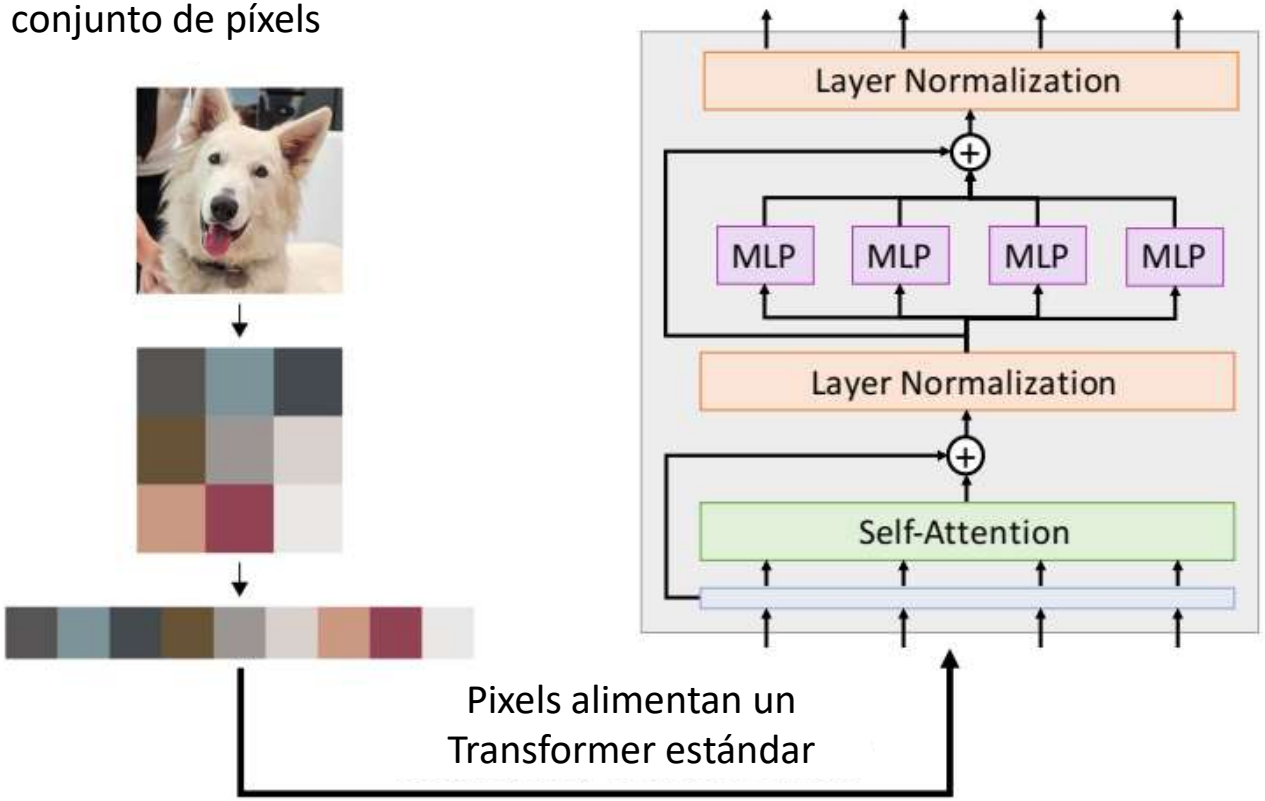
Visión Computacional

Ivan Sipiran

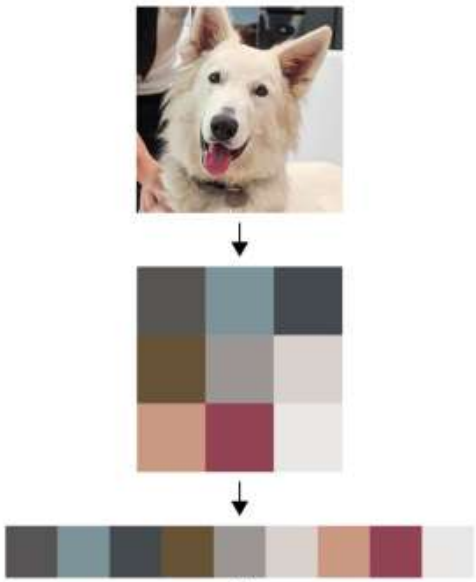
Idea muy básica

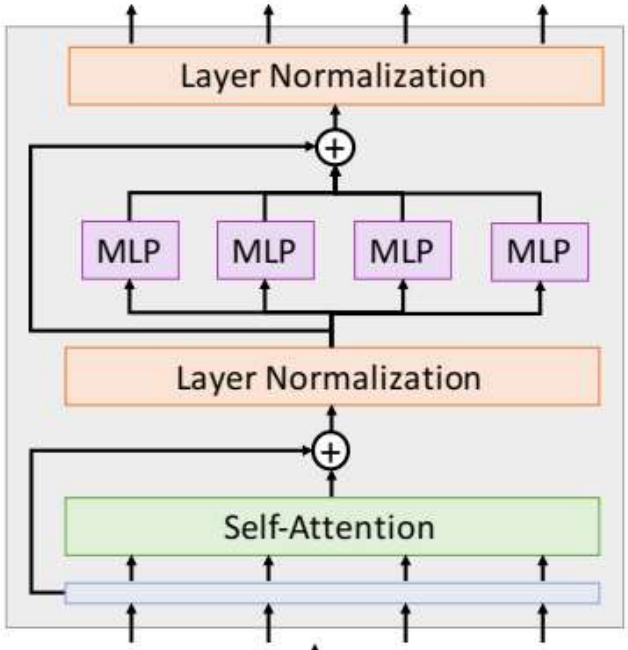
Tratar una imagen como un conjunto de píxels



Idea muy básica

Tratar una imagen como un conjunto de píxels





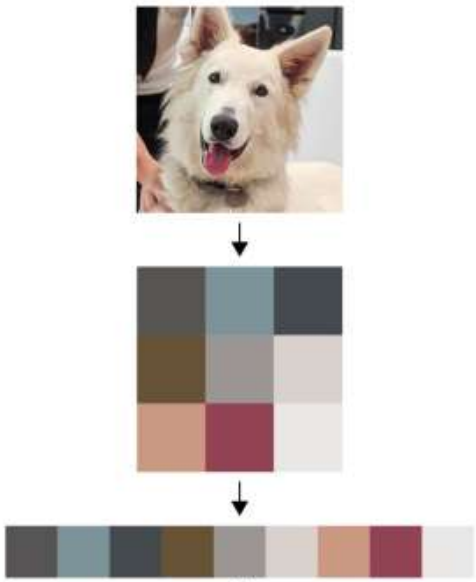
Pixels alimentan un Transformer estándar

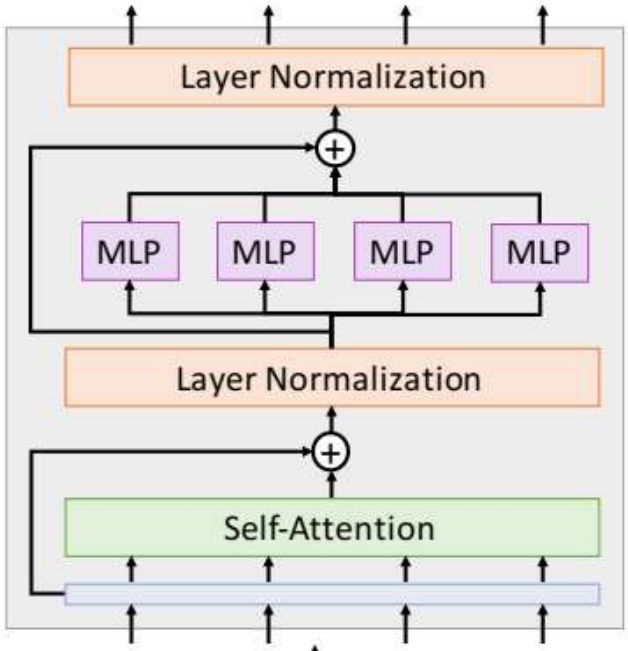
Problema: memoria!

Imagen de RxR necesita R^4 elementos en matriz de atención

Idea muy básica

Tratar una imagen como un conjunto de píxels





Pixels alimentan un Transformer estándar

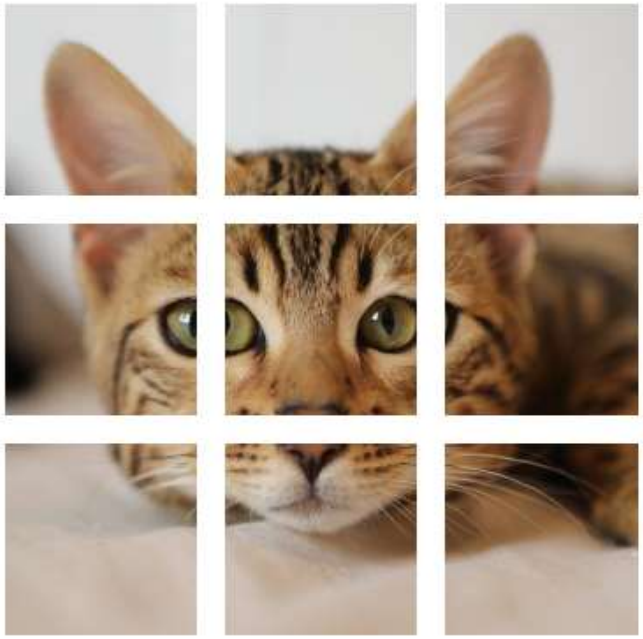
Problema: memoria!

Imagen de RxR necesita R^4 elementos en matriz de atención

R=128, 48 capas, 16 cabezas de atención: 768GB para matrices de atención de una sola imagen

Transformer en parches

Transformer en parches



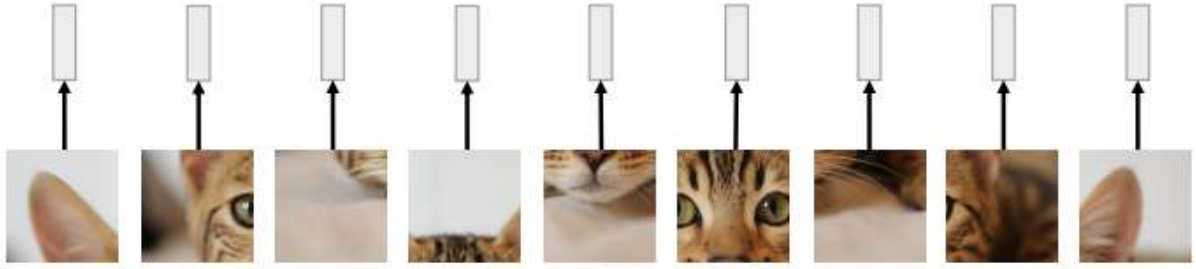
Transformer en parches

N parches de entrada
Tamaño: 3 x 16 x 16

Transformer en parches

Proyección a vector de D dimensiones

N parches de entrada
Tamaño: 3 x 16 x 16

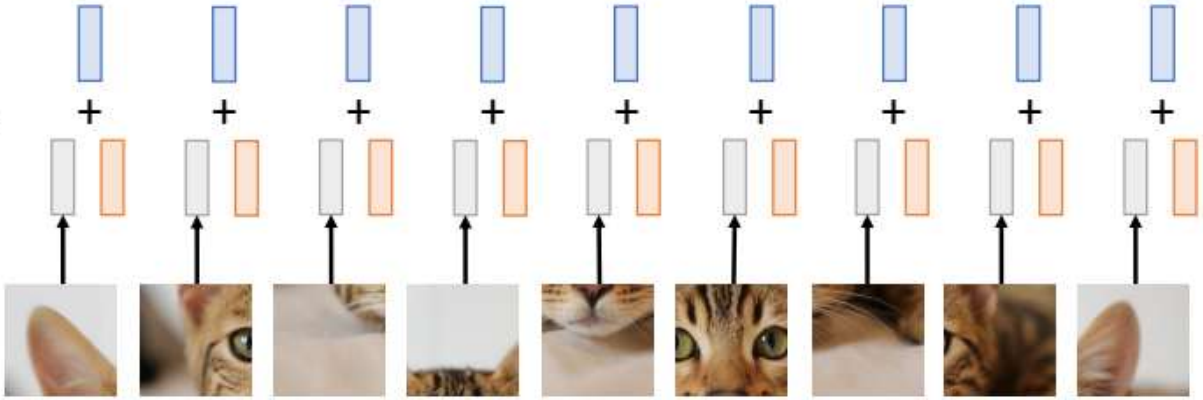


Transformer en parches

Positional embedding

Proyección a vector de D dimensiones

N parches de entrada
Tamaño: 3 x 16 x 16



Transformer en parches

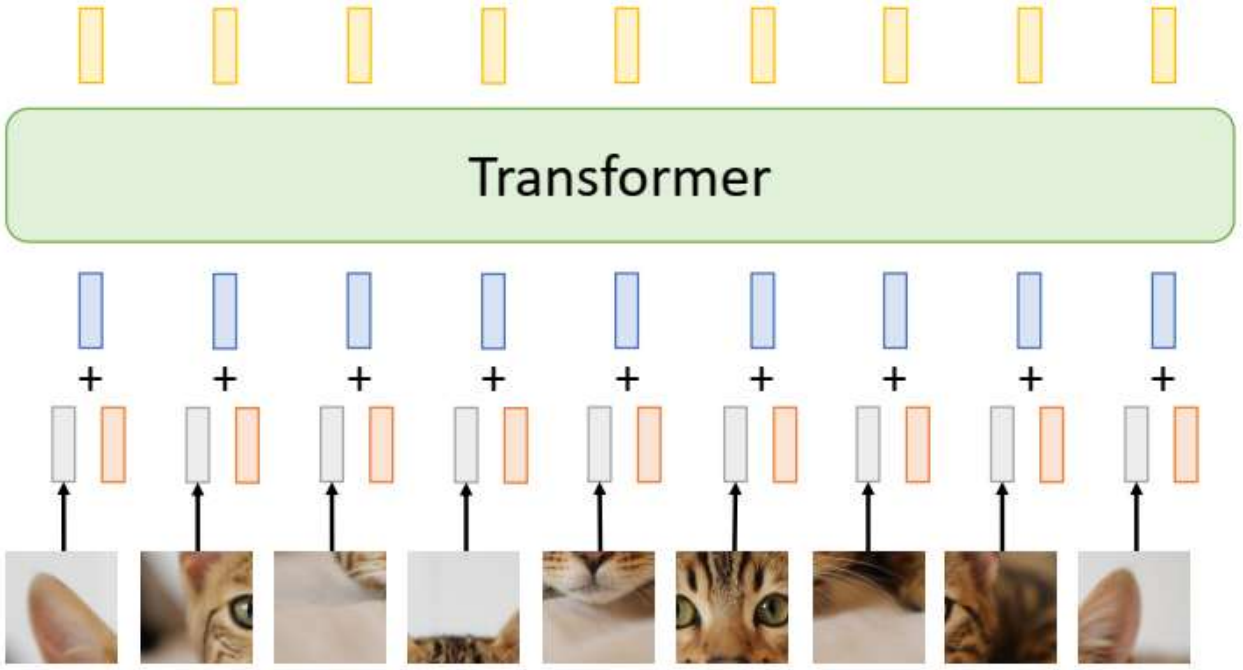
Vector de salida

El mismo Transformer que NLP!

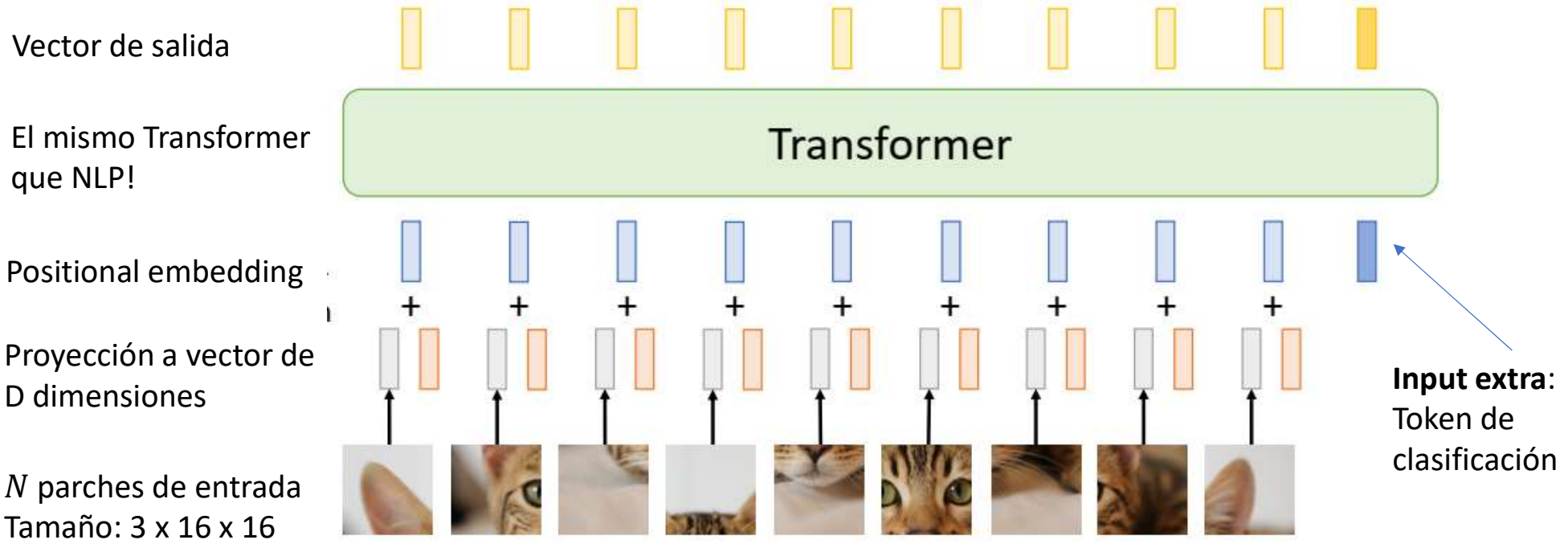
Positional embedding

Proyección a vector de D dimensiones

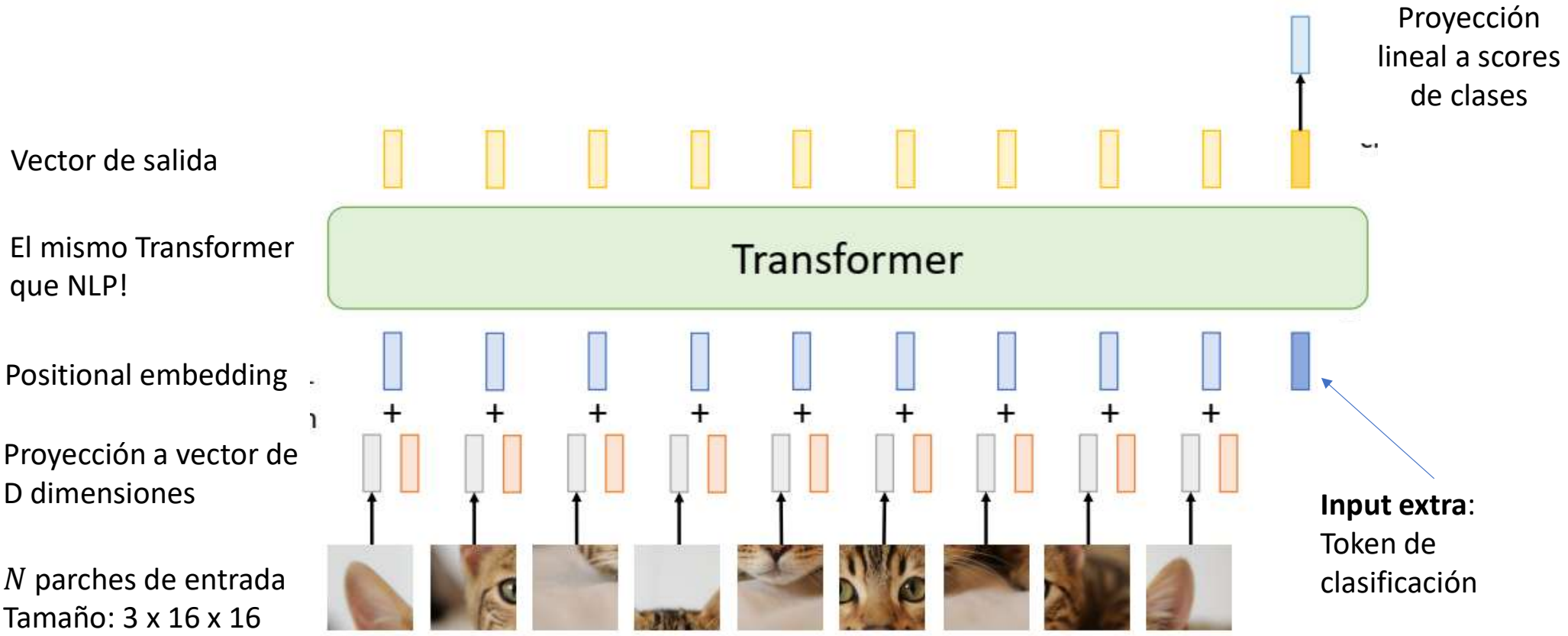
N parches de entrada
Tamaño: 3 x 16 x 16



