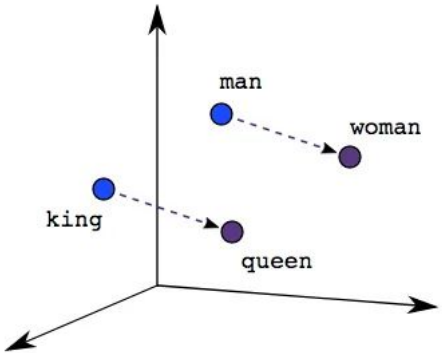
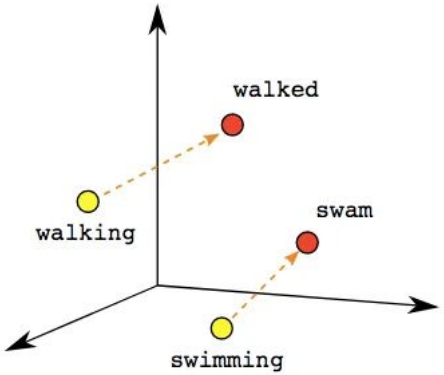
ViT (Vision Transformers)

ViT paper
Video explanation
Best Transformer intro: https://jalammar.github.io/illustrate...
CNNs vs ViT: https://arxiv.org/abs/2108.08810
CNNs vs ViT Blog: https://towardsdatascience.com/do-vis...
Swin Transformer: https://arxiv.org/abs/2103.14030
DeiT: https://arxiv.org/abs/2012.12877
Visualizations:
https://jacobgil.github.io/deeplearning/vision-transformer-explainability
https://arxiv.org/pdf/2012.09838v1

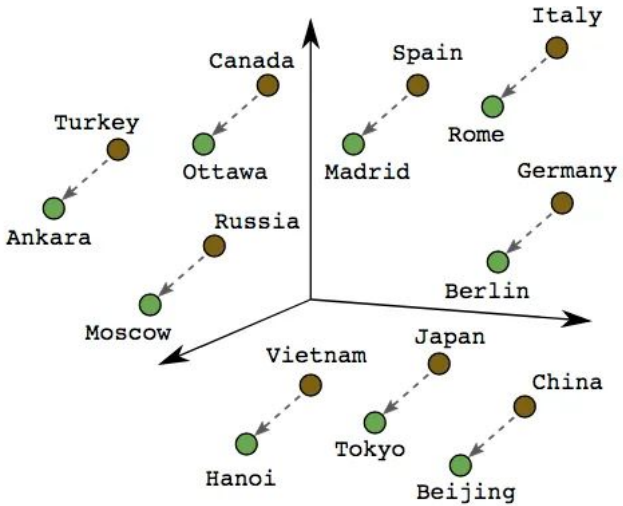
Representación semántica en espacio vectorial
Token embeddings



Male-Female

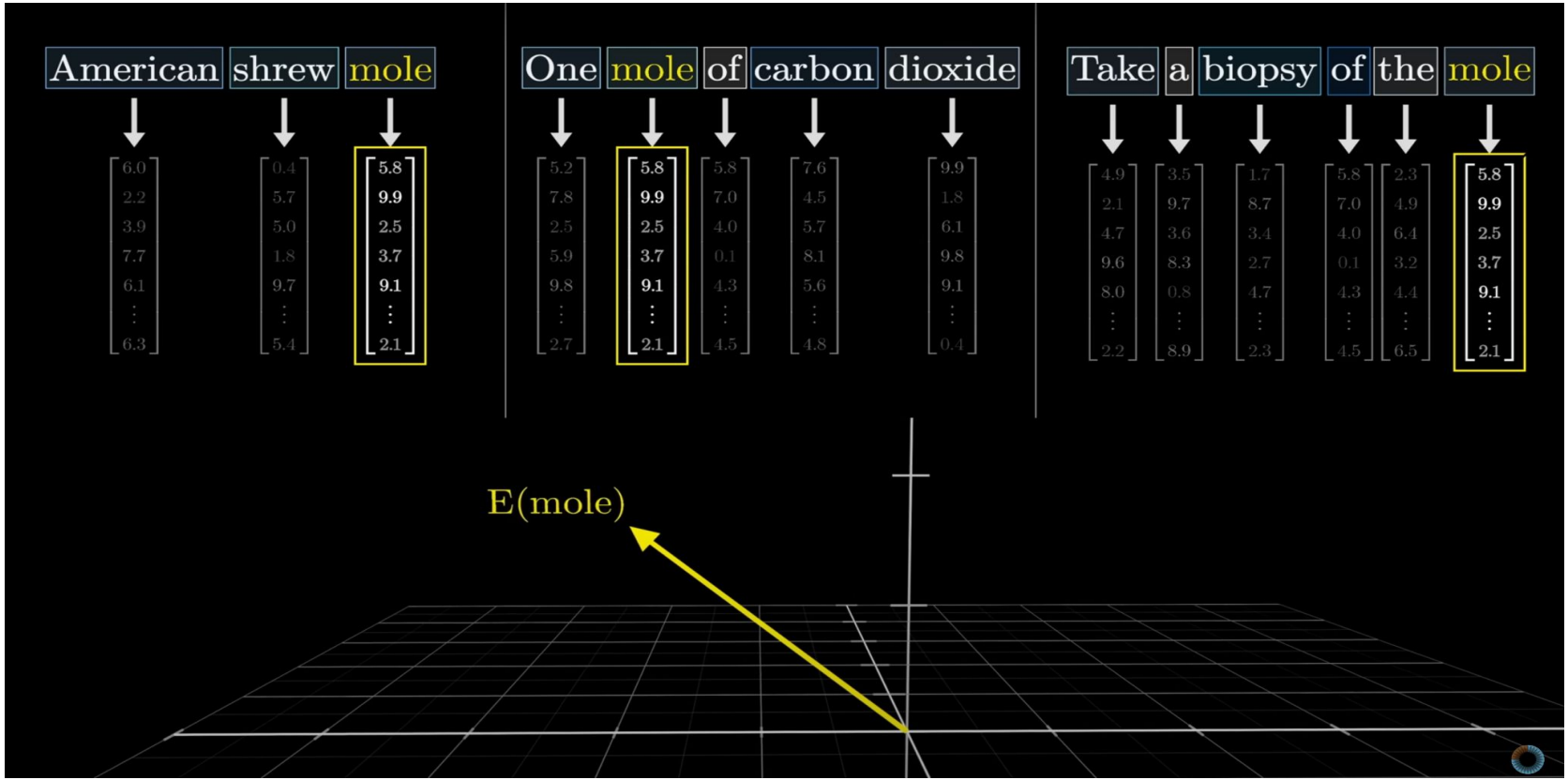


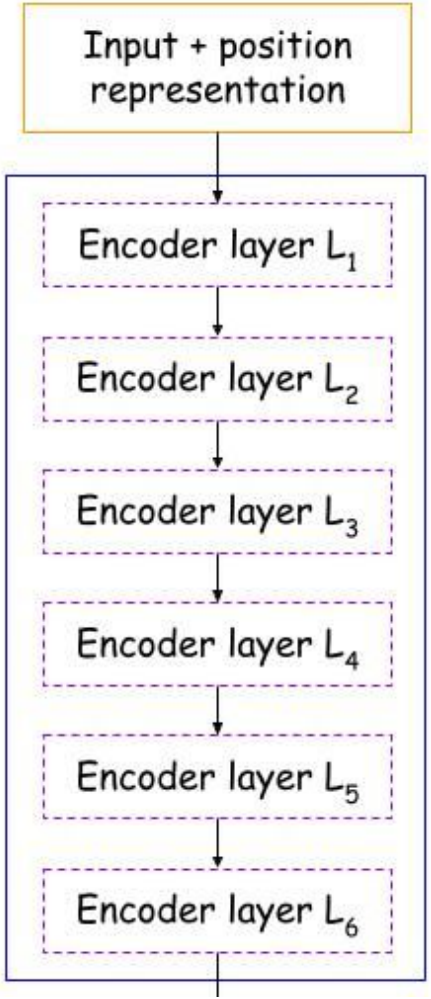
Verb Tense

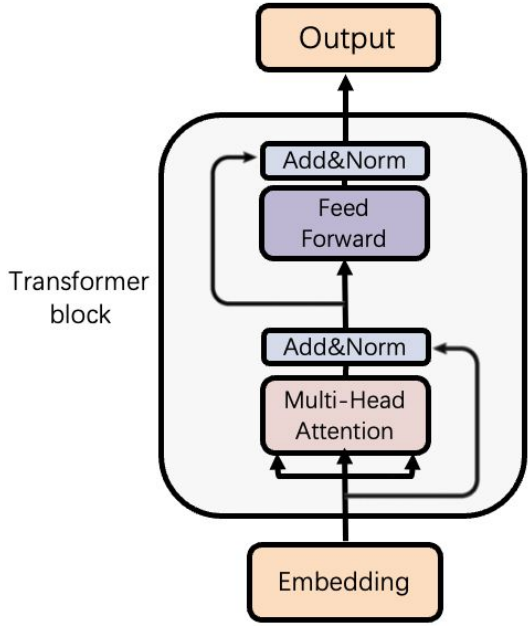


Country-Capital

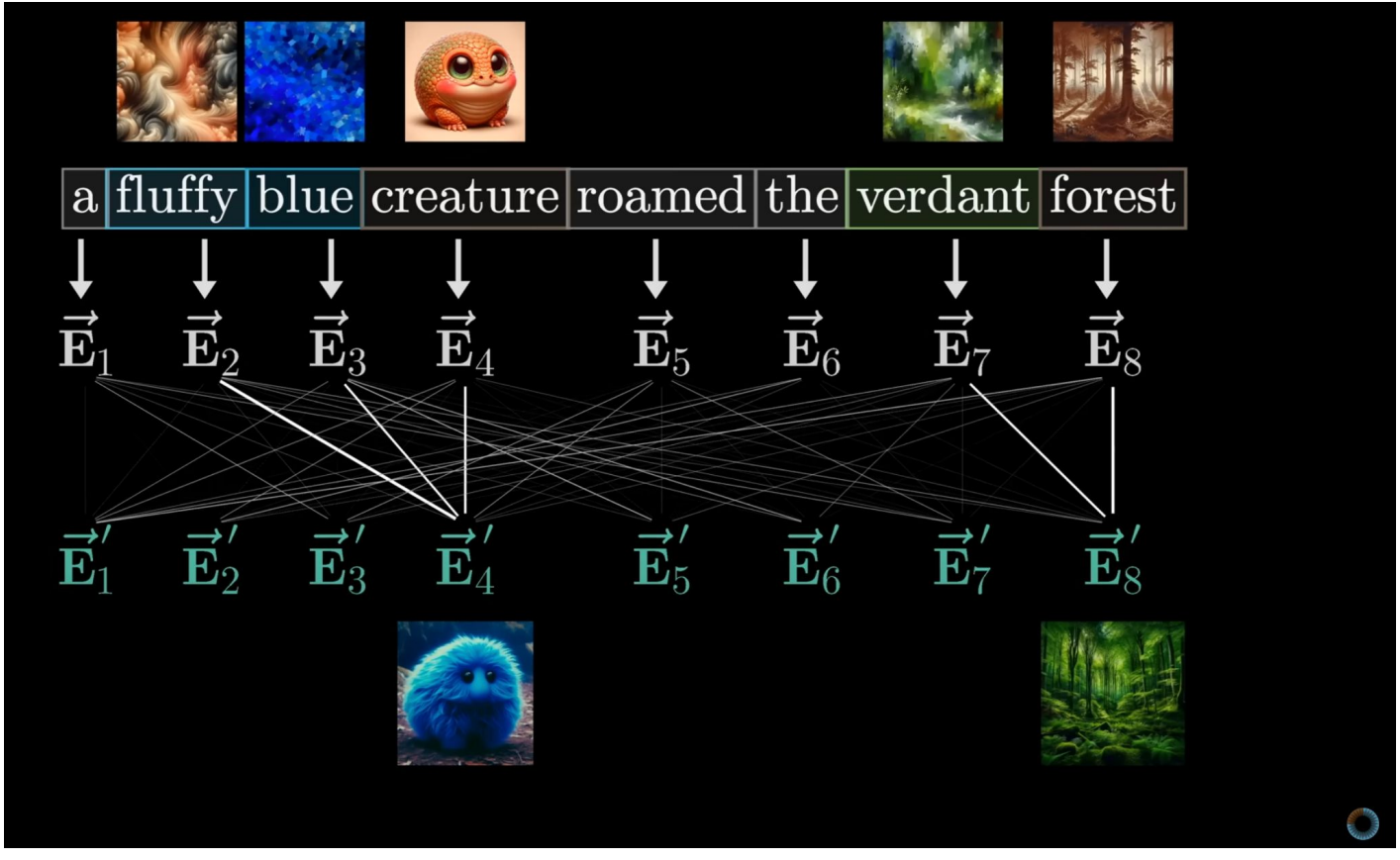
Non contextual embeddings (3blue1brown)




Input + position representation
Encoder layer L_1
Encoder layer L_2
Encoder layer L_3
Encoder layer L_4
Encoder layer L_5
Encoder layer L_6

