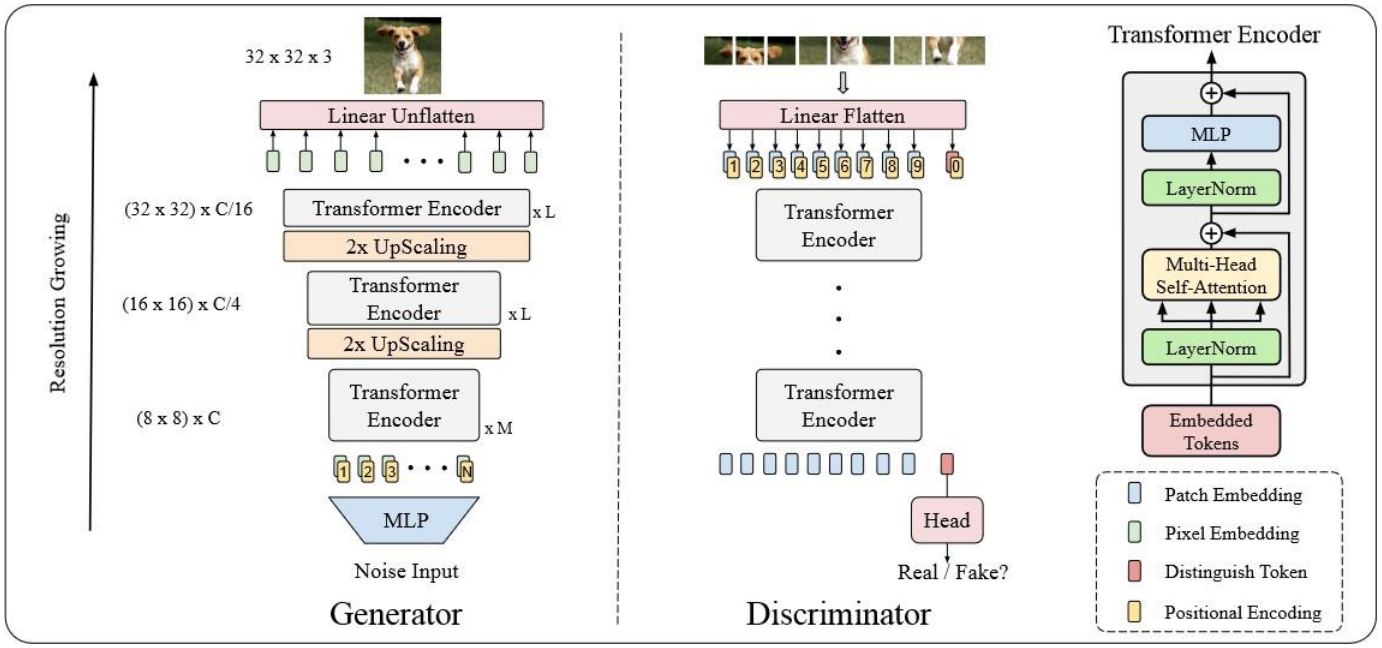
TransGAN: Two Transformers Can Make One Strong GAN

Yifan Jiang[1] Shiyu Chang[2] Zhangyang Wang[1]

CVPR 2021



Outline

- Introduction
- Background and related work
- Methods
- Experiments & Results
- Reproduce
- Conclusion

Introduction

- In the computer vision domain, nearly every successful GAN relies on CNN-based generators and discriminators.
- Convolutions, with the strong inductive bias for natural images, crucially contribute to the appealing visual results and rich diversity achieved by modern GANs.
- Fundamentally, a convolution operator has a local receptive field, and hence CNNs cannot process long-range dependencies unless passing through a sufficient number of layers.
- However, that could cause the loss of feature resolution and fine details, in addition to the difficulty of optimization.

Can We Build a Strong GAN Completely Free of Convolutions?

- Not only arising from intellectual curiosity, but also of practical relevance.
- Inspired by the emerging trend of using Transformer architectures for computer vision tasks (Carion et al., 2020;Zeng et al., 2020; Dosovitskiy et al., 2020).
- Transformers (Vaswani et al., 2017; Devlin et al., 2018) have prevailed in natural language processing (NLP), and lately, start to perform comparably or even better than their CNN competitors in a variety of vision benchmarks.

Carion et al., 2020; End-to-End object detection with transformers
Zeng et al., 2020; Learning joint spatial-temporal transformations for video inpaiting
Dosovitsky et al., 2020; An image is worth 16x16 words: Transformers for image recognition at scale
Vaswani et al., 2017; Attention is all you need
Devlin et al., 2018; Bert: Pre-training of deep bidirectional transformers for language understanding

Can We Build a Strong GAN Completely Free of Convolutions?

- The charm of the transformer to computer vision lies in at least two-fold:

  I. it has strong representation capability and is free of human-defined inductive bias. In comparison, CNNs exhibit a strong bias towards feature locality, as well as spatial invariance due to sharing filter weights across all locations;

  II. the transformer architecture is general, conceptually simple, and has the potential to become a powerful “universal” model across tasks and domains (Dosovitskiy et al., 2020).It can get rid of many ad-hoc building blocks commonly seen in CNN-based pipelines (Carion et al., 2020).

Carion et al., 2020; End-to-End object detection with transformers
Dosovitsky et al., 2020; An image is worth 16x16 words: Transformers for image recognition at scale

Background and related work

GANs

Generative Adversarial Nets

Ian J. Goodfellow, Jean Pouget-Abadie[*], Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair[†], Aaron Courville, Yoshua Bengio[‡]
Département d'informatique et de recherche opérationnelle
Université de Montréal
Montréal, QC H3C 3J7

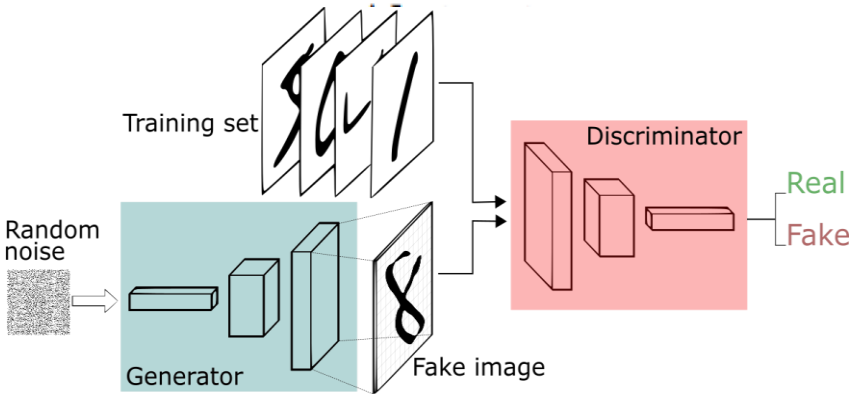


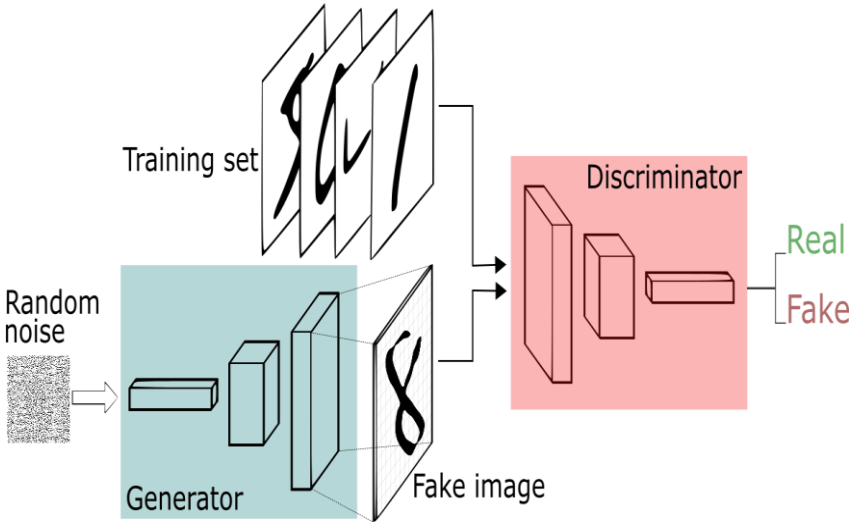
Figure from link

Background and related work

GANs

Loss Function

$$\min_{G} \max_{D} V(D,G) = \mathbb{E}_{x\sim p_{data}(x)}[\log D(x)] + \mathbb{E}_{z\sim p_z(z)}\left[\log\left(1 - D(G(z))\right)\right]$$



Background and related work

GANs

- The original GAN used fully-connected networks and can only generate small images.

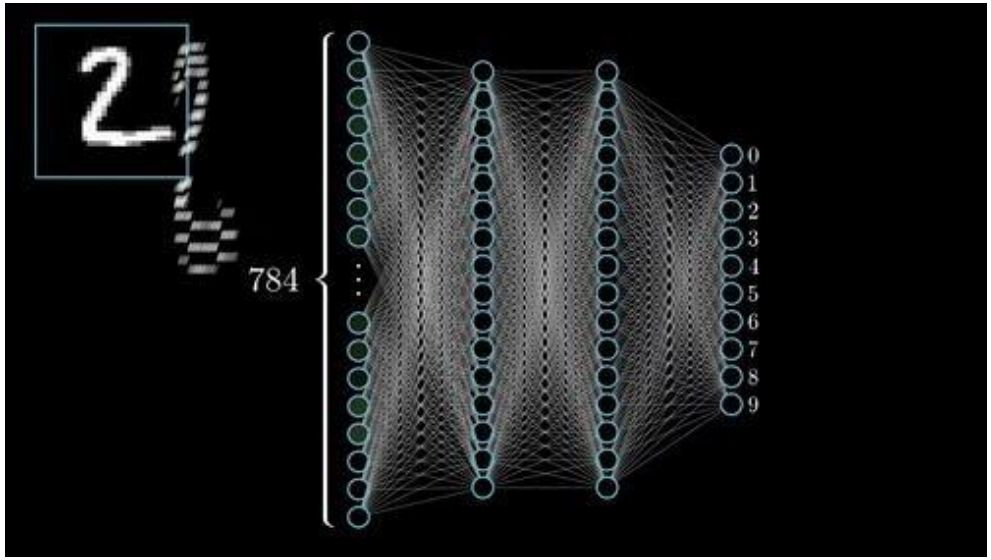


Figure from link

Background and related work

GANs

- DCGAN (Radford et al., 2015) was the first to scale up GANs using CNN architectures, which allowed for stable training for higher resolution and deeper generative models.

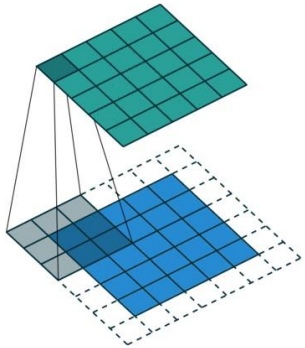


Figure from link

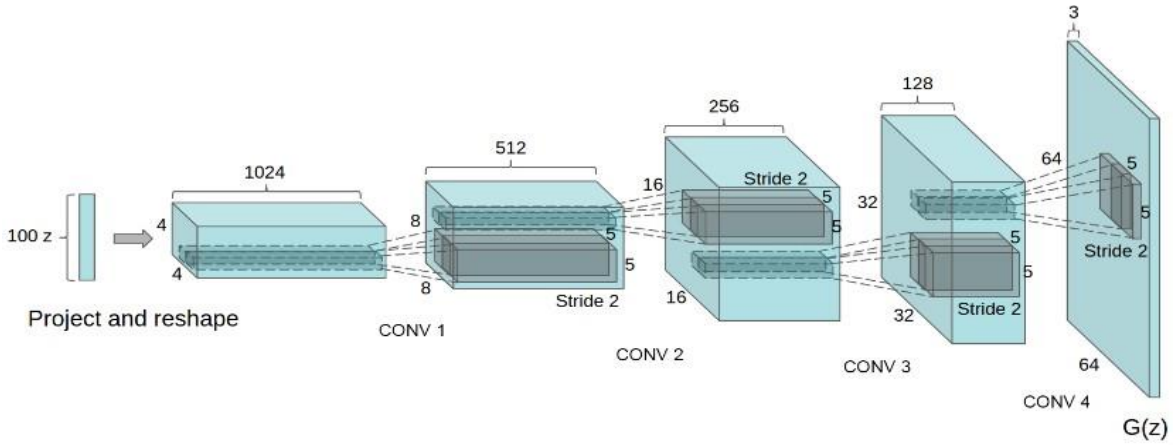


Figure 1: DCGAN generator used for LSUN scene modeling. A 100 dimensional uniform distribution Z is projected to a small spatial extent convolutional representation with many feature maps. A series of four fractionally-strided convolutions (in some recent papers, these are wrongly called deconvolutions) then convert this high level representation into a 64×64 pixel image. Notably, no fully connected or pooling layers are used.

Figure from link

Background and related work

Transformers

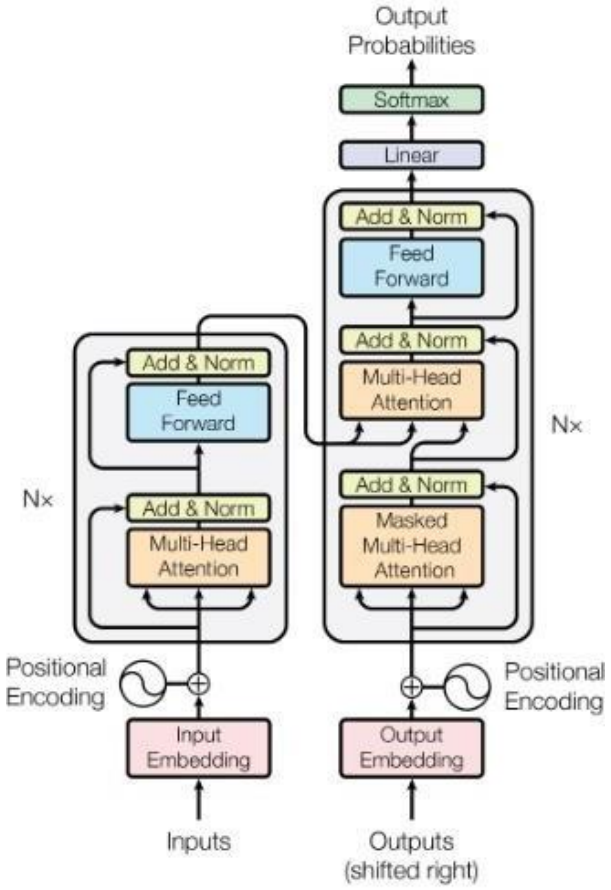


Figure 1: The Transformer - model architecture.

