Title: Vision transformer

URL: https://en.wikipedia.org/wiki/Vision_transformer

PageID: 68212199

Categories: Category:2020 in artificial intelligence, Category:Artificial neural networks, Category:Computer vision, Category:Image processing, Category:Neural network architectures

---

A vision transformer (ViT) is a transformer designed for computer vision . [1] A ViT decomposes an input image into a series of patches (rather than text into tokens), serializes 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.

ViTs were designed as alternatives to convolutional neural networks (CNNs) in computer vision applications. They have different inductive biases, training stability, and data efficiency. [2] Compared to CNNs, ViTs are less data efficient, but have higher capacity. Some of the largest modern computer vision models are ViTs, such as one with 22B parameters. [3] [4]

Subsequent to its publication, many variants were proposed, with hybrid architectures with both features of ViTs and CNNs [5] . ViTs have found application in image recognition , image segmentation , weather prediction , and autonomous driving . [6] [7]

History

Transformers were introduced in Attention Is All You Need (2017), [8] and have found widespread use in natural language processing . A 2019 paper [9] applied ideas from the Transformer to computer vision. Specifically, they started with a ResNet , a standard convolutional neural network used for computer vision, and replaced all convolutional kernels by the self-attention mechanism found in a Transformer. It resulted in superior performance. However, it is not a Vision Transformer.

In 2020, an encoder-only Transformer was adapted for computer vision, yielding the ViT, which reached state of the art in image classification, overcoming the previous dominance of CNN. [1] The masked autoencoder (2022) extended ViT to work with unsupervised training. The vision transformer and the masked autoencoder, in turn, stimulated new developments in convolutional neural networks. [10] [11]

Subsequently, there was cross-fertilization between the previous CNN approach and the ViT approach.

In 2021, some important variants of the Vision Transformers were proposed. These variants are mainly intended to be more efficient, more accurate or better suited to a specific domain. Two studies [12] [13] improved efficiency and robustness of ViT by adding a CNN as a preprocessor. The Swin Transformer [14] achieved state-of-the-art results on some object detection datasets such as COCO , by using convolution-like sliding windows of attention mechanism, and the pyramid process in classical computer vision.

Overview

The basic architecture, used by the original 2020 paper, [1] is as follows. In summary, it is a BERT -like encoder-only Transformer.

The input image is of type R H × W × C {\displaystyle \mathbb {R} ^{H\times W\times C}} , where H , W , C {\displaystyle H,W,C} are height, width, channel (RGB). It is then split into square-shaped patches of type R P × P × C {\displaystyle \mathbb {R} ^{P\times P\times C}} .

For each patch, the patch is pushed through a linear operator, to obtain a vector ("patch embedding"). The position of the patch is also transformed into a vector by "position encoding". The two vectors are added, then pushed through several Transformer encoders.

The attention mechanism in a ViT repeatedly transforms representation vectors of image patches, incorporating more and more semantic relations between image patches in an image. This is analogous to how in natural language processing, as representation vectors flow through a transformer, they incorporate more and more semantic relations between words, from syntax to semantics.

The above architecture turns an image into a sequence of vector representations. To use these for downstream applications, an additional head needs to be trained to interpret them.

For example, to use it for classification, one can add a shallow MLP on top of it that outputs a probability distribution over classes. The original paper uses a linear- GeLU -linear-softmax network. [1]

Variants

Original ViT

The original ViT was an encoder-only Transformer supervise-trained to predict the image label from the patches of the image. As in the case of BERT , it uses a special token in the input side, and the corresponding output vector is used as the only input of the final output MLP head. The special token is an architectural hack to allow the model to compress all information relevant for predicting the image label into one vector.

