Vision and Multi-modal Transformers

Camille Guinaudeau

Vision and Multi-modal Transformers

Camille Guinaudeau[0000−0001−7249−8715]

University Paris-Saclay, CNRS - LISN, 91400, Orsay, France
camille.guinaudeau@lisn.upsaclay.fr

Abstract. Transformers that rely on the self-attention mechanism to capture global dependencies have dominated in natural language modelling and their use in other domains, e.g. speech processing, has shown great potential. The impressive results obtained on these domains leads computer vision researchers to apply transformers to visual data. However, the application of an architecture designed for sequential data is not straightforward for data represented as 2-D matrices. This chapter presents how Transformers were introduced in the domain of vision processing, challenging the historical Convolutional Neural Networks based approaches. After a brief reminder about historical methods in computer vision, namely convolution and self-attention, the chapter focuses on the modifications introduced in the Transformers architecture to deal with the peculiarities of visual data, using two different strategies. In a last part, recent work applying Transformer architecture in a multimodal context is also presented.

Keywords: Convolutional Neural Networks · Vision Transformers · Multimodal Transformer

1 Introduction

Transformers introduced by Vaswani et al. in 2017 [23] are learning models designed to handle sequential data using attention mechanisms. In particular, they allow computers to learn sequences automatically, without having been programmed specifically for this purpose. The Transformer is an instance of the sequence-to-sequence (seq2seq) learning models, taking a sequence as input, processing it, before returning another as output. If the Transformer uses an attention mechanism, it still inherited the encoder-decoder pattern system from the Recurrent Neural Networks (RNNs), the encoder being the input to the sequence and the decoder being the output. Each of these two blocks includes two layers of neural networks: the so-called self-attention layer which allows to keep the interdependence of data in a sequence, and the layer called Feed-forward Neural Network that leads the data to the output.

Transformers, originally designed for Natural Language Processing tasks: translation, question-answering, etc. have been successfully applied in the field of speech processing, particularly in speech synthesis [19], as presented in [9]. The impressive results obtained on other domains leads computer vision researchers

to apply transformers to visual data. However, the application of an architecture designed to be applied to sequential data (such as text or speech) is not straightforward for data represented as 2-D matrix and containing spatial dependencies. This difficulty becoming even greater for video data that add a temporal dimension.

This chapter aims at presenting the adaptation of the Transformers architecture, as it was developed in the Natural Language Processing domain, to visual or multi-modal data to solve vision or multimedia tasks. A particular focus is made on data representation for machine learning approaches, starting from the historical Convolutional Neural Networks (CNN) and self-attention mechanism, described in Section 2 to the Vision Transformers, proposed recently by Dosovitskiy et al. [8]. Finally, in the last Section, recent work applying Transformers in a multi-modal context is presented.

In summary, the learning objectives presented in this paper are the following:

- reminisce the historical approaches for computer vision, namely convolutions and self-attention;
- understand the adaptation of the Transformers architecture to deal with the visual data peculiarities, using two different strategies;
- grasp the functioning principles of recent work applying Transformers architecture to multimodal tasks and data.

2 From Convolutional Neural Networks to Transformers

Computer vision is concerned with the automatic extraction, analysis and understanding of useful information from a single image or a sequence of images. Computer vision tasks include object or event detection, object recognition, indexing, motion estimation, image restoration, etc. The most established algorithm among various deep learning models is Convolutional Neural Network (CNN), a class of artificial neural networks that has been a dominant method in computer vision tasks since the astonishing results reported in 2012 in [14].

2.1 Convolutional Neural Networks (CNN)

In the context of computer vision, a Convolutional Neural Network (CNN) is a Deep Learning algorithm that can take in an input image, identify salient regions in the image, through learnable weights and biases, and is able to differentiate one image from the others. A convolutional neural network architecture is composed of a stack of processing layers: the **convolutional layer** which processes the data of an input channel; the **pooling layer** which reduces the size of the intermediate image, performing some information compression; the correction layer; the **fully connected layer**, which is a perceptron-like layer. After several layers of convolution and max-pooling, the high level decision in the neural network is done through fully connected layers. Finally, the **loss Layer** specifies how the gap between the expected and the actual signal is penalized. It is usually

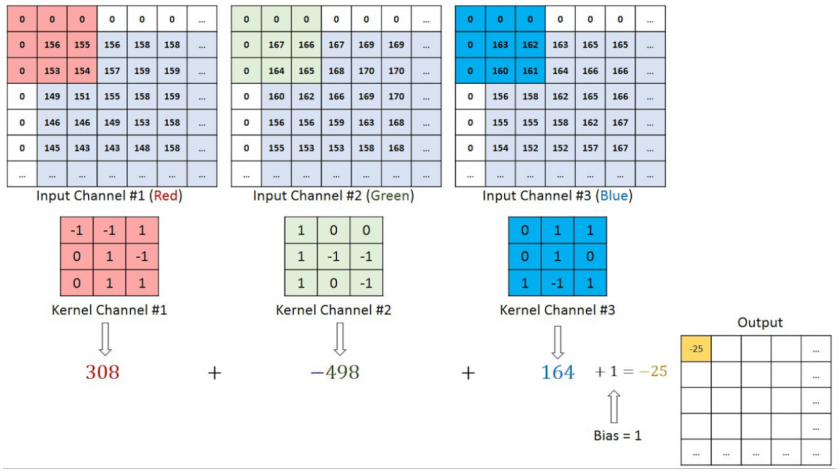

Fig. 1: Convolution of a 3 channel image with a 3x3x3 kernel - Image from Rijul Vohra

the last layer in the network. For more details on the architecture and functioning of Convolutional Neural Network, see [7].