Vaswani et al., 2017; Attention is All You Need

Figure from [link]

Background and related work

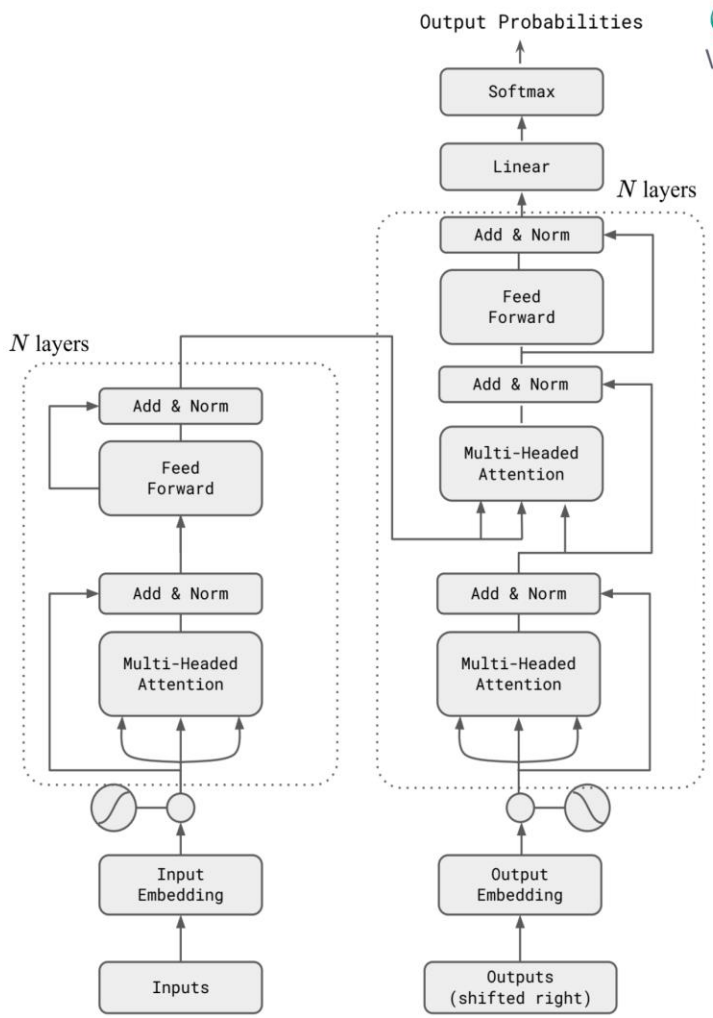
Transformers

PROTON X

Kiến trúc
Architecture

https://arxiv.org/pdf/1706.03762.pdf

1. Lược bỏ kiến trúc Seq2Seq
2. Sử dụng Attention làm lõi chính của model
3. Loss Function vẫn sử dụng Cross Entropy

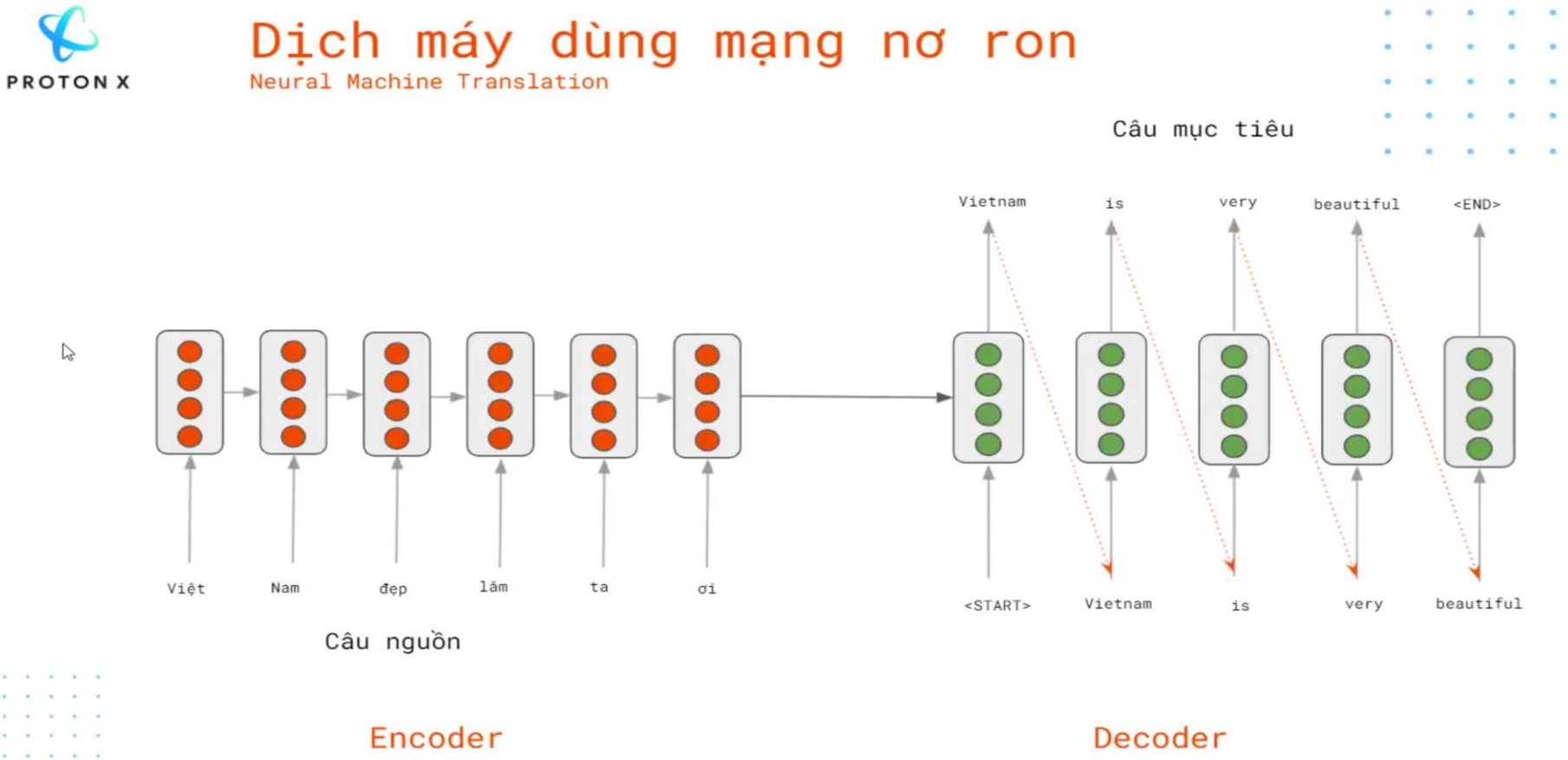


VietAI

TensorFlow Vietnam

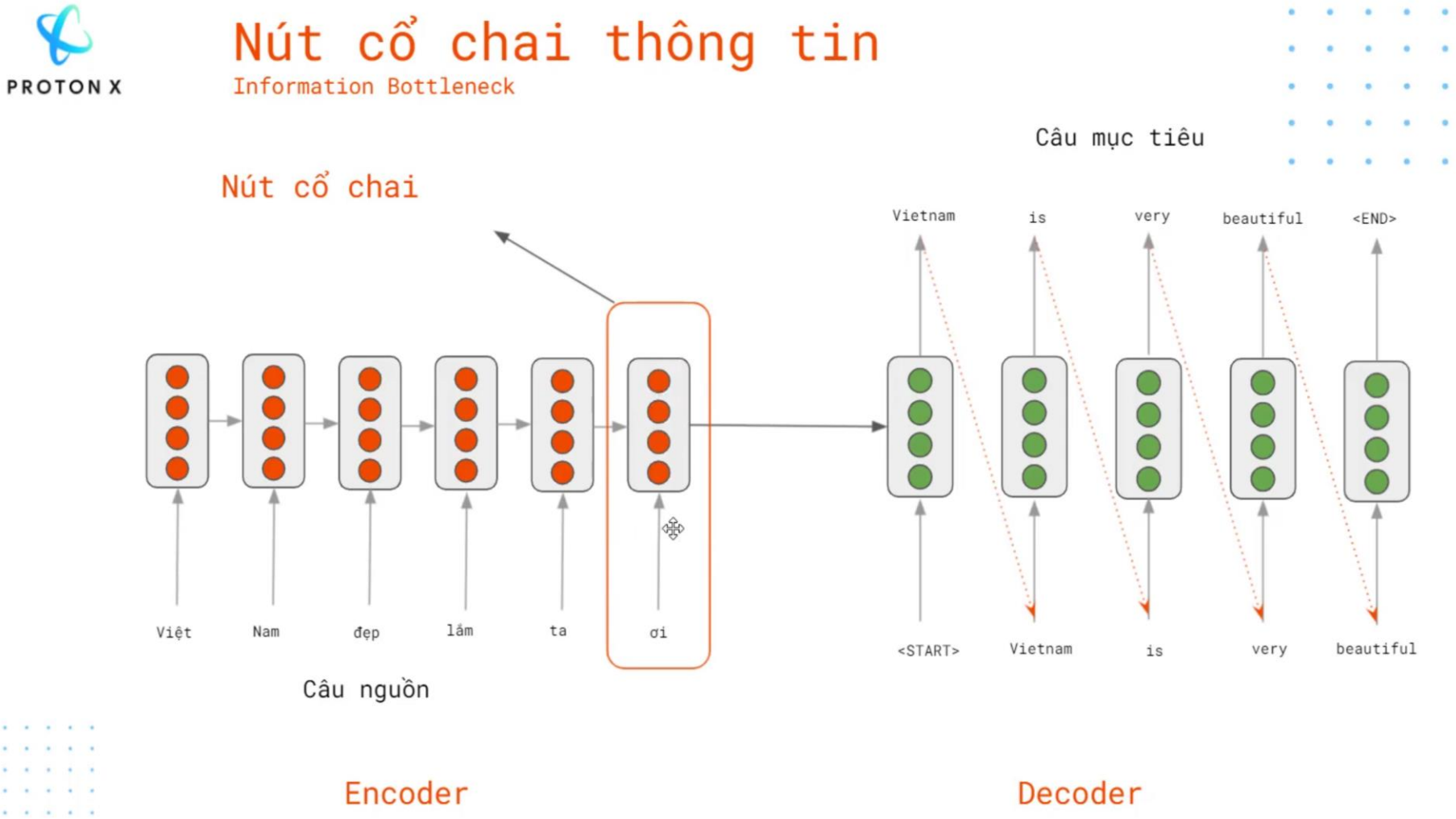
Background and related work

Transformers



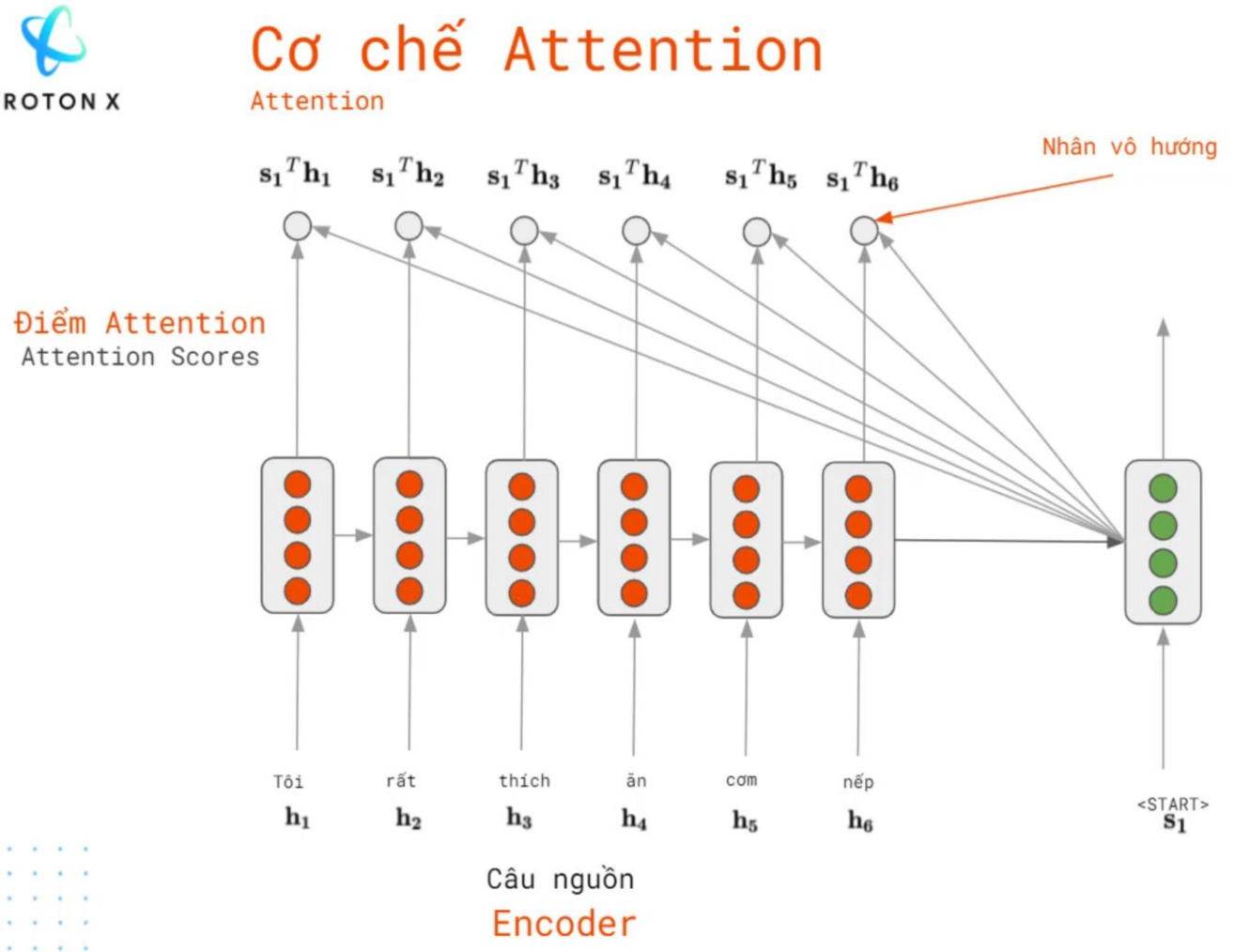
Background and related work

Transformers



