International Journal of Engineering Applied Sciences and Technology, 2023
Vol. 8, Issue 02, ISSN No. 2455-2143, Pages 313-319
Published Online June 2023 in IJEAST (http://www.ijeast.com)

ANALYSIS OF TRANSFORMER-DEEP NEURAL NETWORK USING DEEP LEARNING

Ms. Rekha Kumari, Ms. Gurpreet Kaur
Assistant Professor
Department of Computer Science and Engineering
HMRITM, Delhi, India

Aditya Rawat, Harshit Chauhan, Kartik Singh Negi, Rishi Mishra
Department of Computer Science and Engineering
HMRITM, Delhi, India

Abstract: **Transformers were first used for natural language processing (NLP) tasks, but they quickly spread to other deep learning fields, including computer vision. They assess the interdependence of pairs. Attention is a part that enables to dynamically highlight relevant features of the input data (words in the case of text strings, parts of images in the case of visual Transformers). The cost grows continually with the number of tokens. The most common Trans- former Architecture for image classification uses only the Transformer Encoder to transform the various input tokens. However, the decoder component of the traditional Transformer Architecture is also used in a variety of other applications. In this section, we first introduce the Attention Mechanism (Section 1), followed by the Basic Transformer Block, which includes the Vision Transformer (Section 2).**

Keywords: **Transformer, Deep Neural Network, attention, Visual Transformers, Deep Learning, multi-modal attention**

I. INTRODUCTION

With the release of the paper "Attention Is All You Need" by Vaswani et al. in 2017, a new type of neural network design known as a transformer was developed. Transformers have gained popularity for a variety of natural language processing (NLP) applications, including machine translation, question answering, and text summarization. Transformers are meant to interpret sequential data, such as natural language text.

Initially, Transformers were developed for carrying out natural language processing activities, but they have now gained acceptance in various other domains of deep learning, such as computer vision. The Transformer design evaluates the connections between input tokens in pairs, which could be words in the case of textual strings or segments of visuals for visual Transformers. This association is ascertained via an attention mechanism, which can become computationally demanding with an increase in the number of tokens.

In image classification, the Transformer Encoder is the most commonly used Transformer design that is employed to transform the input tokens.

Nevertheless, there are other applications that also integrate the decoder component of the conventional Transformer design.

This introduction will explore the fundamental aspects of the Transformer design, with a specific emphasis on the attention mechanism (Section 1) and the Vision Transformer (Section 2), which is an adaptation of the Transformer design for computer vision activities. We will scrutinize how these mechanisms have revolutionized the field of computer vision and discuss their potential in enhancing the precision and efficiency of visual recognition tasks.

Transformers are special in that they heavily rely on systems for self-attention, which enables them to choose concentrate on various segments of the input sequence while processing. The performance of NLP has significantly improved as a result of this attention-based approach, especially for tasks that call for the model to comprehend long-range dependencies and interactions between various elements of the input sequence.

II. ATTENTION

One of the most important developments in deep learning research during the past ten years is the attention mechanism in NLP. Natural language processing (NLP) innovations like Google's BERT and the Transformer architecture have arisen as a result of it.

Given two lists of tokens, $X \in R^{N \times dx}$ and $Y \in R^{N \times dy}$, attention encodes information from Y into X, where N is the length of inputs X and Y, d_x and d_y are their respective dimensions. For this, we first define three linear mappings: query mapping $W^Q \in R^{dx \times dq}$, key mapping $W^K \in R^{dy \times dk}$ and value mapping $W^V \in R^{dy \times dv}$, where d_q, d_k, and d_v is the embedding dimension in which the query,

International Journal of Engineering Applied Sciences and Technology, 2023
Vol. 8, Issue 02, ISSN No. 2455-2143, Pages 313-319
Published Online June 2023 in IJEAST (http://www.ijeast.com)

key, and value are going to be computed, respectively.