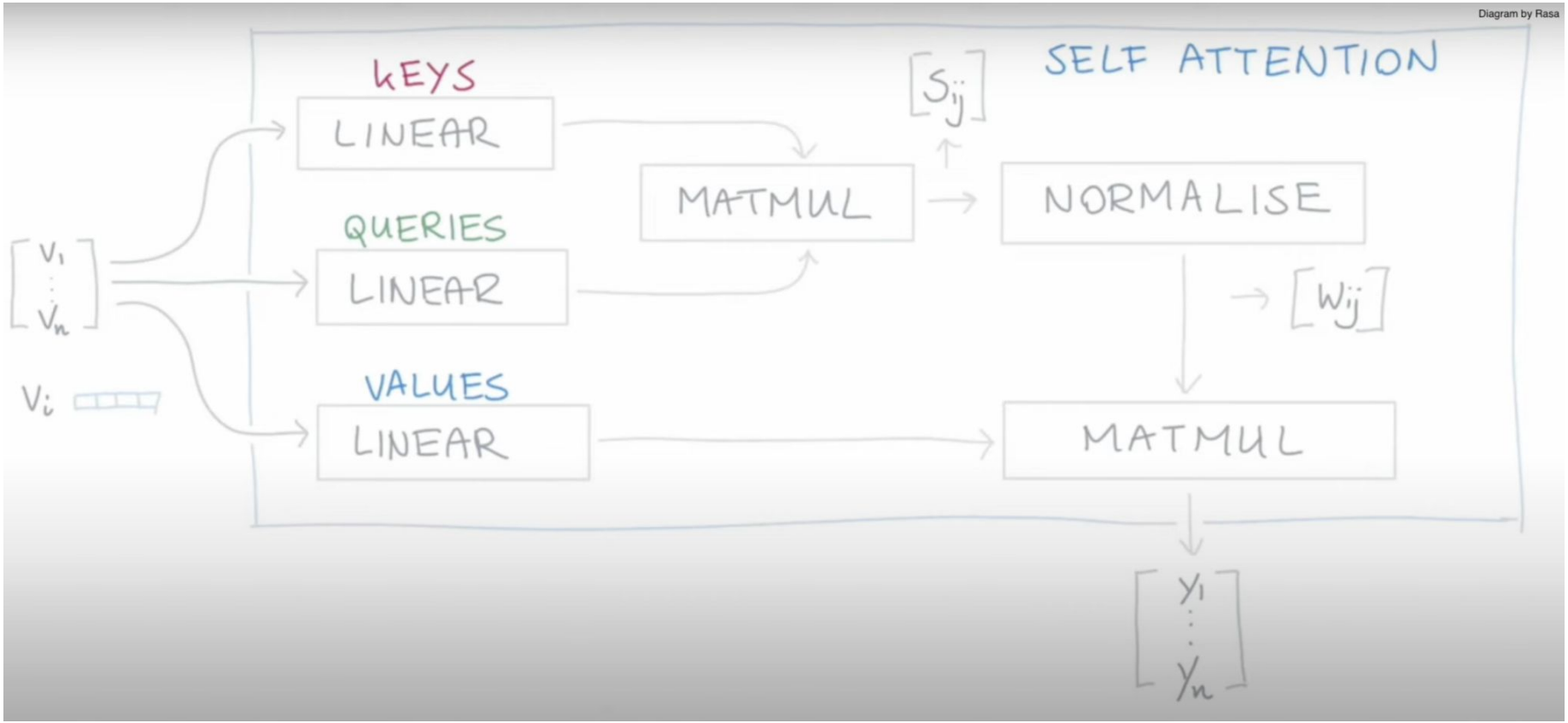

Output
Add&Norm
Feed Forward
Transformer block
Add&Norm
Multi-Head Attention
Embedding

Contextual embeddings (Transformer architecture)



Self Attention

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



Concept	CNN	Self-Attention
Input	Image: $H \times W \times C$	Sequence: $T \times d_{\text{model}}$
“Filters”	Learnable kernels (e.g., $3 \times 3 \times C$)	Learnable linear projections: W^Q, W^K, W^V
Computation	Dot product of kernel with input patch	Dot product of query with key: $QK^\top$
Output	Feature maps per filter	Weighted sum of values: $\text{softmax}(QK^\top)V$
Spatial Context	Local (e.g., 3×3 window)	Global (attends to all tokens)
Channels	Depth (number of filters)	Embedding dimension (model depth)

CNNs apply fixed-size **local filters** to detect patterns in space.

Attention uses **adaptive, content-dependent filters** (via dot products) that can look at the **whole sequence**

Analogía canales / embedding / features

Contexto 2021

Transformers aplicados a NLP

- pre entrenamiento con corpus grandes
- ajuste fino con dataset en tareas específicas
- modelos y datasets en crecimiento, sin indicios de saturación de performance

Computer Vision:

- Estado del arte para CNNs
- Se empezaron a combinar con self attention

Resultados preliminares con transformers

Dataset medianos como ImageNet:

- Pocos puntos debajo de ResNets de tamaños comparables
- Resultado esperado
- Carecen del sesgo inductivo de las CNNs (equivarianza a la translación y localidad)
- Necesitan mucha data para generalizar (debido a no tener sesgos)

Dataset grandes (14M a 300M)

- ImageNet-21k: 21k classes and 14M images
- JFT: 18k classes and 303M high-resolution images

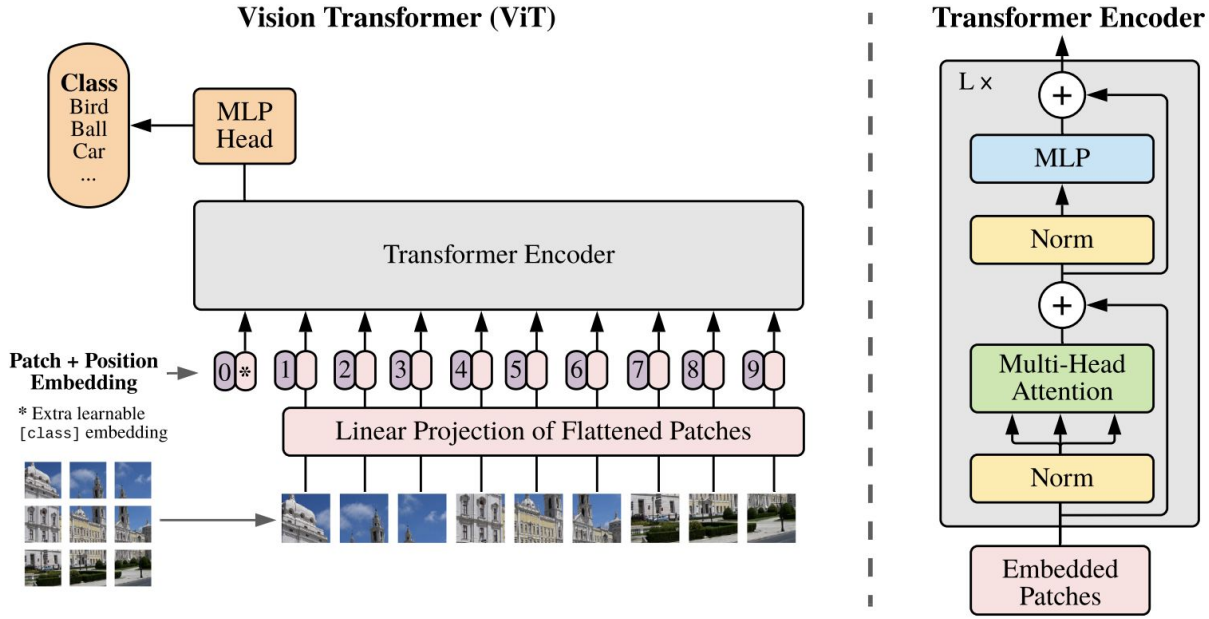


Figure 1: Model overview. We split an image into fixed-size patches, linearly embed each of them, add position embeddings, and feed the resulting sequence of vectors to a standard Transformer encoder. In order to perform classification, we use the standard approach of adding an extra learnable "classification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

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** $\pm$ 0.04	87.76 $\pm$ 0.03	85.30 $\pm$ 0.02	87.54 $\pm$ 0.02	88.4/88.5*
ImageNet ReaL	**90.72** $\pm$ 0.05	90.54 $\pm$ 0.03	88.62 $\pm$ 0.05	90.54	90.55
CIFAR-10	**99.50** $\pm$ 0.06	99.42 $\pm$ 0.03	99.15 $\pm$ 0.03	99.37 $\pm$ 0.06	–
CIFAR-100	**94.55** $\pm$ 0.04	93.90 $\pm$ 0.05	93.25 $\pm$ 0.05	93.51 $\pm$ 0.08	–
Oxford-IIIT Pets	**97.56** $\pm$ 0.03	97.32 $\pm$ 0.11	94.67 $\pm$ 0.15	96.62 $\pm$ 0.23	–
Oxford Flowers-102	99.68 $\pm$ 0.02	**99.74** $\pm$ 0.00	99.61 $\pm$ 0.02	99.63 $\pm$ 0.03	–
VTAB (19 tasks)	**77.63** $\pm$ 0.23	76.28 $\pm$ 0.46	72.72 $\pm$ 0.21	76.29 $\pm$ 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).

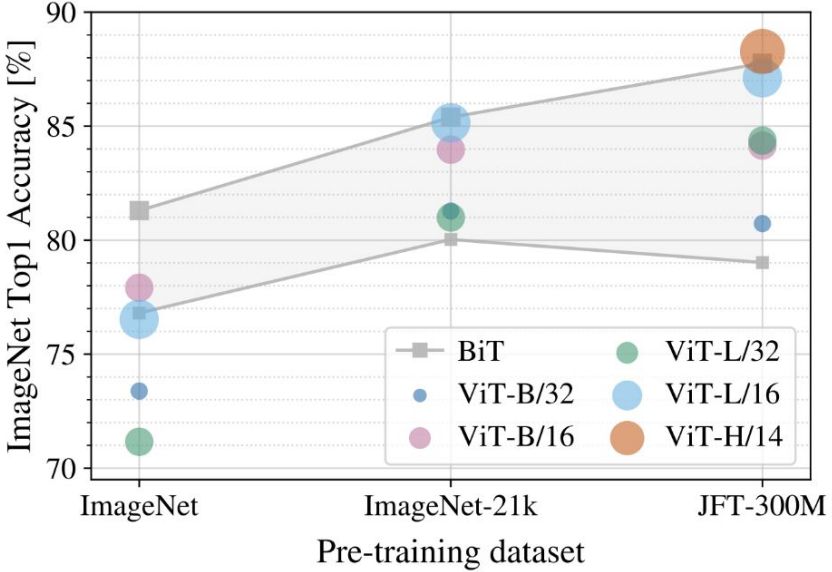


Figure 3: Transfer to ImageNet. While large ViT models perform worse than BiT ResNets (shaded area) when pre-trained on small datasets, they shine when pre-trained on larger datasets. Similarly, larger ViT variants overtake smaller ones as the dataset grows.

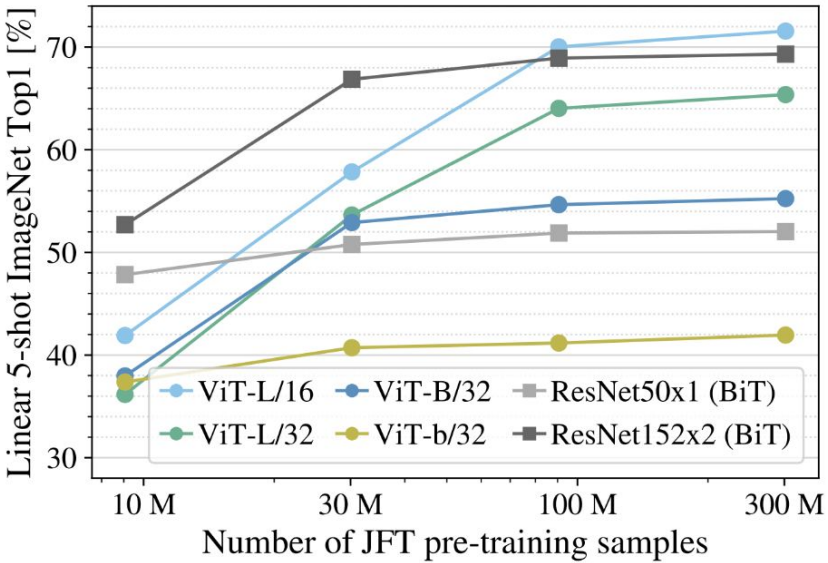


Figure 4: Linear few-shot evaluation on ImageNet versus pre-training size. ResNets perform better with smaller pre-training datasets but plateau sooner than ViT, which performs better with larger pre-training. ViT-b is ViT-B with all hidden dimensions halved.

Training

Models	Dataset	Epochs	Base LR	LR decay	Weight decay	Dropout
ViT-B/{16,32}	JFT-300M	7	$8 \cdot 10^{-4}$	linear	0.1	0.0
ViT-L/32	JFT-300M	7	$6 \cdot 10^{-4}$	linear	0.1	0.0
ViT-L/16	JFT-300M	7/14	$4 \cdot 10^{-4}$	linear	0.1	0.0
ViT-H/14	JFT-300M	14	$3 \cdot 10^{-4}$	linear	0.1	0.0
R50x{1,2}	JFT-300M	7	10^{-3}	linear	0.1	0.0
R101x1	JFT-300M	7	$8 \cdot 10^{-4}$	linear	0.1	0.0
R152x{1,2}	JFT-300M	7	$6 \cdot 10^{-4}$	linear	0.1	0.0
R50+ViT-B/{16,32}	JFT-300M	7	$8 \cdot 10^{-4}$	linear	0.1	0.0
R50+ViT-L/32	JFT-300M	7	$2 \cdot 10^{-4}$	linear	0.1	0.0
R50+ViT-L/16	JFT-300M	7/14	$4 \cdot 10^{-4}$	linear	0.1	0.0
ViT-B/{16,32}	ImageNet-21k	90	10^{-3}	linear	0.03	0.1
ViT-L/{16,32}	ImageNet-21k	30/90	10^{-3}	linear	0.03	0.1
ViT-$*$	ImageNet	300	$3 \cdot 10^{-3}$	cosine	0.3	0.1

Table 3: Hyperparameters for training. All models are trained with a batch size of 4096 and learning rate warmup of 10k steps. For ImageNet we found it beneficial to additionally apply gradient clipping at global norm 1. Training resolution is 224.