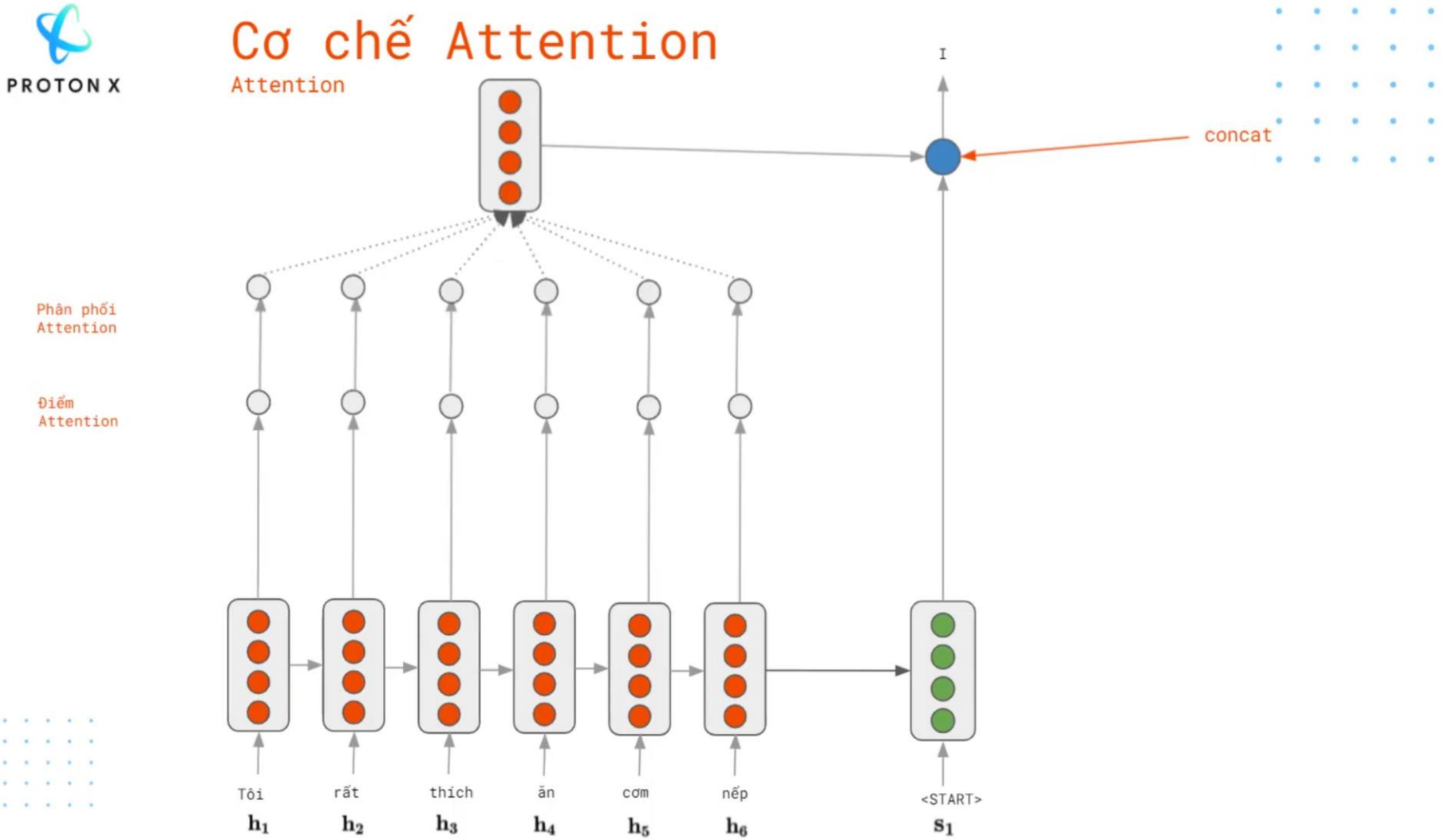
Background and related work

Transformers



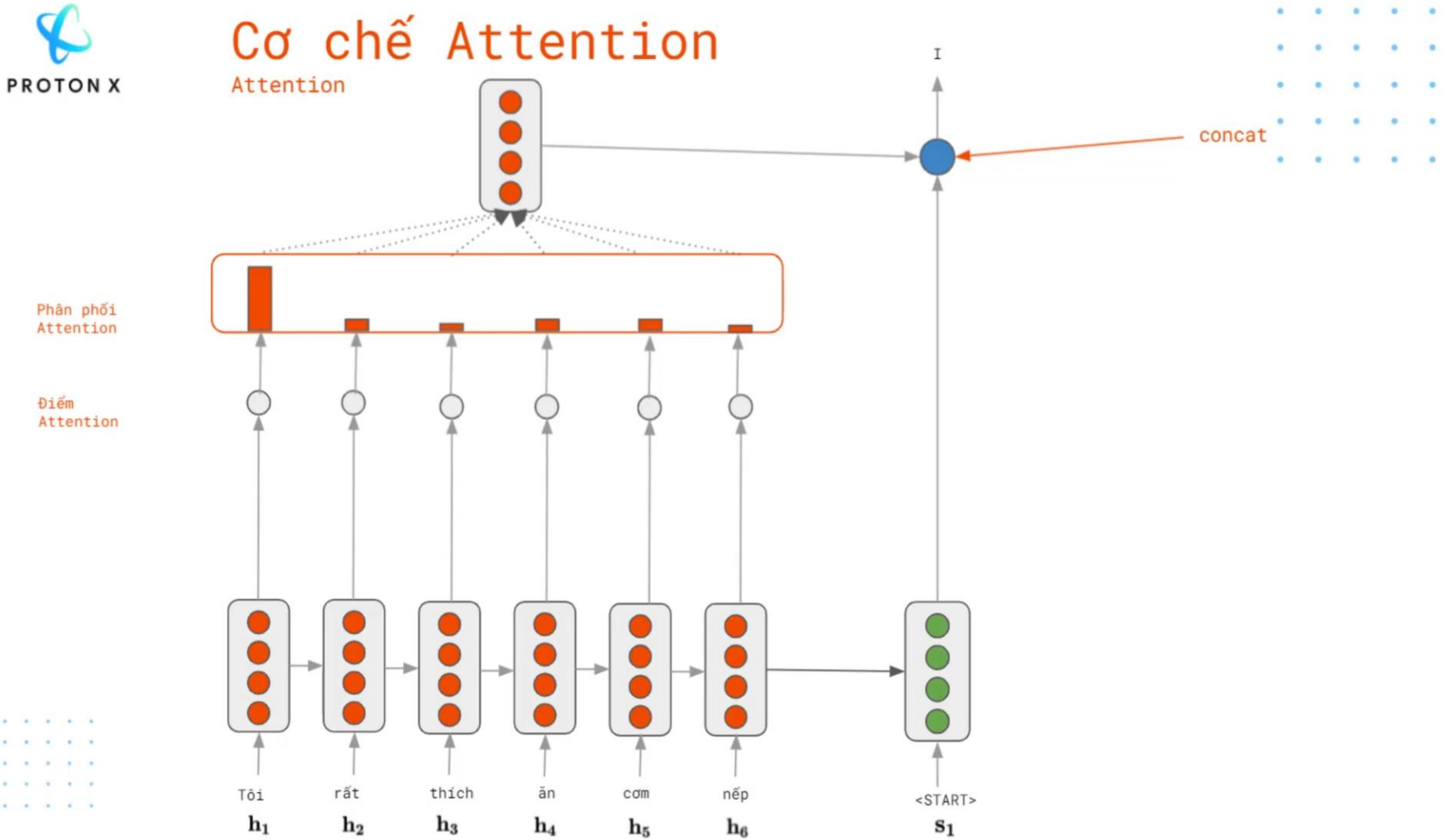
Background and related work

Transformers



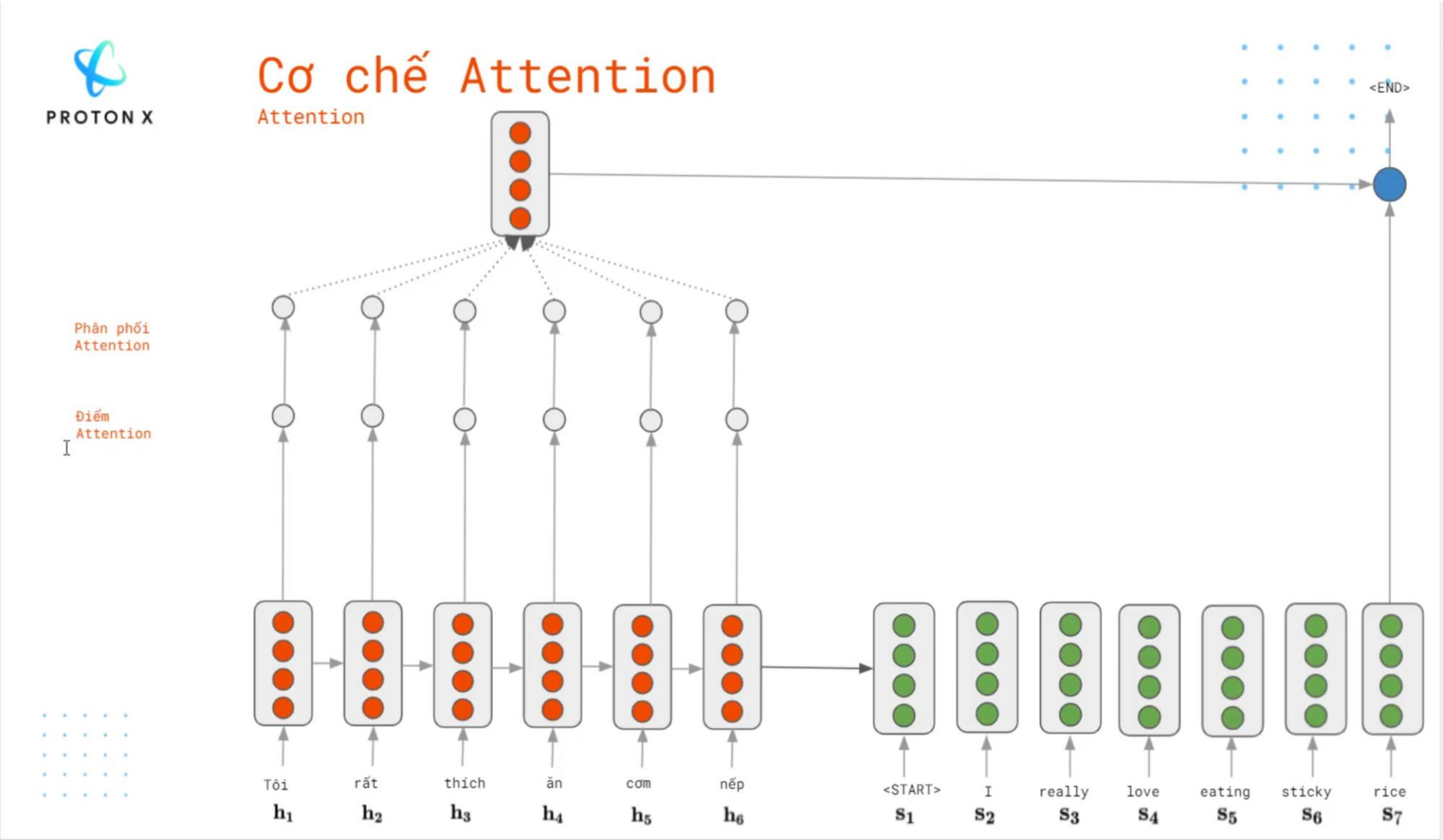
Background and related work

Transformers



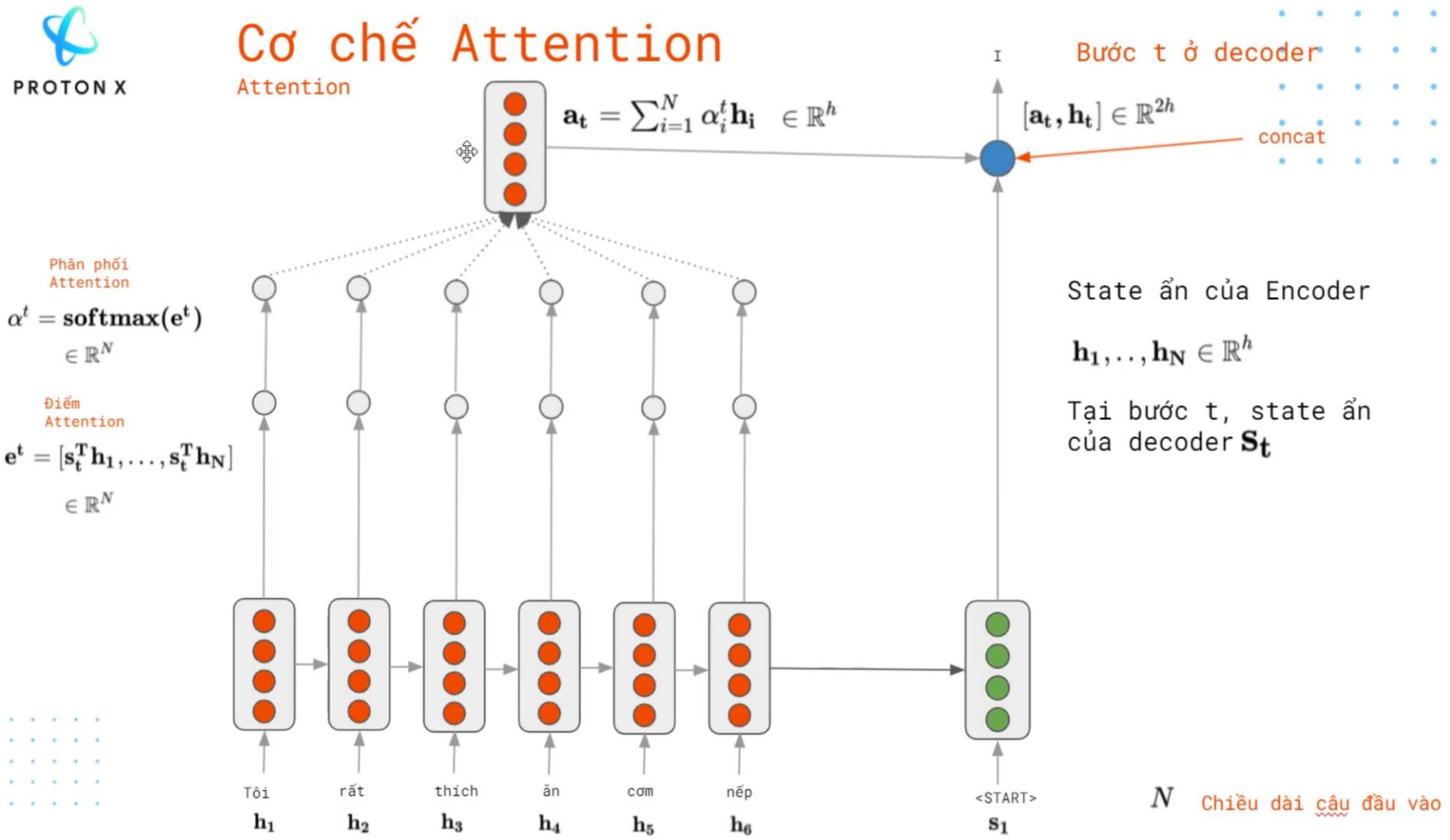
Background and related work

Transformers



Background and related work

Transformers



Background and related work

Transformers

PROTON X

Encoder

Encoder

Nhiệm vụ của Encoder

- Học mối tương quan giữa mỗi từ trong câu với các từ còn lại (self Attention)
- Bổ sung mối quan hệ này vào Embedding của từng từ đầu.