The goal of the convolutional layer is to provide a representation of the input image. Even if an image is just a matrix of pixel values, a simple flattening of the image as a vector would not be enough to represent complex images having pixel dependencies throughout. Therefore, to model the spatial and temporal dependencies in an image, convolutions were introduced to capture them through the application of relevant filters.

Figure 1 presents the process of applying a convolution filter to an image. At each step, a convolution takes a set of weights (a kernel, for 3D structures, or a filter, for 2-D arrays) and multiplies them with the input channels representing the image. Each filter multiplies the weights with different input values; the total inputs are summed, providing a unique value for each filter position. In the case of images with multiple channels (e.g. RGB), all the results are summed with the bias to give a value of the Convoluted Feature Output. To cover the entire image, the filter is applied from right to left and from top to bottom, giving the output matrix.

The advantage of convolutions is two-folds: they can be efficiently parallelized using GPUs and their operations impose two important spatial constraints that facilitate the learning of visual features. Indeed, the features extracted from a convolution layer are 1) not sensitive to the global position of a feature 2) locality sensitive as the operation only takes into account a local region of the image. However, the image representation obtained through convolution operations lack a global view of the image. If they are able to extract visual features, they are not able to model the dependencies between them.

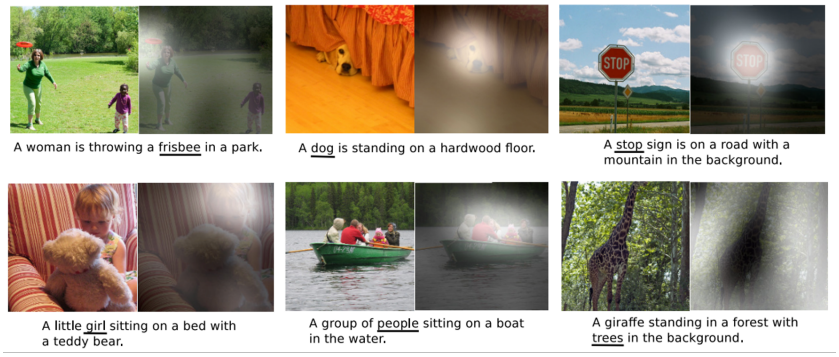

Fig. 2: Show, attend and tell – examples of attending to the correct object - Image extracted from [28]

2.2 Self-attention

To overcome these limitations, recent studies [2, 3, 25, 28] have proposed to use self-attention layers in combination with or instead of convolutional layers. The main difference between convolutional and self-attention layers is that the computed value of a pixel depends on every other pixel of the image, instead of a K×K neighborhood grid.

Introduced by [1] for neural machine translation, attention shows the ability to learn to focus on important subparts of the input, as explained in [6]. This ability can also be used in computer vision where the objective of self-attention layers is to compute attention weights so each position in the image has information about all the other features in the same image. Self-attention layers can either replace or be combined with convolutions, as they are able to model dependencies between spatially distant features by attending to larger receptive fields than regular convolutions.

In [28], the attention mecanism is applied on images to generate captions. The image is first encoded by a convolutional neural network to extract features. To do so, the authors use a lower convolutional layer instead of a fully connected layer to allow the decoder to selectively focus on certain parts of an image by selecting a subset of all the feature vectors. Then a Long short-term memory (LSTM) decoder is used to generate a caption by producing one word at every time step based on a context vector, the previous hidden state and the previously generated words. Figure 2 presents examples of attending to the correct object where white, in the image, indicates the attended regions, and underline, in the text, indicates the corresponding word.

When combined with self-attention, convolution models can improve the results obtained on several vision tasks. For example, in [25], self-attention is used for video classification and object detection with performance that can compete or outperform with state-of-the-art approaches. Another milestone is [3], where the authors obtain improvements on image classification and achieve state-of-the-

art results on video action recognition when using self-attention with convolution models. Finally, [2] augment convolutional operators with self-attention mechanism and show that attention augmentation leads to consistent improvements in image classification and object detection.

However, self-attention layers can have expensive computational costs for high resolution inputs, and can therefore be used only on small spatial dimensions. Some works have already presented ways to overcome this problem, as [24], which computes attention along the two spatial axes sequentially instead of dealing directly with the whole image or [18], which uses patches of feature maps instead of the whole spatial dimensions.

3 Transformers for Computer Vision

Instead of including self-attention within convolutional pipelines, other works have proposed to adapt the original encoder-decoder architecture presented for Transformers to Computer Vision tasks. In this section, works that have proposed to use the transformer architecture to deal with images are described with a focus on the way images are represented. As the original text Transformer takes as input a sequence of words to perform translation, classification, or other NLP tasks, two different strategies can be used to apply this architecture on image data. In the first one, the fewest possible modifications are made to the transformer while input are modified to add information about positions in the image. [8], [21] and [4] are using this strategy. The second strategy consists in modifying the architecture, for example, by introducing convolutions to the vision transformer as proposed in [26] to fit images peculiarities.

