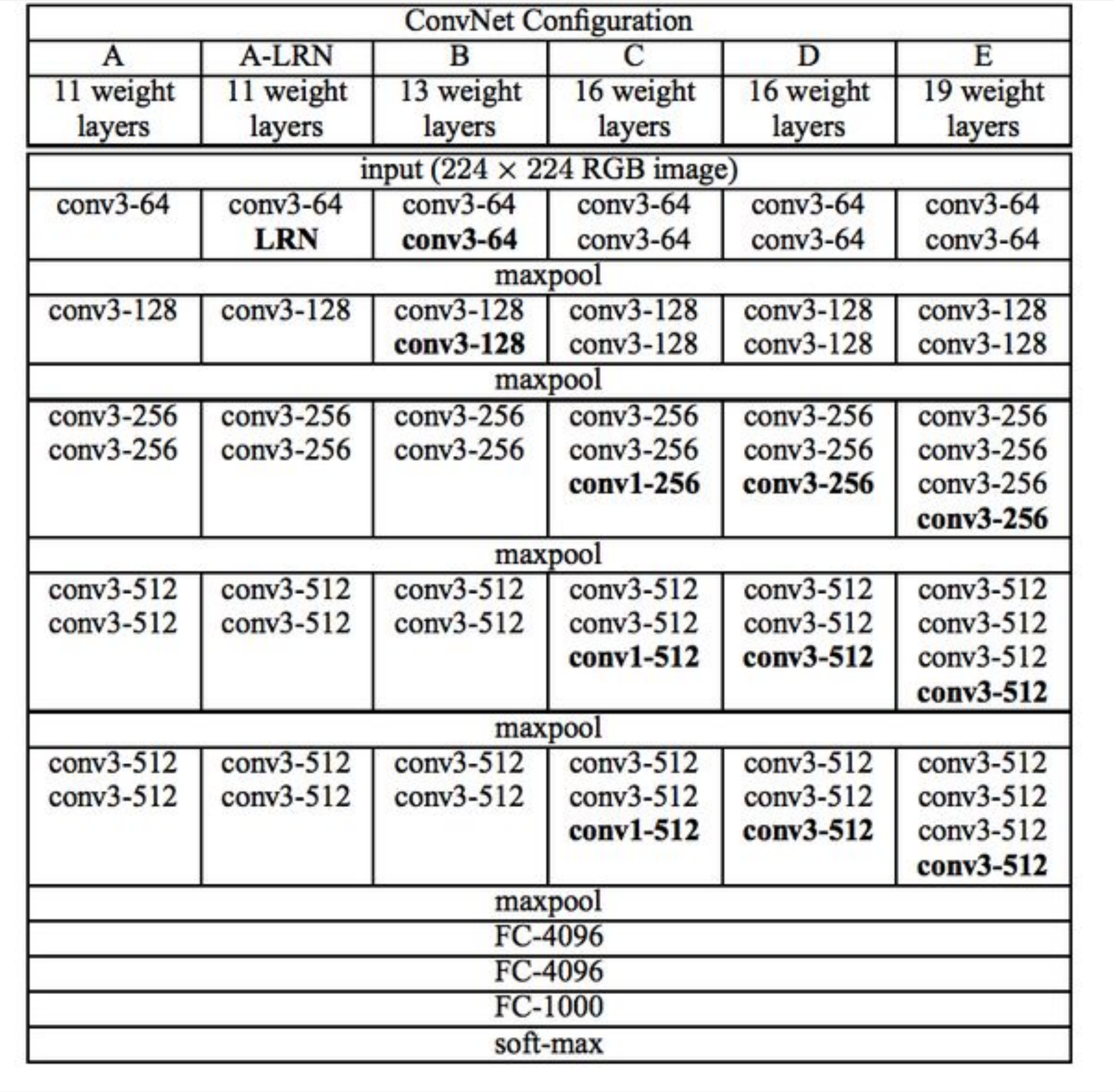
Aprendizaje Automático Profundo
(Deep Learning)

Dr. Facundo Quiroga - Dr. Franco Ronchetti

Arquitectura VGG

VGG: (Visual Geometry Group, Oxford) (notebook, paper)

- Ganador de ILSVRC 2012 (competición ImageNet)
- Ideas principales
  - Muchas convoluciones 3x3
    - 2 capas Conv(3x3) ~= 1 capa Conv(7x7)
  - Diseño en bloques
    - 5 bloques
    - Bloque: varias Conv2D seguido de MaxPooling
- 6 versiones
  - VGG D (16 capas) más popular
    - También llamada VGG16



ConvNet Configuration					
A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers
input (224 × 224 RGB image)					
conv3-64	conv3-64 **LRN**	conv3-64 **conv3-64**	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64
maxpool					
conv3-128	conv3-128	conv3-128 **conv3-128**	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128
maxpool					
conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256 **conv1-256**	conv3-256 conv3-256 **conv3-256**	conv3-256 conv3-256 conv3-256 **conv3-256**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