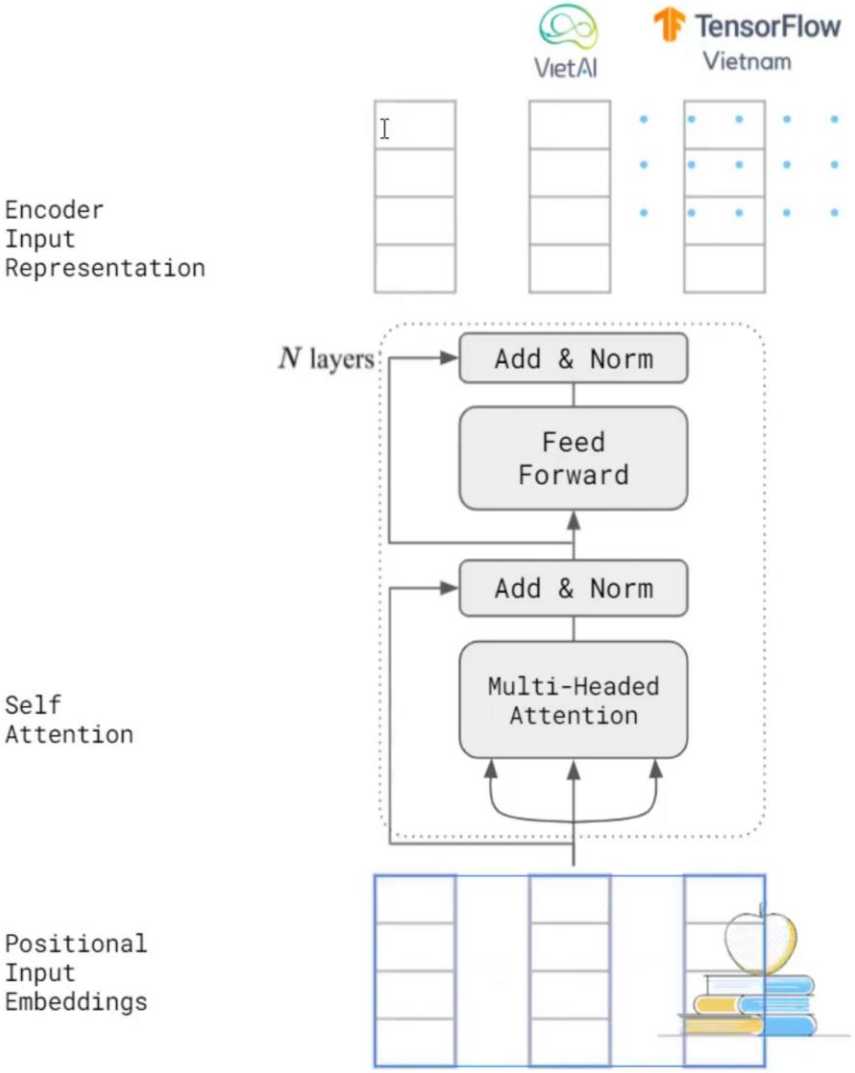
Encoder bao gồm

6 layer giống nhau, mỗi Layer bao gồm 2 phần nhỏ

- Multi-Headed Attention
- Position-wise fully connected feed-forward network

VietAI

TensorFlow Vietnam



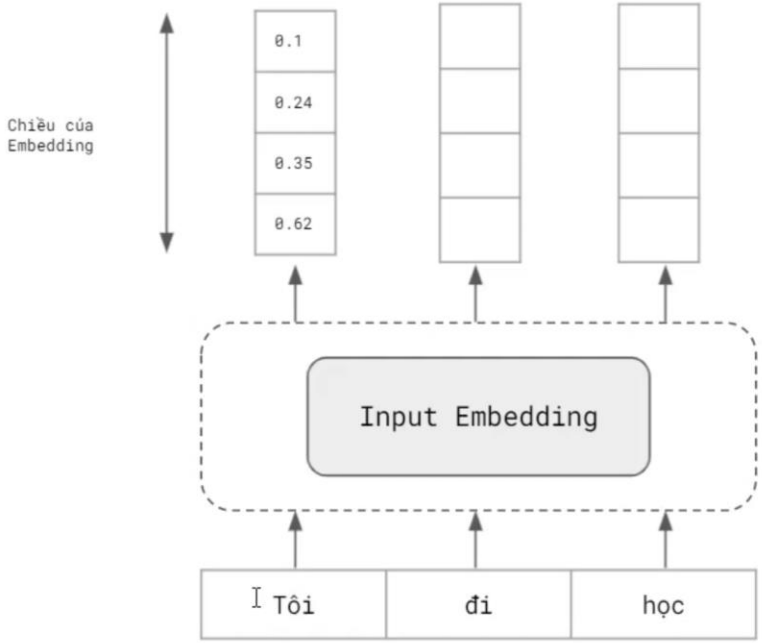
Background and related work

Transformers

PROTON X

Encoder

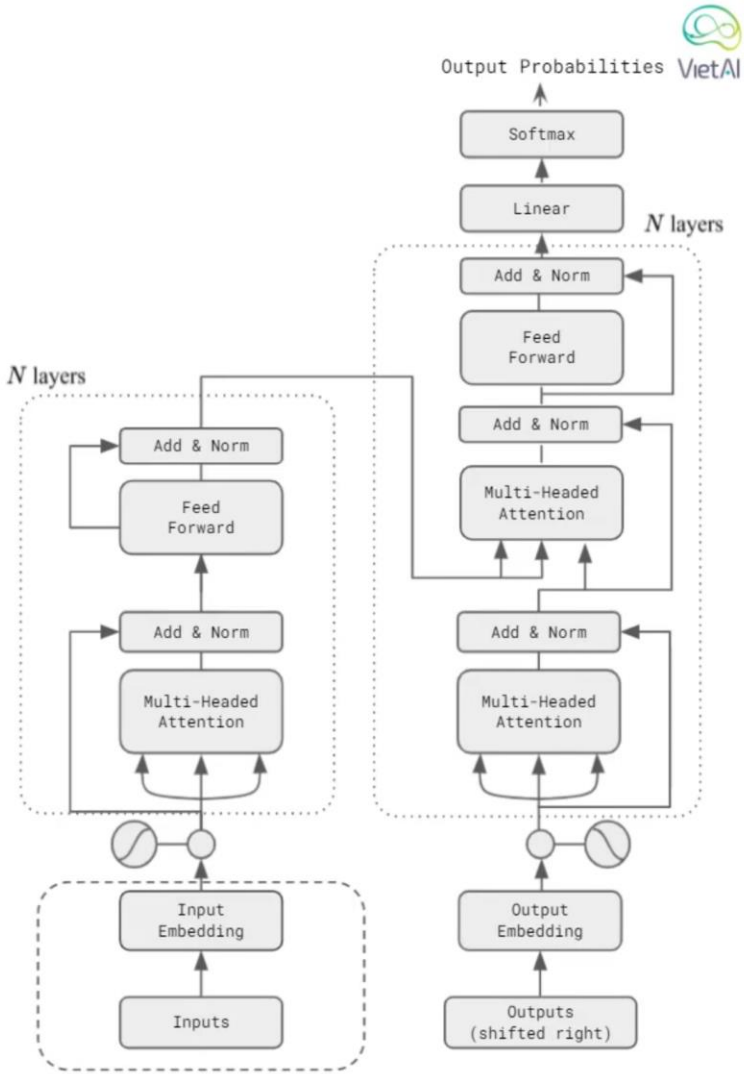
Encoder



Tương tự như RNN, Seq2Seq mô hình vẫn sử dụng Embedding để mô phỏng từ dưới dạng vector có chiều

VietAI

TensorFlow Vietnam



Background and related work

Transformers

PROTON X

VietAI

TensorFlow Vietnam

Positional encodings

Positional encodings

Bổ sung thông tin về vị trí của từ

$$z_i = WE(x_i) + PE(i)$$

Từ đi qua Embedding

Positional Encoding

1

Embedding có thể học
(learnable embedding)

2

Hàm định nghĩa trước
(pre-defined function)

Background and related work

Transformers

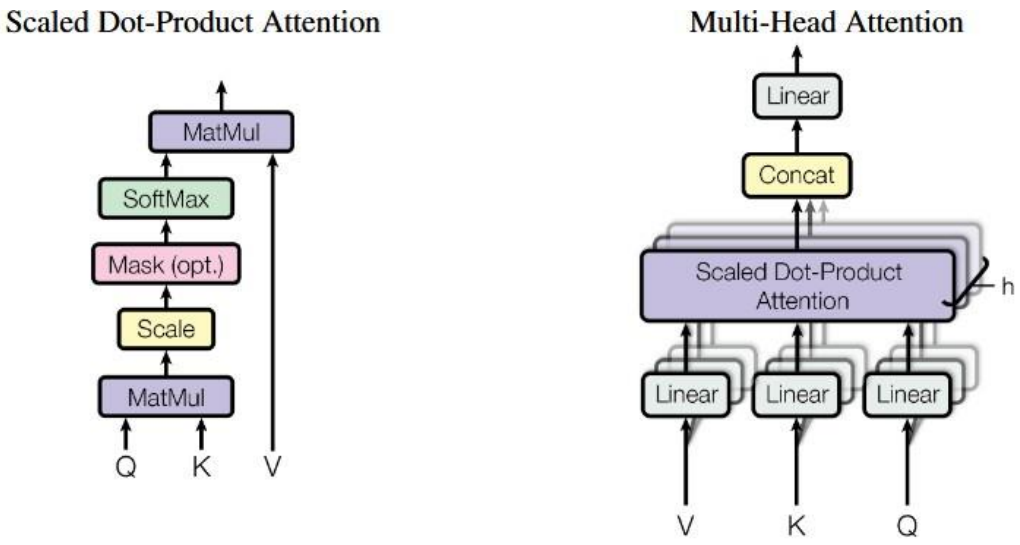


Figure 2: (left) Scaled Dot-Product Attention. (right) Multi-Head Attention consists of several attention layers running in parallel.

$$Attention(Q, K, V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

Vaswani et al., 2017; Attention is All You Need

Figure from link

Background and related work

Transformers

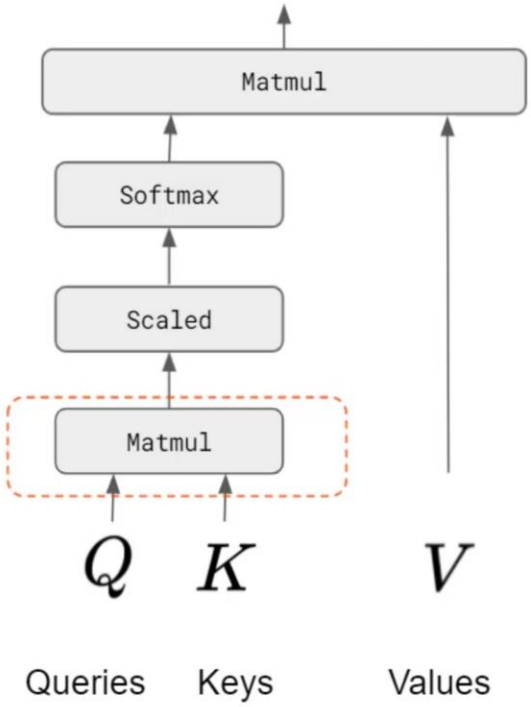
PROTON X

VietAI

TensorFlow Vietnam

Scaled dot product attention

Scaled dot product attention



Q

Tôi
đi
học

1.	2.	3.	4.
5.	2.	1.	3.
4.	3.	2.	1.

K

Tôi
đi
học

1.	2.	3.	4.
5.	2.	1.	3.
4.	3.	2.	1.

V

Tôi
đi
học

1.	2.	3.	4.
5.	2.	1.	3.
4.	3.	2.	1.

$Q \in \mathbb{R}^{(\text{length}_q) \times d_k}$ $K \in \mathbb{R}^{(\text{length}_k) \times d_k}$ $V \in \mathbb{R}^{(\text{length}_v) \times d_v}$

Background and related work

Transformers

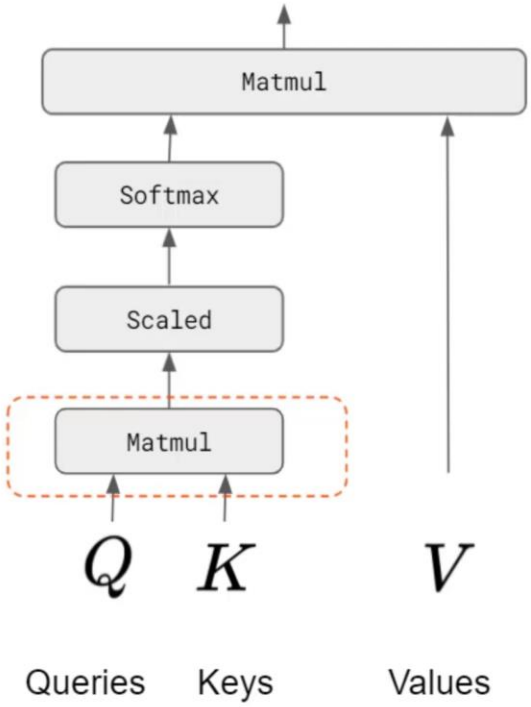
PROTON X

Scaled dot product attention

Scaled dot product attention

VietAI

TensorFlow Vietnam



Q

Tôi
đi
học

1.	2.	3.	4.
5.	2.	1.	3.
4.	3.	2.	1.

K

Tôi
đi
học

1.	2.	3.	4.
5.	2.	1.	3.
4.	3.	2.	1.

QK^T

	Tôi	đi	học
Tôi	30.	24.	21.
đi	24.	39.	33.
học	21.	33.	30.

$Q \in \mathbb{R}^{(\text{length}_q) \times d_k}$

$K \in \mathbb{R}^{(\text{length}_k) \times d_k}$

$QK^T \in \mathbb{R}^{(\text{length}_q \times \text{length}_k)}$

Background and related work

Transformers

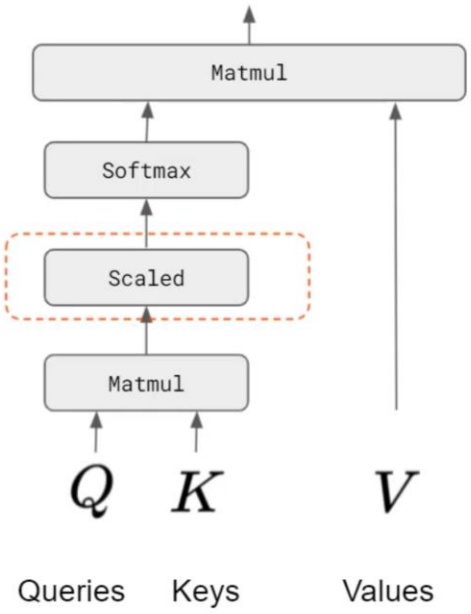
PROTON X

Scaled dot product attention

Scaled dot product attention

VietAI

TensorFlow Vietnam



QK^T

$\frac{QK^T}{\sqrt{d_k}}$

Tại sao phải chia cho $\sqrt{d_k}$

Giải thích

Việc nhân dot product có xu hướng **tăng cường độ lớn**, có xu hướng làm cho hàm softmax trả về những giá trị rất nhỏ (Hàm softmax có xu hướng tăng cường giá trị lớn, giảm nhẹ giá trị nhỏ).

