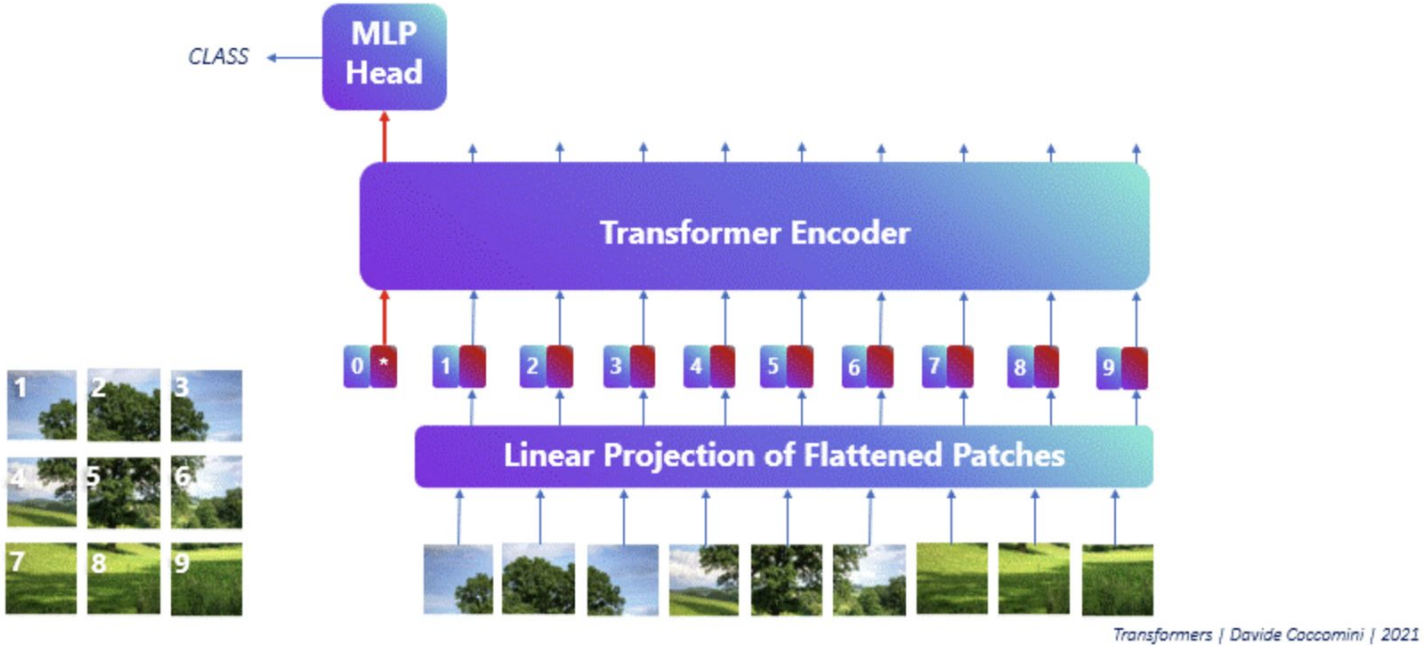
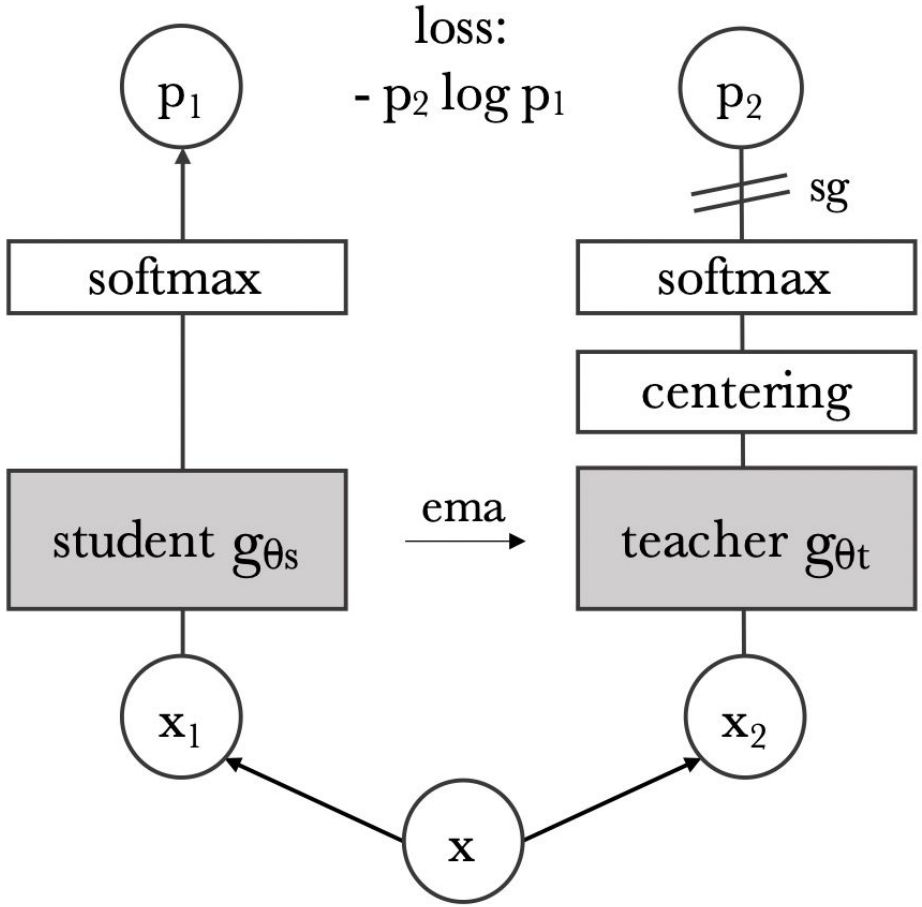
Are Large-scale Datasets Necessary for Self-Supervised Pre-training?

