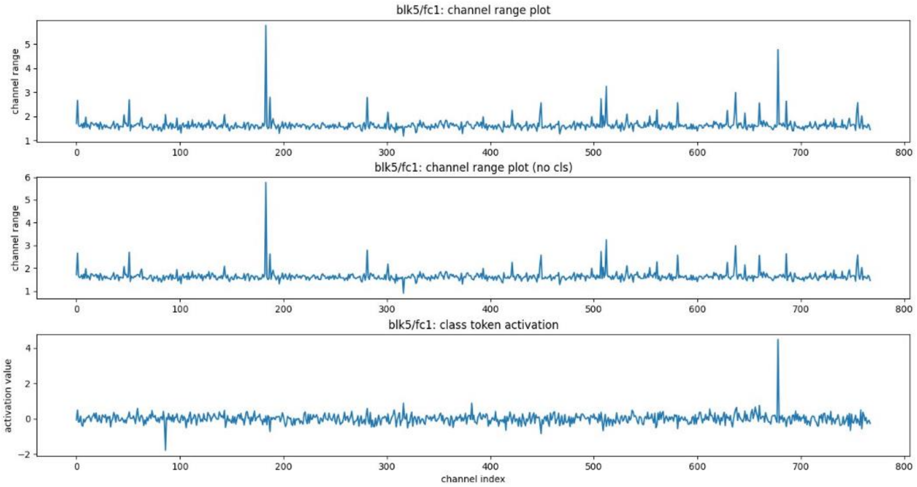


*Fig. 8: Channel range plot of all tokens (top), without class token (mid), and class token activation values (bottom) , for 'fc1 MatMul + Add' operator in block 5 of ViT*

Fig. 8 shows an example of the input activations of Block5/MLP/FC1 layer. Some channels have a high range of values, a key parameter in minmax quantization, compared to most of the other channels. As the difference is no more than 5, it is acceptable as reflected in the results of our experiments. However, as the maximum channel range increases, what we expect is that either there are very few channels with the extreme range values (because they can be addressed by the percentile strategy of per-tensor quantization) or the channel range values of different channels fluctuate around a not very high constant, which is the case of the class token activation in Fig. 8. In this case, the min-max range would be around 4 if we use per-tensor quantization, but the 8-bit capacity may ensure that even if the maximum range of class token is around 1.5, it can still be described smoothly.

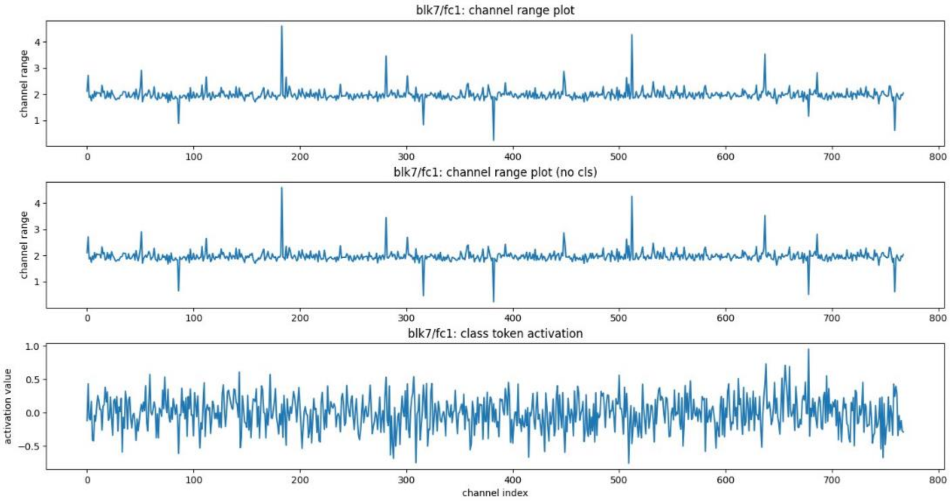


*Fig. 9: Channel range plot of all tokens (top), without class token (mid), and class token activation values (bottom) , for 'fc1 MatMul + Add' operator in block 7 of ViT*

However, through all our channel-range visualization, we discover that in the FC2 operator of Block 5 and Block 6, the maximum channel range values are the largest, around 30 and 40. In all other visualization, this value is no more than 15. While no distinct patterns can be observed from these channel-range visualizations, it is clear from Block5/FC2 and Block6/FC2 that these two operators play an important role in quantization performance.

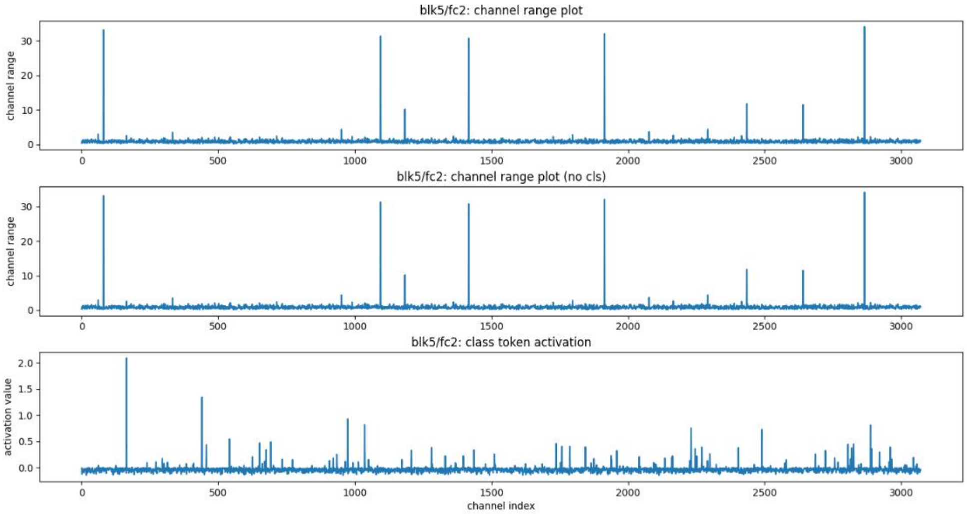


*Fig. 10: Channel range plot of all tokens (top), without class token (mid), and class token activation values (bottom), for 'fc2 MatMul + Add' operator in block 5 of ViT*

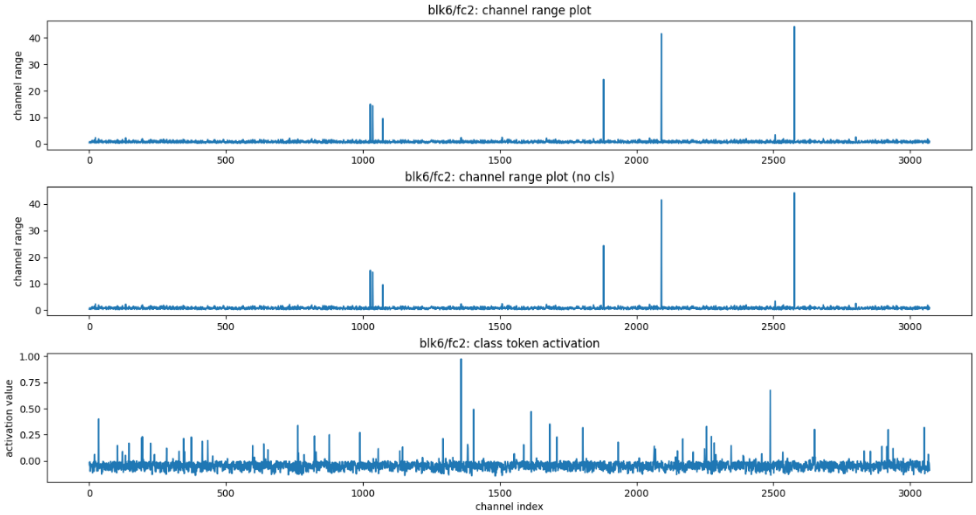


*Fig. 11: Channel range plot of all tokens (top), without class token (mid), and class token activation values (bottom), for 'fc2 MatMul + Add' operator in block 6 of ViT*

To gain further insights, we perform error analysis in the next section.

### 4.3.2. Quantization Error Analysis

In this section, we analyzed the quantization error to gain more insights on channel activations. Due to the existence of the attention mechanism, the quantization error of other tokens in a preceding attention stack will lead to the deviation of class token in the following stack, It is thus important to focus on the class token. If the quantization error of the class token in a certain layer is large after quantization, it can have a significant degradation on the performance of the model. Based on the previous visualization, we chose to analyze the quantization error of class tokens and other tokens separately to find out more insights. The below figures show the average MSE through blocks error for four operators for both class token and rest (patch) tokens.

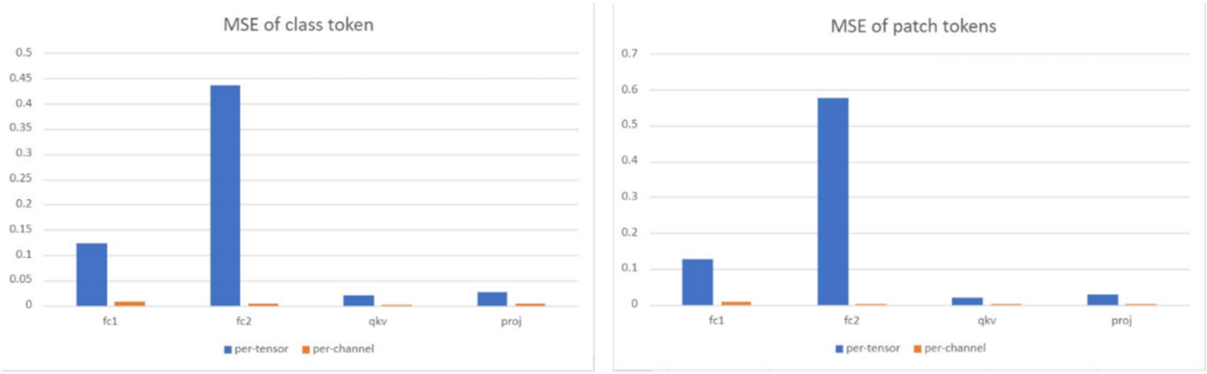


*Fig. 12: The average MSE error of class token and patch toekns through all 12 blocks.*

Fig. 12 shows that the errors of FC2 operator is much higher, and performing per-channel quantization reduces the MSE to the least value. The results of relevant experiments of calculating MSE are given in Appendix 7.4.

The metric of MSE is about absolute values, as is mentioned, we add a new metric, which is the signal-noise ratio for measuring the percentage error of quantization. This is in the case of the situation that sometimes the MSE could be large, but if the channel has a quite big range, then actually the effect of the error is not obvious. Therefore, we believe it is more objective to add a new criterion for measuring quantization errors.

The snr (signal-noise ratio) is calculated by:

$$\frac{1}{N}\cdot\sum_{N}\frac{(quant\ -\ fp)^{\ 2}}{fp^{2}+\varepsilon} \qquad (1)$$

N: number of points in the activations,
quant: value after dequantization,
fp: original value,
$\varepsilon$ : constant value to prevent the denominator from being 0.

