

# **Лекция 11. Vision Transformers. Self-supervised и contrastive Learning**

Денис Деркач, Дмитрий Тарасов

Использовались слайды Антона Кленицкого



# План

Вспоминаем трансформеры

- ▶ Внимание
- ▶ Encoder-Decoder Transformer
- ▶ Encoder-only Transformer

Архитектуры:

- ▶ Vision Transformer
- ▶ Swin transformer

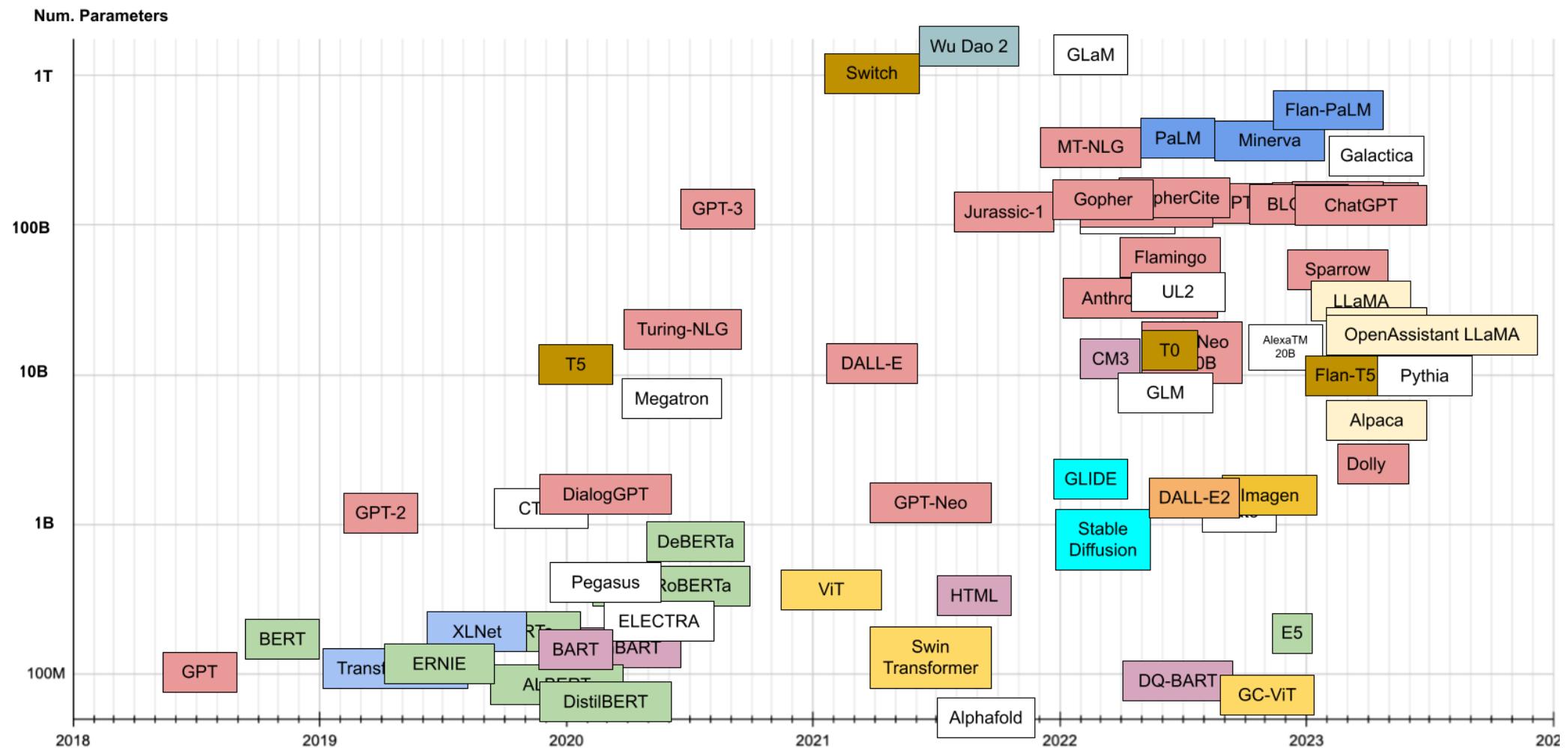
Методы обучения:

- ▶ Self-supervised Learning
- ▶ Contrastive Learning
- ▶ Multimodal Contrastive Learning

В предыдущих лекциях



# Зоопарк трансформеров



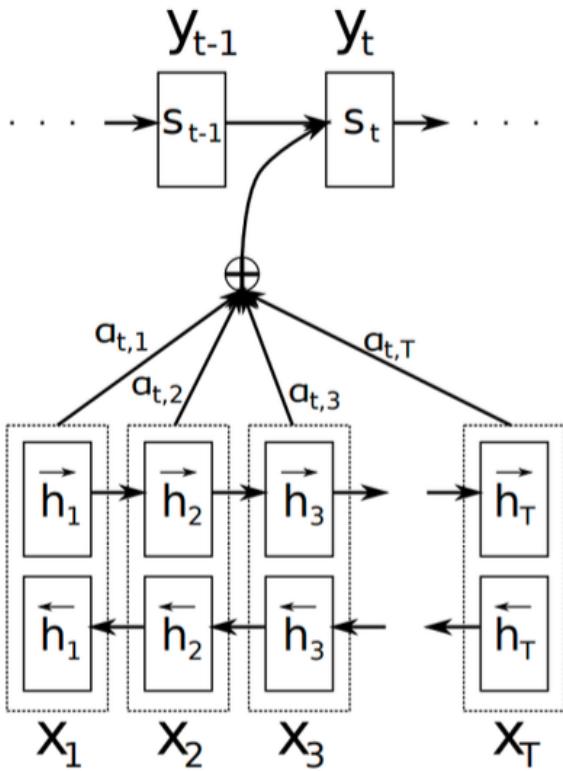
<https://amatria.in/blog/transformer-models-an-introduction-and-catalog-2d1e9039f376/>

# Методы построения эмбедингов

- ▶ Word2Vec: используйте, когда семантические связи имеют решающее значение, и у вас большой набор данных.
- ▶ GloVe: подходит для разнообразных наборов данных и когда важен охват глобального контекста.
- ▶ FastText: выбирайте морфологически богатые языки или когда обработка слов, не входящих в словарный запас, имеет решающее значение.

# Attention

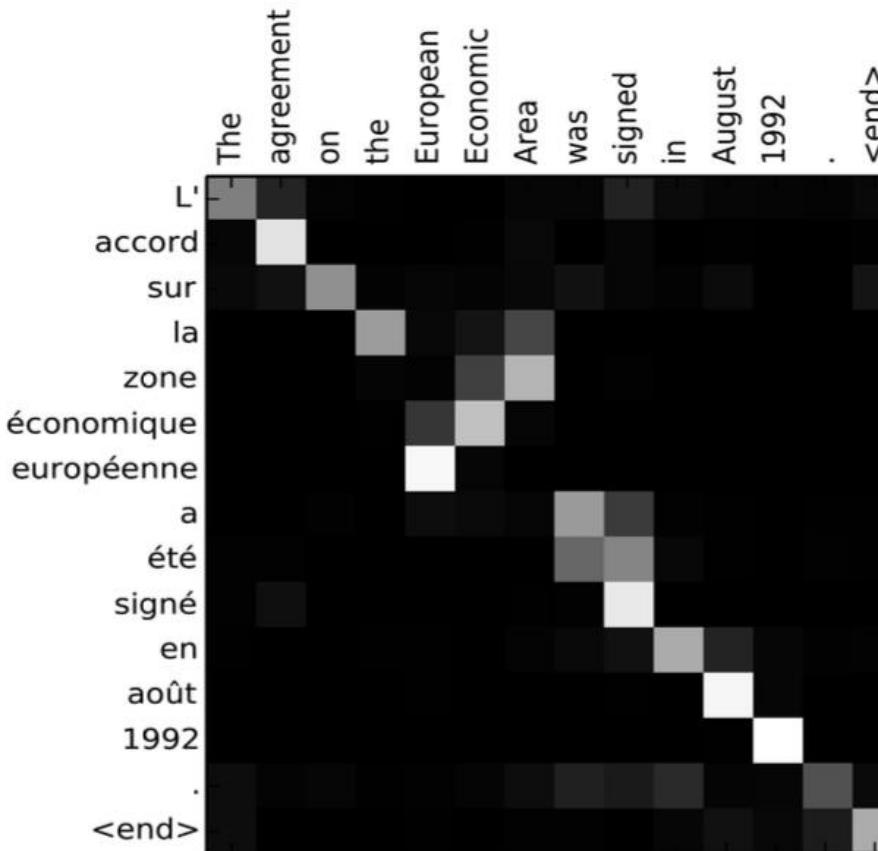
Bahdanau, D., Cho, K., & Bengio, Y. (2015) Neural machine translation by jointly learning to align and translate.



- Умный pooling
- Вектор контекста  $c_t$  свой на каждом шаге декодера
- $c_t$  - сумма  $h_i$  со всех шагов энкодера с обучаемыми весами
- Внимание выбирает релевантные элементы во входной последовательности

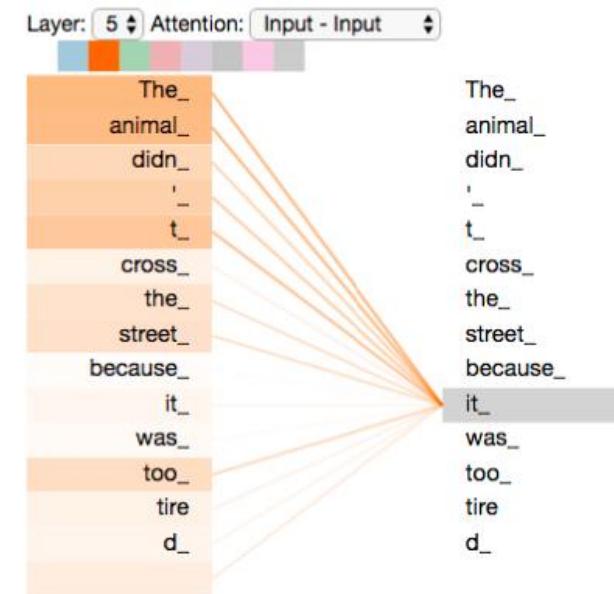
# Интерпретация внимания

Матрица весов  $\alpha_{ti}$  определяет alignment (выравнивание) элементов входной и выходной последовательностей



# Self-attention

- Attention позволяет получить представление входной последовательности
- как и RNN
- Почему бы не отказаться от рекуррентности полностью?



# Self-attention

$x_i$  - элементы последовательности с размерностью  $d$

$$q_i = W_q x_i \quad query$$

$$k_i = W_k x_i \quad key$$

$$v_i = W_v x_i \quad value$$

Добавили матрицы  $W_q, W_k, W_v \in \mathbb{R}^{d \times d}$  для большей выразительности

$$y_i = \sum_j \text{softmax}\left(\frac{q_i^T k_j}{\sqrt{d}}\right) v_j$$

Делим на  $\sqrt{d}$ , чтобы лучше обучалось, градиенты софтмакса не были слишком маленькими