Nếu q và k là biến ngẫu nhiên độc lập với mean = 0 và variance = 1, thì dot produc của 2 biến này có mean = 0 và variance = d_k

Để tránh tình trạng này chúng ta scale kết quả dot product với giá trị: $\sqrt{d_k}$

Background and related work

Transformers

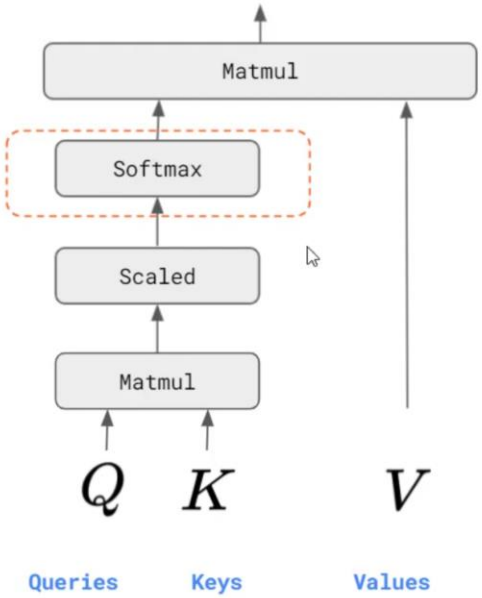
PROTON X

VietAI

TensorFlow Vietnam

Scaled dot product attention

Scaled dot product attention



$$\frac{QK^T}{\sqrt{d_k}}$$

	Tôi	đi	học
Tôi	15.	12.	10.5
đi	12.	19.5	16.5
học	10.5	16.5	15.

Sự tương xứng của "Tôi" với các từ còn lại

$$\text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)$$ Attention Weights

Tổng các giá trị bằng 1

	Tôi	đi	học
Tôi	9.42599405e-01	4.69292610e-02	1.04713335e-02
đi	5.26576432e-04	9.52072524e-01	4.74008998e-02
học	2.02246585e-03	8.15920960e-01	1.82056574e-01

Background and related work

Transformers

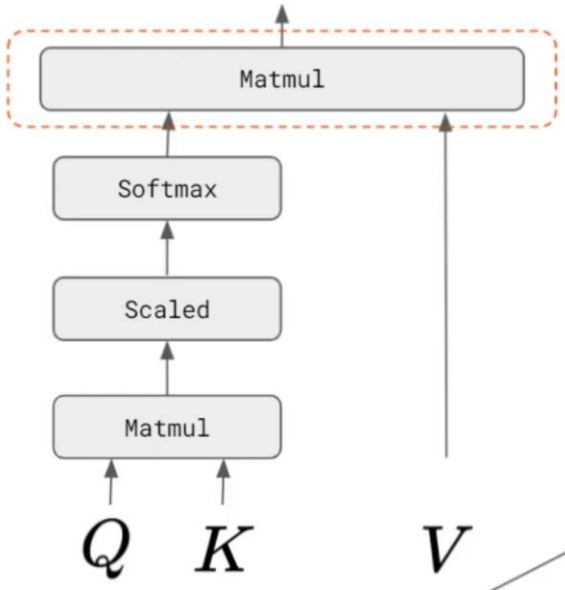
PROTON X

Scaled dot product attention

Scaled dot product attention

VietAI

TensorFlow Vietnam



$$\text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)$$

	Tôi	đi	học
Tôi	9.42599405e-01	4.69292610e-02	1.04713335e-02
đi	5.26576432e-04	9.52072524e-01	4.74008998e-02
học	2.02246585e-03	8.15920960e-01	1.82056574e-01

$$\text{Attention(Q, K, V)} = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \qquad \text{Attention(Q, K, V)} \in \mathbb{R}^{(\text{length}_q) \times d_v}$$

Tôi	1.21913104	2.01047133	2.88519881	3.93212807
đi	4.95049279	2.0474009	1.00105315	2.95312568
học	4.80985356	2.18205657	1.00404493	2.81996589

Những vector này tương đồng với context vector trong những bài trước

Background and related work

Transformers

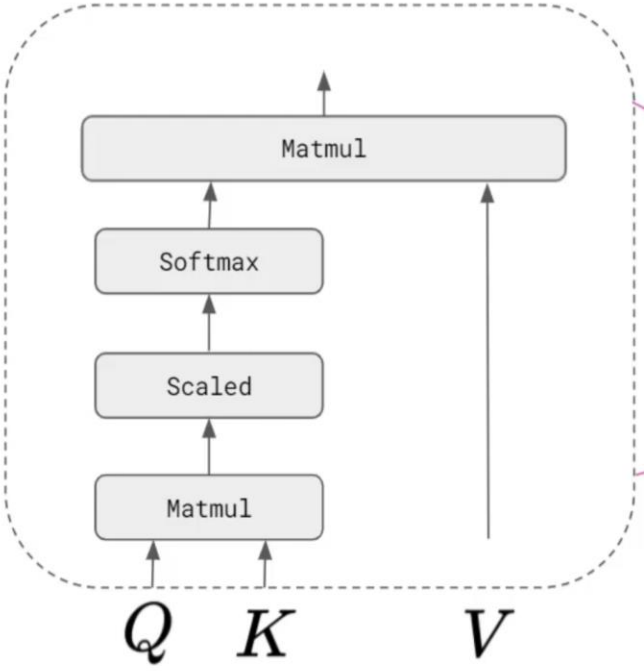
PROTON X

Why Self Attention

Why Self Attention

VietAI

TensorFlow Vietnam



Scaled Dot-Product Attention

1. Độ phức tạp tính toán mỗi lớp giảm đi
2. Các phép tính toán có thể thực hiện song song
3. Giải quyết vấn đề học long-range dependencies

Background and related work

Transformers

PROTON X

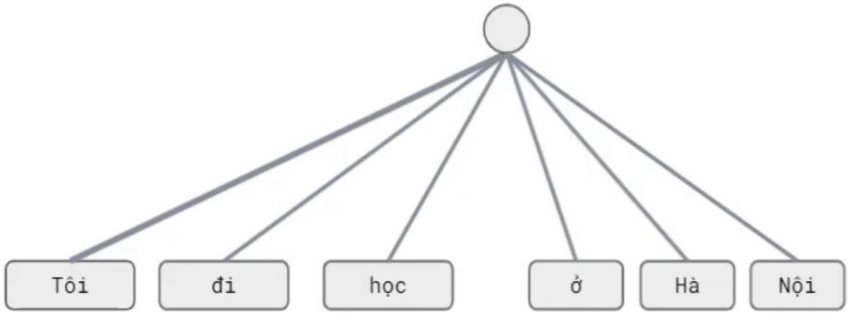
VietAI

TensorFlow Vietnam

Self Attention Problem

Self Attention Solution

Self-Attention



Khi sử dụng nhất duy nhất một Self-Attention, model thực hiện biến đổi tuyến tính trên toàn bộ embedding của các từ, cho nên không có khả năng tương tự như Convolution - bóc tách từng phần nhỏ thông tin tại các khu vực nhất định.

Cách giải quyết???

Background and related work

Transformers

PROTON X

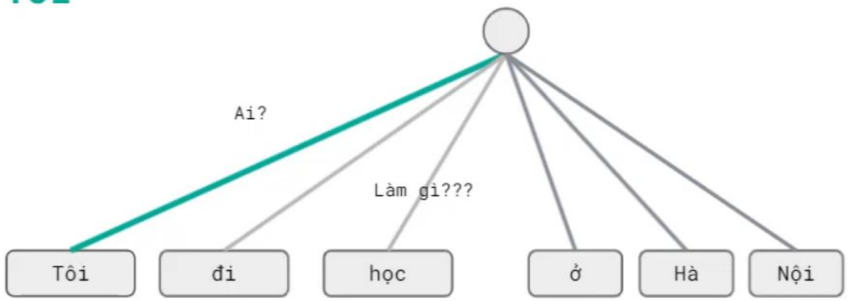
Self Attention Solution

Self Attention Solution

VietAI

TensorFlow Vietnam

Attention Head: Tôi



Mô hình sẽ sử dụng nhiều Self-Attention, mỗi attention này sẽ phụ trách học một phần thông tin của câu.

Attention Head này học embedding của từ "Tôi" để học cách đặt câu hỏi: "Ai?"

Background and related work

Transformers

PROTON X

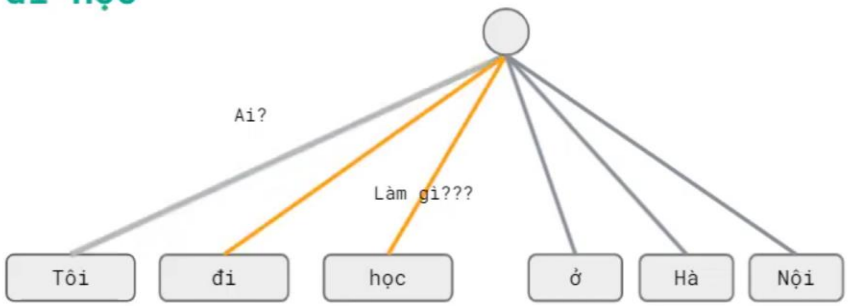
Self Attention Solution

Self Attention Solution

VietAI

TensorFlow Vietnam

Attention Head: đi học



Mô hình sẽ sử dụng nhiều Self-Attention, mỗi attention này sẽ phụ trách học một phần thông tin của câu.

Attention Head này học embedding của từ "đi" và từ "học" để đặt câu hỏi: "Làm gì?"

Background and related work

Transformers

PROTON X

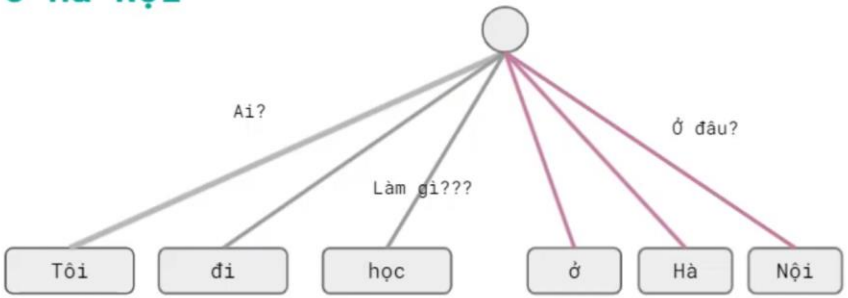
Self Attention Solution

Self Attention Solution

VietAI

TensorFlow Vietnam

Attention Head: ở Hà Nội



Mô hình sẽ sử dụng nhiều Self-Attention, mỗi attention này sẽ phụ trách học một phần thông tin của câu.

Attention Head này học embedding của từ "ở" và "Hà Nội" để đặt câu hỏi "Ở đâu?"

Background and related work

Transformers

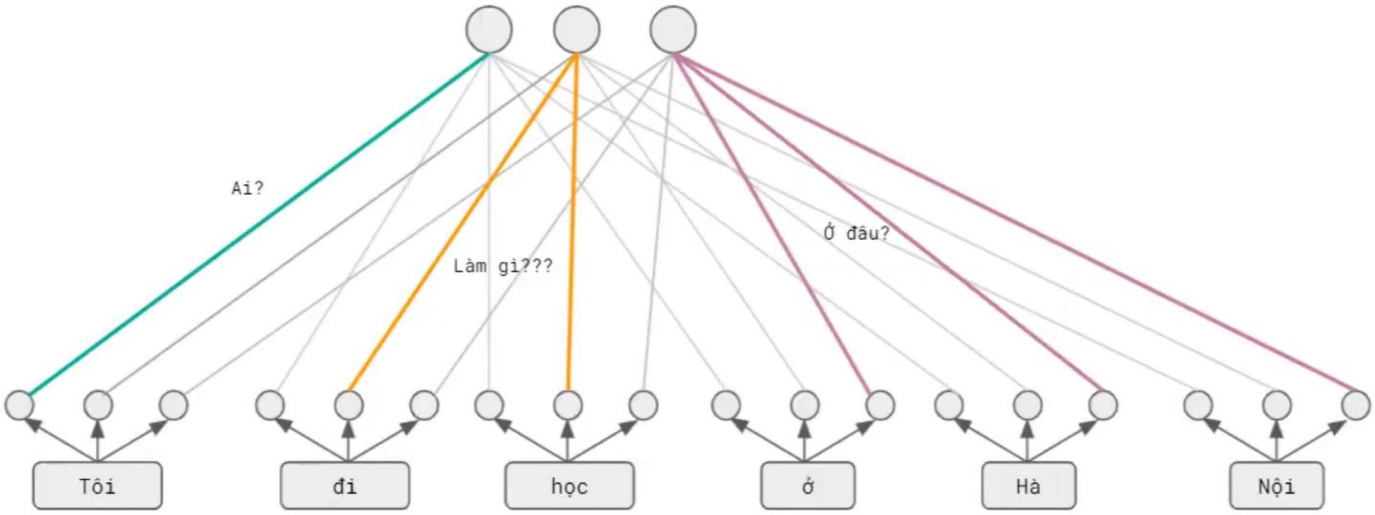
PROTON X

VietAI

TensorFlow Vietnam

Multi-Headed Attention

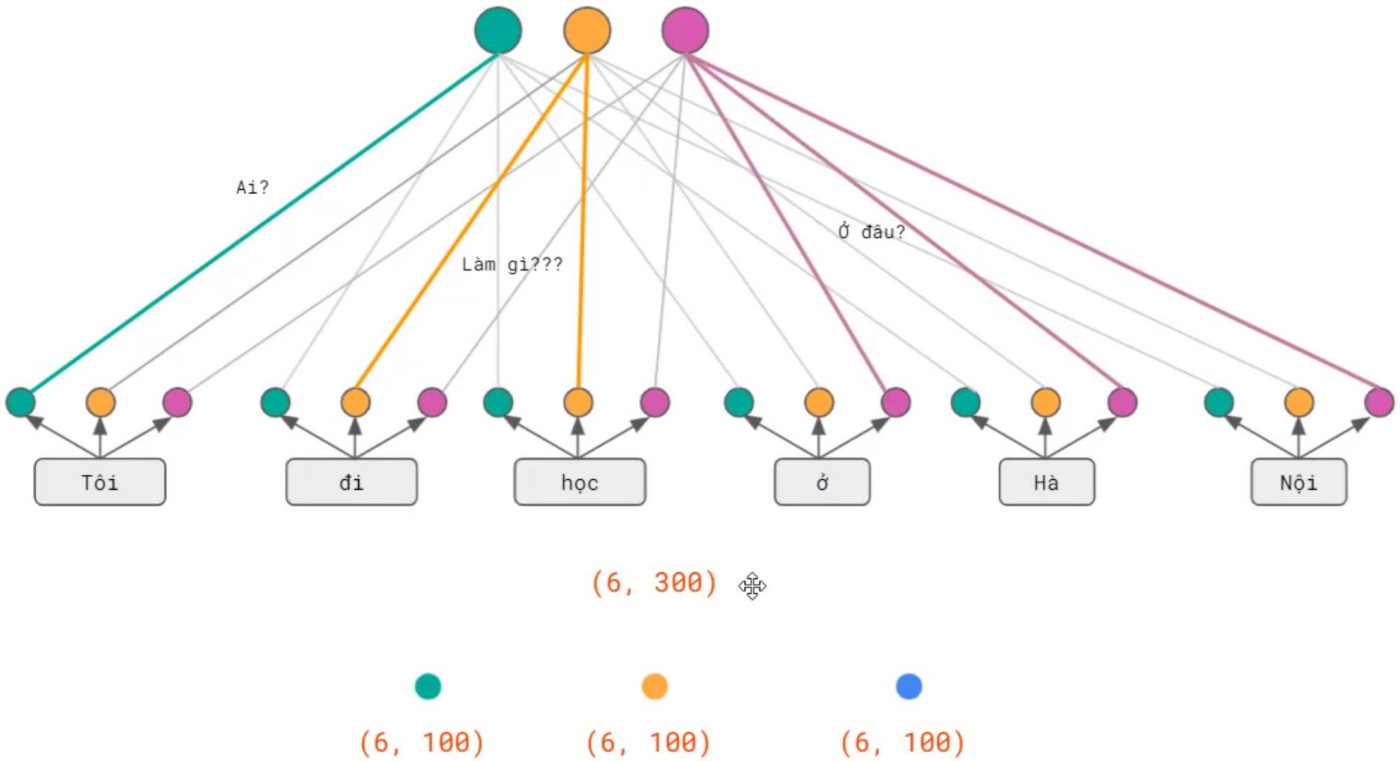
Multi-Headed Attention



Multi-Headed Attention mô phỏng việc học tương tự như Convolutions, và đặc biệt là tận dụng sức mạnh tính toán song song.