Transformer en parches

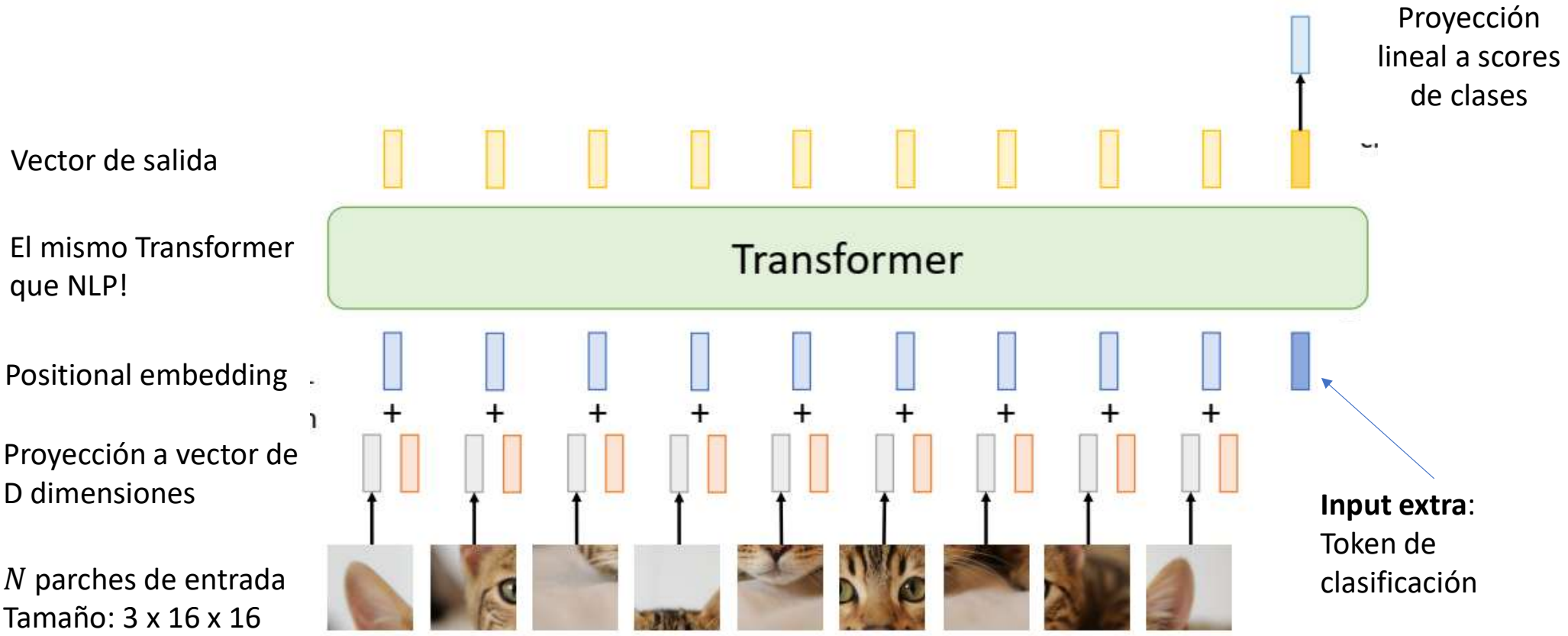


Transformer en parches



Transformer en parches

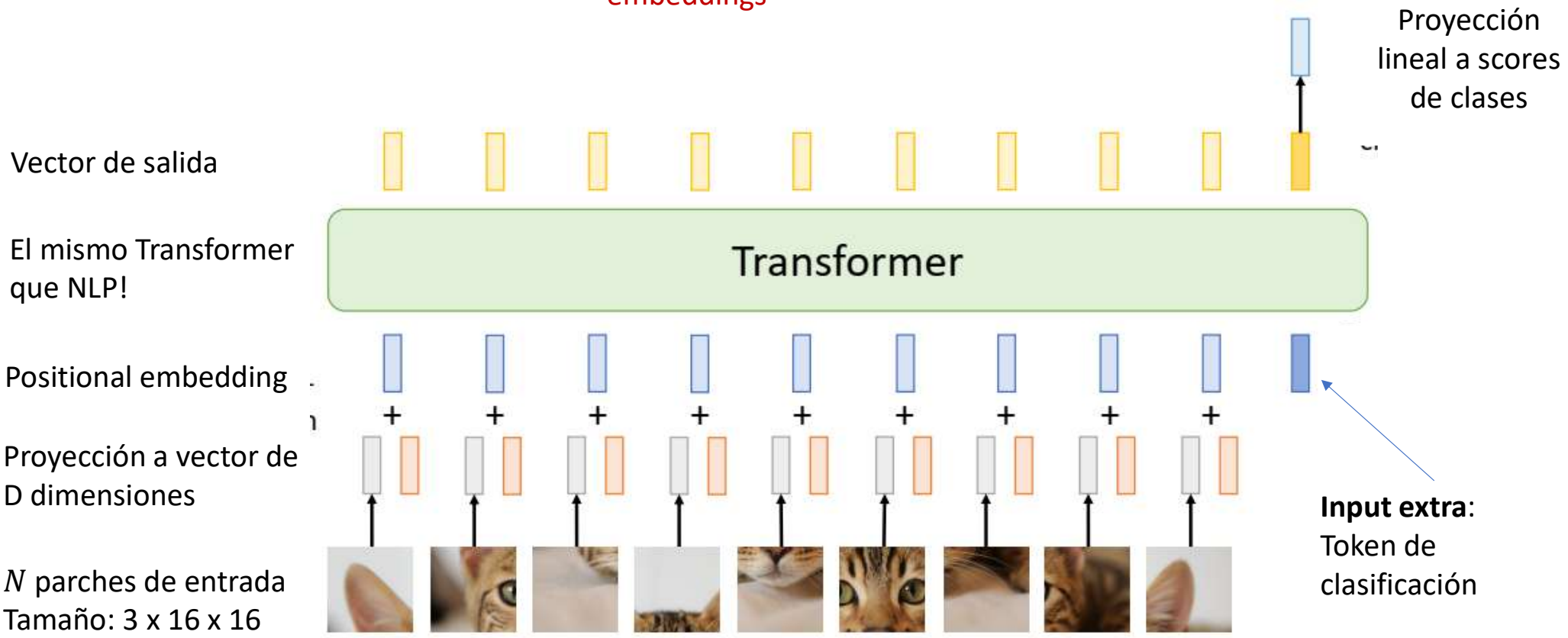
Modelo de visión sin convoluciones!



Transformer en parches

Modelo de visión sin convoluciones!

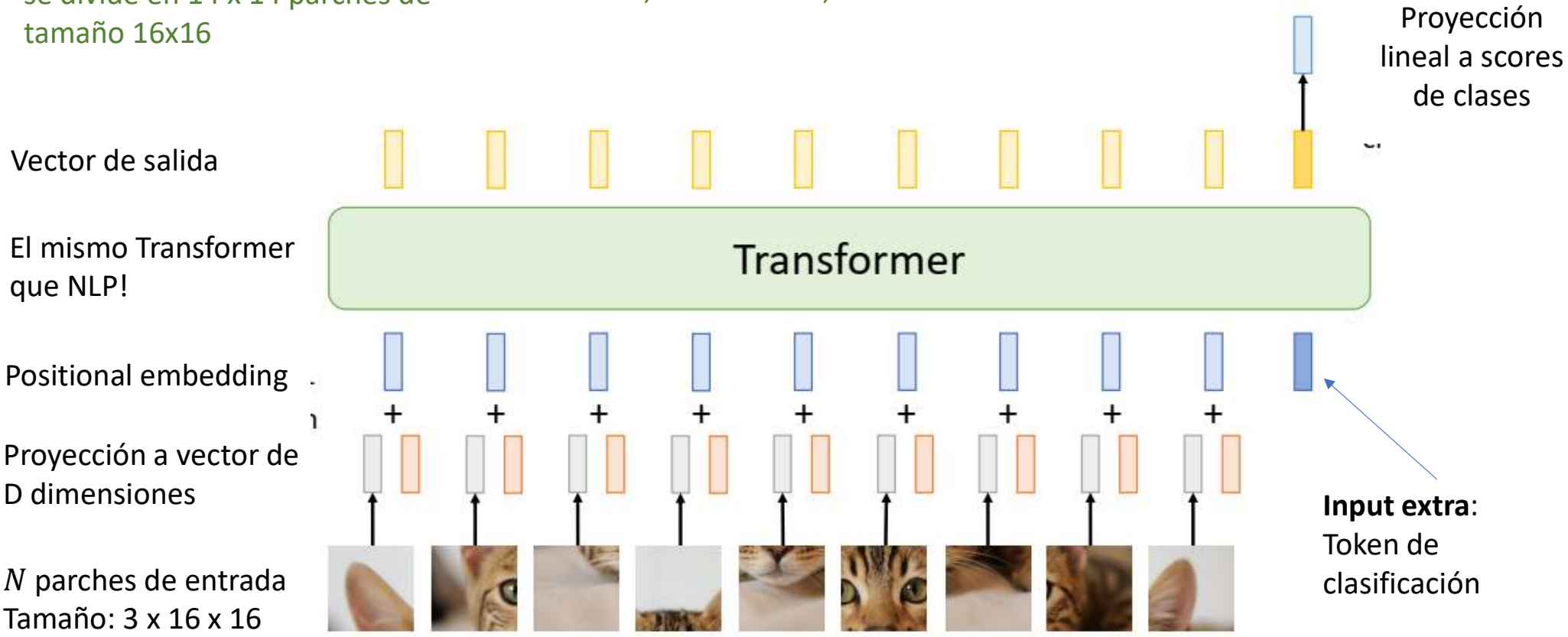
No del todo: la primera capa es Conv2D para embeddings



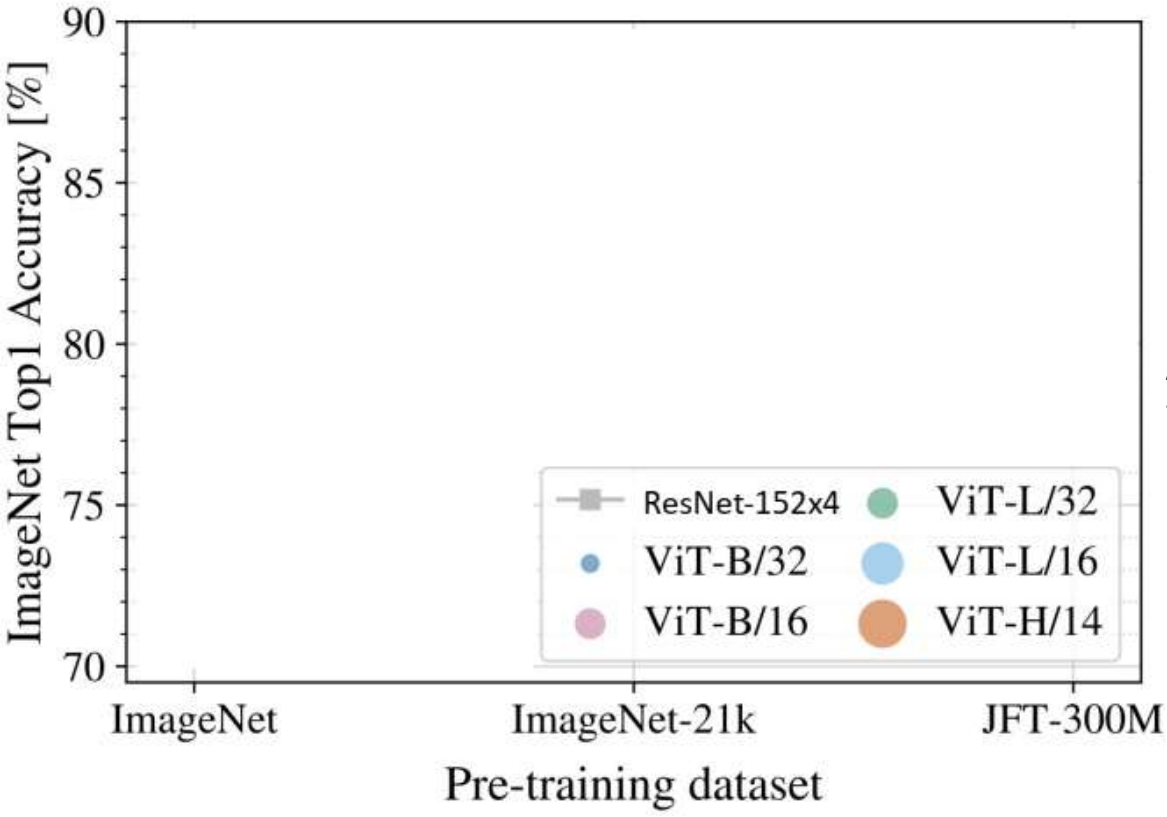
Transformer en parches

En la práctica: Imagen de 224x224, se divide en 14 x 14 parches de tamaño 16x16

Cada matriz de atención tiene 38,416 entradas, con 150KB



ViT vs. ResNets



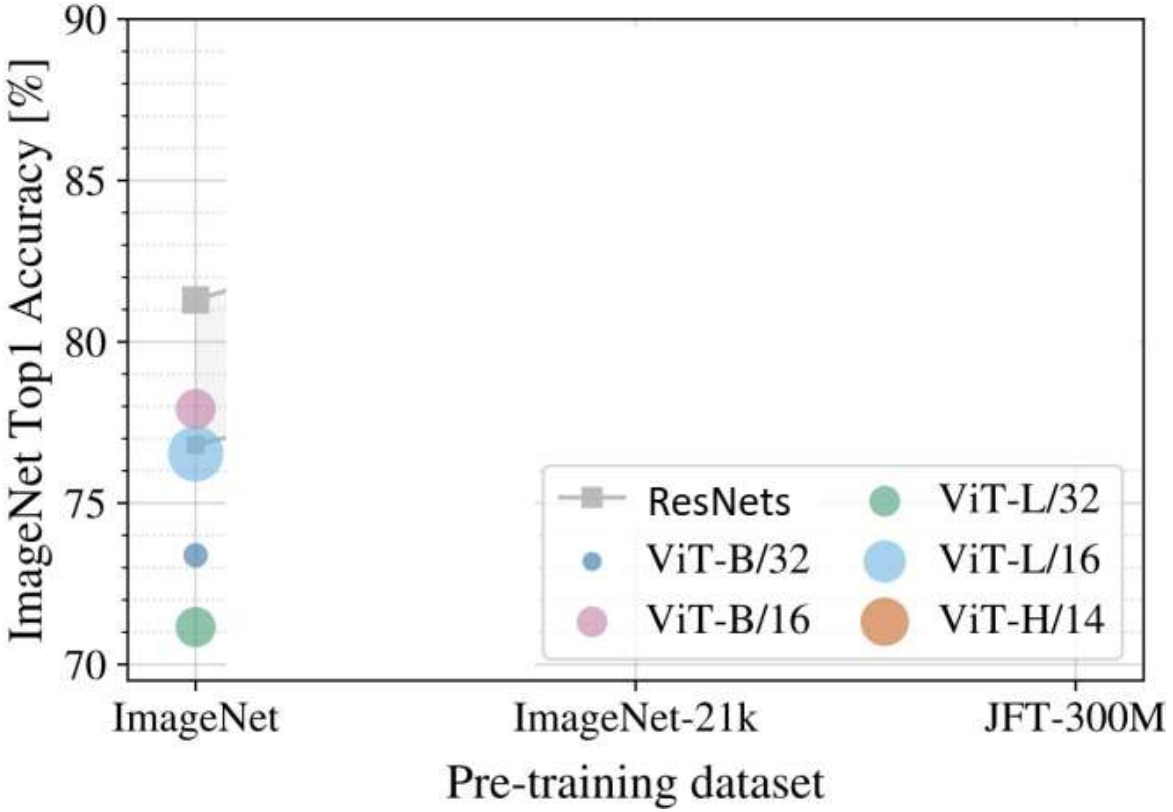
B: base
L: Large
H: huge

/32, /16, /14 es el tamaño del patch. Mientras más pequeño es el patch, el modelo es más grande