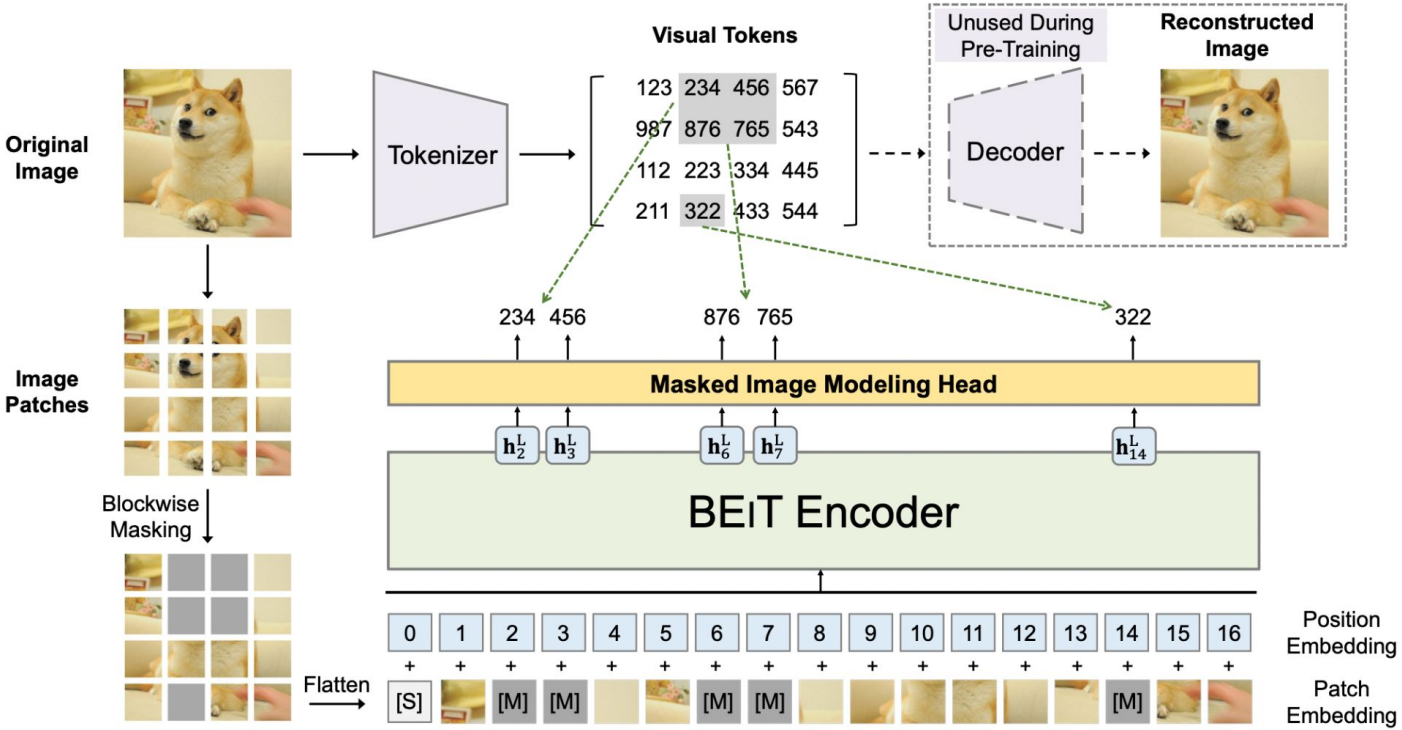
Vision Transformers: напоминание



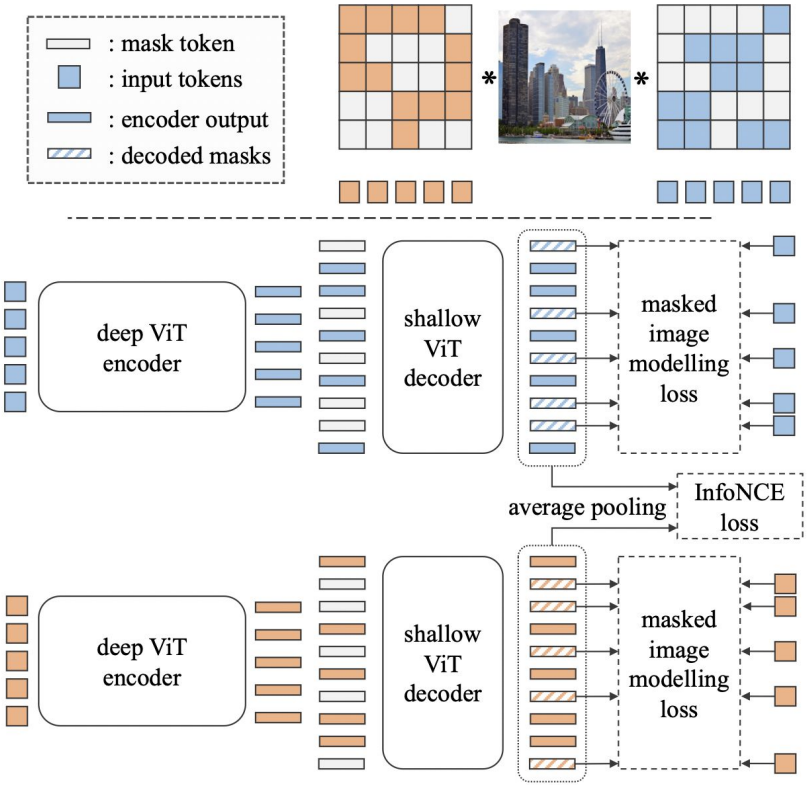
DINO: напоминание