# Матричный вид Self-attention

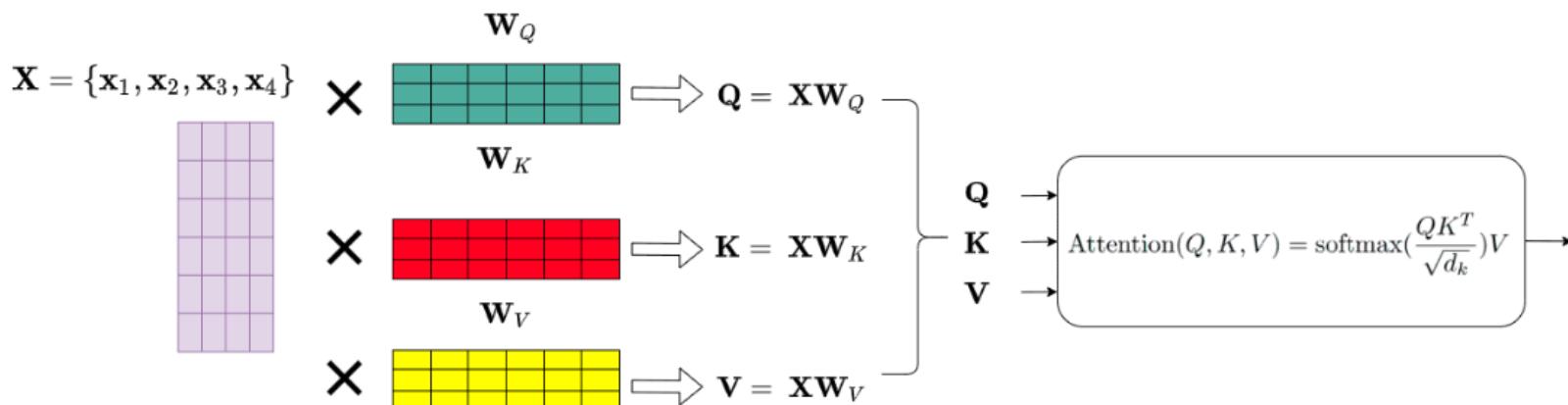
В матричном виде

$$Q = XW^Q$$

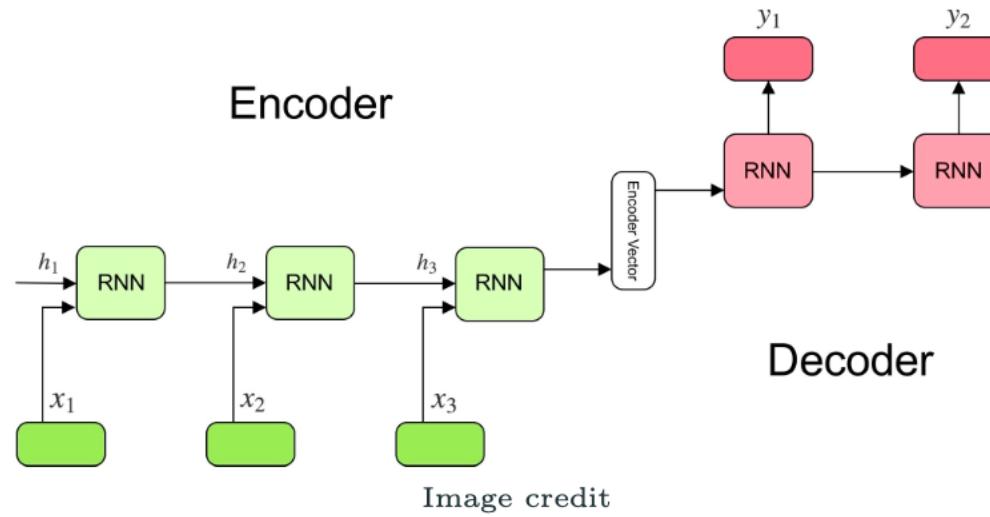
$$K = XW^K$$

$$V = XW^V$$

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



# Архитектура кодировщик-декодировщик

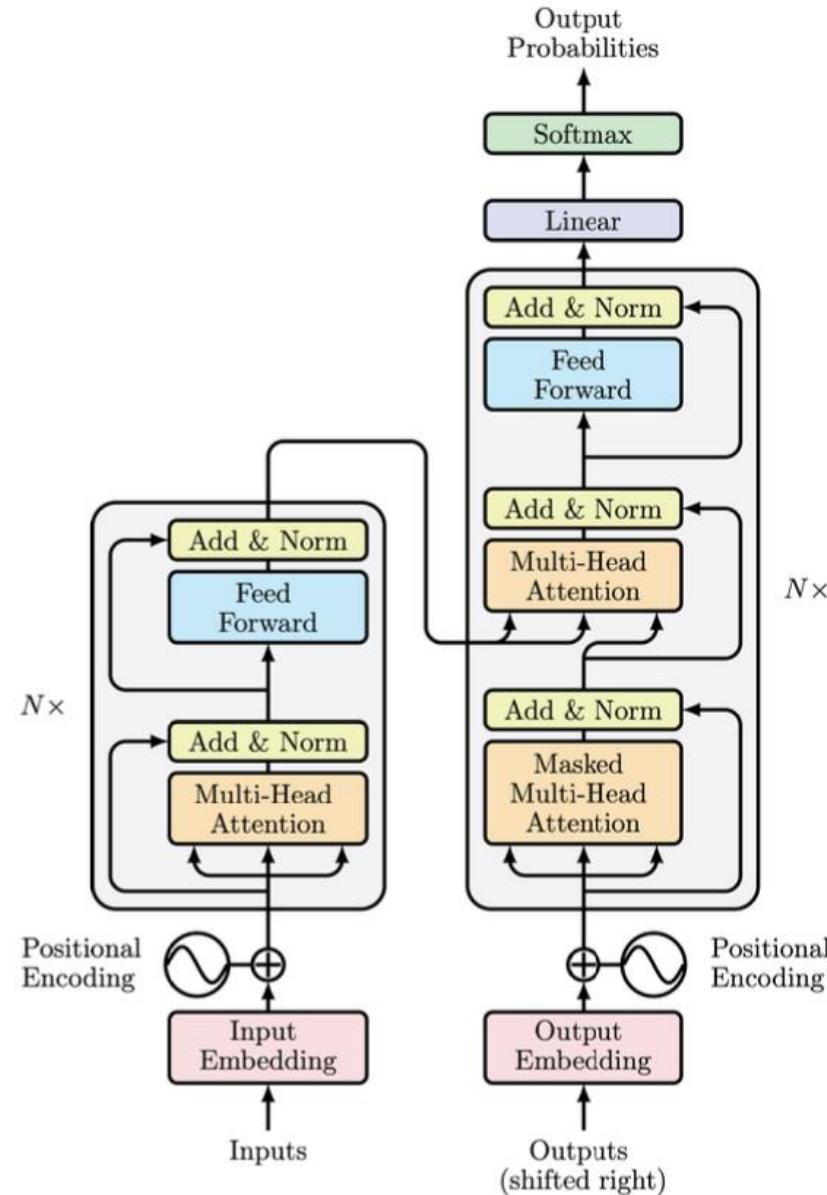


Encoder сворачивает входную последовательность в вектор контекста  $c$  (как правило, последнее скрытое состояние  $h_T$ ):

Decoder предсказывает следующий элемент на основе скрытого состояния  $s_t$ , предыдущего элемента  $y_{t-1}$  и вектора контекста  $c$

# Encoder-Decoder Transformer

- Vaswani A. et al. (2017)  
Attention is all you need.
- Encoder-Decoder  
архитектура
- Изначально придумали  
для machine translation
- Трансформеры стали  
универсальной  
архитектурой для  
обработки  
последовательностей



# Encoder-Only Transformer

Bert (2018) [arxiv](#)

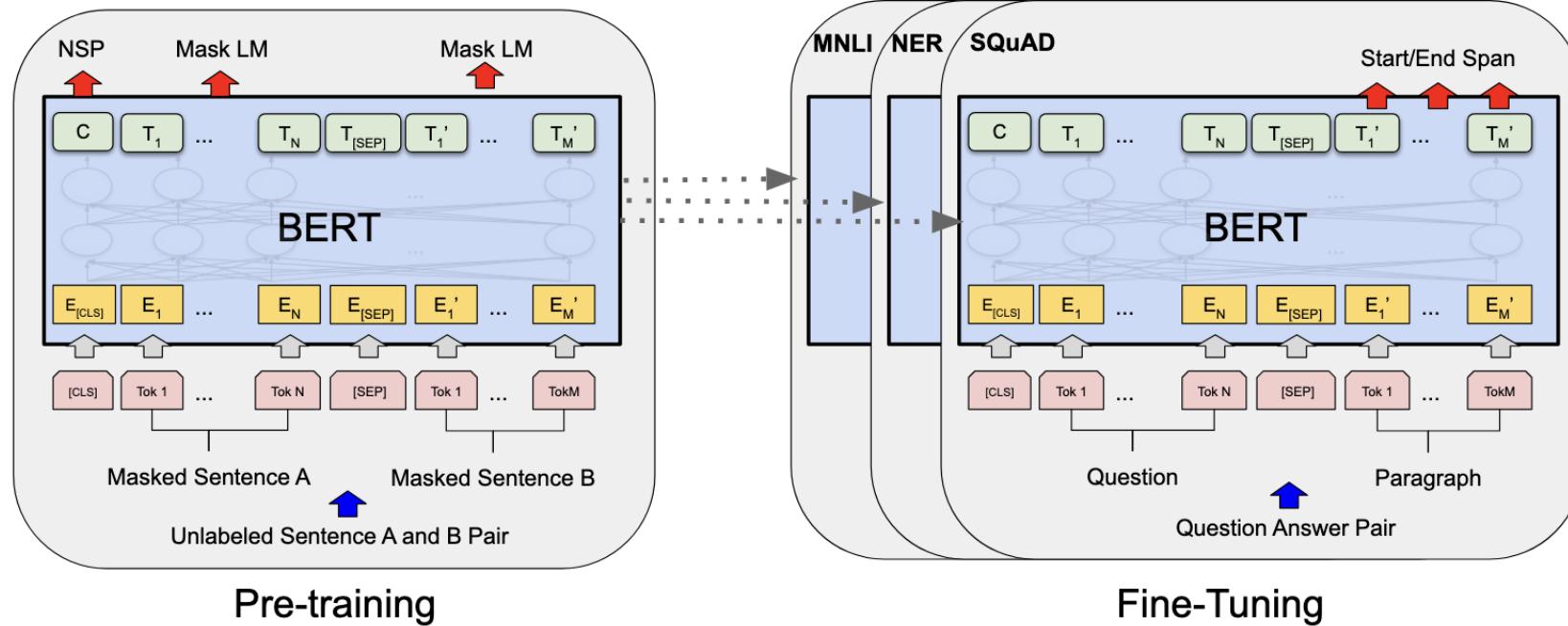


Figure 1: Overall pre-training and fine-tuning procedures for BERT. Apart from output layers, the same architectures are used in both pre-training and fine-tuning. The same pre-trained model parameters are used to initialize models for different down-stream tasks. During fine-tuning, all parameters are fine-tuned. [CLS] is a special symbol added in front of every input example, and [SEP] is a special separator token (e.g. separating questions/answers).

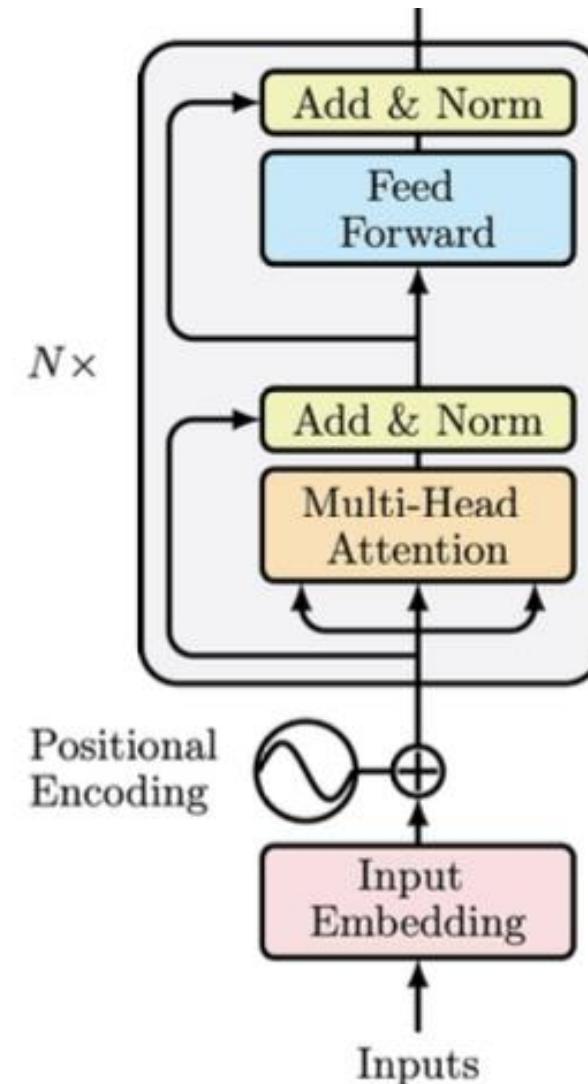
# Encoder-Only Transformer

Bert (2018) [arxiv](#)

Задача для предобучения (self-supervised):

- Masked Language Modeling

Последующий файнтюн на конкретную задачу



# Задачи с визуальной модальностью



# Задачи с визуальной модальностью

- ▶ Perception (Восприятие)
  - Image Classification
  - Zero-shot Image Classification
  - Representation Learning
  - Image Captioning
  - Visual Question Answering
- ▶ Generation (Генерация)
  - Text-To-Image
  - Image Inpainting

# Задачи с визуальной модальностью

- ▶ Perception (Восприятие)
  - Image Classification
  - Zero-shot Image Classification
  - Representation Learning
  - Image Captioning
  - Visual Question Answering
- ▶ Generation (Генерация)
  - Text-To-Image
  - Image Inpainting

# Vision Transformers



# Vision Transformers. Задачи

- ▶ Perception (Восприятие)
  - **Image Classification**
  - Zero-shot Image Classification
  - Representation Learning

# Ваши предложения?

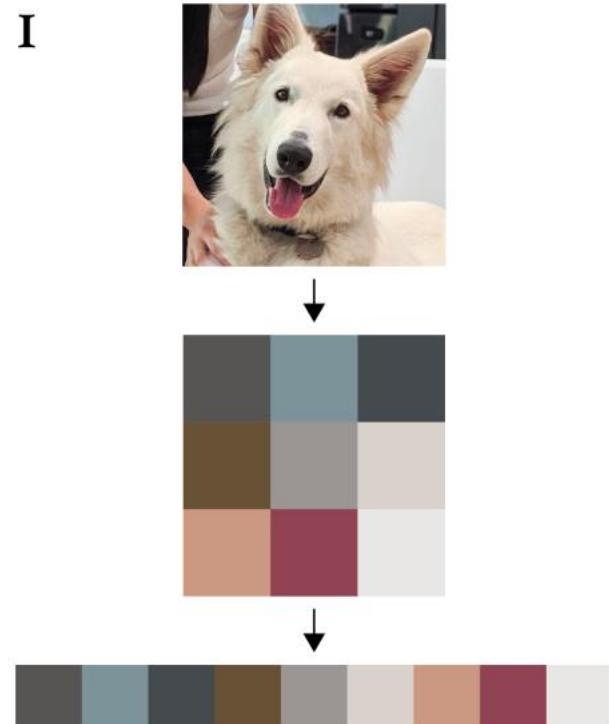
Как можно научить трансформер работать с  
визуальной модальностью?

