Notations and Definitions

The architectures used in the experiments of this thesis are based on the Transformer [1], and vision Transformer [2] architecture. Therefore, both image and text are represented as sequences of embeddings, which are processed by the Transformer blocks.

Image Representation

We define an image as a 3-dimensional tensor $\boldsymbol{v} \in \mathbb{R}^{C \times H \times W}$. Because we will use the base variant of the vision Transformer, ViT-B/16 [2], the image is patchified into 14x14 patches, each being a square of size 16x16 pixels. Each image patch represents one timestep in the sequence, and the number of patches N is given by $N = H \times \frac{W}{P^2}$, with P being the number of patches per dimension, and $P = 14$. Since we use an image size of 224x224 pixels, so $\boldsymbol{v} \in \mathbb{R}^{3 \times 244 \times 244}$, we will have $N = 244 \times \frac{244}{14^2} = 196$ patches, or timesteps respectively. Each patch is flattened into a 256-dimensional vector, and then projected into a 768 dimensions $\boldsymbol{e}_i^v \in \mathbb{R}^{768}$, using a fully connected layer. The image sequence is prepended with a special learnable [I_CLS] $\in \mathbb{R}^{768}$ token, which is used to aggregate the global information/content of the image, and following [2]. The result is a sequence of patch embeddings, which we define as $\boldsymbol{E}_v$, where v indicates an image:

$$\boldsymbol{E}_v = \left[\boldsymbol{e}_{[\texttt{I\_CLS}]}^v, \boldsymbol{e}_1^v, \boldsymbol{e}_2^v, ..., \boldsymbol{e}_N^v\right] \tag{1}$$

To give the Transformer a sense of order in the image patches/timestep, a unique positional encoding is added to each patch embedding. This can either be learned or fixed, with the latter being for example a sinusoidal positional encoding [1]. This positional encoding is also represented as a sequence of 768-dimensional vectors:

$$\boldsymbol{T}_v^{\text{pos}} = \left[0, \boldsymbol{t}_{\text{pos}_1}^v, \boldsymbol{t}_{\text{pos}_2}^v, ..., \boldsymbol{t}_{\text{pos}_N}^v\right] \tag{2}$$

Since the [I_CLS] token is not part of the image, the positional encoding for the [I_CLS] token is set to zero, so nothing is added to it. An image representation is defined as:

$$\boldsymbol{H}_{v,l}^s = \left[\boldsymbol{h}_{v,l,[\texttt{I\_CLS}]}^s, \boldsymbol{h}_{v,l,1}^s, ..., \boldsymbol{h}_{v,l,N}^s\right] \tag{3}$$

In Equation 3, l denotes the layer of the Transformer block that returned the image representation, and v indicates that the representation is an image. Since we use Knowledge Distillation (KD) in some parts of this thesis, representations will be, if neccessary, superscripted with s or t, for a student and teacher representation, respectively.

We define $l = 0$ as the input to the Transformer, and $l = L$ as the output of the Transformer, where L is the number of layers in the Transformer. Consequently, the image input to the Transformer is defined as:

$$\boldsymbol{H}_{v,0}^s = \left[\boldsymbol{h}_{v,0,[\texttt{I\_CLS}]}^s, \boldsymbol{h}_{v,0,1}^s, ..., \boldsymbol{h}_{v,0,N}^s\right] = \boldsymbol{E}_v + \boldsymbol{T}_v^{\text{pos}} \tag{4}$$

The output of the Transformer is defined as:

$$\boldsymbol{H}_{v,L}^s = \left[\boldsymbol{h}_{v,L,[\texttt{I\_CLS}]}^s, \boldsymbol{h}_{v,L,1}^s, ..., \boldsymbol{h}_{v,L,N}^s\right] \tag{5}$$

Text Representation

We define a text as a sequence of discrete tokens, which are, similiar to image patches, embedded into 768-dimensional vectors using an embedding matrix. A single token i is represented as $\boldsymbol{e}_i^t \in \mathbb{R}^{768}$, and the sequence of tokens, representing the text, is prepended with a start-of-sequence token [T_CLS] $\in \mathbb{R}^{768}$, and appended with an end-of-sequence token [T_SEP] $\in \mathbb{R}^{768}$. The purpose of the [T_CLS] token is, as with [I_CLS], to aggregate the global information/content of the text. The [T_SEP] token is used to indicate the end of the text sequence. A text sequence consists of M tokens, and we use w to denote a text sequence:

$$\boldsymbol{E}_w = \left[\boldsymbol{e}^w_{[\texttt{T\_CLS}]}, \boldsymbol{e}^w_1, \boldsymbol{e}^w_2, \ldots, \boldsymbol{e}^w_M, \boldsymbol{e}^w_{[\texttt{T\_SEP}]}\right] \tag{6}$$

The maximum text sequence length M is not fixed, and will be defined when neccessary in the experimental part of this work.