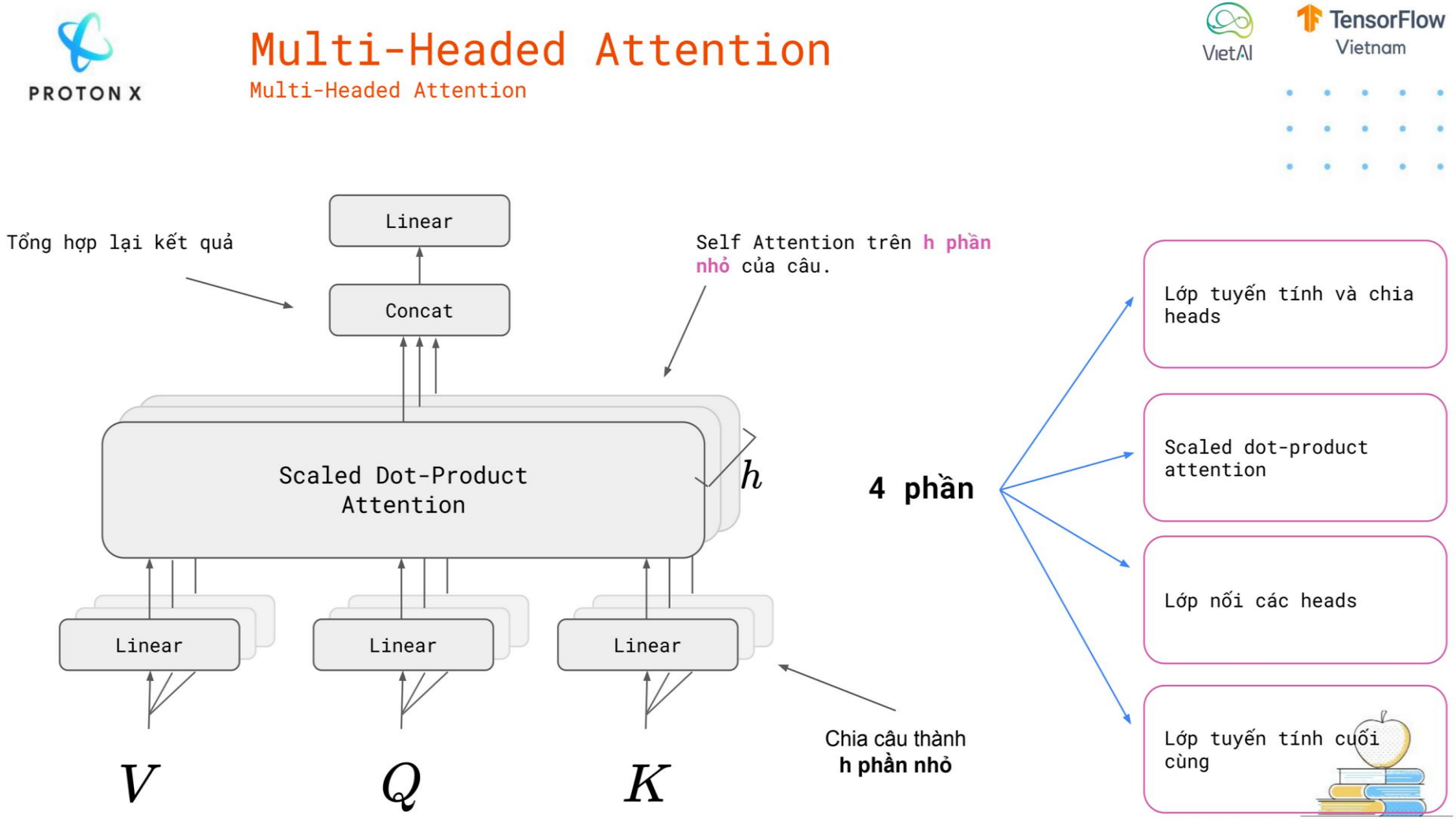
Background and related work

Transformers



Background and related work

Transformers



Background and related work

Transformers

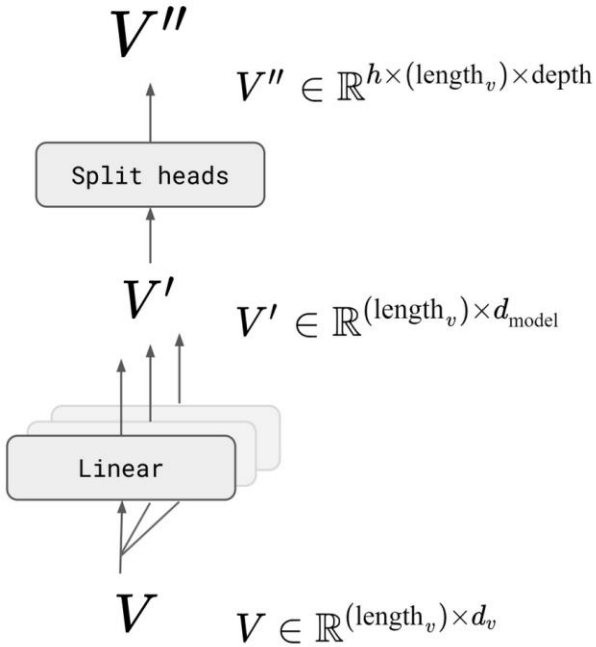
PROTON X

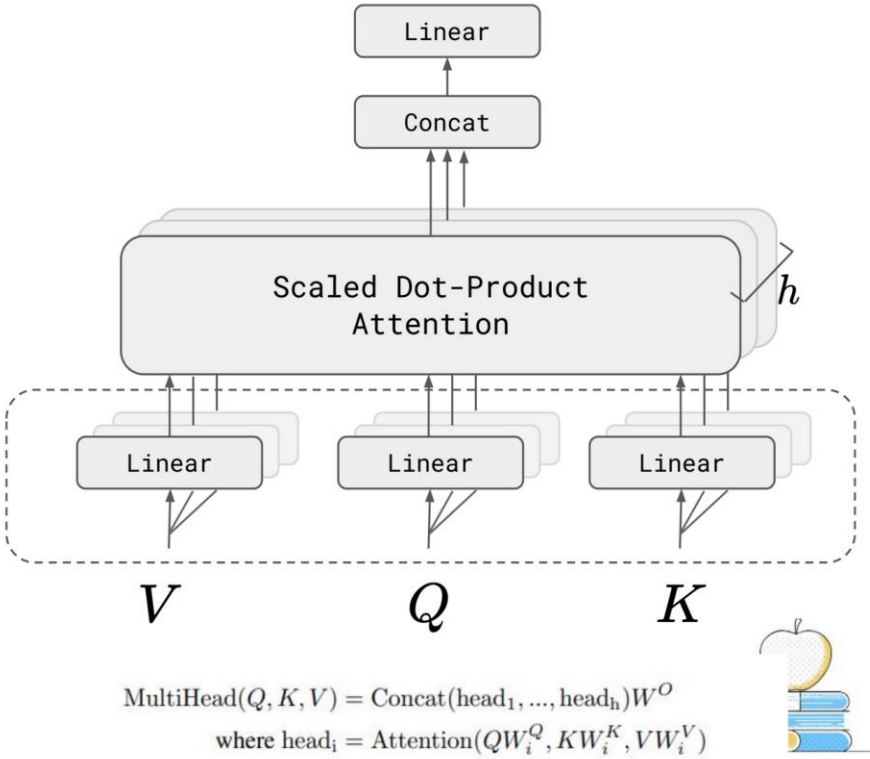
Linear layers and split into heads

Linear layers and split into heads

VietAI

TensorFlow Vietnam





Background and related work

Vision Transformer (ViT)

AN IMAGE IS WORTH 16X16 WORDS: TRANSFORMERS FOR IMAGE RECOGNITION AT SCALE

Alexey Dosovitskiy[*,†], **Lucas Beyer**[*], **Alexander Kolesnikov**[*], **Dirk Weissenborn**[*], **Xiaohua Zhai**[*], **Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, Neil Houlsby**[*,†]

[*]equal technical contribution, [†]equal advising

Google Research, Brain Team

{adosovitskiy, neilhoulsby}@google.com

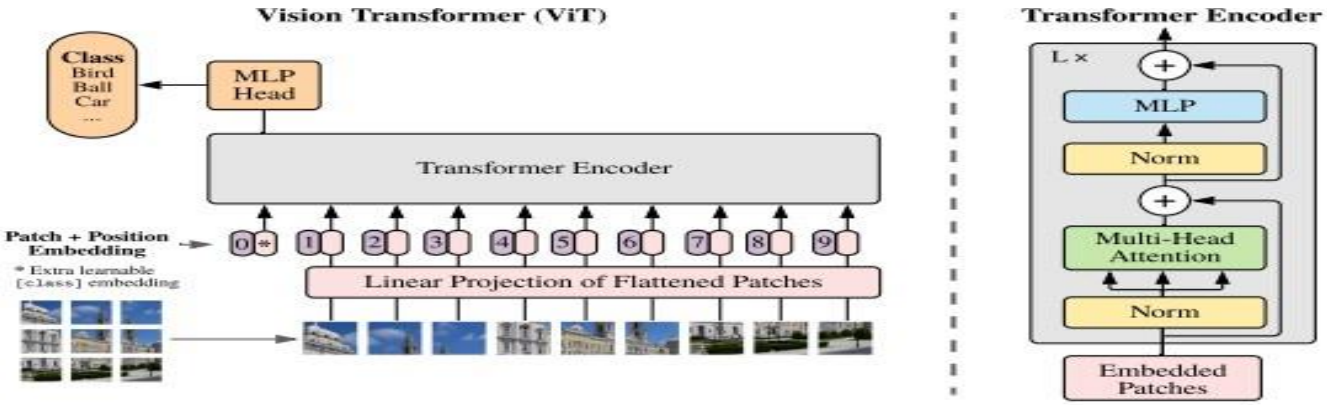


Figure 1: Model overview. We split an image into fixed-size patches, linearly embed each of them, add position embeddings, and feed the resulting sequence of vectors to a standard Transformer encoder. In order to perform classification, we use the standard approach of adding an extra learnable "classification token" to the sequence. The illustration of the Transformer encoder was inspired by Vaswani et al. (2017).

Figure from link

Dosovitsky et al., 2020; An image is worth 16x16 words: Transformers for image recognition at scale

Background and related work

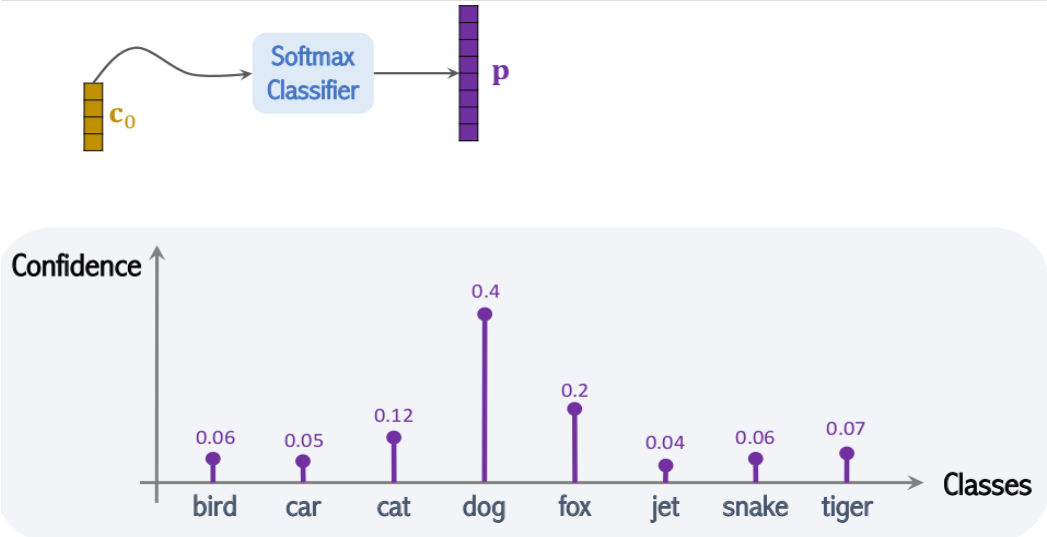
Vision Transformer (ViT)

Figure from link

Dosovitsky et al., 2020; An image is worth 16x16 words: Transformers for image recognition at scale

Background and related work

Classification Head



- Takes input as context vector c returns from Transformer Encoder
- Give the final result the probability corresponding to the classes..

Tables from link

Dosovitsky et al., 2020; An image is worth 16x16 words: Transformers for image recognition at scale

Background and related work

Model	Layers	Hidden size D	MLP size	Heads	Params
ViT-Base	12	768	3072	12	86M
ViT-Large	24	1024	4096	16	307M
ViT-Huge	32	1280	5120	16	632M

Table 1: Details of Vision Transformer model variants.

	Ours-JFT (ViT-H/14)	Ours-JFT (ViT-L/16)	Ours-I21K (ViT-L/16)	BiT-L (ResNet152x4)	Noisy Student (EfficientNet-L2)
ImageNet	**88.55** ± 0.04	87.76 ± 0.03	85.30 ± 0.02	87.54 ± 0.02	88.4/88.5*
ImageNet ReaL	**90.72** ± 0.05	90.54 ± 0.03	88.62 ± 0.05	90.54	90.55
CIFAR-10	**99.50** ± 0.06	99.42 ± 0.03	99.15 ± 0.03	99.37 ± 0.06	–
CIFAR-100	**94.55** ± 0.04	93.90 ± 0.05	93.25 ± 0.05	93.51 ± 0.08	–
Oxford-IIIT Pets	**97.56** ± 0.03	97.32 ± 0.11	94.67 ± 0.15	96.62 ± 0.23	–
Oxford Flowers-102	99.68 ± 0.02	**99.74** ± 0.00	99.61 ± 0.02	99.63 ± 0.03	–
VTAB (19 tasks)	**77.63** ± 0.23	76.28 ± 0.46	72.72 ± 0.21	76.29 ± 1.70	–
TPUv3-core-days	2.5k	0.68k	0.23k	9.9k	12.3k

Table 2: Comparison with state of the art on popular image classification benchmarks. We report mean and standard deviation of the accuracies, averaged over three fine-tuning runs. Vision Transformer models pre-trained on the JFT-300M dataset outperform ResNet-based baselines on all datasets, while taking substantially less computational resources to pre-train. ViT pre-trained on the smaller public ImageNet-21k dataset performs well too. *Slightly improved 88.5% result reported in Touvron et al. (2020).