# Трансформеры в компьютерном зрении

- ▶ Можно рассматривать изображение как последовательность пикселей

Проблемы:

- ▶ Большая размерность изображений
  - > Разбить изображение на куски (патчи) и рассматривать их как токены
- ▶ В отличие от сверточных сетей трансформер не учитывает структуру данных
  - > Взять больше данных, чтобы модель смогла выучить структуру



- ▶ Трансформеры - data hungry, но лучше масштабируются, чем CNN

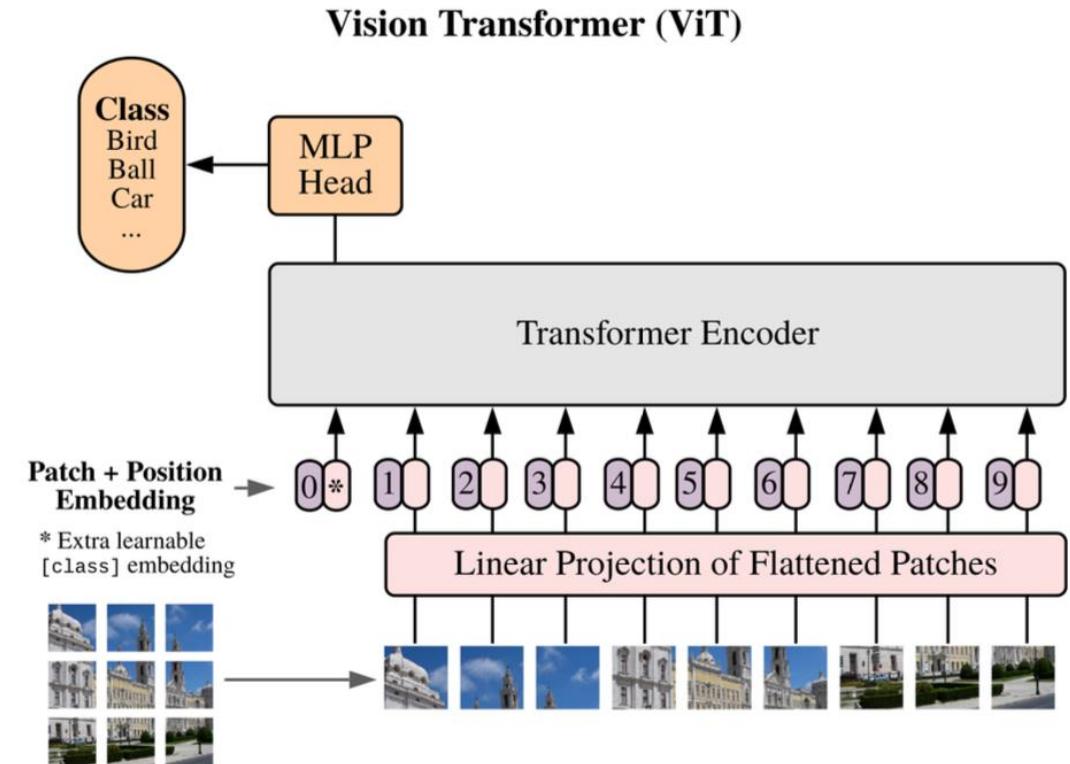
[https://cdn.openai.com/papers/Generative\\_Pretraining\\_from\\_Pixels\\_V2.pdf](https://cdn.openai.com/papers/Generative_Pretraining_from_Pixels_V2.pdf)

# Vision Transformer

- ▶ Используем стандартную трансформерную архитектуру с минимальными дополнениями.
- ▶ Основная проблема:

Части:

- ▶ Position Encoding
- ▶ Linear projection
- ▶ MLP Head



<https://arxiv.org/abs/2010.11929>

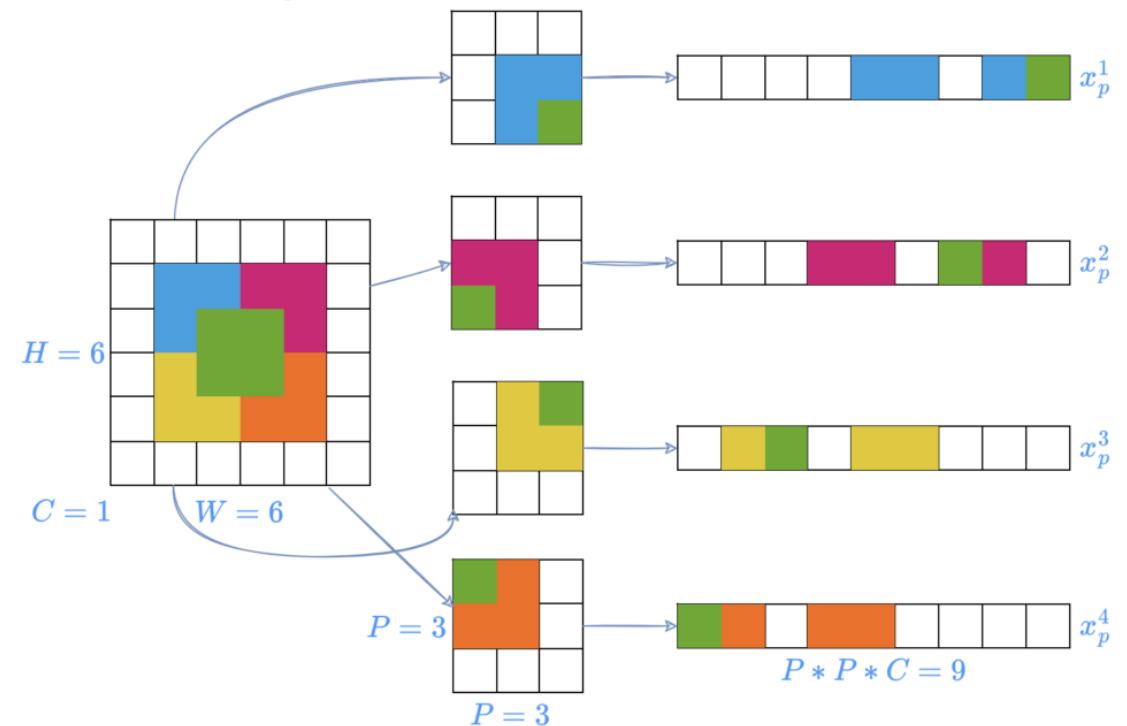
# Vision Transformer - Linear Projection

Дано картинка размерности ( $W \times H \times C$ )

Первый шаг:

- ▶ Делим на патчи размерности ( $P \times P \times C$ );
- ▶ Патчи переводим в вектора размерности ( $1 \times (P \times P \times C)$ ).

$N$  - количество патчей, тут оно равно 4  
 $H$  - высота изображения  $x$   
 $W$  - ширина изображения  $x$   
 $C$  - каналы изображения  $x$   
 $P$  - сторона патча  $x_p$



<https://blog.deepschool.ru/cv/vit-vision-transformer/>

# Vision Transformer - Linear Projection

Дано картинка размерности ( $W \times H \times C$ )

Первый шаг:

- ▶ Делим на патчи размерности ( $P \times P \times C$ );
- ▶ Патчи в вектора размерности ( $1 \times (P \times P \times C)$ ).

Второй шаг:

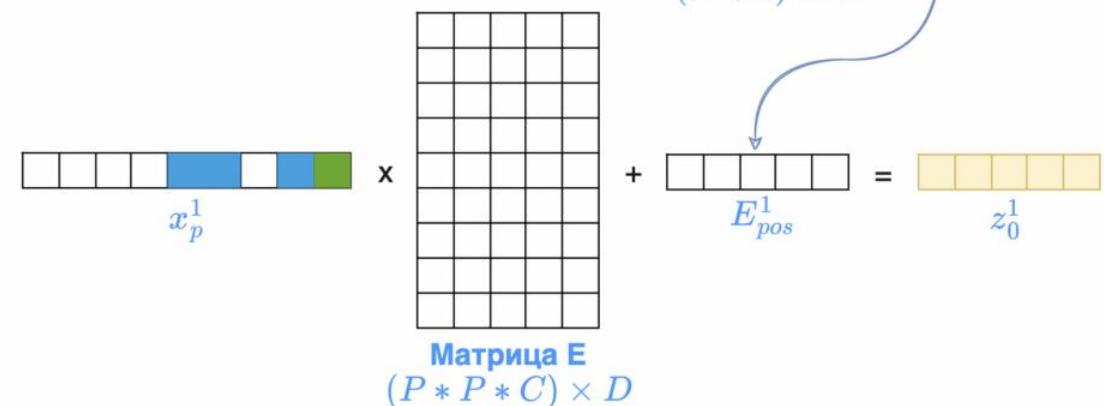
- ▶ ( $1 \times (P \times P \times C)$ ) переводим в вектора размерности ( $1 \times D$ ) с помощью матрицы  $E$ ;
- ▶ Складываем вектора размерности ( $1 \times D$ ) с векторами некоторой матрицы  $E_{pos}$ .

NB: [cls] вектор в  $E_{pos}$  – наследие BERT.

Матрицы  $E$  и  $E_{pos}$  – обучаемые параметры модели.

Цель  $E_{pos}$  – внести информацию о 2D структуре картинки.

$C$  - каналы изображения  $x$   
 $P$  - сторона патча  $x_p$   
 $N$  - количество патчей  $x_p$



<https://blog.deepschool.ru/cv/vit-vision-transformer/>

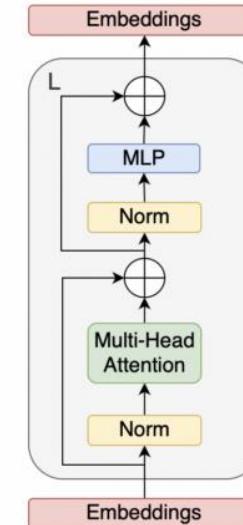
# Vision Transformer - Transformer Encoder

Дано картинка размерности ( $W \times H \times C$ )

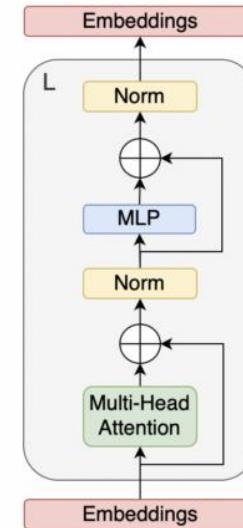
- ▶ Добавляем learnable position embedding
- ▶ Прогоняем через стандартный Transformer Encoder
- ▶ Поверх CLS токена - MLP Head для классификации

NB: Энкодер немного отличается от оригинального трансформера – так лучше работает.

Это трансформер энкодер из ViT



Это трансформер энкодер из "Attention is all you need"



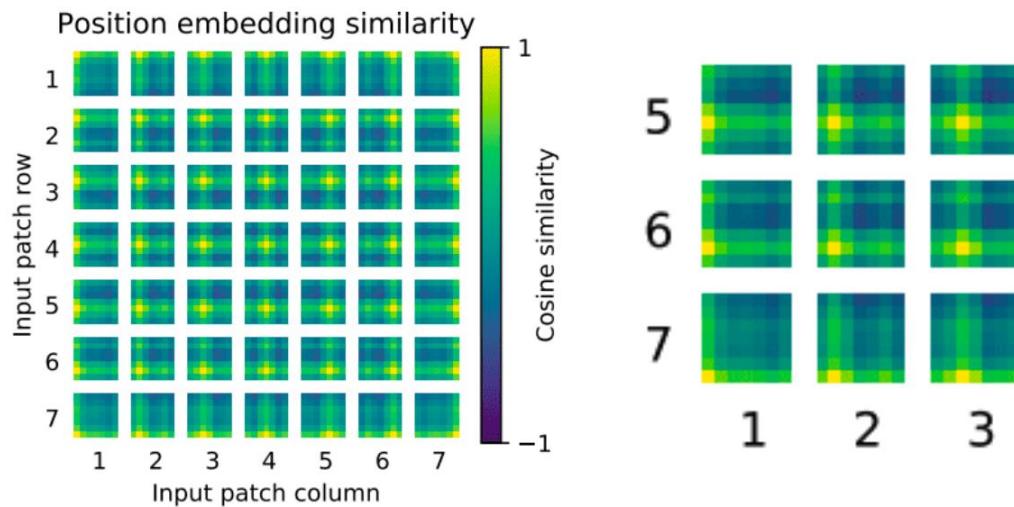
# Vision Transformer - Transformer Encoder

- ▶ Предобучение на большом датасете
  - ▶ JFT - 18k классов, 303M изображений
  - ▶ Для сравнения - в ImageNet-21k 14M изображений
- ▶ Finetune на конкретном задании.

