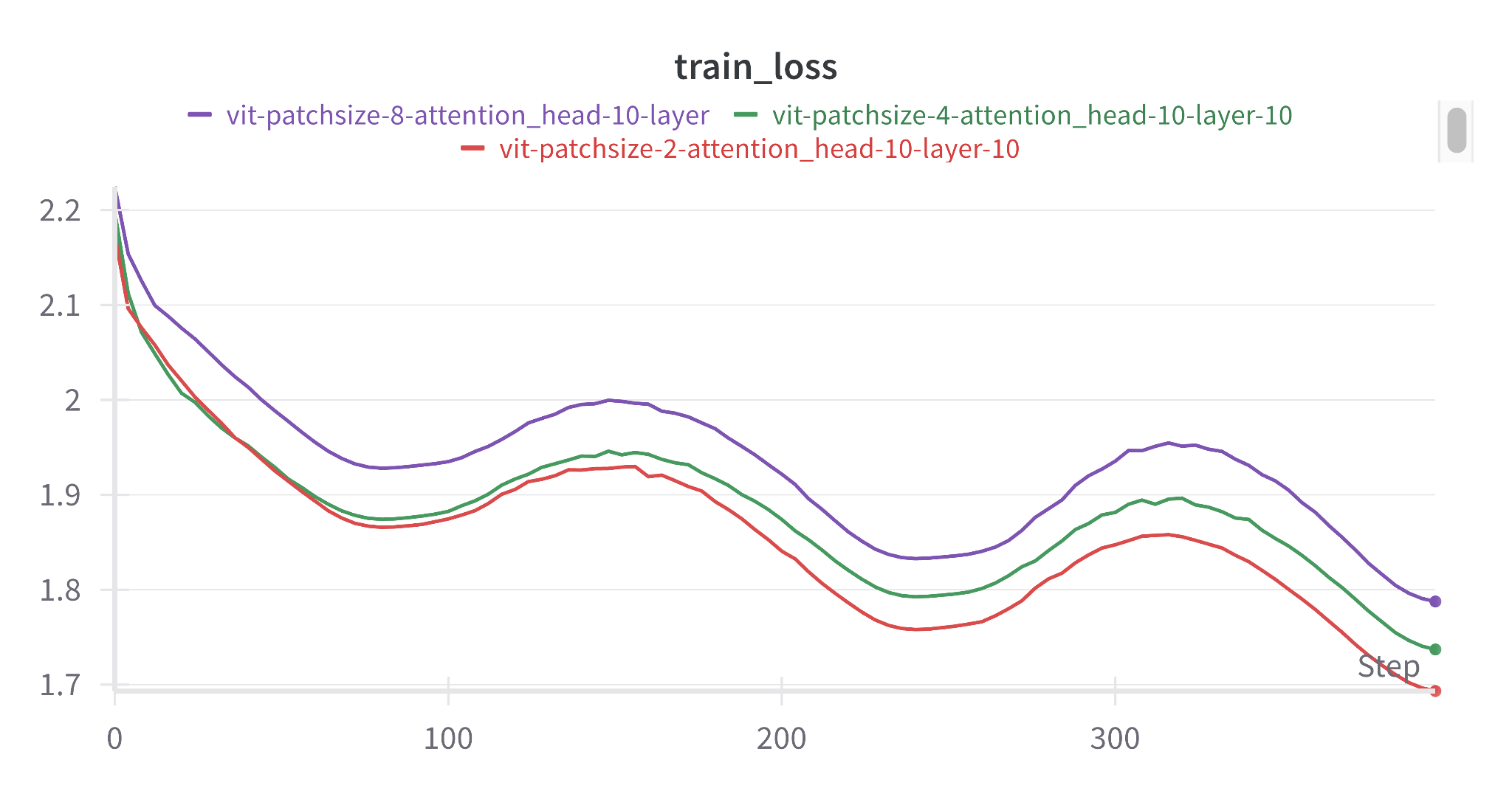
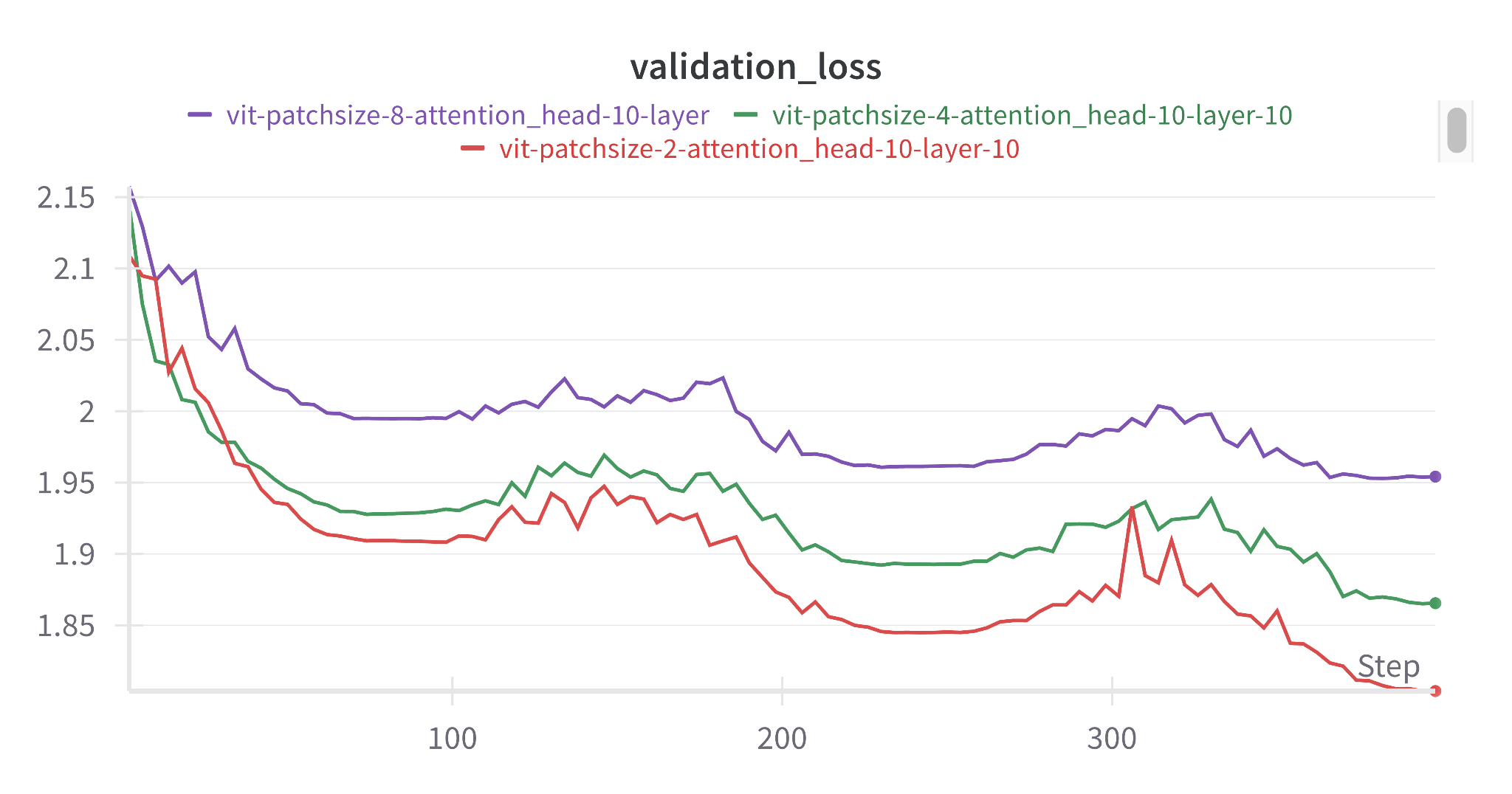
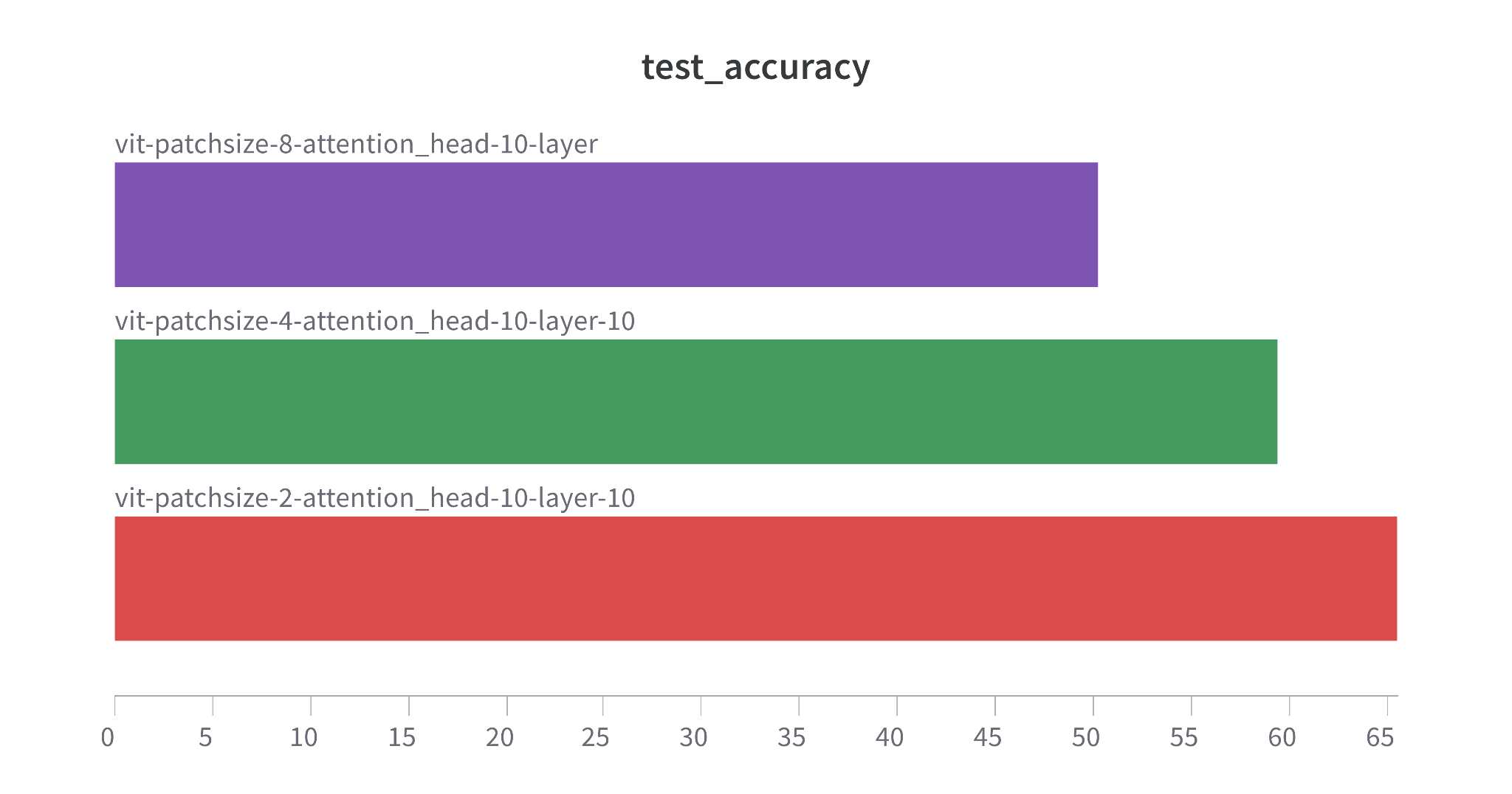
### **1.2.1 Patch Size Variation:**

