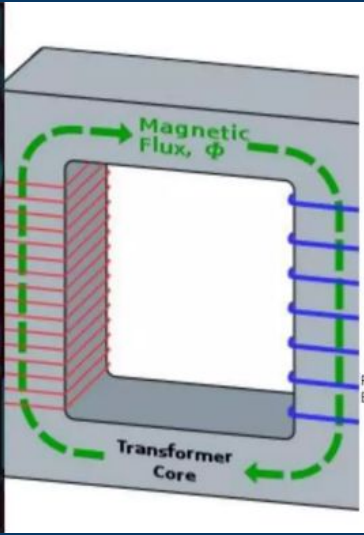
Transformers in Vision-An Overview

Arshit Mankodi
me22b026

What are transformers ?



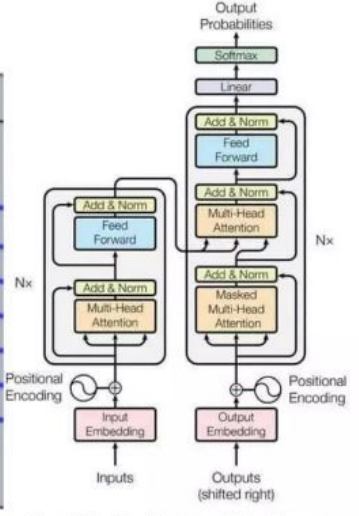


Figure 1: The Transformer - model architecture.

Transformers at school

Transformers during JEE

Transformers at the SENAI Lab

A brief history of NLP Including the development of Transformers

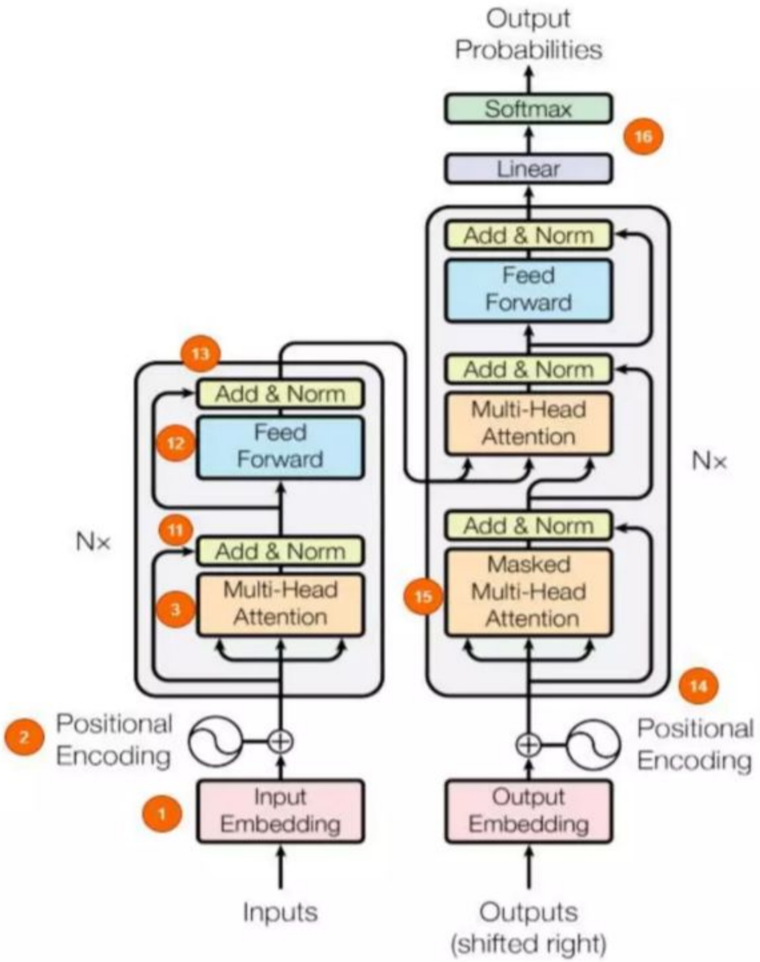
Note: The next three slides consist of the major advancements in NLP with the titles as links to the research papers. The list is in chronological order with a brief description given alongside.

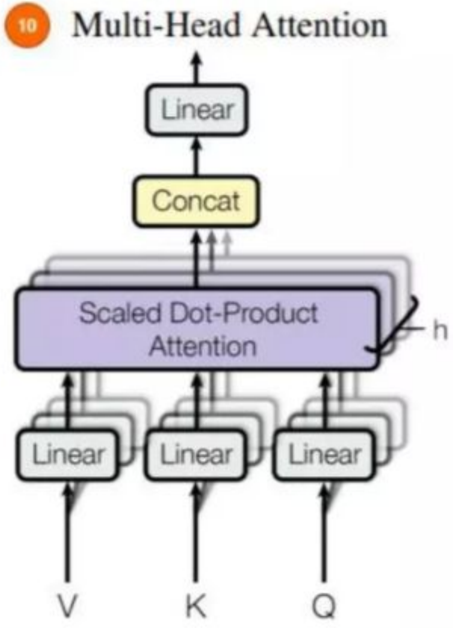
- **Bag of Words (BOW)** [1954]: count the occurrences of each word in the documents and use them as features.
- **TF-IDF** [1972]: the BOW scores are modified so that rare words have high scores and common words have low scores.
- **Word2Vec** [2013]: each word is mapped to a high-dimensional vector called word embedding, which captures its semantic. Word embeddings are learned by a neural network looking for word correlations on a large corpus.
- **RNN** [1986]: RNNs compute document embeddings leveraging word context in sentences, which was not possible with word embeddings alone. Later evolved with **LSTM** [1997] to capture long-term dependencies, and to **Bidirectional RNN** [1997] to capture left-to-right and right-to-left dependencies. Eventually, **Encoder-Decoder RNNs** [2014] emerged, where an RNN creates a document embedding (i.e. the encoder) and another RNN decodes it into text (i.e. the decoder).
- **Transformer** [2017]: an encoder-decoder model that leverages attention mechanisms to compute better embeddings and to better align output to input.
- **BERT** [2018]: bidirectional Transformer pre-trained using a combination of Masked Language Modeling and Next Sentence Prediction objectives. It uses global attention.
- **GPT** [2018]: the first autoregressive model based on the Transformer architecture. Later evolved into **GPT-2** [2019], a bigger and optimized version of GPT pre-trained on WebText, and **GPT-3** [2020], a further bigger and optimized version of GPT-2, pre-trained on Common Crawl.