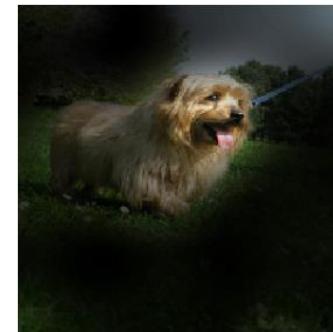
Размеры моделей:

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

# Vision Transformer – позиционные эмбединги, внимание



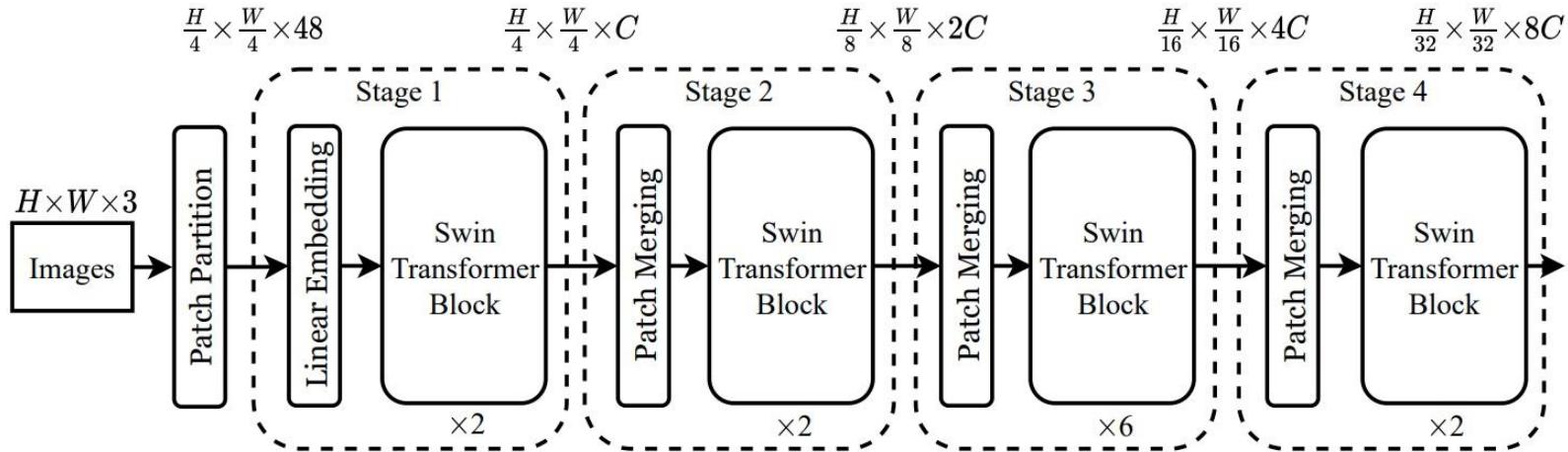
Input      Attention



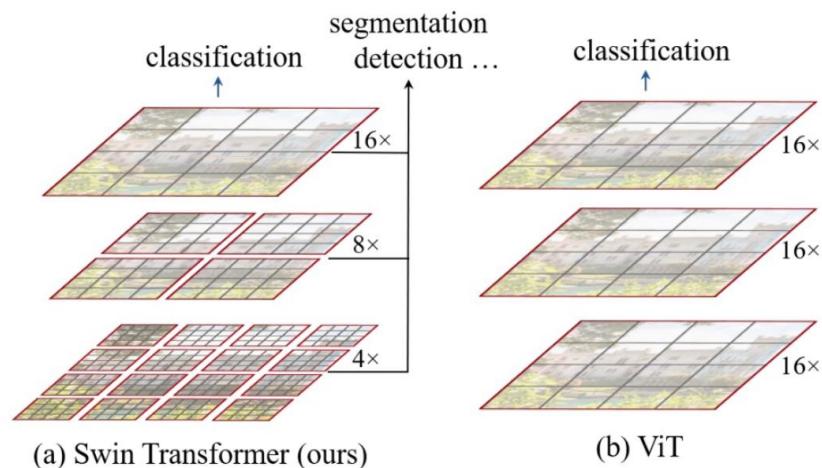
Модель учится кодировать расстояние в изображении так, что более близкие участки, как правило, имеют более похожие позиционные эмбединги

# Swin transformer

Проблема ViT - низкое разрешение, плохо подходит для задач, где важны мелкие детали (object detection, segmentation). Кроме того, он квадратичен по размеру картинки.



Объединение патчей на верхних уровнях



- ▶ Первый слой, такой же как ViT (патчи имеют размерность 4X4)
- ▶ Patch Merging занимается тем, что конкатенирует фичи соседних (в окне 2x2) токенов и понижает размерность
- ▶ Получается иерархическое представление изображения

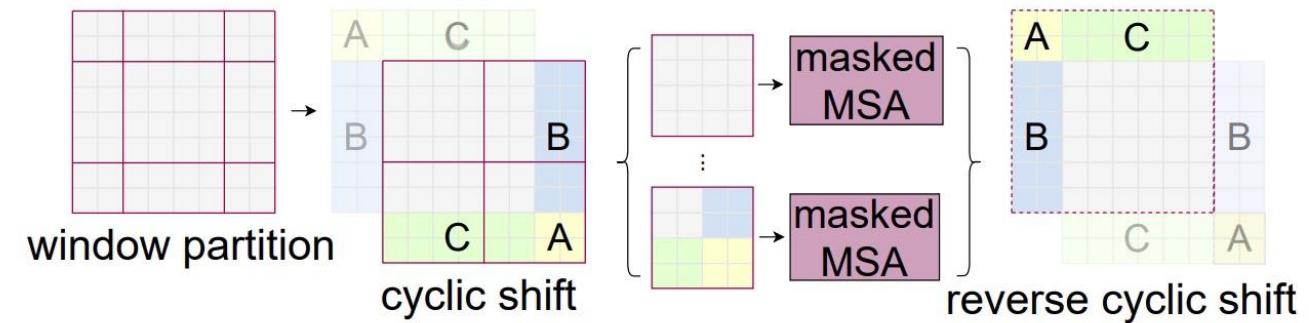
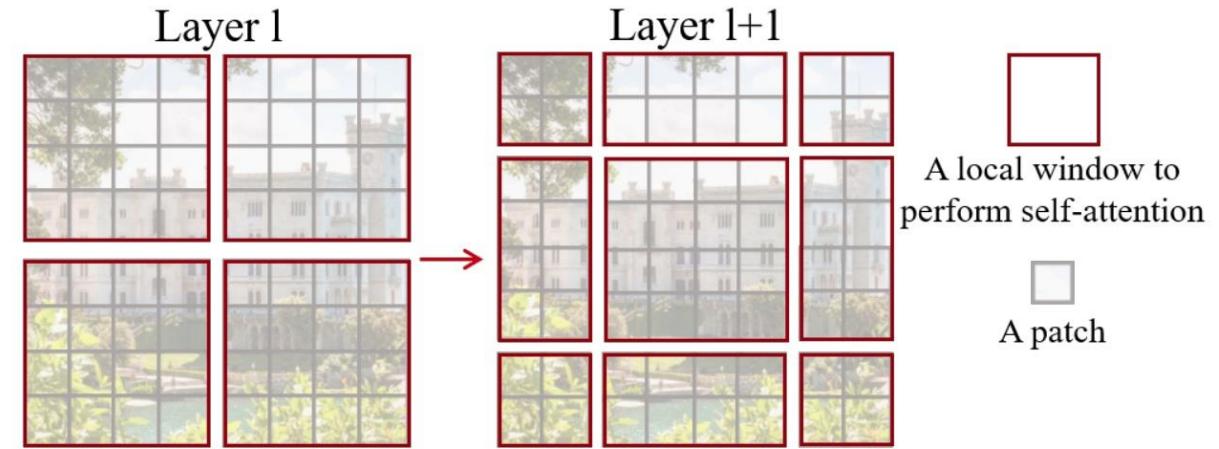
<https://arxiv.org/abs/2103.14030>

# Swin transformer - (Shifted) Window Multi-Head Attention

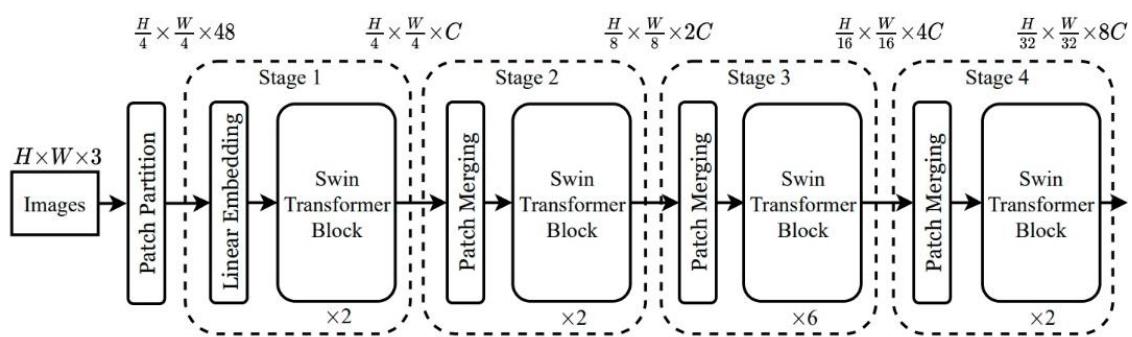
Проблемой Multi-Head Attention является его квадратичная сложность

- Local self-attention within window – даёт линейную сложность.
- Окна сдвинуты относительно друг друга в разных слоях: После каждого блока с Window Multi-Head Attention поставили аналогичный слой, со смещёнными по диагонали окнами Attention.

Циклическое перемещение изображения, чтобы не изменять патчи



# Swin transformer - результаты



Размеры моделей

- Swin-T:  $C = 96$ , layer numbers = {2, 2, 6, 2}
- Swin-S:  $C = 96$ , layer numbers = {2, 2, 18, 2}
- Swin-B:  $C = 128$ , layer numbers = {2, 2, 18, 2}
- Swin-L:  $C = 192$ , layer numbers = {2, 2, 18, 2}

Method	Backbone	val	test	#param.	FLOPs	FPS
		mIoU	score			
DANet [23]	ResNet-101	45.2	-	69M	1119G	15.2
DLab.v3+ [11]	ResNet-101	44.1	-	63M	1021G	16.0
ACNet [24]	ResNet-101	45.9	38.5	-	-	-
DNL [71]	ResNet-101	46.0	56.2	69M	1249G	14.8
OCRNet [73]	ResNet-101	45.3	56.0	56M	923G	19.3
UperNet [69]	ResNet-101	44.9	-	86M	1029G	20.1
OCRNet [73]	HRNet-w48	45.7	-	71M	664G	12.5
DLab.v3+ [11]	ResNeSt-101	46.9	55.1	66M	1051G	11.9
DLab.v3+ [11]	ResNeSt-200	48.4	-	88M	1381G	8.1
SETR [81]	T-Large <sup>‡</sup>	50.3	61.7	308M	-	-
UperNet	DeiT-S <sup>†</sup>	44.0	-	52M	1099G	16.2
UperNet	Swin-T	46.1	-	60M	945G	18.5
UperNet	Swin-S	49.3	-	81M	1038G	15.2
UperNet	Swin-B <sup>‡</sup>	51.6	-	121M	1841G	8.7
UperNet	Swin-L <sup>‡</sup>	<b>53.5</b>	<b>62.8</b>	234M	3230G	6.2

Table 3. Results of semantic segmentation on the ADE20K val and test set. <sup>†</sup> indicates additional deconvolution layers are used to produce hierarchical feature maps. <sup>‡</sup> indicates that the model is pre-trained on ImageNet-22K.

# Self-supervised learning



# Self Supervised Learning. Задачи

- ▶ Perception (Восприятие)
  - Image Classification
  - Zero-shot Image Classification
  - **Representation Learning**

Ваши предложения?

Какие модели и задачи мы уже знаем в NLP  
для Self Supervised Learning?