3.1 Introduction of positional encodings

The first work that modifies the Transformer design to make it operate directly on images instead of words is the Vision transformers (ViT) proposed by Dosovitskiy et al. [8]. In this paper, the authors make the fewest possible modifications to the Transformer architecture and observe to which extend the model can learn about image structure on its own[1].

Figure 3 presents the Vision Transformers architecture that divides an image into a grid of square patches. Each patch is flattened into a single vector by concatenating the channels of all pixels in a patch and then linearly projecting it to the desired input dimension. To introduce information about the structure of the input elements, the authors add learnable position embeddings to each patch. The idea is to make the Vision Transformers learn relevant information about image structure from the training data and encode structural information in the position embeddings.

To gain some intuition into what the model learns, the authors propose two figures to visualize some of its internal workings. First, they present the position embeddings: the parameters learned by the model to encode the relative

[1] `https://github.com/google-research/vision_transformer`.

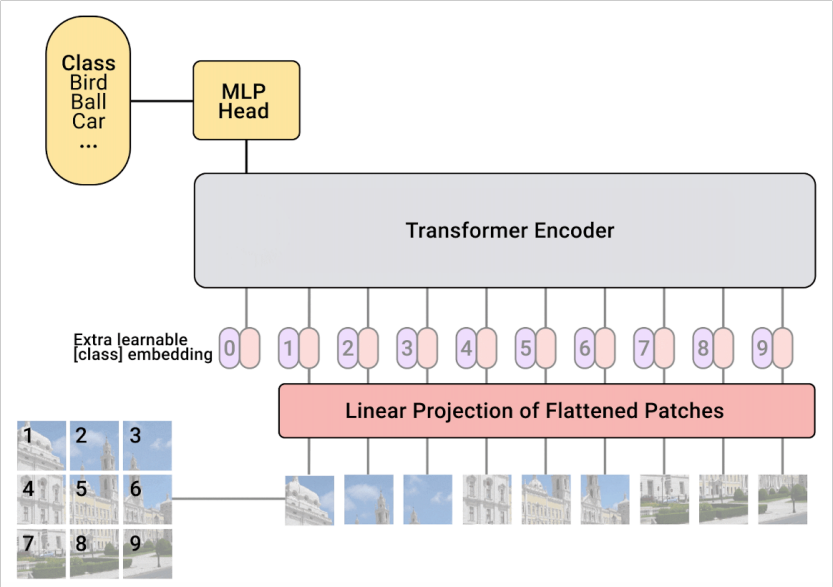

Fig. 3: Vision Transformers (ViT) architecture - Image extracted from [8]

location of patches. On the right part of Figure 4, it can be seen that the vision transformer is able to reproduce the image structure as closer patches tend to have more similar position embeddings. Second, the authors also present the size of attended area by head and network depth in order to evaluate to which extend the network uses the ability to integrate information across the entire image, even in the lowest layers, thanks to self-attention. If for depths between 10 and 20 only large attention distances are visible, meaning that only global features are used, in the lowest layers, a large range in the mean attention distance, showing that the ability to integrate information globally, is indeed used by the model.

Table 1 reports the results obtained for variants of the ViT transformers, compared to previous state-of-the-art models, applied on popular image classification benchmarks. The first comparison model used is Big Transfer (BiT) which performs supervised transfer learning with large ResNets [13] and the second one is Noisy Student [27] which is a large EfficientNet trained using semi-supervised learning on ImageNet and JFT300M with the labels removed. From this table, we can see that the smaller ViT-L model, with a 16x16 image patch size, pre-trained on JFT-300M outperforms BiT-L, which is pre-trained on the same dataset, on all datasets. The larger model, ViT-H (with a 14x14 image patch size), further improves the performance, especially on ImageNet, CIFAR-100, and the VTAB suite, that are the more challenging datasets. The ViT-L model, with a 16x16 image patch size, pre-trained on the public ImageNet-21k dataset performs well on most tasks too. Finally, the last line of the Table reports the

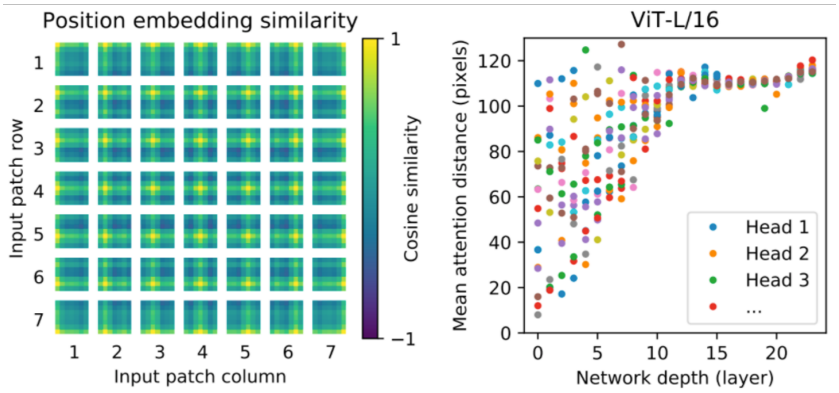

Fig. 4: Position embeddings and size of attended area by head and network depth - Image extracted from [8]

number of TPUv3-core-days taken to pre-train each of the models on TPUv3 hardware, that is, the number of TPU v3 cores (2 per chip) used for training multiplied by the training time in days. From these values, it can be seen that Vision Transformers results use fewer computing resources compared to previous state-of-the-art CNNs.