Fine tuning

- SGD con momentum = 0.5
- Small Grid Search para learning rates
- resolución: 384 (diferente que train - práctica común)
- batch_size = 512 (diferente que train)
- no weight decay
- Se reemplaza la Head (dos capas densas) por una de DxC donde D es la dimensión de los embeddings y C la cantidad de clases (inicializada en 0)

Algunas cosas cambian según el dataset

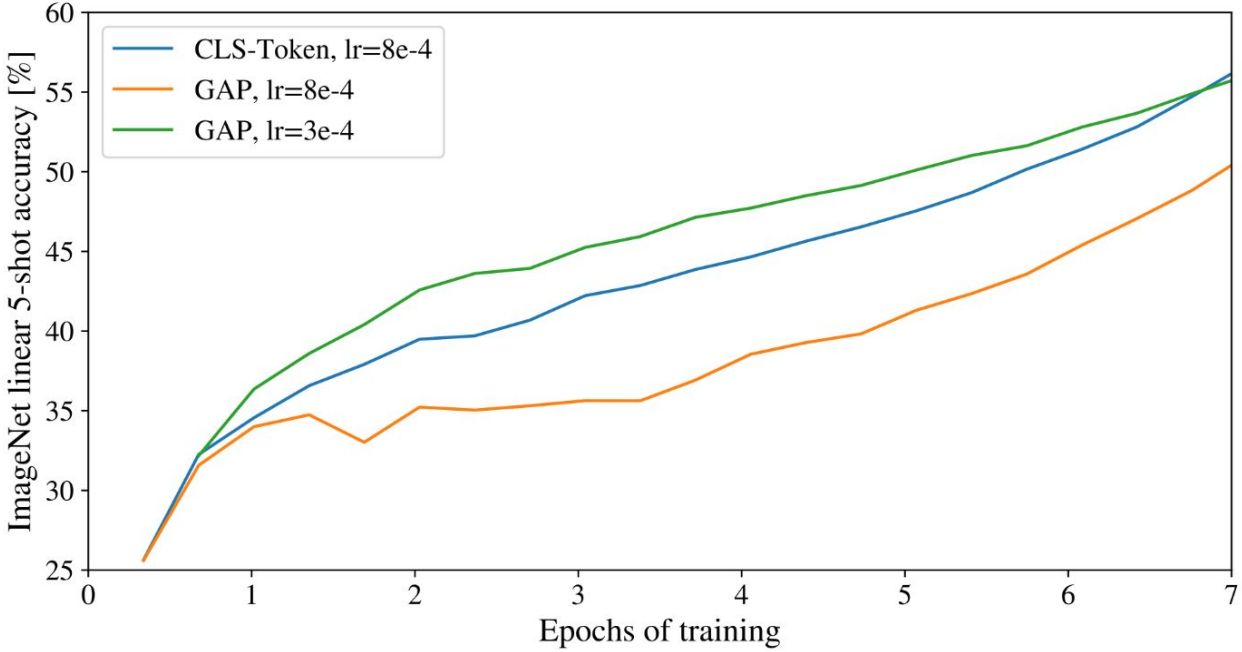


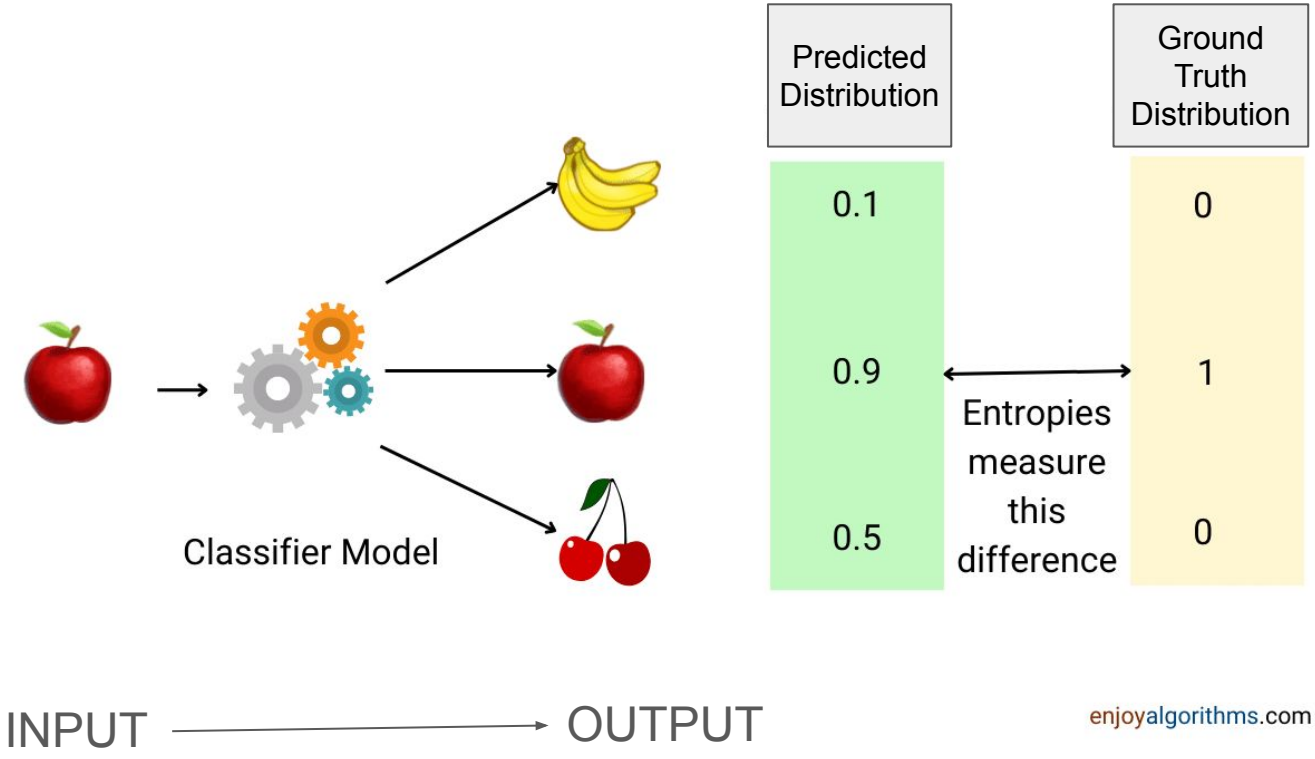
Figure 9: Comparison of class-token and global average pooling classifiers. Both work similarly well, but require different learning-rates.

Foundational Models

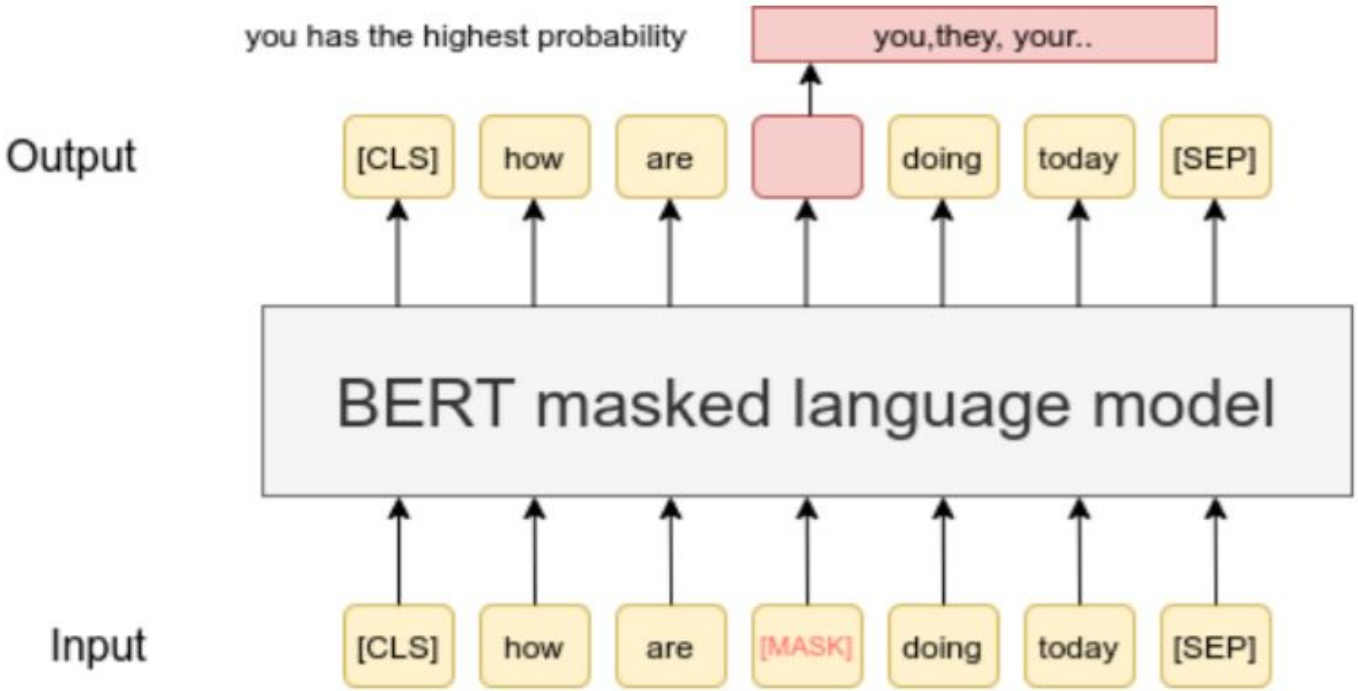
AI Foundational model

1. **Pre-trained on Large Datasets ✅**
    a. Self-supervised (unsupervised)
    b. Massive dataset
2. Generalizable Representations
    a. Learn Meaningful features (semantics)
    b. Embeddings
3. Transferable
    a. Fine-tunable in small diverse datasets
    b. Multiple downstream tasks: Classification, Generation, Prediction

Supervised learning Training - Needs labeling



Self-supervised (Encoder) - **BERT** ([Paper])

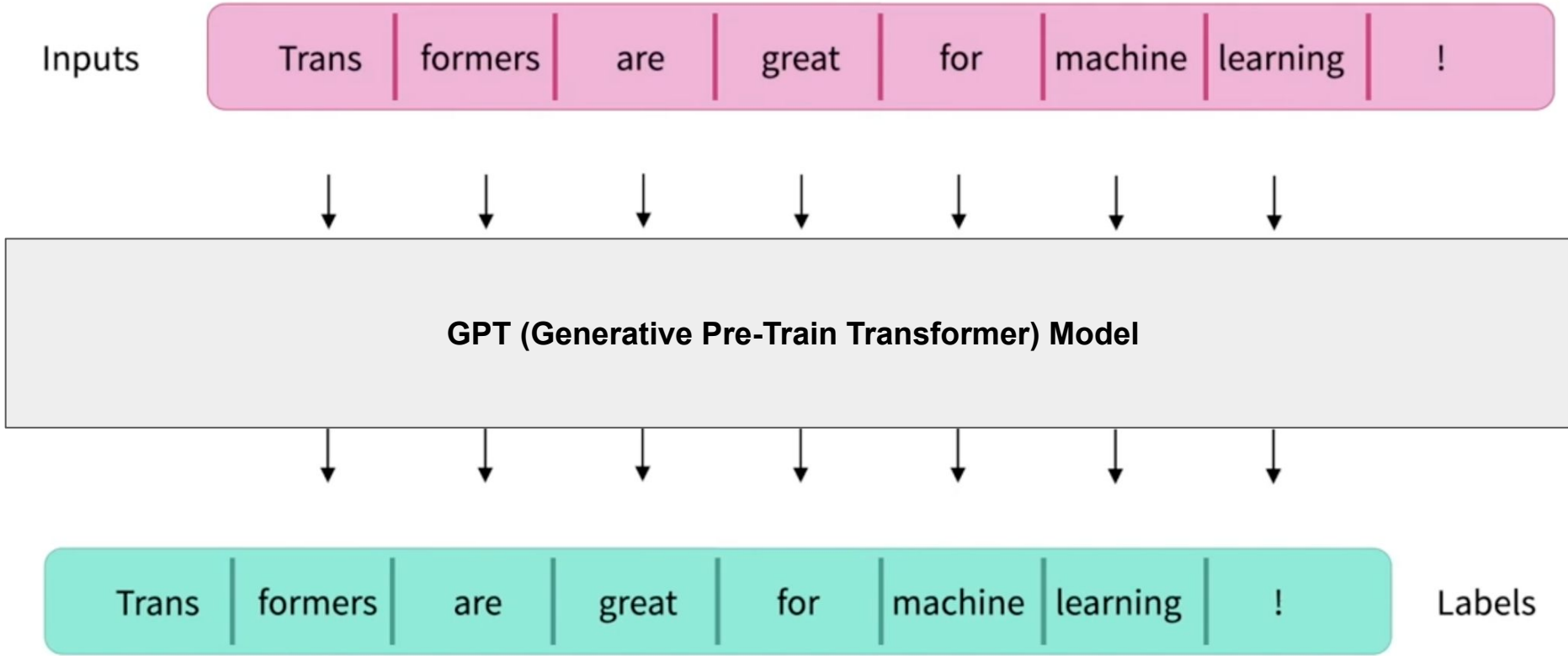


you	0.459
we	0.212
they	0.153
things	0.038
people	0.035
others	0.019
students	0.018
kids	0.016

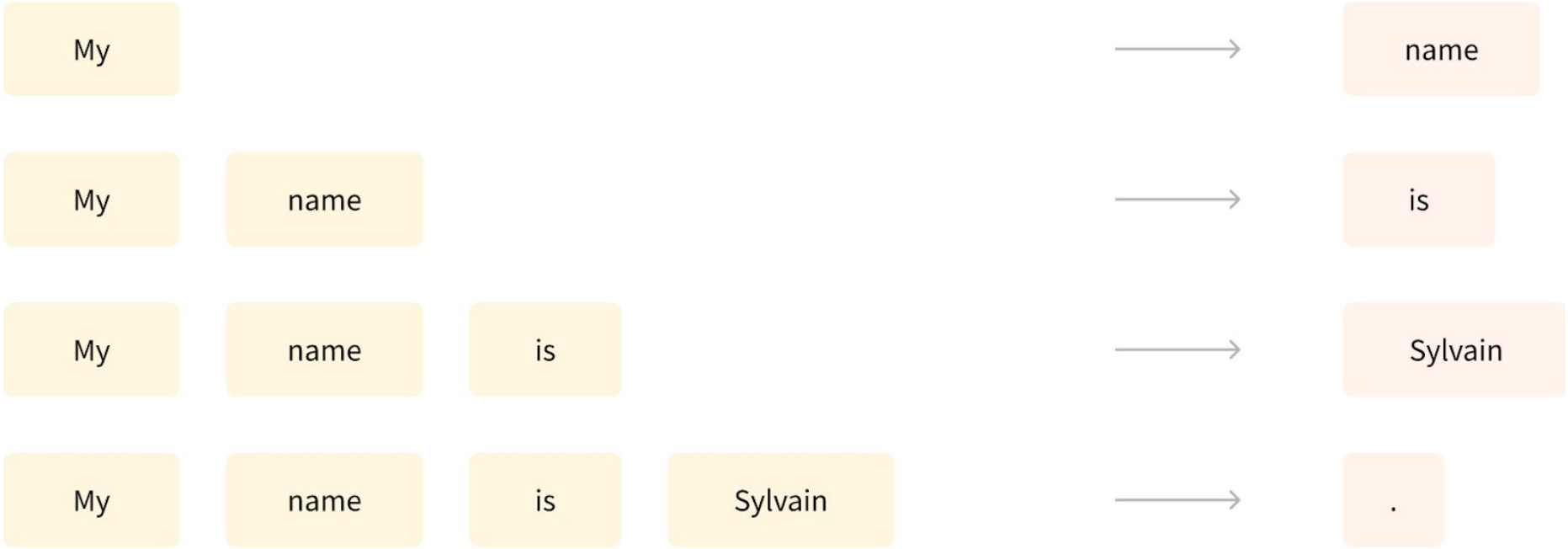
Bidirectional **E**ncoder **R**epresentations **T**ransformer