# Self-supervised learning

## NLP

- ▶ Word2vec
- ▶ Masked Language Modeling (BERT)
- ▶ Causal Language Modeling (GPT)
- ▶ Другие задачи

# Self-supervised learning

Если есть много неразмеченных данных, то можно:

- ▶ Придумываем вспомогательную задачу - **Pretext task**
- ▶ Обучаем модель на этой задаче: **Representation learning**
- ▶ На **малом** кол-ве данных делаем **Finetuning**

# Self-supervised learning

**Если есть много неразмеченных данных, то можно:**

- ▶ Придумываем вспомогательную задачу - **Pretext task**
- ▶ Обучаем модель на этой задаче: **Representation learning**
- ▶ На **малом** кол-ве данных делаем **Finetuning**

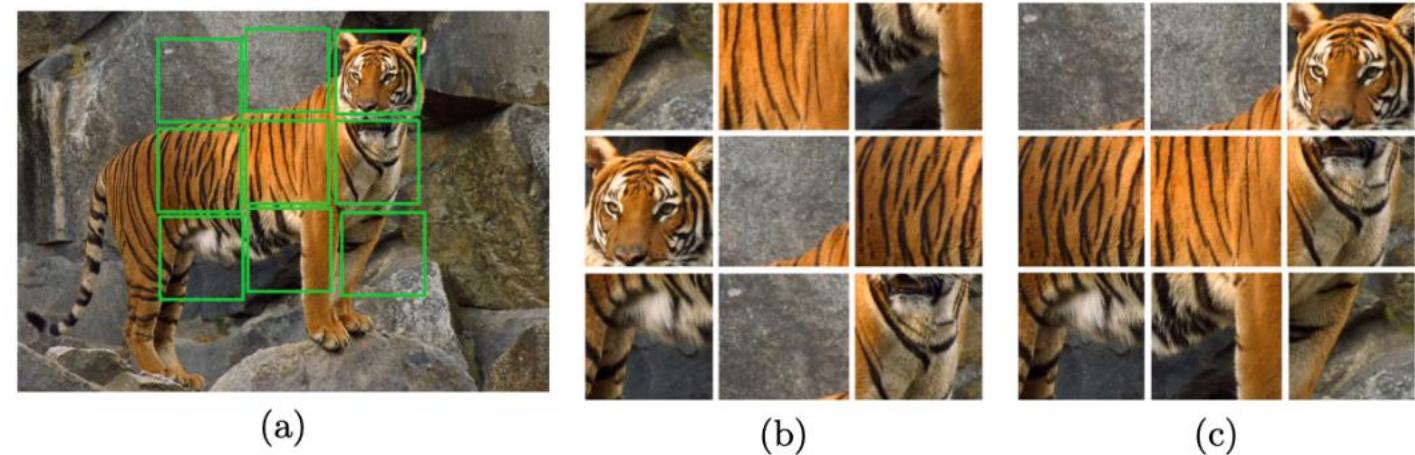
**Зачем нужен SSL, если все равно нужны размеченные  
данные для этапа Finetuning?**

# Jigsaw (разгадывание паззла) (2017)

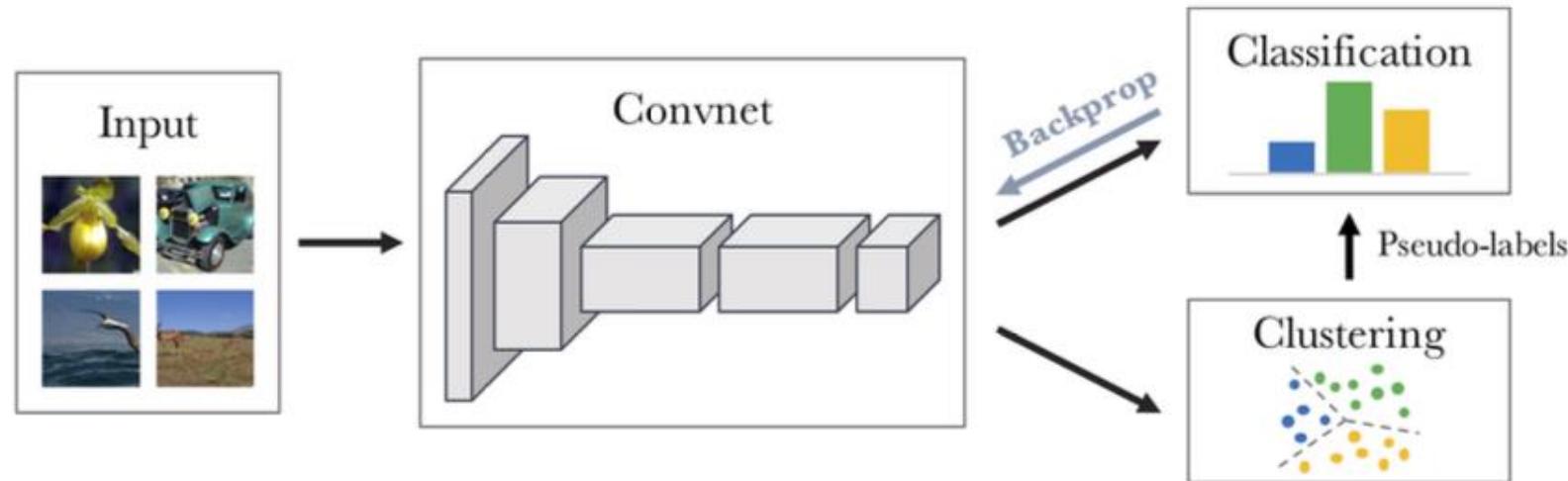
Сверточная нейронная сеть ,  
обученная решению  
головоломок в качестве  
претекстовой задачи (без  
ручной разметки)

Затем повторно использовать  
для решения задач  
классификации и обнаружения  
объектов

*Norooz M. et al. (2016) Unsupervised learning of visual representations by solving jigsaw puzzles.*



# DeepCluster (2018)

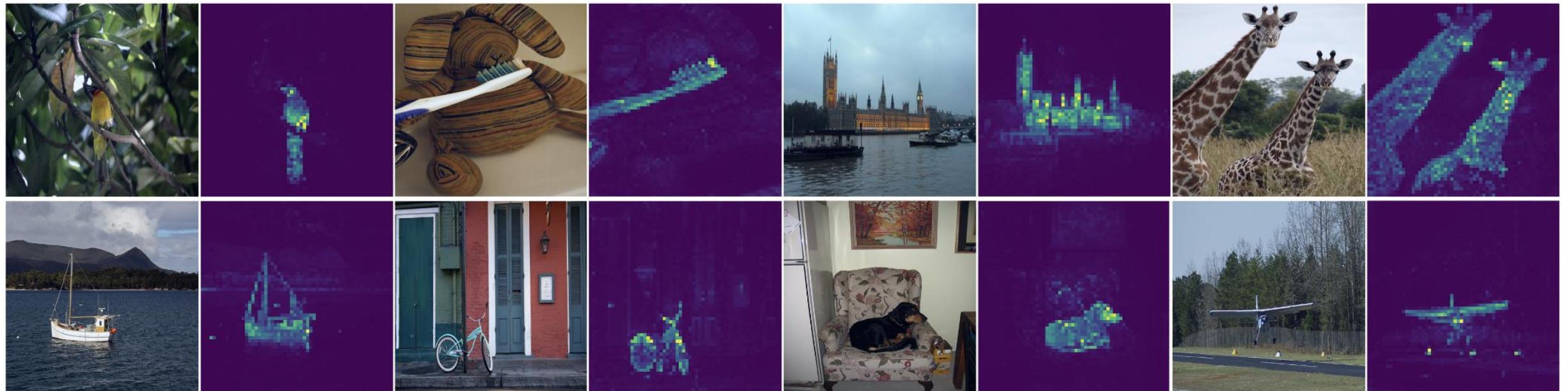


- ▶ Прогоняем через нейросеть данные, чтобы получить эмбеддинги
- ▶ Кластеризуем эмбеддинги с помощью K-means
- ▶ Обучаем модель предсказывать полученные кластера
- ▶ Повторяем эту процедуру по кругу

<https://arxiv.org/abs/1807.05520>

# Dino (2021 arxiv)

- ViT Backbone
- Pseudo-labels self-supervised

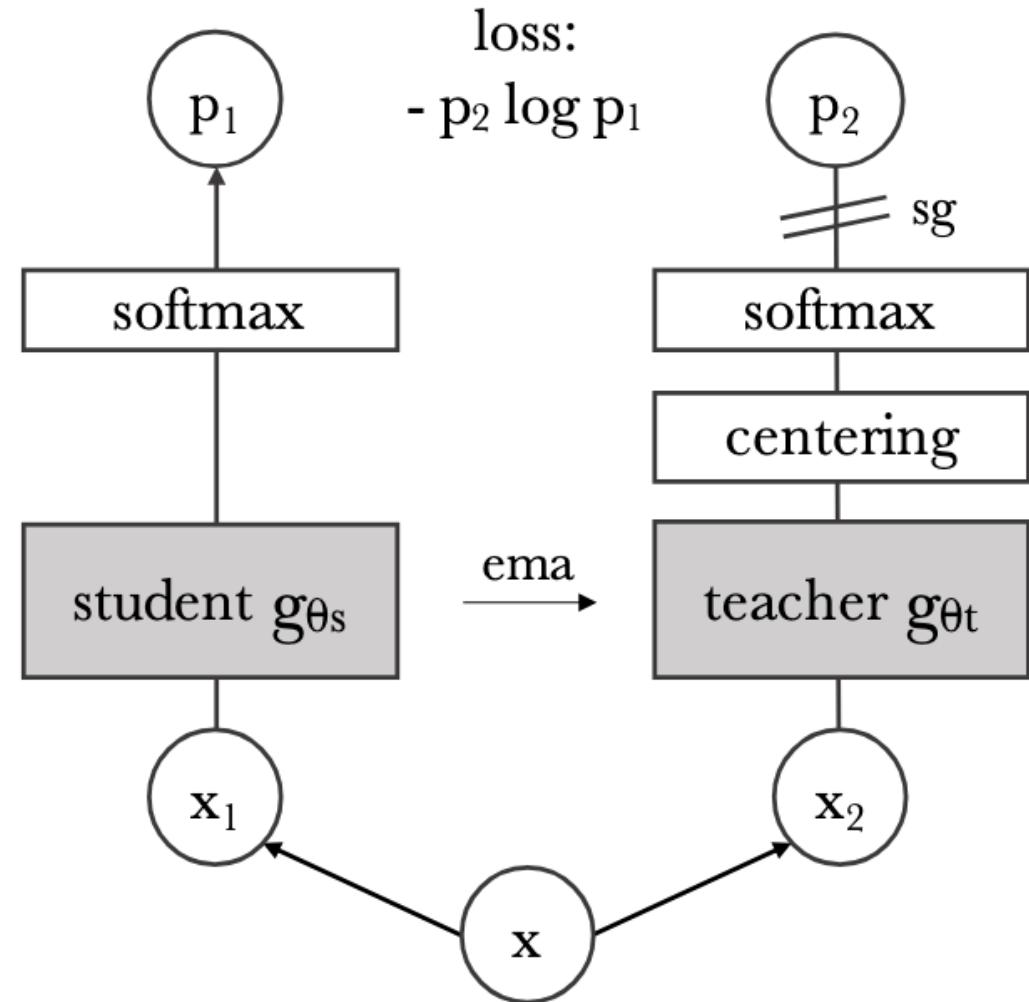


**Figure 1: Self-attention from a Vision Transformer with  $8 \times 8$  patches trained with no supervision.** We look at the self-attention of the [CLS] token on the heads of the last layer. This token is not attached to any label nor supervision. These maps show that the model automatically learns class-specific features leading to unsupervised object segmentations.

# Dino (2021 arxiv)

## ► Self-Distillation with No Labels

Figure 2: **Self-distillation with no labels.** We illustrate DINO in the case of one single pair of views ( $x_1, x_2$ ) for simplicity. The model passes two different random transformations of an input image to the student and teacher networks. Both networks have the same architecture but different parameters. The output of the teacher network is centered with a mean computed over the batch. Each network outputs a  $K$  dimensional feature that is normalized with a temperature softmax over the feature dimension. Their similarity is then measured with a cross-entropy loss. We apply a stop-gradient (sg) operator on the teacher to propagate gradients only through the student. The teacher parameters are updated with an exponential moving average (ema) of the student parameters.