	ViT-H/14 JFT	ViT-L/16 JFT	ViT-L/16 I21k	BiT-L ResNet152x4	Noisy Student EfficientNet-L2
ImageNet	88.55	87.76	85.30	87.54	88.5
ImageNet ReaL	90.72	90.54	88.62	90.54	90.55
CIFAR-10	99.50	99.42	99.15	99.37	
CIFAR-100	94.55	93.90	93.25	93.51	
Oxford-IIIT Pets	97.56	97.32	94.67	96.62	
Oxford Flowers-102	99.68	99.74	99.61	99.63	
VTAB (19 tasks)	77.63	76.28	72.72	76.29	
TPUv3-core-days	2.5k	0.68k	0.23k	9.9k	12.3k

Table 1: Accuracy for Vision Transformers and state of the art approaches on 7 image classification benchmarks.

If the Vision Transformers proposed in [8] reduces the need of computing resources compared to previous state-of-the-art CNNs, while presenting excellent results when trained with large labelled image dataset they do not generalize well when trained on insufficient amounts of data. Furthermore, the training of these models still involve extensive computing resources.

In order to overcome these problems, Touvron et al. [21] propose competitive convolution-free transformers trained on a single 8-GPU node in two to three days that is competitive with convolutional networks. Their Data-efficient Image Transformers (DeiT) has a similar number of parameters and uses Imagenet as the sole training set[2]. To do so, the authors proposed the knowledge distillation procedure specific for vision transformers. The idea of knowledge distillation is to train one neural network (the student) on an output of another network (the teacher). Such training improves the performance of the vision transformers. The authors have tested the distillation of a transformer student by a CNNs and a transformer teacher and surprisingly, image transformers learn more from CNNs than from another transformer.

In the distillation procedure a new *distillation token* — i.e. a trainable vector, appended to the patch tokens before the first layer — is included in order to interact with the class and patch tokens through the self-attention layers. Similarly to the class token, the objective of the distillation token is to reproduce the label predicted by the teacher, instead of the true label. Both the class and distillation tokens input to the transformers are learned by back-propagation. There are different types of distillation techniques, in [21], the authors use what is called hard-label distillation, so the loss penalizes the student when it misclassifies real target and the target produced by the teacher.

To speed up the training of the system and improve its accuracy, [22] show that it is preferable to use a lower training resolution and fine-tune the network at the larger resolution. As the patch size stays the same when increasing the resolution of an input image, the number of input patches does change and, due to the architecture of transformer blocks and the class token, the model and classifier do not need to be modified to process more tokens. However, it is necessary to adapt the positional embeddings, because there are one for each patch. To do so, [8] and [21] interpolate the positional encoding when changing the resolution with a 2-D interpolation and a bicubic interpolation respectively.

Table 2 reports the accuracy obtained by DeiT models on ImageNet with no external training data, compared with two variants of ViT with 16×16 input patch size. It also presents the number of parameters and the throughput measured for images at resolution 224×224. This table first shows that DeiT models have a lower parameter count than ViT models, and a faster throughput, while having a better accuracy for images at resolution 384x384, when fine-tuned at a larger resolution. It also presents that the transformer-specific distillation increases the accuracy obtained for the three models DeiT-Ti, DeiT-S and DeiT-B.

3.2 Dynamic positional encodings

In the two previous systems, the absolute positional encodings were added to each token in the input sequence to take into account the order of the tokens. If these absolute positional encodings are effective, they also have a negative impact on the flexibility of the Transformers. For example, the encodings are

[2] `https://github.com/facebookresearch/deit`.

Model	Params	Image size	Throughput	Accuracy on ImageNet
ViT-B/16	86M	384^2	85.9	77.9
ViT-L/16	307M	384^2	27.3	76.5
DeiT-Ti	5M	224^2	2536.5	72.2
DeiT-S	22M	224^2	940.4	79.8
DeiT-B	86M	224^2	292.3	81.8
DeiT-B↑384	86M	384^2	85.9	83.1
DeiT-Ti dist	6M	224^2	2529.5	74.5
DeiT-S dist	22M	224^2	936.2	81.2
DeiT-B dist	87M	224^2	290.9	83.4

Table 2: Accuracy and Throughput for Transformers model on ImageNet.