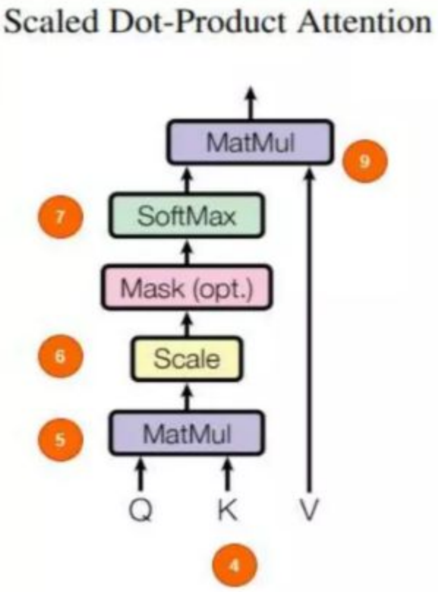
- **CTRL** [2019]: similar to GPT but with control codes for conditional text generation.
- **Transformer-XL** [2019]: it's an autoregressive Transformer that can reuse previously computed hidden-states to attend to longer context.
- **ALBERT** [2019]: a lighter version of BERT, where (1) Next Sentence Prediction is replaced by Sentence Order Prediction, and (2) parameter-reduction techniques are used for lower memory consumption and faster training.
- **RoBERTa** [2019]: better version of BERT, where (1) the Masked Language Modeling objective is dynamic, (2) the Next Sentence Prediction objective is dropped, (3) the BPE tokenizer is employed, and (4) better hyperparameters are used.
- **XLM** [2019]: Transformer pre-trained on a corpus of several languages using objectives like Causal Language Modeling, Masked Language Modeling, and Translation Language Modeling.
- **XLNet** [2019]: Transformer-XL with a generalized autoregressive pre-training method that enables learning bidirectional dependences.
- **PEGASUS** [2019]: a bidirectional encoder and a left-to-right decoder pre-trained with Masked Language Modeling and Gap Sentence Generation objectives.
- **DistilBERT** [2019]: same as BERT but smaller and faster, while preserving over 95% of BERT's performances. Trained by distillation of the pre-trained BERT model.
- **XLM-RoBERTa** [2019]: RoBERTa trained on a multilanguage corpus with the Masked Language Modeling objective.

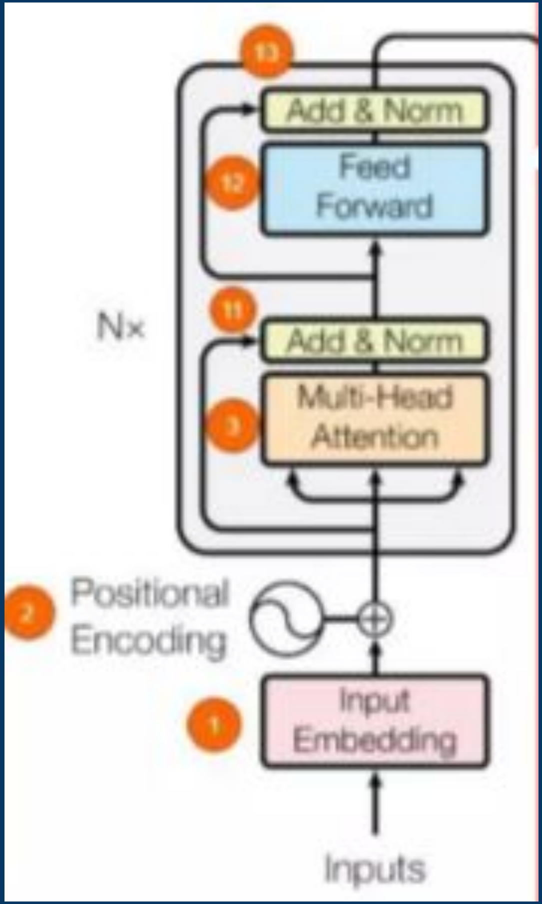
- **BART** [2019]: a bidirectional encoder and a left-to-right decoder trained by corrupting text with an arbitrary noising function and learning a model to reconstruct the original text.
- **ConvBERT** [2019]: a better version of BERT, where self-attention blocks are replaced with new ones that leverage convolutions to better model global and local context.
- **Funnel Transformer** [2020]: a type of Transformer that gradually compresses the sequence of hidden states to a shorter one and hence reduces the computation cost.
- **Reformer** [2020]: a more efficient Transformer thanks to local-sensitive hashing attention, axial position encoding, and other optimizations.
- **T5** [2020]: a bidirectional encoder and a left-to-right decoder pre-trained on a mix of unsupervised and supervised tasks.
- **Longformer** [2020]: a Transformer model replacing the attention matrices with sparse matrices for higher training efficiency.
- **ProphetNet** [2020]: a Transformer model trained with the Future N-gram Prediction objective and with a novel self-attention mechanism.
- **ELECTRA** [2020]: same as BERT but lighter and better. The model is trained with the Replaced Token Detection objective.
- **Switch Transformers** [2021]: a sparsely-activated expert Transformer model that aims to simplify and improve over Mixture of Experts.