As mentioned (TODO: cite data preparation), to obtain discrete tokens, a sentence is tokenized into subwords using the GPT-2 byte-pair encoder, so one token does not necessarily represent a whole word.

A positional encoding is also added to the text embeddings, to give the Transformer a sense of order in the text sequence. Since the special token [T_SEP] denotes the end of the text sequence, it is part of the sequence, and therefore has a positional encoding. The latter does not hold for the [T_CLS] token, as it is used to aggregate the global information/content of the text.

$$\boldsymbol{T}^{\text{pos}}_w = \left[0, \boldsymbol{t}^w_{\text{pos}_1}, \boldsymbol{t}^w_{\text{pos}_2}, \ldots, \boldsymbol{t}^w_{\text{pos}_M}, \boldsymbol{t}^w_{\text{pos}_{[\texttt{T\_SEP}]}}\right] \tag{7}$$

A text representation is defined as:

$$\boldsymbol{H}^s_{w,l} = \left[\boldsymbol{h}^s_{w,l,[\texttt{T\_CLS}]}, \boldsymbol{h}^s_{w,l,1}, \ldots, \boldsymbol{h}^s_{w,l,M}, \boldsymbol{h}^s_{w,l,[\texttt{T\_SEP}]}\right] \tag{8}$$

Equation 8 denotes the representation denoted by a student model s, but it can also be a teacher representation t.

The input to the Transformer for text is Equation 8 with $l = 0$, and the output of the Transformer is Equation 8 with $l = L$.

Transformer Block

Unless we use pretrained architectures that follow a different architecture, which we will then specify, we follow the Pre-LayerNorm definition of the Transformer block as given in [3]. As the name suggests, it applies LayerNorm before the Multi-Head Attention, instead of after.

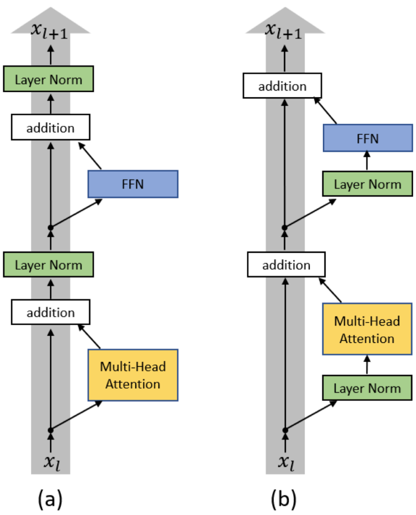

Figure 1: Comparison of a Post-Norm Transformer block/layer (a), and a Pre-Norm Transformer block/layer (b). (a) is the architecture as defined in the original "Attention is all you need" paper [1]. We follow the Pre-Norm architecture [3].

One Transformer block performs the following operations:

$$\boldsymbol{H}_l' = \mathrm{MHA}(\mathrm{LN}(\boldsymbol{H}_{l-1})) + \boldsymbol{H}_{l-1} \tag{9}$$

$$\boldsymbol{H}_l = \mathrm{FFN}(\mathrm{LN}(\boldsymbol{H}_l')) + \boldsymbol{H}_l' \tag{10}$$

We denote LN as LayerNorm, MHA as Multi-Head Attention, and FFN as a 2 layer MLP, all following the original Transformer of [1]. As previously mentioned, the only difference is the order of operations [3]. $\boldsymbol{H}_{v,l}^s$ and $\boldsymbol{H}_{w,l}^s$ can be used as a drop-in replacement for image and text, respectively. Both equations are, with slight adjustment, taken from VLMo [4].

We define a Transformer as multiple Transformer blocks stacked on top of each other.

Bibliography

[1] A. Vaswani *et al.*, "Attention is all you need," in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, in NIPS'17. Long Beach, California, USA: Curran Associates Inc., 2017, pp. 6000–6010.

[2] A. Dosovitskiy *et al.*, "An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale," *ICLR*, 2021.

[3] R. Xiong *et al.*, "On layer normalization in the transformer architecture," in *Proceedings of the 37th International Conference on Machine Learning*, in ICML'20. JMLR.org, 2020.

[4] H. Bao *et al.*, "VLMo: Unified Vision-Language Pre-Training with Mixture-of-Modality-Experts," in *Advances in Neural Information Processing Systems*, 2022. [Online]. Available: https://openreview.net/forum?id=bydKs84JEyw