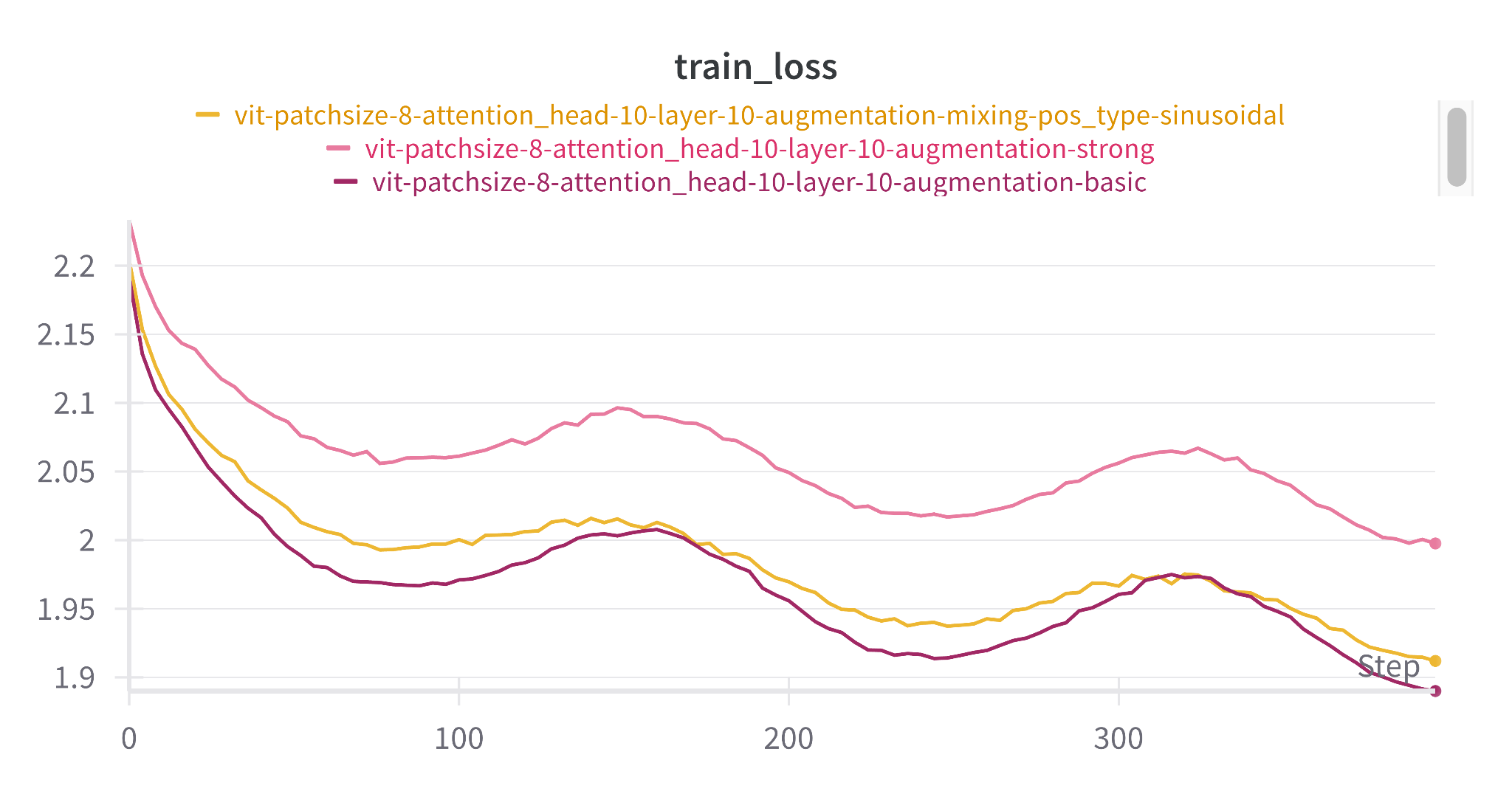
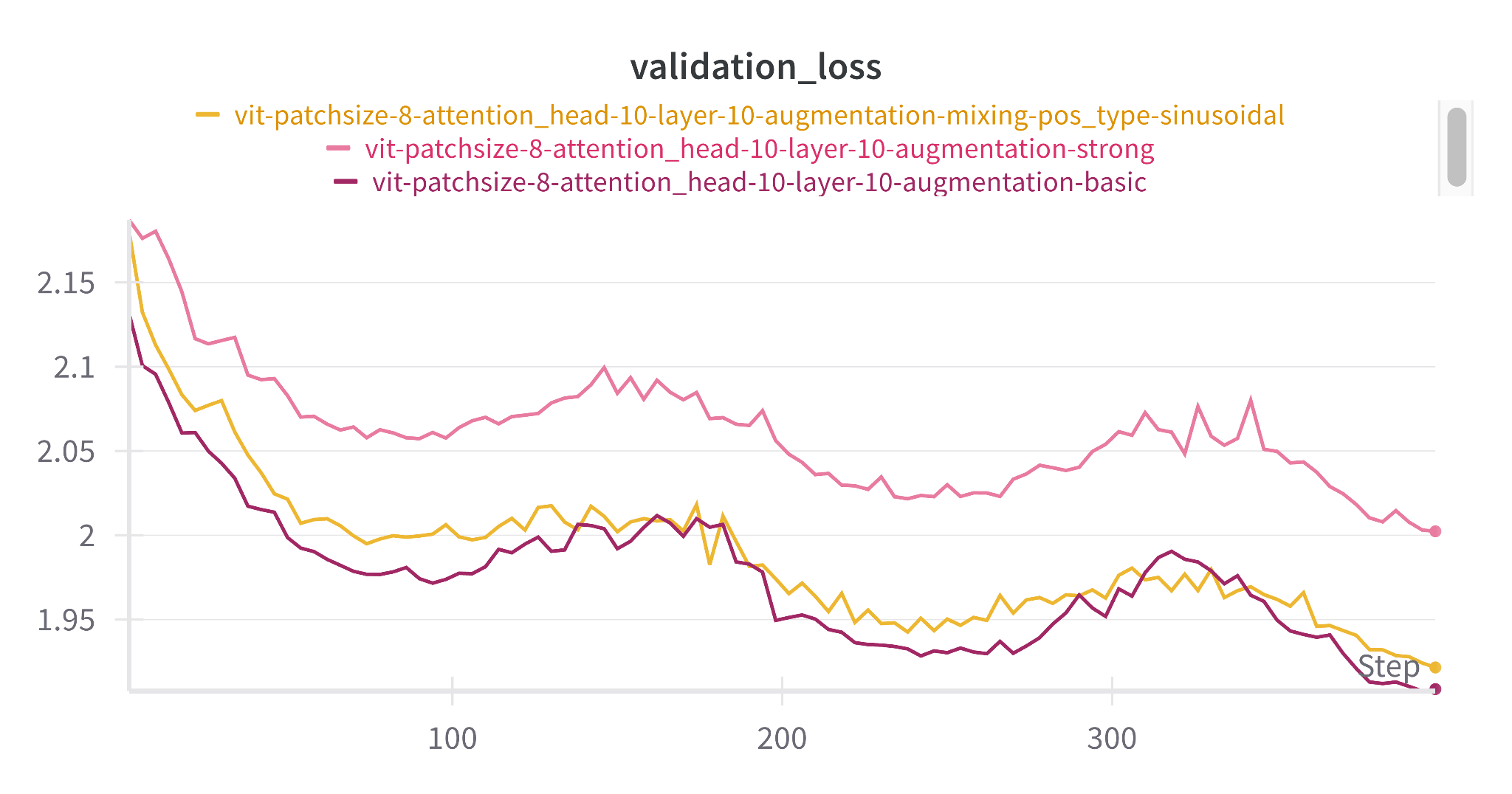
* Training Loss:
* Validation Loss:
* Testing Accuracy:
  + With patchsize of 2, the model is performing best with 65% accuracy because it retains maximum spatial information as compared to 4 and 8.

### **1.2.2 Hyper-parameter exploration:**

* Training with :
  + N\_heads = 20,
  + N\_layers = 20,
  + Patch size=2,
  + Learning rate= 0.005
  + Optimizer Adaam

Gives accuracy of >80% within 50 epochs. Making n\_heads and n\_layers higher makes it heavy and compute time very high.