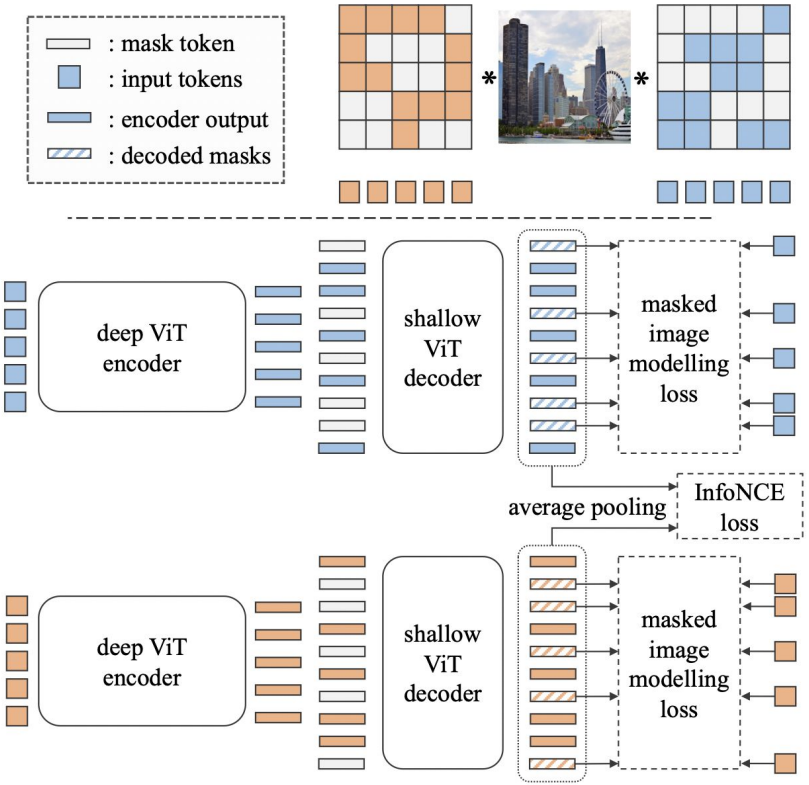
Denoising Autoencoders: BEiT



Denoising Autoencoders: SplitMask



Denoising Autoencoders: SplitMask



Основные отличия от BEiT:

1. На каждом шаге используются все патчи для каждой из картинок
2. Через encoder пропускаются только входные токены (без масок)
3. Есть декодер
4. Есть дополнительный контрастив лосс на все изображение

Анализ: выбор токенизатора

В BEiT сначала отдельно обучается discrete VAE, хочется полностью отказаться от больших датасетов для претрейнинга. Пусть есть V единичных векторов e_i. Для патча x, мы выбираем токен t как:

Авторы предлагают несколько способов выбора e_i: $t = \mathbf{argmax}_{i \in \{1,\dots,V\}} \mathbf{x}^\top \mathbf{e}_i$

1. (random projection) Сэмплировать случайные векторы
2. (random patches) Сэмплировать случайные патчи из изображений
3. (k-means) Кластеризовать патчи из изображений и использовать центроиды кластеров

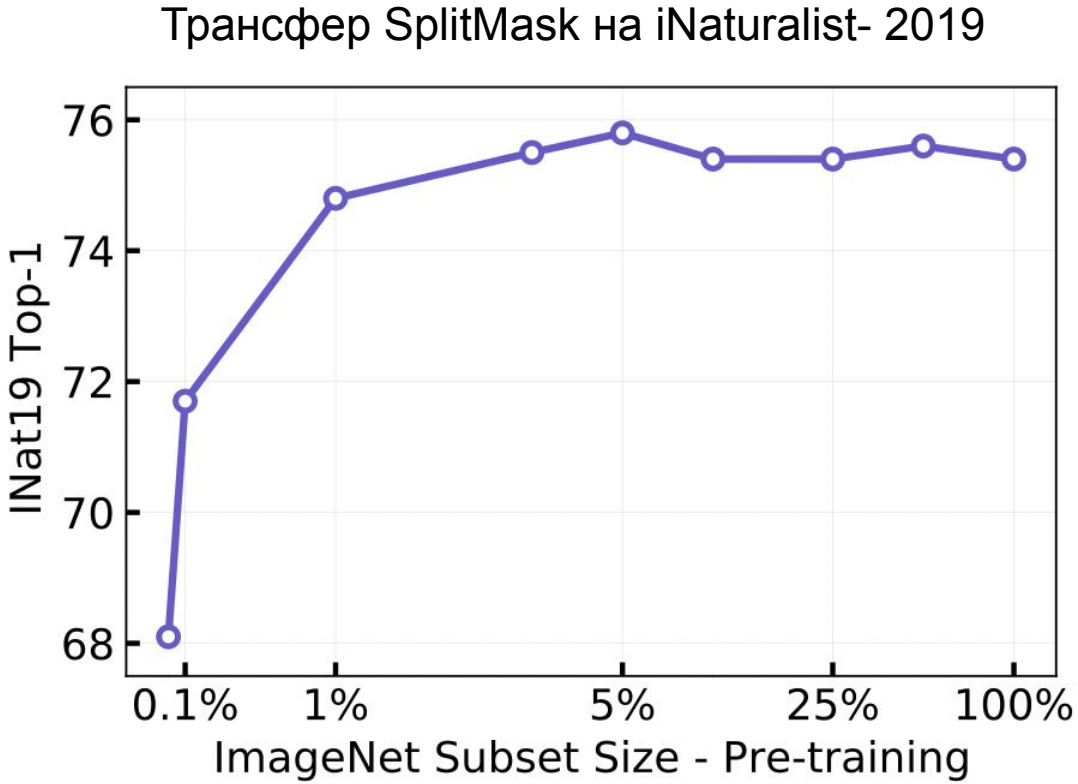
	DALL-E	Rand. Proj.	Rand. Patches	K-Means
iNat19	75.2	75.2	75.3	75.0

Анализ: размер датасета

Method	IMNet 1% *epochs: 30k*	IMNet 10% *epochs: 3k*	IMNet Full *epochs: 300*	COCO *epochs: 3k*
Supervised	71.6	75.0	75.8	-
DINO [18]	70.1	73.1	**78.4**	71.9
BEiT [24]	74.1	74.5	75.2	74.4
SplitMask	**74.8**	**75.4**	75.4	**76.3**