A technical dissection of the Transformer model


Output
Probabilities
Softmax
16
Linear
Add & Norm
Feed
Forward
Add & Norm
Multi-Head
Attention
Nx
Add & Norm
Masked
Multi-Head
Attention
15
14
13
Add & Norm
12
Feed
Forward
11
Add & Norm
Multi-Head
Attention
3
Nx
2
Positional
Encoding
Positional
Encoding
1
Input
Embedding
Output
Embedding
Inputs
Outputs
(shifted right)


10
Multi-Head Attention
Linear
Concat
Scaled Dot-Product
Attention
h
Linear
Linear
Linear
V
K
Q

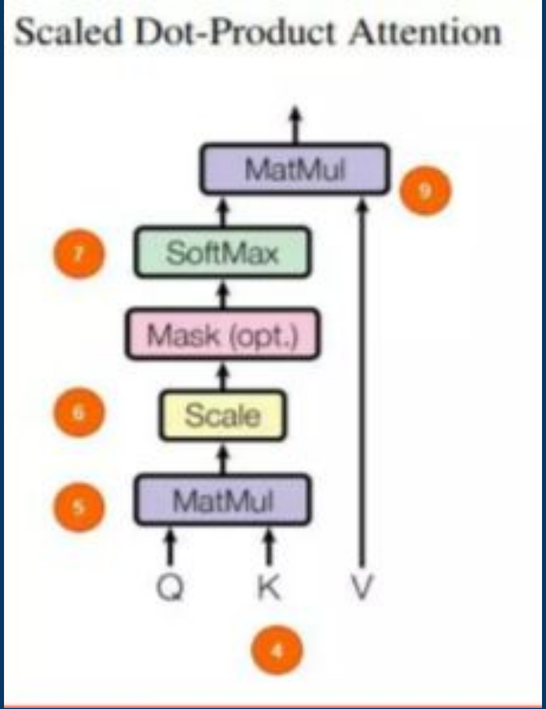

Scaled Dot-Product Attention
MatMul
9
7
SoftMax
Mask (opt.)
6
Scale
5
MatMul
Q
K
V
4



(1) The input data first gets embedded into a vector. The embedding layer helps us grab a learned vector representation for each word.

(2) In the next stage a positional encoding is injected into the input embeddings. This is because a transformer has no idea about the order of the sequence that is being passed as input-for example a sentence.

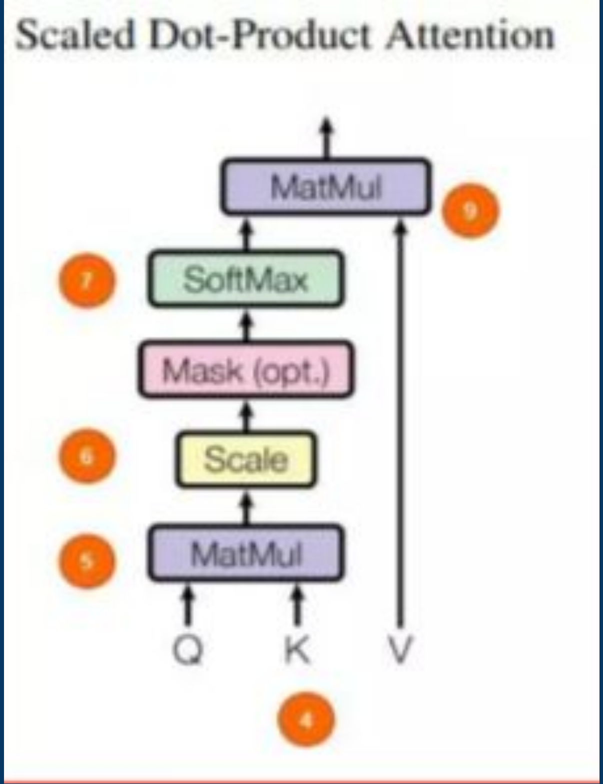
(3) Now the multi-headed attention is where things get a little different.



(4) Multi-Headed Attention consists of three learnable vectors. Query, Key and Value vectors. The motivation of this reportedly comes from information retrieval where you search (query) and the search engine compares your query with a key and responds with a value.

(5) The Q and K representations undergo a dot product matrix multiplication to produce a score matrix which represents how much a word has to attend to every other word. Higher score means more attention and vice-versa.

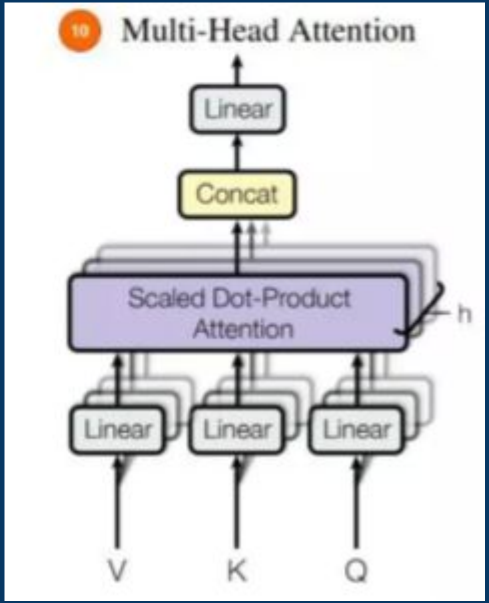
(6) Then the Score matrix is scaled down according to the dimensions of the Q and K vectors. This is to ensure more stable gradients as multiplication can have exploding effects.



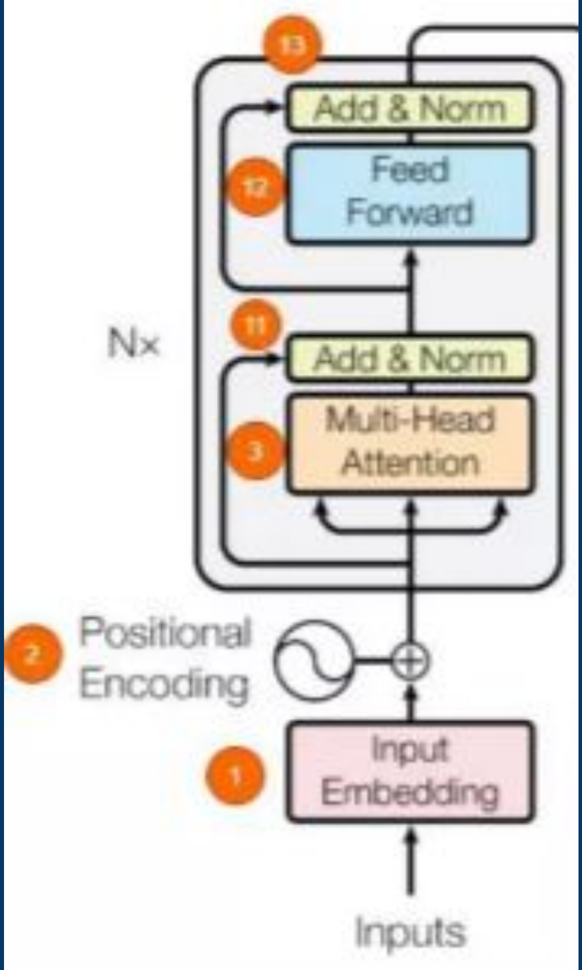
(7) Next the Score matrix is softmaxed to turn attention scores into probabilities. Obviously higher scores are heightened and lower scores are depressed. This ensures the model to be confident on which words to attend to.