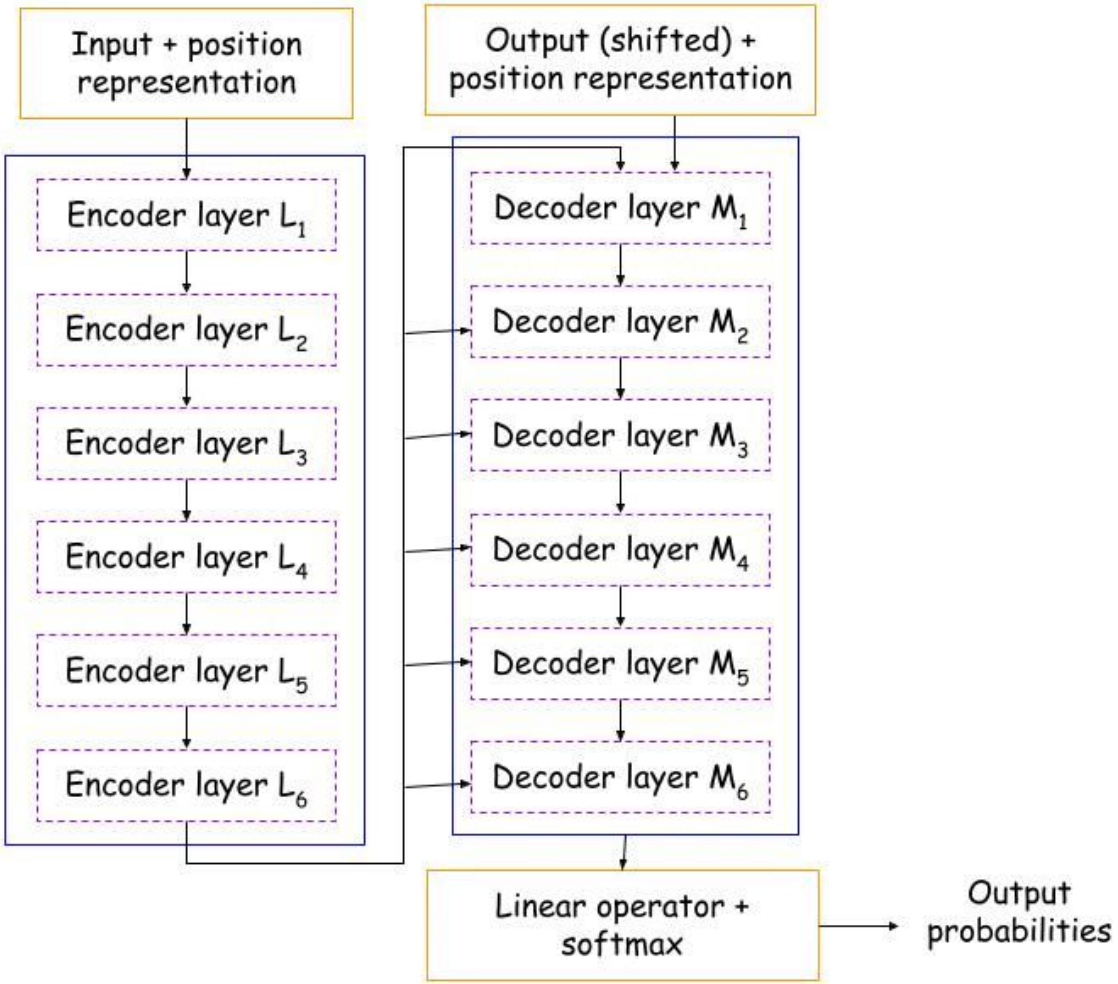
Randomly masks 15% of the words.

Self-supervised (decoder) - **GPT**



Inference - **GPT**




Input + position representation
Output (shifted) + position representation
Encoder layer L_1
Encoder layer L_2
Encoder layer L_3
Encoder layer L_4
Encoder layer L_5
Encoder layer L_6
Decoder layer M_1
Decoder layer M_2
Decoder layer M_3
Decoder layer M_4
Decoder layer M_5
Decoder layer M_6
Linear operator + softmax
Output probabilities

What does the model learn?

Sintaxis: Language structure, grammar rules, sentence construction, etc. Learned not by rules but by exposure to billions of examples

Semantics: word meanings, concept relationships, and how words/phrases are used in different contexts.

Context: They pick up on contextual dependencies, even across long passages.

Zero-shot

Zero-shot

The model predicts the answer given only a natural language description of the task. No gradient updates are performed.

```
1  Translate English to French:        ← task description
2  cheese =>                           ← prompt
```

One-shot

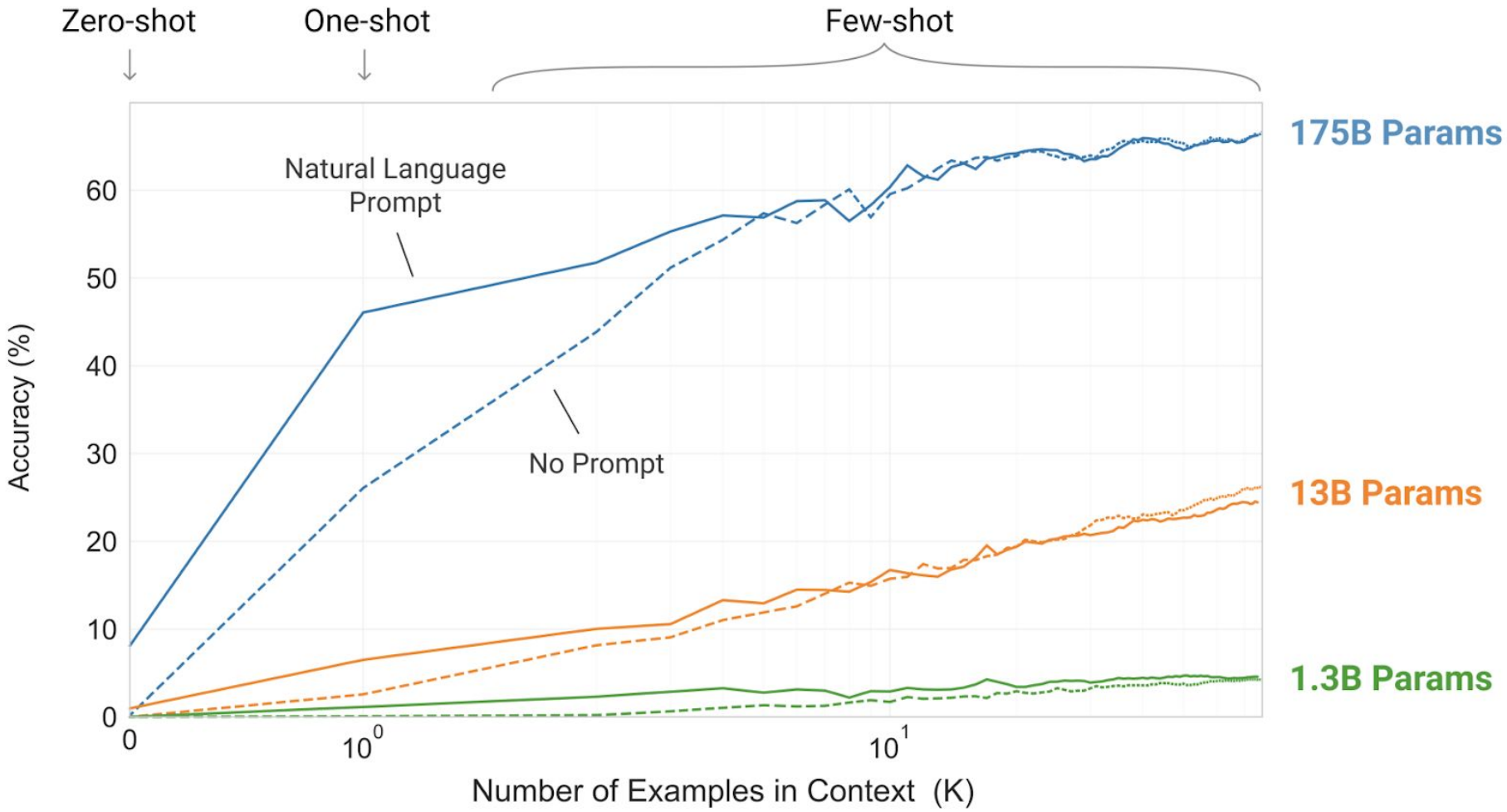
In addition to the task description, the model sees a single example of the task. No gradient updates are performed.

```
1  Translate English to French:        ← task description
2  sea otter => loutre de mer          ← example
3  cheese =>                           ← prompt
```

Few-shot

In addition to the task description, the model sees a few examples of the task. No gradient updates are performed.

```
1  Translate English to French:        ← task description
2  sea otter => loutre de mer          ← examples
3  peppermint => menthe poivrée        ←
4  plush girafe => girafe peluche      ←
5  cheese =>                           ← prompt
```


Zero-shot
One-shot
Few-shot
175B Params
13B Params
1.3B Params
Natural Language Prompt
No Prompt
Accuracy (%)
0
10
20
30
40
50
60
0
10^0
10^1
Number of Examples in Context (K)

Massive dataset

Token: Word fraction, example: tele-vision

Total tokens: 50k a 200K

(GPT3: 50,257, LLaMA3: 128,256)

Training Total tokens: 300B a 15T

(LLaMA1: 1.4T, LLaMA3: 15T)

Total different data tokens:

(LLaMA1: <1T, LLaMA3: ~3/5T)

More than one epoch per training

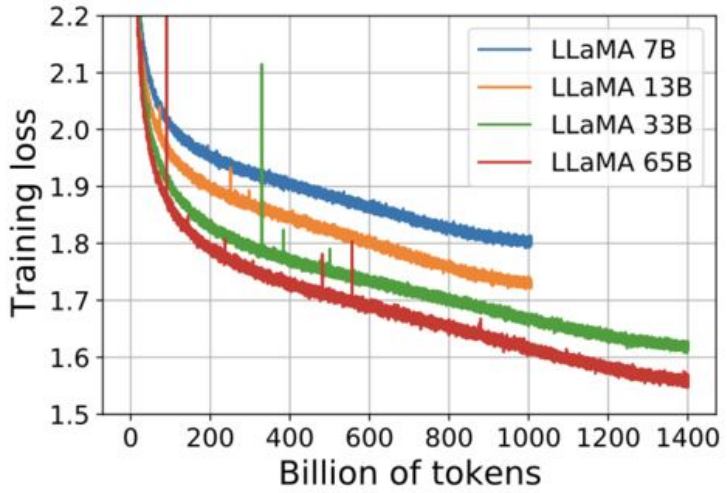


Figure 1: **Training loss over train tokens for the 7B, 13B, 33B, and 65 models.** LLaMA-33B and LLaMA-65B were trained on 1.4T tokens. The smaller models were trained on 1.0T tokens. All models are trained with a batch size of 4M tokens.

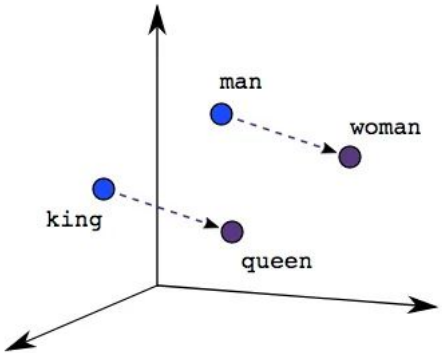
AI Foundational model

1. Pre-trained on Large Datasets
    a. Self-supervised (unsupervised)
    b. Massive dataset
2. Generalizable Representations
    a. Learn Meaningful features (semantics)
    b. Embeddings
3. Transferable
    a. Fine-tunable in small diverse datasets
    b. Multiple downstream tasks: Classification, Generation, Prediction

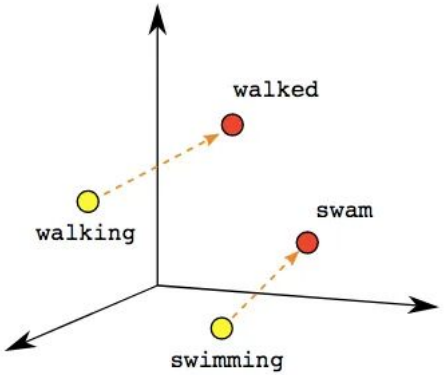
AI Foundational model

1. Pre-trained on Large Datasets
    a. Self-supervised (unsupervised)
    b. Massive dataset
2. **Generalizable Representations ✅**
    a. Learn Meaningful features (semantics)
    b. Embeddings
3. Transferable
    a. Fine-tunable in small diverse datasets
    b. Multiple downstream tasks: Classification, Generation, Prediction

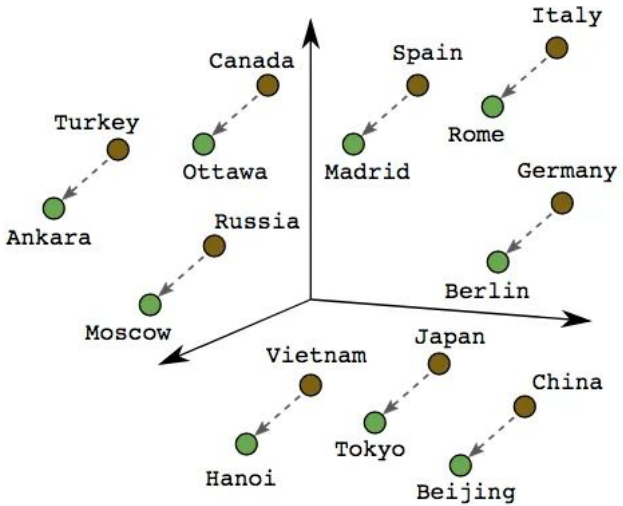
Representación semántica en espacio vectorial
Token embeddings