Transformers found their initial applications in natural language processing tasks, as demonstrated by language models such as BERT and GPT-3 . By contrast the typical image processing system uses a convolutional neural network (CNN). Well-known projects include Xception, ResNet , EfficientNet , [15] DenseNet , [16] and Inception . [17]

Transformers measure the relationships between pairs of input tokens (words in the case of text strings), termed attention . The cost is quadratic in the number of tokens. For images, the basic unit of analysis is the pixel . However, computing relationships for every pixel pair in a typical image is prohibitive in terms of memory and computation. Instead, ViT computes relationships among pixels in various small sections of the image (e.g., 16x16 pixels), at a drastically reduced cost. The sections (with positional embeddings) are placed in a sequence. The embeddings are learnable vectors. Each section is arranged into a linear sequence and multiplied by the embedding matrix. The result, with the position embedding is fed to the transformer. [17]

Architectural improvements

Pooling

After the ViT processes an image, it produces some embedding vectors. These must be converted to a single class probability prediction by some kind of network. In the original ViT and Masked Autoencoder, they used a dummy [CLS] token , in emulation of the BERT language model. The output at [CLS] is the classification token, which is then processed by a LayerNorm -feedforward-softmax module into a probability distribution.

Global average pooling (GAP) does not use the dummy token, but simply takes the average of all output tokens as the classification token. It was mentioned in the original ViT as being equally good. [1]

Multihead attention pooling (MAP) applies a multiheaded attention block to pooling. Specifically, it takes as input a list of vectors x 1 , x 2 , … , x n {\displaystyle x_{1},x_{2},\dots ,x_{n}} , which might be thought of as the output vectors of a layer of a ViT. The output from MAP is M u l t i h e a d e d A t t e n t i o n (Q , V , V) {\displaystyle \mathrm {MultiheadedAttention} (Q,V,V)} , where q {\displaystyle q} is a trainable query vector, and V {\displaystyle V} is the matrix with rows being x 1 , x 2 , … , x n {\displaystyle x_{1},x_{2},\dots ,x_{n}} . [18] This was first proposed in the Set Transformer architecture. [19]

Later papers demonstrated that GAP and MAP both perform better than BERT-like pooling. [18] [20] A variant of MAP was proposed as class attention , which applies MAP, then feedforward, then MAP again. [21]

Re-attention was proposed to allow training deep ViT. It changes the multiheaded attention module. [22]

Masked Autoencoder

The Masked Autoencoder [23] took inspiration from denoising autoencoders and context encoders. [24] It has two ViTs put end-to-end. The first one ("encoder") takes in image patches with positional encoding, and outputs vectors representing each patch. The second one (called "decoder", even though it is still an encoder-only Transformer) takes in vectors with positional encoding and outputs image patches again. During training, both the encoder and the decoder ViTs are used. During inference, only the encoder ViT is used.

During training, each image is cut into patches, and with their positional embeddings added. Of these, only 25% of the patches are selected. The encoder ViT processes the selected patches. No mask tokens are used. Then, mask tokens are added back in, and positional embeddings added again. These are processed by the decoder ViT, which outputs a reconstruction of the full image. The loss is the total mean-squared loss in pixel-space for all masked patches (reconstruction loss is not computed for non-masked patches).

A similar architecture was BERT ViT (BEiT), published concurrently. [25]

DINO

Like the Masked Autoencoder, the DINO (self- di stillation with no labels) method is a way to train a ViT by self-supervision . [26] DINO is a form of teacher-student self-distillation . In DINO, the student is the model itself, and the teacher is an exponential average of the student's past states. The method is similar to previous works like momentum contrast [27] and bootstrap your own latent (BYOL). [28]

The loss function used in DINO is the cross-entropy loss between the output of the teacher network (f θ t ′ {\displaystyle f_{\theta '_{t}}}) and the output of the student network (f θ t {\displaystyle f_{\theta _{t}}}). The teacher network is an exponentially decaying average of the student network's past parameters: θ t ′ = α θ t + α (1 – α) θ t – 1 + ■ {\displaystyle \theta '_{t}=\alpha \theta _{t}+\alpha (1-\alpha)\theta _{t-1}+\cdots } . The inputs to the networks are two different crops of the same image, represented as T (x) {\displaystyle T(x)} and T ′ (x) {\displaystyle T'(x)} , where x {\displaystyle x} is the original image. The loss function is written as L (f θ t ′ (T (x)) , f θ t (T ′ (x))) {\displaystyle L(f_{\theta '_{t}}(T(x)),f_{\theta _{t}}(T'(x)))} One issue is that the network can "collapse" by always outputting the same value (y {\displaystyle y}), regardless of the input. To prevent this collapse, DINO employs two strategies:

