Computer Vision Exercise

Done by:

Name: Tarek Abou Chakra

Matriculation number: 5450275

Submitted for:

Deep Learning Lab

Exercise 2

1. Introduction:

This report explores three exercises in computer vision that address various aspects of image understanding and analysis. The first exercise focuses on semantic segmentation using transformers, employing their attention mechanisms and global context understanding to assign class labels to image pixels. With three output heads (linear, convolutional, and transformer), different strategies are examined for extracting semantic information. Additionally, the report delves into image captioning, aiming to bridge visual content and natural language by generating descriptive and coherent captions using Transformer-based architectures. Furthermore, the bidirectional relationship between images and textual descriptions is investigated. Hence, the report explores image-to-text and text-to-image generation. Throughout the report, results and performance metrics are presented, highlighting the strengths and limitations of the approaches used.

2. Segmentation:

In this section, I will present four distinct segmentation-type heads along with the corresponding results achieved by each. All of these heads were trained for 15 epochs using the same learning rate, batch size, and weight decay value, ensuring a fair comparison and evaluation of their performance.

1. Linear Head:

   The role of this particular head is to take the output from the transformer encoder layer and represent the data using a linear layer.

2. Convolutional Head:

   To achieve the desired segmentation results, a convolutional layer was employed in this specific head. The convolutional layer used has a number of input features equal to the embedding size and a number of output features equal to the desired number of classes for segmentation. In this case, the convolutional layer utilized a "same" convolution setup with a kernel size 3 and padding of 1, ensuring that the output image size remains unchanged.

3. Transformer head:

   Following the outputs from the first transformer, we incorporate class embedding's and proceed to compute the similarity between image patches and class embedding's using another transformer head. This computation can be performed using either a shared qk head or separate qk heads.

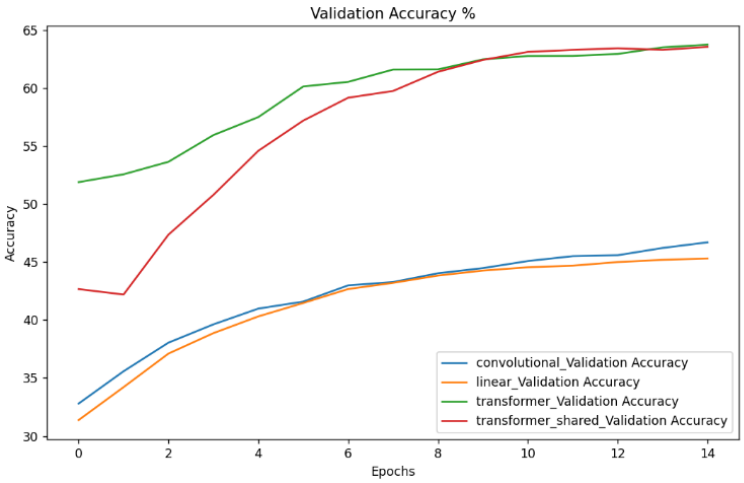
   a. Seperate qk:

b. Shared qk:

3. Results segmentation:

The following graphs and table present the results:

Model	Validation Accuracy
Linear	45.26%
Convolutional	47.3%
Transformer	63.65%
Transformer Shared	63.49%



In comparing different models, the convolutional layer outperformed the linear layer due to its ability to capture local patterns and spatial dependencies. Although both were implemented in a convolutional manner, a larger kernel size captures more contextual information and spatial dependencies, enabling the layer to learn more complex patterns. However, the transformer model showed even better performance than the convolutional layer, benefiting from its global context understanding and attention mechanisms. Among the transformer heads, the unshared query and key (qk) initially performed better than the shared qk variant. This can be attributed to the unshared qk weights' ability to attend to different regions of the image independently for different queries. Although the shared qk variant started with lower performance, it gradually caught up over time and training, showcasing the adaptability and benefits of parameter sharing between queries and keys. Nevertheless, sharing qk can also lead to a regularization effect and hence counter overfitting.

4. Image captioning:

In this section, image captioning was explored through two different approaches: greedy algorithms and top-k sampling. The greedy algorithm directly utilizes the maximum logits to determine the next word in the caption generation process. On the other hand, the top-k sampling variant introduces probability considerations and is influenced by a temperature parameter. The objective was to investigate the impact of various hyperparameter settings, prompts, and temperature values on the performance of the image captioning models.