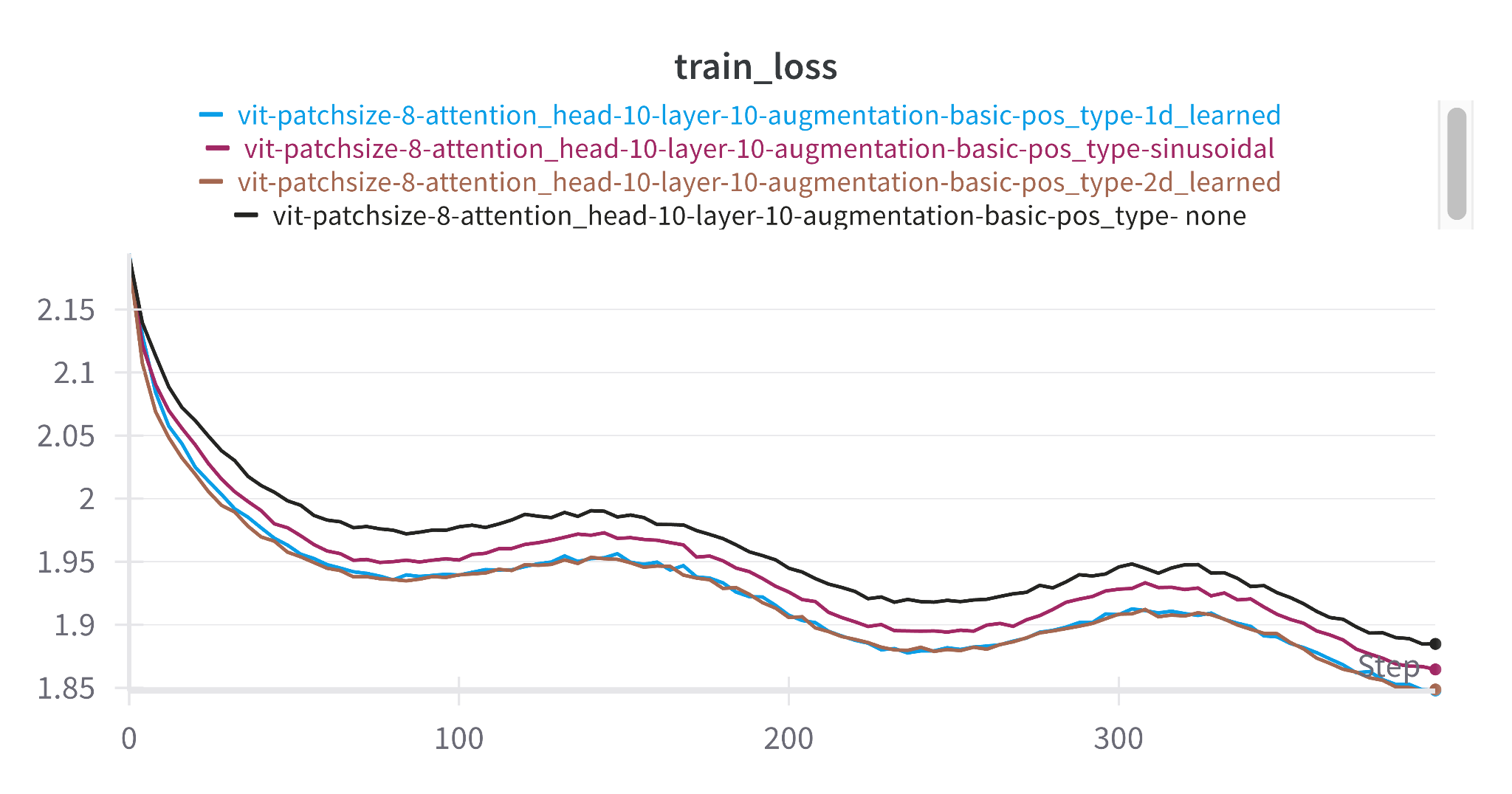
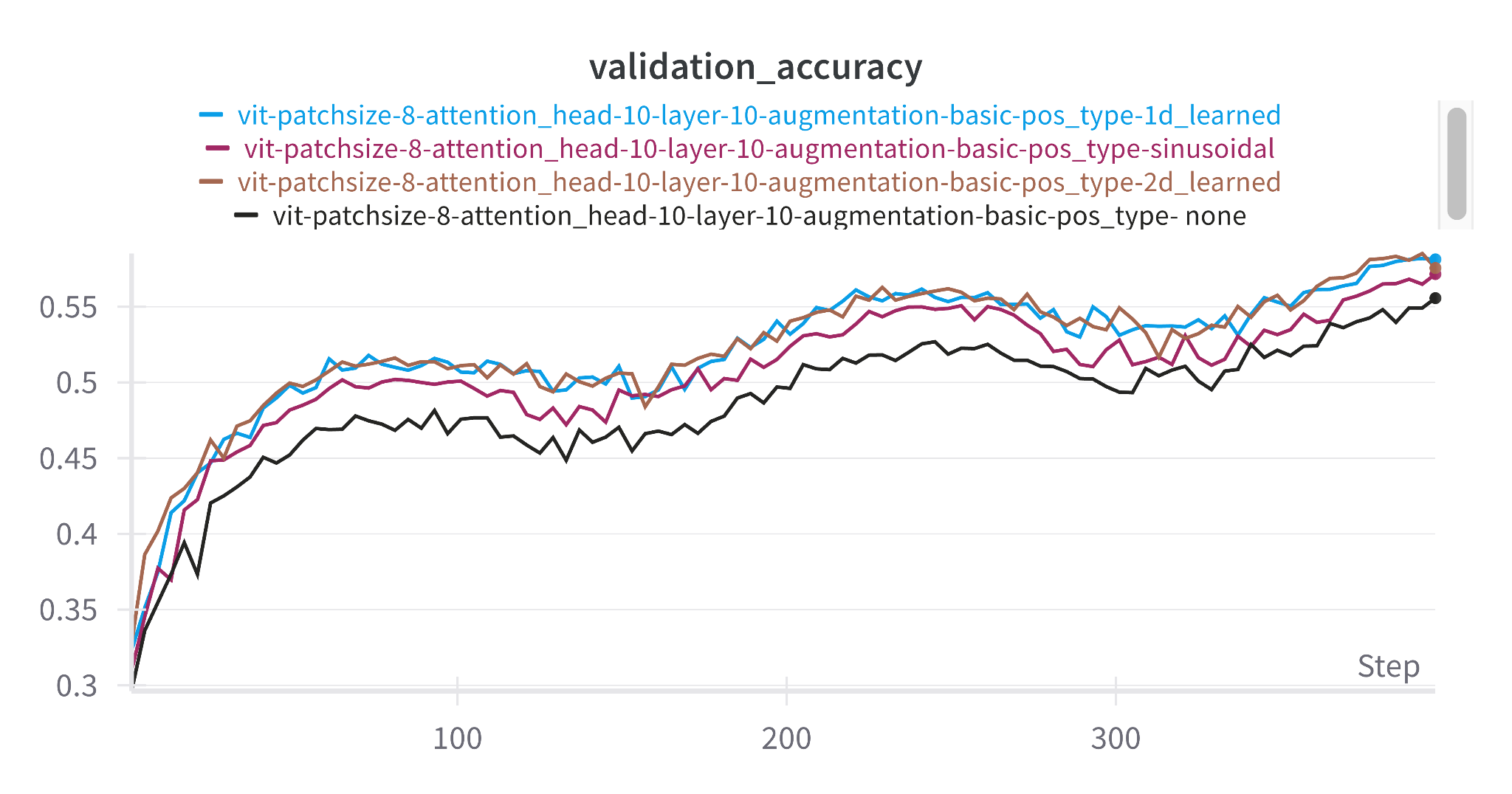
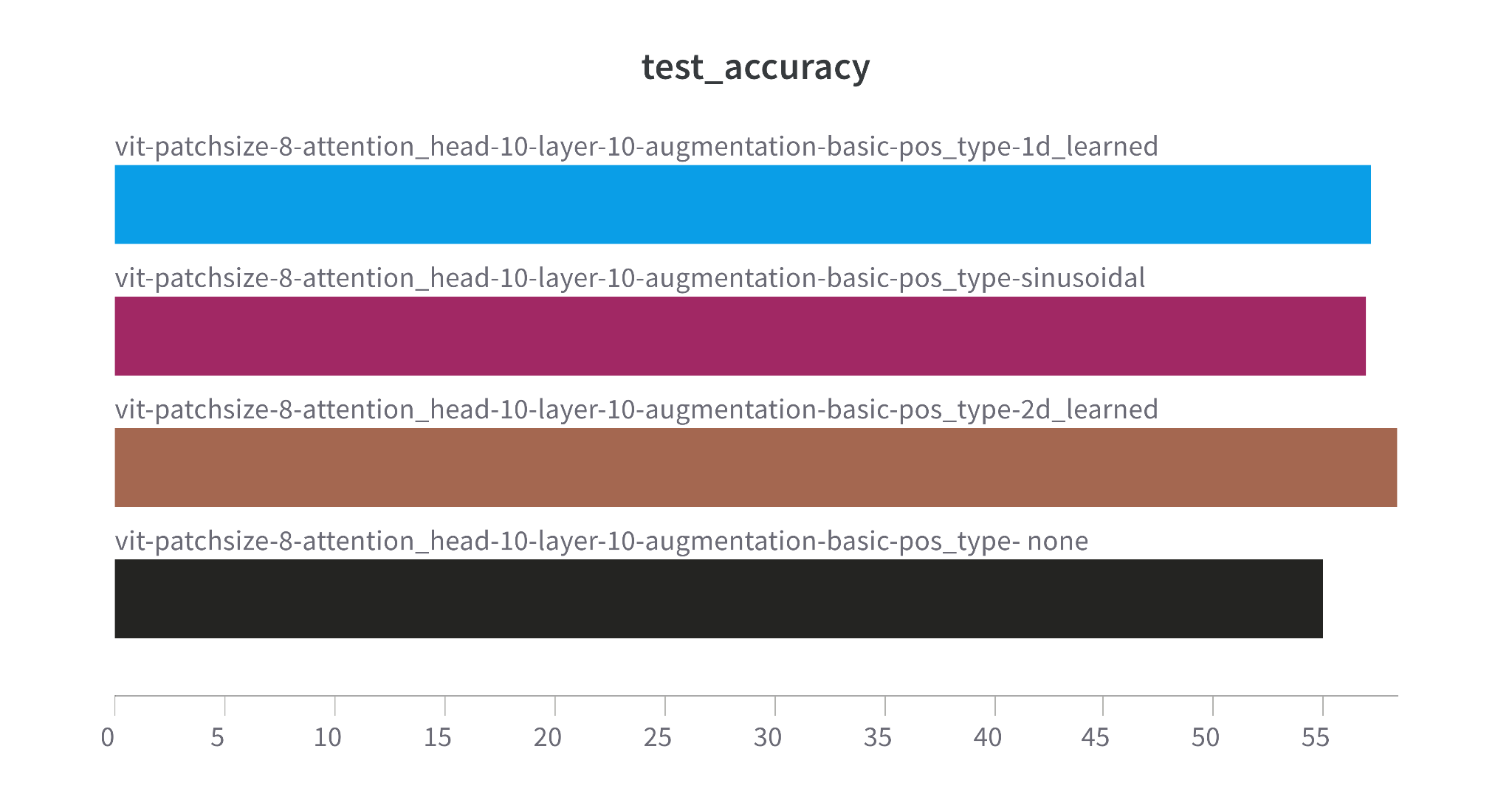
### **1.2.3 Data Augmentation techniques:**