The visualization of snr error is shown in the figure below:

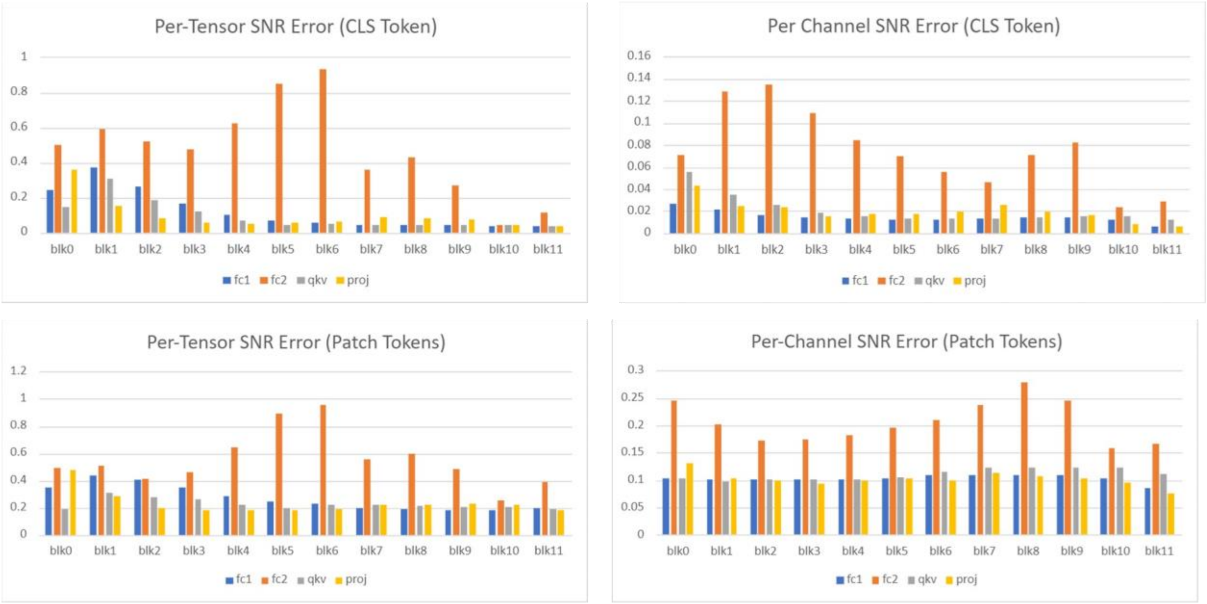


*Fig. 13: Noise-signal ratio for both class token and patch tokens of all 12 blocks for 'fc1 MatMul + Add', 'fc2 MatMul + Add', 'qkv MatMul + Add', and 'proj MatMul + Add' operators, using per-tensor and per-channel quantization.*

Per-tensor error critical because it intuitively shows the severity of inter-channel variation in different operators. Generally, FC2 suffers the most from the inter-channel variation, as the shown previously in our ablation study, and activation visualization. An interesting finding is that FC2 of Block 5 and Block 6 seem to have the largest per-tensor SNR error, while we have shown that some channels of FC2 in Block 5 and Block 6 have the largest channel range value among all situations. Therefore, we can think that the maximum channel range value will have a serious impact on quantization when it rises to a certain extent. In addition, this could happen because unlike other operators, FC2 inputs have a high dimensional feature activated by GELU. The dimension of the feature vector in the MLP block in ViT corresponds to the channels of the image feature, which may cause the channel range value of the FC2 activation value to be different (larger) from other fused MatMul operators.

Since we have discovered that the maximum channel range of Block5/FC2 and Block6/FC2 have significantly larger values than that of other operators, we customize our quantizer and only perform per-channel quantization on these two blocks. The quantized model's performance shows a significant improvement.

| | **Full-precision** | **Quant-normal** | **Quant-b5&6.fc2-c** |
|---|---|---|---|
| top1 | 75.850 | 64.060 | 70.970 |
| top5 | 92.990 | 84.960 | 89.580 |

*Table 5: Model performance comparison, Full-precision is the original full-precision model; Quant-normal is fully-quantized model using channel-wise quantization on only LayerNorm; Quant-b5&6.fc2-c is the fully-quantized model using channel-wise quantization on LayerNorm, and the fused "fc2-MatMul + Add" operator of block 5 and block 6*

### 4.3.3. Error Distribution

Some other visualizations are also performed as an argument for "FC2 is the bottleneck operator among all fused MatMul + Add operators".

The figure below shows the average percentage error of quantization of class token in histogram, in Block5/FC2 and Block6/FC2. The x-axis stands for the error and the y-axis means the number of channels in each bin. As is shown, the FC2 operator has the most channels in high-error bins. (Also noted that the channels of FC2 is 4 times that of the other three operators, but it can be intuitively seen that a larger portion of channels of FC2 lies in the high-error bins). Other visualizations of such histogram can be found in Appendix 7.6.

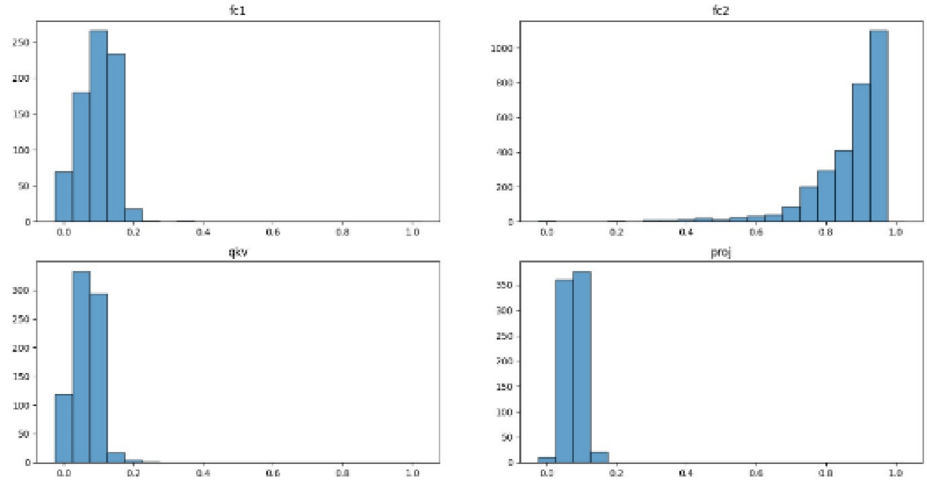


*Fig. 14 The histogram of percentage error (channel-wise) in class token on block 5 of Vision Transformer*

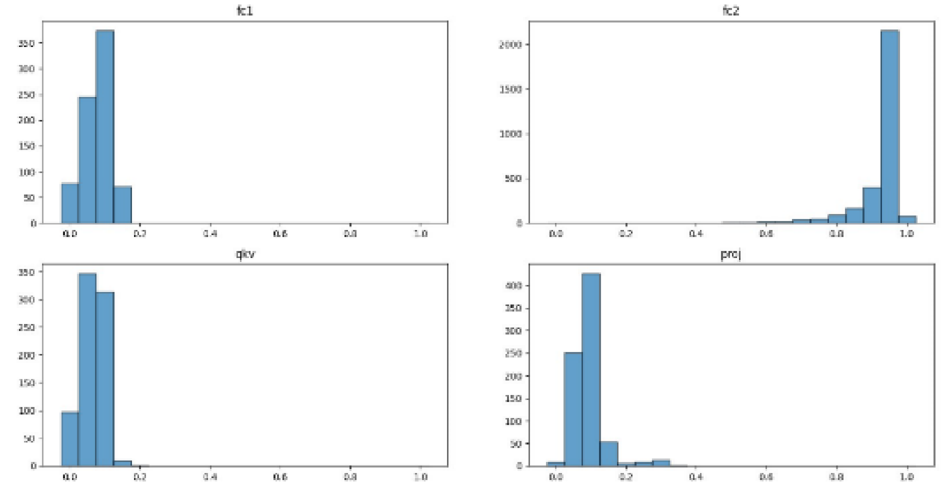


*Fig. 15 The histogram of percentage error (channel-wise) in class token on block 6 of Vision Transformer*

### 4.3.4. Attention Map Analysis

In addition to the visualization of the activations and errors, we visualize the attention map to have a more intuitive comparison among different quantization tactics. The figures below compare the visualization of attention maps on original images using the original model and quantized ones, showing how the attention mechanism perceives images.

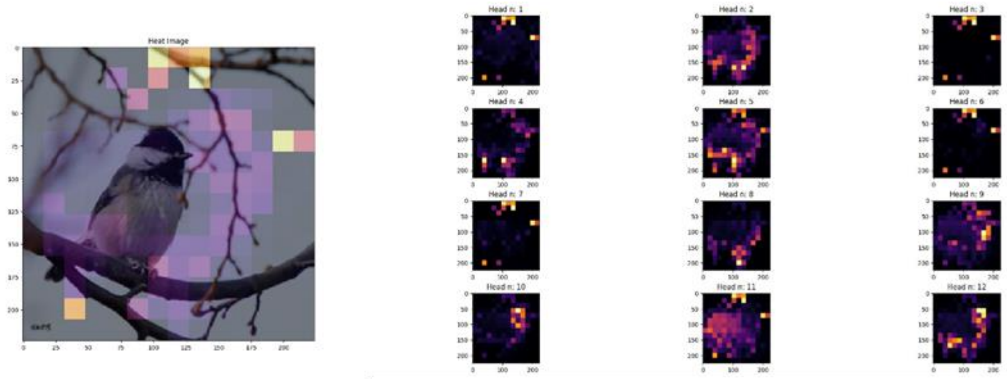


*Fig. 16: Attention map visualization (full precision model)*

As shown in case of Fig. 17, where only LayerNorm operators are per-channel quantized, while all others are under per-tensor quantization. The focusing area seems to shift from the bird. In comparison, Fig. 18, where we customize and perform channel-wise quantization on the fc2 operators of block 5 and block6, the focusing area is still around the bird. A few more samples of attention map visualization are given in the Appendix.7.7

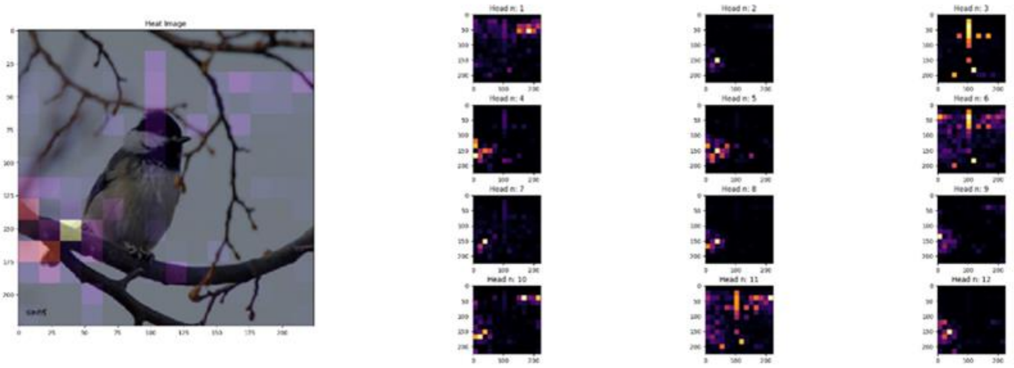


*Fig. 17: Attention map visualization (vanilla quantized model)*

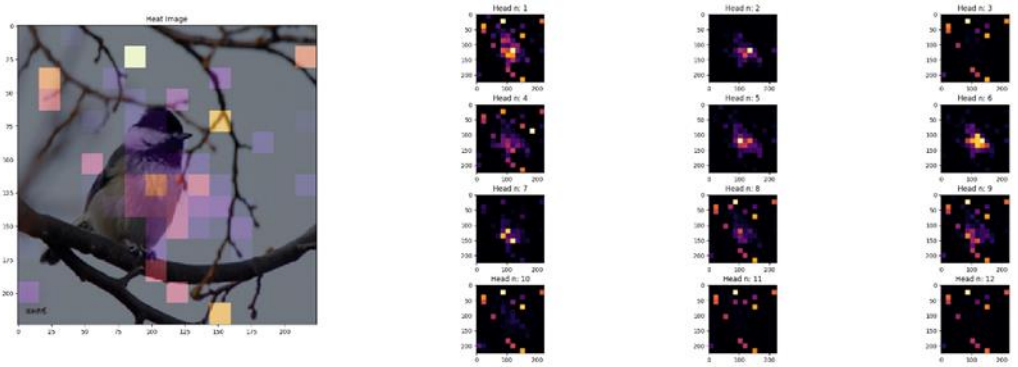


*Fig. 18: Attention map visualization (our customized quantizer)*

## 5. Conclusion and Recommendations

### 5.1. Summary of Findings

In this section, we provide a summary of the key findings from our experiments on different approaches to PTQ of ViT.

- PPQ is a key tool that we have used in our experiments. Through the graph fusion methods implemented by PPQ, we were able to focus our efforts on analyzing a few layers we believe to be critical to the quantization efforts to uncover the insights.
- Our ablation study on the key layers of Proj, QKV, LayerNorm, FC1 and FC2 shows that LayerNorm is a critical layer that affects PTQ performance. Additionally, the MLP

block should not be ignored in PTQ, specifically FC2, as it can determine some significant gains on model performance.