# Dino (2021 arxiv)

---

**Algorithm 1** DINO PyTorch pseudocode w/o multi-crop.

---

```
# gs, gt: student and teacher networks
# C: center (K)
# tps, tpt: student and teacher temperatures
# l, m: network and center momentum rates
gt.params = gs.params
for x in loader: # load a minibatch x with n samples
    x1, x2 = augment(x), augment(x) # random views

    s1, s2 = gs(x1), gs(x2) # student output n-by-K
    t1, t2 = gt(x1), gt(x2) # teacher output n-by-K

    loss = H(t1, s2)/2 + H(t2, s1)/2
    loss.backward() # back-propagate

    # student, teacher and center updates
    update(gs) # SGD
    gt.params = l*gt.params + (1-l)*gs.params
    C = m*C + (1-m)*cat([t1, t2]).mean(dim=0)

def H(t, s):
    t = t.detach() # stop gradient
    s = softmax(s / tps, dim=1)
    t = softmax((t - C) / tpt, dim=1) # center + sharpen
    return - (t * log(s)).sum(dim=1).mean()
```

---

# Dino (2021 arxiv)

Table 2: **Linear and  $k$ -NN classification on ImageNet.** We report top-1 accuracy for linear and  $k$ -NN evaluations on the validation set of ImageNet for different self-supervised methods. We focus on ResNet-50 and ViT-small architectures, but also report the best results obtained across architectures. \* are run by us. We run the  $k$ -NN evaluation for models with official released weights. The throughput (im/s) is calculated on a NVIDIA V100 GPU with 128 samples per forward. Parameters (M) are of the feature extractor.

Method	Arch.	Param.	im/s	Linear	$k$ -NN
Supervised	ViT-S	21	1007	79.8	79.8
BYOL* [30]	ViT-S	21	1007	71.4	66.6
MoCov2* [15]	ViT-S	21	1007	72.7	64.4
SwAV* [10]	ViT-S	21	1007	73.5	66.3
DINO	ViT-S	21	1007	<b>77.0</b>	<b>74.5</b>

# Self Supervised Learning. Задачи

- ▶ Perception (Восприятие)
  - Image Classification
  - Zero-shot Image Classification
  - Representation Learning
  - **Masked Image Modeling**

# SimMIM (2021, arxiv)

**Архитектура:** SWIN

**Задача:** Masked Image Modeling

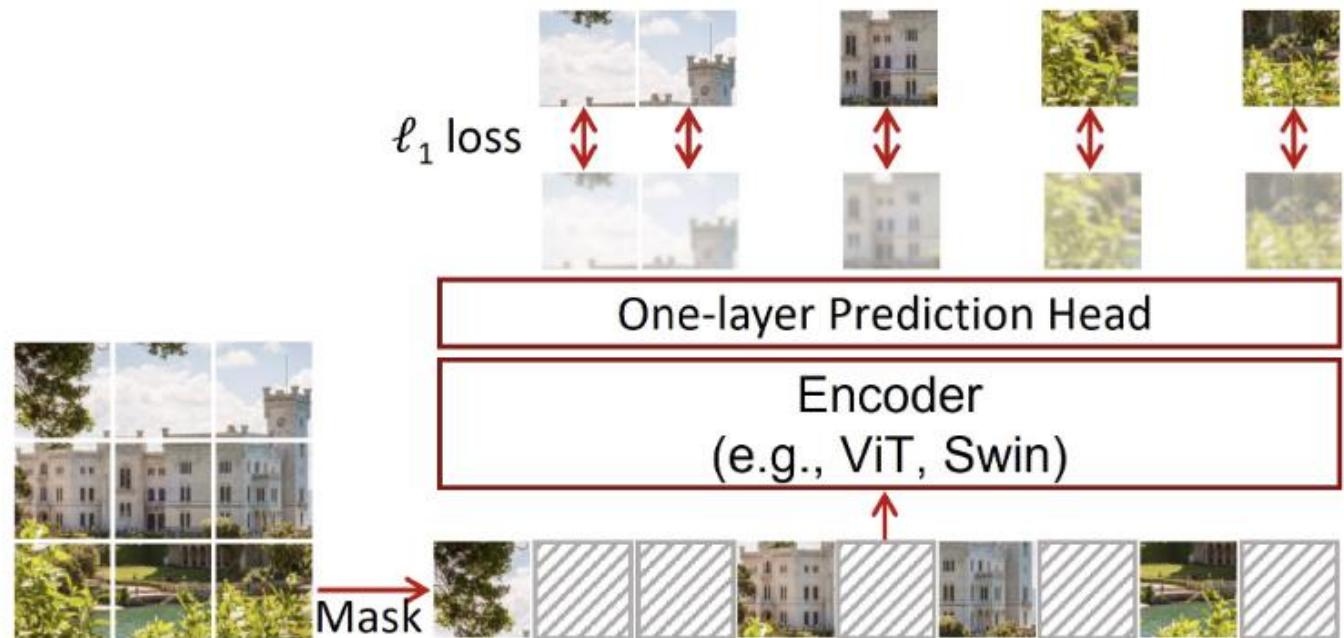


Figure 1. An illustration of our simple framework for masked language modeling, named *SimMIM*. It predicts raw pixel values of the randomly masked patches by a lightweight one-layer head, and performs learning using a simple  $\ell_1$  loss.

# SimMIM (2021, arxiv)

Methods	Input	Fine-tuning	Linear eval	Pre-training
	Size	Top-1 acc (%)	Top-1 acc (%)	costs
Sup. baseline [46]	$224^2$	81.8	-	-
DINO [5]	$224^2$	82.8	78.2	2.0×
MoCo v3 [9]	$224^2$	83.2	76.7	1.8×
ViT [15]	$384^2$	79.9	-	~4.0×
BEiT [1]	$224^2$	83.2	56.7	$1.5 \times^\dagger$
Ours	$224^2$	<b>83.8</b>	56.7	1.0×

Table 6. System-level comparison using ViT-B as the encoder. Training costs are counted in relative to our approach.  $^\dagger$  BEiT requires an additional stage to pre-train dVAE, which is not counted.

# Contrastive learning



# Contrastive learning. Задачи

- ▶ Perception (Восприятие)
  - Image Classification
  - Zero-shot Image Classification
  - **Representation Learning**

# Contrastive learning

Обучение представлений объектов, что:

- ▶ Похожие объекты класса имели похожие представления
- ▶ Непохожие объекты имели различные представления

# Triplet loss

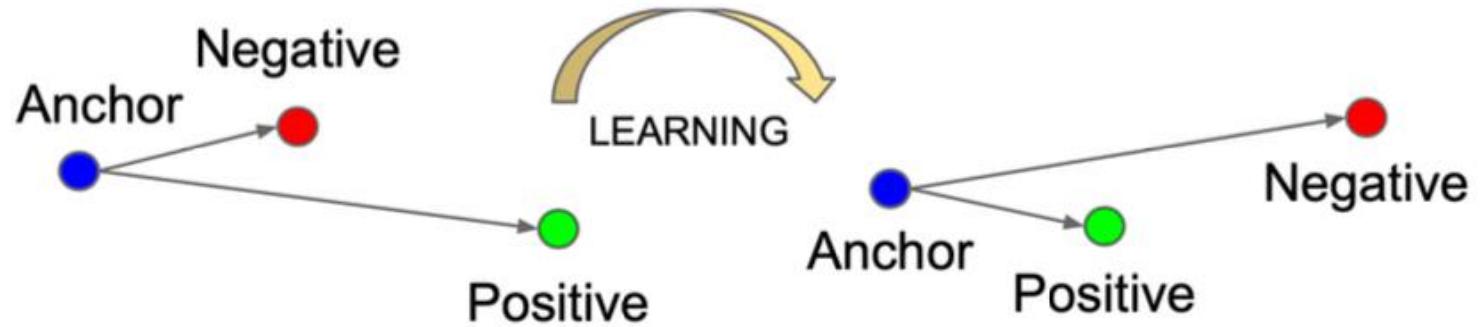


Image credit

$$L(x, x^+, x^-) = \max(0, \|f(x) - f(x^+)\|^2 - \|f(x) - f(x^-)\|^2 + m)$$

# Triplet loss

На практике используется редко.

## Проблемы:

- Эффективность сильно зависит от **качества и сложности** триплетов.
- Сложно подбирать триплеты

# InfoNCE - Information Noise-Contrastive Estimation

$$\mathcal{L}_i = -\log \frac{\exp(f(x_i)^T f(x_j))}{\sum_k \exp(f(x_i)^T f(x_k))}$$



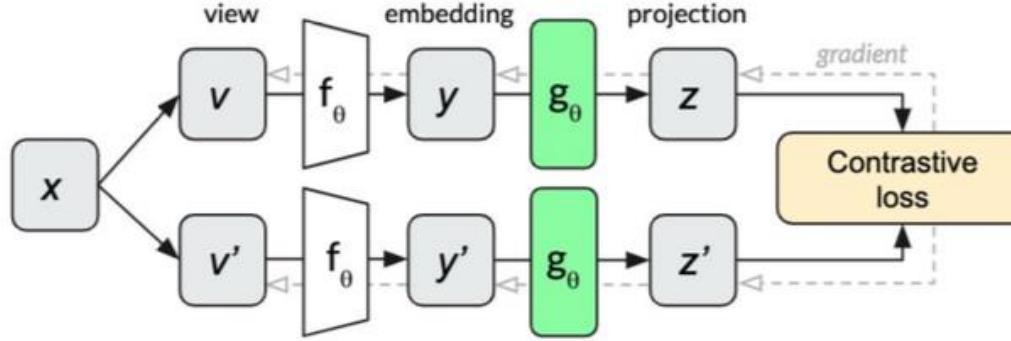
На что похожа формула?

# InfoNCE vs CrossEntropy

$$\mathcal{L}_{\text{InfoNCE}} = - \log \frac{\exp(\text{sim}(z_q, z_k^+)/\tau)}{\sum_{i=1}^K \exp(\text{sim}(z_q, z_k^i)/\tau)}$$

$$\mathcal{L}_{\text{CE}} = - \log \frac{\exp(s_y)}{\sum_{j=1}^C \exp(s_j)}$$

# Simple Framework for Contrastive Learning of Visual Representations



► Идея: аугментированные версии одного изображения должны быть похожи, разные изображения – непохожи

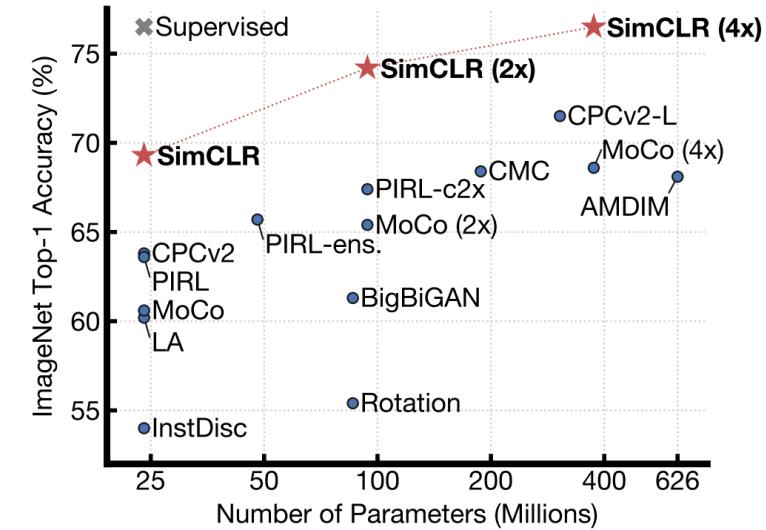
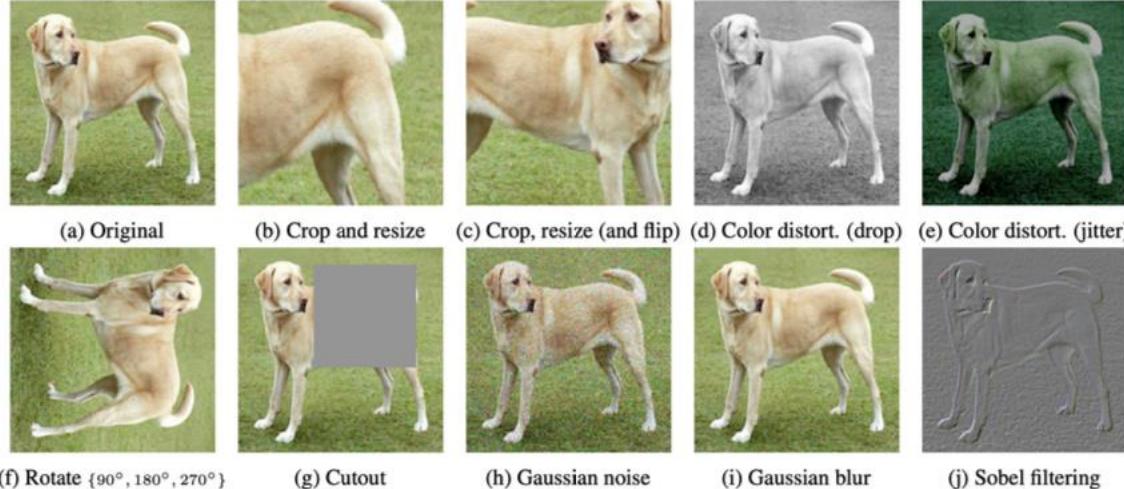


Figure 1. ImageNet Top-1 accuracy of linear classifiers trained on representations learned with different self-supervised methods (pretrained on ImageNet). Gray cross indicates supervised ResNet-50. Our method, SimCLR, is shown in bold.