Трансфер на iNaturalist-2019

Анализ: размер датасета



Эксперименты: датасеты

Dataset	#Train	#Test	#Classes	Epochs
ImageNet [7]	1,281,167	50,000	1000	300
iNaturalist 2018 [12]	437,513	24,426	8,142	800
iNaturalist 2019 [13]	265,240	3,003	1,010	1,400
Food 101 [61]	75,750	25,250	101	5,000
Stanford Cars [60]	8,144	8,041	196	5,000
Clipart [62]	34,019	14,818	345	5,000
Painting [62]	52,867	22,892	345	5,000
Sketch [62]	49,115	21,271	345	5,000
ADE20k [64]	20,210	2,000	150	21,000
COCO [63]	118,287	5,000	80	3,000

Эксперименты: COCO object detection and instance segmentation

Method	Backbone	Pre-training			AP^b	AP^b_{50}	AP^b_{75}	AP^m	AP^m_{50}	AP^m_{75}
		Supervised	IMNet	COCO						
Random Initialization	ViT-S	✗	✗	✗	38.3	60.1	41.4	35.6	57.1	37.7
Random Initialization†		✗	✗	✗	42.8	64.5	45.6	39.1	61.5	41.7
DeiT [50]		✓	✓	✗	44.2	66.6	47.9	40.1	63.2	42.7
BEiT [24]		✗	✓	✗	44.5	66.2	48.8	40.3	63.2	43.1
DINO [18]		✗	✗	✓	43.7	65.5	47.7	39.6	62.3	42.3
BEiT		✗	✗	✓	44.7	66.3	48.8	40.2	63.1	43.2
SplitMask		✗	✗	✓	**45.3**	**66.9**	**49.4**	**40.6**	**63.6**	**43.5**
Random Initialization	ViT-B	✗	✗	✗	40.7	62.7	44.2	37.1	59.1	39.4
Random Initialization†		✗	✗	✗	43.0	64.2	46.9	38.8	61.3	41.6
DeiT [50]		✓	✓	✗	45.5	**67.9**	49.2	41.0	64.6	43.8
BEiT [24]		✗	✓	✗	46.3	67.6	50.6	41.6	64.5	44.9
DINO [18]		✗	✗	✓	43.1	64.4	46.9	38.9	61.4	41.4
BEiT		✗	✗	✓	46.7	67.7	51.2	41.8	65.0	44.6
SplitMask		✗	✗	✓	**46.8**	**67.9**	**51.5**	**42.1**	**65.3**	**45.1**

Эксперименты: ADE20k semantic segmentation

Method	Pre-training			mIoU
	Supervised	IMNet	ADE20k	
Random Init.	✗	✗	✗	25.4
DeiT [50]	✓	✗	✗	46.1
BEiT [24]	✗	✓	✗	45.6
BEiT	✗	✗	✓	45.6
SplitMask	✗	✗	✓	45.7

Эксперименты: classification fine tuning

Method	Backbone	Supervised pre-training	Data Used		iNat-18	iNat-19	Food 101	Cars	Clipart	Painting	Sketch
			IMNet	Target	437k	265k	75k	8k	34k	52k	49k
Liu et al. [67]∓	CVT-13	✗	✗	✓	-	-	-	-	60.6	55.2	57.6
	ResNet-50	✗	✗	✓	-	-	-	-	63.9	53.5	59.6
Random Init.	ViT-S	✗	✗	✓	59.6	67.5	84.7	35.3	41.0	38.4	37.2
DeiT [50]		✓	✓	✓	69.9	75.8	**91.5**	92.2	**79.6**	**74.2**	**72.5**
BEiT [24]		✗	✓	✓	68.1	75.2	90.5	92.4	75.3	68.7	68.5
BEiT		✗	✗	✓	68.8	76.1	90.7	92.7	-	69.0	-
SplitMask		✗	✗	✓	**70.1**	**76.3**	**91.5**	**92.8**	78.3	69.2	70.7
Random Init.	ViT-B	✗	✗	✓	59.6	68.1	83.3	36.9	41.9	37.6	34.9
DeiT [50]		✓	✓	✓	73.2	77.7	**91.9**	92.1	**80.0**	**73.8**	**72.6**
BEiT [24]		✗	✓	✓	71.6	78.6	91.0	**93.9**	78.0	71.5	71.4
BEiT		✗	✗	✓	72.4	79.3	91.7	92.7	-	70.7	-
SplitMask		✗	✗	✓	**74.6**	**80.4**	91.2	93.1	79.3	72.0	72.1