Male-Female

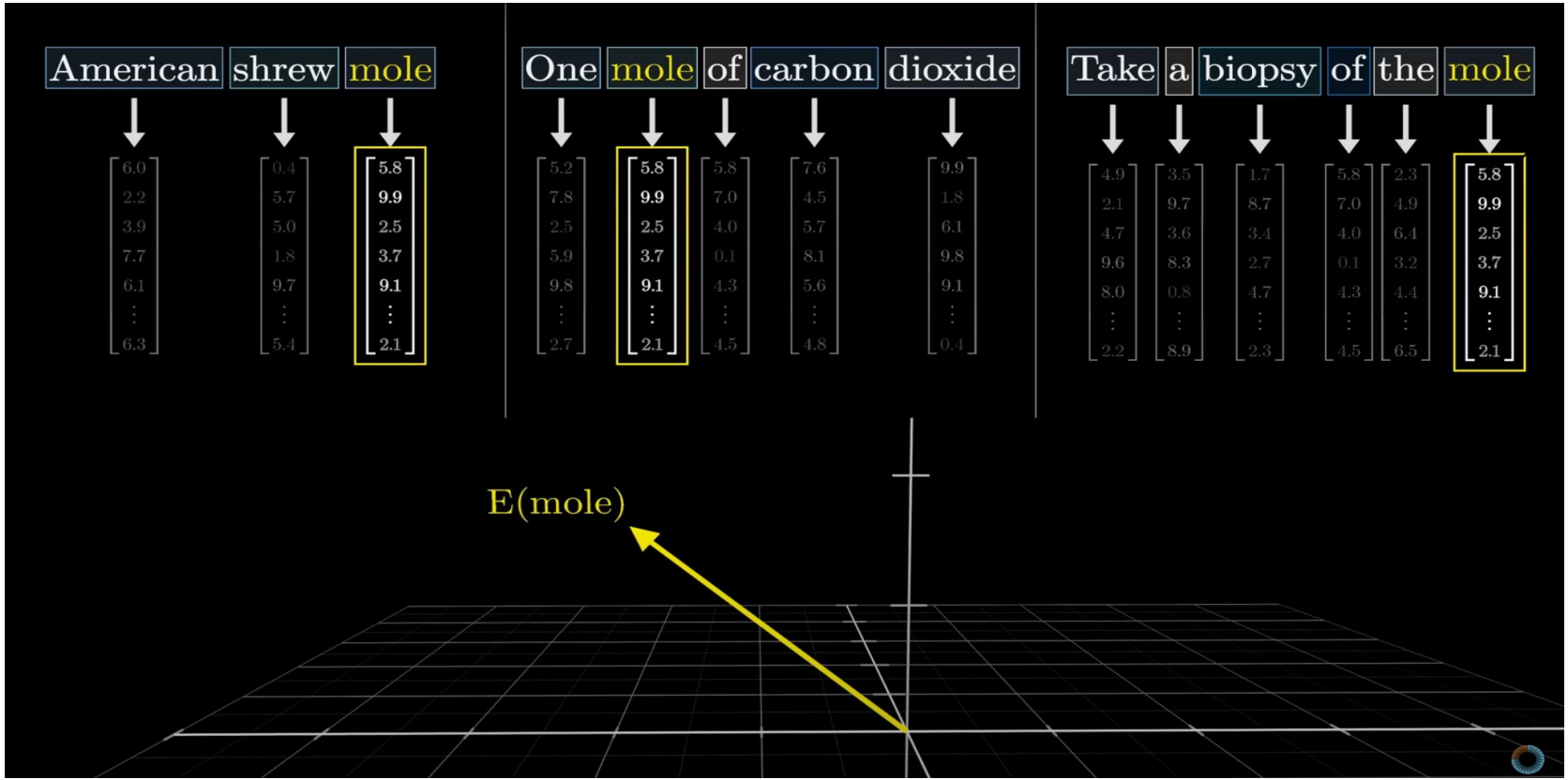


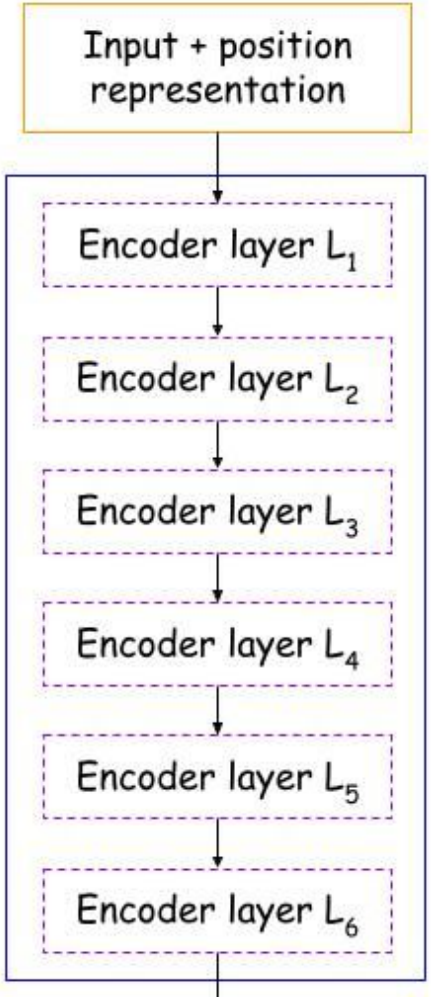
Verb Tense

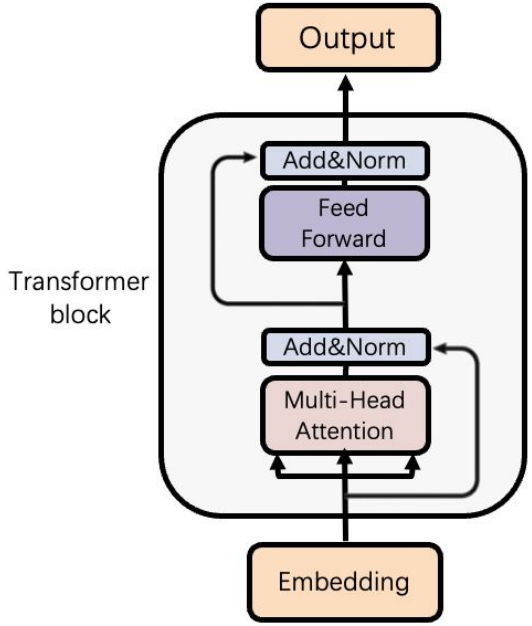


Country-Capital

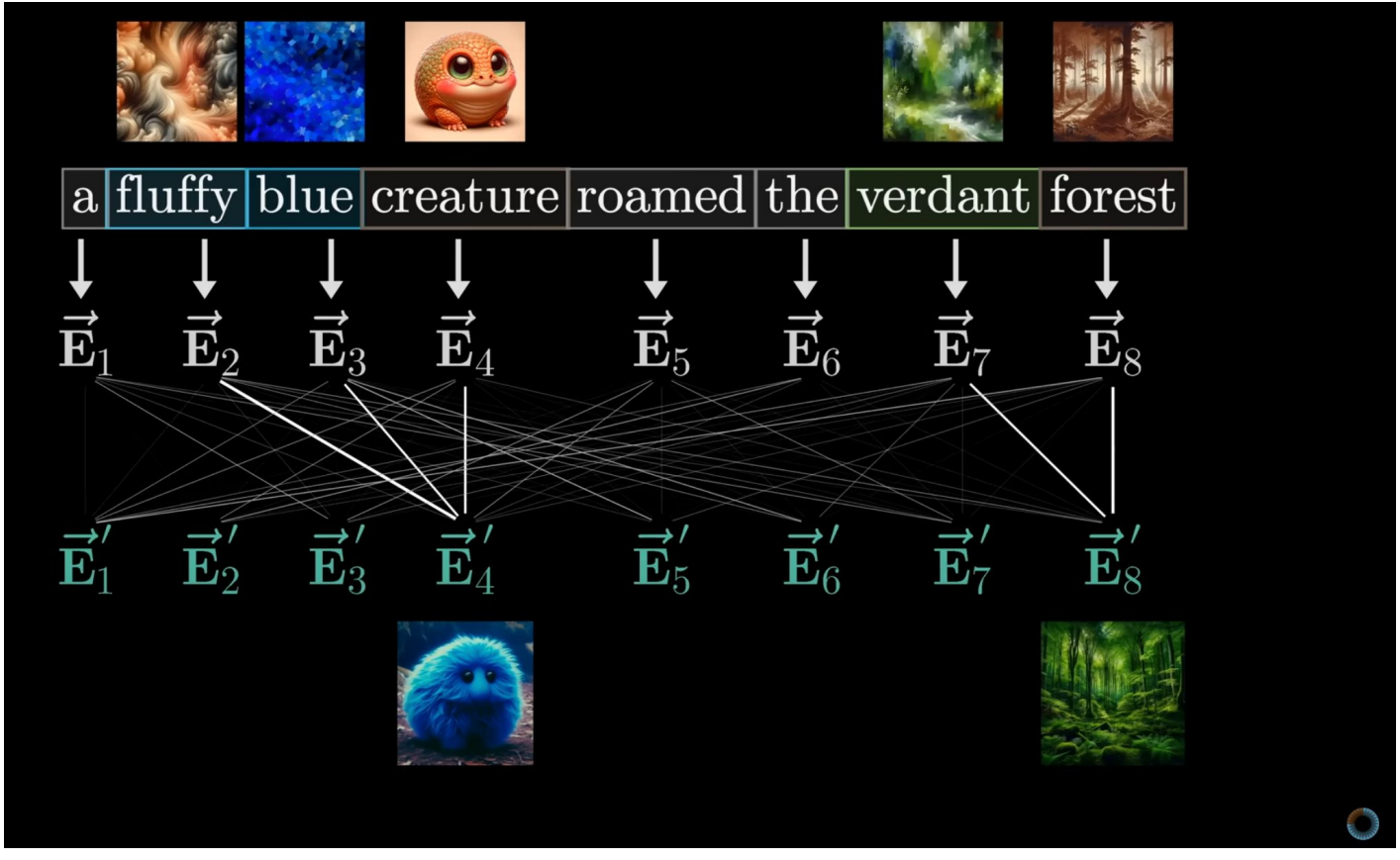
Non contextual embeddings (3blue1brown)



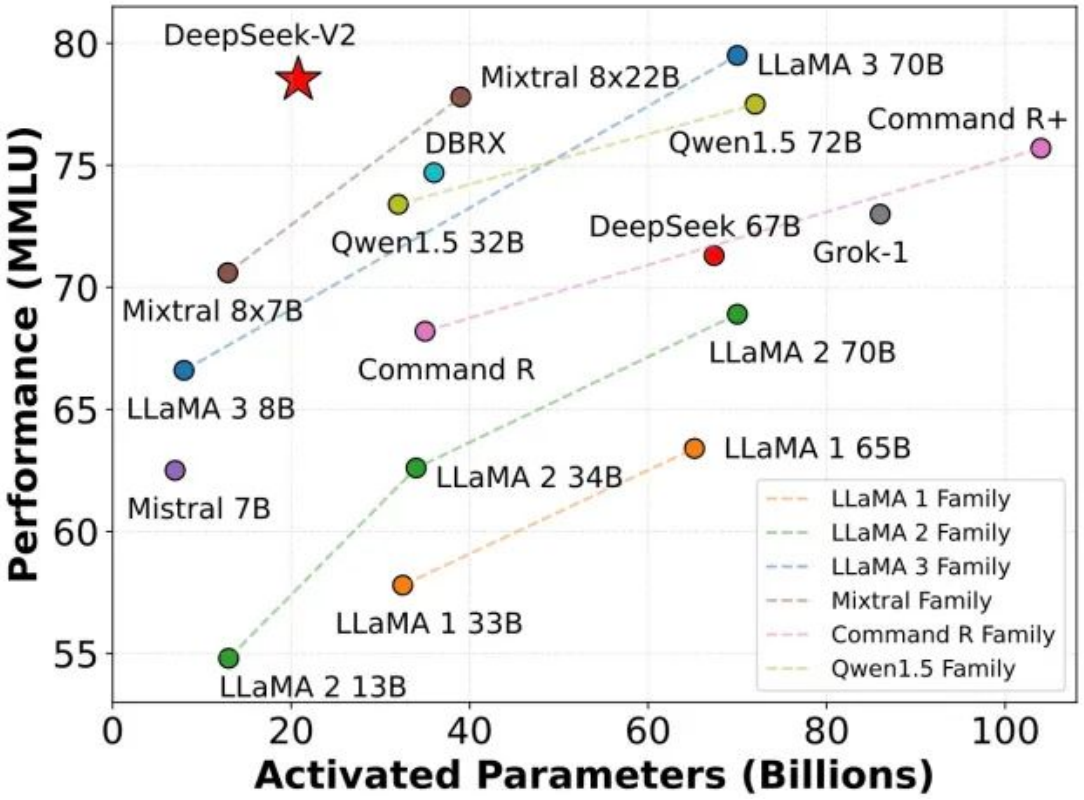

Input + position representation
Encoder layer L_1
Encoder layer L_2
Encoder layer L_3
Encoder layer L_4
Encoder layer L_5
Encoder layer L_6


Output
Add&Norm
Feed Forward
Transformer block
Add&Norm
Multi-Head Attention
Embedding

Contextual embeddings (Transformer architecture)



Will this tendency continue?