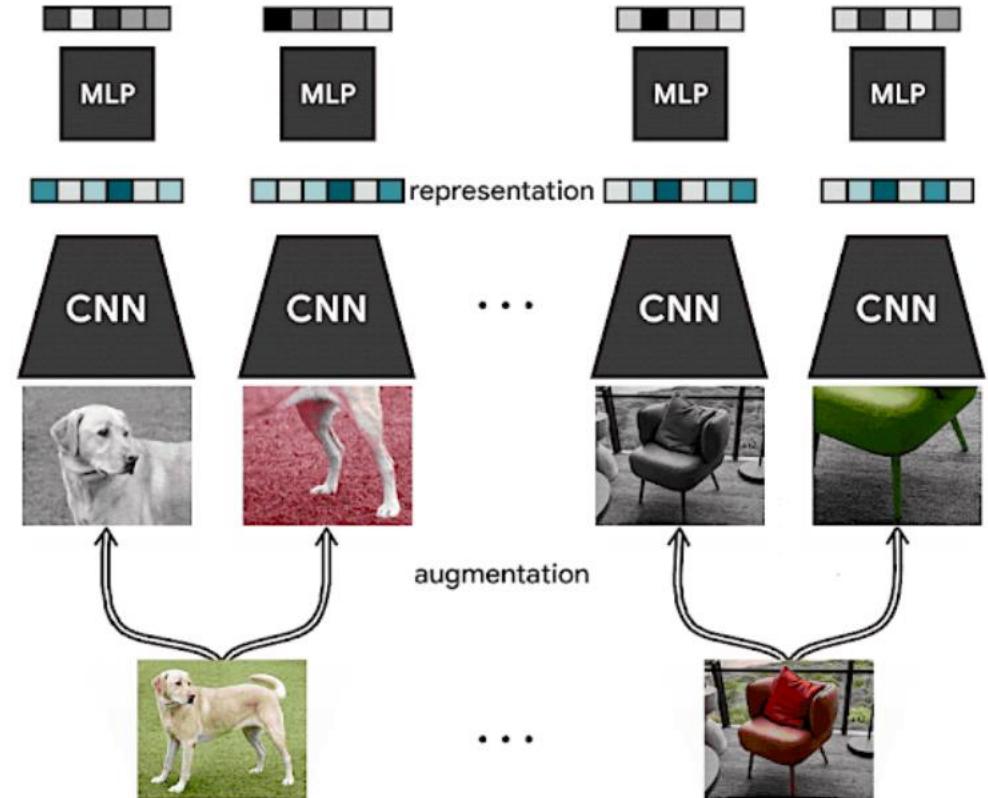
<https://arxiv.org/pdf/2002.05709>

# SimCLR - аугментация

## Аугментации данных



- ▶ Использовались разные аугментации
- ▶ На этом датасете лучше всего crop и color distortion



<https://research.google/blog/advancing-self-supervised-and-semi-supervised-learning-with-simclr/>

# SimCLR - Loss

NT-Xent (Normalized Temperature-scaled Cross Entropy) Loss

$$L_{i,j} = -\log \frac{\exp(\text{sim}(z_i, z_j)/\tau)}{\sum_{k=1}^{2N} 1_{[k \neq i]} \exp(\text{sim}(z_i, z_k)/\tau)}$$

$$\text{sim}(z_i, z_j) = \frac{z_i^T z_j}{\|z_i\| \|z_j\|}$$

- $N$  примеров в батче
- $2N$  примеров после аугментаций
- Для каждой позитивной пары есть  $2(N - 1)$  негативных пар

<https://research.google/blog/advancing-self-supervised-and-semi-supervised-learning-with-simclr/>

- ▶ **NT-Xent — это частный случай InfoNCE, в котором:**
- ▶ Используется **нормализованная косинусная похожесть** (через dot product двух **L2-нормированных** эмбеддингов),
- ▶ **Обязательно** применяется температурный коэффициент  $\tau$  для масштабирования logits (чтобы управлять "остротой" softmax).

# SimCLR - результаты

- ▶ Не все аугментации одинаково полезны
- ▶ Композиция нескольких аугментаций помогает
- ▶ Нужны более сильные аугментации по сравнению с supervised learning
- ▶ Projection head важна для хороших результатов
- ▶ Нужны больший размер батча и более длительное обучение по сравнению с supervised learning

# Multimodal Contrastive learning



# Multimodal Contrastive learning. Задачи

- ▶ Perception (Восприятие)
  - Image Classification
  - **Zero-shot Image Classification**
  - **Representation Learning**

# Что такое zero-shot?

- ▶ Модель явно не обучалась решать задачу, но умеет ее решать без дополнительного файнтюнинга.
- ▶ Мы с этим сталкивались в контексте LLM.

# Что такое модальности в данном контексте?

Данные разной природы:

- ▶ Текст
- ▶ Изображения
- ▶ Аудио
- ▶ Видео

# Contrastive learning

Обучение представлений объектов, что:

- ▶ Похожие объекты класса имели похожие представления
- ▶ Непохожие объекты имели различные представления

# Ваши предложения?

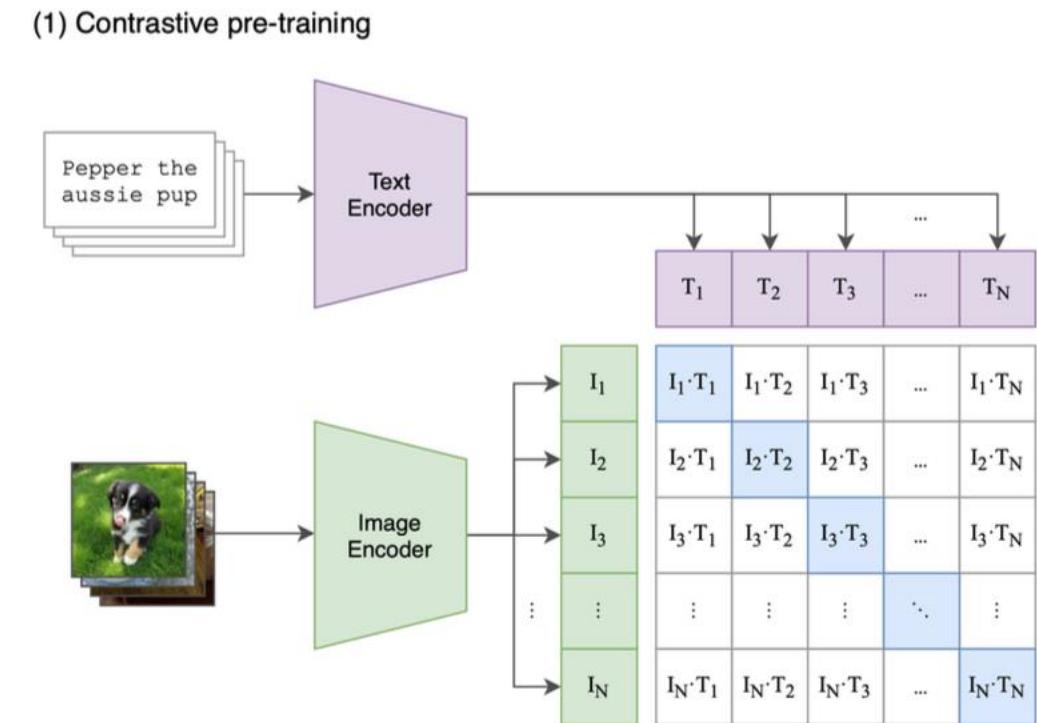
- ▶ Как научить модель работать одновременно с несколькими модальностями?
- ▶ Чем будет отличаться от унимодального contrastive learning?

# Contrastive Language-Image Pre-training (CLIP) (2021 arxiv)

Мультимодальная модель - текст + изображения

Преобразует изображения и текст в общее пространство эмбеддингов, в котором их можно сравнивать между собой

Обучена на 400М пар (картинка, подпись) из интернета



Для получения эмбеддингов использовался  
ViT для картинок  
GPT-трансформер для текста

Если произведение большое – два вектора близки

# Contrastive Language-Image Pre-training (CLIP) (2021 [arxiv](#)) **Loss**

$$\mathcal{L}_{\text{CLIP}} = \frac{1}{2} (\mathcal{L}_{\text{image} \rightarrow \text{text}} + \mathcal{L}_{\text{text} \rightarrow \text{image}})$$

$$\mathcal{L}_{\text{image} \rightarrow \text{text}} = -\frac{1}{N} \sum_{i=1}^N \log \frac{\exp(\text{sim}(z_i^I, z_i^T)/\tau)}{\sum_{j=1}^N \exp(\text{sim}(z_i^I, z_j^T)/\tau)}$$

$$\mathcal{L}_{\text{text} \rightarrow \text{image}} = -\frac{1}{N} \sum_{i=1}^N \log \frac{\exp(\text{sim}(z_i^T, z_i^I)/\tau)}{\sum_{j=1}^N \exp(\text{sim}(z_i^T, z_j^I)/\tau)}$$

$\tau$  — **обучаемый** температурный параметр

# Contrastive Language-Image Pre-training (CLIP) (2021 arxiv)

```
# image_encoder - ResNet or Vision Transformer
# text_encoder - CBOW or Text Transformer
# I[n, h, w, c] - minibatch of aligned images
# T[n, l]       - minibatch of aligned texts
# W_i[d_i, d_e] - learned proj of image to embed
# W_t[d_t, d_e] - learned proj of text to embed
# t             - learned temperature parameter

# extract feature representations of each modality
I_f = image_encoder(I) #[n, d_i]
T_f = text_encoder(T)  #[n, d_t]

# joint multimodal embedding [n, d_e]
I_e = l2_normalize(np.dot(I_f, W_i), axis=1)
T_e = l2_normalize(np.dot(T_f, W_t), axis=1)

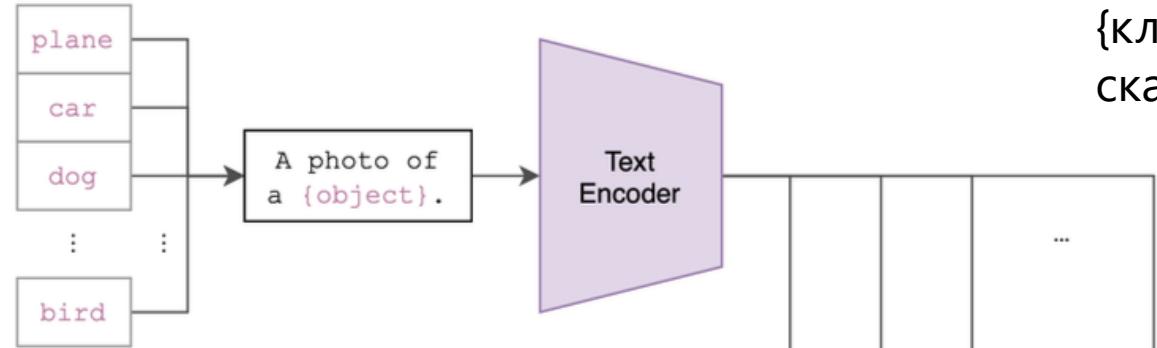
# scaled pairwise cosine similarities [n, n]
logits = np.dot(I_e, T_e.T) * np.exp(t)

# symmetric loss function
labels = np.arange(n)
loss_i = cross_entropy_loss(logits, labels, axis=0)
loss_t = cross_entropy_loss(logits, labels, axis=1)
loss   = (loss_i + loss_t)/2
```

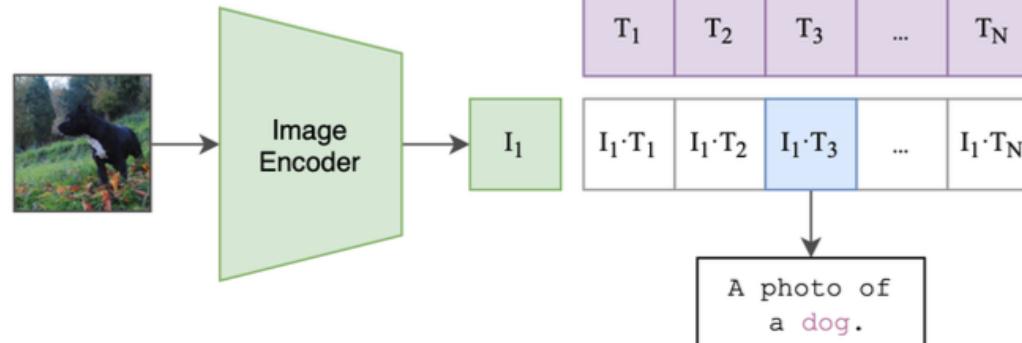
*Figure 3.* Numpy-like pseudocode for the core of an implementation of CLIP.