* Training Loss:
* Validation Loss:
* Testing Accuracy:
  + if set\_name == "basic":
  + transform = transforms.Compose([
  + transforms.RandomHorizontalFlip(),
  + transforms.RandomCrop(32, padding=4),
  + transforms.ToTensor(),
  + transforms.Normalize(mean=mean, std=std)
  + ])
  + elif set\_name == "strong":
  + transform = transforms.Compose([
  + transforms.RandomHorizontalFlip(),
  + transforms.RandomCrop(32, padding=4),
  + transforms.RandAugment(),
  + transforms.ToTensor(),
  + transforms.RandomErasing(p=0.25),
  + transforms.Normalize(mean=mean, std=std)
  + ])
  + elif set\_name == "mixing":
  + # Same as strong; Mixup/CutMix applied separately in training loop
  + transform = transforms.Compose([
  + transforms.RandomHorizontalFlip(),
  + transforms.RandomCrop(32, padding=4),
  + transforms.ColorJitter(brightness=0.4, contrast=0.4, saturation=0.4, hue=0.1), # Color jittering
  + transforms.GaussianBlur(kernel\_size=3, sigma=(0.1, 2.0)), # Gaussian blur
  + transforms.ToTensor(),
  + transforms.Normalize(mean=mean, std=std)
  + ])

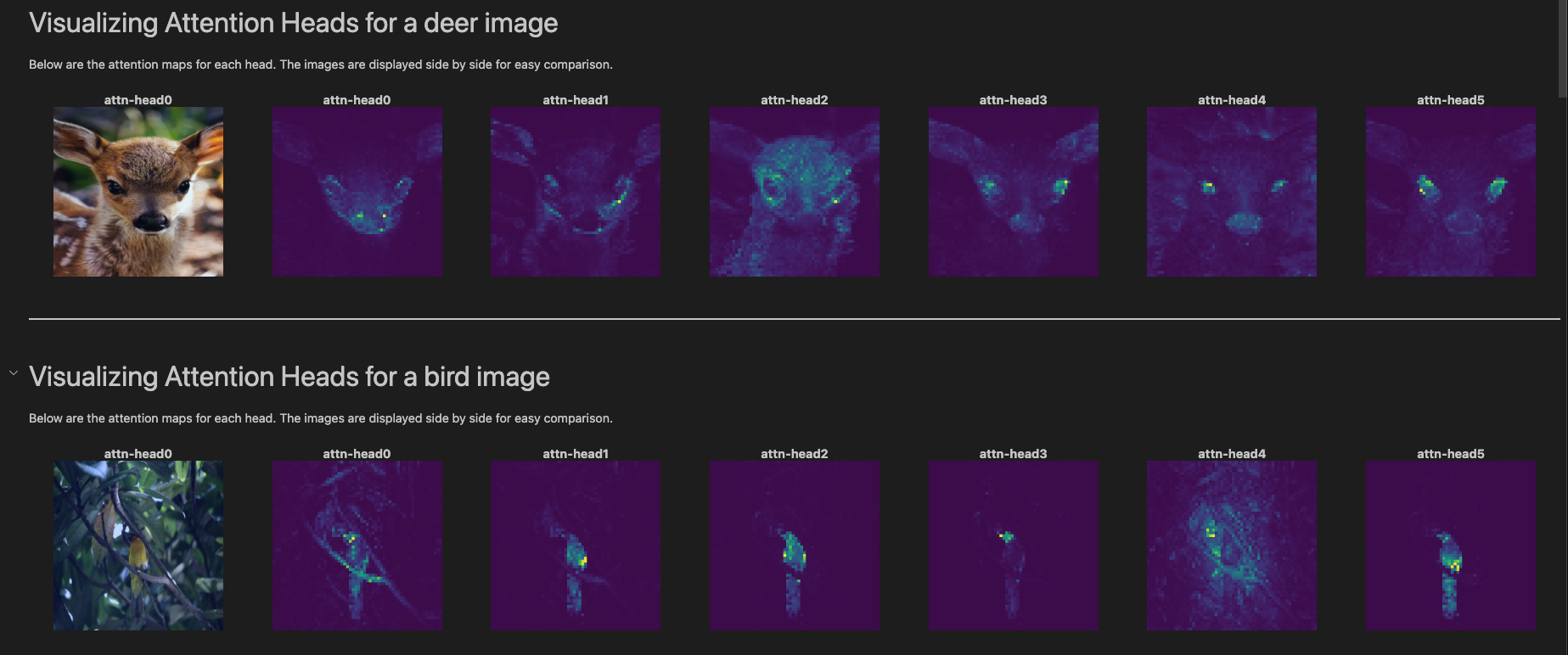
| **Strategy** | Basic | Strong | Mix |
| --- | --- | --- | --- |
| **Accuracy** | 54.97 | 45 | 53 |

* + If using Mix of augmentation techniques nakes the model robust and making higher accuracy.

### **1.3 Positional Embeddings:**

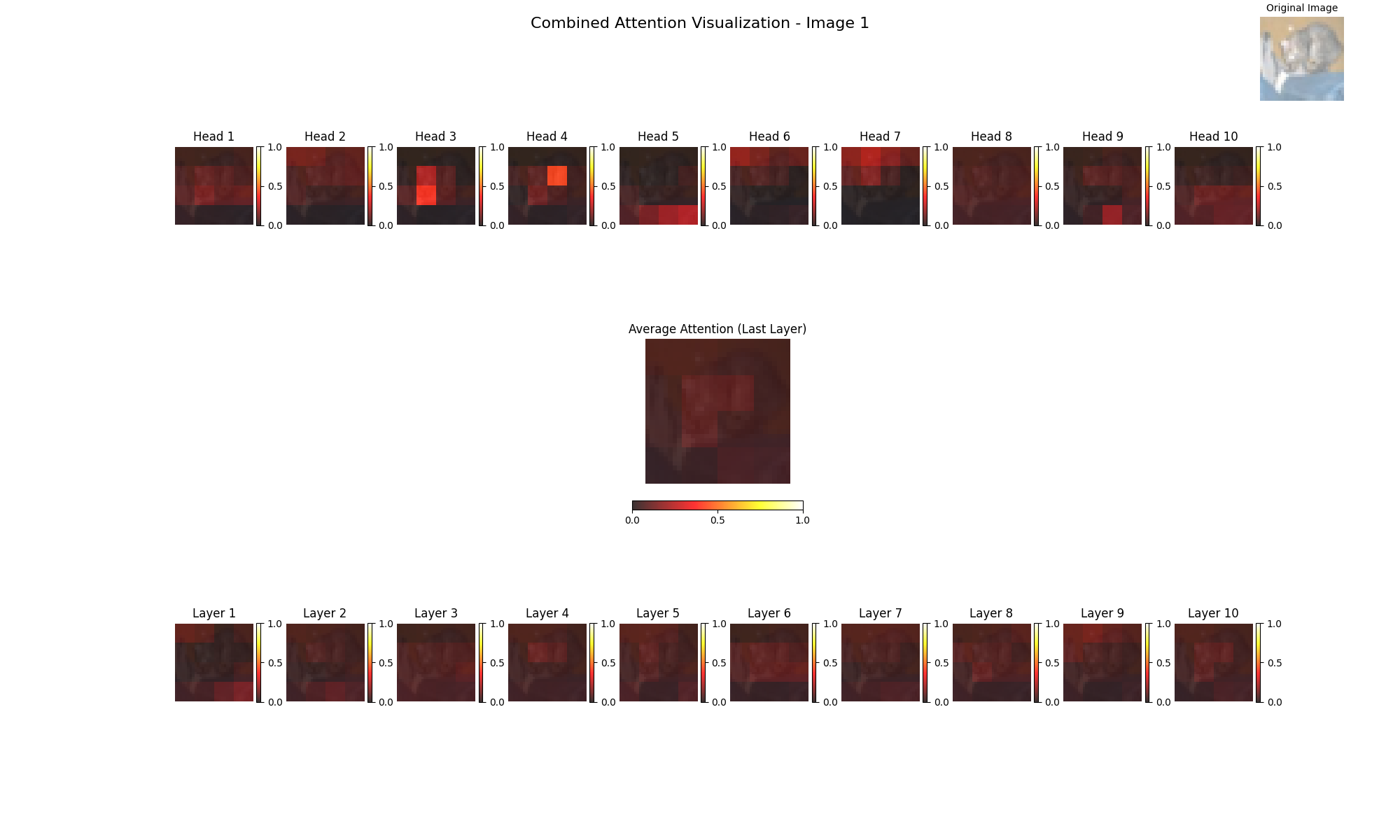
* Training Loss:
* Validation Accuracy:
  + 2D positional embedding giving the best accuracy among all of them and ```None``` gives the worst of it.
  + 2D positional embeddings encode both horizontal (x) and vertical (y) positions independently, allowing the model to capture true two-dimensional spatial relationships between elements. This is crucial for tasks where the arrangement of data in two dimensions carries semantic or structural meaning, such as in images or tables.
  + Standard sinusoidal embeddings are inherently 1D and only capture sequence order, not spatial structure. When 2D data is flattened into a 1D sequence, vital spatial context is lost, limiting the model’s ability to reason about relationships between elements that are neighbors in 2D but distant in 1D
* Test Accuracy:

### **1.4.1 Dino Attention Map:**



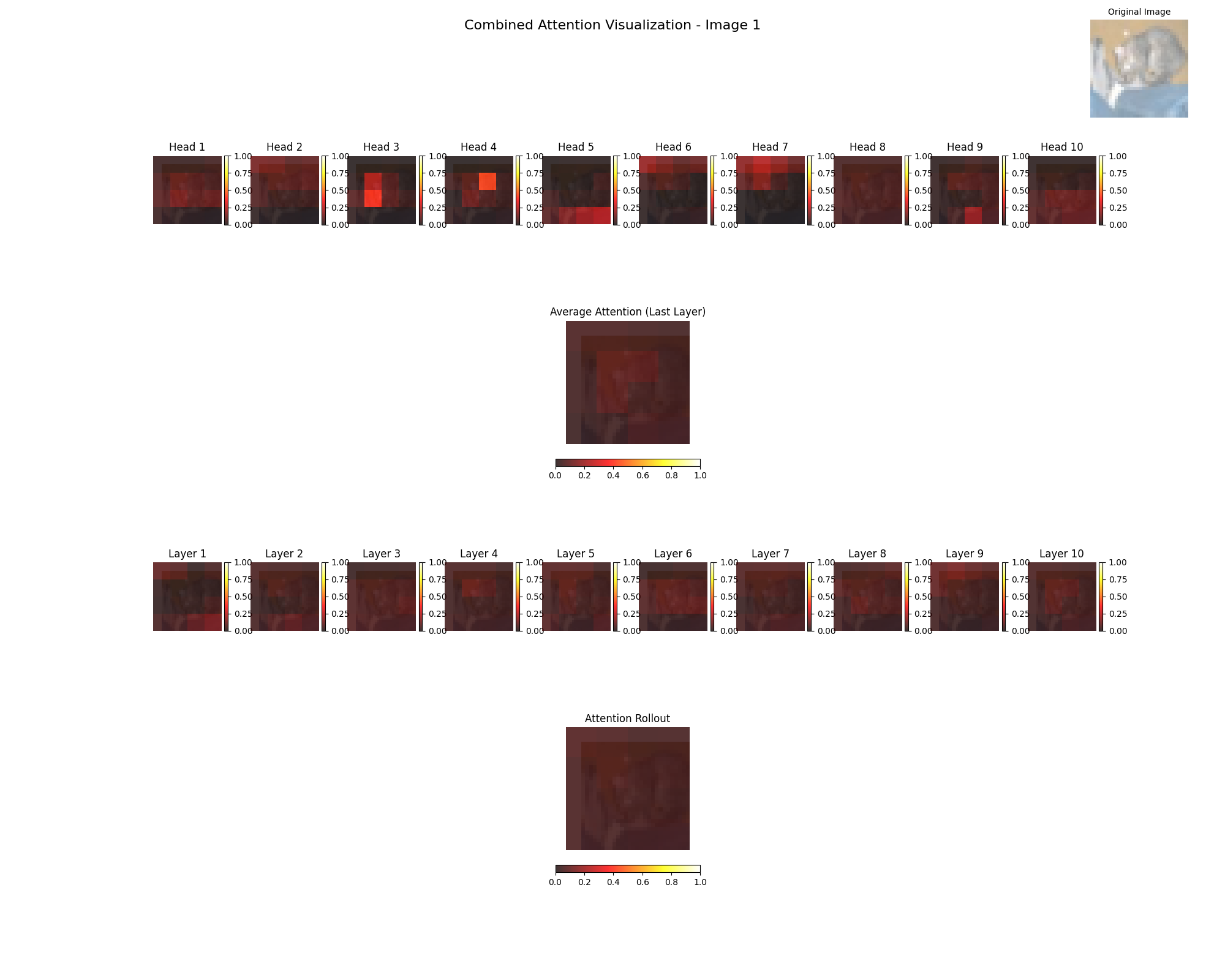
### **1.4.2 ViT CIFAR-10 Attention Map**

* Setup: vit-patchsize-8-attention\_head-10-layer-10-augmentation-basic-pos\_type-sinusoidal

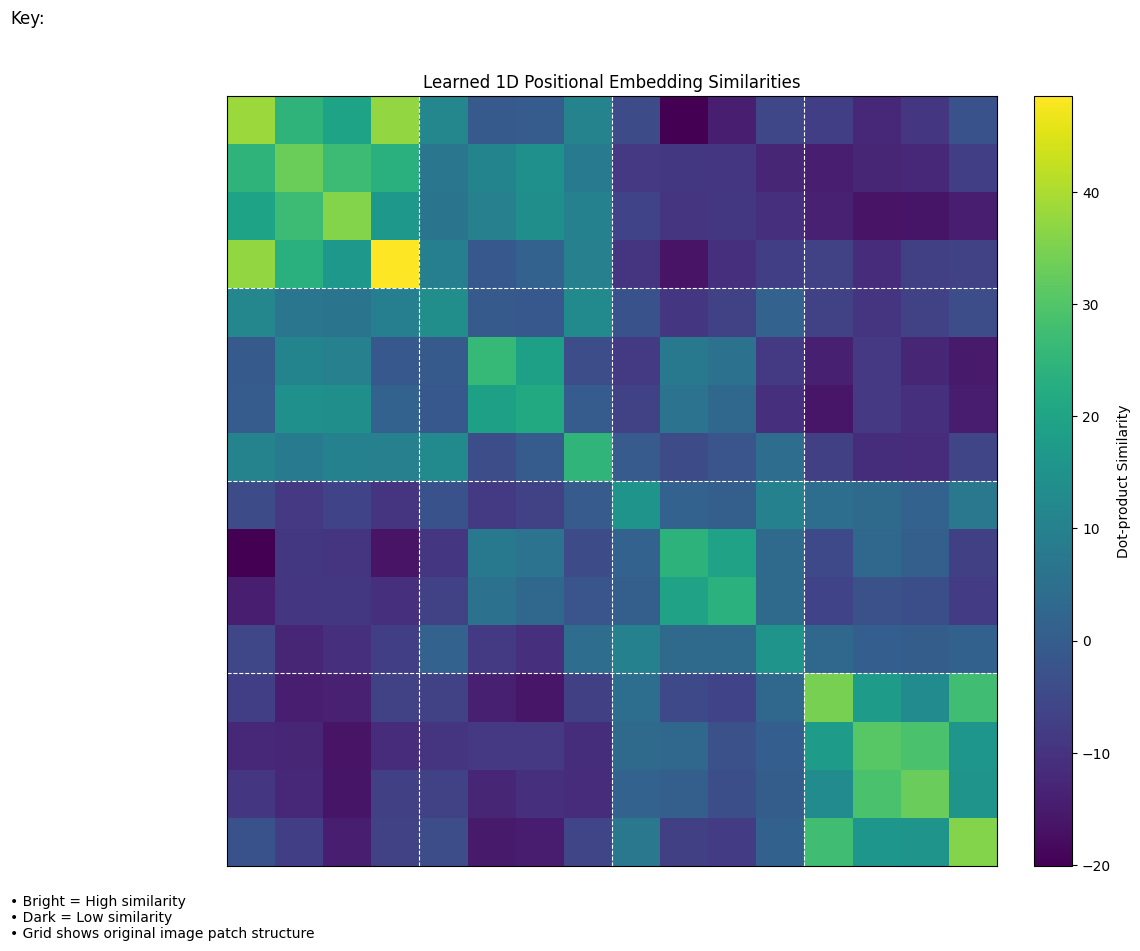


* As Visible from Average Attention map(from last layer), the model is able to give correct attention to the Object of interest at centre.
* At last layer, Head#3 and Head#4, is most important.
* Among 10 layers, layer-{6,8,10} is most effective.

### **1.4.3 Attention Rollout:**



### **1.4.4 Positional Embedding Visualization:**



* As patch size is 8X8 and image is of 32X32, grid size of 4X4.
* The ViT learns a **positional bias** where embeddings for spatially adjacent patches are more aligned.