(8) Then the resultant matrix with probabilities is multiplied with the value vector. This will make the higher probability scores the model has learned to be more important. The low scoring words will effectively drown out to become irrelevant.

(9) Then, the concatenated output of QK and V vectors are fed into the Linear layer to process further.



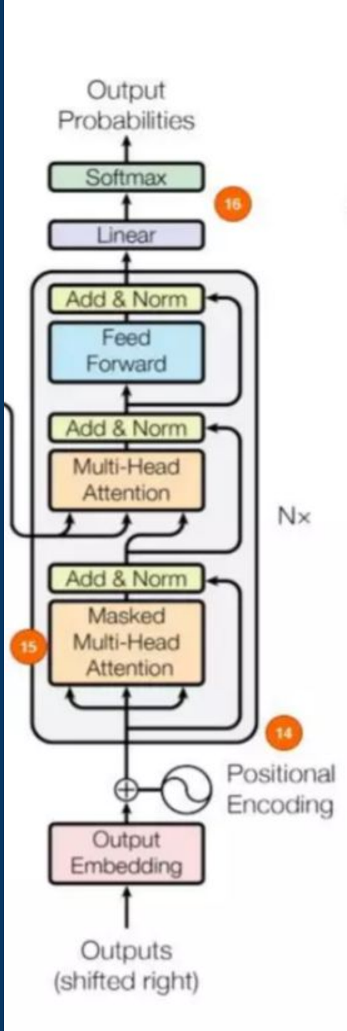
(10) Self-Attention is performed for each word in the sequence. Since one doesn't depend on the other a copy of the self attention module can be used to process everything simultaneously making this multi- headed.



(11) Then the output value vectors are concatenated and added to the residual connection coming from the input layer and then the resultant representation is passed into a LayerNorm for normalization. (Residual connection help gradients flow through the network and LayernNorm helps reduce the training time by a small fraction and stabilize the network

(12) Further, the output is passed into a point-wise feed forward network to obtain an even richer representation.

(13) The outputs are again Layer-normed and residuals are added from the previous layer.



(14) The output from the encoder along with the inputs (if any) from the previous time steps/words are fed into the decoder where the outputs undergo masked-multi headed attention before being fed into the next attention layer along with the output from encoder.

(15) Masked multi headed attention is necessary because the network shouldn't have any visibility into the words that are to come later in the sequence while decoding, to ensure there is no leak. This is done by masking the entries of words that come later in the series in the Score matrix. Current and previous words in the sequence are added with 1 and the future word scores are added with-inf. This ensures the future words in the series get drowned out into 0 when performing softmax to obtain the probabilities, while the rest are retained.

(16) There are residual connections here as well, to improve the flow of gradients. Finally the output is sent to a Linear layer and softmaxed to obtain the outputs in probabilities.