# Contrastive Language-Image Pre-training (CLIP) (2021 arxiv)

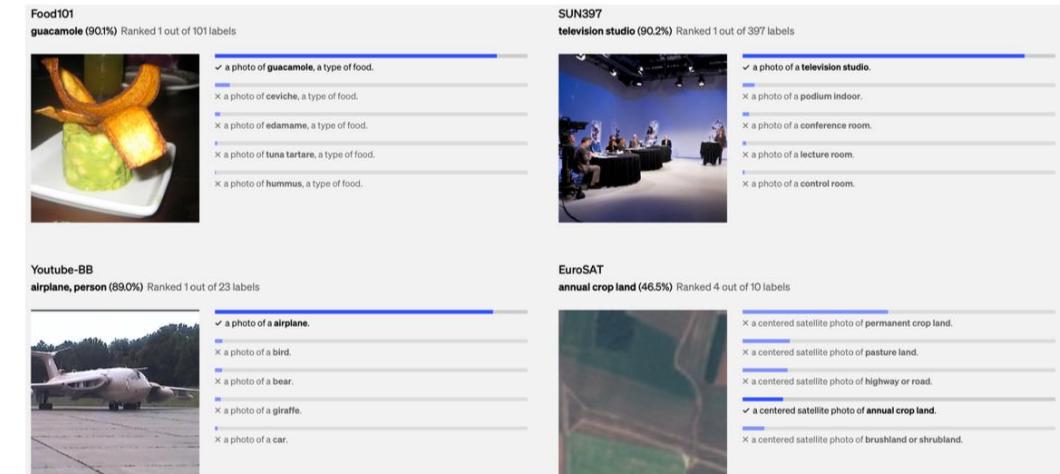
## (2) Create dataset classifier from label text



## (3) Use for zero-shot prediction

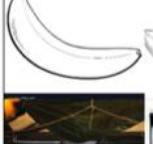
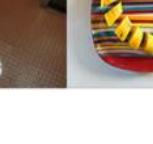
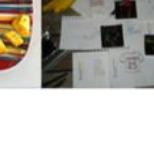


Чтобы классифицировать изображение, сравниваем эмбединг изображения с эмбедингом текста «Фотография {класса}.», и выдаём {класс}, дающий наибольшее скалярное произведение.



# Contrastive Language-Image Pre-training (CLIP) (2021 arxiv)

Более устойчивая модель, так как не подгонялась под конкретные классы в конкретном датасете

	Dataset Examples	ImageNet	Zero-Shot	$\Delta$ Score
		ResNet101	CLIP	
ImageNet	     	76.2	76.2	0%
ImageNetV2	     	64.3	70.1	+5.8%
ImageNet-R	     	37.7	88.9	+51.2%
ObjectNet	     	32.6	72.3	+39.7%
ImageNet Sketch	     	25.2	60.2	+35.0%
ImageNet-A	     	2.7	77.1	+74.4%

# SigLip (2023, arxiv)

Заменили Softmax на log\_sigmoid.

- ▶ Улучшилось масштабирование: не нужно синхронизировать между GPU знаменатель softmaxа

Method	Image Encoder		ImageNet-1k				COCO R@1	
	ViT size	# Patches	Validation	v2	ReaL	ObjectNet	I → T	T → I
CLIP	B	196	68.3	61.9	-	55.3	52.4	33.1
OpenCLIP	B	196	70.2	62.3	-	56.0	59.4	42.3
EVA-CLIP	B	196	74.7	67.0	-	62.3	58.7	42.2
SigLIP	B	196	<b>76.2</b>	<b>69.6</b>	82.8	<b>70.7</b>	<b>64.4</b>	<b>47.2</b>
SigLIP	B	256	76.7	70.0	83.1	71.3	65.1	47.4
SigLIP	B	576	78.6	72.1	84.5	73.8	67.5	49.7
SigLIP	B	1024	<b>79.2</b>	<b>73.0</b>	<b>84.9</b>	<b>74.7</b>	<b>67.6</b>	<b>50.4</b>

---

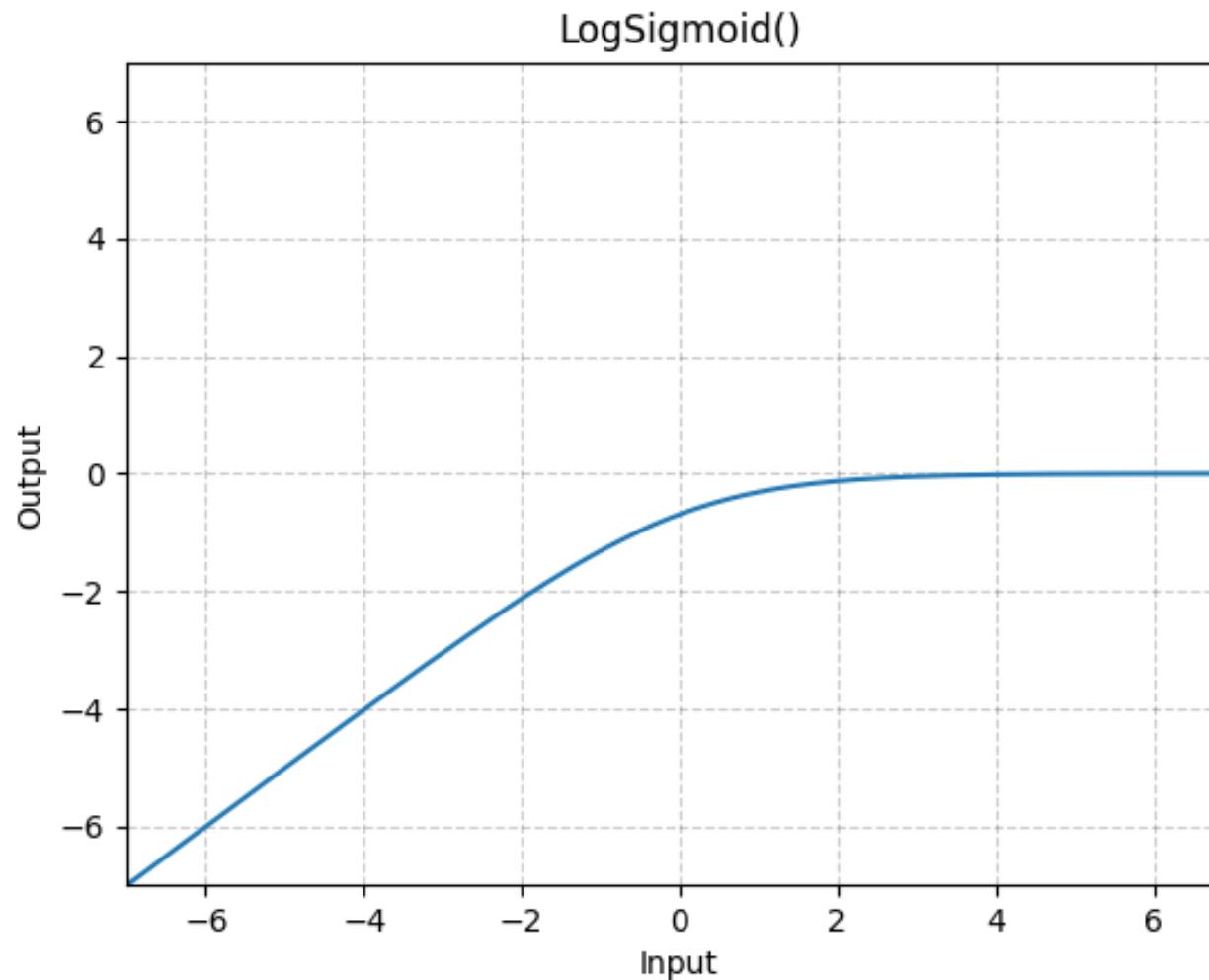
## Algorithm 1 Sigmoid loss pseudo-implementation.

---

```
1 # img_emb          : image model embedding [n, dim]
2 # txt_emb          : text model embedding [n, dim]
3 # t_prime, b       : learnable temperature and bias
4 # n                : mini-batch size
5
6 t = exp(t_prime)
7 zimg = l2_normalize(img_emb)
8 ztxt = l2_normalize(txt_emb)
9 logits = dot(zimg, ztxt.T) * t + b
10 labels = 2 * eye(n) - ones(n) # -1 with diagonal 1
11 l = -sum(log_sigmoid(labels * logits)) / n
```

---

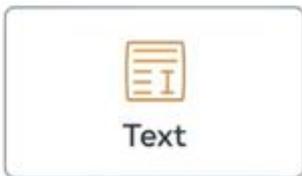
# SigLip (2023, arxiv)



# ImageBind (2023 [arxiv](#))



Depth



Text



Image/Video



Heat map



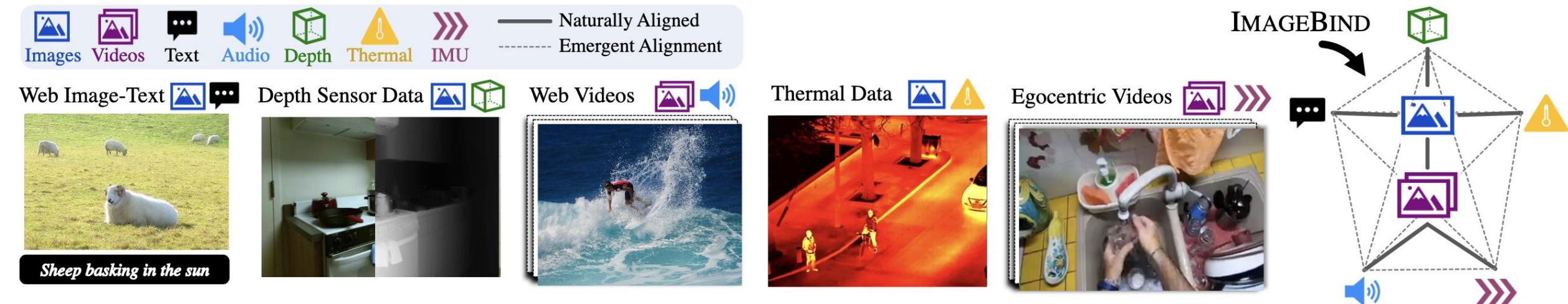
Audio



IMU

 Meta AI

# ImageBind (2023 arxiv)



**Figure 2. IMAGEBIND overview.** Different modalities occur naturally aligned in different data sources, for instance images+text and video+audio in web data, depth or thermal information with images, IMU data in videos captured with egocentric cameras, *etc.* IMAGEBIND links all these modalities in a common embedding space, enabling new emergent alignments and capabilities.

# ImageBind (2023 arxiv)

	IN1K	P365	K400	MSR-VTT	NYU-D	SUN-D	AS-A	VGGS	ESC	LLVIP	Ego4D
Random	0.1	0.27	0.25	0.1	10.0	5.26	0.62	0.32	2.75	50.0	0.9
IMAGEBIND	77.7	45.4	50.0	36.1	54.0	35.1	17.6	27.8	66.9	63.4	25.0
Text Paired	-	-	-	-	41.9*	25.4*	28.4 <sup>†</sup> [27]	-	68.6 <sup>†</sup> [27]	-	-
Absolute SOTA	91.0 [82]	60.7 [67]	89.9 [80]	57.7 [79]	76.7 [21]	64.9 [21]	49.6 [39]	52.5 [36]	97.0 [9]	-	-

**Table 2. Emergent zero-shot classification** of IMAGEBIND using text prompts highlighted in blue. IMAGEBIND aligns images with text, depth, audio, thermal and IMU modalities. The resulting embedding space can associate text embeddings with the non-image modalities, and leads to strong emergent zero-shot classification. We show strong performance even on non-visual modalities such as audio and IMU. We compare to ‘Text Paired’ baselines wherever possible, which trains with paired text data for that modality. \*We use the OpenCLIP ViT-H [30] on depth rendered as grayscale images. <sup>†</sup>[27] that uses AS class names as supervision during training, and hence is not “zero-shot”. Overall, IMAGEBIND shows strong emergent zero-shot performance, even compared to such upper bounds. We also report the absolute state-of-the-art (SOTA) on each dataset for reference, which typically uses additional supervision, model ensembles *etc.* We report the top-1 classification accuracy for all datasets except MSR-VTT (Recall@1) and Audioset Audio-only (mAP).

# ИТОГО



# План

Вспоминаем трансформеры

- ▶ Внимание
- ▶ Encoder-Decoder Transformer
- ▶ Encoder-only Transformer

Архитектуры:

- ▶ Vision Transformer
- ▶ Swin transformer

Методы обучения:

- ▶ Self-supervised Learning
- ▶ Contrastive Learning
- ▶ Multimodal Contrastive Learning