Then, we define the query Q, key K and value V as:
$Q = XW^Q$
$K = Y W^K$
$V = YW^V$
Next, the attention matrix is defined as:

$$A(Q, K) = \text{Softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right) \quad (1)$$

The nominator $QK^T \in R^{N \times N}$ represents how each part of the input in X attends to each part of the input in Y. This dot-product is then put through the Softmax function to normalize its values and get positive values that add to 1. However, for large values of dk, this may result in the Softmax to have incredibly small gradients, so it is scaled down by $\sqrt{d_k}$.
The resulting N ×N matrix encodes the relationship between X with respect to Y: it measures how important a token in X is with respect to another one in Y.

Finally, the attention output is defined as:
Attention (Q, K, V) = A (Q, K) V(2)
Consider the following movie review to better understand the concept of attention and its significance: "Despite some claims that the film's plot is boring, I found it to be captivating." A computer may not be able to tell whether this review is positive or negative because the first part of the sentence appears to be negative while the second part expresses a positive sentiment. The meaning of a word or phrase can only be understood in relation to the other words in the sentence, so context is critical in natural language processing. In the case of the movie review, the word "boring" alone would imply a negative review, but its contextual relationship with the other words allows the reader to reach a correct conclusion. Similarly, in computer vision tasks like object detection, the context of a pixel is critical for determining its true nature. In sequential data, the concept of context can be formalized through attention mechanisms, which allow deep learning models to focus on important parts of the input sequence when making prediction.

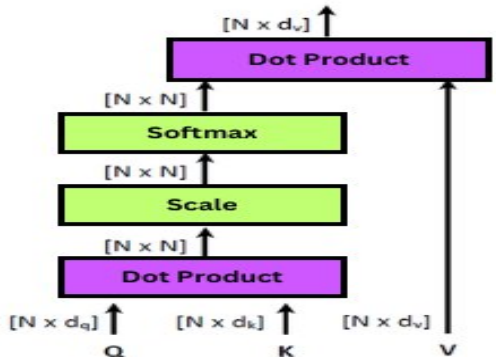

Figure 1: Attention block. Next to each element, we denote its dimensionality. Figure inspired from [5]

1.1 Types Of Attention

Cross-attention and self-attention are the two main divisions of attention mechanisms [4]. Cross-attention uses queries from an input different from the key and value vectors, whereas self-attention uses queries from the same input, i.e., X = Y. These mechanisms are explained in more detail in the subsequent sections.

Self-Attention
The tokens of X pay attention to themselves while they are in self-attention (X=Y). The following is how it is modelled:

$$SA(X) = \text{Attention}(XW^Q, XW^K, XW^V) \quad (3)$$

By comprehending how various components are connected, self-attention defines the concept of context. The underlying structures connecting different elements of the input are identified. A token can examine its nearby tokens to produce its output when given a sequence of tokens. By learning from the neighbors' analyses, a token can gather insights from the same group.

Cross-Attention
Many types of data in the real world exhibit multimodal characteristics. For example, videos consist of frames, audio, and subtitles, while images are accompanied by captions. Consequently, models capable of processing such multimodal information have become crucial.

Cross-attention is an attention mechanism that is specifically designed to handle inputs of a multimodal nature. Unlike self-attention, it involves extracting queries from one input source and key-value pairs from another (where $X \neq Y$). The question it seeks to answer is: "Which elements of input X correspond to those in input Y?" The definition of cross-attention (CA) is as follows:

$$CA(X,Y) = \text{Attention}(XW^Q, YW^K, YW^V) \quad (4)$$

Local-Attention
The local attention is a type of attention mechanism that has been widely used in neural machine translation and other sequence-to-sequence tasks. In contrast to global attention, which attends to the entire input sequence, local attention restricts the attention window to a subset of positions around the current position. This means that the model only attends to a limited context of the input sequence, which is centered around the current position. The size of the attention window can be controlled by the center position and width parameter, allowing the model to focus on a relevant subset of the input sequence for generating the output, as shown in Figure 2.