Tables from link

Dosovitsky et al., 2020; An image is worth 16x16 words: Transformers for image recognition at scale

Methods

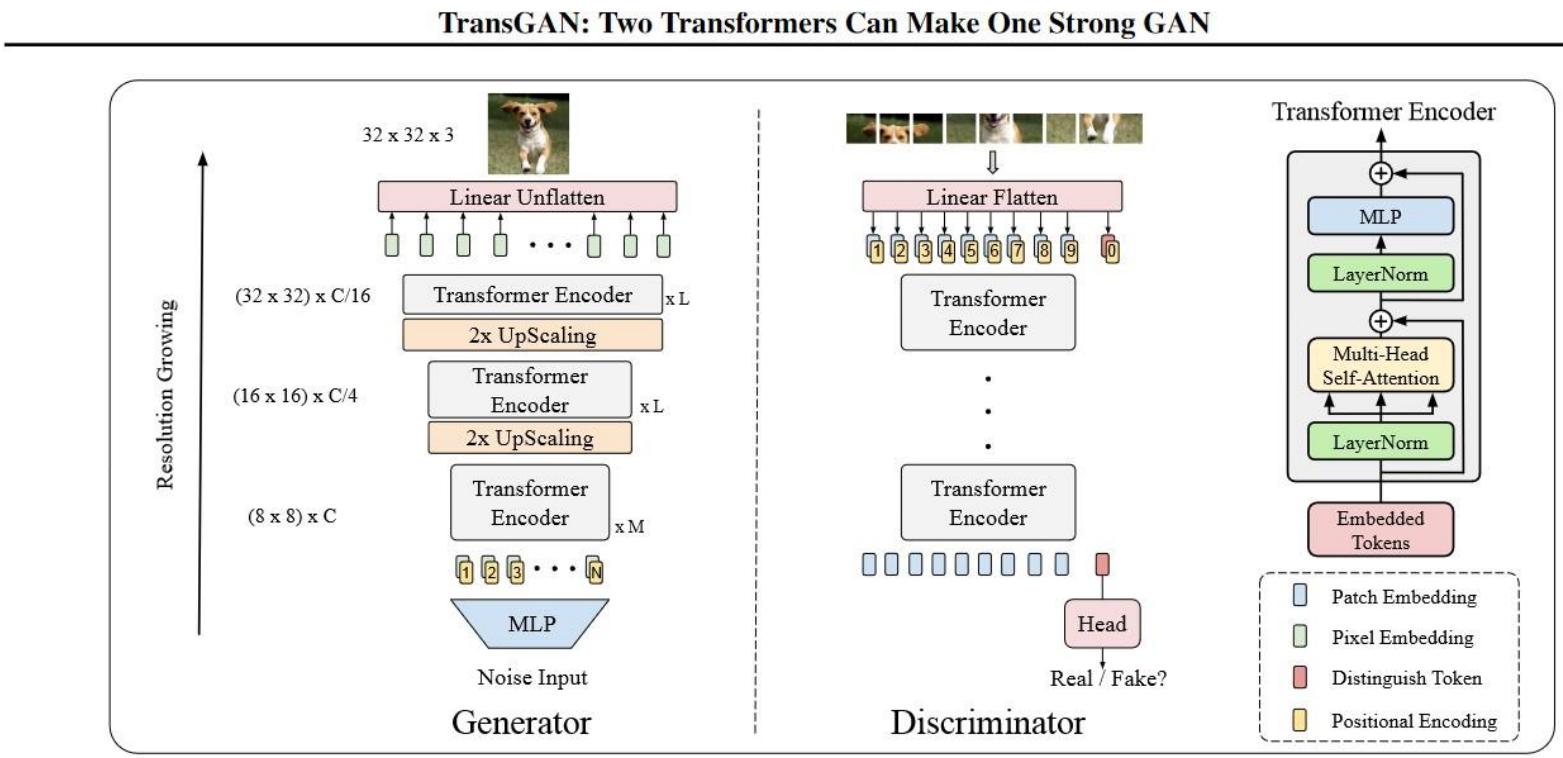


Figure 1. The pipeline of the pure transform-based generator and discriminator of TransGAN. Here $H = W = 8$ and $H_T = W_T = 32$. We show 9 patches for discriminator as an example while in practice we use 8×8 patches across all datasets.

Figure from paper

Up Scaling

Real-Time Single Image and Video Super-Resolution Using an Efficient Sub-Pixel Convolutional Neural Network

Wenzhe Shi[1], Jose Caballero[1], Ferenc Huszár[1], Johannes Totz[1], Andrew P. Aitken[1], Rob Bishop[1], Daniel Rueckert[1], Zehan Wang[1]
[1]Twitter
[1]{wshi, jcaballero, fhuszar, jtotz, aitken, rbishop, zehanw}@twitter.com

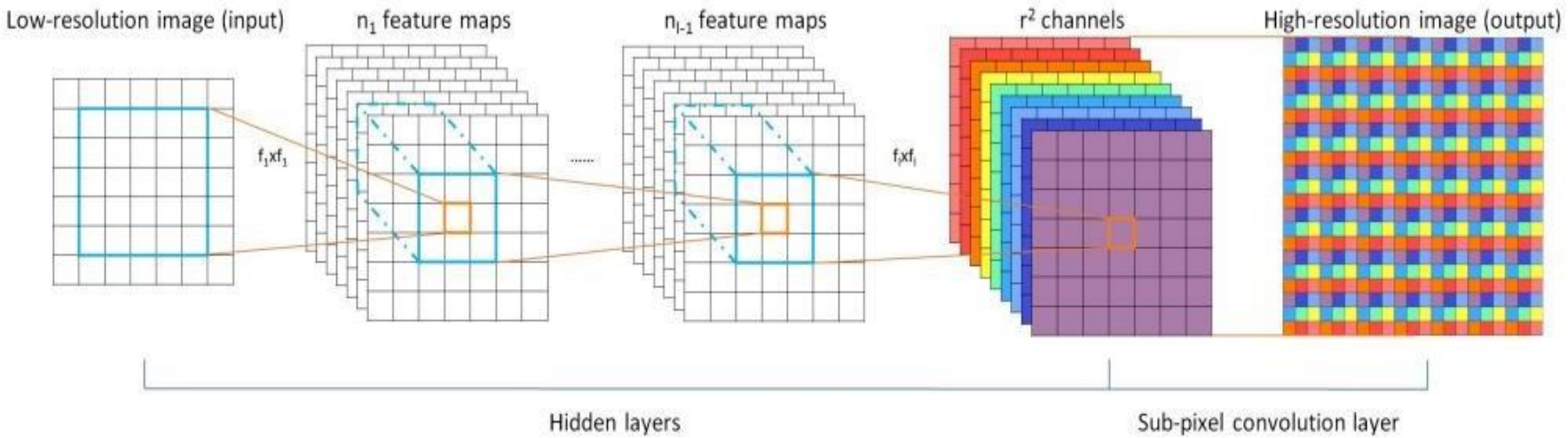


Figure 1. The proposed efficient sub-pixel convolutional neural network (ESPCN), with two convolution layers for feature maps extraction, and a sub-pixel convolution layer that aggregates the feature maps from LR space and builds the SR image in a single step.

Figure from link

Evaluation

Table 1. Inception Score (IS) and FID results on CIFAR-10. The first row shows the AutoGAN results (Gong et al., 2019); the second and thirds row show the mixed transformer-CNN results; and the last row shows the pure-transformer GAN results.

GENERATOR	DISCRIMINATOR	IS↑	FID↓
AUTOGAN	AUTOGAN	8.55± 0.12	**12.42**
TRANSFORMER	AUTOGAN	**8.59± 0.10**	13.23
AUTOGAN	TRANSFORMER	6.17± 0.12	49.83
TRANSFORMER	TRANSFORMER	6.95 ± 0.13	41.41

Table from paper

Methods

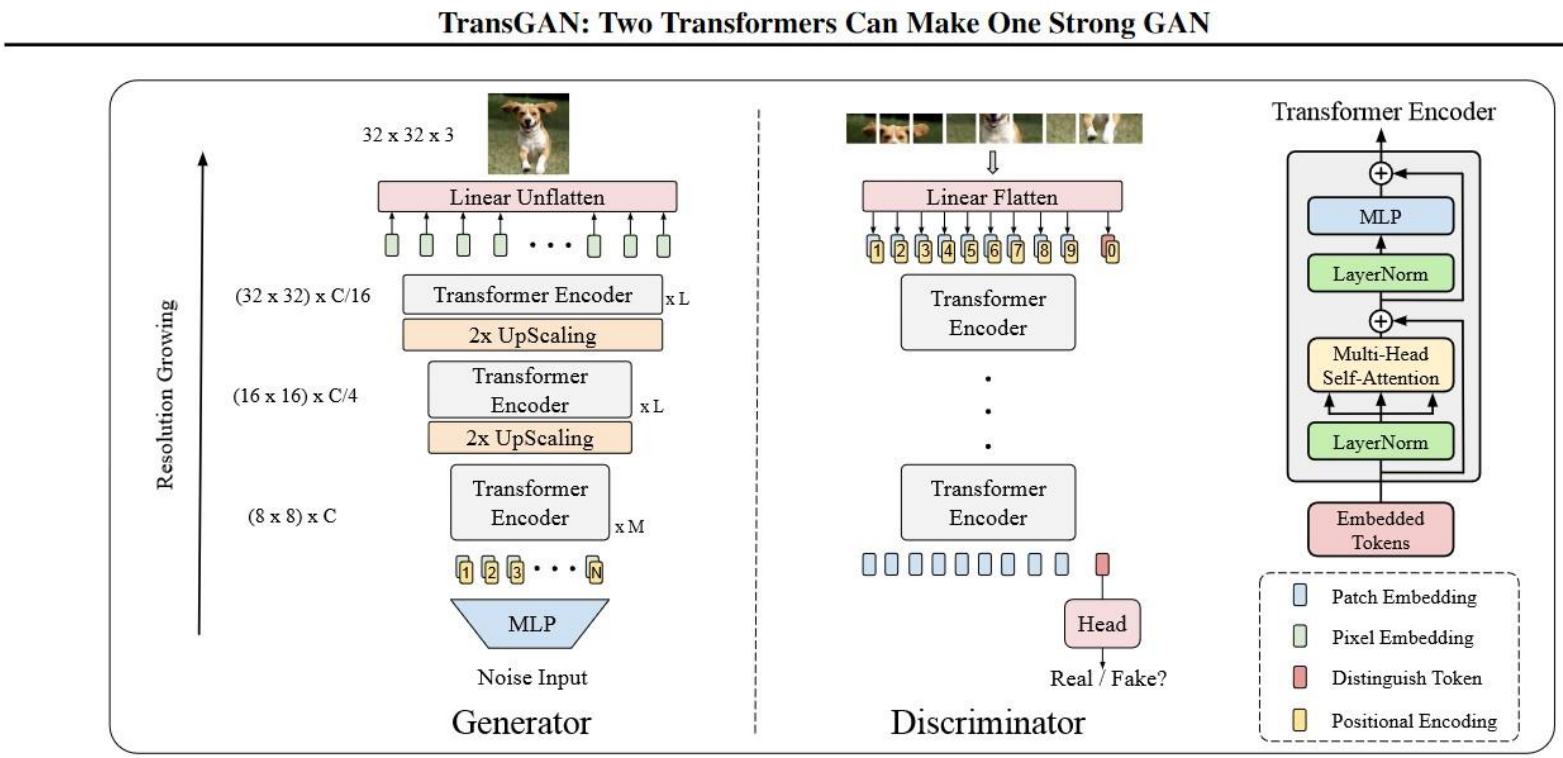


Figure 1. The pipeline of the pure transform-based generator and discriminator of TransGAN. Here $H = W = 8$ and $H_T = W_T = 32$. We show 9 patches for discriminator as an example while in practice we use 8×8 patches across all datasets.

Figure from paper

Data Augmentation

Differentiable Augmentation for Data-Efficient GAN Training

Shengyu Zhao
IIIS, Tsinghua University and MIT

Zhijian Liu
MIT

Ji Lin
MIT

Jun-Yan Zhu
Adobe and CMU

Song Han
MIT

Github: **https://github.com/mit-han-lab/data-efficient-gans**

Data Augmentation

Differentiable Augmentation for Data-Efficient GAN Training

Shengyu Zhao
IIS, Tsinghua University and MIT

Zhijian Liu
MIT

Ji Lin
MIT

Jun-Yan Zhu
Adobe and CMU

Song Han
MIT

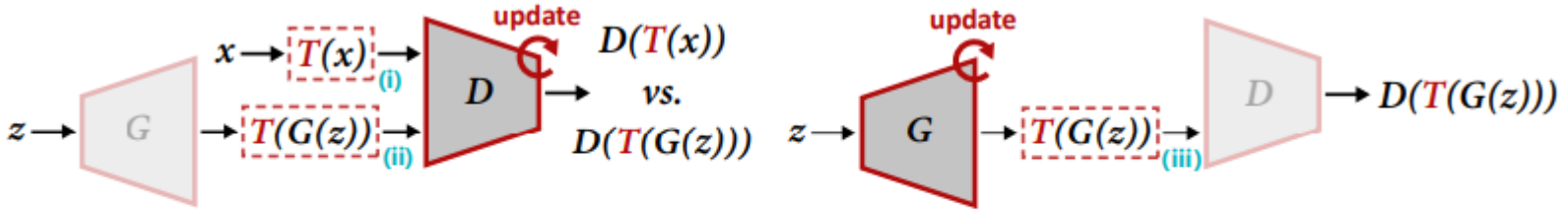


Figure 4: **Overview of DiffAugment** for updating D (left) and G (right). DiffAugment applies the augmentation T to both the real samples $\boldsymbol{x}$ and the generated output $G(\boldsymbol{z})$. When we update G, gradients need to be back-propagated through T, which requires T to be differentiable w.r.t. the input.

Paper: https://arxiv.org/pdf/2006.10738.pdf

Data Augmentation

Table 2. The effectiveness of Data Augmentation (DA) on both CNN-based GANs and TransGAN. We used the full CIFAR-10 training set and DiffAug (Zhao et al., 2020b).

METHODS	DA	IS ↑	FID ↓
WGAN-GP	×	**6.49** ± 0.09	39.68
(GULRAJANI ET AL., 2017)	√	6.29 ± 0.10	**37.14**
AUTOGAN	×	8.55 ± 0.12	**12.42**
(GONG ET AL., 2019)	√	**8.60 ± 0.10**	12.72
STYLEGAN V2	×	9.18	11.07
(ZHAO ET AL., 2020B)	√	**9.40**	**9.89**
TRANSGAN	×	6.95 ± 0.13	41.41
	√	**8.15 ± 0.14**	**19.85**

Table from paper

Co-Training with Self-Supervised Auxiliary Task

- An auxiliary task of super resolution, in addition to the GAN loss. This task comes "for free", since we can just treat the available real images as high-resolution, and down sample them to obtain low-resolution counterparts.