- By running our quantized model strategy on a subset of Imagenet2012, we observed that our quantized DeiT-Tiny, -Small, -Base, and ViT-Base models yield competitive results against their full precision counterparts.
- By conducting a deeper analysis into the channel value distribution and channel range distribution of the MLP blocks, we uncovered that the FC1 channel value distribution is heavily influenced by the class token, but less so for FC2.
- We performed error analysis on the quantization values to determine the why a channel-wise quantization strategy worked better for FC2 but affects FC1 less. Results show that errors were much higher in FC2, especially Block5 and Block6. Further analysis of the error distribution confirms our suspicion.
- Lastly, we demonstrated the attention maps of the full precision model, the vanilla quantization method and our strategy to visualize how the attention map perceives images across different strategies. Our quantized method was able to capture the essence of the attention map better than the vanilla method when compared against the full precision method.

### 5.2. Challenges and Limitations

- The overhead of hardware remains an issue. The purpose of performing post-training quantization is to reduce storage space and computing resource requirements for neural network models, while making all computations at fixed precision level, such that model can be deployed on hardware that do not support floating computations. However, though channel-wise quantization maintains the model’s performance, it also needs more resources for the computing process of quantization, increasing inference time and consuming more energy. The ideal situation would be to maintain the model’s performance while minimizing the use of channel-wise quantization.

- Over the phase of this project, our GPU resource is quite limited, which makes it challenging to conduct experiments on a large scale. Perhaps, in the future, more experiments can be done to validate the theories that we have presented.
- The design of our own customized quantizer is not flexible at code level. If we want to use our quantizer on other transformer-based models, we need to make some adjustments to our conditions for searching different operators.

### 5.3. Future Work

While we have implemented a quantization strategy using channel-wise quantization for FC2 in the MLP block, this has been tested on only the DeiT and ViT models. It would be useful to validate this strategy on other transformer-based vision transformers, such as the Swin Transformer.

Considerable efforts through literature review have been down to uncover some of the more promising research on PTQ for ViT. Similar to the efforts of "*A White Paper on Neural Network Quantization*" (Nagel et al., 2021), we believe that this provides the foundation for a white paper specific to the PTQ techniques of ViT. To the best of our knowledge, this has not been done as existing quantization methods are not readily applicable to transformer-based models.

Lastly, while considerable efforts have been made to realize a competitive quantization method to full precision models, the computational cost of our quantizing method can still be quite significant. This concern cannot be ignored in the pursuit of truly sustainable AI. We believe that adaptive methods to a grouped channel-wise quantization may help to speed up inference time of our model by reducing costs.

## 6. References

Bahdanau, D., Cho, K., & Bengio, Y. (2014). Neural Machine Translation by Jointly Learning to Align and Translate. *CoRR*, *abs/1409.0473*.

Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T. J., Child, R., Ramesh, A., Ziegler, D. M., Wu, J., Winter, C., . . . Amodei, D. (2020). Language Models are Few-Shot Learners. *ArXiv*, *abs/2005.14165*.

Chen, H., Wang, Y., Xu, C., Yang, Z., Liu, C., Shi, B., Xu, C., Xu, C., & Tian, Q. (2019). Data-Free Learning of Student Networks. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, 3513-3521.

Cubuk, E. D., Zoph, B., Mané, D., Vasudevan, V., & Le, Q. V. (2018). AutoAugment: Learning Augmentation Policies from Data. *ArXiv*, *abs/1805.09501*.

Cubuk, E. D., Zoph, B., Shlens, J., & Le, Q. V. (2019). Randaugment: Practical automated data augmentation with a reduced search space. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 3008-3017.

Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2019). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *ArXiv*, *abs/1810.04805*.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., & Houlsby, N. (2020). An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. *ArXiv*, *abs/2010.11929*.

Frumkin, N., Gope, D., & Marculescu, D. (2022). CPT-V: A Contrastive Approach to Post-Training Quantization of Vision Transformers. *ArXiv*, *abs/2211.09643*.

Gholami, A., Kim, S., Dong, Z., Yao, Z., Mahoney, M. W., & Keutzer, K. (2021). A Survey of Quantization Methods for Efficient Neural Network Inference. *ArXiv*, *abs/2103.13630*.

Gholami, A., Mahoney, M. W., & Keutzer, K. (2020). An integrated approach to neural network design, training, and inference. *Univ. California, Berkeley, Berkeley, CA, USA, Tech. Rep*.

Gray, R. M., & Neuhoff, D. L. (1998). Quantization. *IEEE Transactions on Information Theory*, *44*(6), 2325-2383. https://doi.org/10.1109/18.720541

Kim, Y., Denton, C., Hoang, L., & Rush, A. M. (2017). Structured Attention Networks. *ArXiv*, *abs/1702.00887*.

Li, Z., Chen, M., Xiao, J., & Gu, Q. (2022). PSAQ-ViT V2: Towards Accurate and General Data-Free Quantization for Vision Transformers. *ArXiv*, *abs/2209.05687*.

Li, Z., Ma, L., Chen, M., Xiao, J., & Gu, Q. (2022). Patch Similarity Aware Data-Free Quantization for Vision Transformers. *ArXiv*, *abs/2203.02250*.

Li, Z., Xiao, J., Yang, L., & Gu, Q. (2022). RepQ-ViT: Scale Reparameterization for Post-Training Quantization of Vision Transformers. *ArXiv*, *abs/2212.08254*.

Lin, Y., Zhang, T., Sun, P., Li, Z., & Zhou, S. (2021). FQ-ViT: Post-Training Quantization for Fully Quantized Vision Transformer. International Joint Conference on Artificial Intelligence,

Liu, Y., Yang, H., Dong, Z., Keutzer, K., Du, L., & Zhang, S. (2022). NoisyQuant: Noisy Bias-Enhanced Post-Training Activation Quantization for Vision Transformers. *ArXiv*, *abs/2211.16056*.

Liu, Z., Hu, H., Lin, Y., Yao, Z., Xie, Z., Wei, Y., Ning, J., Cao, Y., Zhang, Z., Dong, L., Wei, F., & Guo, B. (2021). Swin Transformer V2: Scaling Up Capacity and Resolution. *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 11999-12009.

Liu, Z., Wang, Y., Han, K., Ma, S., & Gao, W. (2021). Post-Training Quantization for Vision Transformer. Neural Information Processing Systems,

Mehta, S., & Rastegari, M. (2021). MobileViT: Light-weight, General-purpose, and Mobile-friendly Vision Transformer. *ArXiv*, *abs/2110.02178*.

Nagel, M., Amjad, R. A., Baalen, M. v., Louizos, C., & Blankevoort, T. (2020). Up or Down? Adaptive Rounding for Post-Training Quantization. *ArXiv*, *abs/2004.10568*.

Nagel, M., Fournarakis, M., Amjad, R. A., Bondarenko, Y., Baalen, M. v., & Blankevoort, T. (2021). A White Paper on Neural Network Quantization. *ArXiv*, *abs/2106.08295*.

Przewlocka-Rus, D., Sarwar, S. S., Sumbul, H. E., Li, Y., & Salvo, B. d. (2022). Power-of-Two Quantization for Low Bitwidth and Hardware Compliant Neural Networks. *ArXiv*, *abs/2203.05025*.

Radosavovic, I., Kosaraju, R. P., Girshick, R. B., He, K., & Dollár, P. (2020). Designing Network Design Spaces. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 10425-10433.

SenseTime. (2023). PPL Quantization Tool *GitHub repository*. https://github.com/openppl-public/ppq

Touvron, H., Cord, M., Douze, M., Massa, F., Sablayrolles, A., & J'egou, H. e. (2020). Training data-efficient image transformers & distillation through attention. International Conference on Machine Learning,

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L. u., & Polosukhin, I. (2017). *Attention is All you Need* https://proceedings.neurips.cc/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf

Wightman, R. (2019). PyTorch Image Models. *GitHub repository*. https://doi.org/10.5281/zenodo.4414861

Wu, H., Judd, P., Zhang, X., Isaev, M., & Micikevicius, P. (2020). Integer Quantization for Deep Learning Inference: Principles and Empirical Evaluation.

Xiao, T., Singh, M., Mintun, E., Darrell, T., Dollár, P., & Girshick, R. B. (2021). Early Convolutions Help Transformers See Better. Neural Information Processing Systems,

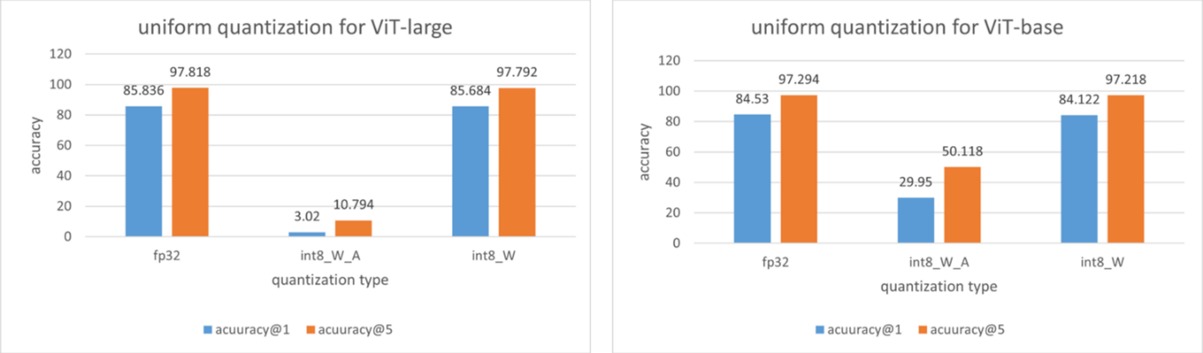
Yuan, Z., Xue, C., Chen, Y., Wu, Q., & Sun, G. (2022). *PTQ4ViT: Post-Training Quantization For Vision Transformers With Twin Uniform Quantization* Computer Vision – ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XII, https://doi.org/10.1007/978-3-031-19775-8_12

Zhao, R., Hu, Y., Dotzel, J., Sa, C. D., & Zhang, Z. (2019). Improving Neural Network Quantization without Retraining using Outlier Channel Splitting. *ArXiv*, *abs/1901.09504*.

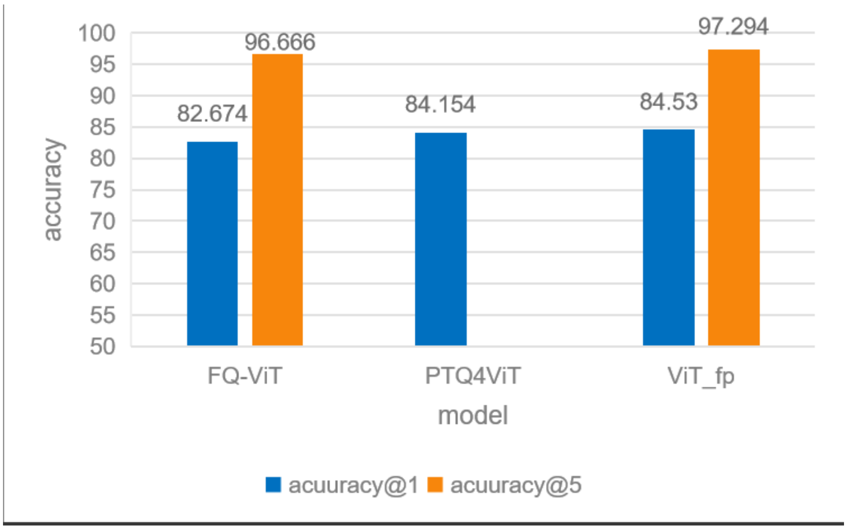
# 7. Appendix

## 7.1. Replicating Experiments of Previous Research

The dataset is different from the one used in PPQ quantization pipeline. It is a tiny subset due to the limitations of our GPU.