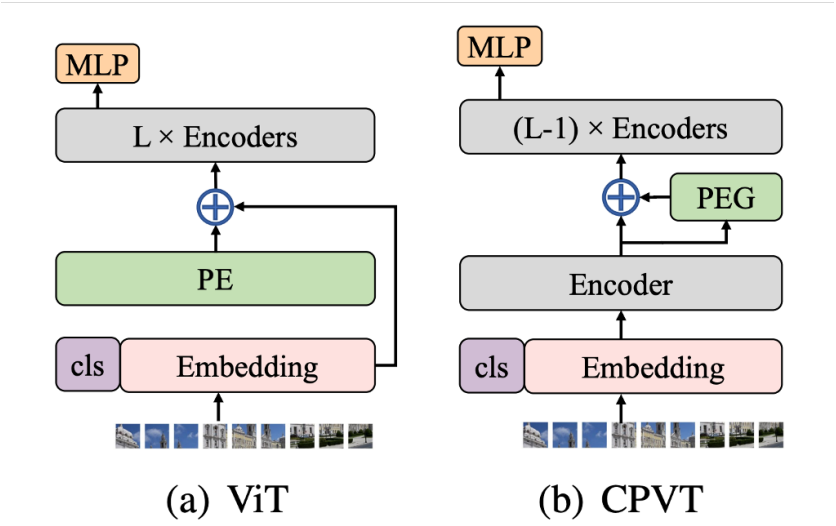

Fig. 5: ViT vs. CPVT architecture - Image extracted from [5]

often a vector of dimension equal to the length of the input sequence to the input sequence, which are jointly updated with the network weights during training, causing difficulty to handle the sequences longer than the ones in the training data at test time. This limits the generalization of the Transformers. Moreover, by adding unique positional encodings to each token (or each image patch), these absolute positional encodings breaks the translation-invariance

To overcome these limitations, Chu et al. propose the Conditional Positional Encoding Vision Transformers (CPVT) architecture, [5], that integrates positional encodings that are dynamically generated and conditioned on the local

neighborhood of an input token[3]. Figure 5 presents the architecture of the Vision Transformers, proposed by [8], with explicit 1-D learnable positional encodings and the CPVT architecture proposed by [5] with conditional positional encoding from the proposed Position Encoding Generator (PEG) plugin. Except that the positional encodings are conditional, the authors exactly follow the Vision Transformers and the Data-efficient Image Transformers architectures to design their vision transformers. To condition the positional encodings on the local neighborhood of an input token, the authors first reshape the flattened input sequence used in the Vision Transformers [8] back in the 2-D image space. Then, a function is repeatedly applied to the local patch in the 2-D structure to produce the conditional positional encodings.

Table 3 presents the accuracy obtained by the Conditional Positional Encoding Vision Transformers (CPVT) [5] and the Data-efficient Image Transformers (DeiT) [21] on ImageNet for two image sizes with direct evaluation on higher resolutions without fine-tuning. From this Table, it can be seen that performance degrades when DeiT models are applied to 384x384 images while CPVT model with the proposed PEG can directly process the larger input images.

Model	Params	Top-1@224	Top-1@384
DeiT-Ti	6M	72.2	71.2
DeiT-S	22M	79.9	78.1
DeiT-B	86M	81.8	79.7
CPVT-Ti	6M	72.4	73.2
CPVT-S	22M	79.9	80.4
CPVT-B	86M	81.9	82.3

Table 3: Accuracy on ImageNet for 224x224 and 384×384 images, with direct evaluation on higher resolutions without fine-tuning.

To picture what the model learns, the authors compare the attention weights of three architectures, presented in Figure 6. In the middle of the figure, the attention weights of DeiT with the original positional encodings, on the right, those of DeiT after the positional encodings are removed and on the left the weights of the CPVT model with PEG. In the middle of the figure, the attention weights are high on the diagonal but low for the rest of the image, suggesting that DeiT with the original positional encodings learns to attend the local neighbors of each patch. When the positional encodings are removed (on the left), all the patches produce similar attention weights meaning that they are not able to attend to the patches in their neighbourhood. Finally, like the original positional encodings, the model with PEG can also learn a similar attention pattern, which indicates that the proposed PEG can provide the position information as well.

[3] `https://github.com/Meituan-AutoML/CPVT`.

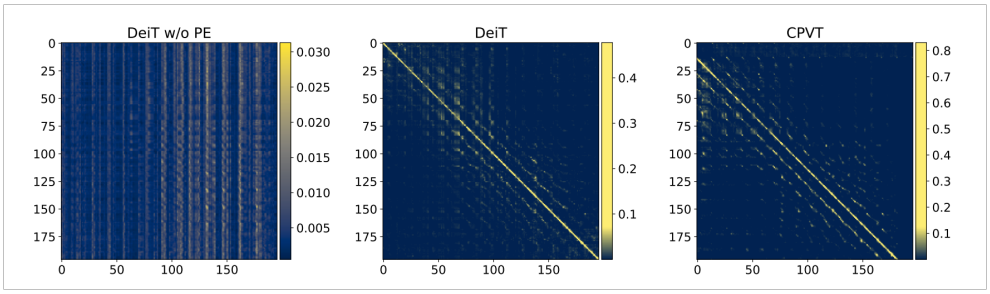

Fig. 6: Normalized attention scores from the second encoder block of DeiT, DeiT without position encoding (DeiT w/o PE), and CPVT on the same input sequence - Image extracted from [5].