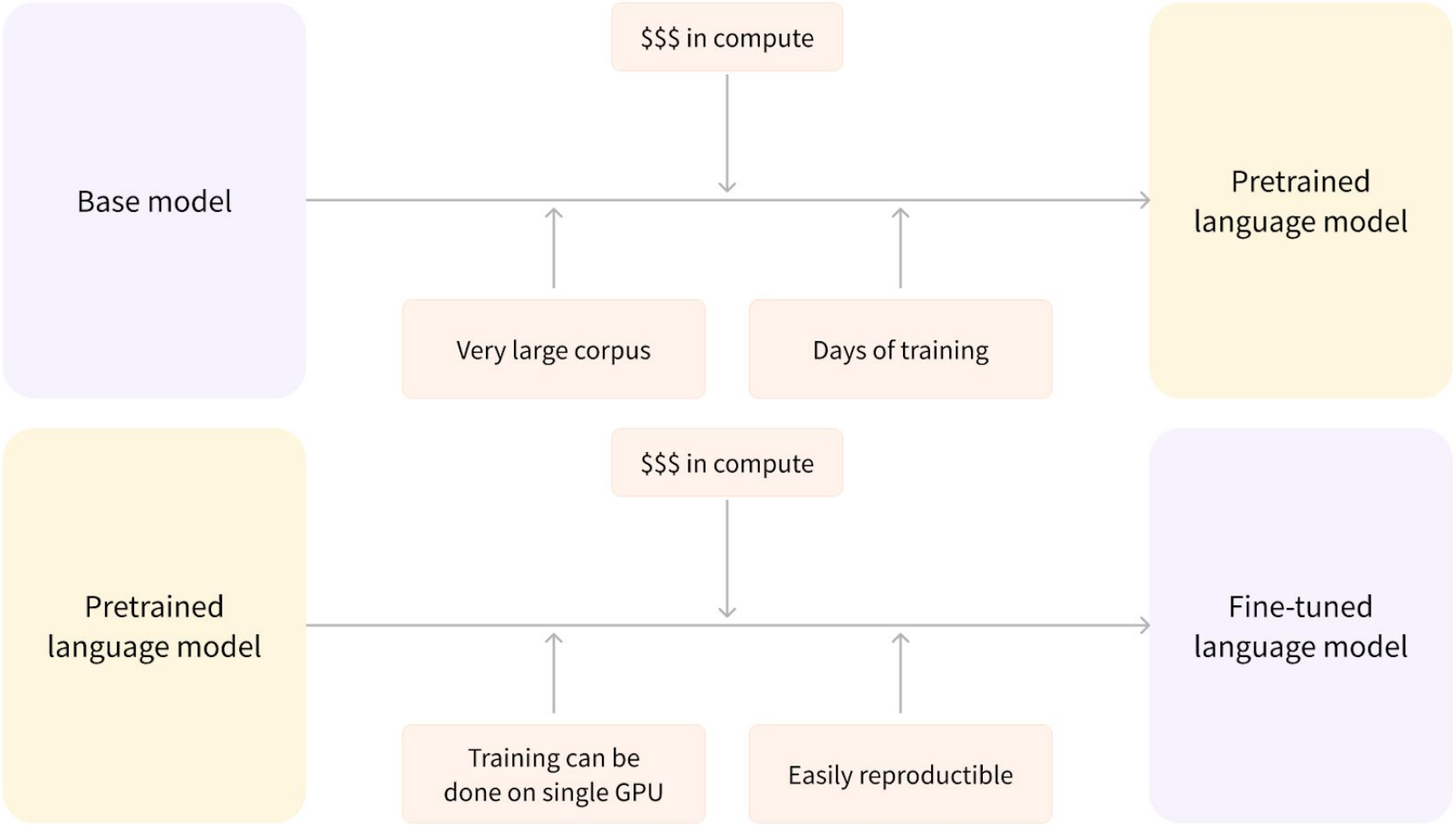
Parameters	Tokens
400 Million	8.0 Billion
1 Billion	20.2 Billion
10 Billion	205.1 Billion
67 Billion	1.5 Trillion
175 Billion	3.7 Trillion
280 Billion	5.9 Trillion
520 Billion	11.0 Trillion
1 Trillion	21.2 Trillion
10 Trillion	216.2 Trillion

AI Foundational model

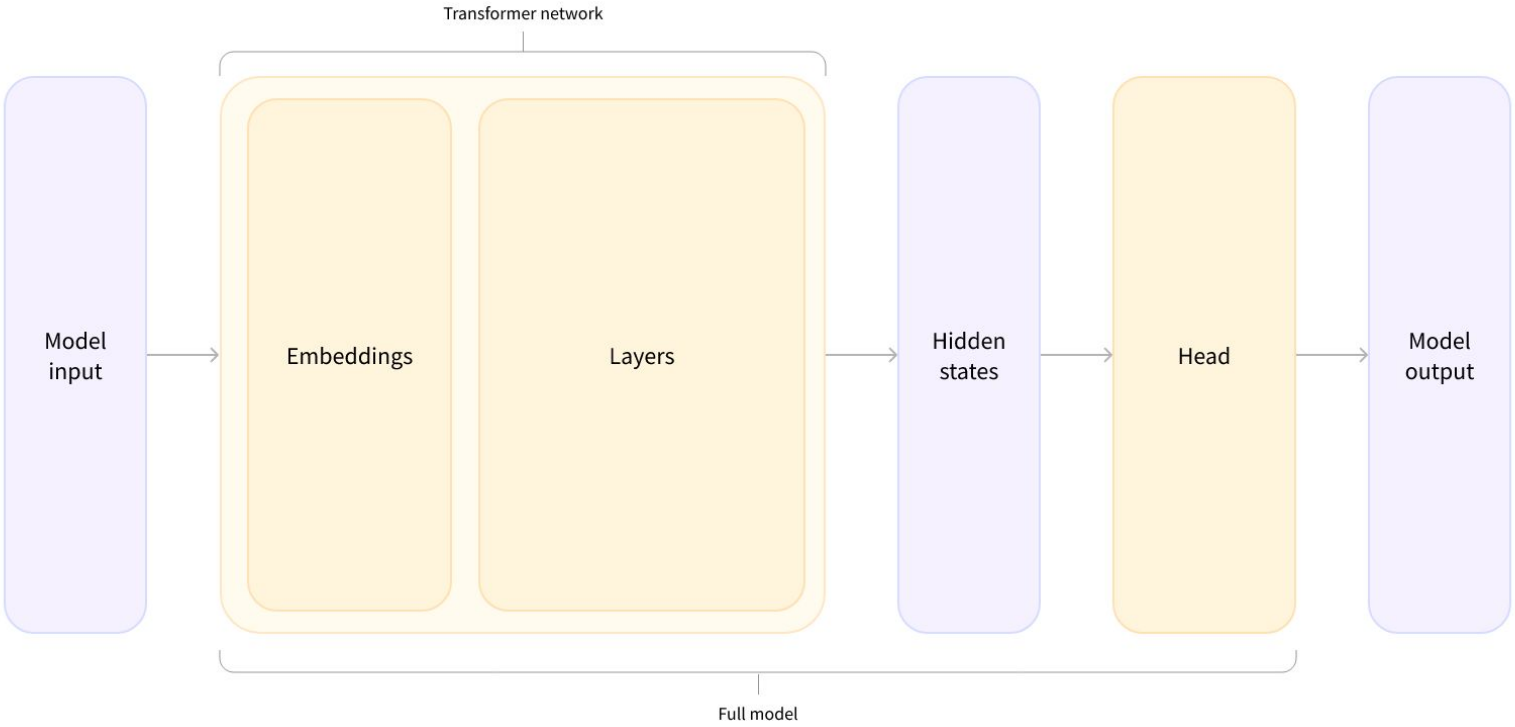
1. Pre-trained on Large Datasets
    a. Self-supervised (unsupervised)
    b. Massive dataset
2. Generalizable Representations
    a. Learn Meaningful features (semantics)
    b. Embeddings
3. Transferable
    a. Fine-tunable in small diverse datasets
    b. Multiple downstream tasks: Classification, Generation, Prediction

AI Foundational model

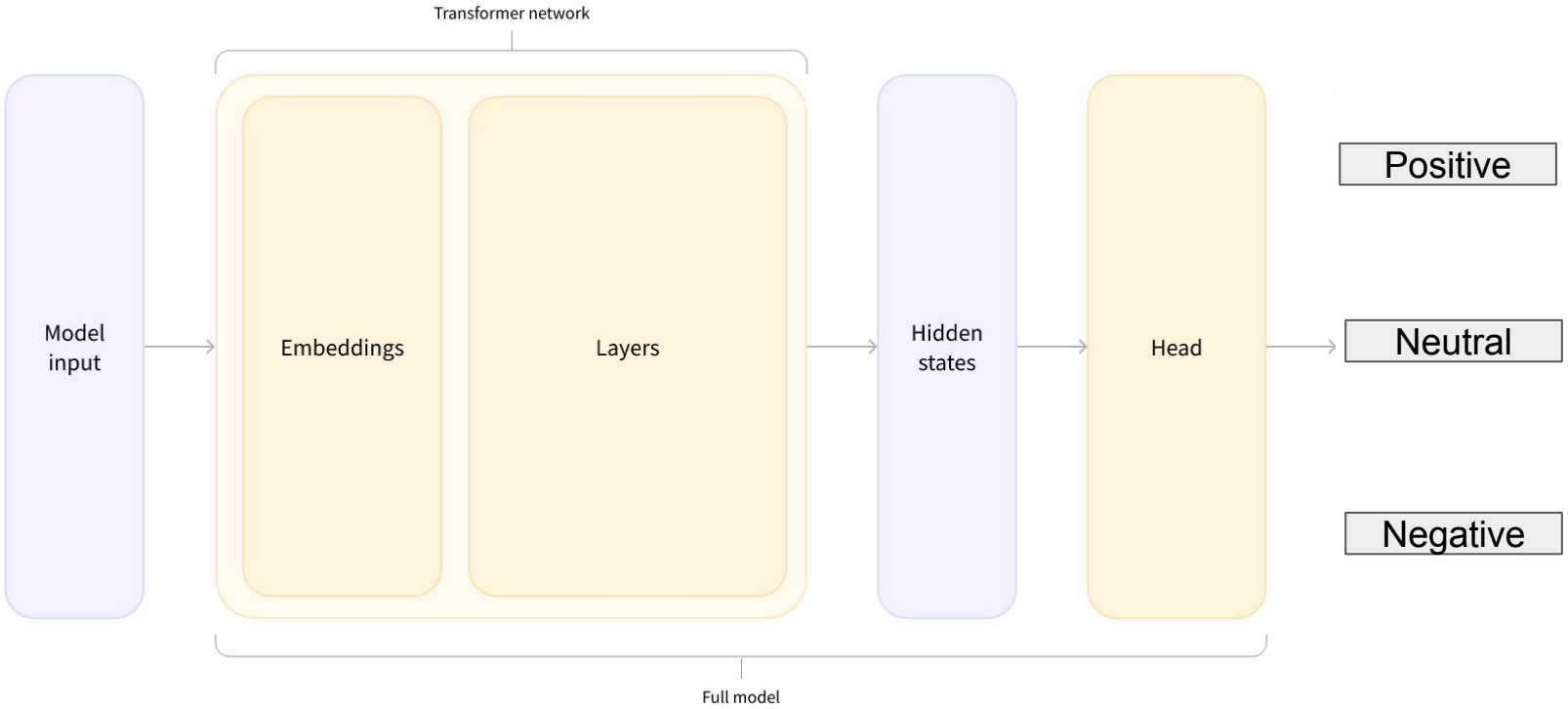
1. Pre-trained on Large Datasets
    a. Self-supervised (unsupervised)
    b. Massive dataset
2. Generalizable Representations
    a. Learn Meaningful features (semantics)
    b. Embeddings
3. **Transferable ✅**
    a. Fine-tunable in small diverse datasets
    b. Multiple downstream tasks: Classification, Generation, Regression


Base model
$$$ in compute
Very large corpus
Days of training
Pretrained language model
Pretrained language model
$$$ in compute
Training can be done on single GPU
Easily reproductible
Fine-tuned language model

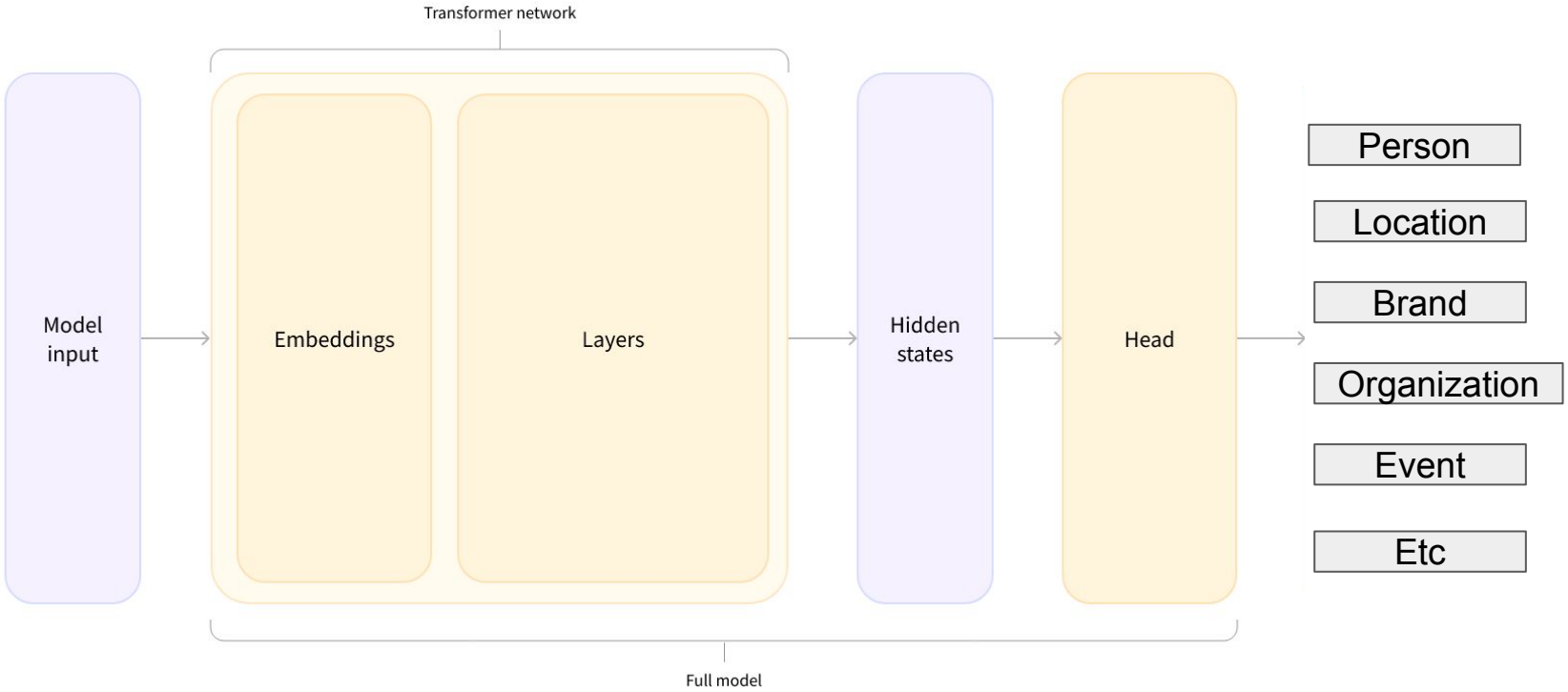
Transferable and fine tuned



Sentiment Analysis (Classification of full sentence)