Local attention is particularly useful in cases where the input sequence is very long and attending to all positions would be computationally expensive and unnecessary for

International Journal of Engineering Applied Sciences and Technology, 2023
Vol. 8, Issue 02, ISSN No. 2455-2143, Pages 313-319
Published Online June 2023 in IJEAST (http://www.ijeast.com)

generating the output. By attending only to a subset of positions around the current position, the model can reduce the computational complexity and improve the efficiency of the attention mechanism. Moreover, local attention can capture more fine-grained relationships between the input and output sequences by attending to only the most relevant positions, which can lead to better translation performance.

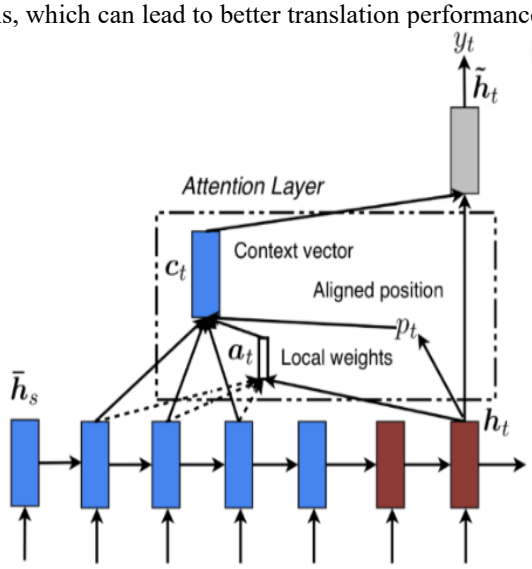

Figure 2: Local Attention

Global-Attention

Global attention is a type of attention mechanism used in deep learning that enables models to attend to all parts of an input sequence when making predictions. Unlike local attention, which only attends to a limited set of positions around a target position, global attention allows the model to access information from the entire input sequence. This makes it particularly useful for tasks where the entire input sequence is relevant to the output, such as machine translation or speech recognition. In global attention, a weighted sum of all input positions is computed based on the similarity between the target position and each input position. This weighted sum is then used as a context vector that is passed to the decoder to generate the output, as depicted in figure 3.

Global attention is particularly useful for models that need to generate long sequences, as it allows the model to consider all input positions when making each prediction. For example, in machine translation, the input sequence may be a long sentence that needs to be translated into another language. In this case, global attention can be used to ensure that the entire sentence is considered when generating each word of the translation. This can lead to more accurate and fluent translations, as the model has access to all the information it needs to make the best prediction at each step. Overall, global attention is a powerful tool that has been shown to improve the performance of many deep learning models on a variety of tasks.

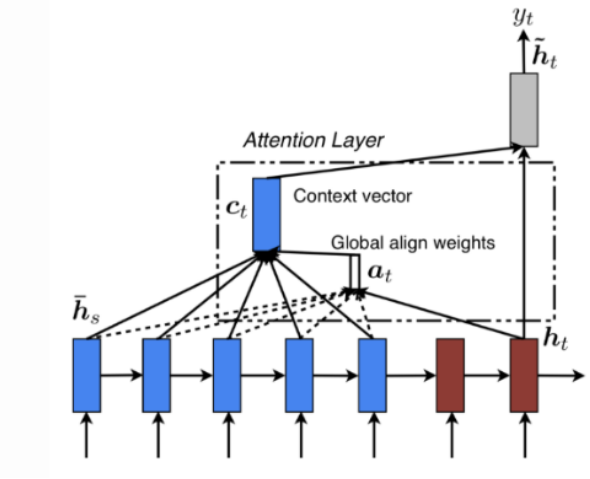

Figure 3: Global Attention

1.2 Variation of Attention

Attention is typically employed in two ways:

(1) Multi-head Self-Attention
(2) Masked Multi-head Attention

Attention Head. We call Attention Head the mechanism i.e., query-key-value projection, followed by scaled dot product attention (Equations 1 and 2).