3.3 Convolution and transformers

To apply the Transformers, designed for NLP, to vision tasks, the three previous architectures proposed minimal modifications. Despite the success of these models at large scale, their performances are still below similarly sized convolutional neural network (CNN) counterparts when trained on smaller amounts of data. This difference can be explained by the properties of convolutions. As explained in section 2.1, convolutions are able to capture the local structure of an image, and also achieve some degree of shift, scale, and distortion invariance. To account for the properties of convolutions, Wu et al. proposed to introduce two convolution-based operations into the Vision Transformer architecture: Convolutional Token Embedding and Convolutional Projection.

The Convolution vision Transformer (CvT), [26], introduces convolutions to two core sections of the Vision Transformer architecture[4]. First, the authors create a hierarchical structure of Transformers by partitioning the Transformers in multiple stages. The beginning of each stage consists of a convolutional token embedding that performs an overlapping convolution operation with stride on a 2-D-reshaped token map, followed by layer normalization. Hence, the model proposed captures local information and progressively decreases the sequence length while increasing the dimension of token features across stages. This way the model achieves spatial downsampling while increasing the number of feature maps, as is performed in Convolutional Neural Networks. In a second step, the linear projection prior to every self-attention block in the Transformer module is replaced with a convolutional projection that allows the model to further capture local spatial context and reduce semantic ambiguity in the attention mechanism.

Table 4 reports the results of the CvT architecture, compared to the ones obtained by CNN and Vision Transformers on ImageNet. In this table, CvT-X stands for Convolutional vision Transformer with X Transformer Blocks in total. The authors also experiment with a wider model with a larger token dimension for each stage, namely CvT-W24 (W stands for Wide) to validate the

[4] `https://github.com/microsoft/CvT`.

scaling ability of the proposed architecture. On the upper part of the Table, it can be seen that the two convolution-based operations introduced in the Vision Transformer architecture yield improved performance when compared to CNN and to Vision Transformer for images with different resolutions. On the lower part of the table, when the models are pre-trained on ImageNet22k at resolution 224 × 224, and fine-tuned on ImageNet1k at resolution of 384 × 384 (or 480 x 480 for BiT), the accuracy of the CvT models are almost on par with the accuracy of Transformers and CNNs while having a much lower number of model parameters. Finally, when more data is involved, the wide model CvT-W24 pre-trained on ImageNet22k reaches 87.7% of accuracy surpassing the previous best Transformer Vit-L/16.

Method type	Network	Params	image size	ImageNet top-1
Convolutional Networks	ResNet-50 [11]	25	224^2	76.2
	ResNet-101 [11]	45M	224^2	77.4
	ResNet-152 [11]	60M	224^2	78.3
Transformers	ViT-B/16	86M	384^2	77.9
	ViT-L/16	307M	384^2	76.5
	DeiT-S	22M	224^2	79.8
	DeiT-B	86M	224^2	81.8
Convolutional Transformers	CvT-13	20M	224^2	81.6
	CvT-21	32M	224^2	82.5
	CvT-13$\uparrow^{384}$	20M	384^2	83.0
	CvT-21$\uparrow^{384}$	32M	384^2	83.3
Convolutional Networks$_{22k}$	BiT$\uparrow^{480}$ [13]	928M	480^2	85.4
Transformers$_{22k}$	ViT-B/16$\uparrow^{384}$	86M	384^2	84.0
	ViT-L/16$\uparrow^{384}$	307M	384^2	85.2
	ViT-H/16$\uparrow^{384}$	632M	384^2	85.1
Convolutional Transformers$_{22k}$	CvT-13$\uparrow^{384}$	20M	384^2	83.3
	CvT-21$\uparrow^{384}$	32M	384^2	84.9
	CvT-W24$\uparrow^{384}$	277M	384^2	87.7

Table 4: Accuracy of CNN and Vision Transformers architectures on ImageNet. *Subscript*$_{22k}$ indicates that the model is pre-trained on ImageNet22k, and fine-tuned on ImageNet1k with the input size of 384×384 (except for BiT-M that is finetuned with input size of 480×480).

4 Transformers for multimedia data

As the Transformers architecture has been diverted from its primary use in Natural Language Processing to be applied to other modalities (audio and image), very recent works have proposed to use the transformer architecture to solve multimodal challenges. For example, Sterpu et al. have proposed to adapt their tool, AV Align for Speech Recognition to the Transformer architecture [20]. In [16], Radford et al. present the Contrastive Language-Image Pre-training (CLIP) system that is able to learn image representations from scratch on a dataset of 400 million (image, text) pairs collected from the internet. Another example is the work of Gabeur [10] that uses video and text modalities to tackle the tasks of caption-to-video and video-to-caption retrieval (MMT). In the last section of this chapter, two multimodal systems using Transformers architectures are presented, CLIP and MMT.

The idea behind the Contrastive Language-Image Pre-training (CLIP) system proposed by Radford et al. [16] is to learn about images from free-text to recognize objects in a visual scene and solve a variety of visual tasks[5]. So instead of predicting the exact words of the text accompanying each image, the authors try to predict only which text as a whole is paired with which image. Concerning the training part of the system, given a batch of N (image, text) pairs, CLIP is trained to predict which of the $N \times N$ possible (image, text) pairings across a batch actually occurred. To do this, CLIP learns a multi-modal embedding space by jointly training an image encoder and text encoder to maximize the cosine similarity of the image and text embeddings of the N real pairs in the batch while minimizing the cosine similarity of the embeddings of the N^2-N incorrect pairings. CLIP is trained from scratch without initializing the image encoder with ImageNet weights or the text encoder with pre-trained weights. They use only a linear projection to map from each encoder's representation to the multi-modal embedding space.