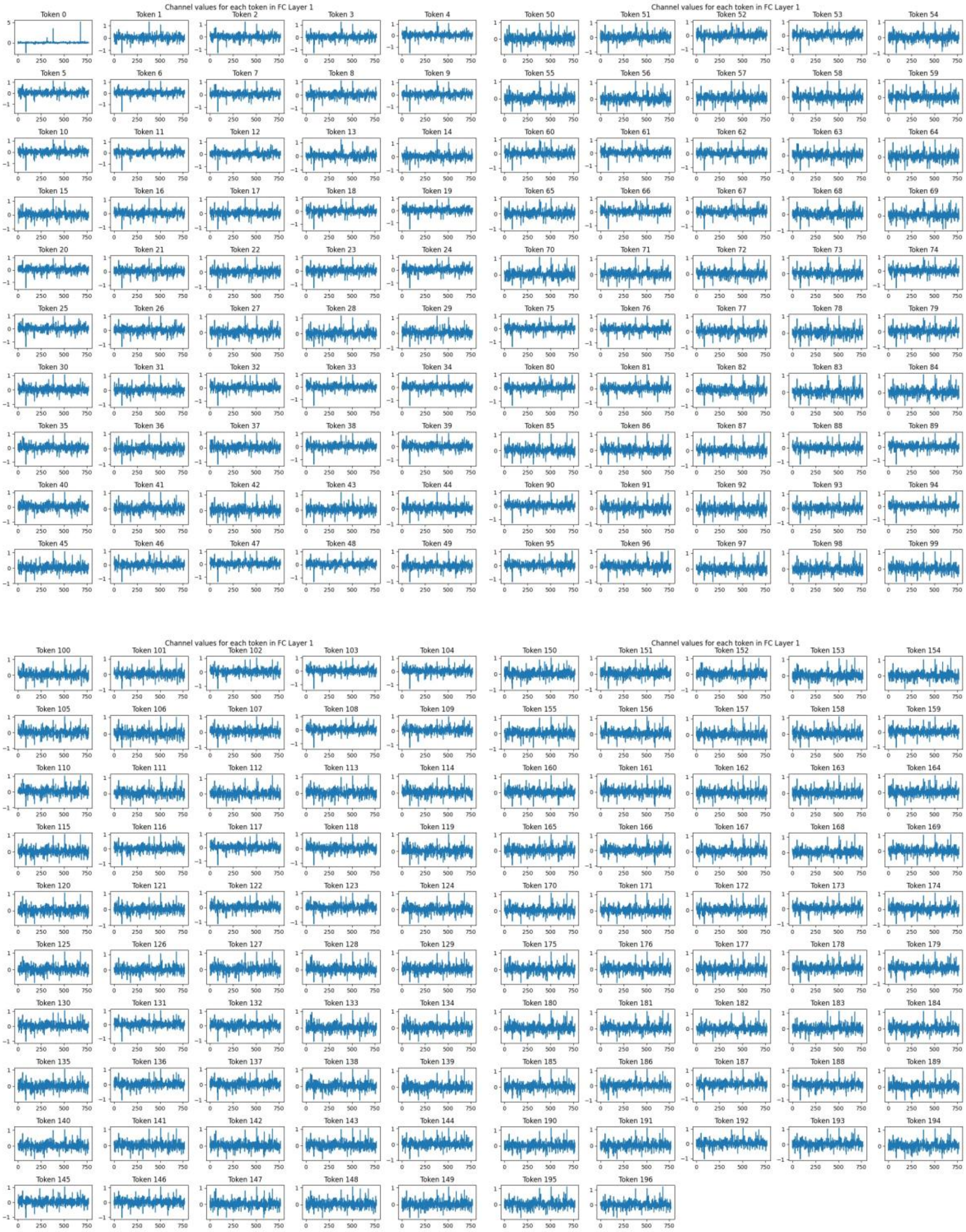


*Uniform quantization on ViT models, fp32 means the original full-precision models that are not quantized, int8_W means to perform 8-bits weights-only quantization, int8_W_A means to perform 8-bits quantization on all activations and weights. accuracy@1 and accuracy@5 refers to top1 and top5 accuracy. Validation data is 50000 samples from ILSVRC2012*

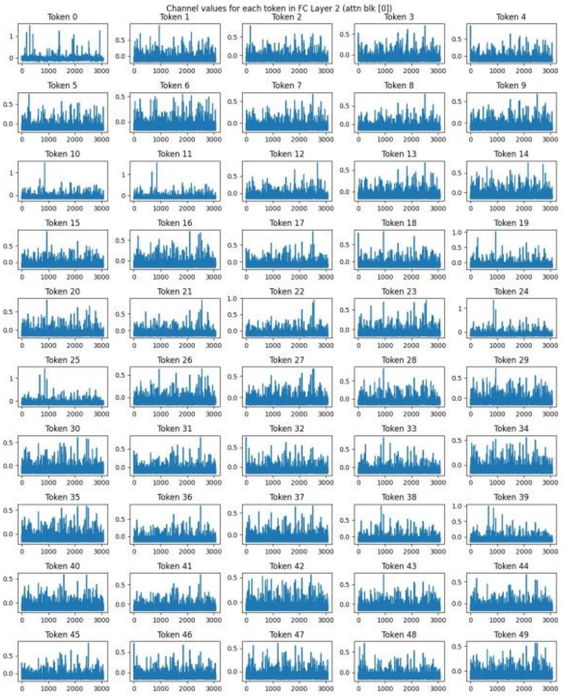


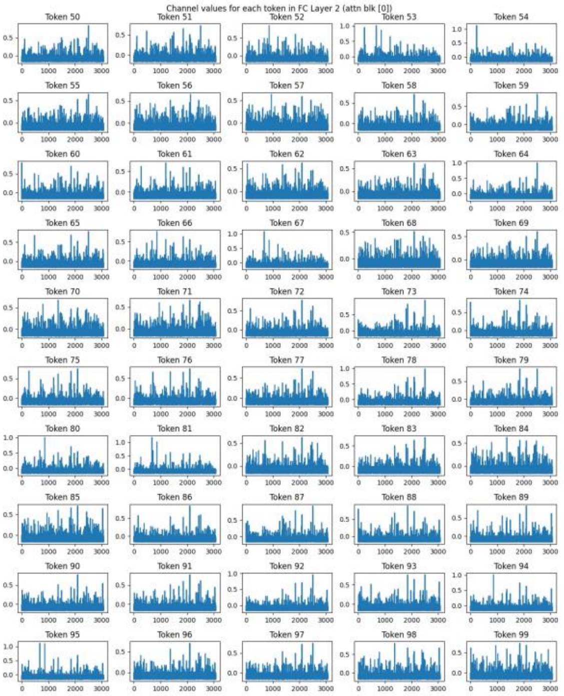
*Comparison of some of models we reproduce (FQ-ViT and PTQ4ViT) with the original full-precision model.*

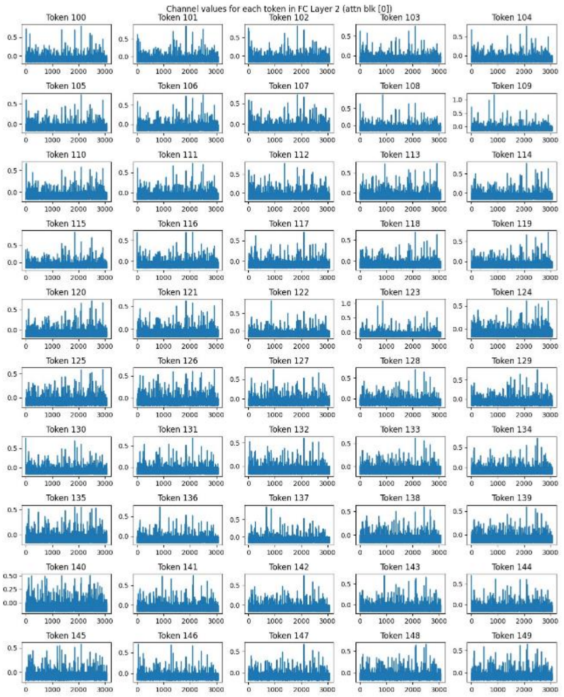
## 7.2. Class Tokens and Patch Tokens of FC1 & FC2

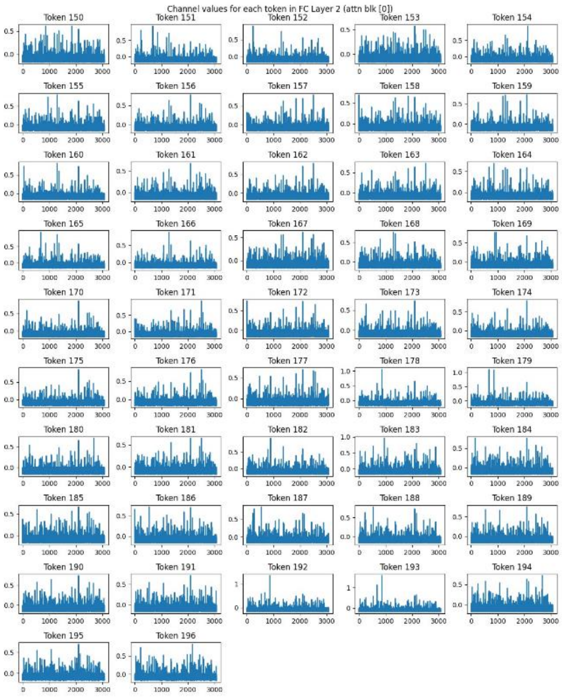


This figure shows the activation values of all 197 tokens in the first attention block. We observed that the activation profile of the first token, representing the class token, is distinct from the rest of the tokens (patch tokens).





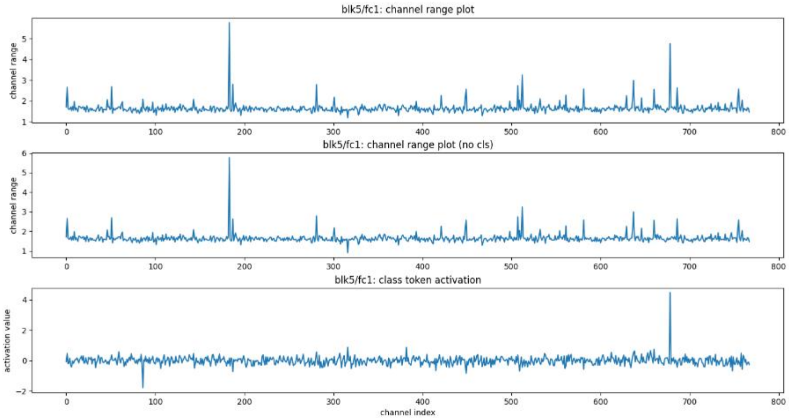


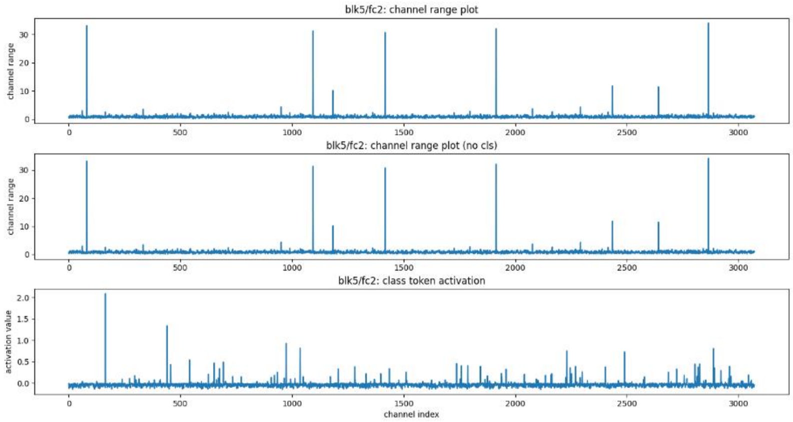


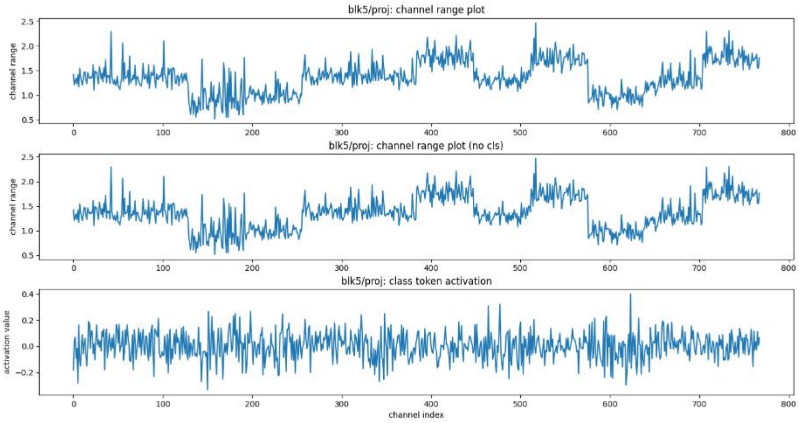
The same patten does not seem to occur in FC2. This is likely due to the change in embedding dimensions, increasing the number of channels from 768 to 3072 from FC1 to FC2. Thus, resulting in a change in the channel values of the class token

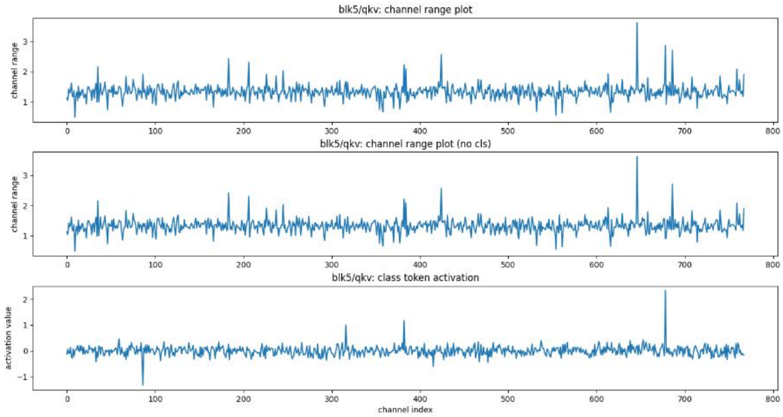
### 7.3. Channel Range and Activation Values of Tokens

The following figures show the channel range of class and patch tokens as well as the activation values of the class token in FC1, FC2, QKV and Proj MatMul + Add in attention Block 5 and 6.