ViT vs. ResNets

ImageNet tiene 1k clases y 1.2M imágenes

Entrenado en ImageNet ViT no es mejor que ResNets



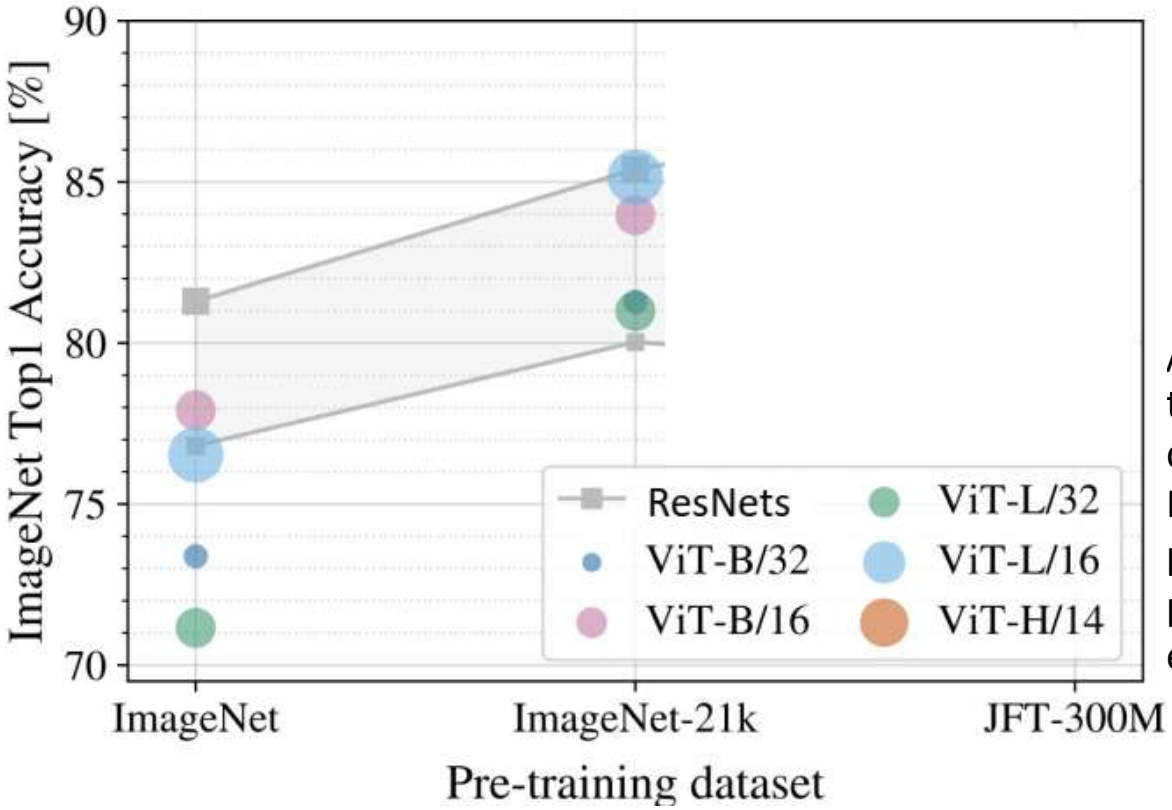
B: base
L: Large
H: huge

/32, /16, /14 es el tamaño del patch. Mientras más pequeño es el patch, el modelo es más grande

ViT vs. ResNets

ImageNet-21k tiene 21k clases y 14M imágenes

Pre-entrenamiento con ImageNet-21k hace que ViT alcance el performance de ResNets grandes



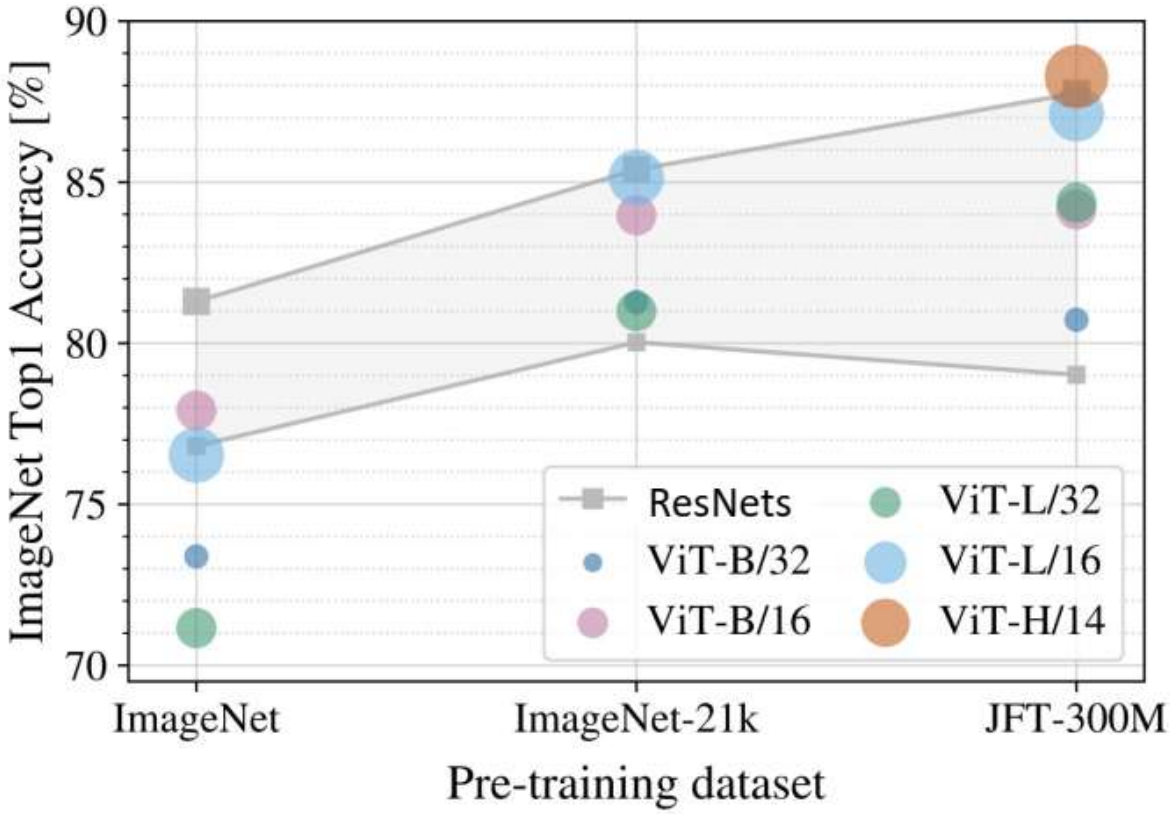
B: base
L: Large
H: huge

/32, /16, /14 es el tamaño del patch. Mientras más pequeño es el patch, el modelo es más grande

ViT vs. ResNets

JFT-300M dataset interno de Google con 300M de imágenes etiquetadas

Pre-entrenamiento con JFT-300M hace que ViT supere a ResNets



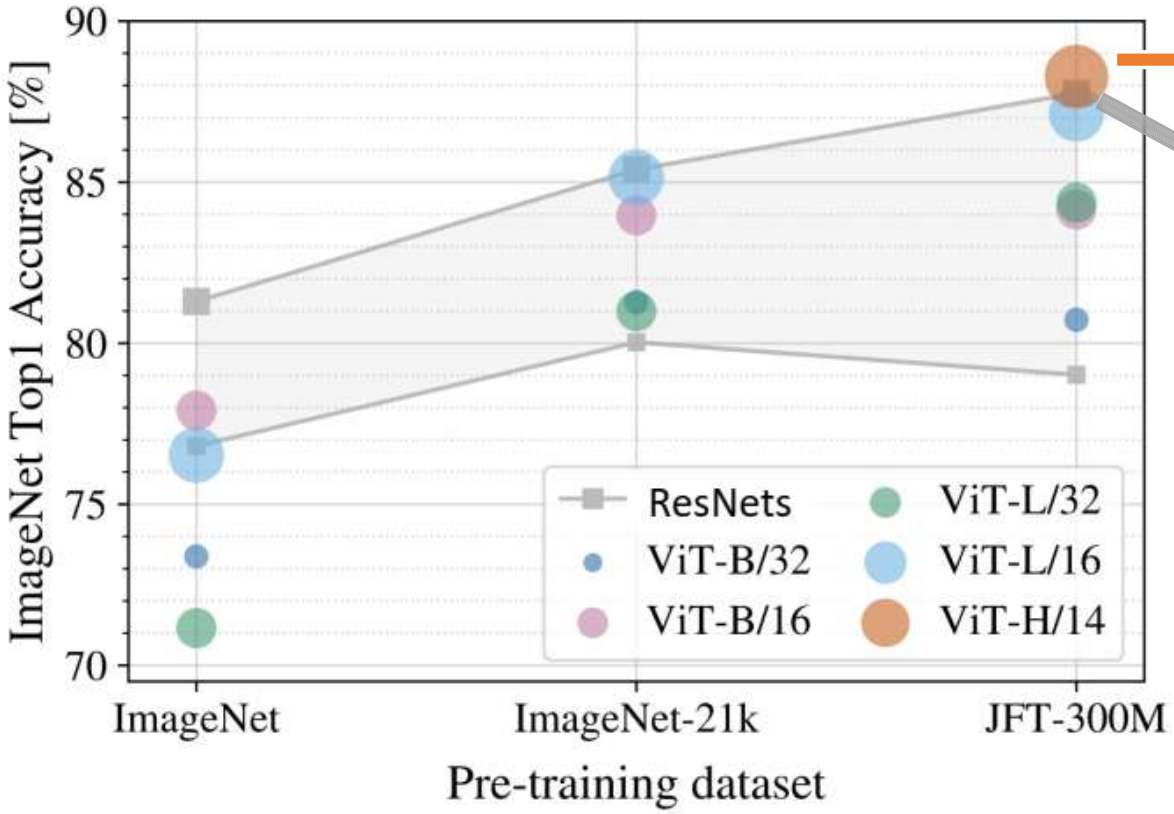
B: base
L: Large
H: huge

/32, /16, /14 es el tamaño del patch. Mientras más pequeño es el patch, el modelo es más grande

ViT vs. ResNets

JFT-300M dataset interno de Google con 300M de imágenes etiquetadas

Pre-entrenamiento con JFT-300M hace que ViT supere a ResNets



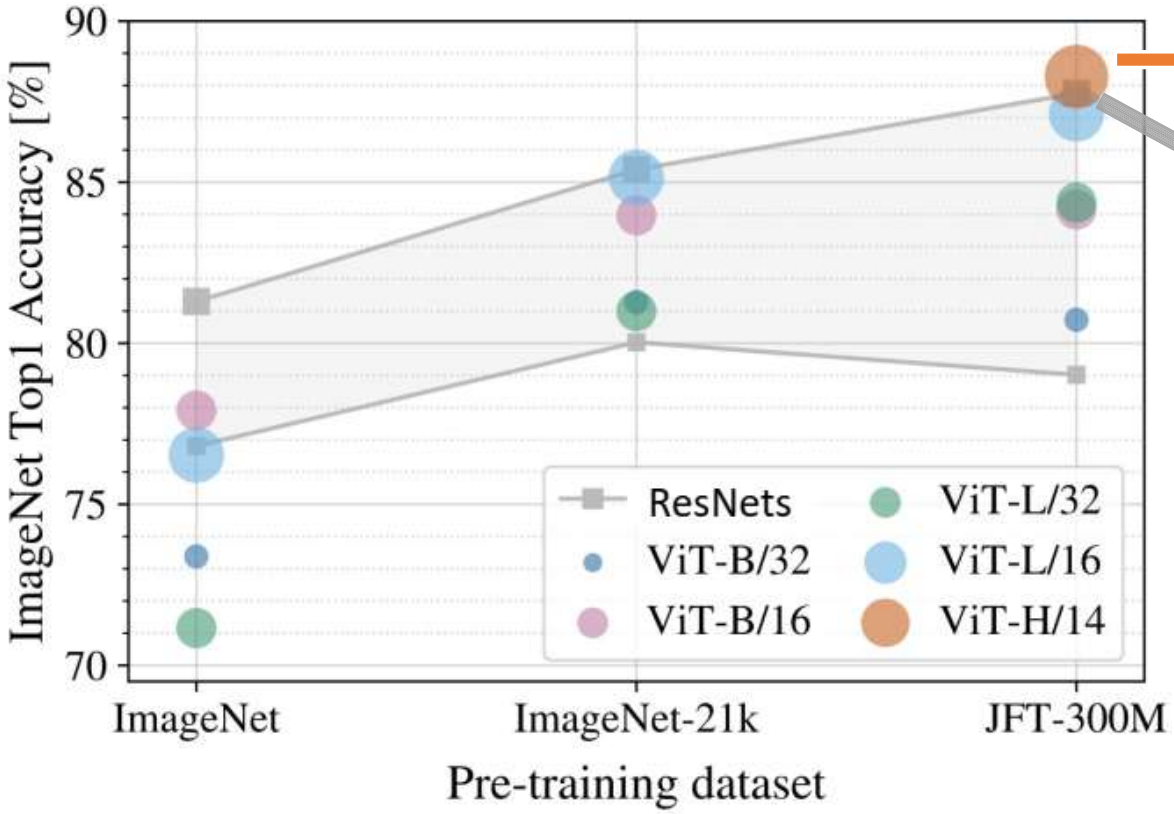
ViT: 2.5k TPU-v3 días de training

ResNets: 9.9k TPU-v3 días de training

ViT hace uso más eficiente de GPUs

ViT vs. ResNets

ViT tiene “menos vías inductivo” que ConvNets, por eso necesitamos más datos para pre-entrenar



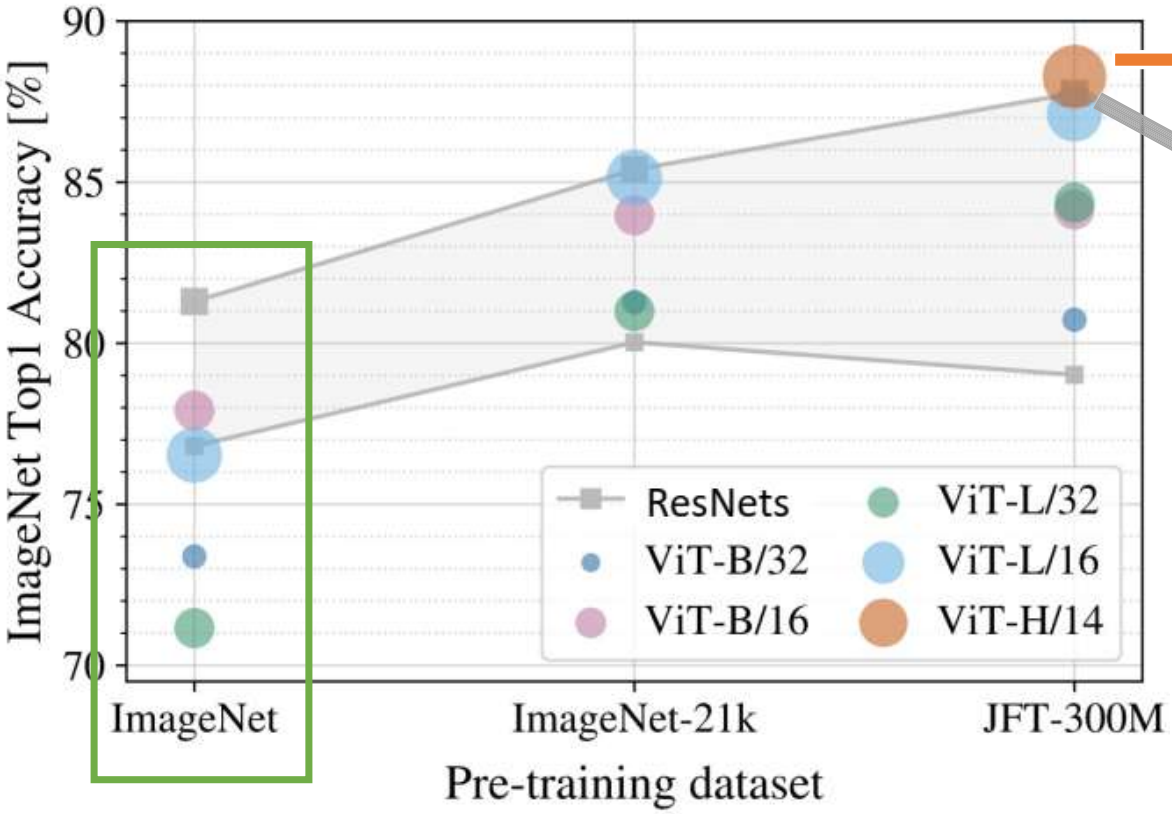
ViT: 2.5k TPU-v3 días de training

ResNets: 9.9k TPU-v3 días de training

ViT hace uso más eficiente de GPUs

ViT vs. ResNets

Cómo podemos mejorar
el performance con ImageNet?



ViT: 2.5k TPU-v3
días de training

ResNets: 9.9k TPU-v3
días de training

ViT hace uso más
eficiente de GPUs

Mejorando ViT: Aumentación y Regularización

Regularización para ViT:

- Weight decay
- Dropout (en capas MLP del Transformer)

Data augmentation para ViT:

- MixUp
- RandAugment

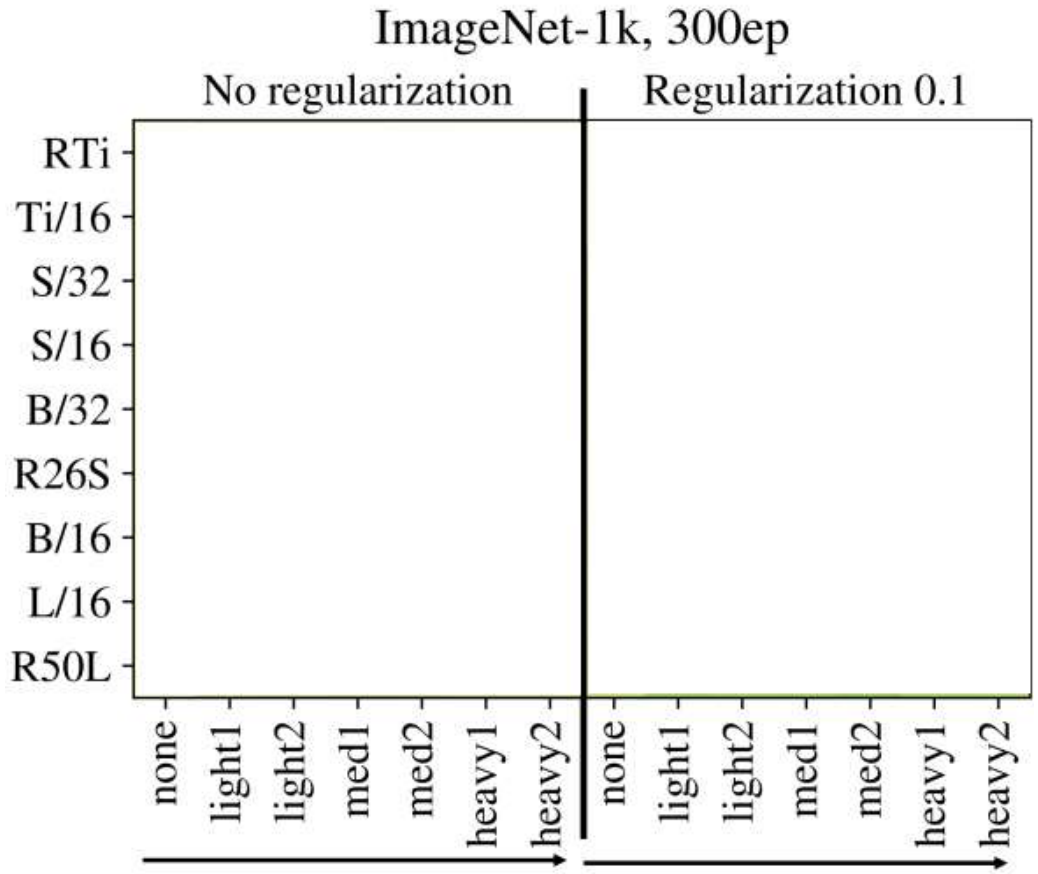
Mejorando ViT: Aumentación y Regularización

Regularización para ViT:

- Weight decay
- Dropout (en capas MLP del Transformer)

Data augmentation para ViT:

- MixUp
- RandAugment



Mejorando ViT: Aumentación y Regularización

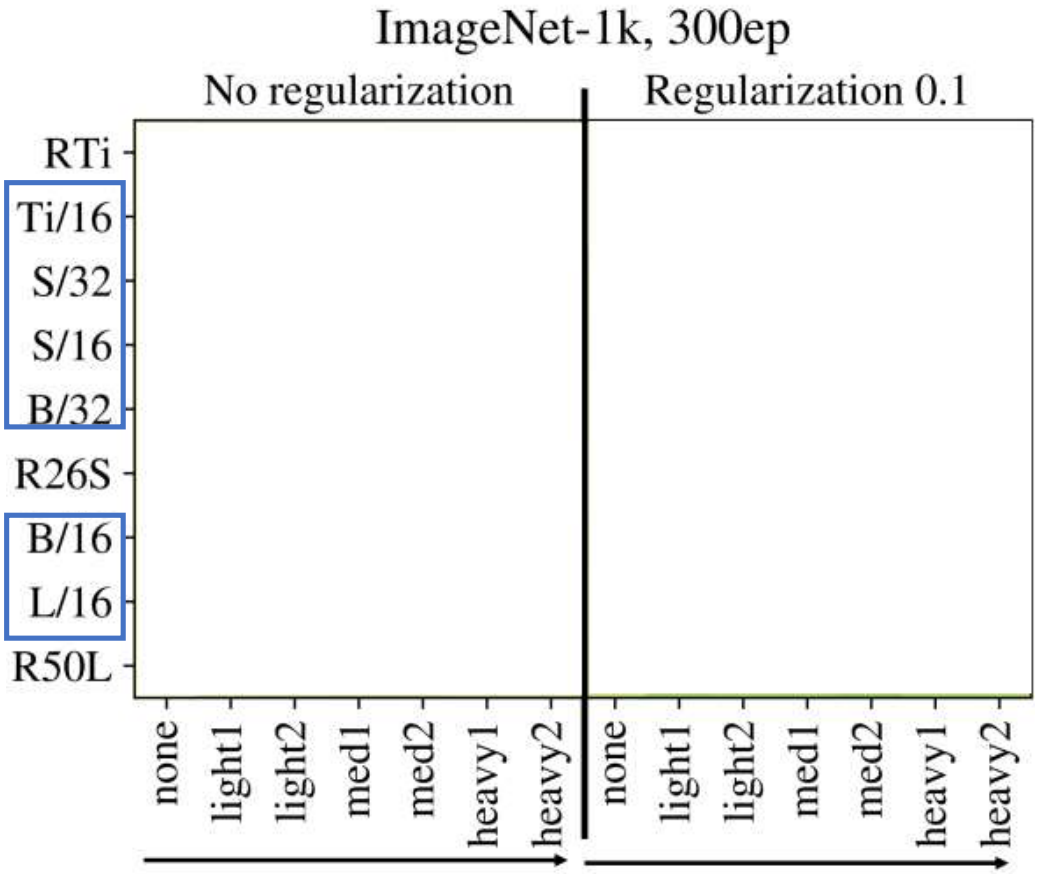
Regularización para ViT:

- Weight decay
- Dropout (en capas MLP del Transformer)

Data augmentation para ViT:

- MixUp
- RandAugment

ViT:
Ti: tiny
S: Small
B: Base
L: Large



Mejorando ViT: Aumentación y Regularización

Regularización para ViT:

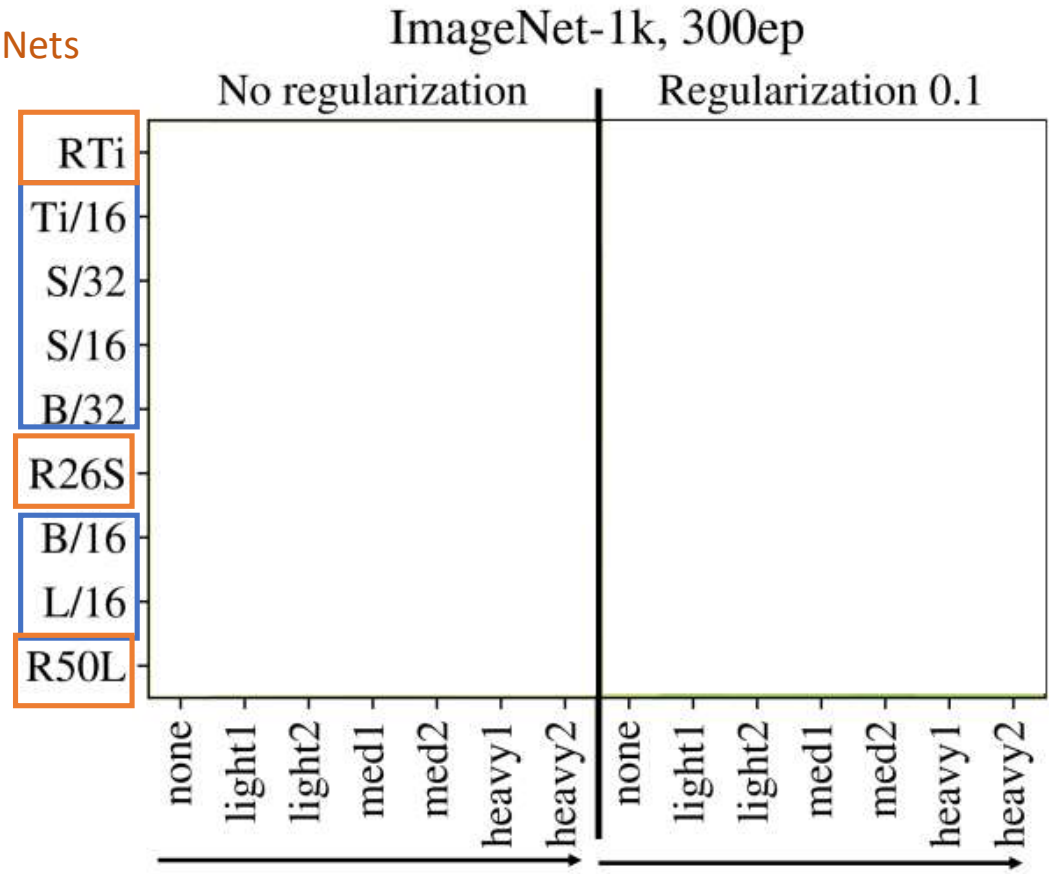
- Weight decay
- Dropout (en capas MLP del Transformer)

Data augmentation para ViT:

- MixUp
- RandAugment

Híbridos: bloques ResNets
Con bloques ViT

ViT:
Ti: tiny
S: Small
B: Base
L: Large



Mejorando ViT: Aumentación y Regularización

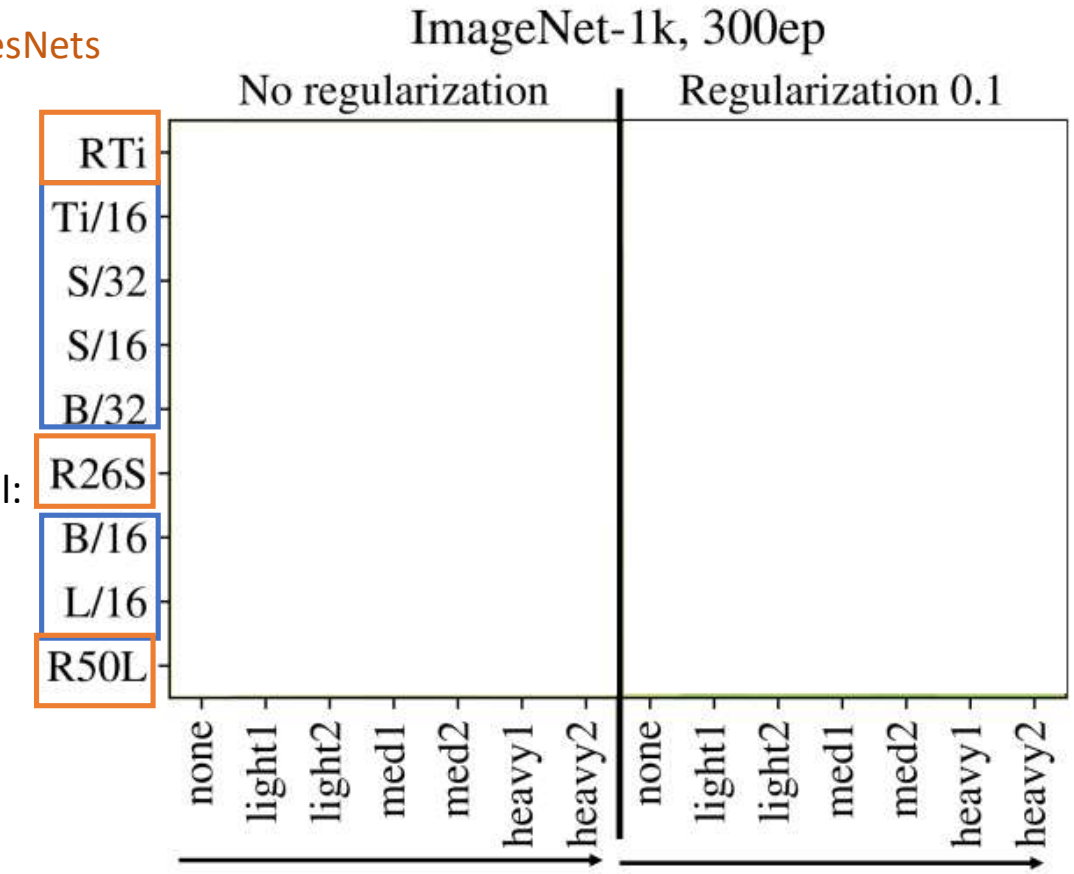
Regularización para ViT:
- Weight decay
- Dropout (en capas MLP del Transformer)

Data augmentation para ViT:
- MixUp
- RandAugment

Híbridos: bloques ResNets
Con bloques ViT

ViT:
Ti: tiny
S: Small
B: Base
L: Large

Original:
77.9
76.53



Mejorando ViT: Aumentación y Regularización

Regularización para ViT:

- Weight decay
- Dropout (en capas MLP del Transformer)

Data augmentation para ViT:

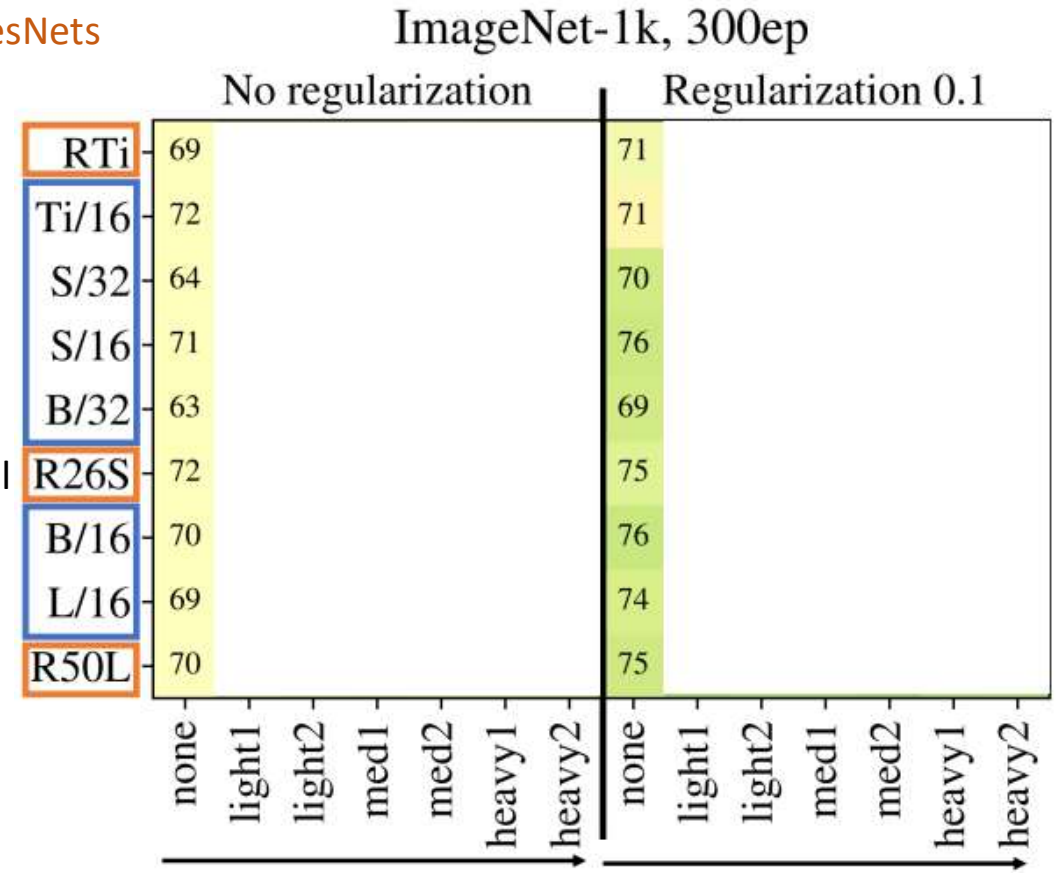
- MixUp
- RandAugment

Regularización siempre ayuda!

Híbridos: bloques ResNets
Con bloques ViT

ViT:
Ti: tiny
S: Small
B: Base
L: Large

Original
77.9
76.53



Mejorando ViT: Aumentación y Regularización

Regularización para ViT:

- Weight decay
- Dropout (en capas MLP del Transformer)

Data augmentation para ViT:

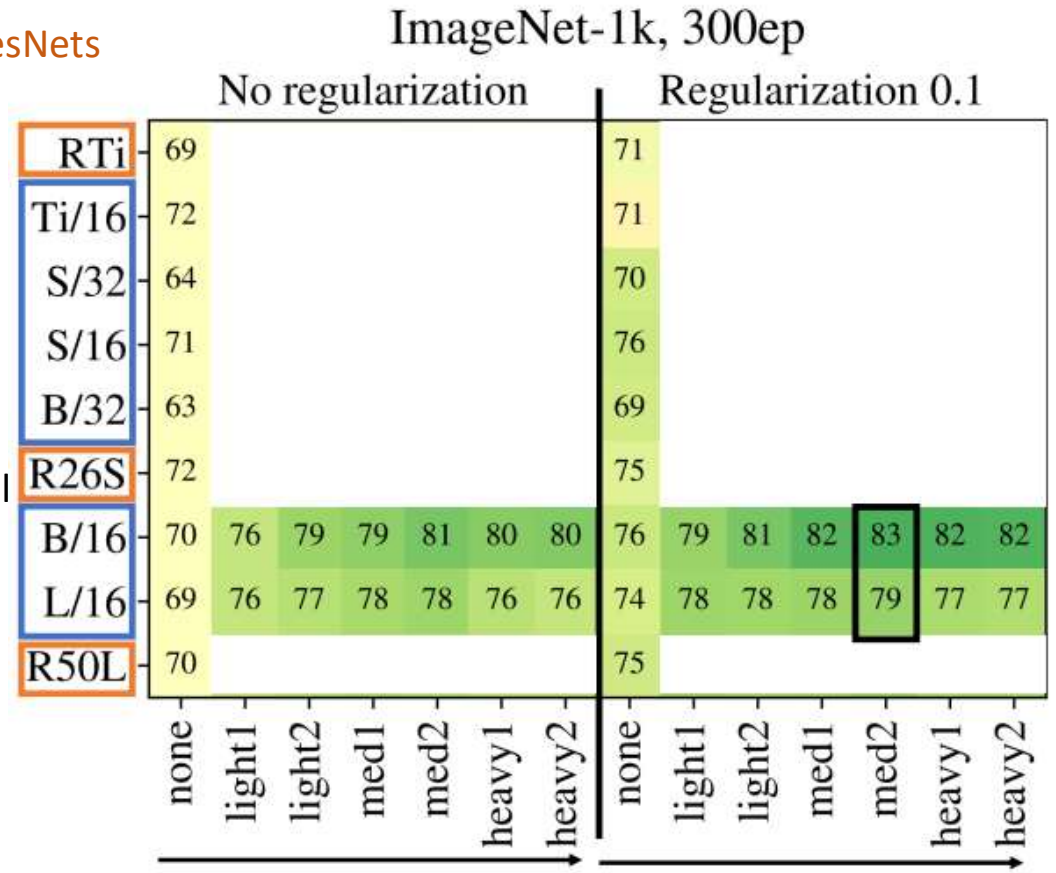
- MixUp
- RandAugment

Regularización + aumentación ayuda más!

Híbridos: bloques ResNets
Con bloques ViT

ViT:
Ti: tiny
S: Small
B: Base
L: Large

Original
77.9
76.53



Mejorando ViT: Aumentación y Regularización

Regularización para ViT:

- Weight decay
- Dropout (en capas MLP del Transformer)

Data augmentation para ViT:

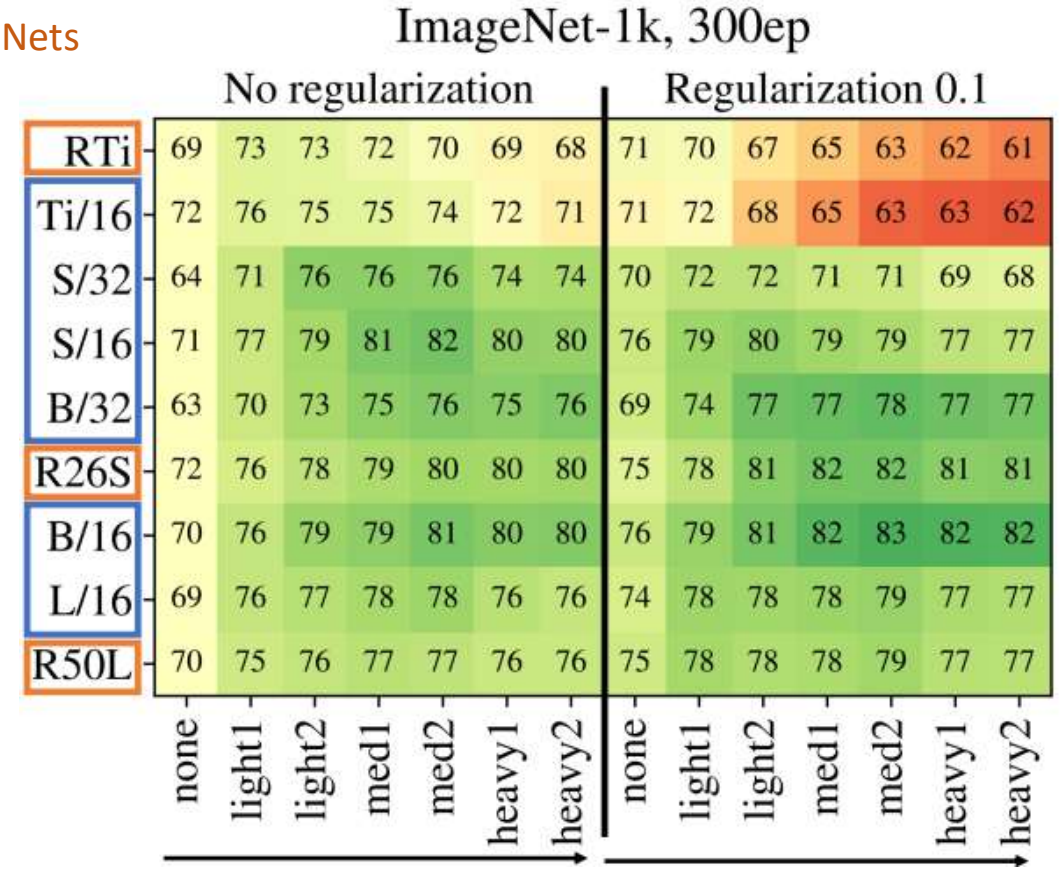
- MixUp
- RandAugment

Todos los resultados

Híbridos: bloques ResNets
Con bloques ViT

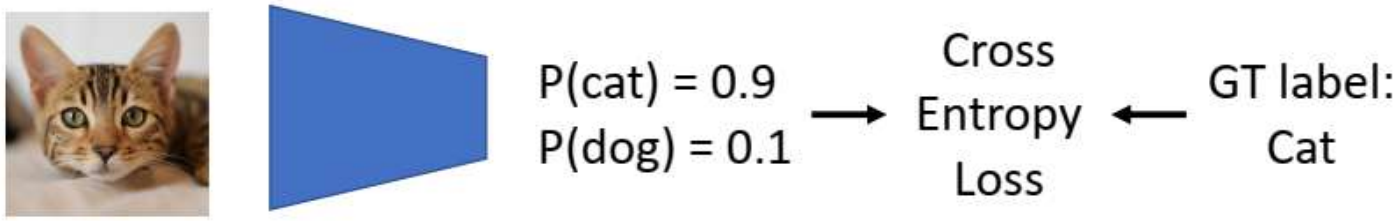
ViT:
Ti: tiny
S: Small
B: Base
L: Large

Original:
77.9
76.53



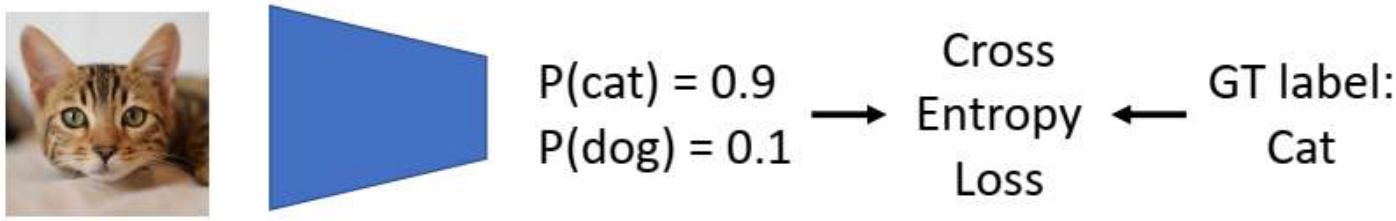
Mejorando ViT: Destilación

Paso 1: Entrenar un modelo teacher en imágenes con etiquetas

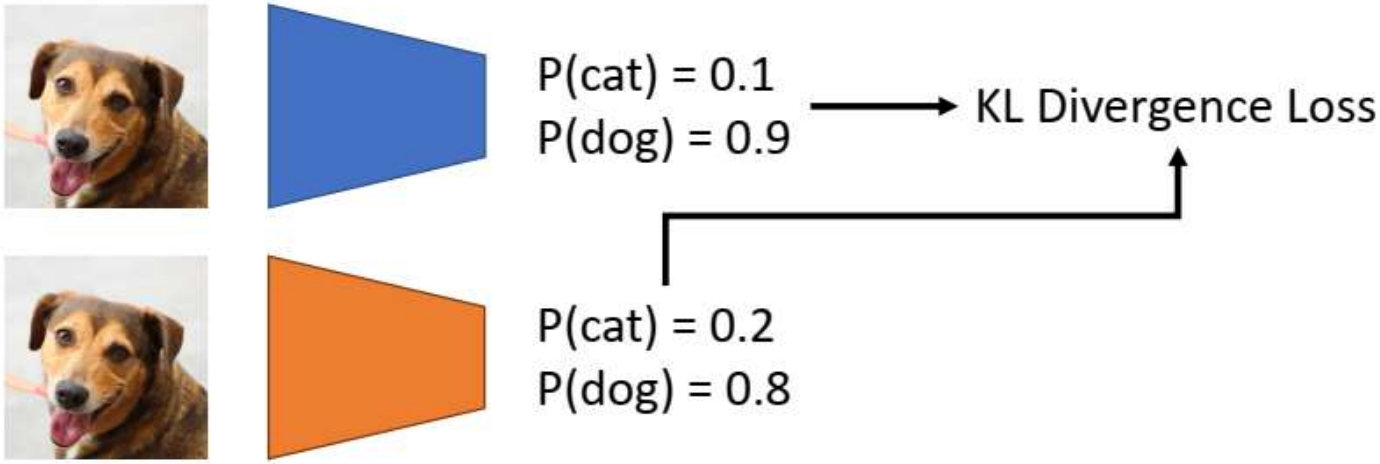


Mejorando ViT: Destilación

Paso 1: Entrenar un modelo teacher en imágenes con etiquetas

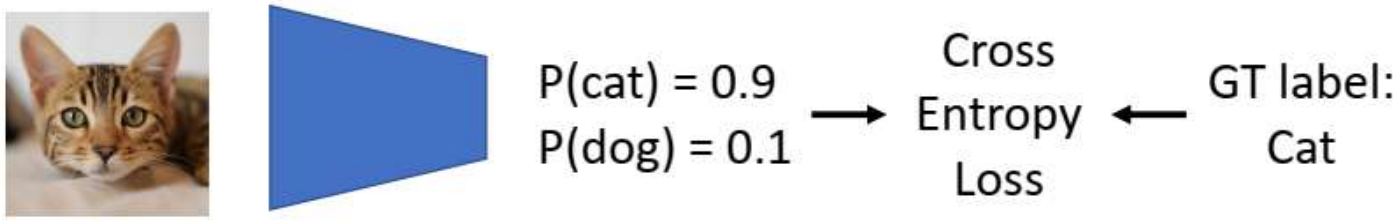


Paso 2: Entrenar un modelo estudiante para matchear predicciones del teacher

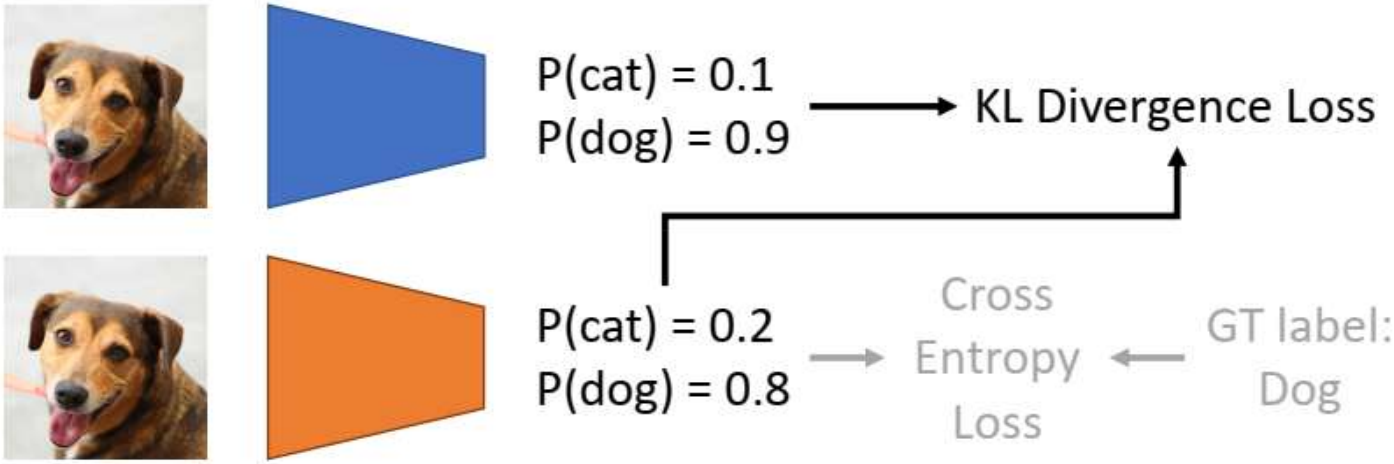


Mejorando ViT: Destilación

Paso 1: Entrenar un modelo teacher en imágenes con etiquetas

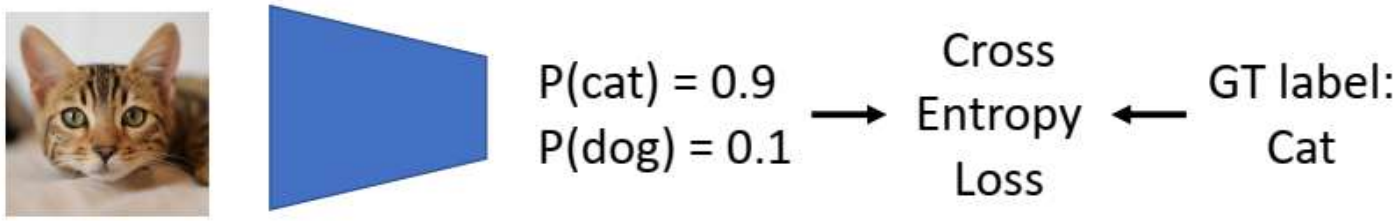


Paso 2: Entrenar un modelo estudiante para matchear predicciones del teacher (algunas veces también las etiquetas)

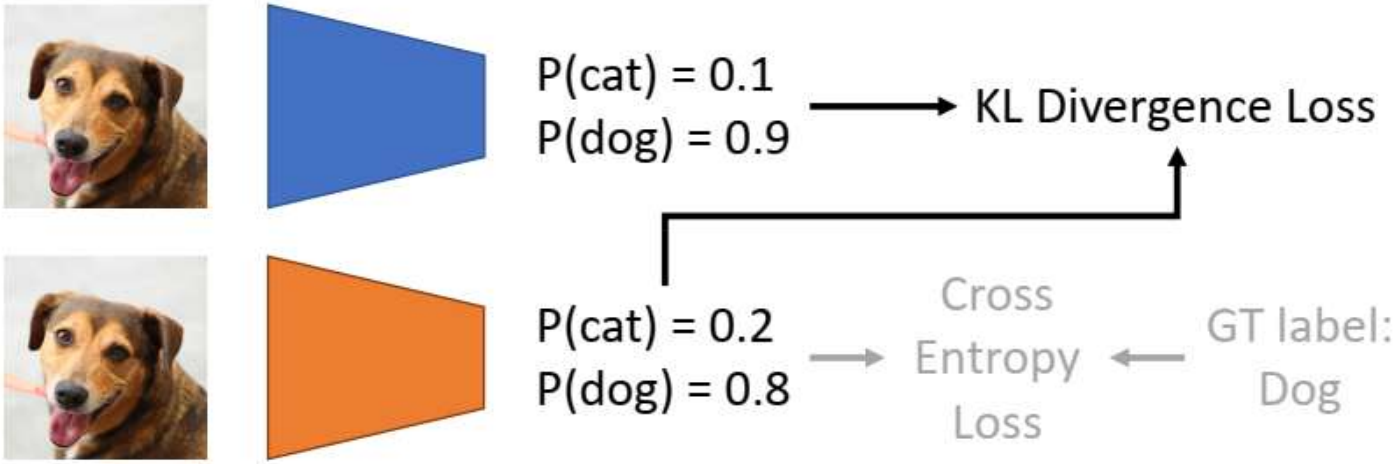


Mejorando ViT: Destilación

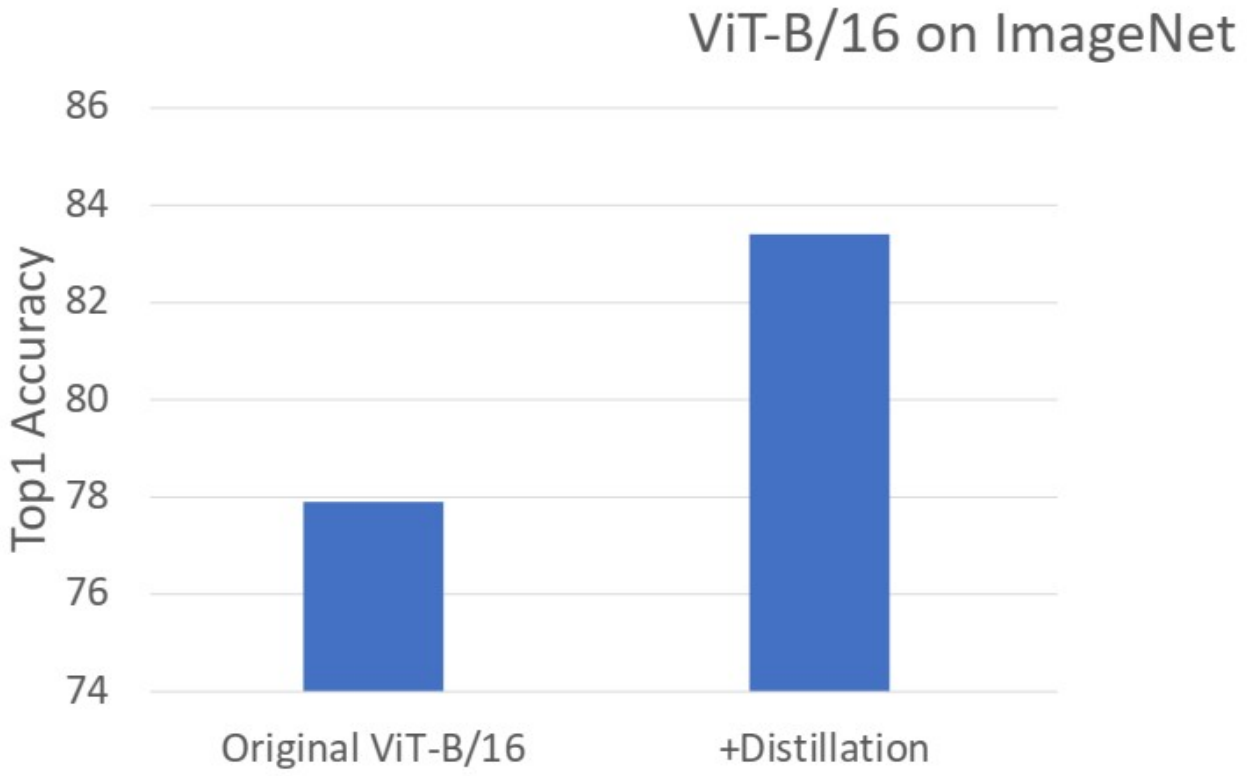
Paso 1: Entrenar un modelo Teacher CNN en imágenes con etiquetas



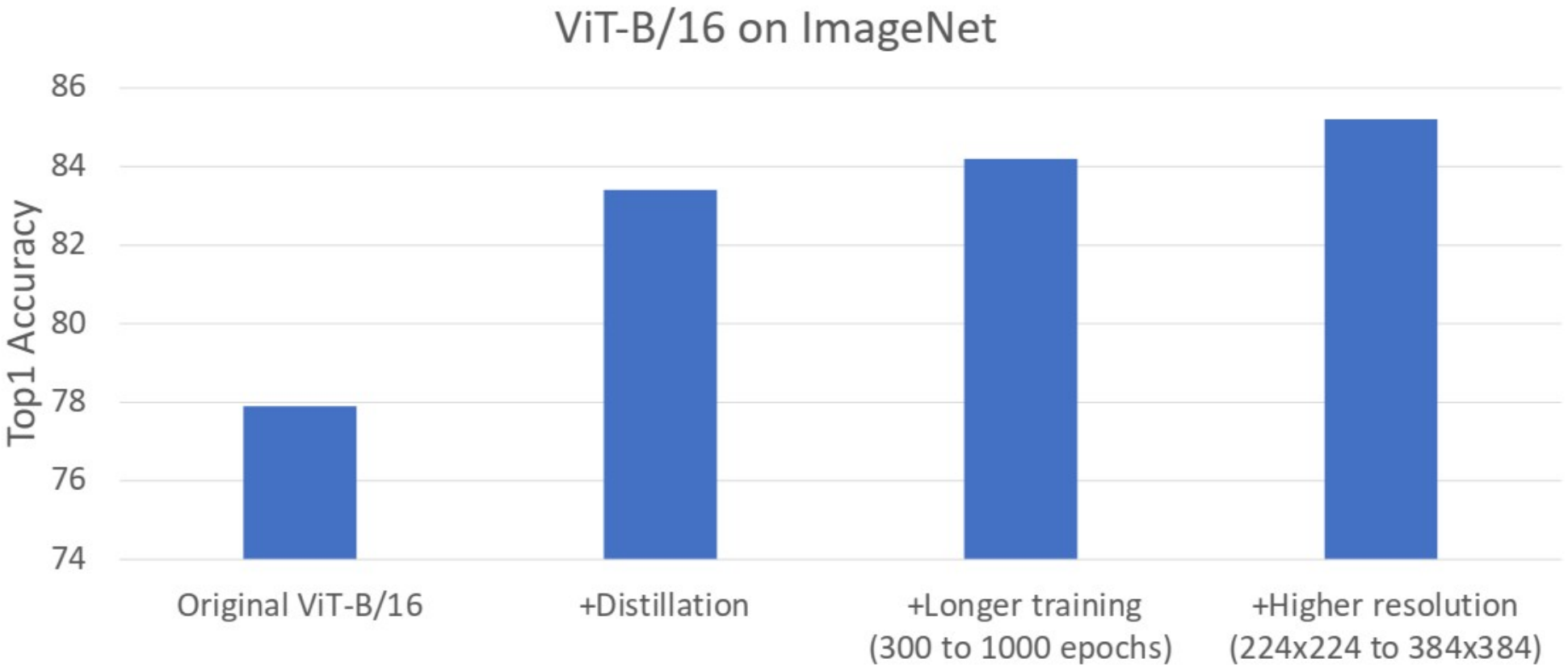
Paso 2: Entrenar un modelo Estudiante ViT para matchear predicciones del teacher CNN (algunas veces también las etiquetas)