As CLIP is pre-trained to predict if an image and a text are paired together in its dataset, the authors reuse this capability to perform zero-shot classification of images. For each image dataset, they use the names of all the classes in the dataset as the set of potential text pairings and predict the most probable (image, text) pair according to CLIP. In a bit more details, they first compute the feature embedding of the image and the feature embedding of the set of possible texts with their respective encoders. The cosine similarity of these embeddings is then calculated and normalized into a probability distribution via a softmax operation.

The system was trained on 400M image-text pairs from the internet and evaluated on 27 image datasets that contains different kinds of images: satellite images, car models, medical images, city classification, etc. Figure 7 shows that CLIP is competitive with a fully supervised linear classifier fitted on ResNet-50 features as the CLIP classifier outperforms it on 16 datasets, including ImageNet. On the right of the figure, the number of labeled examples per class a

[5] `https://github.com/openai/CLIP`.

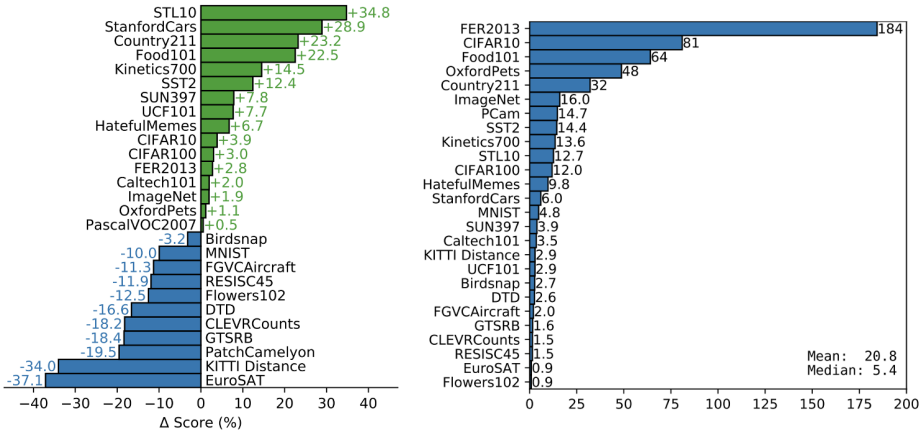

Fig. 7: CLIP performance on 27 image datasets - Image extracted from [16]

linear classifier requires to match the performance of the zero-shot classifier is represented. Performance varies widely from still underperforming a one-shot classifier on two datasets to matching an estimated 184 labeled examples per class.

The second example of multi-modal transformers is the "Multi-modal transformer for video retrieval"[6] (MMT) proposed in [10]. This system tries to solve two tasks. In the first task of caption-to-video retrieval, it is given a query in the form of a caption (e.g., "How to build a house") and its goal is to retrieve the videos best described by it (i.e., videos explaining how to build a house). The other task is video-to-caption retrieval where it has to find among a collection of captions the ones that best describe the query video. To solve these two tasks, the multi-modal transformers use the self-attention mechanism to collect cross-modal and temporal cues about events occurring in a video. The multi-modal transformer is integrated in a cross-modal framework, which takes into account both captions and videos, and estimates their similarity.

The video-level representation computed by the multi-modal transformer (MMT) consists of stacked self-attention layers and fully collected layers. The input is a set of embeddings, all of the same dimension, each of them representing the semantics of a feature, its modality, and the time in the video when the feature was extracted. In order to learn an effective representation from different modalities, the authors use video feature extractors called "experts". Each expert is a model trained for a particular task that is then used to extract features from video. A transformer encoder produces an embedding for each of its feature inputs, resulting in several embeddings for an expert. To obtain a unique embedding for each expert, an aggregated embedding is defined to collect the expert's information. To take into account the cross-modality information, the multi-modal transformer needs to identify which expert it is attending to. To do

[6] `https://github.com/gabeur/mmt`.

so, the authors learn N embeddings to distinguish between embeddings of different experts. Finally the temporal embeddings provide temporal information about the time in the video where each feature was extracted to the multi-modal transformer.

The authors apply their method on three datasets: MSRVTT, ActivityNet and LSMDC. While MSRVTT and LSMDC contain short video-caption pairs (average video duration of 13s for MSRVTT, one-sentence captions), ActivityNet contains much longer videos (several minutes) and each video is captioned with multiple sentences. The authors show that the proposed system obtains state-of-the-art results on all the three datasets.

5 Conclusion

In this chapter, the adaptation of the Transformers architecture, developed for natural language processing tasks, to visual or multimodal data is presented. The chapter mainly focuses on data representation and the necessary modifications to the management of data represented in the form of a 2-D matrix. To this end, some papers proposed to introduce positionnal embeddings, either absolute or dynamically generated, while others integrate convolutions in the Transformers architecture to capture local spatial context. Dealing with video data further complicates the problem as it requires to account for the temporal dimension. These recent applications of the Transformers architecture to these new domains have shown great potential, outperforming previous approaches when apply on purely visual data to reaching state-of-the-art results on multi-modal data.