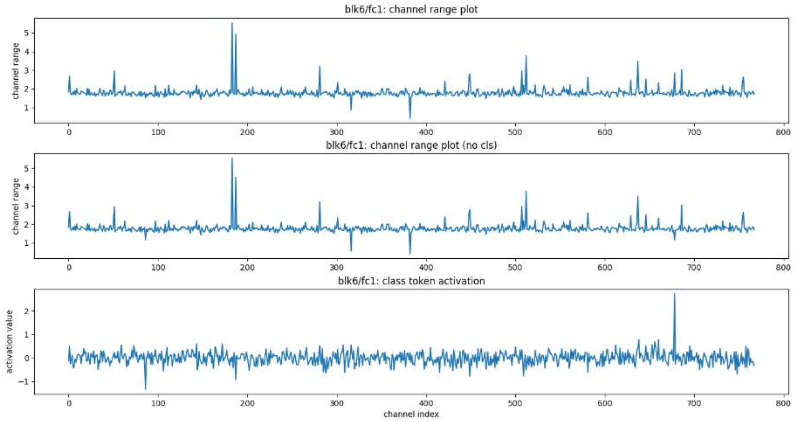




blk5/qkv: channel range plot

blk5/qkv: channel range plot (no cls)

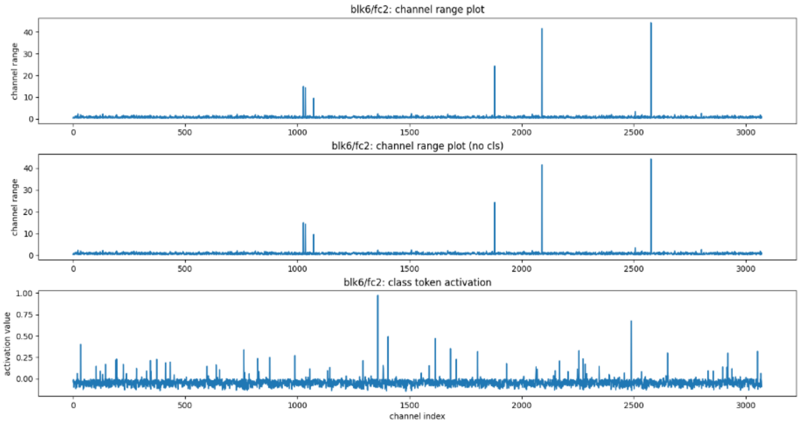
blk5/qkv: class token activation



blk6/fc1: channel range plot

blk6/fc1: channel range plot (no cls)

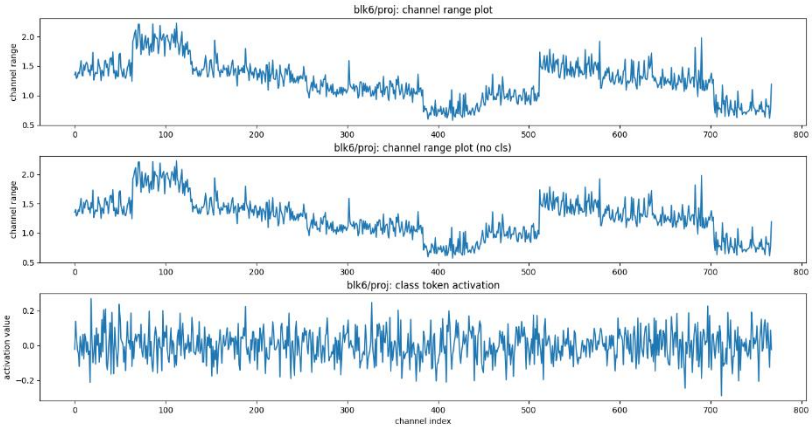
blk6/fc1: class token activation



blk6/fc2: channel range plot

blk6/fc2: channel range plot (no cls)

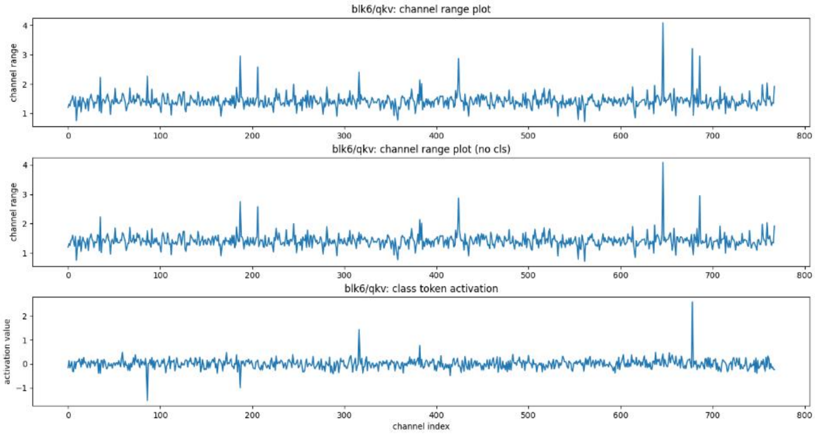
blk6/fc2: class token activation



blk6/proj: channel range plot

blk6/proj: channel range plot (no cls)

blk6/proj: class token activation



blk6/qkv: channel range plot

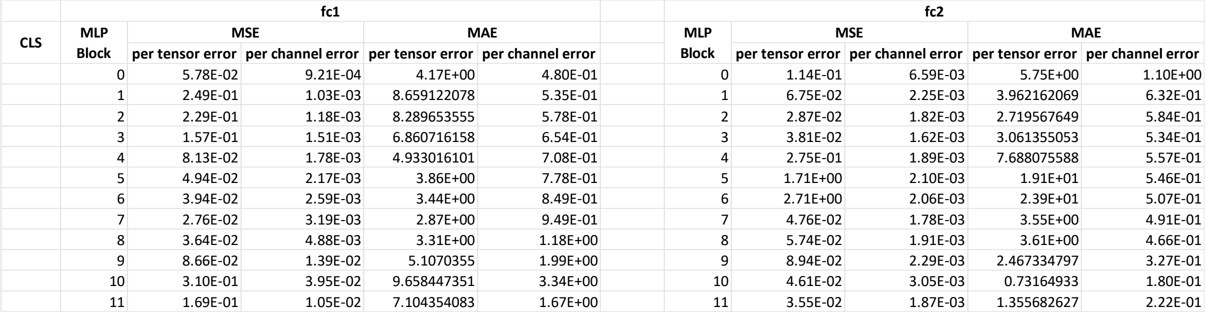
blk6/qkv: channel range plot (no cls)

blk6/qkv: class token activation

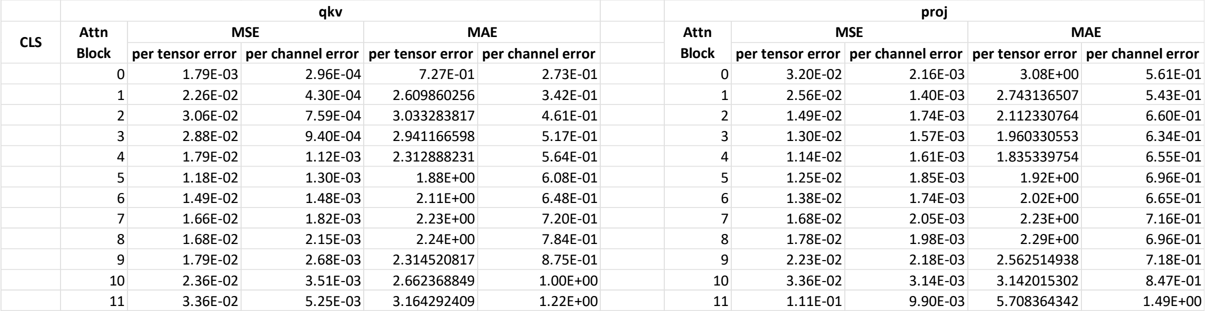
## 7.4. MSE and MAE of class token and patch tokens,

The following figures show the MSE and MAE of class token and patch tokens using per-tensor and per-channel quantization.

### 7.4.1. Class token’s MSE and MAE for quantization

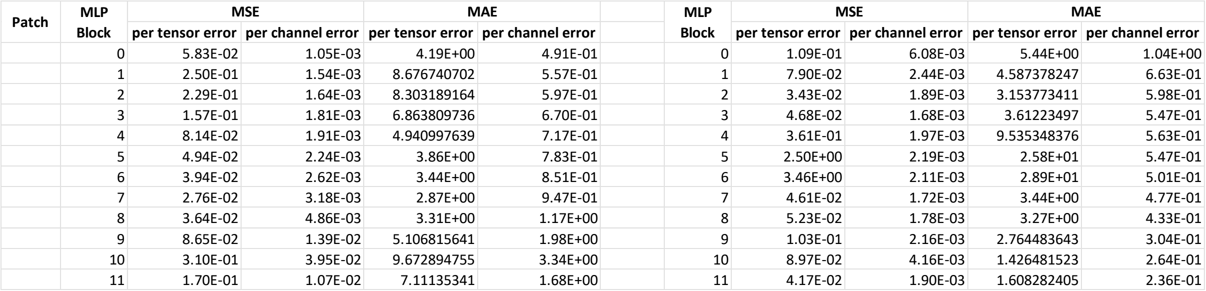


| | fc1 | | | | | | fc2 | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| CLS | MLP | MSE | | MAE | | | MLP | MSE | | MAE | |
| | Block | per tensor error | per channel error | per tensor error | per channel error | | Block | per tensor error | per channel error | per tensor error | per channel error |
| | 0 | 5.78E-02 | 9.21E-04 | 4.17E+00 | 4.80E-01 | | 0 | 1.14E-01 | 6.59E-03 | 5.75E+00 | 1.10E+00 |
| | 1 | 2.49E-01 | 1.03E-03 | 8.659122078 | 5.35E-01 | | 1 | 6.75E-02 | 2.25E-03 | 3.962162069 | 6.32E-01 |
| | 2 | 2.29E-01 | 1.18E-03 | 8.289653555 | 5.78E-01 | | 2 | 2.87E-02 | 1.82E-03 | 2.719567649 | 5.84E-01 |
| | 3 | 1.57E-01 | 1.51E-03 | 6.860716158 | 6.54E-01 | | 3 | 3.81E-02 | 1.62E-03 | 3.061355053 | 5.34E-01 |
| | 4 | 8.13E-02 | 1.78E-03 | 4.933016101 | 7.08E-01 | | 4 | 2.75E-01 | 1.89E-03 | 7.688075588 | 5.57E-01 |
| | 5 | 4.94E-02 | 2.17E-03 | 3.86E+00 | 7.78E-01 | | 5 | 1.71E+00 | 2.10E-03 | 1.91E+01 | 5.46E-01 |
| | 6 | 3.94E-02 | 2.59E-03 | 3.44E+00 | 8.49E-01 | | 6 | 2.71E+00 | 2.06E-03 | 2.39E+01 | 5.07E-01 |
| | 7 | 2.76E-02 | 3.19E-03 | 2.87E+00 | 9.49E-01 | | 7 | 4.76E-02 | 1.78E-03 | 3.55E+00 | 4.91E-01 |
| | 8 | 3.64E-02 | 4.88E-03 | 3.31E+00 | 1.18E+00 | | 8 | 5.74E-02 | 1.91E-03 | 3.61E+00 | 4.66E-01 |
| | 9 | 8.66E-02 | 1.39E-02 | 5.1070355 | 1.99E+00 | | 9 | 8.94E-02 | 2.29E-03 | 2.467334797 | 3.27E-01 |
| | 10 | 3.10E-01 | 3.95E-02 | 9.658447351 | 3.34E+00 | | 10 | 4.61E-02 | 3.05E-03 | 0.73164933 | 1.80E-01 |
| | 11 | 1.69E-01 | 1.05E-02 | 7.104354083 | 1.67E+00 | | 11 | 3.55E-02 | 1.87E-03 | 1.355682627 | 2.22E-01 |