Sharpening : The teacher network's output is sharpened using a softmax function with a lower temperature. This makes the teacher more "confident" in its predictions, forcing the student to learn more meaningful representations to match the teacher's sharpened output.

Centering : The teacher network's output is centered by averaging it with its previous outputs. This prevents the teacher from becoming biased towards any particular output value, encouraging the student to learn a more diverse set of features.

In January 2024, Meta AI Research released an updated version called DINOv2 [29] with improvements in architecture, loss function, and optimization technique. It was trained on a larger and more diverse dataset. The features learned by DINOv2 were more transferable , meaning it had better performance in downstream tasks.

In August 2025, Meta AI Research released DINOv3, an update to DINOv2. It introduced image-text alignment like CLIP . It scaled up the model to 7B parameters and the training dataset to 1.7B images (obtained by diversity-sampling an initial dataset with 17B images). Architecturally, it introduced two improvements: Gram anchoring and axial RoPE (Rotary Positional Embeddings) with jittering. Gram anchoring applies teacher-student self-distillation for the Gram matrix between the feature vectors of the patches of an image. It avoids the previously observed problem of degradation of dense feature maps: While performance on global tasks (like classification) continued to improve, performance on dense tasks (like segmentation) would peak early and then decline, with feature maps becoming noisy. Axial RoPE makes the model more robust to varying

image resolutions, scales, and aspect ratios. [30] [31]

Swin Transformer

The Swin Transformer (" S hifted win dows") [14] took inspiration from standard CNNs:

Instead of performing self-attention over the entire sequence of tokens, one for each patch, it performs "shifted window based" self-attention, which means only performing attention over square-shaped blocks of patches. One block of patches is analogous to the receptive field of one convolution.

After every few attention blocks, there is a "merge layer", which merges neighboring 2x2 tokens into a single token. This is analogous to pooling (by 2x2 convolution kernels, with stride 2). Merging means concatenation followed by multiplication with a matrix.

It is improved by Swin Transformer V2, [32] which modifies upon the ViT by a different attention mechanism [14] : Figure 1 :

LayerNorm immediately after each attention and feedforward layer ("res-post-norm");

scaled cosine attention to replace the original dot product attention;

log-spaced continuous relative position bias , which allows transfer learning across different window resolutions.

TimeSformer

The TimeSformer [33] was designed for video understanding tasks, and it applied a factorized self-attention, similar to the factorized convolution kernels found in the Inception CNN architecture. [34] Schematically, it divides a video into frames, and each frame into a square grid of patches (same as ViT). Let each patch coordinate be denoted by x , y , t {\displaystyle x,y,t} , denoting horizontal, vertical, and time.

A space attention layer is a self-attention layer where each query patch q x , y , t {\displaystyle q_{x,y,t}} attends to only the key and value patches k x ′ , y ′ , t ′ , v x ′ , y ′ , t ′ {\displaystyle k_{x',y',t'},v_{x',y',t'}} such that t = t ′ {\displaystyle t=t'} .

A time attention layer is where the requirement is x ′ = x , y ′ = y {\displaystyle x'=x,y'=y} instead.

The TimeSformer also considered other attention layer designs, such as the "height attention layer" where the requirement is x ′ = x , t ′ = t {\displaystyle x'=x,t'=t} . However, they found empirically that the best design interleaves one space attention layer and one time attention layer.