Interval

Transformers - Application in computer vision

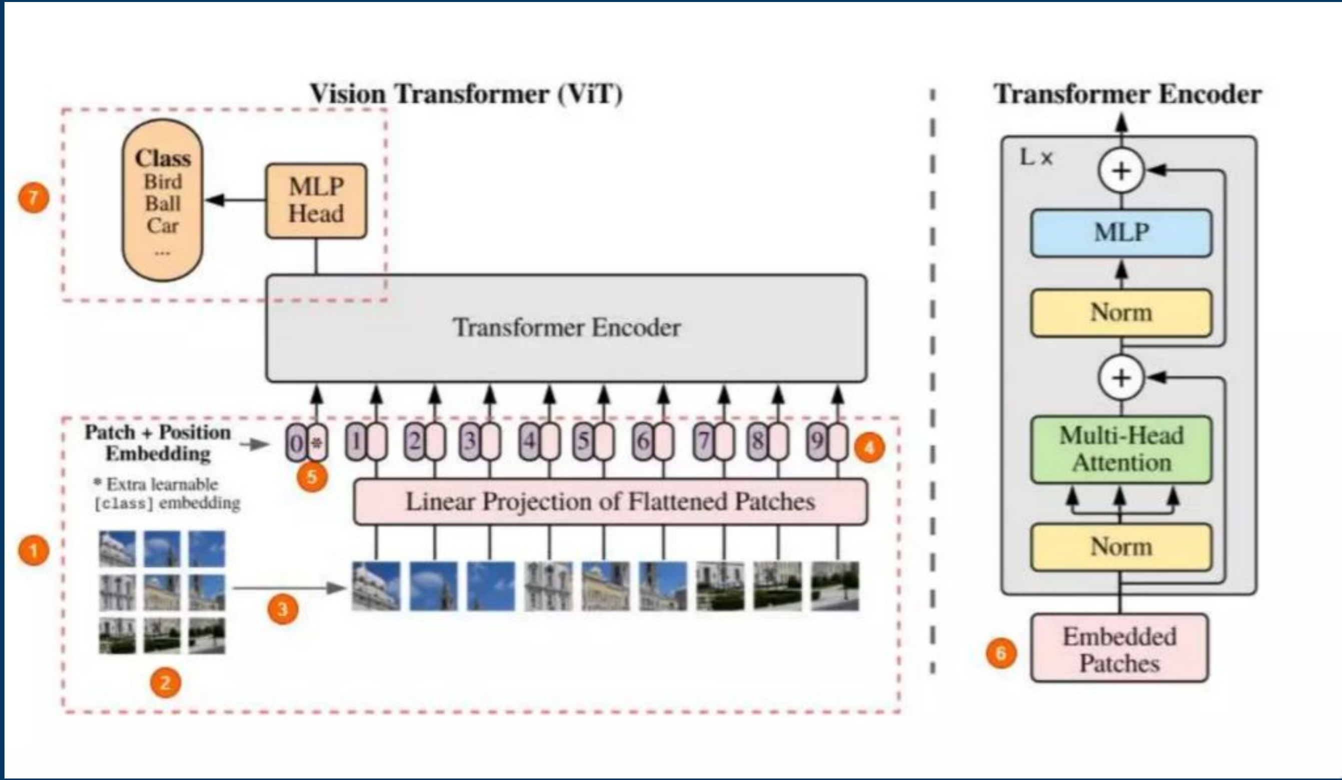
Vision Transformers - Overview

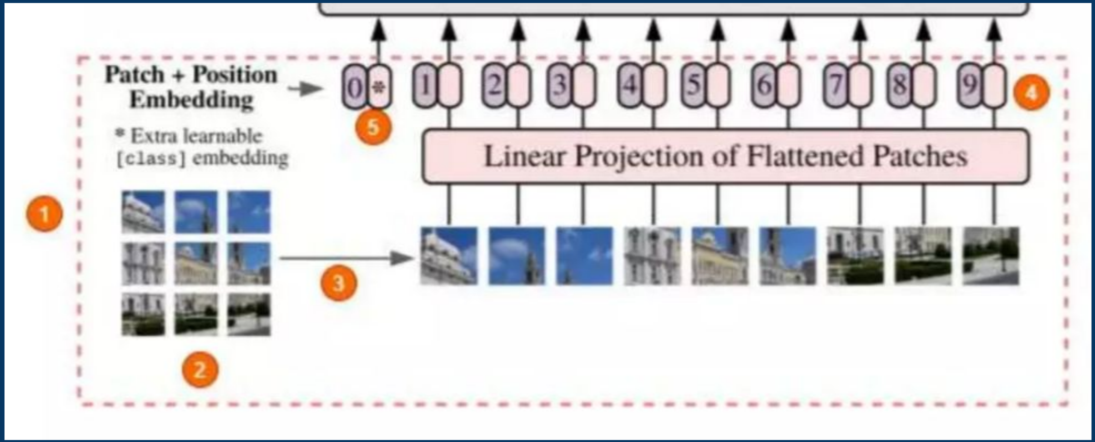
A vision transformer (ViT) is a transformer designed for computer vision. A ViT breaks down an input image into a series of patches (rather than breaking up text into tokens), serialises each patch into a vector, and maps it to a smaller dimension with a single matrix multiplication. These vector embeddings are then processed by a transformer encoder as if they were token embeddings.

Variants:

- Original ViT
- Masked Autoencoders
- Swin Transformer
- ViT-VQGAN

Architecture of a vision transformer

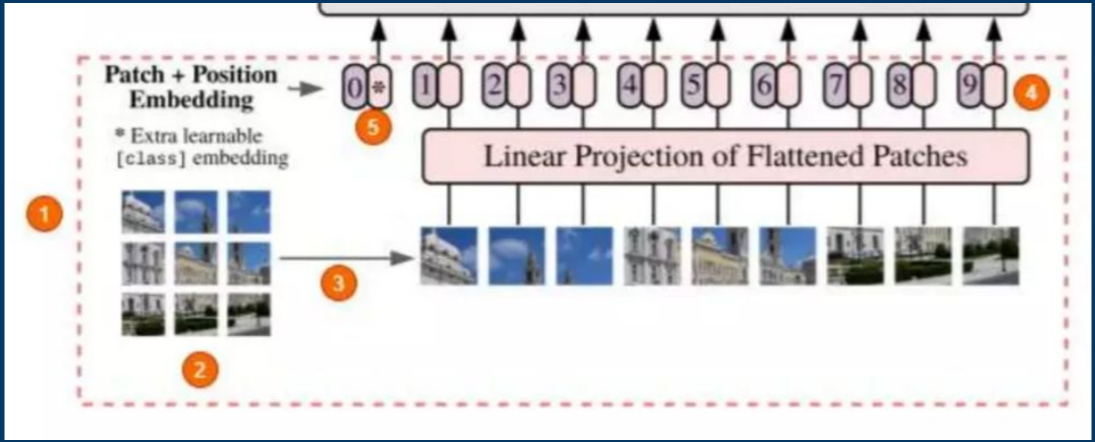




(1) They are only using the Encoder part of the transformer but the difference is in how they are feeding the images into the network.

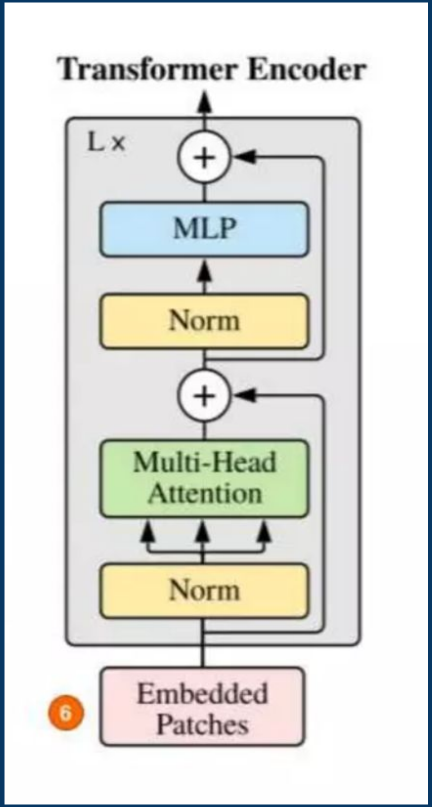
(2) They are breaking down the image into fixed size patches. So one of these patches can be of dimension 16x16 or 32x32 as proposed in the paper. More patches means more simpler it is to train these networks as the patches themselves get smaller. Hence we have that in the title - "An Image is worth 16x16 words".

(3) The patches are then unrolled (flattened) and sent for further processing into the network.