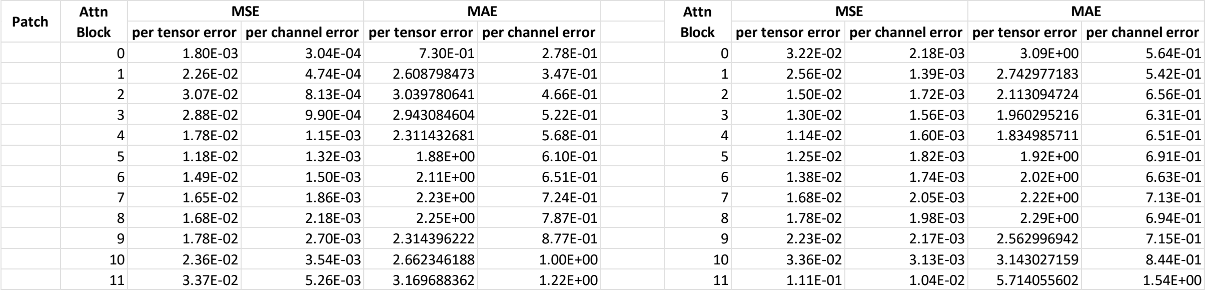


| | qkv | | | | | | proj | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| CLS | Attn | MSE | | MAE | | | Attn | MSE | | MAE | |
| | Block | per tensor error | per channel error | per tensor error | per channel error | | Block | per tensor error | per channel error | per tensor error | per channel error |
| | 0 | 1.79E-03 | 2.96E-04 | 7.27E-01 | 2.73E-01 | | 0 | 3.20E-02 | 2.16E-03 | 3.08E+00 | 5.61E-01 |
| | 1 | 2.26E-02 | 4.30E-04 | 2.609860256 | 3.42E-01 | | 1 | 2.56E-02 | 1.40E-03 | 2.743136507 | 5.43E-01 |
| | 2 | 3.06E-02 | 7.59E-04 | 3.033283817 | 4.61E-01 | | 2 | 1.49E-02 | 1.74E-03 | 2.112330764 | 6.60E-01 |
| | 3 | 2.88E-02 | 9.40E-04 | 2.941166598 | 5.17E-01 | | 3 | 1.30E-02 | 1.57E-03 | 1.960330553 | 6.34E-01 |
| | 4 | 1.79E-02 | 1.12E-03 | 2.312888231 | 5.64E-01 | | 4 | 1.14E-02 | 1.61E-03 | 1.835339754 | 6.55E-01 |
| | 5 | 1.18E-02 | 1.30E-03 | 1.88E+00 | 6.08E-01 | | 5 | 1.25E-02 | 1.85E-03 | 1.92E+00 | 6.96E-01 |
| | 6 | 1.49E-02 | 1.48E-03 | 2.11E+00 | 6.48E-01 | | 6 | 1.38E-02 | 1.74E-03 | 2.02E+00 | 6.65E-01 |
| | 7 | 1.66E-02 | 1.82E-03 | 2.23E+00 | 7.20E-01 | | 7 | 1.68E-02 | 2.05E-03 | 2.23E+00 | 7.16E-01 |
| | 8 | 1.68E-02 | 2.15E-03 | 2.24E+00 | 7.84E-01 | | 8 | 1.78E-02 | 1.98E-03 | 2.29E+00 | 6.96E-01 |
| | 9 | 1.79E-02 | 2.68E-03 | 2.314520817 | 8.75E-01 | | 9 | 2.23E-02 | 2.18E-03 | 2.562514938 | 7.18E-01 |
| | 10 | 2.36E-02 | 3.51E-03 | 2.662368849 | 1.00E+00 | | 10 | 3.36E-02 | 3.14E-03 | 3.142015302 | 8.47E-01 |
| | 11 | 3.36E-02 | 5.25E-03 | 3.164292409 | 1.22E+00 | | 11 | 1.11E-01 | 9.90E-03 | 5.708364342 | 1.49E+00 |

### 7.4.2. Patch tokens MSE and MAE for quantization

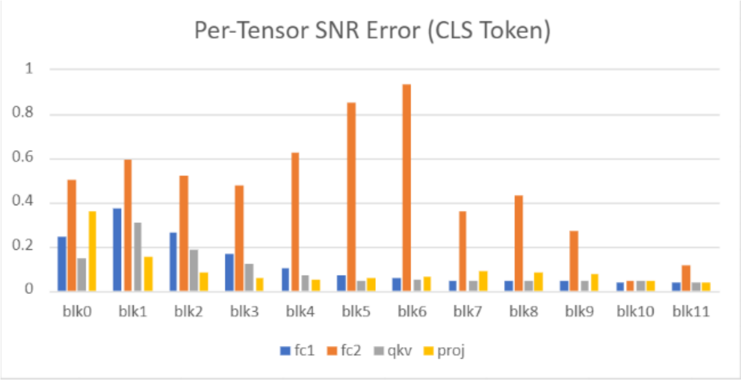


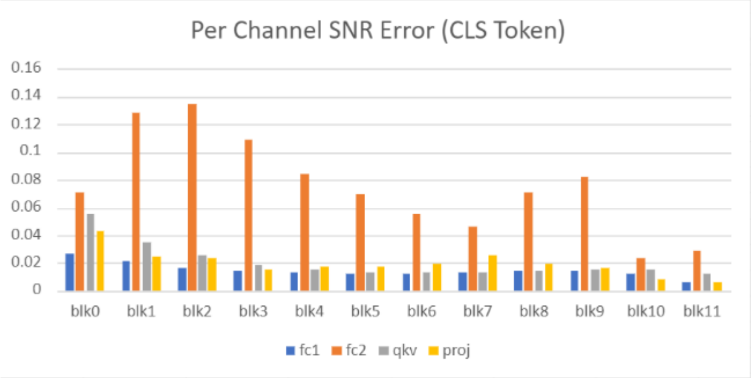
| Patch | MLP | MSE | | MAE | | | MLP | MSE | | MAE | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | Block | per tensor error | per channel error | per tensor error | per channel error | | Block | per tensor error | per channel error | per tensor error | per channel error |
| | 0 | 5.83E-02 | 1.05E-03 | 4.19E+00 | 4.91E-01 | | 0 | 1.09E-01 | 6.08E-03 | 5.44E+00 | 1.04E+00 |
| | 1 | 2.50E-01 | 1.54E-03 | 8.676740702 | 5.57E-01 | | 1 | 7.90E-02 | 2.44E-03 | 4.587378247 | 6.63E-01 |
| | 2 | 2.29E-01 | 1.64E-03 | 8.303189164 | 5.97E-01 | | 2 | 3.43E-02 | 1.89E-03 | 3.153773411 | 5.98E-01 |
| | 3 | 1.57E-01 | 1.81E-03 | 6.863809736 | 6.70E-01 | | 3 | 4.68E-02 | 1.68E-03 | 3.61223497 | 5.47E-01 |
| | 4 | 8.14E-02 | 1.91E-03 | 4.940997639 | 7.17E-01 | | 4 | 3.61E-01 | 1.97E-03 | 9.535348376 | 5.63E-01 |
| | 5 | 4.94E-02 | 2.24E-03 | 3.86E+00 | 7.83E-01 | | 5 | 2.50E+00 | 2.19E-03 | 2.58E+01 | 5.47E-01 |
| | 6 | 3.94E-02 | 2.62E-03 | 3.44E+00 | 8.51E-01 | | 6 | 3.46E+00 | 2.11E-03 | 2.89E+01 | 5.01E-01 |
| | 7 | 2.76E-02 | 3.18E-03 | 2.87E+00 | 9.47E-01 | | 7 | 4.61E-02 | 1.72E-03 | 3.44E+00 | 4.77E-01 |
| | 8 | 3.64E-02 | 4.86E-03 | 3.31E+00 | 1.17E+00 | | 8 | 5.23E-02 | 1.78E-03 | 3.27E+00 | 4.33E-01 |
| | 9 | 8.65E-02 | 1.39E-02 | 5.106815641 | 1.98E+00 | | 9 | 1.03E-01 | 2.16E-03 | 2.764483643 | 3.04E-01 |
| | 10 | 3.10E-01 | 3.95E-02 | 9.672894755 | 3.34E+00 | | 10 | 8.97E-02 | 4.16E-03 | 1.426481523 | 2.64E-01 |
| | 11 | 1.70E-01 | 1.07E-02 | 7.11135341 | 1.68E+00 | | 11 | 4.17E-02 | 1.90E-03 | 1.608282405 | 2.36E-01 |