NER (Named Entity Recognition) | For each token



Generation

✅ Chat - Instruct

✅ Translation

✅ Summarization

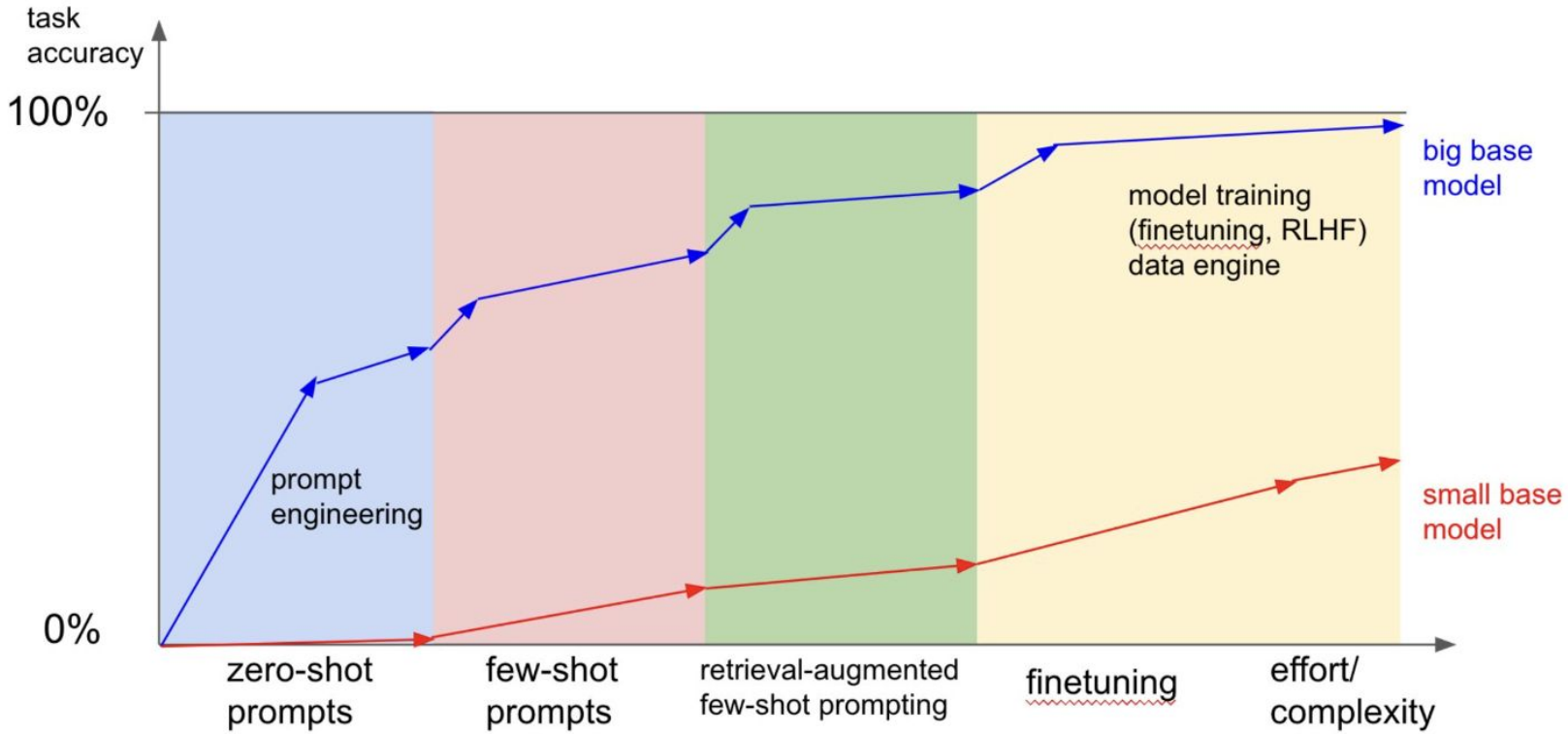
Regression Examples

Stock pricing movement prediction: Use news articles, earnings call transcripts, or SEC filings as input to predict stock returns or volatility.

Credit scoring: Use textual reports (e.g., customer descriptions, loan applications) to regress a risk score

Personality trait estimation: Based on user-written text, LLMs can output Big Five personality trait scores (continuous values).

Mental health screening: Text input (journals, social media) used to predict depression or anxiety scores.


task
accuracy
100%
0%
prompt
engineering
model training
(finetuning, RLHF)
data engine
big base
model
small base
model
zero-shot
prompts
few-shot
prompts
retrieval-augmented
few-shot prompting
finetuning
effort/
complexity

Masked Autoencoders are Scalable Visual Learners

“This paper shows that masked autoencoders (MAE) are scalable self-supervised learners for computer vision”

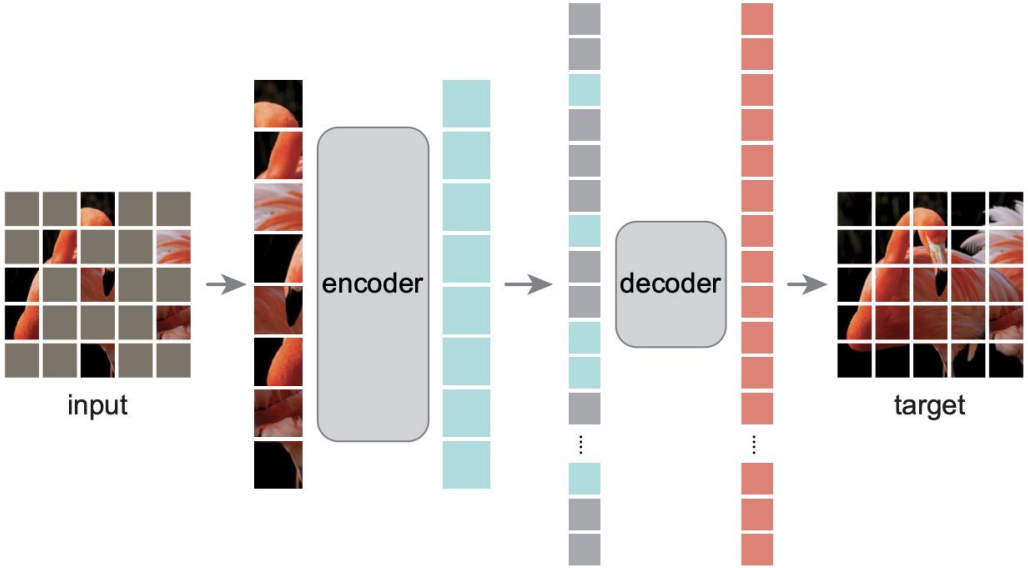


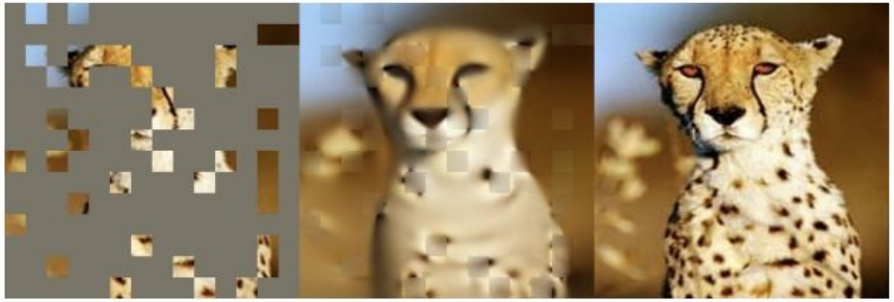
Figure 1. **Our MAE architecture**. During pre-training, a large random subset of image patches (*e.g.*, 75%) is masked out. The encoder is applied to the small subset of *visible patches*. Mask tokens are introduced *after* the encoder, and the full set of encoded patches and mask tokens is processed by a small decoder that reconstructs the original image in pixels. After pre-training, the decoder is discarded and the encoder is applied to uncorrupted images (full sets of patches) for recognition tasks.

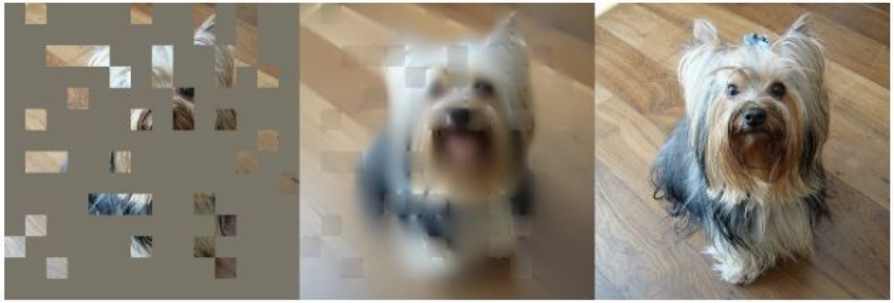
¿Porque tanto % se enmascara?

Language:
Human-generated, Highly semantic and high density

Images: Natural Signals, heavy spatial redundancy

"Un patch faltante puede ser recuperado de los patches vecinos con poca comprensión semántica global de las partes, objetos y escenas"



original
mask 75%
mask 85%
mask 95%

MAE Encoder

- ViT
- Solo aplicado en los patches visibles (Eficiencia)
- Embeddings de los patches con una proyección lineal
- Positional embeddings added

MAE Decoder

- Tokens visibles del encoder
- Tokens enmascarados
- Positional embeddings a todos
- Una serie de bloques de transformers
- Decoder solo para pre-training (Tarea de reconstrucción)
- Arquitectura independiente del encoder (flexibilidad y eficiencia)
- 10% de procesamiento por token respecto al Encoder
- La última capa es lineal (dimensión transformer → cantidad de pixels/patch)
- Loss = MSE (Solo en patches enmascarados)

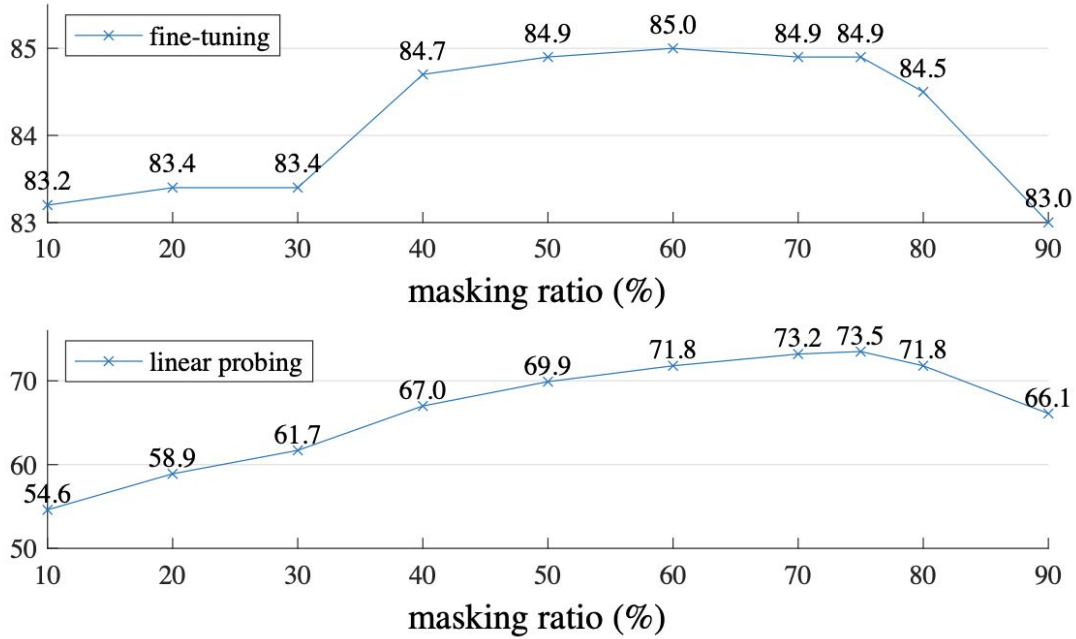


Figure 5. **Masking ratio**. A high masking ratio (75%) works well for both fine-tuning (top) and linear probing (bottom). The y-axes are ImageNet-1K validation accuracy (%) in all plots in this paper.

scratch, original [16]	scratch, our impl.	baseline MAE
76.5	82.5	84.9

blocks	ft	lin
1	84.8	65.5
2	**84.9**	70.0
4	**84.9**	71.9
8	**84.9**	**73.5**
12	84.4	73.3

(a) **Decoder depth**. A deep decoder can improve linear probing accuracy.

dim	ft	lin
128	**84.9**	69.1
256	84.8	71.3
512	**84.9**	**73.5**
768	84.4	73.1
1024	84.3	73.1

(b) **Decoder width**. The decoder can be narrower than the encoder (1024-d).

case	ft	lin	FLOPs
encoder w/ [M]	84.2	59.6	3.3×
encoder w/o [M]	**84.9**	**73.5**	**1**×

(c) **Mask token**. An encoder without mask tokens is more accurate and faster (Table 2).

case	ft	lin
pixel (w/o norm)	84.9	73.5
pixel (w/ norm)	**85.4**	**73.9**
PCA	84.6	72.3
dVAE token	85.3	71.6

(d) **Reconstruction target**. Pixels as reconstruction targets are effective.

case	ft	lin
none	84.0	65.7
crop, fixed size	84.7	73.1
crop, rand size	**84.9**	**73.5**
crop + color jit	84.3	71.9

(e) **Data augmentation**. Our MAE works with minimal or no augmentation.

case	ratio	ft	lin
random	75	**84.9**	**73.5**
block	50	83.9	72.3
block	75	82.8	63.9
grid	75	84.0	66.0

(f) **Mask sampling**. Random sampling works the best. See Figure 6 for visualizations.

encoder	dec. depth	ft acc	hours	speedup
ViT-L, w/ [M]	8	84.2	42.4	-
ViT-L	8	84.9	15.4	2.8×
ViT-L	1	84.8	11.6	**3.7**×
ViT-H, w/ [M]	8	-	119.6†	-
ViT-H	8	85.8	34.5	3.5×
ViT-H	1	85.9	29.3	**4.1**×

Table 2. **Wall-clock time** of our MAE training (800 epochs), benchmarked in 128 TPU-v3 cores with TensorFlow. The speedup is relative to the entry whose encoder has mask tokens (gray). The decoder width is 512, and the mask ratio is 75%. †: This entry is estimated by training ten epochs.