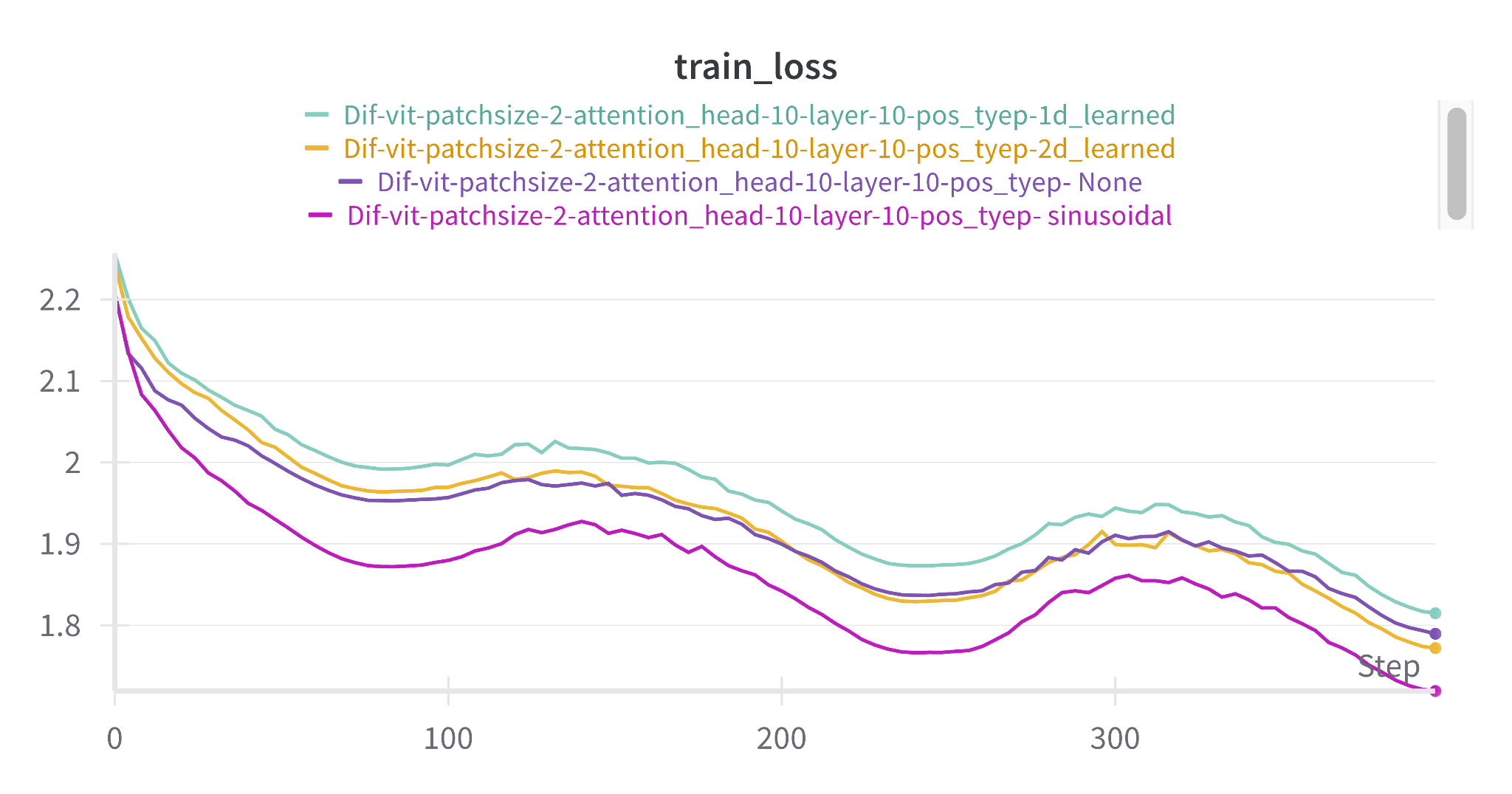
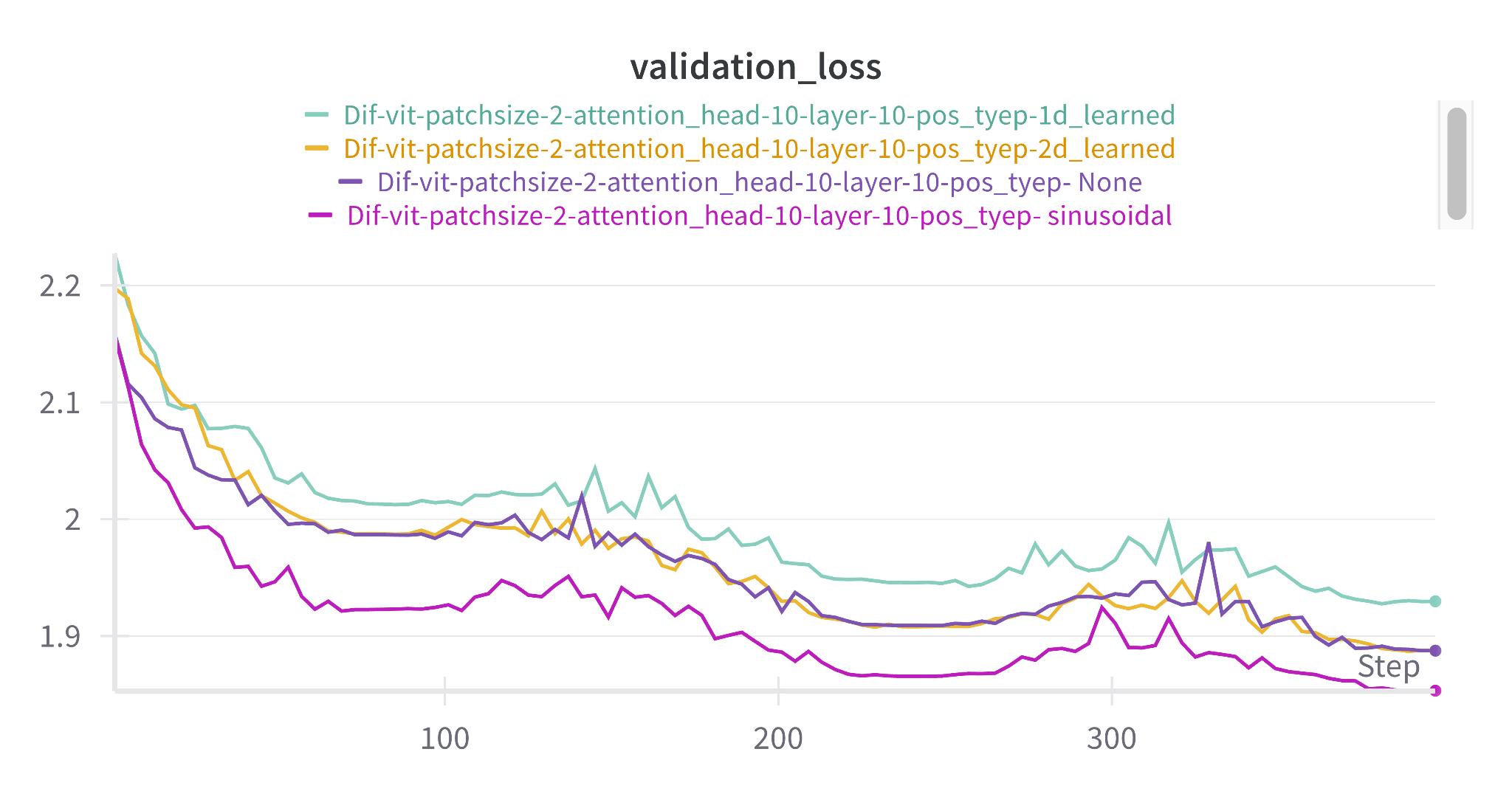
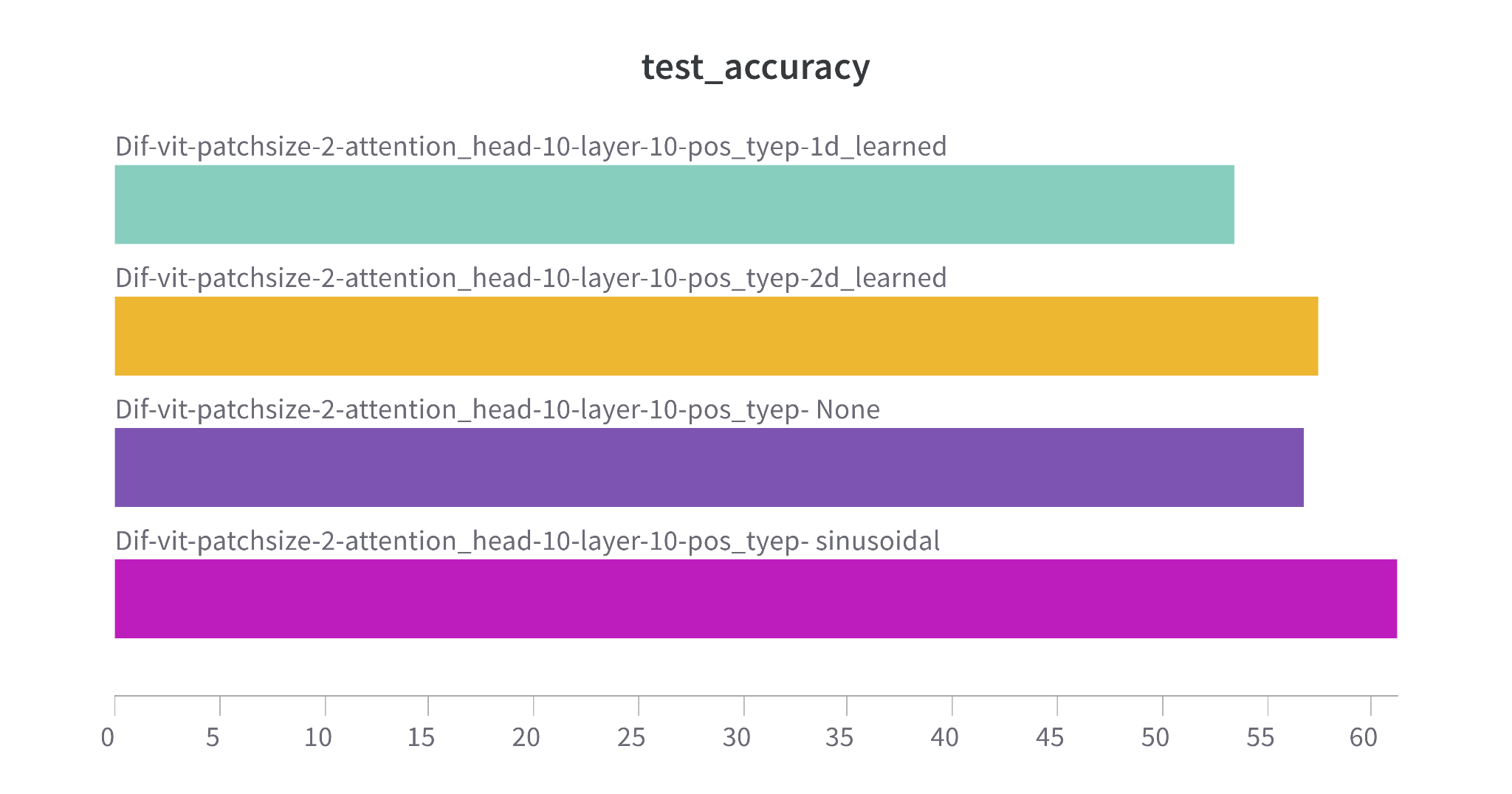
### 2. **Differentional Vision transformer:**

Implemented Differential Vision Transformer with Patch Size 2.