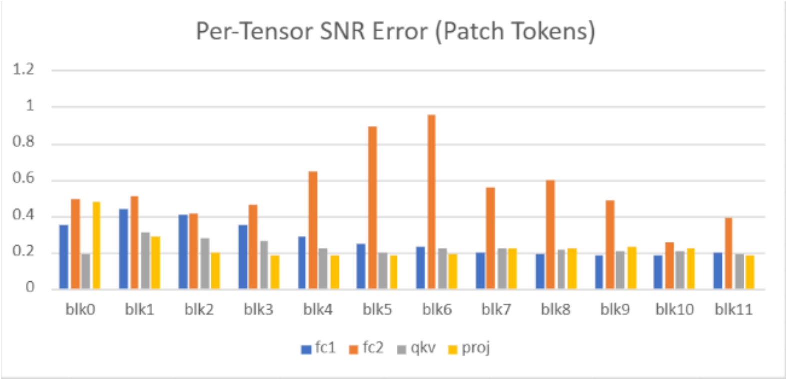


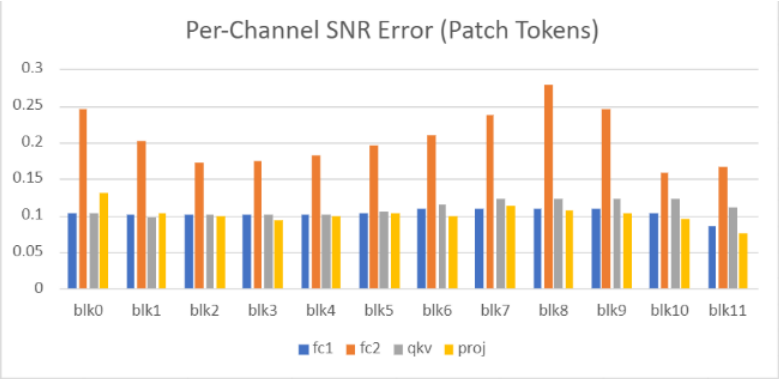
| Patch | Attn | MSE | | MAE | | | Attn | MSE | | MAE | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | Block | per tensor error | per channel error | per tensor error | per channel error | | Block | per tensor error | per channel error | per tensor error | per channel error |
| | 0 | 1.80E-03 | 3.04E-04 | 7.30E-01 | 2.78E-01 | | 0 | 3.22E-02 | 2.18E-03 | 3.09E+00 | 5.64E-01 |
| | 1 | 2.26E-02 | 4.74E-04 | 2.608798473 | 3.47E-01 | | 1 | 2.56E-02 | 1.39E-03 | 2.742977183 | 5.42E-01 |
| | 2 | 3.07E-02 | 8.13E-04 | 3.039780641 | 4.66E-01 | | 2 | 1.50E-02 | 1.72E-03 | 2.113094724 | 6.56E-01 |
| | 3 | 2.88E-02 | 9.90E-04 | 2.943084604 | 5.22E-01 | | 3 | 1.30E-02 | 1.56E-03 | 1.960295216 | 6.31E-01 |
| | 4 | 1.78E-02 | 1.15E-03 | 2.311432681 | 5.68E-01 | | 4 | 1.14E-02 | 1.60E-03 | 1.834985711 | 6.51E-01 |
| | 5 | 1.18E-02 | 1.32E-03 | 1.88E+00 | 6.10E-01 | | 5 | 1.25E-02 | 1.82E-03 | 1.92E+00 | 6.91E-01 |
| | 6 | 1.49E-02 | 1.50E-03 | 2.11E+00 | 6.51E-01 | | 6 | 1.38E-02 | 1.74E-03 | 2.02E+00 | 6.63E-01 |
| | 7 | 1.65E-02 | 1.86E-03 | 2.23E+00 | 7.24E-01 | | 7 | 1.68E-02 | 2.05E-03 | 2.22E+00 | 7.13E-01 |
| | 8 | 1.68E-02 | 2.18E-03 | 2.25E+00 | 7.87E-01 | | 8 | 1.78E-02 | 1.98E-03 | 2.29E+00 | 6.94E-01 |
| | 9 | 1.78E-02 | 2.70E-03 | 2.314396222 | 8.77E-01 | | 9 | 2.23E-02 | 2.17E-03 | 2.562996942 | 7.15E-01 |
| | 10 | 2.36E-02 | 3.54E-03 | 2.662346188 | 1.00E+00 | | 10 | 3.36E-02 | 3.13E-03 | 3.143027159 | 8.44E-01 |
| | 11 | 3.37E-02 | 5.26E-03 | 3.169688362 | 1.22E+00 | | 11 | 1.11E-01 | 1.04E-02 | 5.714055602 | 1.54E+00 |

## 7.5. SNR error for class token and patch tokens



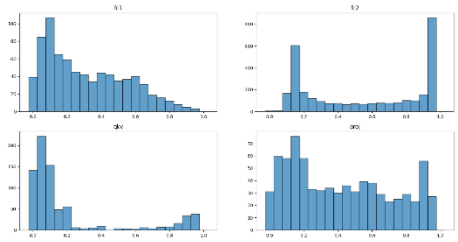




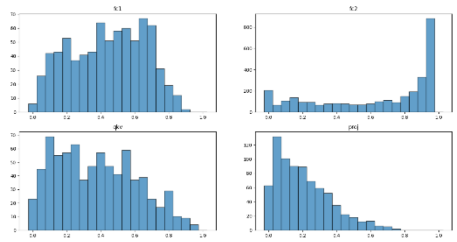


### 7.6. Quantization Percentage Error

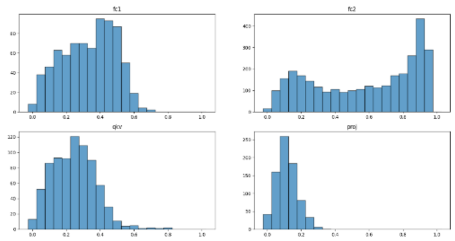
The following figures show the quantization percentage error of the four fused MatMul operators for each attention block.



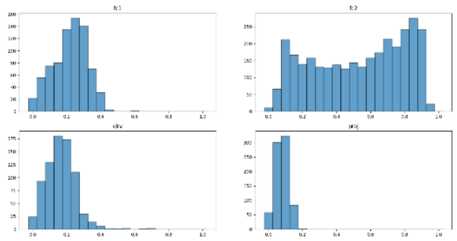
Block.0



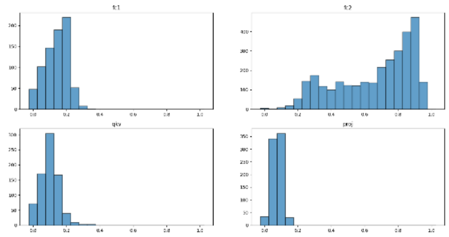
Block.1



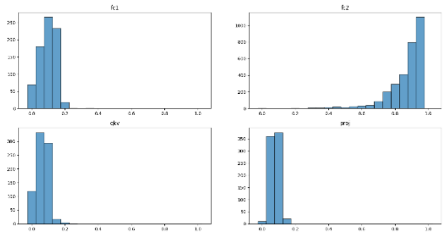
Block.2



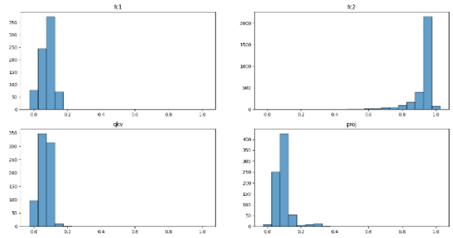
Block.3



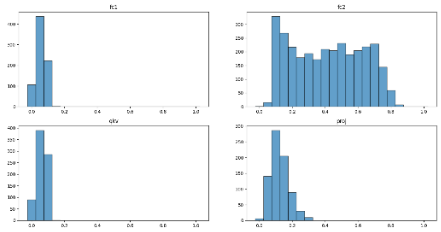
Block.4



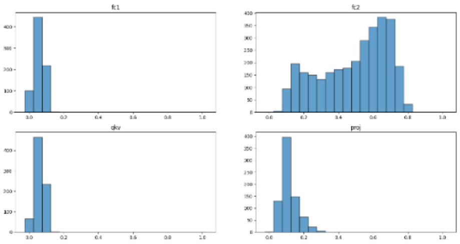
Block.5



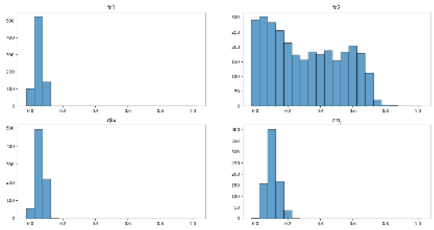
Block.6



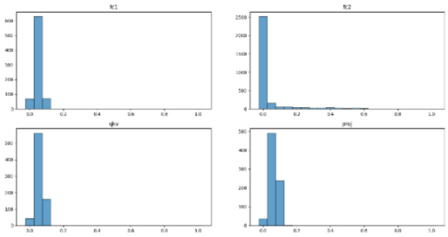
Block.7



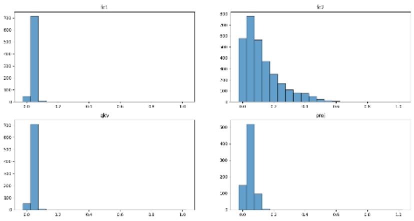
Block.8



Block.9



Block.10



Block.11

### 7.7. Attention Map Visualization

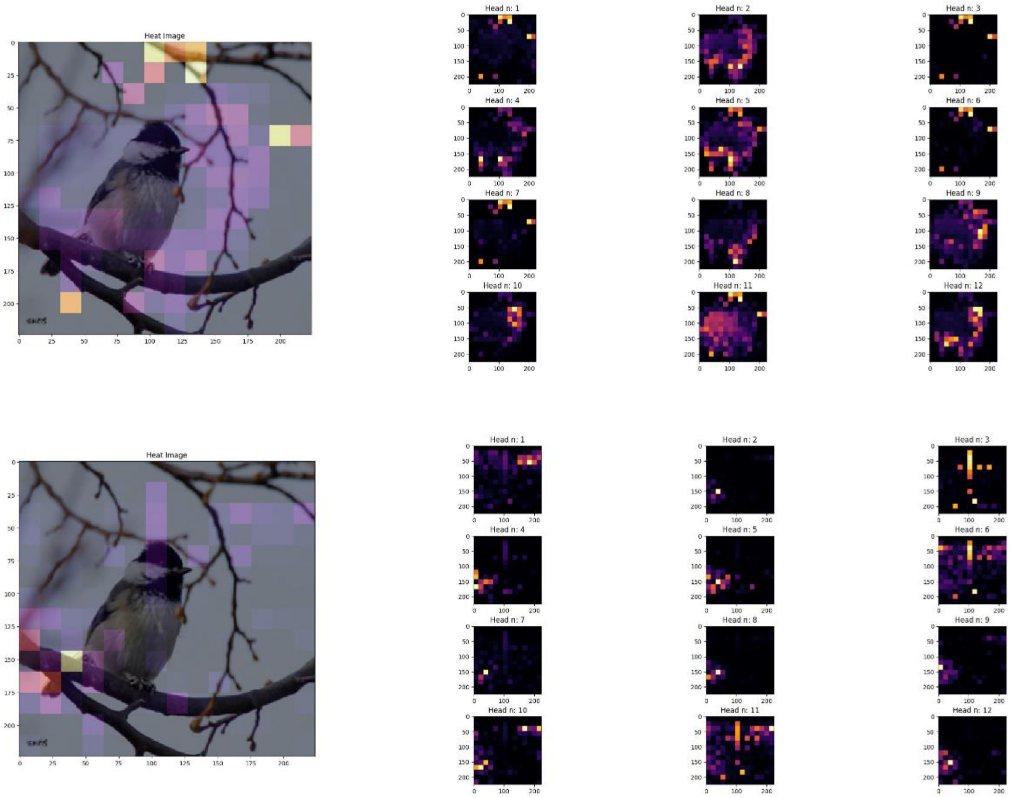
Remainder:

a. Full-precision model

b. Quantized model, only LayerNorm is per-channel quantized, other operators are per-tensor quantized.

c. Quantized model, LayerNorm and Block5/FC2, Block5/FC2 are per-channel quantized, other operators are per-tensor quantized.

Figure 1



a

b

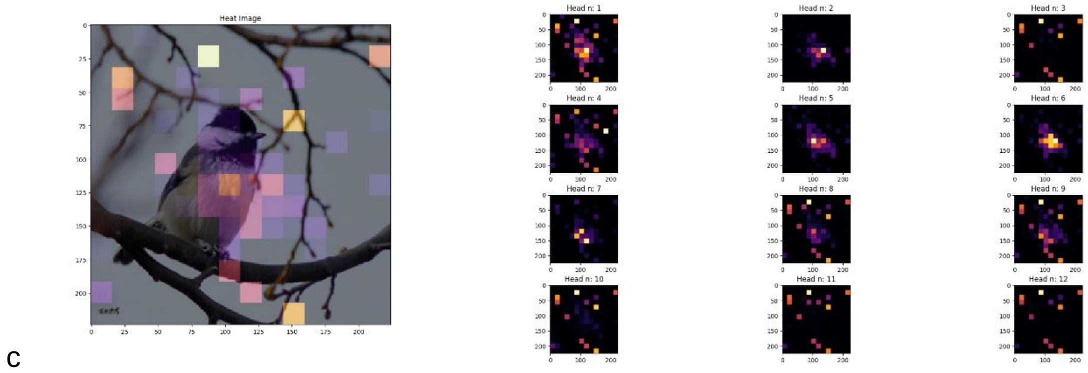

Heat Image
Head n: 1
Head n: 2
Head n: 3
Head n: 4
Head n: 5
Head n: 6
Head n: 7
Head n: 8
Head n: 9
Head n: 10
Head n: 11
Head n: 12
c