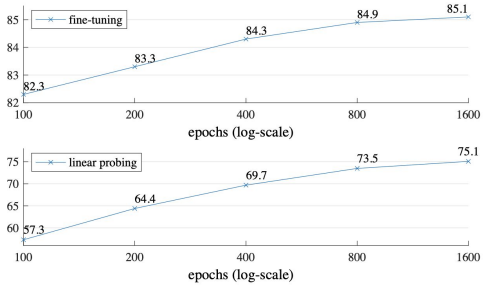


Figure 7. **Training schedules**. A longer training schedule gives a noticeable improvement. Here each point is a full training schedule. The model is ViT-L with the default setting in Table 1.

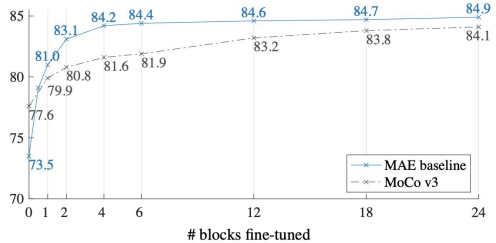


Figure 9. **Partial fine-tuning** results of ViT-L w.r.t. the number of fine-tuned Transformer blocks under the default settings from Table 1. Tuning 0 blocks is linear probing; 24 is full fine-tuning. Our MAE representations are less linearly separable, but are consistently better than MoCo v3 if one or more blocks are tuned.

CLIP (Contrastive Language-Image Pre-Training)

Learning Transferable Visual Models From Natural Language Supervision

repo openai

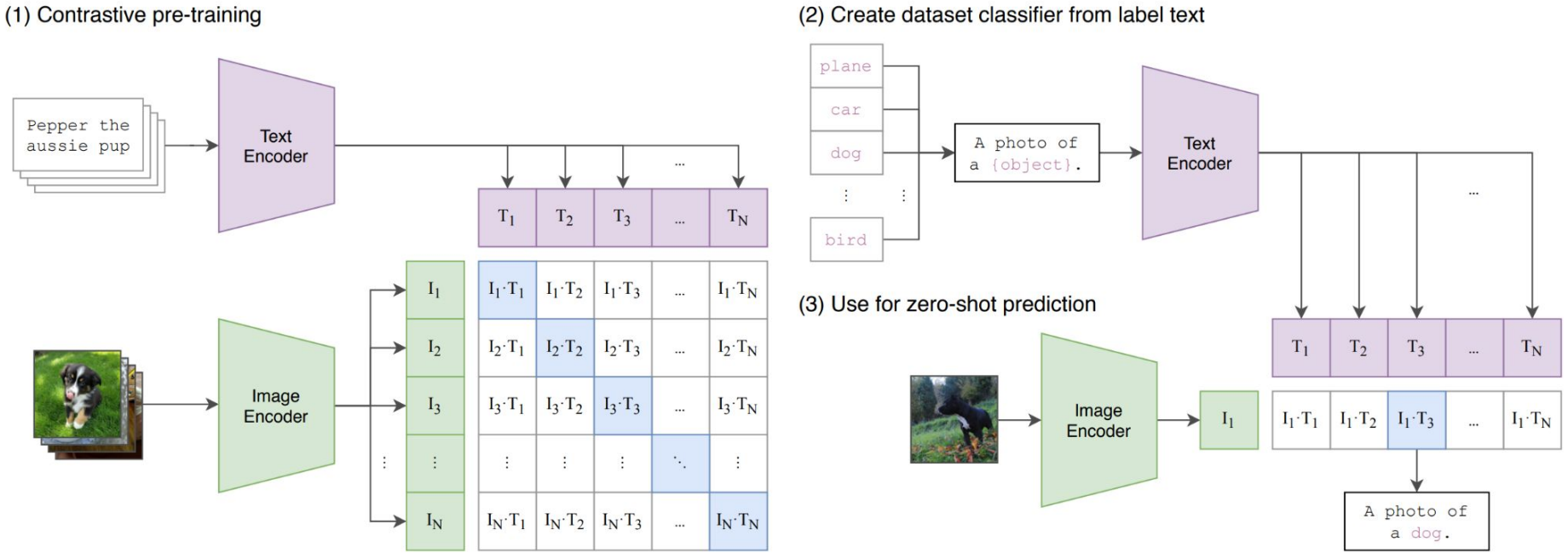


Figure 1. Summary of our approach. While standard image models jointly train an image feature extractor and a linear classifier to predict some label, CLIP jointly trains an image encoder and a text encoder to predict the correct pairings of a batch of (image, text) training examples. At test time the learned text encoder synthesizes a zero-shot linear classifier by embedding the names or descriptions of the target dataset's classes.

Mini batch = 32K

Datasets:

Training: 400 millones de imágenes y texto (SOTA)

Benchmark:

- 30 CV datasets
- Tareas como: OCR, Reconocimiento de acciones (videos), Geo y otros
- Competitivo con fully supervised sin necesidad de entrenamiento específico
- Iguala la performance de Resnet50 en Imagenet (zero shot sin entrenar con 1.28 millones de imágenes)

Contrastive Loss (Paper - Yan Lecun et all)

$$D_W(\vec{X}_1, \vec{X}_2) = \|G_W(\vec{X}_1) - G_W(\vec{X}_2)\|_2 \quad (1)$$

$$\mathcal{L}(W) = \sum_{i=1}^{P} L(W, (Y, \vec{X}_1, \vec{X}_2)^i) \quad (2)$$

$$L(W, (Y, \vec{X}_1, \vec{X}_2)^i) = (1 - Y) L_S\left(D_W^i\right) + Y L_D\left(D_W^i\right) \quad (3)$$

$$L(W, Y, \vec{X}_1, \vec{X}_2) = (1 - Y)\frac{1}{2}(D_W)^2 + (Y)\frac{1}{2}\{max(0, m - D_W)\}^2 \quad (4)$$

Contrastive loss CLIP

- N (image, text) pairs (32K)
- Jointly training an image encoder and text encoder
- **Maximize** the **cosine similarity** of the image and text embeddings of the N **real pairs** in the batch
- **Minimizing** the **cosine similarity** of the embeddings of the N2 − N **incorrect** pairings.
- We optimize a **symmetric cross entropy loss** over these similarity scores.

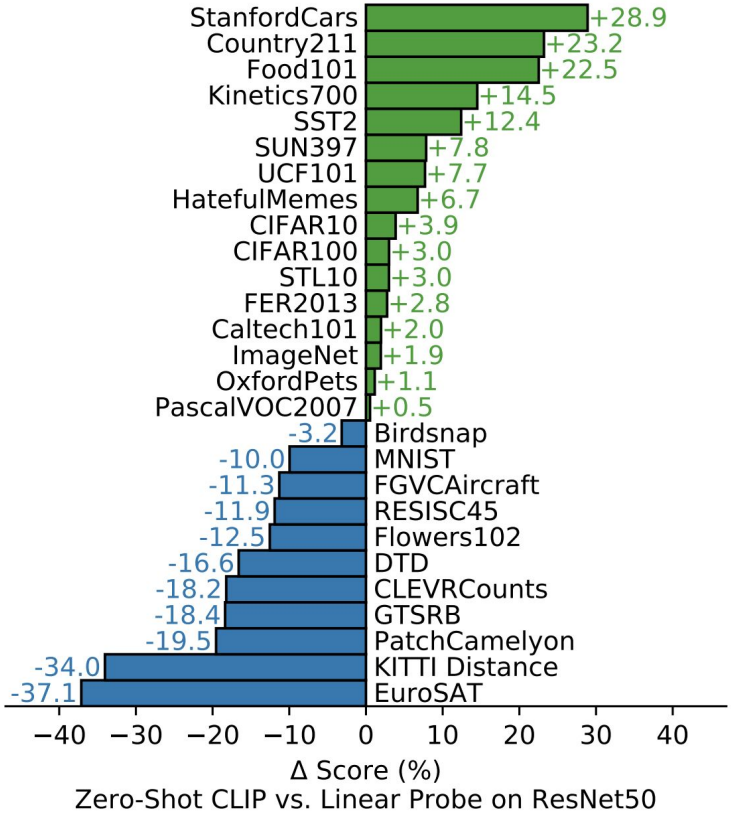


Figure 5. **Zero-shot CLIP is competitive with a fully supervised baseline.** Across a 27 dataset eval suite, a zero-shot CLIP classifier outperforms a fully supervised linear classifier fitted on ResNet-50 features on 16 datasets, including ImageNet.

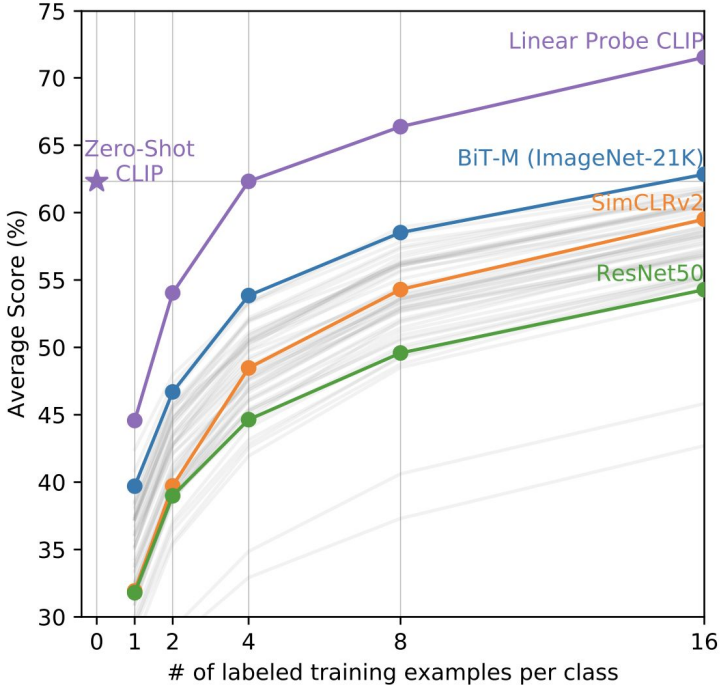


Figure 6. **Zero-shot CLIP outperforms few-shot linear probes.** Zero-shot CLIP matches the average performance of a 4-shot linear classifier trained on the same feature space and nearly matches the best results of a 16-shot linear classifier across publicly available models. For both BiT-M and SimCLRv2, the best performing model is highlighted. Light gray lines are other models in the eval suite. The 20 datasets with at least 16 examples per class were used in this analysis.

		Food101	CIFAR10	CIFAR100	Birdsnap	SUN397	Stanford Cars	FGVC Aircraft	VOC2007	DTD	Oxford Pets	Caltech101	Flowers102	MNIST	FER2013	STL10	EuroSAT	RESISC45	GTSRB	KITTI	Country211	PCam	UCF101	Kinetics700	CLEVR	HatefulMemes	Rendered SST2	ImageNet
CLIP-ResNet	RN50	81.1	75.6	41.6	32.6	59.6	55.8	19.3	82.1	41.7	85.4	82.1	65.9	66.6	42.2	94.3	41.1	54.2	35.2	42.2	16.1	57.6	63.6	43.5	20.3	59.7	56.9	59.6
	RN101	83.9	81.0	49.0	37.2	59.9	62.3	19.5	82.4	43.9	86.2	85.1	65.7	59.3	45.6	96.7	33.1	58.5	38.3	33.3	16.9	55.2	62.2	46.7	28.1	61.1	64.2	62.2
	RN50x4	86.8	79.2	48.9	41.6	62.7	67.9	24.6	83.0	49.3	88.1	86.0	68.0	75.2	51.1	96.4	35.0	59.2	35.7	26.0	20.2	57.5	65.5	49.0	17.0	58.3	66.6	65.8
	RN50x16	90.5	82.2	54.2	45.9	65.0	72.3	30.3	82.9	52.8	89.7	87.6	71.9	80.0	56.0	97.8	40.3	64.4	39.6	33.9	24.0	62.5	68.7	53.4	17.6	58.9	67.6	70.5
	RN50x64	91.8	86.8	61.3	48.9	66.9	76.0	35.6	83.8	53.4	93.4	90.6	77.3	90.8	61.0	98.3	59.4	69.7	47.9	33.2	29.6	65.0	74.1	56.8	27.5	62.1	70.7	73.6
CLIP-ViT	B/32	84.4	91.3	65.1	37.8	63.2	59.4	21.2	83.1	44.5	87.0	87.9	66.7	51.9	47.3	97.2	49.4	60.3	32.2	39.4	17.8	58.4	64.5	47.8	24.8	57.6	59.6	63.2
	B/16	89.2	91.6	68.7	39.1	65.2	65.6	27.1	83.9	46.0	88.9	89.3	70.4	56.0	52.7	98.2	54.1	65.5	43.3	44.0	23.3	48.1	69.8	52.4	23.4	61.7	59.8	68.6
	L/14	92.9	96.2	77.9	48.3	67.7	77.3	36.1	84.1	55.3	93.5	92.6	78.7	87.2	57.5	99.3	59.9	71.6	50.3	23.1	32.7	58.8	76.2	60.3	24.3	63.3	64.0	75.3
	L/14-336px	93.8	95.7	77.5	49.5	68.4	78.8	37.2	84.3	55.7	93.5	92.8	78.3	88.3	57.7	99.4	59.6	71.7	52.3	21.9	34.9	63.0	76.9	61.3	24.8	63.3	67.9	76.2

Table 11. Zero-shot performance of CLIP models over 27 datasets.