When employing an attention-based model, relying only on a single attention head can inhibit learning. Therefore, the Multi-head Attention block is introduced. [4]

MSA stands for Multi-Head Self-Attention. Figure 4depicts MSA, which is defined as:

$$\text{MSA}(X) = \text{Concat}(\text{head}_1(X),\ldots,\text{head}_h(X))W^O,$$
$$\text{head}_i(X) = \text{SA}(X), \forall \; _i \in \{1, h\} \quad (5)$$

Concat is the union of h attention heads, and $W^O \in R^{h\times d}$ is the projection matrix. As a result, the initial embedding dimension d_x is decomposed into $h*d_v$, and the computation is performed independently for each head. To match the desired output dimension, the independent attention heads are usually concatenated and multiplied by a linear layer. Often, the output dimension is the same as the input embedding dimension d. This facilitates the stacking of multiple blocks.

Multi-head Cross-Attention (MCA). Similar to MSA, MCA is defined as:

$$\text{MCA}(X, Y) = \text{Concat}(\text{head}_1(X, Y), \ldots,\text{head}_h(X, Y))W^O,\text{head}_i(X, Y) = \text{CA}(X, Y), \forall \; _i \in \{1, h\} \quad (6)$$

International Journal of Engineering Applied Sciences and Technology, 2023
Vol. 8, Issue 02, ISSN No. 2455-2143, Pages 313-319
Published Online June 2023 in IJEAST (http://www.ijeast.com)

MMSA stands for Masked Multi-head Self-Attention.
Another type of attention is the MMSA layer [4]. It has the same structure as the Multi-head Self-Attention block except that all of the later vectors in the target output are masked. When dealing with sequential data, this can aid in parallelizing training.

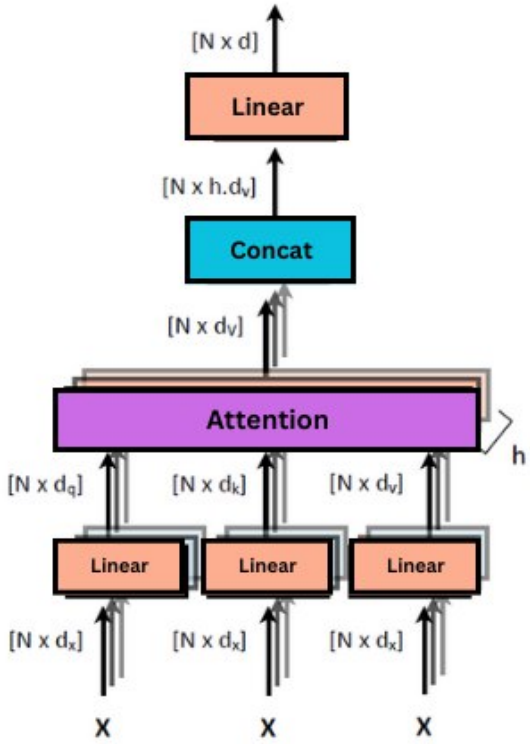

Figure 1: **Multi-head Self-Attention block (MSA)**.

First, the input X is projected to queries, keys and values and then passed through h attention blocks. The h resulting attention outputs are then concatenated together and finally projected to a d-dimensional output vector. Next to each element, we denote its dimensionality. Figure inspired from [5].

1.3 Properties Of Attention

While attention encodes contextual relationships, it is permutation equivalent because the mechanism does not account for input data order.
The attention computations are all matrix multiplications and normalization, as shown in Equation 2. As a result, a permuted input yields a permuted output. However, in practice, this may not be an accurate representation of the information. Consider the sentences 'the monkey ate the banana' and 'the banana ate the monkey'. Because of the order of the words, they have different meanings. If the order of the input is critical, various mechanisms, such as the Positional Encoding are used to capture this nuance.