ViT-VQGAN

In ViT-VQGAN , [35] there are two ViT encoders and a discriminator. One encodes 8x8 patches of an image into a list of vectors, one for each patch. The vectors can only come from a discrete set of "codebook", as in vector quantization . Another encodes the quantized vectors back to image patches. The training objective attempts to make the reconstruction image (the output image) faithful to the input image. The discriminator (usually a convolutional network, but other networks are allowed) attempts to decide if an image is an original real image, or a reconstructed image by the ViT.

The idea is essentially the same as vector quantized variational autoencoder (VQVAE) plus generative adversarial network (GAN).

After such a ViT-VQGAN is trained, it can be used to code an arbitrary image into a list of symbols, and code an arbitrary list of symbols into an image. The list of symbols can be used to train into a standard autoregressive transformer (like GPT), for autoregressively generating an image. Further, one can take a list of caption-image pairs, convert the images into strings of symbols, and train a standard GPT-style transformer. Then at test time, one can just give an image caption, and have it autoregressively generate the image. This is the structure of Google Parti. [36]

Others

Other examples include the visual transformer, [37] CoAtNet, [38] CvT, [39] the data-efficient ViT (DeiT), [40] etc.

In the Transformer in Transformer architecture, each layer applies a vision Transformer layer on each image patch embedding, add back the resulting tokens to the embedding, then applies another vision Transformer layer. [41]

Comparison with CNNs

Typically, ViT uses patch sizes larger than standard CNN kernels (3x3 to 7x7). ViT is more sensitive to the choice of the optimizer, hyperparameters , and network depth. Preprocessing with a layer of smaller-size, overlapping (stride < size) convolutional filters helps with performance and stability. [13]

This different behavior seems to derive from the different inductive biases they possess.

CNN applies the same set of filters for processing the entire image. This allows them to be more data efficient and less sensitive to local perturbations. [2] ViT applies self-attention, allowing them to easily capture long-range relationships between patches [42] . They also require more data to train, but they can ingest more training data compared to CNN, which might not improve after training on a large enough training dataset. ViT also appears more robust to input image distortions such as adversarial patches or permutations. [43]

Applications

ViT have been used in many Computer Vision tasks with excellent results and in some cases even state-of-the-art. Image Classification , Object Detection , Video Deepfake Detection , [44] Image segmentation , [45] Anomaly detection , Image Synthesis , Cluster analysis , Autonomous Driving . [6] [7]

ViT had been used for image generation as backbones for GAN [46] and for diffusion models (diffusion transformer, or DiT). [47]

DINO [26] has been demonstrated to learn useful representations for clustering images and exploring morphological profiles on biological datasets, such as images generated with the Cell Painting assay. [48]

In 2024, a 113 billion-parameter ViT model was proposed (the largest ViT to date) for weather and climate prediction , and trained on the Frontier supercomputer with a throughput of 1.6 exaFLOPs . [49]

See also

Transformer (machine learning model)

Convolutional neural network

Attention (machine learning)

Perceiver

Deep learning

PyTorch

TensorFlow

References

Further reading

Zhang, Aston; Lipton, Zachary; Li, Mu; Smola, Alexander J. (2024). "11.8. Transformers for Vision" . Dive into deep learning . Cambridge New York Port Melbourne New Delhi Singapore: Cambridge University Press. ISBN 978-1-009-38943-3 .

Steiner, Andreas; Kolesnikov, Alexander; Zhai, Xiaohua; Wightman, Ross; Uszkoreit, Jakob; Beyer, Lucas (June 18, 2021). "How to train your ViT? Data, Augmentation, and Regularization in Vision Transformers". arXiv : 2106.10270 [cs.CV].