Mejorando ViT: Destilación



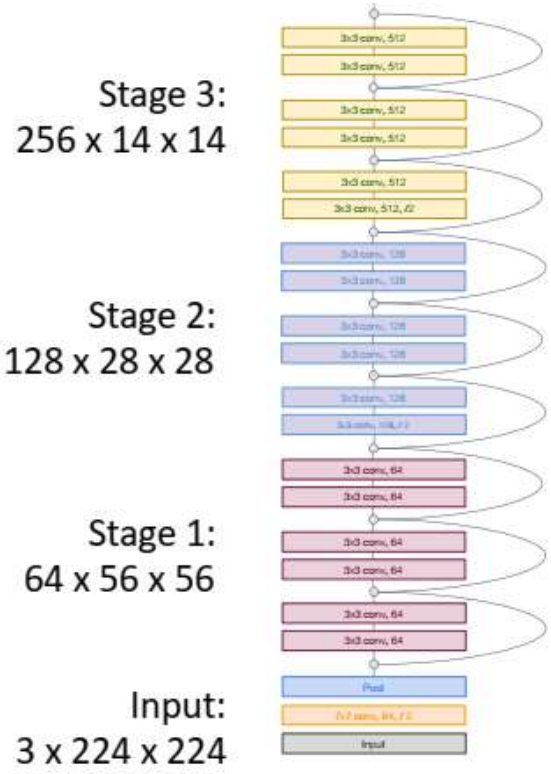
Mejorando ViT: Destilación



ViT vs. CNN

CNN se va reduciendo la resolución y ampliando la profundidad

Útil para ser robustos a La escala



ViT vs. CNN

CNN se va reduciendo la resolución y ampliando la profundidad

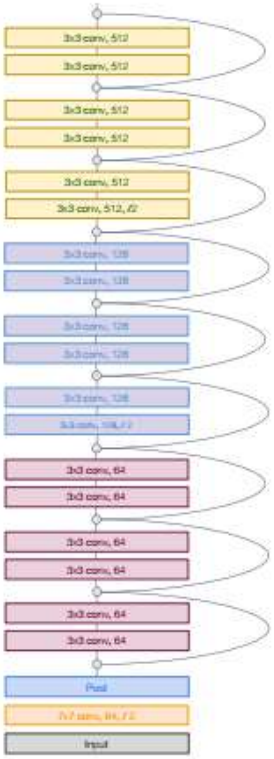
Útil para ser robustos a La escala

Stage 3:
256 x 14 x 14

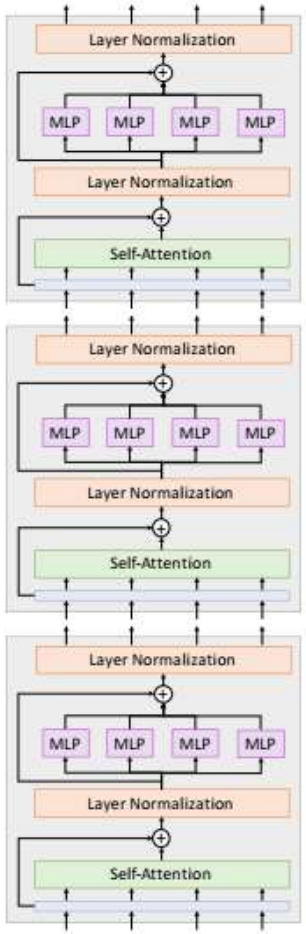
Stage 2:
128 x 28 x 28

Stage 1:
64 x 56 x 56

Input:
3 x 224 x 224



En ViT, todos los bloques tienen la misma resolución



3rd block:
768 x 14 x 14

2nd block:
768 x 14 x 14

1st block:
768 x 14 x 14

Input:
3 x 224 x 224

ViT vs. CNN

CNN se va reduciendo la resolución y ampliando la profundidad

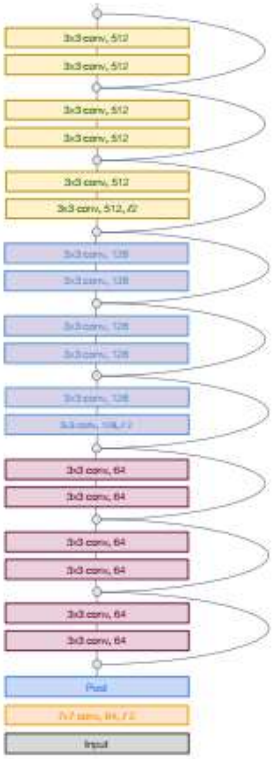
Útil para ser robustos a La escala

Stage 3:
256 x 14 x 14

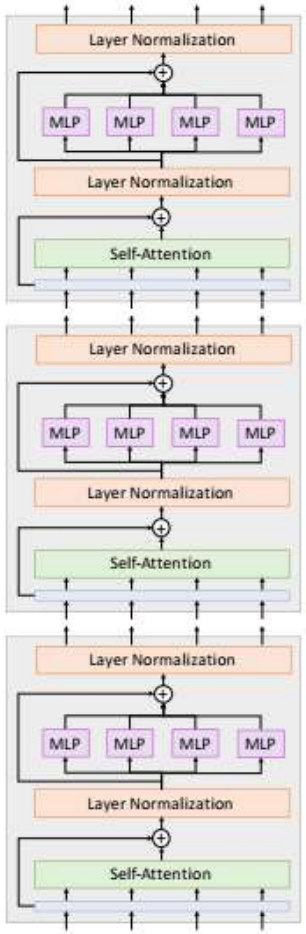
Stage 2:
128 x 28 x 28

Stage 1:
64 x 56 x 56

Input:
3 x 224 x 224



En ViT, todos los bloques tienen la misma resolución



3rd block:
768 x 14 x 14

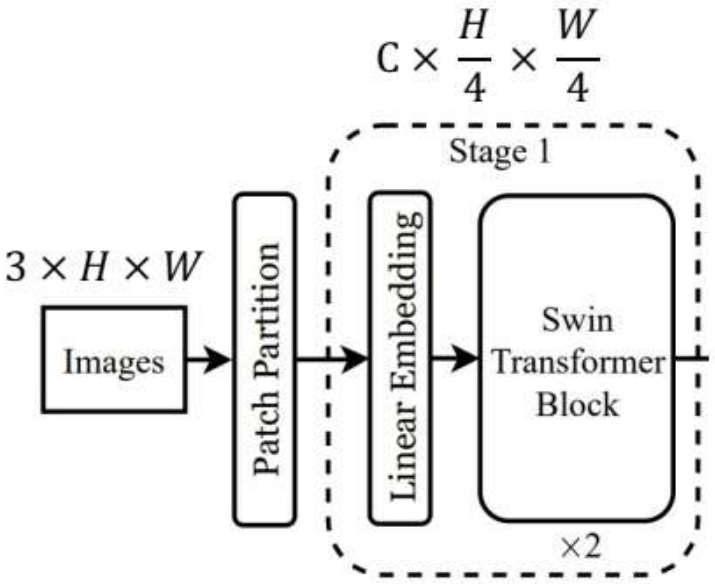
2nd block:
768 x 14 x 14

1st block:
768 x 14 x 14

Input:
3 x 224 x 224

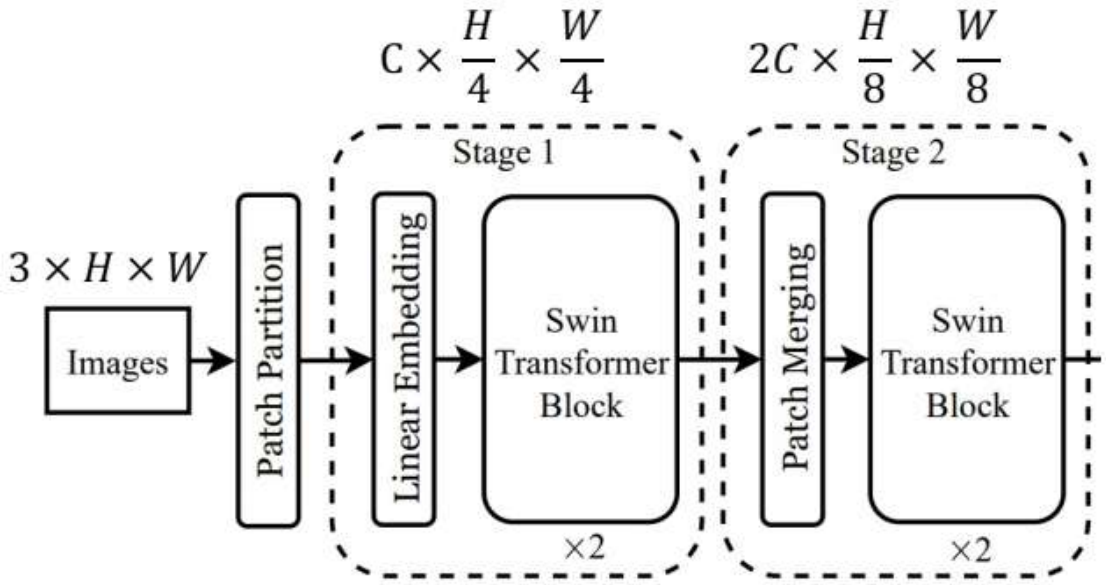
Se puede construir un **ViT jerárquico**?

ViT Jerárquico: Swin Transformer



Dividir imagen en 4x4 Parches y proyectar a C dimensiones

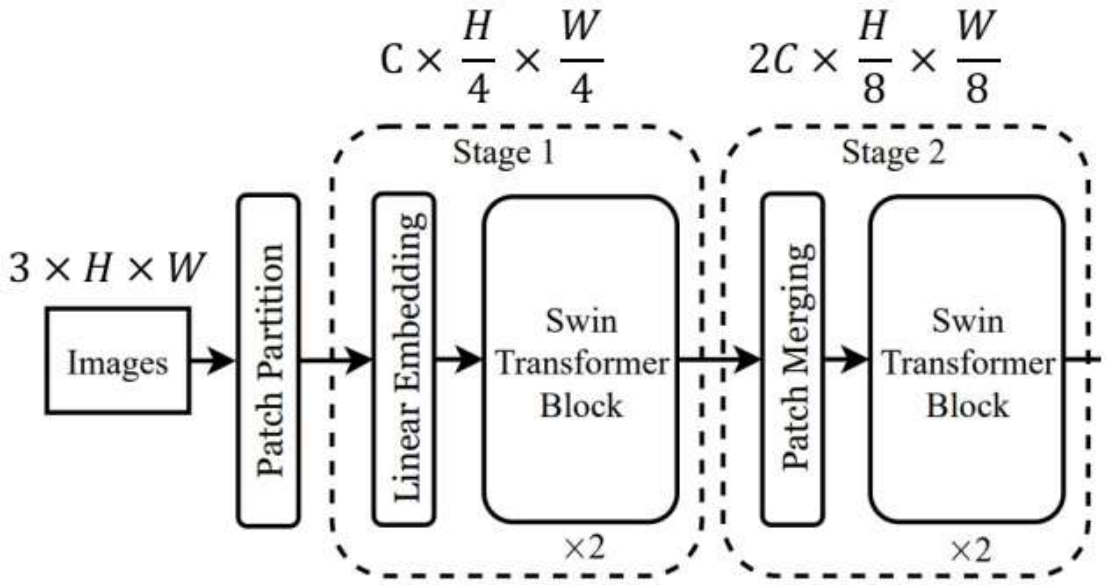
ViT Jerárquico: Swin Transformer

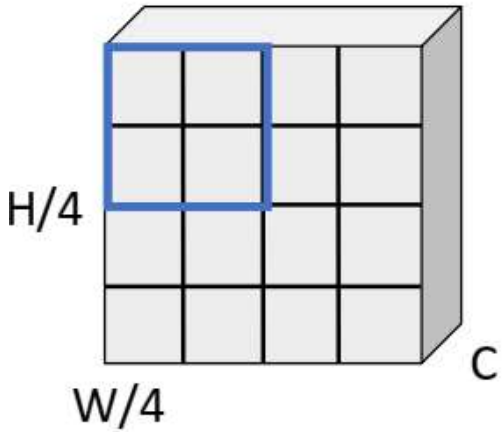


Dividir imagen en 4x4 Parches y proyectar a C dimensiones

Unir 2x2 vecinos: parches de 8x8

ViT Jerárquico: Swin Transformer

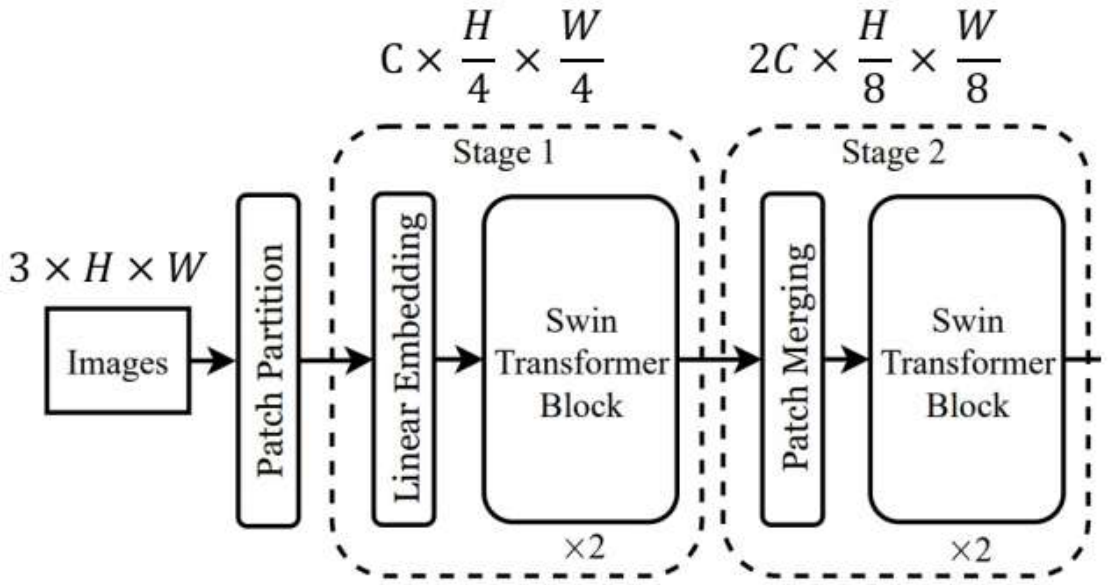


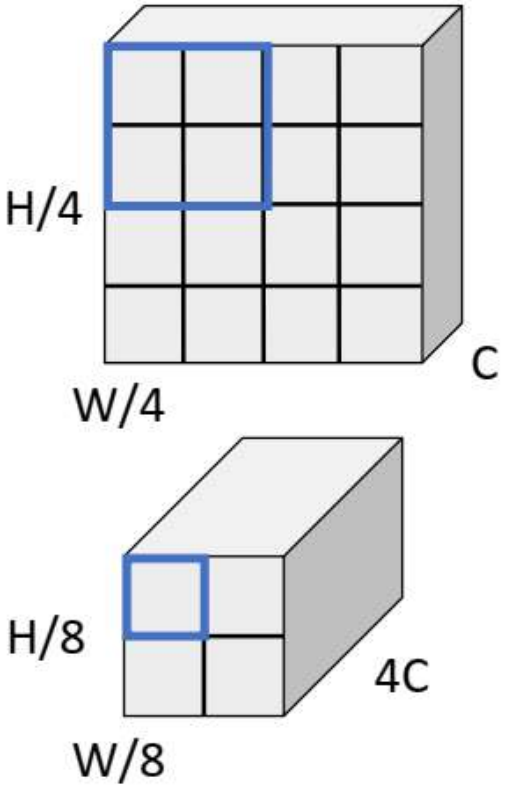


Dividir imagen en 4x4 Parches y proyectar a C dimensiones

Unir 2x2 vecinos: parches de 8x8

ViT Jerárquico: Swin Transformer

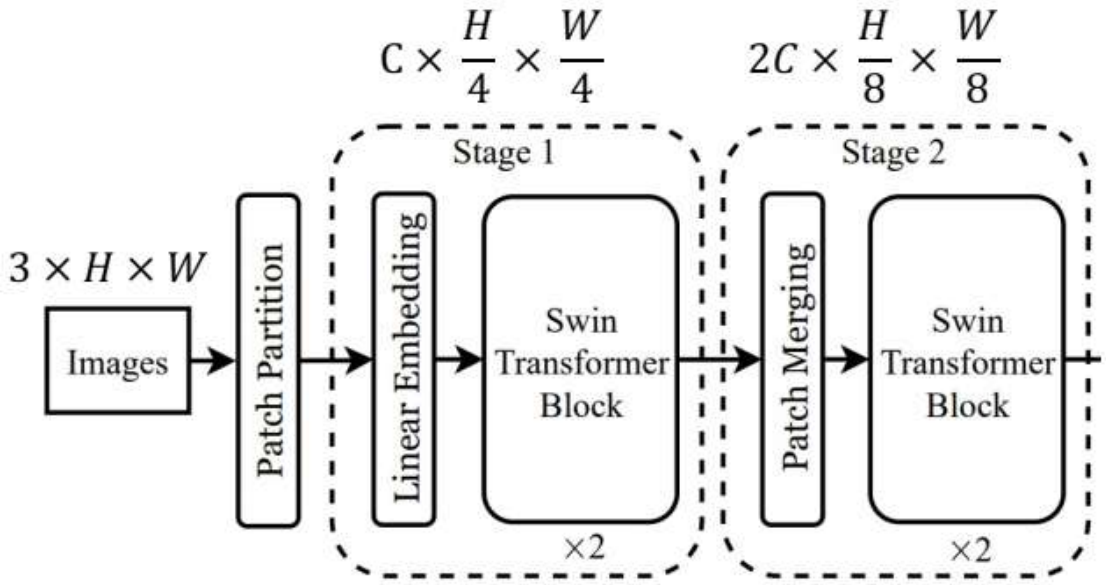




Dividir imagen en 4x4 Parches y proyectar a C dimensiones

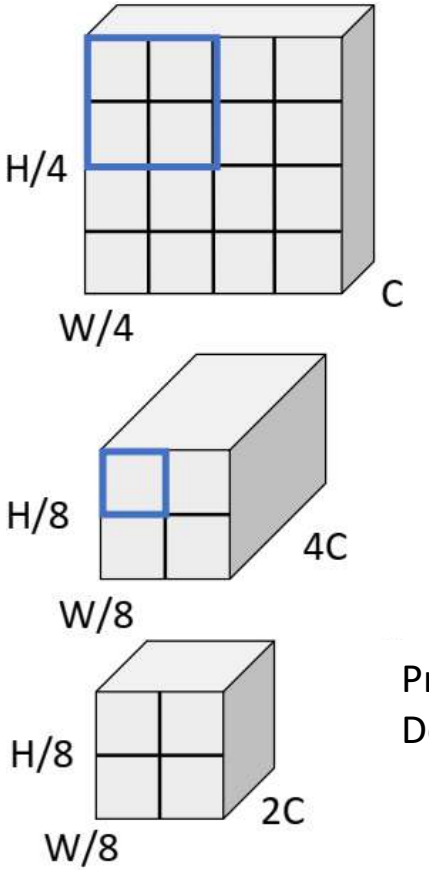
Unir 2x2 vecinos: parches de 8x8

ViT Jerárquico: Swin Transformer



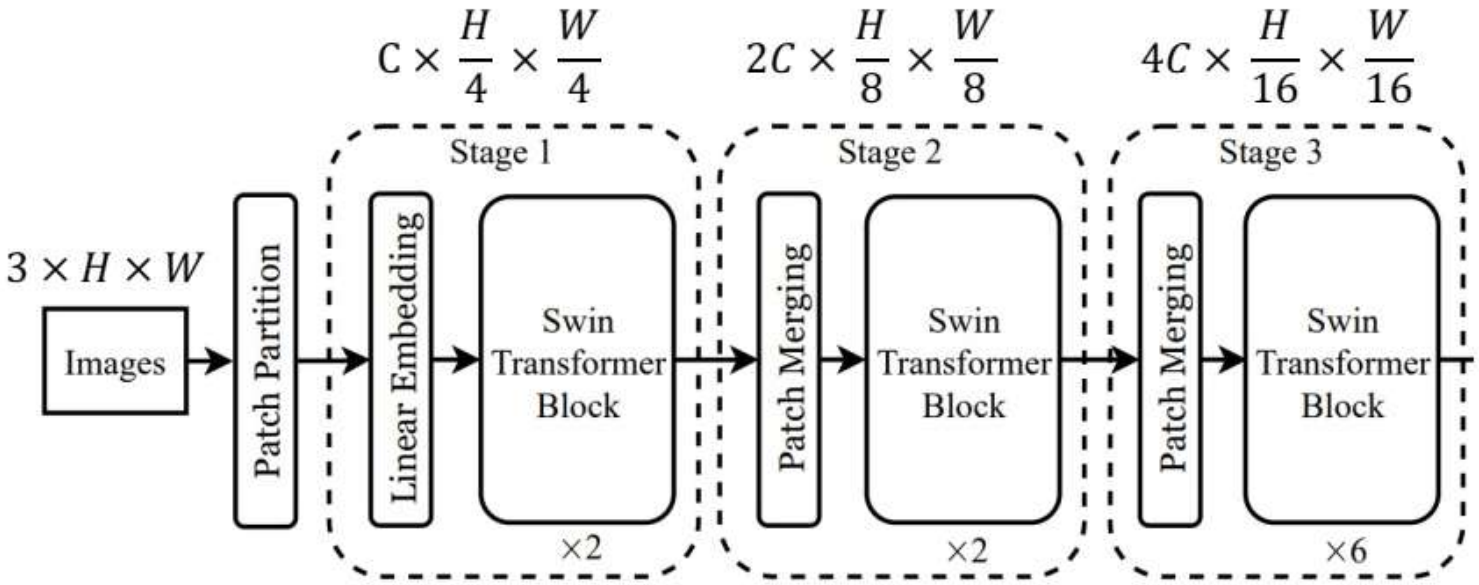
Dividir imagen en 4x4
Parches y proyectar a C
dimensiones