III. VISUAL TRANSFORMERS

The Transformer architecture was introduced in [4], and it is the first architecture that relies solely on attention to connect inputs and outputs. Since its introduction, it has revolutionized Deep Learning, making breakthroughs in a variety of fields such as Natural Language Processing, Computer Vision, Chemistry, and Biology on its way to becoming the default architecture for learning representations. The standard Transformer [4] has recently been adapted for vision tasks [5]. Once again, visual Transformer has emerged as a key architecture in computer vision.
In this section, we first introduce the basic architecture of Transformers (Section 3.1).

3.1Basic Architecture of Transformer

As depicted in Figure 3, the Transformer architecture is a model that comprises an encoder-decoder. Initially, it incorporates input tokens $X = (x_1,. . .,x_N)$ into a hidden space, resulting in hidden vectors $Z = (z_1,. . .,z_N)$, which are subsequently provided to the decoder to generate output $Y = (y_1,. . .,y_M)$. The encoder constitutes a series of L layers, with each layer comprising of two sub-components: Multi-head Self-Attention (MSA) layers and a Multi-Layer Perceptron (MLP). Similarly, the decoder is also composed of L layers, with each layer having three sub-components: Masked Multi-head Self-Attention (MMSA), Multi-head Cross-Attention (MCA), and a MLP.

International Journal of Engineering Applied Sciences and Technology, 2023
Vol. 8, Issue 02, ISSN No. 2455-2143, Pages 313-319
Published Online June 2023 in IJEAST (http://www.ijeast.com)

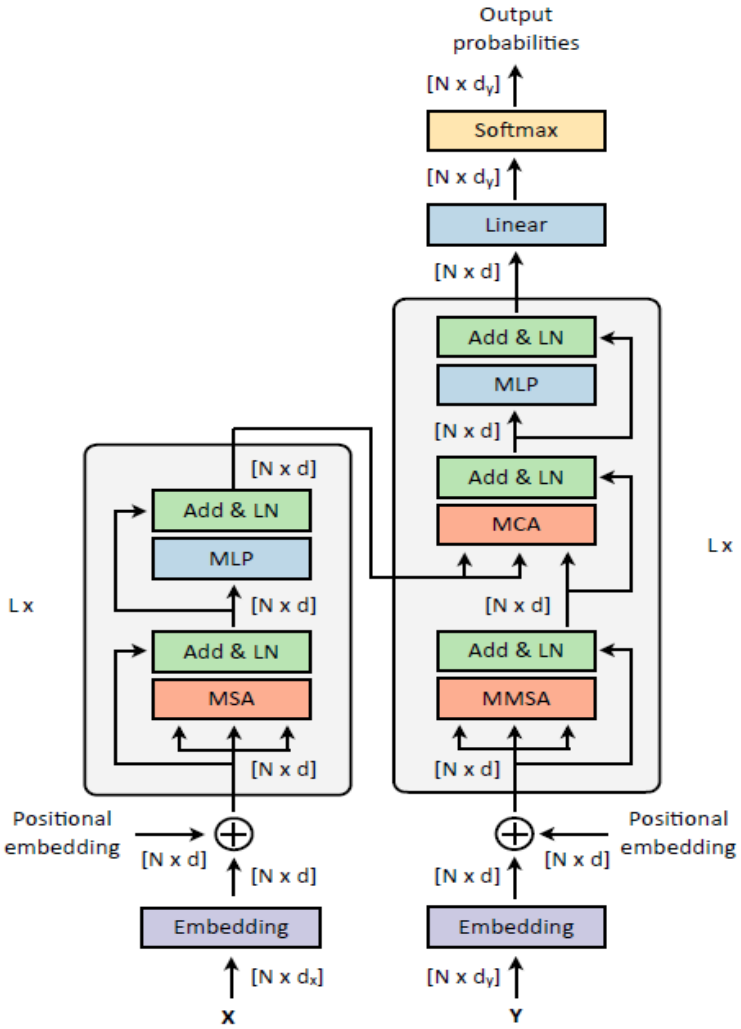

Figure 5: **The transformer architecture.**

It consists of an encoder (left) and a decoder (right) block, each one consisting from a series of attention blocks(multi-head and Masked Multi-head attention) and MLP layers. Next to each element, we denote its dimensionally.[5]