First, I present an example from the output of the greedy algorithm to provide an illustrative understanding of its performance. And then showcase the results in a table for all the models and different setups:

- Reference caption: a computer is sitting on a desk next to a monitor
- Generated caption: a computer and a monitor on a desk

Table showcasing results of various models and hyperparameters:

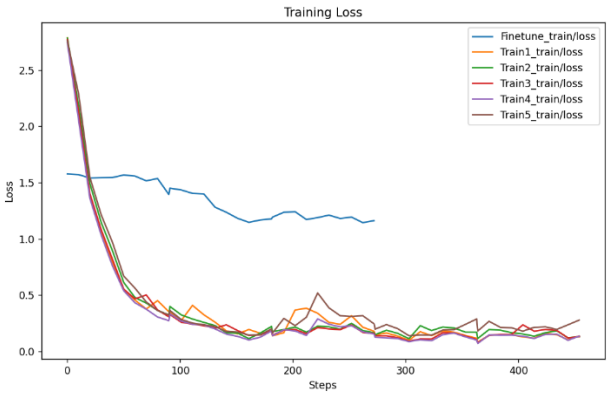
Model	Prompt	k	Temperature Value	Validation
Greedy1	“a picture of”	-	-	27.73%
Greedy2	“this is”	-	-	21.77%
Topk1	“a picture of”	50	1	4.36%
Topk2	“a picture of”	50	0.7	7.02%
Topk3	“this is”	100	0.7	5.73%
Topk4	“this is”	25	0.2	9.58%

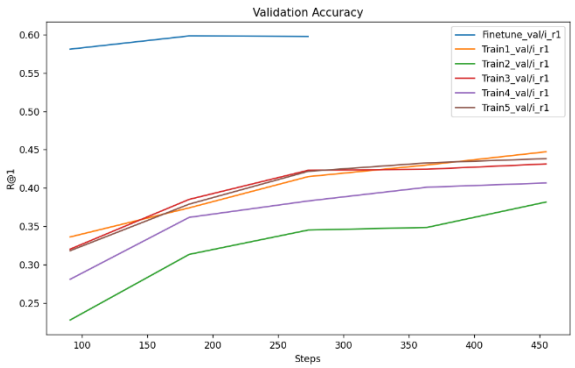
In image captioning, I explored different approaches to enhance the model's performance. While the greedy algorithm provided good results initially, I introduced randomness through top-k sampling and prompt variations. Lowering the temperature value (<1) reduced randomness, resulting in focused and deterministic captions (it makes tokens with higher probability more likely to be chosen by increasing their value). Decreasing the k value also decreased the diversity and creativity of the generated captions. The experiments showed that decreasing k and temperature values increased validation performance, making the model more deterministic and reducing random results. These findings highlight the importance of balancing randomness and determinism for better image captioning outcomes.

5. Image-Text Retrieval:

In this section, I performed a series of experiments to find the optimal solution for image text retrieval models. We trained the retrieval projection layers from scratch with various hyperparameter settings, examining their impact on performance. Additionally, I explored the option of fine-tuning the head instead of training from random initialization. The accompanying graphs and table present the findings, allowing us to determine the most effective strategy for our task.

Model	Learning Rate	Weight Decay	Epochs	R@1 Score
Train1	1e-3	1e-3	5	44.72%
Train2	5e-3	1e-4	5	38.16%
Train3	1e-5	1e-5	5	43.13%
Train4	2.5e-3	1e-4	5	40.65%
Train5	5e-4	1e-2	5	43.82%
Finetune	1e-5	0	3	59.77%





As seen from the graph and table results, it was easy to deduce that a better validation accuracy is observed when fine-tuning an already trained model. This can be attributed to the transfer learning effect, where the model benefits from pre-existing knowledge and adapts it to the new task. Whereas training from scratch requires the model to learn everything, which may lead to lower performance initially. Regarding hyperparameters, the learning rate played a significant role. A higher learning rate initially led to rapid improvement, but the decay rate was not sufficient for effective fine-tuning in later epochs. On the other hand, weight decay showed positive results, with higher values acting as a form of regularization to prevent overfitting. While improving results based on the given initial hyperparameters proved challenging, these findings shed light on the importance of choosing appropriate hyperparameters and the benefits of leveraging pre-existing models.