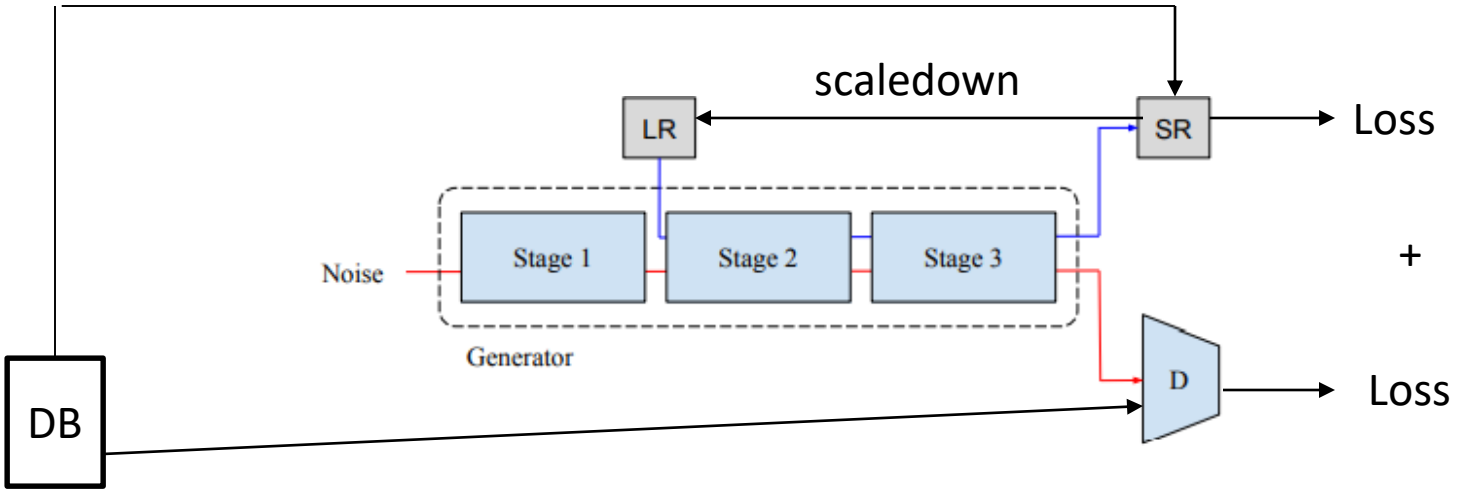


Figure 2. Co-training the transformer G with an auxiliary task of super resolution. "LR" and "SR" represent low-resolution input and high-resolution output respectively.

Figure from paper

Co-Training with Self-Supervised Auxiliary Task

- The generator loss is added with a auxiliary term $\lambda * L_{SR}$, where L_{SR} is the mean-square-error (MSE) loss and the coefficient λ is empirically set as 50.

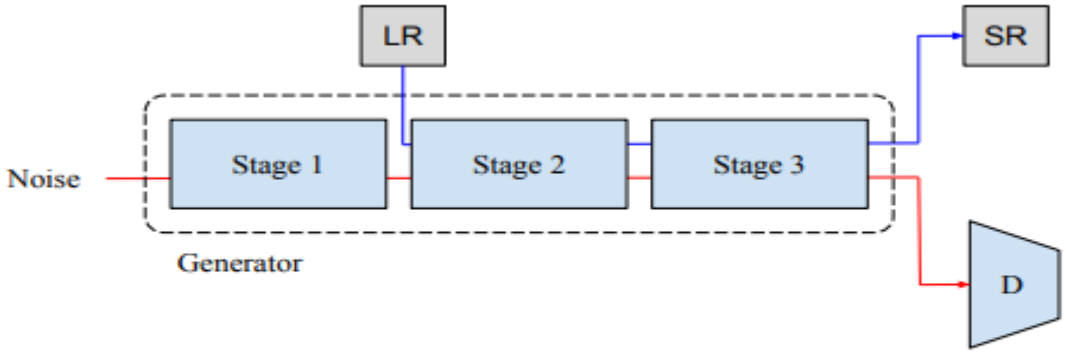


Figure 2. Co-training the transformer G with an auxiliary task of super resolution. "LR" and "SR" represent low-resolution input and high-resolution output respectively.

Table from paper

Co-Training with Self-Supervised Auxiliary Task

Table 3. Ablation studies for multi-task co-training (MT-CT) and locality-aware self-attention initialization on TransGAN.

MODEL	IS↑	FID↓
TRANSGAN + DA (*)	8.15± 0.14	19.85
(*) + MT-CT	8.20± 0.14	19.12
(*) + MT-CT + LOCAL INIT.	**8.22± 0.12**	**18.58**

Table from paper

Locality-Aware Initialization for Self-Attention

- The CNN architecture has built-in natural image smoothing (Ulyanov et al., 2018), which is believed to contribute to natural image generation.
- (Dosovitskiy et al., 2020) observed that transformers still tend to learn convolutional structures from images.
- Therefore, a meaningful question arises as, can we efficiently encode inductive image biases while still retaining the flexibility of transformers?
- Introduced a mask by which each query is only allowed to interact with its local neighbors that are not “masked”.

Dosovitsky et al., 2020; An image is worth 16x16 words: Transformers for image recognition at scale
Ulyanov et al., 2018; Deep Image Prior

Locality-Aware Initialization for Self-Attention

- Different from previous methods (Daras et al., 2020; Parmar et al., 2018; Child et al., 2019; Beltagy et al., 2020) during training we gradually reduce the mask until diminishing it, and eventually the self-attention is fully global.

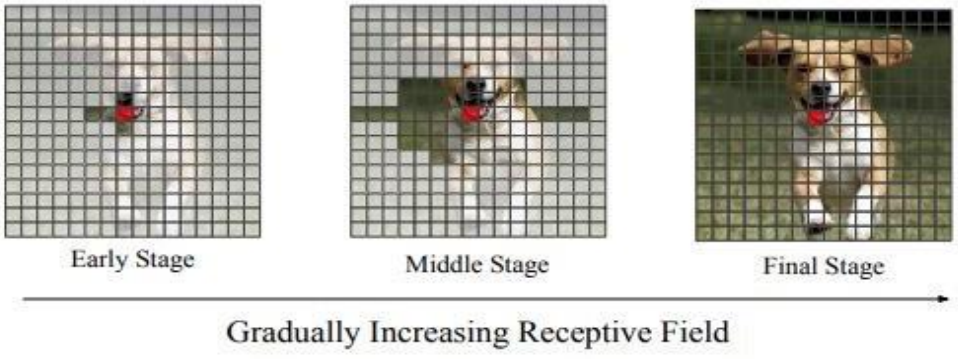


Figure 3. Locality-aware initialization for self-attention. The red block indicates a query location, the transparent blocks are its allowable key locations to interact with, and the gray blocks indicate the masked region. TransGAN gradually increases the allowable region during the training process.

Figure from paper

Locality-Aware Initialization for Self-Attention

- During training the mask gradually reduced until diminishing it, and eventually the self-attention is fully global.
- A localized self-attention (Parmar et al., 2018) is most helpful at the early training stage, but can hurt the later training stage and the final achievable performance.
- We consider this locality-aware initialization as a regularizer that comes for the early training dynamics and then gradually fades away (Golatkar et al., 2019).

Parmar et al., 2018; Image transformer

Golatkar et al., 2019 ; Time matters in regularizing deep networks: Weight decay and data augmentation affect early learning dynamics, matter little near convergence.

Scaling up to Large Models

Table 4. Scaling-up the model size of TransGAN on CIFAR-10. Here "Dim" represents the embedded dimension of transformer and "Depth" is the number of transformer encoder block in each stage.

MODEL	DEPTH	DIM	IS ↑	FID ↓
TRANSGAN-S	{5,2,2}	384	8.22 ± 0.14	18.58
TRANSGAN-M	{5,2,2}	512	8.36 ± 0.12	16.27
TRANSGAN-L	{5,2,2}	768	8.50 ± 0.14	14.46
TRANSGAN-XL	{5,4,2}	1024	$\mathbf{8.63 \pm 0.16}$	**11.89**

Tables from paper

Experiments & Results

(a) Cifar-10 (b) STL-10 (c) CelebA 64 x 64

Figure 4. Visual Results of TransGAN on CIFAR-10, STL-10, and CelebA 64 × 64.

Figure from paper

Datasets and Implementation

Datasets

- CIFAR-10: 60k 32×32 color images in 10 classes
  - 50k training and 10k testing images respectively
  - 50k training images without labels
  - 32×32 resolution
- STL-10
  - 5k training images and 100k unlabeled images for training
  - 48 × 48 resolution
- CelebA
  - 200k unlabeled face images
  - 64 × 64 resolution.

Datasets and Implementation

Implementation

- The author follows the implementation of WGAN [46] and uses WGAN-GP loss
- Learning rate là 1e − 4 cho cả generator và discriminator
- adam optimize
- batch size 128 for Generator and 64 for Discriminator
- For image with high resolution 64x64 thì batch size 32 for Generator and 16 for Discriminator.
- Device: 4 V100 GPUs
- Training took 2 days on the CIFAR-10 dataset, 3 days on STL-10 dataset and CelebA.dataset

Evaluation Methods

$$\mathrm{IS}(G) = \exp\left(\mathbb{E}_{\mathbf{x}\sim p_g} D_{KL}\left(p(y|\mathbf{x}) \parallel p(y) \right) \right)$$

$$\mathrm{FID}(x, g) = \left\| \mu_x - \mu_g \right\|_2^2 + \mathrm{Tr}\left(\Sigma_x + \Sigma_g - 2(\Sigma_x \Sigma_g)^{\frac{1}{2}} \right),$$

Tables from paper

Experiments & Results

Table 5. Unconditional image generation results on CIFAR-10.

METHODS	IS	FID
WGAN-GP (GULRAJANI ET AL., 2017)	6.49 ± 0.09	39.68
LRGAN (YANG ET AL., 2017)	7.17 ± 0.17	-
DFM (WARDE-FARLEY & BENGIO, 2016)	7.72 ± 0.13	-
SPLITTING GAN (GRINBLAT ET AL., 2017)	7.90 ± 0.09	-
IMPROVING MMD-GAN (WANG ET AL., 2018A)	8.29	16.21
MGAN (HOANG ET AL., 2018)	8.33 ± 0.10	26.7
SN-GAN (MIYATO ET AL., 2018)	8.22 ± 0.05	21.7
PROGRESSIVE-GAN (KARRAS ET AL., 2017)	8.80 ± 0.05	15.52
AUTOGAN (GONG ET AL., 2019)	8.55 ± 0.10	12.42
STYLEGAN V2 (ZHAO ET AL., 2020B)	**9.18**	**11.07**
TRANSGAN-XL	8.63 ± 0.16	11.89

Tables from paper

Experiments & Results

Table 5. Unconditional image generation results on CIFAR-10.

METHODS	IS	FID
WGAN-GP (GULRAJANI ET AL., 2017)	6.49 ± 0.09	39.68
LRGAN (YANG ET AL., 2017)	7.17 ± 0.17	-
DFM (WARDE-FARLEY & BENGIO, 2016)	7.72 ± 0.13	-
SPLITTING GAN (GRINBLAT ET AL., 2017)	7.90 ± 0.09	-
IMPROVING MMD-GAN (WANG ET AL., 2018A)	8.29	16.21
MGAN (HOANG ET AL., 2018)	8.33 ± 0.10	26.7
SN-GAN (MIYATO ET AL., 2018)	8.22 ± 0.05	21.7
PROGRESSIVE-GAN (KARRAS ET AL., 2017)	8.80 ± 0.05	15.52
AUTOGAN (GONG ET AL., 2019)	8.55 ± 0.10	12.42
STYLEGAN V2 (ZHAO ET AL., 2020B)	**9.18**	**11.07**
TRANSGAN-XL	8.63 ± 0.16	11.89

Tables from paper

Experiments & Results

Table 6. Unconditional image generation results on STL-10.

METHODS	IS ↑	FID ↓
DFM (WARDE-FARLEY & BENGIO, 2016)	8.51 ± 0.13	-
D2GAN (NGUYEN ET AL., 2017)	7.98	-
PROBGAN (HE ET AL., 2019)	8.87 ± 0.09	47.74
DIST-GAN (TRAN ET AL., 2018)	-	36.19
SN-GAN (MIYATO ET AL., 2018)	9.16 ± 0.12	40.1
IMPROVING MMD-GAN (WANG ET AL., 2018A)	9.23 ± 0.08	37.64
AUTOGAN (GONG ET AL., 2019)	9.16 ± 0.12	31.01
ADVERSARIALNAS-GAN (GAO ET AL., 2020)	9.63 ± 0.19	26.98
TRANSGAN-XL	$\mathbf{10.10 \pm 0.17}$	**25.32**

Tables from paper

Our Experiment and Results

In this project, we aimed to reproduce qualitative results(generating image samples by CIFAR-10 Dataset) and quantitative results in Table 2 and Table 4 of the original paper that shown below.

Table 2. The effectiveness of Data Augmentation (DA) on both CNN-based GANs and TransGAN. We used the full CIFAR-10 training set and DiffAug (Zhao et al., 2020b).

METHODS	DA	IS ↑	FID ↓
WGAN-GP	×	**6.49** ± 0.09	39.68
(GULRAJANI ET AL., 2017)	√	6.29 ± 0.10	**37.14**
AUTOGAN	×	8.55 ± 0.12	**12.42**
(GONG ET AL., 2019)	√	**8.60** ± **0.10**	12.72
STYLEGAN V2	×	9.18	11.07
(ZHAO ET AL., 2020B)	√	**9.40**	**9.89**
TRANSGAN	×	6.95 ± 0.13	41.41
	√	**8.15** ± **0.14**	**19.85**

Table 4. Scaling-up the model size of TransGAN on CIFAR-10. Here "Dim" represents the embedded dimension of transformer and "Depth" is the number of transformer encoder block in each stage.

MODEL	DEPTH	DIM	IS ↑	FID ↓
TRANSGAN-S	{5,2,2}	384	8.22 ± 0.14	18.58
TRANSGAN-M	{5,2,2}	512	8.36 ± 0.12	16.27
TRANSGAN-L	{5,2,2}	768	8.50 ± 0.14	14.46
TRANSGAN-XL	{5,4,2}	1024	**8.63** ± **0.16**	**11.89**

Tables from paper

Since we have limited computational resource and time for the training all size of TransGAN model on CIFAR-10 Dataset, we only trained the largest model with data augmentation, TransGAN-XL, for Table 4 results.

Conclusion

- **Model Architecture:**
  - Build the first GAN using purely transformers and no convolution. To avoid over-whelming memory overheads,
  - Memory-friendly generator and a patch-level discriminator created, both transformer-based without bells and whistles.
  - Trans-GAN can be effectively scaled up to larger models.

Conclusion

- **Training Technique:**
  - To train TransGAN better, ranging from **data augmentation**, **multi-task co-training for generator with self-supervised auxiliary loss**, **and localized initialization for self-attention**.
- **Performance:**
  - TransGAN achieves highly competitive performance compared to current **state-of-the-art CNN**-based GANs.

Thank you for listening!