Overview

In this article, we will outline the different components of the Transformer structure as shown in Figure 3. Initially, the input tokens are transformed into embedding tokens (Section 3.1.1). Next, Positional encoding is applied to each embedding token, adding a positional token to signify the token order (Section 3.1.2). The Transformer encoder is then introduced (Section 3.1.3), consisting of L multi-head attention, normalization, and MLP layers that encode the input into semantically meaningful features. Following that, the decoder is presented (Section 2.1.4), which includes L stacked masked multi-head attention, multi-head attention, and MLP layers, along with normalizations, to decode the input features with regards to the output embedding tokens. Lastly, the output is projected to linear and Softmax layers.

Embedding

The initial stage of Transformers involves transforming input tokens[2], into embedding tokens, which are vectors containing significant characteristics. This is accomplished by projecting each input into an embedding space to produce embedding tokens Z^e, following the conventional technique. The embedding space is organized such that the proximity between two vectors is linked to the semantic similarity of the corresponding words. In the context of the starting NLP case, this implies that we obtain a vector for each word, with the vectors closer to each other sharing similar meanings.

International Journal of Engineering Applied Sciences and Technology, 2023
Vol. 8, Issue 02, ISSN No. 2455-2143, Pages 313-319
Published Online June 2023 in IJEAST (http://www.ijeast.com)

Positional Encoding
As mentioned in Section 1.5, the focus mechanism is not reliant on the position of each input, making it positional agnostic. Nevertheless, the sequence of input tokens is generally significant and should be considered. For example, the arrangement of words in a sentence can alter its meaning. To address this issue, [4] proposed the inclusion of a Positional Encoding **PE** $\in R^{N\times d_x}$, which appends a positional token to each embedding token $Z^e \in R^{N\times d_x}$.

Sinusoidal Positional Encoding. The Sinusoidal Positional Encoding [4] is the main positional encoding method, which encodes the position of each token with sinusoidal waves of multiples frequency. For an embedding token $Z^e \in R^{N\times d_x}$, its Positional Encoding **PE** $\in R^{N\times d_x}$ is Transformers.

Defined as:

$$PE(i,2j) = \sin\left(\frac{i}{10000^{2j/d}}\right)$$

$$PE(i,2j+1) = \cos\left(\frac{i}{10000^{2j/d}}\right), \forall_{i,j} \in [|1,n|]x[|1,d|] \quad (7)$$

Learnable Positional Encoding
Encoding of Position that is Learnable. Another method is to allow the model to learn the positional encoding in an orthogonal manner. In this situation, **PE** $\in R^{N\times d_x}$ becomes a parameter that can be learned. Nevertheless, this leads to an increase in memory requirements, without necessarily resulting in enhancements over the sinusoidal encoding.

Positional Encoding
After its computation, the Positional Encoding PE is either added to the embedding tokens, or they are concatenated as follows:

$$Z^{pe} = Z^e + PE \text{ , or}$$
$$Z^{pe} = \text{Concat}(Z^e + PE) \text{ ,} \quad (8)$$

where Concat denotes vector concatenation. Note that the concatenation has the advantage of not altering the information contained in Z^e, since the positional information is only added to the unused dimension.
Nevertheless, it augments the input dimension, leading to higher memory requirements. Instead, the addition does preserve the same input dimension, while altering the content of the embedding tokens. When the input dimension is high, this content altering is trivial, as most of the content is preserved. Therefore, in practice, for high-dimension summing positional encodings is preferred, whereas for low dimensions, concatenating them prevails.