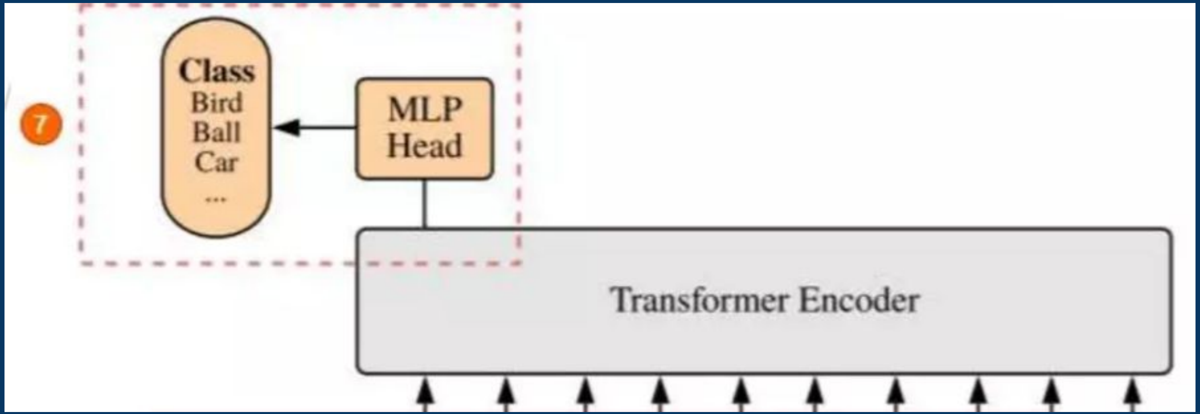
(4) Unlike NNs here the model has no idea whatsoever about the position of the samples in the sequence, here each sample is a patch from the input image. So the image is fed along with a positional embedding vector and into the encoder. One thing to note here is the positional embeddings are also learnable so you don't actually feed hard-coded vectors w.r.t to their positions.

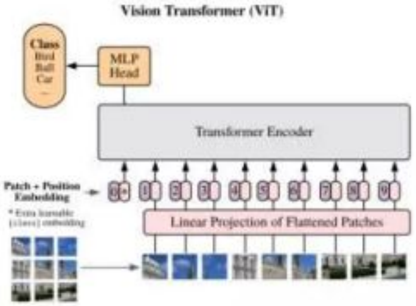
(5) There is also a special token at the start just like BERT in NLP.



(6) So each image patch is first unrolled (flattened) into a big vector and gets multiplied with an embedding matrix which is also learnable, creating embedded patches. And these embedded patches are combined with the positional embedding vector and that gets fed into the Transformer.

(7) With the only difference being, instead of a decoder the output from the encoder is passed directly into a Feed Forward Neural Network to obtain the classification output.





Dosovitskiy et.al, ICLR 2021

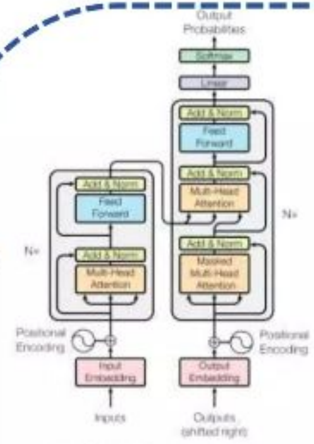
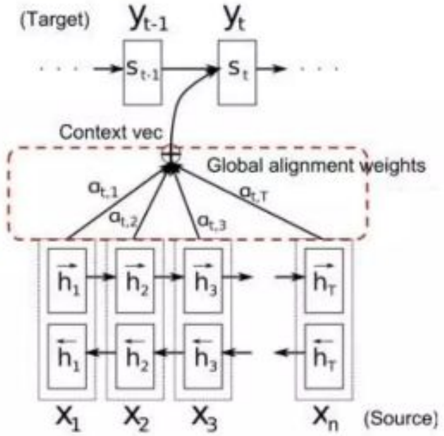
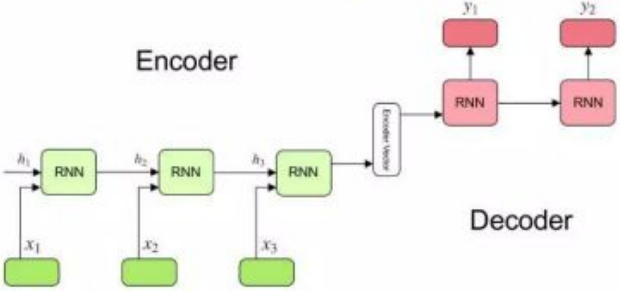


Figure 1: The Transformer - model architecture.

Vaswani et.al, NeurIPS 2017



Bahdanau et.al, ICLR 2015



Sutskever et.al, NeurIPS 2014

Quick summary of seminal papers.

Applications of Vision Transformers

- Image Classification
- Object Detection
- Video Deepfake Detection
- Image segmentation
- Anomaly detection
- Image Synthesis
- Cluster analysis
- Autonomous Driving

List of Tasks as mentioned in the paper

Task	Method	Metric	Dataset	Performance	Highlights	Limitations
Image Classification	ViT [11] ICLR'21	Top-1 Acc.	ImageNet	88.55	**a)** First application of Transformer (global self-attention) directly on image patches, **b)** Convolution-free network architecture, **c)** Outperforms CNN models such as ResNet.	**a)** Requires training on large-scale data *e.g.*, 300-Million images, **b)** Requires careful transfer learning to the new task, **c)** Requires large model with 632-Million parameters to achieve SOTA results.
	DeiT [12] arXiv'20	Top-1 Acc.	ImageNet	83.10	**a)** Successfully trains Transformer on ImageNet only, **b)** Introduces attention-based distillation method. **c)** Produces competitive performance with small (86-Million parameters) Transformers.	**a)** Requires access to pretrained CNN based teacher model thus performance depends on the quality of the teacher model.
	Swin-T [36] arXiv'21	Top-1 Acc.	ImageNet	84.5	**a)** Provides a general purpose backbone for different vision tasks e.g., classification, detection and segmentation **b)** A hierarchical design using shifted-windows operation.	**a)** Hard to train from scratch on smaller datasets **b)** Quadratic compute complexity inherent to the self-attention operation.
Low-Shot Learning	CT [25] NeurIPS'20	Top-1 Acc.	ImageNet COCO	62.25 60.35	**a)** Self-supervised pre-training mechanism that does not need manual labels, **b)** Dynamic inference using Transformer achieving stat-of-the-art results.	Proposed algorithm is limited in its capacity to perform on datasets that lack spatial details such as texture.