Эксперименты: classification ImageNet

Method	Backbone	Epochs	Top-1
MocoV3 [68]	ViT-S	300	81.4
DINO [18]		300	**81.5**
BEiT [24]		300	81.3
SplitMask		300	**81.5**
MocoV3 [68]	ViT-B	300	83.2
DINO [18]		400	**83.6**
BEiT [24]		300	82.8
BEiT [24]		800	83.2
SplitMask		300	**83.6**

Рецензия

В данной статье авторы исследовали self-supervised pretraining на примере denoising autoencoders. Авторы пришли к выводу, что denoising autoencoders более устойчивы к типу и размеру данных для предобучения. Авторы предложили свой метод для self-supervised pretraining - SplitMask.

Сильные стороны:

1. Статья в целом написана доходчиво, простым языком
2. Авторы провели достаточно полный эмпирический анализ self-supervised pretraining. Есть ablation study предложенного авторами метода
3. Значимость этой статьи заключается в том, что авторы развенчивают миф, что нужно предобучаться на больших наборах данных

Слабые стороны:

1. Нет авторской реализации SplitMask
2. Упущены детали в статье для воспроизводимости такой же модели

Оценка: 8 | Уверенность: 3

Авторы

- Alaaeldin El-Nouby: PhD Student, Facebook AI Research, INRIA

  Работы по Vision Transformer:

  - LeViT: a Vision Transformer in ConvNet's Clothing for Faster Inference
  - Training vision transformers for image retrieval
  - XCiT: Cross-Covariance Image Transformers

- Gautier Izacard: Facebook AI Research, INRIA
- Hugo Touvron: Facebook AI Research, Sorbonne University
- Ivan Laptev: Research director, INRIA
- Hervé Jégou: Facebook AI Research
- Edouard Grave: Facebook AI Research

Контекст

Опорные работы:

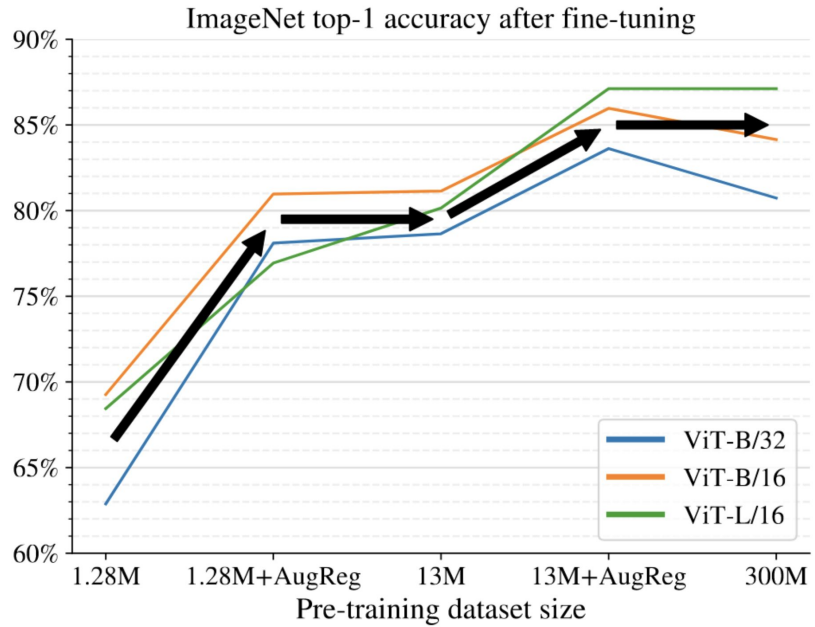
- An image is worth 16x16 words: Transformers for image recognition at scale
- BEiT: BERT Pre-Training of Image Transformers
- Emerging Properties in Self-Supervised Vision Transformers (DINO)
- An empirical study of training self-supervised vision transformers (Moco V3)

Статья опубликована 20.12.2021. Цитирований нет.

Контекст

[How to train your ViT? Data, Augmentation, and Regularization in Vision Transformers]

Google Research, Brain Team



Дальнейшие возможные исследования:

Рассмотреть насколько данное исследование справедливо для моделей, отличных от Vision Transformer, а также на других задачах

Применение и практическое значение:

- Уменьшение необходимых ресурсов для получения предобученных моделей
- Улучшение в качестве, если предобучаться сразу на наборе данных для которого решается задача