Figure 2

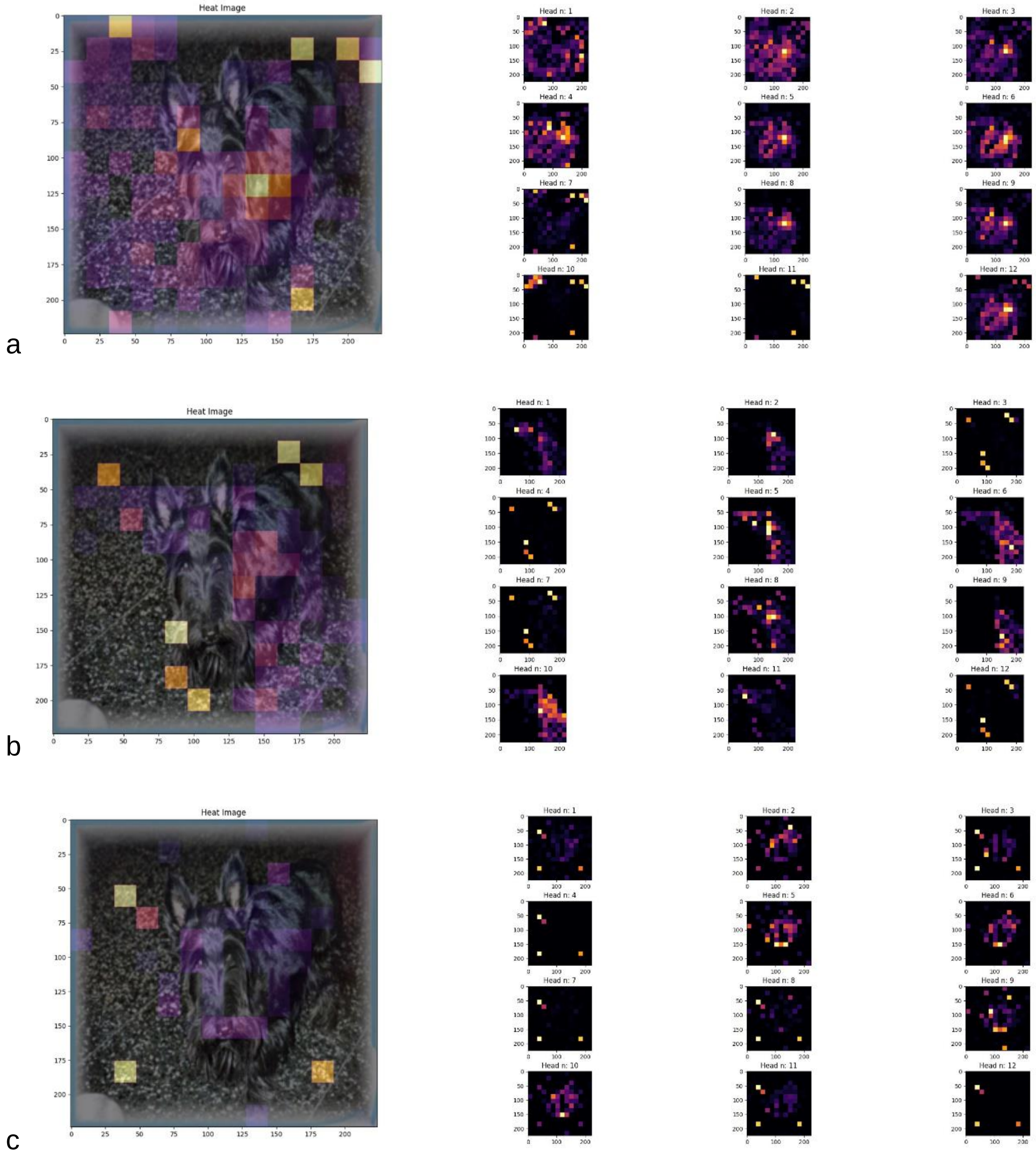

Heat Image
Head n: 1
Head n: 2
Head n: 3
Head n: 4
Head n: 5
Head n: 6
Head n: 7
Head n: 8
Head n: 9
Head n: 10
Head n: 11
Head n: 12
a
Heat Image
Head n: 1
Head n: 2
Head n: 3
Head n: 4
Head n: 5
Head n: 6
Head n: 7
Head n: 8
Head n: 9
Head n: 10
Head n: 11
Head n: 12
b
Heat Image
Head n: 1
Head n: 2
Head n: 3
Head n: 4
Head n: 5
Head n: 6
Head n: 7
Head n: 8
Head n: 9
Head n: 10
Head n: 11
Head n: 12
c

Figure 3

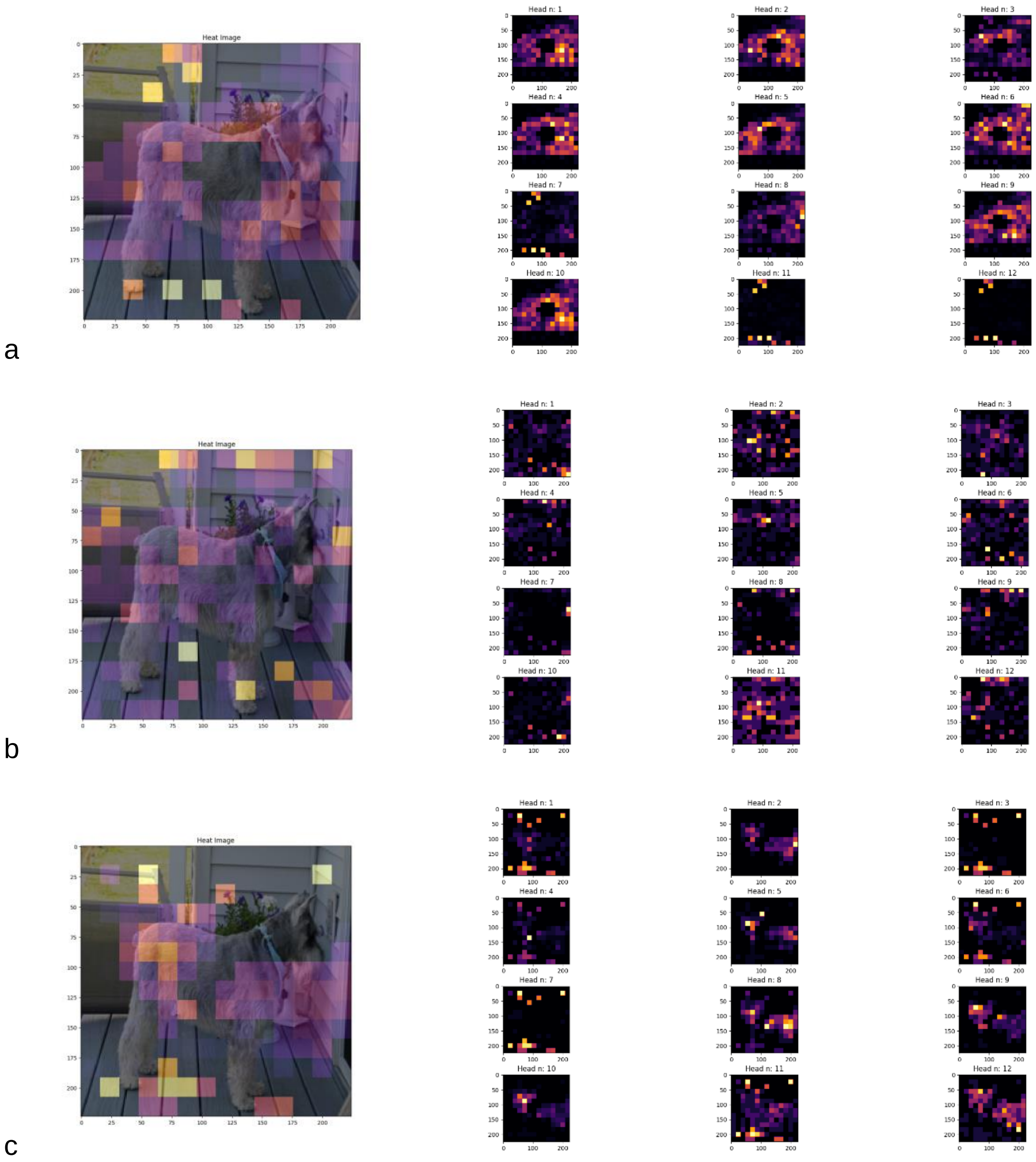
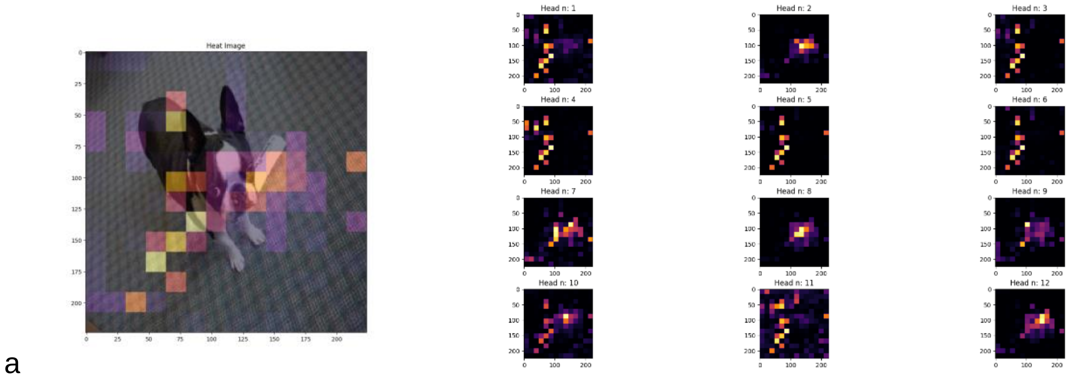
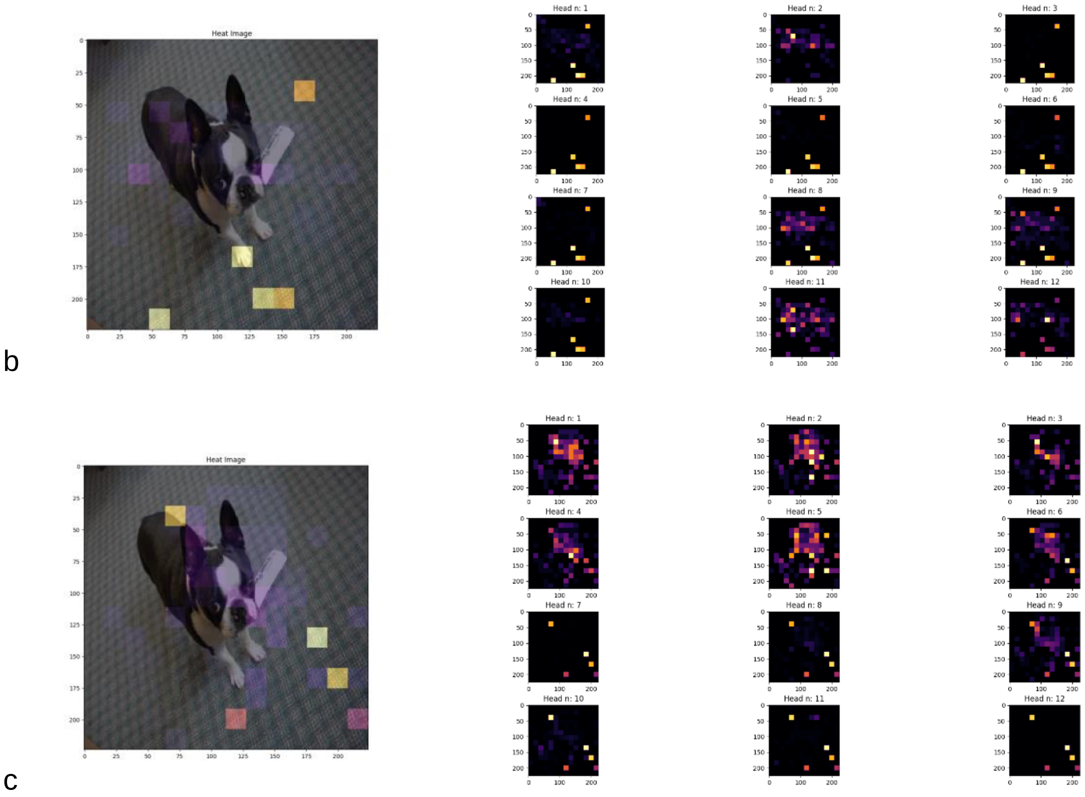


Figure 4




b
Heat Image
Head n: 1
Head n: 2
Head n: 3
Head n: 4
Head n: 5
Head n: 6
Head n: 7
Head n: 8
Head n: 9
Head n: 10
Head n: 11
Head n: 12
c
Heat Image
Head n: 1
Head n: 2
Head n: 3
Head n: 4
Head n: 5
Head n: 6
Head n: 7
Head n: 8
Head n: 9
Head n: 10
Head n: 11
Head n: 12