Object Detection	DETR [13] ECCV'20	AP	COCO	44.9	**a)** Use of Transformer allows end-to-end training pipeline for object detection, **b)** Removes the need for hand-crafted post-processing steps.	**a)** Performs poorly on small objects, **b)** Requires long training time to converge.
	D-DETR [14] ICLR'21	AP	COCO	43.8	**a)** Achieves better performance on small objects than DETR [13], **b)** Faster convergence than DETR [13]	Obtain SOTA results with 52.3 AP but with two stage detector design and test time augmentations.
Image Colorization	ColTran [24] ICLR'21	FID	ImageNet	19.71	**a)** First successful application of Transformer to image colorization, **b)** Achieves SOTA FID score.	**a)** Lacks end-to-end training, **b)** limited to images of size 256×256.
Action Recognition	ST-TR [216] arXiv'20	Top-1 Acc.	NTU 60/120	94.0/84.7	**a)** Successfully applies Transformer to model relations between body joints both in spatial and temporal domain, **b)** Achieves SOTA results.	Proposed Transformers do not process joints directly rather operate on features extracted by a CNN, thus the overall model is based on hand-crafted design.
Super-Resolution	TTSR [16] CVPR'20	PSNR/ SSIM	CUFED5 Sun80 Urban100 Manga109	27.1 / 0.8 30.0 / 0.81 25.9 / 0.78 30.1 / 0.91	**a)** Achieves state-of-the-art super-resolution by using attention, **b)** Novel Transformer inspired architectures that can process multi-scale features.	**a)** Proposed Transformer does not process images directly but features extracted by a convolution based network, **b)** Model with large number of trainable parameters, and **c)** Compute intensive.

Multi-Model Learning	ViLBERT [181] NeurIPS'19	Acc./ mAP ($R@1$)	VQA [183]/ Retrieval [239]	70.6/ 58.2	**a)** Proposed Transformer architecture can combine text and visual information to understand inter-task dependencies, **b)** Achieves pre-training on unlabelled dataset.	**a)** Requires large amount of data for pre-training, **b)** Requires fine tuning to the new task.
	Oscar [44] ECCV'20	Acc./ mAP ($R@1$)	VQA [240]/ COCO	80.37/57.5	**a)** Exploit novel supervisory signal via object tags to achieve text and image alignment, **b)** Achieves state-of-the-art results.	Requires extra supervision through pre-trained object detectors thus performance is dependent on the quality of object detectors.
	UNITER [43] ECCV'20	Acc./ Avg. ($R@1/5/10$)	VQA [183]/ Flickr30K [241]	72.47/83.72	Learns fine-grained relation alignment between text and images	Requires large multi-task datasets for Transformer training which lead to high computational cost.
3D Analysis	Point Transformer [230] arXiv'20	Top-1 Acc. IoU	ModelNet40 [232]	92.8 85.9	**a)** Transformer based attention capable to process unordered and unstructured point sets, **b)** Permutation invariant architecture.	**a)** Only moderate improvements over previous SOTA, **b)** Large number of trainable parameters around $6\times$ higher than PointNet++ [242].
	METRO [45] arXiv'20	MPJPE PA-MPJPE MPVE	3DPW [235]	77.1 47.9 88.2	**a)** Does not depend on parametric mesh models so easily extendable to different objects, **b)** Achieves SOTA results using Transformers.	Dependent on hand-crafted network design.

Benchmarking with CNNs

Method	#Param (M)	GFLOPs	Top-1 Acc (%)
ResNet18 [67]⋆	11.7	1.8	69.8
EfficientNet-B3 [87]⋆	12.0	1.8	81.6
DeiT-T [12]	5.7	1.3	72.2
T2T-ViT$_t$-7 [35]	5.0	1.3	71.7
LocalViT-T [107]	5.9	1.3	74.8
CrossViT-T [104]	6.9	1.6	73.4
PVTv1-T [93]	13.2	1.9	75.1
ResT-Lite [110]	10.5	1.4	77.2
CaiT-XXX-24 [243]	12.0	2.5	77.6
PVTv2-B1 [97]	13.1	2.1	78.7
Lv-ViT-T [89]	8.5	–	79.1
RegionViT-T [100]	13.8	2.4	80.4
ResNet50 [67]⋆	25.6	4.1	76.1
ResNeXt50-32x4d [244]⋆	25.0	4.3	77.6
RegNetY-4G [86]⋆	21.0	4.0	80.0
EfficientNet-B4 [87]⋆	19.0	4.2	82.9
DeiT-S [12]	22.1	4.6	79.9
PVTv1-S [93]	24.5	3.8	79.8
LocalViT-S [107]	22.4	4.6	80.8
CrossViT-S [104]	26.7	5.6	81.0
TNT-S [88]	23.8	5.2	81.3
Swin-T [36]	29.0	4.5	81.3
NesT-T [111]	17.0	5.8	81.5
T2T-ViT$_t$-14 [35]	21.5	5.2	81.5
CvT-13 [96]	20.0	4.5	81.6
ResT-B [110]	30.3	4.3	81.6
Twins-SVT-S [37]	24.0	2.8	81.7
PVTv2-B2-Li [97]	22.6	3.9	82.1
RegionViT-S [100]	30.6	5.6	82.5
Lv-ViT-S [89]	26.0	6.6	83.3

Method	#Param (M)	GFLOPs	Top-1 Acc (%)
ResNet101 [67] ⋆	44.7	7.9	77.4
ResNeXt101-32x4d [244]⋆	44.2	8.0	78.8
RegNetY-8G [86]⋆	39.0	8.0	81.7
EfficientNet-B5 [87] ⋆	30.0	9.9	83.6
CvT-21 [96]	32.0	7.1	82.5
CaiT-S-24 [243]	32.2	9.4	82.7
T2T-ViT$_t$-19 [35]	39.0	9.8	81.4
PVTv1-M [93]	44.2	6.7	81.2
PVTv2-B3 [97]	45.2	6.9	83.2
NesT-S [111]	38.0	10.4	83.3
ResNet152 [67] ⋆	60.2	11.6	78.3
CaiT-S-36 [243]	48.0	13.9	83.3
T2T-ViT$_t$-24 [35]	64.0	15.0	82.2
PVTv1-L [93]	61.4	9.8	81.7
TNT-B [88]	66.0	14.1	82.8
Swin-S [36]	50.0	8.7	83.0
Twins-SVT-B [37]	56.0	8.3	83.2
RegionViT-B [100]	72.7	13.0	83.3
PVTv2-B4 [97]	62.6	10.1	83.6
ResNeXt101-64x4d [244] ⋆	83.5	15.6	79.6
RegNetY-16G [86] ⋆	84.0	16.0	82.9
EfficientNet-B6 [87] ⋆	43.0	19.0	84.0
NesT-B [111]	68.0	17.9	83.8
ViT-B/16 [11]	86.6	17.6	79.8
DeiT-B/16 [12]	86.6	17.6	81.8
Swin-B [36]	88.0	15.4	83.3
Twins-SVT-L [37]	99.2	14.8	83.7
PVTv2-B5 [97]	82.0	11.8	83.8
Lv-ViT-M [89]	56.0	16.0	84.1