v

t

e

History timeline

timeline

Companies

Projects

Parameter Hyperparameter

Hyperparameter

Loss functions

Regression Bias–variance tradeoff Double descent Overfitting

Bias–variance tradeoff

Double descent

Overfitting

Clustering

Gradient descent SGD Quasi-Newton method Conjugate gradient method

SGD

Quasi-Newton method

Conjugate gradient method

Backpropagation

Attention

Convolution

Normalization Batchnorm

Batchnorm

Activation Softmax Sigmoid Rectifier

Softmax

Sigmoid

Rectifier

Gating

Weight initialization

Regularization

Datasets Augmentation

Augmentation

Prompt engineering

Reinforcement learning Q-learning SARSA Imitation Policy gradient

Q-learning

SARSA

Imitation

Policy gradient

Diffusion

Latent diffusion model

Autoregression

Adversary

RAG

Uncanny valley

RLHF

Self-supervised learning

Reflection

Recursive self-improvement

Hallucination

Word embedding

Vibe coding

Machine learning In-context learning

In-context learning

Artificial neural network Deep learning

Deep learning

Language model Large language model NMT

Large language model

NMT

Reasoning language model

Model Context Protocol

Intelligent agent

Artificial human companion

Humanity's Last Exam

Artificial general intelligence (AGI)

AlexNet

WaveNet

Human image synthesis

HWR

OCR

Computer vision

Speech synthesis 15.ai ElevenLabs

15.ai

ElevenLabs

Speech recognition Whisper

Whisper

Facial recognition

AlphaFold

Text-to-image models Aurora DALL-E Firefly Flux Ideogram Imagen Midjourney Recraft Stable Diffusion

Aurora

DALL-E

Firefly

Flux

Ideogram

Imagen

Midjourney

Recraft

Stable Diffusion

Text-to-video models Dream Machine Runway Gen Hailuo AI Kling Sora Veo

Dream Machine

Runway Gen

Hailuo AI

Kling

Sora

Veo

Music generation Riffusion Suno AI Udio

Riffusion

Suno AI

Udio

Word2vec

Seq2seq

GloVe

BERT

T5

Llama

Chinchilla AI

PaLM

GPT 1 2 3 J ChatGPT 4 4o o1 o3 4.5 4.1 o4-mini 5

1

2

3

J

ChatGPT

4

4o

o1

o3

4.5

4.1

o4-mini

5

Claude

Gemini Gemini (language model) Gemma

Gemini (language model)

Gemma

Grok

LaMDA

BLOOM

DBRX

Project Debater

IBM Watson

IBM Watsonx

Granite

PanGu-Σ

DeepSeek

Qwen

AlphaGo

AlphaZero

OpenAI Five

Self-driving car

MuZero

Action selection AutoGPT

AutoGPT

Robot control

Alan Turing

Warren Sturgis McCulloch

Walter Pitts

John von Neumann

Claude Shannon

Shun'ichi Amari

Kunihiko Fukushima

Takeo Kanade
Marvin Minsky
John McCarthy
Nathaniel Rochester
Allen Newell
Cliff Shaw
Herbert A. Simon
Oliver Selfridge
Frank Rosenblatt
Bernard Widrow
Joseph Weizenbaum
Seymour Papert
Seppo Linnainmaa
Paul Werbos
Geoffrey Hinton
John Hopfield
Jürgen Schmidhuber
Yann LeCun
Yoshua Bengio
Lotfi A. Zadeh
Stephen Grossberg
Alex Graves
James Goodnight
Andrew Ng
Fei-Fei Li
Alex Krizhevsky
Ilya Sutskever
Oriol Vinyals
Quoc V. Le
Ian Goodfellow
Demis Hassabis
David Silver
Andrej Karpathy
Ashish Vaswani
Noam Shazeer
Aidan Gomez
John Schulman
Mustafa Suleyman

Jan Leike

Daniel Kokotajlo

François Chollet

Neural Turing machine

Differentiable neural computer

Transformer Vision transformer (ViT)

Vision transformer (ViT)

Recurrent neural network (RNN)

Long short-term memory (LSTM)

Gated recurrent unit (GRU)

Echo state network

Multilayer perceptron (MLP)

Convolutional neural network (CNN)

Residual neural network (RNN)

Highway network

Mamba

Autoencoder

Variational autoencoder (VAE)

Generative adversarial network (GAN)

Graph neural network (GNN)

Category