Encoder Block
The encoder module receives the token embeddings and positional tokens as input and generates input features, which are then decoded by the decoder module. The encoder is composed of L Multi-head Self-Attention (MSA) layers and a Multi-Layer Perceptron (MLP). The input, $Z\frac{pe}{x} \in R^{N\times d}$, is processed through a multi-head self-attention layer, followed by a residual connection with layer normalization, which is executed after each sub-layer in the Transformer. The output of each sub-layer is then fed into an MLP and a normalization layer. This process is repeated L times, where each encoder block (of size N x d) outputs the input for the subsequent block. On the Lth iteration, the output of the normalization layer becomes the input for the cross-attention block in the decoder.

Decoder Block
The decipherer possesses two inputs: first, an input that constitutes the inquiries Q $\in R^{N\times d}$, of the encoder, and second, the outcome of the encoder that comprises the key-value **K,V** $\in R^{N\times d}$, pair. Similar to Sections 3.1.1-3.1.2, the initial step involves encoding the outcome symbol to outcome embedding symbol and outcome positional symbol. These symbols are inserted into the primary component of the decipherer, which comprises a stack of L Masked Multi-head Self-Attention (MMSA) layers, Multi-Head Cross-Attention (MCA) layers, and Multi-Layer Perceptron (MLP) followed by normalizations. Specifically, the embedding and positional symbols, $Z\frac{pe}{x} \in R^{N\times d}$, undergo a MMSA block. Then, a residual connection with layer normalization ensues. Next, a MCA layer (followed by normalization) maps the inquiries to the encoded keys-values before forwarding the outcome to a MLP. Finally, we project the outcome of the L decipherer blocks (of dimension N x d_y) through a linear layer and obtain outcome probability via a soft-max layer.

IV. CONCLUSION

The technique of attention is an effective and instinctive method that allows for the management of both local and global indications. The Transformer was the first pure attention-based architecture developed for NLP applications. Soon after, the Transformer architecture was adopted by the Computer Vision domain for image classification, leading to the creation of the first visual Transformer model known as the Vision Transformer.
The technique of attention is an effective and instinctive method that allows for the management of both local and global indications. The Transformer was the first pure attention-based architecture developed for NLP applications. Soon after, the Transformer architecture was adopted by the Computer Vision domain for image classification, leading to the creation of the first visual Transformer model known as the Vision Transformer.
Although Transformers naturally result in high performance, the raw attention mechanism is a computationally

demanding and resource-intensive technique. Therefore, many improved and refined versions of attention mechanisms. Subsequently, Transformer-based architectures have been successfully implemented in various other tasks, including object detection, image segmentation, self-supervised learning, and image generation. Moreover, Transformer-based architectures are particularly suitable for handling multidimensional tasks since multimodal signals can be easily integrated through attention blocks, especially for vision and language cues and spatiotemporal signals can also be easily managed.

Therefore, Transformer-based architectures have facilitated significant progress in many fields in recent years. In the future, Transformers will need to become increasingly computationally efficient, for example, to be usable on mobile devices, and will play a crucial role in addressing multimodal challenges and bridging most AI fields.

V. REFERENCES

[1]. Bahdanau D, Cho K, Bengio Y (2015) Neural machine translation by jointly learning to align and translate. In: International Conference on Learning Representations

[2]. Khan S, Naseer M, Hayat M, Zamir SW, Khan FS, Shah M (2021) Transformers in vision: A survey. ACM Computing Surveys

[3]. Sutskever I, Vinyals O, Le QV (2014) Sequence to sequence learning with neural networks. In: Advances in Neural Information Processing Systems, vol 27

[4]. Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, Kaiser L, Polosukhin I (2017) Attention is all you need. In: Advances in Neural Information Processing Systems, vol 30

[5]. Robin Courant1, Maika Edberg, Nicolas Dufour, and Vicky Kalogeiton (2019) Machine Learning for Brain Disorders(CSUR).