Unir 2x2 vecinos:
parches de 8x8



Proyección lineal
De 4C a 2C canales

ViT Jerárquico: Swin Transformer

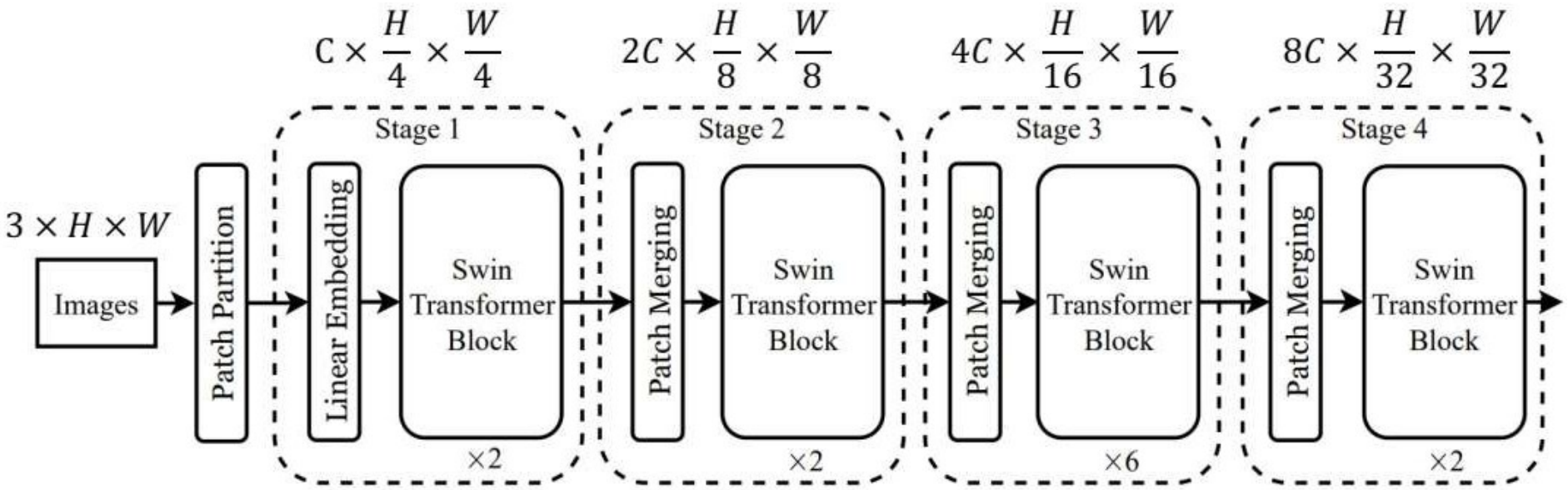


Dividir imagen en 4x4 Parches y proyectar a C dimensiones

Unir 2x2 vecinos: parches de 8x8

Unir 2x2 vecinos: parches de 16x16

ViT Jerárquico: Swin Transformer



Dividir imagen en 4x4 Parches y proyectar a C dimensiones

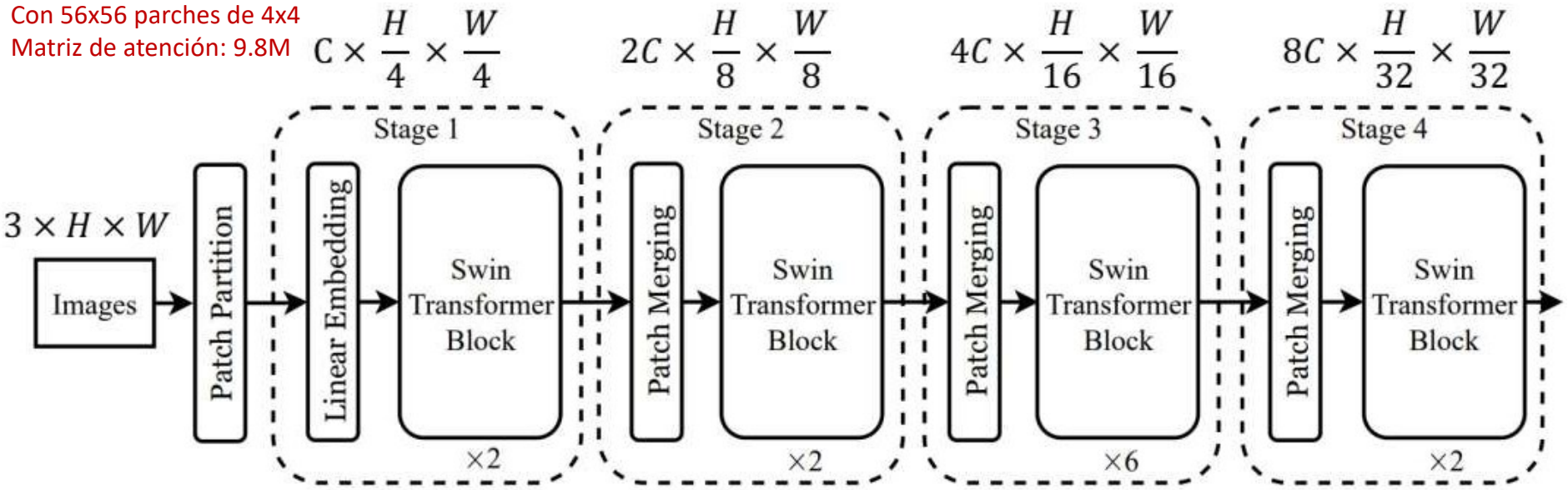
Unir 2x2 vecinos: parches de 8x8

Unir 2x2 vecinos: parches de 16x16

Unir 2x2 vecinos: parches de 32x32

ViT Jerárquico: Swin Transformer

Problema:224x224 imagen
Con 56x56 parches de 4x4
Matriz de atención: 9.8M



Dividir imagen en 4x4 Parches y proyectar a C dimensiones

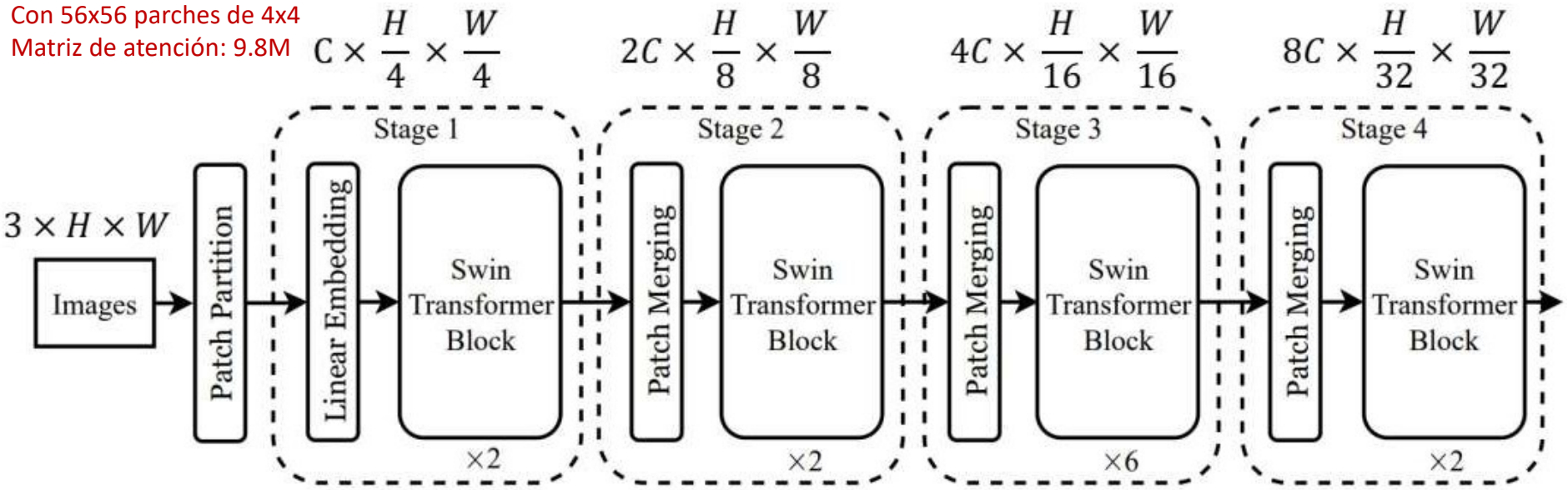
Unir 2x2 vecinos: parches de 8x8

Unir 2x2 vecinos: parches de 16x16

Unir 2x2 vecinos: parches de 32x32

ViT Jerárquico: Swin Transformer

Problema:224x224 imagen
Con 56x56 parches de 4x4
Matriz de atención: 9.8M



Solución: no usar full attention
Atención sobre parches

Dividir imagen en 4x4 Parches y proyectar a C dimensiones

Unir 2x2 vecinos: parches de 8x8

Unir 2x2 vecinos: parches de 16x16

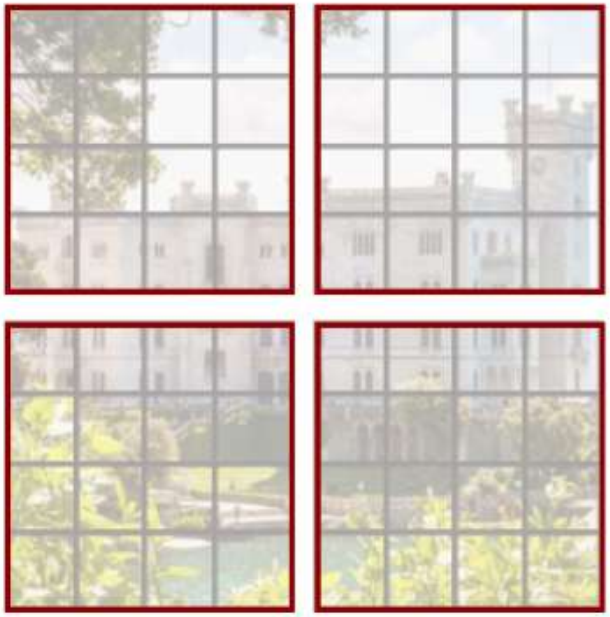
Unir 2x2 vecinos: parches de 32x32

Swin Transformer: Window Attention

Con una grilla de H x W de tokens, cada matriz de atención necesita H^2W^2 - **cuadrático**

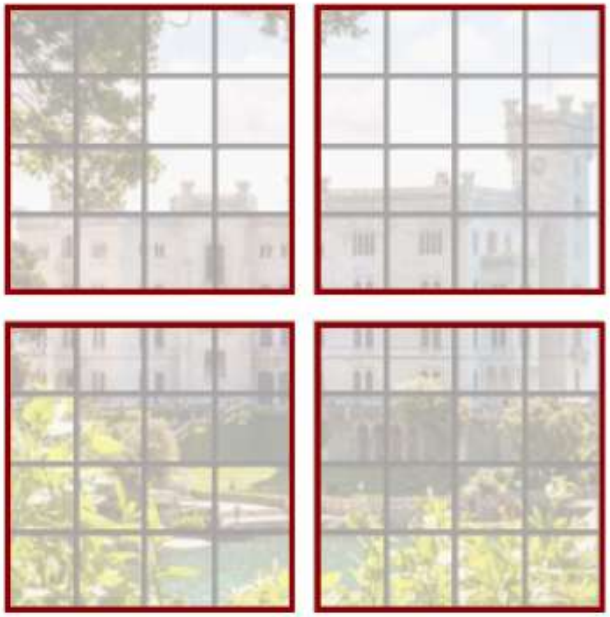
Swin Transformer: Window Attention

Con una grilla de H x W de tokens, cada matriz de atención necesita H^2W^2 - **cuadrático**



En vez que un token atienda a todos los otros tokens,
Dividir en ventanas de M x M tokens (aquí M = 4);
Solo computar atención dentro de las ventanas

Swin Transformer: Window Attention



Con una grilla de H x W de tokens, cada matriz de atención necesita H^2W^2 - **cuadrático**

En vez que un token atienda a todos los otros tokens,
Dividir en ventanas de M x M tokens (aquí M = 4);
Solo computar atención dentro de las ventanas

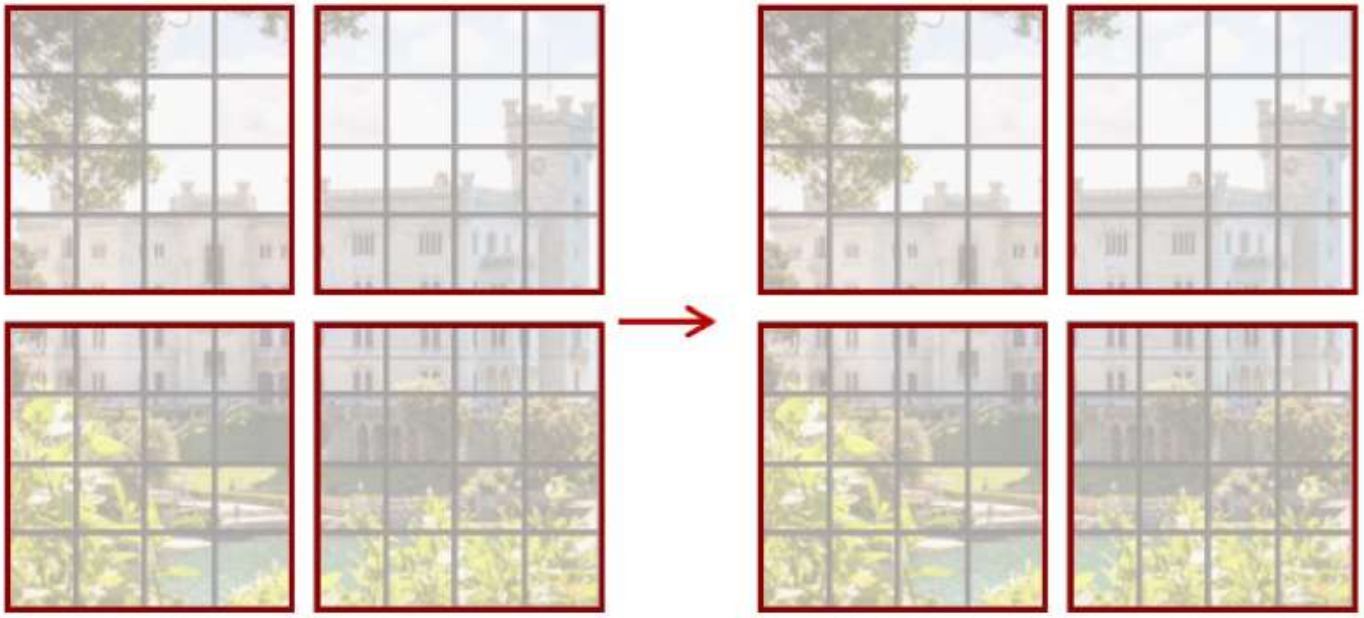
Tamaño total de matriz de atención:

$$M^4 \left(\frac{H}{M}\right)\left(\frac{W}{M}\right) = M^2HW$$

Lineal cuando M es fijo. Swin usa M=7

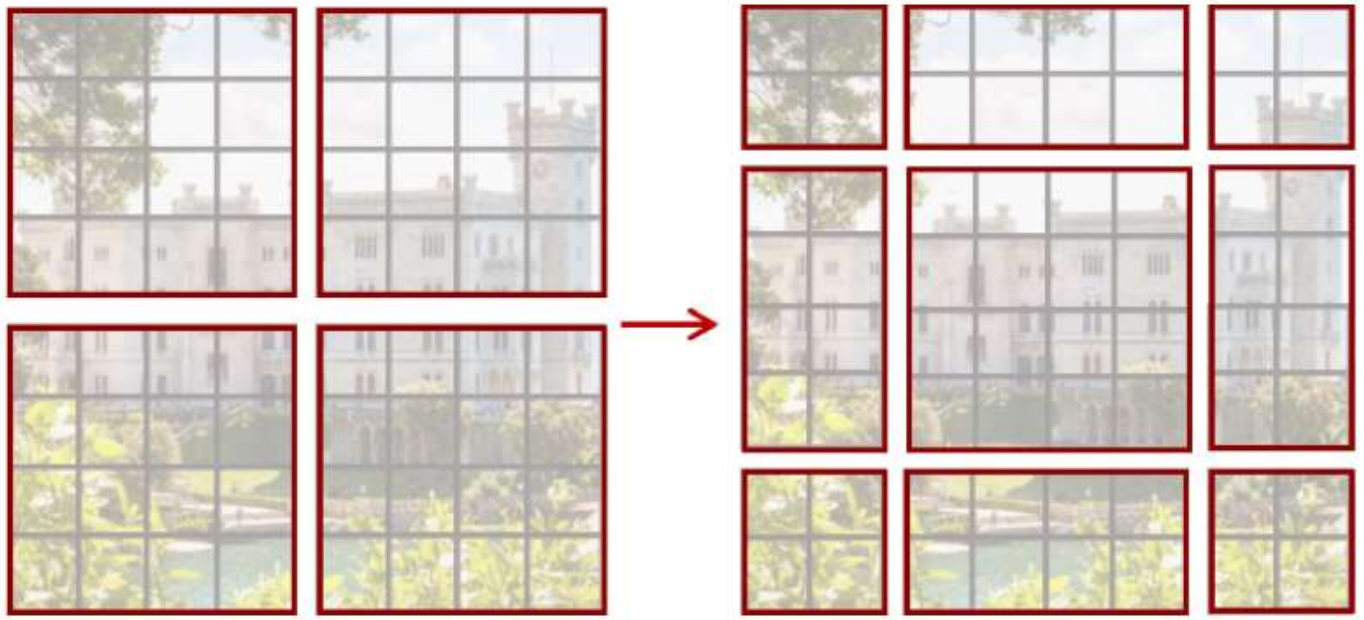
Swin Transformer: Window Attention

Problema: tokens solo interactúan con otros tokens en la misma ventana, pero no hay comunicación entre ventanas