#### **2.2**

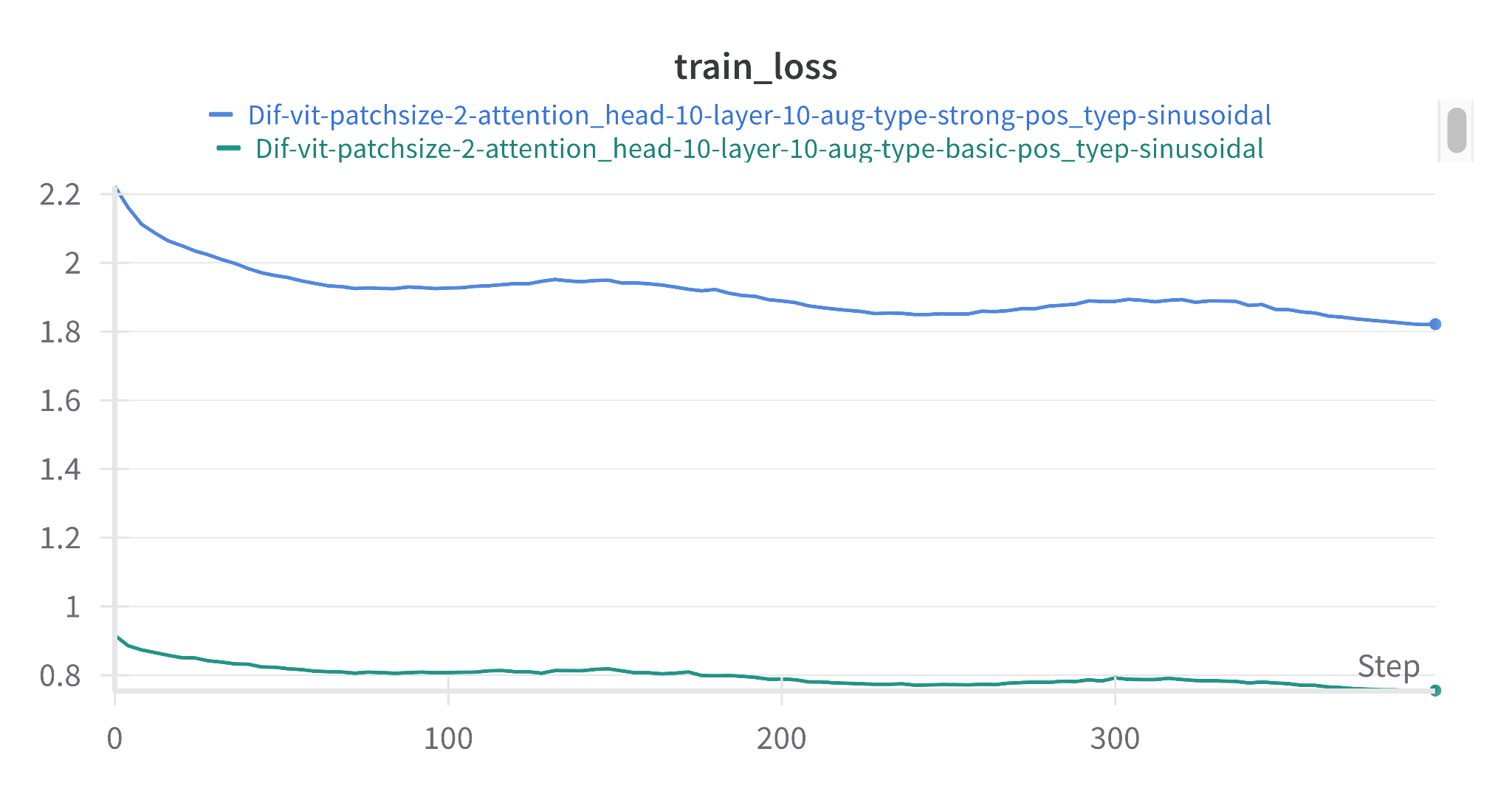
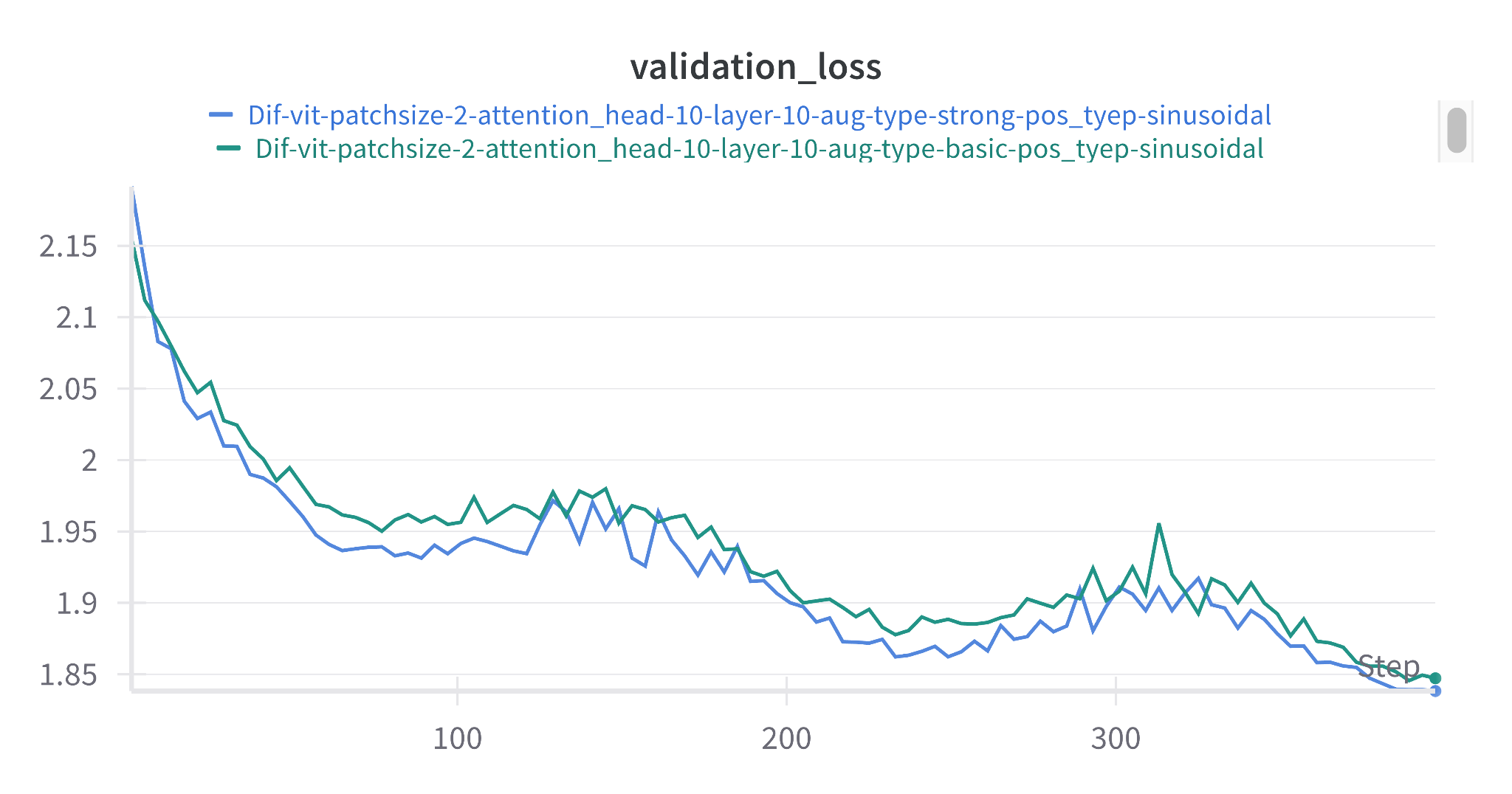
#### **Positional Embedding Type:**

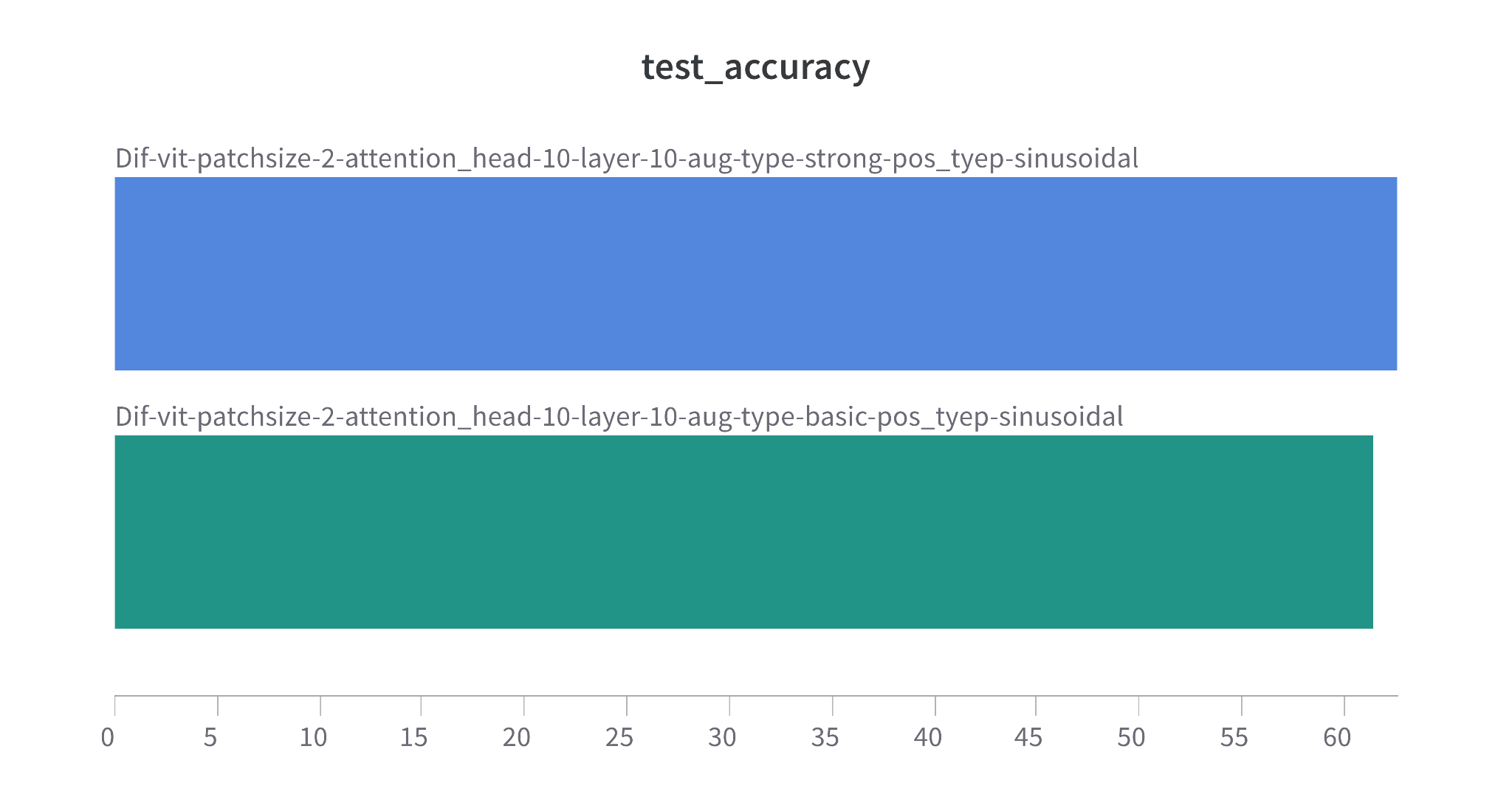
—--------------------------------------

* Training Loss:
* Validation Loss:
* Test Accuracy:
* Sinusoidal position embedding makes the best model out of it and having 66% accuracy.