In order to go further in understanding the issues related to the use of Transformers in computer vision, several additional readings are recommended. Concerning the specificities related to the representation of images, Dong Ping Tian proposes an extensive overview on image feature extraction and representation techniques in Computer Vision [15]. More details about CNNs architecture can be found in [12] to understand both the theory behind CNNs and to gain hands-on experience on the application of CNNs in computer vision. Finally, regarding the internal representation structure of Vision Transformers (ViT) and CNNs, Raghu et al. analyze the differences between the two architectures, how are Vision Transformers solving these tasks ; are they acting like convolutional networks, or learning entirely different visual representations [17]?

References

1. Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. In *Proceedings of the International Conference on Learning Representations*, 2015.
2. Irwan Bello, Barret Zoph, Ashish Vaswani, Jonathon Shlens, and Quoc V Le. Attention augmented convolutional networks. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 3286–3295, 2019.

3. Yunpeng Chen, Yannis Kalantidis, Jianshu Li, Shuicheng Yan, and Jiashi Feng. Aˆ 2-nets: Double attention networks. *Advances in Neural Information Processing Systems*, 31:352–361, 2018.
4. Xiangxiang Chu, Zhi Tian, Bo Zhang, Xinlong Wang, Xiaolin Wei, Huaxia Xia, and Chunhua Shen. Conditional positional encodings for vision transformers. *arXiv preprint arXiv:2102.10882*, 2021.
5. Xiangxiang Chu, Bo Zhang, Zhi Tian, Xiaolin Wei, and Huaxia Xia. Do we really need explicit position encodings for vision transformers? *arXiv e-prints*, pages arXiv–2102, 2021.
6. François Yvon. Transformers in natural language processing. *Advanced course on Human-Centered AI*, 2022.
7. James Crowley. Convolutional neural networks. *Advanced course on Human-Centered AI*, 2022.
8. Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2020.
9. Marc Evrard. Transformers in automatic speech recognition. *Advanced course on Human-Centered AI*, 2022.
10. Valentin Gabeur, Chen Sun, Karteek Alahari, and Cordelia Schmid. Multi-modal transformer for video retrieval. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part IV 16*, pages 214–229. Springer, 2020.
11. Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.
12. Salman Khan, Hossein Rahmani, Syed Afaq Ali Shah, and Mohammed Bennamoun. A guide to convolutional neural networks for computer vision. *Synthesis Lectures on Computer Vision*, 8(1):1–207, 2018.
13. Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Joan Puigcerver, Jessica Yung, Sylvain Gelly, and Neil Houlsby. Big transfer (bit): General visual representation learning. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part V 16*, pages 491–507. Springer, 2020.
14. Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25, 2012.
15. Dong Ping Tian. A review on image feature extraction and representation techniques. *International Journal of Multimedia and Ubiquitous Engineering*, 8(4), 2013.
16. Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. *Image*, 2:T2.
17. Maithra Raghu, Thomas Unterthiner, Simon Kornblith, Chiyuan Zhang, and Alexey Dosovitskiy. Do vision transformers see like convolutional neural networks? In *Advances in Neural Information Processing Systems*, 2021.
18. Prajit Ramachandran, Niki Parmar, Ashish Vaswani, Irwan Bello, Anselm Levskaya, and Jon Shlens. Stand-alone self-attention in vision models. *Advances in Neural Information Processing Systems*, 32, 2019.

19. Yi Ren, Yangjun Ruan, Xu Tan, Tao Qin, Sheng Zhao, Zhou Zhao, and Tie-Yan Liu. Fastspeech: Fast, robust and controllable text to speech. *Advances in Neural Information Processing Systems*, 32, 2019.
20. George Sterpu, Christian Saam, and Naomi Harte. Should we hard-code the recurrence concept or learn it instead? exploring the transformer architecture for audio-visual speech recognition. In *Interspeech*.
21. Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In *International Conference on Machine Learning*, pages 10347–10357. PMLR, 2021.
22. Hugo Touvron, Andrea Vedaldi, Matthijs Douze, and Herve Jegou. Fixing the train-test resolution discrepancy. *Advances in Neural Information Processing Systems*, 32:8252–8262, 2019.
23. Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.
24. Huiyu Wang, Yukun Zhu, Bradley Green, Hartwig Adam, Alan Yuille, and Liang-Chieh Chen. Axial-deeplab: Stand-alone axial-attention for panoptic segmentation. In *European Conference on Computer Vision*, pages 108–126. Springer, 2020.
25. Xiaolong Wang, Ross Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7794–7803, 2018.
26. Haiping Wu, Bin Xiao, Noel Codella, Mengchen Liu, Xiyang Dai, Lu Yuan, and Lei Zhang. Cvt: Introducing convolutions to vision transformers. *arXiv preprint arXiv:2103.15808*, 2021.
27. Qizhe Xie, Minh-Thang Luong, Eduard Hovy, and Quoc V Le. Self-training with noisy student improves imagenet classification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10687–10698, 2020.
28. Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. Show, attend and tell: Neural image caption generation with visual attention. In *International conference on machine learning*, pages 2048–2057. PMLR, 2015.