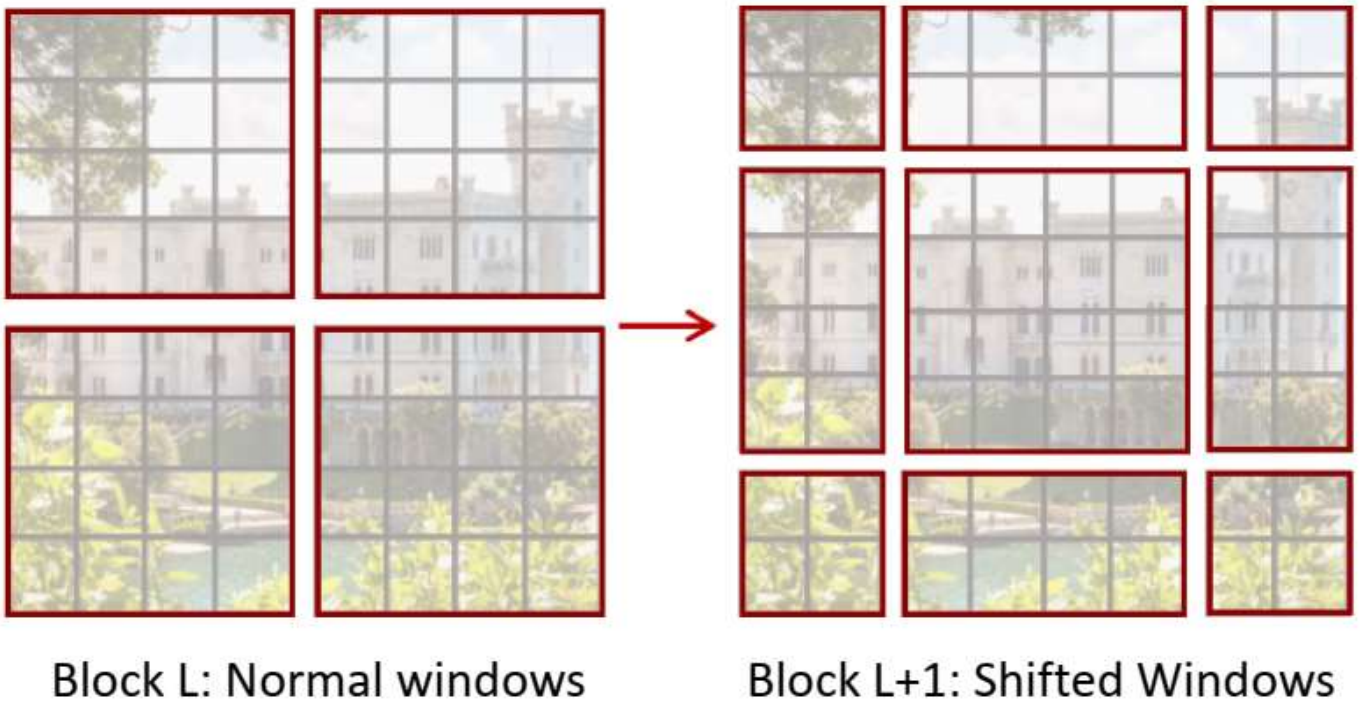
Swin Transformer: Window Attention

Solución: alternar entre ventanas normales y ventanas desplazadas en bloques sucesivos



Swin Transformer: Window Attention

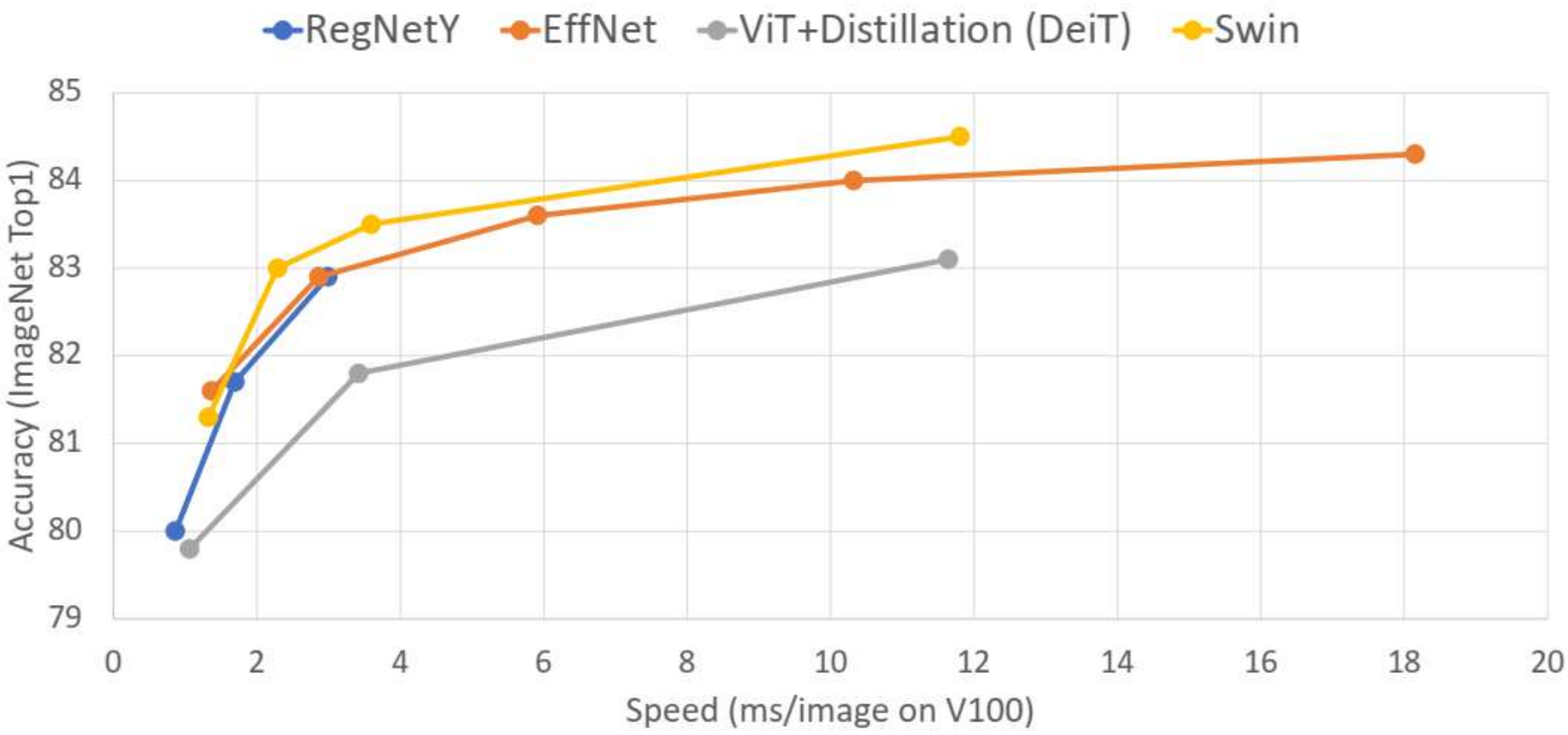
Solución: alternar entre ventanas normales y ventanas desplazadas en bloques sucesivos



ViT añade embeddings posicionales a los tokens del input

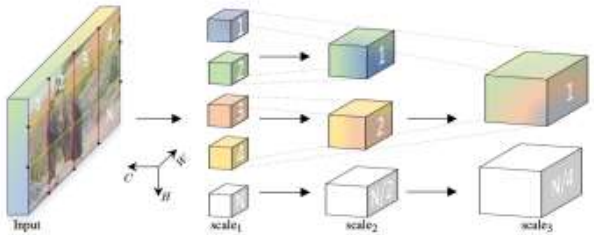
Swin usa relative positions entre parches

Swin Transformers: Speed vs Accuracy



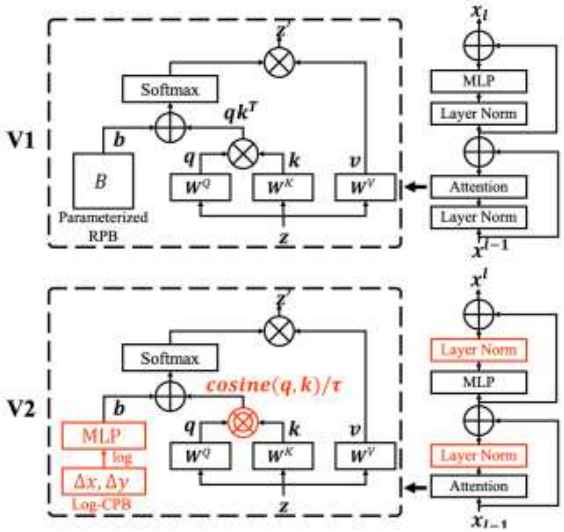
Otros modelos jerárquicos

MViT



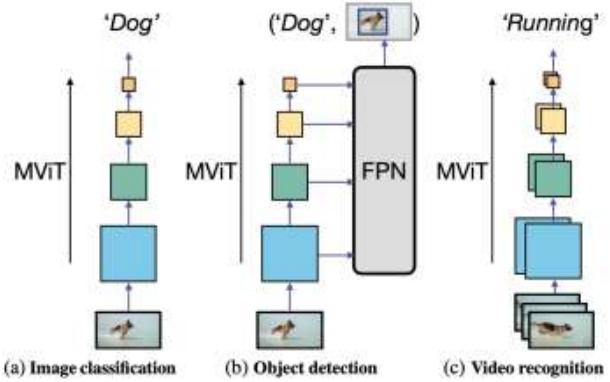
Fan et al, "Multiscale Vision Transformers", ICCV 2021

Swin-V2



Liu et al, "Swin Transformer V2: Scaling up Capacity and Resolution", CVPR 2022

Improved MViT

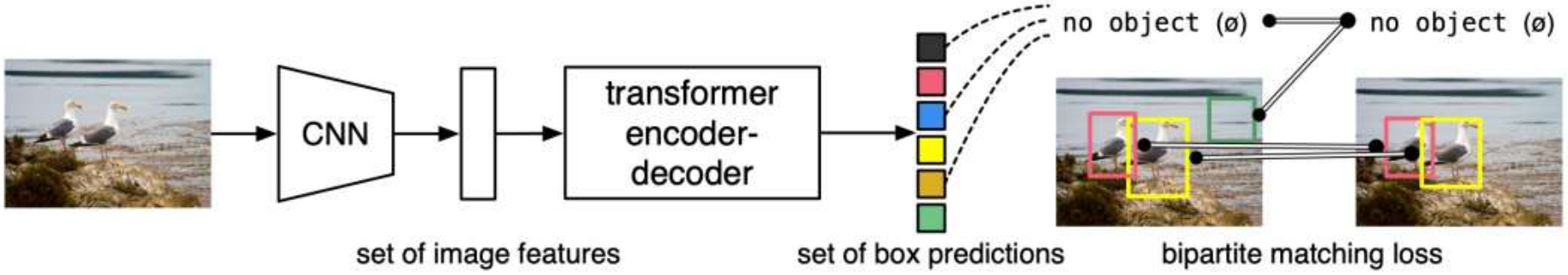


Li et al, "Improved Multiscale Vision Transformers for Classification and Detection", arXiv 2021

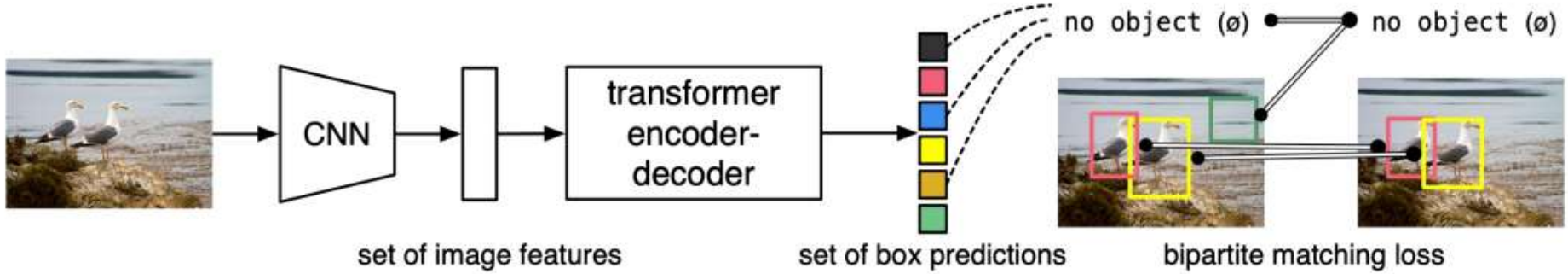
Detección de objetos con DETR

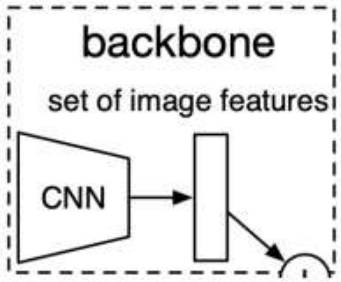
Pipeline simple de detección de objetos: salida conjunto de bounding boxes del Transformer

Matchear los boxes predichos a los boxes del groundtruth. Entrenar para hacer regresión de los boxes

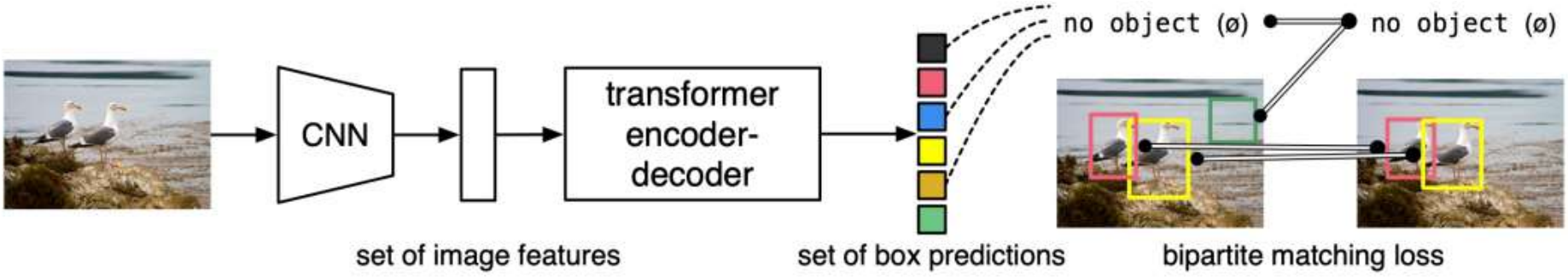


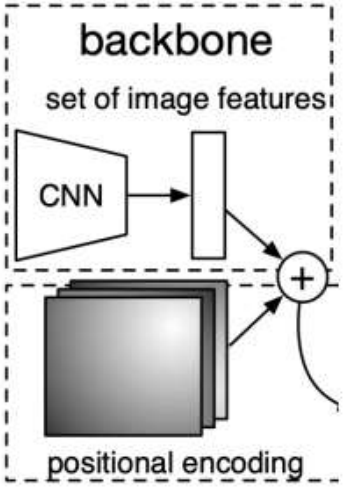
Detección de objetos con DETR



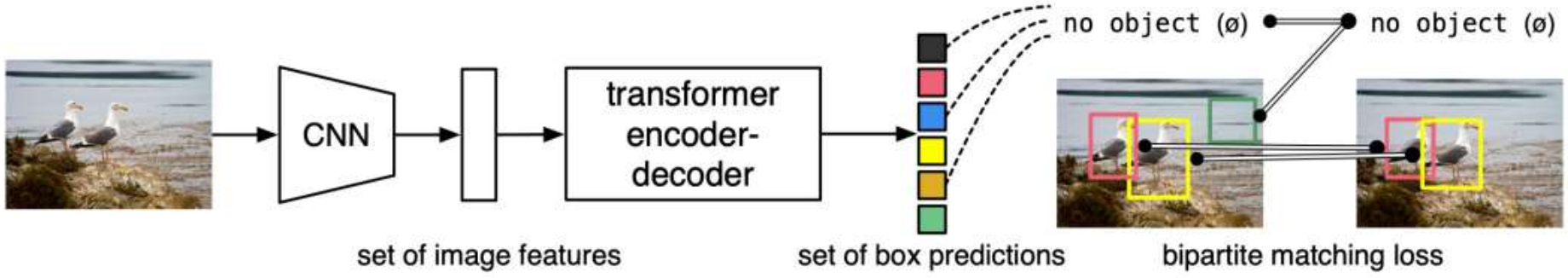


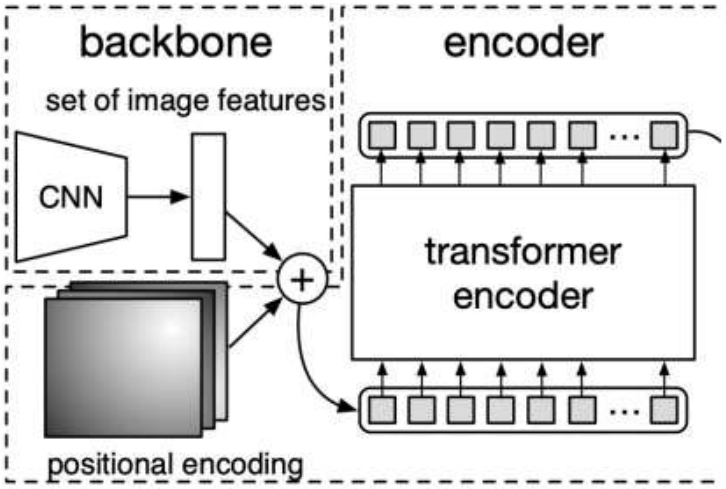
Detección de objetos con DETR



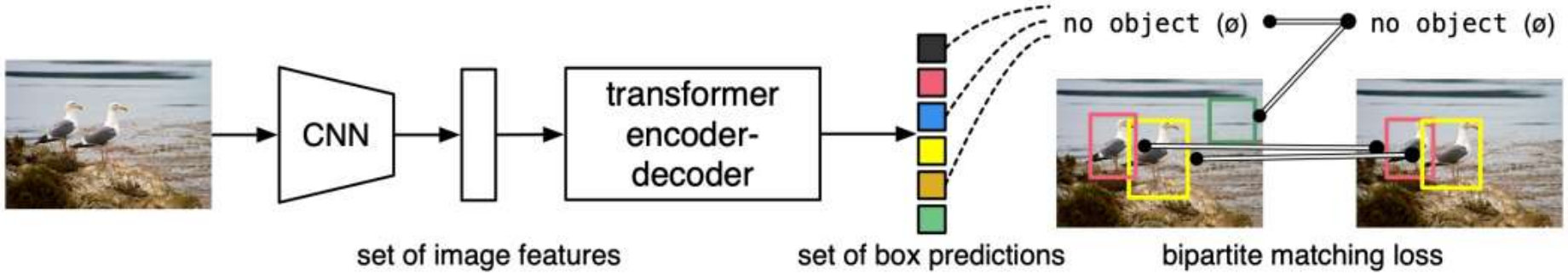


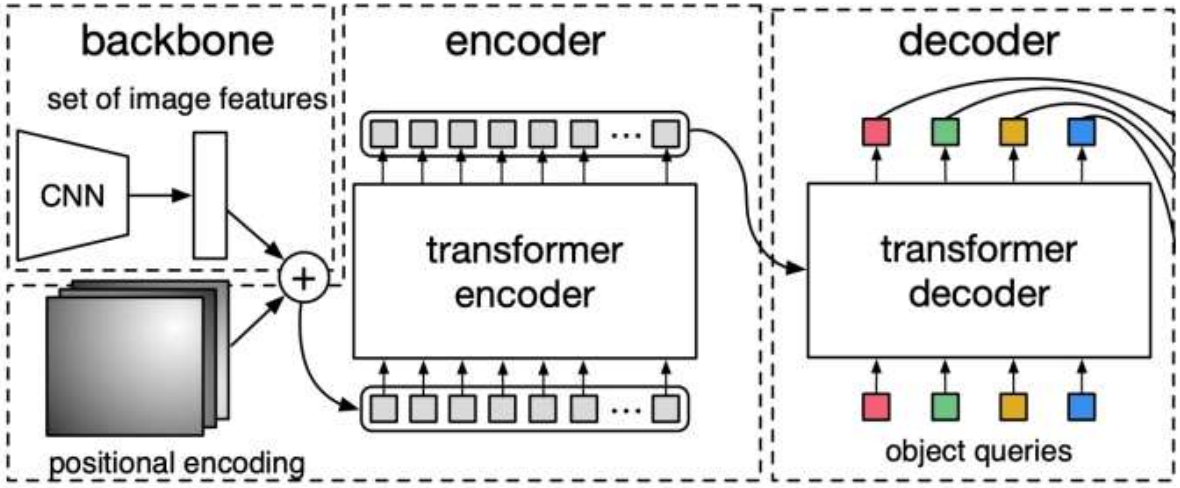
Detección de objetos con DETR





Detección de objetos con DETR





Detección de objetos con DETR