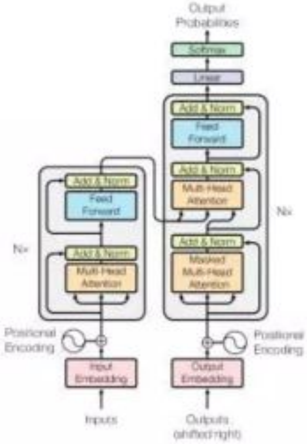
TABLE 3: A Comparative analysis between different vision transformer and CNN models in terms of their parameter complexity and top-1 (%) accuracy on ImageNet validation set. For a direct comparison, we consider models that are trained on ImageNet from scratch on input of size 224x224. ⋆ denotes pure CNN-based methods.

Comparison with Convolutional Neural Networks

- It can be clearly seen that the Transformers based model does clearly better than CNNs , when allowed access to large parameters.
- However, CNNs achieve excellent results even with training based on data volumes that are smaller as those required by Vision Transformers.
- Thus, choosing one architecture over another is not always the wisest choice, and excellent results have been obtained in several Computer Vision tasks through hybrid architectures combining convolutional layers with Vision Transformers.

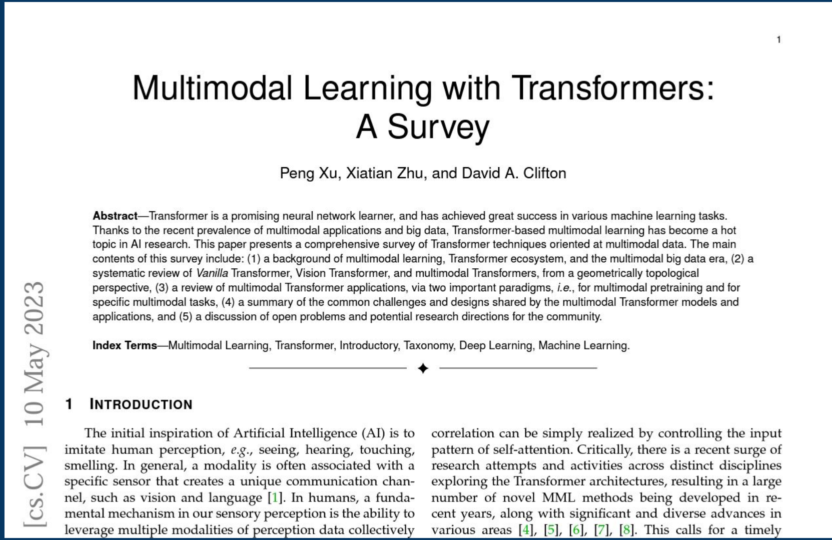
Pathway for future research

- Architecture-level unification across domains



Multi-modal
AI systems

My proposed research/project topic - Multimodal Learning with transformers



Multimodal Learning with Transformers: A Survey

Peng Xu, Xiatian Zhu, and David A. Clifton

Abstract—Transformer is a promising neural network learner, and has achieved great success in various machine learning tasks. Thanks to the recent prevalence of multimodal applications and big data, Transformer-based multimodal learning has become a hot topic in AI research. This paper presents a comprehensive survey of Transformer techniques oriented at multimodal data. The main contents of this survey include: (1) a background of multimodal learning, Transformer ecosystem, and the multimodal big data era, (2) a systematic review of *Vanilla* Transformer, Vision Transformer, and multimodal Transformers, from a geometrically topological perspective, (3) a review of multimodal Transformer applications, via two important paradigms, *i.e.*, for multimodal pretraining and for specific multimodal tasks, (4) a summary of the common challenges and designs shared by the multimodal Transformer models and applications, and (5) a discussion of open problems and potential research directions for the community.

Index Terms—Multimodal Learning, Transformer, Introductory, Taxonomy, Deep Learning, Machine Learning.

[cs.CV] 10 May 2023

1 INTRODUCTION

The initial inspiration of Artificial Intelligence (AI) is to imitate human perception, *e.g.*, seeing, hearing, touching, smelling. In general, a modality is often associated with a specific sensor that creates a unique communication channel, such as vision and language [1]. In humans, a fundamental mechanism in our sensory perception is the ability to leverage multiple modalities of perception data collectively

correlation can be simply realized by controlling the input pattern of self-attention. Critically, there is a recent surge of research attempts and activities across distinct disciplines exploring the Transformer architectures, resulting in a large number of novel MML methods being developed in recent years, along with significant and diverse advances in various areas [4], [5], [6], [7], [8]. This calls for a timely

- Multimodal learning using transformers has gained significant attention in AI research in the recent times.
- With the prevalence of multimodal applications and the availability of big data, researchers have explored the use of transformers for multimodal learning
- The use of transformers in multimodal learning involves techniques such as Vanilla Transformer, Vision Transformer, and multimodal Transformers.
- These models leverage embedding layers to convert different modalities, such as images and text, into visual and text tokens. They also utilize bidirectional blocks with intramodal and intermodal attention to learn holistic representations of multimodal data.

Sources

- Wikipedia
- Publicly available research papers from arxiv.org
- Keynote Talk to International Conference on Research and Development in Science, Engineering and Technology(ICRDSET 3021)
- Davide Coccomini & Nicolo Messina | AICamp 2021
- Public blogs/websites

Thank You

Excited to work as part of SENAI LAB,

ARSHIT MANKODI
ME22B026