### **Augmentation:**

—-------------------

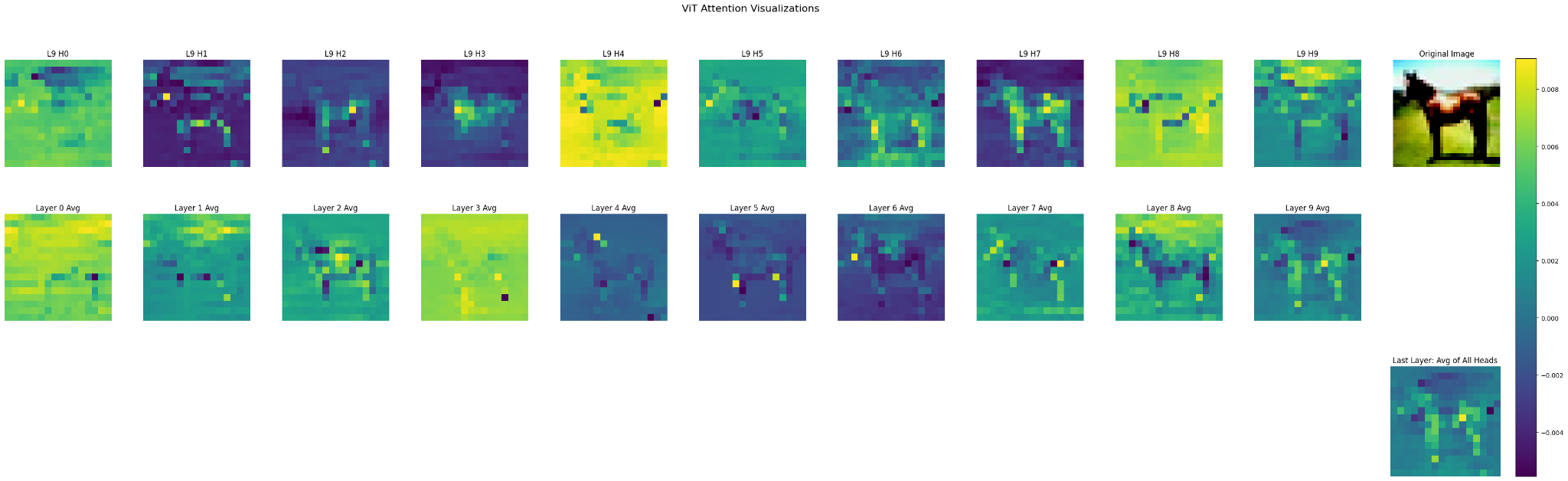
Training Loss:  
Validation Loss:

Testing accuracy:

* Strong augmentation type is working better

### **2.3 Visualization:**

* Setup: Dif-vit-patchsize-2-attention\_head-10-layer-10-aug-type-basic-pos\_tyep-sinusoidal
* [CLS] token attention map



* First row: 10 heads of last layer → H2, H3 and H7 is most prominent.
* Among all layers, Layer-2’s average is most prominent and Layer-8.
* Average Attention layer is also preserving the semantic information.
* Attention Rollout token:

### **3. CLIP Model:**

**\*\*Q1) Do the visual encoders have the same architectures\*\***

- No, CLIP does not use a single visual encoder architecture. Instead, it explores the use of both modified ResNet architectures (CNNs) and Vision Transformer architectures. These architectures differ fundamentally in their approach to processing visual information. ResNets rely on convolutional layers to extract features hierarchically, while ViTs treat an image as a sequence of patches and apply Transformer layers with self-attention mechanisms. The specific modifications made to the ResNet architecture within CLIP, such as the attention pooling, further distinguish it from a standard ResNet. The use of both types of architectures allows the CLIP model to leverage different strengths in visual representation learning.

- It is important to note that the text encoder used in CLIP is a Transformer architecture, distinct from both the ResNet and Vision Transformer-based visual encoders.The text encoder takes text as input, which is first converted into a lower-cased byte pair encoding (BPE) representation, and outputs a textual feature representation. These modality-specific feature representations are then linearly projected into a shared multi-modal embedding space where their similarity is calculated.

**### Q2) ILSVRC: dataset setup**

- ImageNet's label hierarchy is based on the WordNet hierarchy

- Each concept, mostly described by bunch of words, is called "synony set" is called "synset"

**### Q3) Could grouping objects based on synsets lead to problems for visual recognition?**

- Grouping objects based on synsets can lead to problems of visual recognition.

- **\*\*Polysemy:\*\*** Without word context, models like CLIP would struggle to determine correct visual concepts. For example, ImageNet contains synsets for both construction cranes and birds that fly, both referred to as "cranes"

- **\*\*Varying Granularity:\*\*** Levels of granularity used used in ImageNet may not be optimal for recognition task.

- **\*\*Hierarchical Overlap:\*\*** For certain tasks, like the image classification task in the ImageNet Large Scale Visual Recognition Challenge (ILSVRC), the 1000 selected synsets are chosen such that there is no hierarchical overlap between them (no synset is an ancestor of another within this subset). This suggests that directly using the full WordNet hierarchy, where broader synsets contain more specific ones, could lead to complications or ambiguities in classification tasks if not handled carefully.

**### Q4) Visual differences in same synset**

- **\*\*Variation in visual characterstics:\*\*** Objects within the same synset may exhibit differences in their appearancs due to factors such as style, material and colour or pose. e.g. visually similar synset like seals and seal otters mar come closer due to sysnset postulate.

- **\*\*Differences in image context and background:\*\*** Image captured in same synset mat be captured in different environmenta dna context.

- **\*\*Changes in scale, viewpoint and articular:\*\*** Objects in the same synset may depict different scales, viewpoints and sate of articulation, etc.

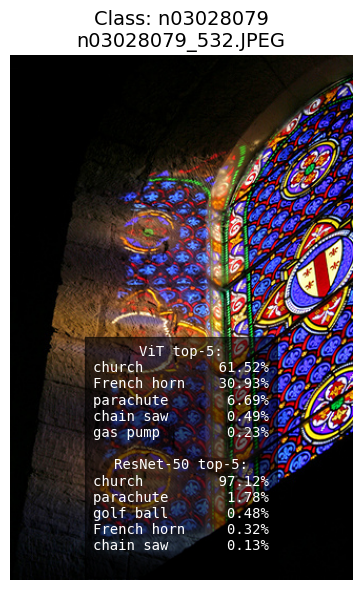
### **Examples of Images working better at CLIP but not on Resnet:**

It will take much space in report, so put it in Notebook itself.

* - resnet is trained on ~1.4 Million dataset while CLIP trained on 400+ million, making it more orbust to zero-shot as compared to resnet.
* - "gas pump" class is top 1% class of image-net trained resnet50 having 70-80% accuracy on that class.
* - CLIP works on supervisory signal of text embedding as compared to only image feature in resnet making CLIP more confident on its prediction.
* CLIP’s transformer-based architecture can capture long-range dependencies and global context better than ResNet50’s convolutional layers, which focus on local features. In the above image the gas pump somehow resembles with background, making it harder for CNN based RN50 to seperate out background and foreground.
* Example:

### **Examples of Images working better at Resnet50 but not on CLIP:**

As a generalised model CLIP Struggles to distinguish similar looking images. Like below image, the patterns in the image may look like a French horn but this is a church. AS a supervised model RN50 has seen similar examples before and that’s why giving high confidence on single class



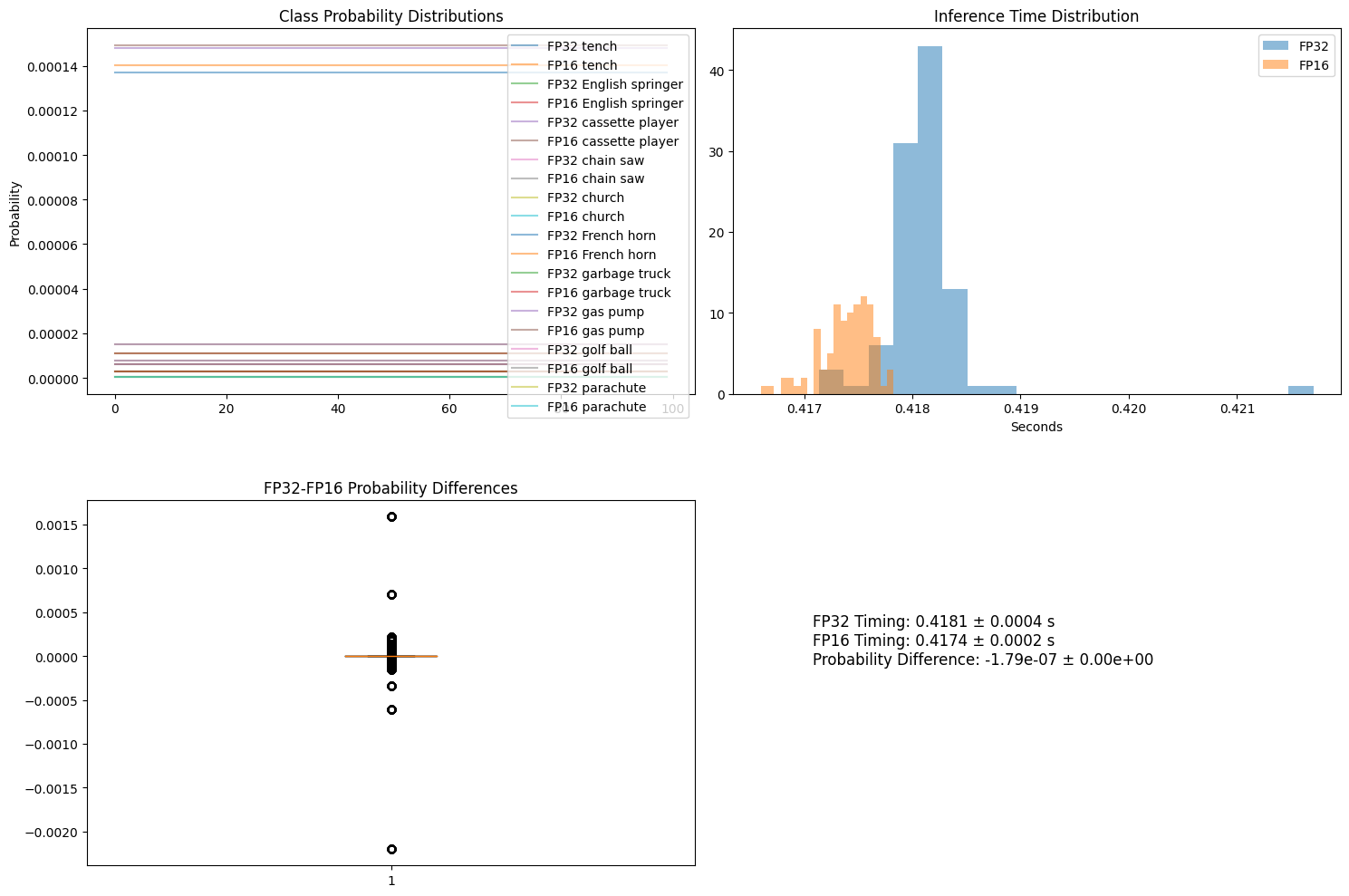
### 

### **FP16 Inference:**

* When inference on random 5 images per class: there is no significant difference in the confidecence for FP32 and FP16.
  + Although precision is being pruned but, Deep learning models in general is robust enough to tolerate this. Specially, CLIP models are designed to be robust and generalize well. The inherent robustness might make them less sensitive to the numerical differences introduced by FP16, especially if the pruning process was effective.

\*

#### **Image Encoding time:**

* 
* FP16 inference time little less than that of FP32.
* On my system, LayerNorm is not being converted to FP16, that requires FP32 inherently makes it difficult to effective FP16 conversion.
* Can be made through ONNX conversion, but due to limitation of scope could not able to do that on this assignment.