VGG16: Arquitectura por **bloques**

- Bloque
    - Lista de capas
    - Nombre para identificar más fácil

```
def block(feature_maps,n_conv,name):
   layers=[]
   for i in range(n_conv):
     layers.append(Conv2D(feature_maps, (3, 3),
                    activation='relu',
                    padding='same',
                    name=f'{name}_conv{i}'))
   layers.append(MaxPooling2D((2, 2), strides=(2, 2),
name=f'{name}_pool'))
   return layers
```

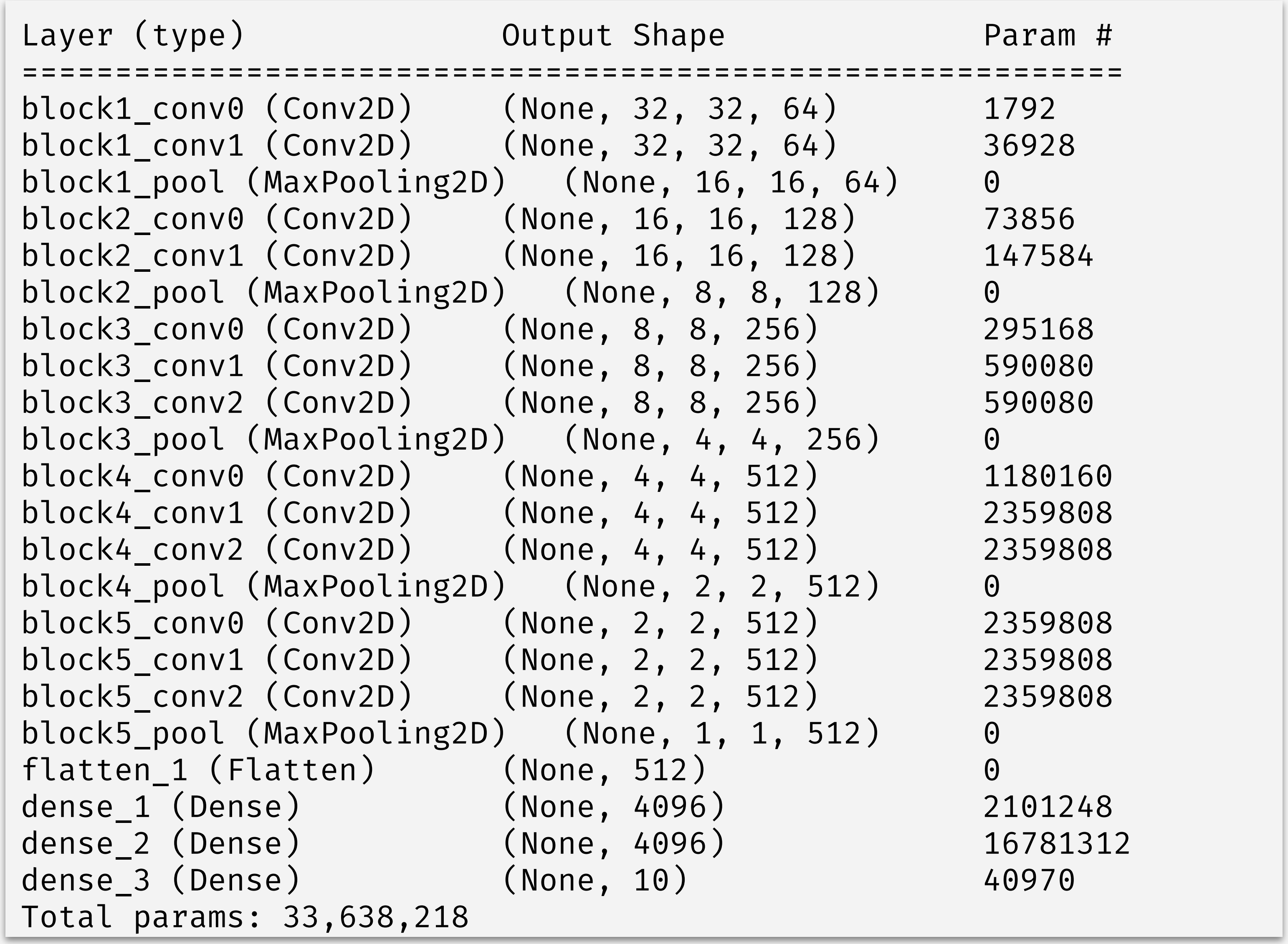
Lista de bloques

- 5 bloques
  - incrementa feature maps, achica tamaño espacial

```
fc_layers = [Flatten(),
             Dense(4096, activation='relu'),
             Dense(4096, activation='relu'),
             Dense(classes, activation='softmax')]
all_layers =  [InputLayer(input_shape)] +
               block(64 ,2,”block1”) +
               block(128,2,”block2”) +
               block(256,3,”block3”) +
               block(512,3,”block4”) +
               block(512,3,”block5”) +
                fc_layers
model = keras.Sequential(all_layers)
```

Relación #featuremaps/tamaño feature maps

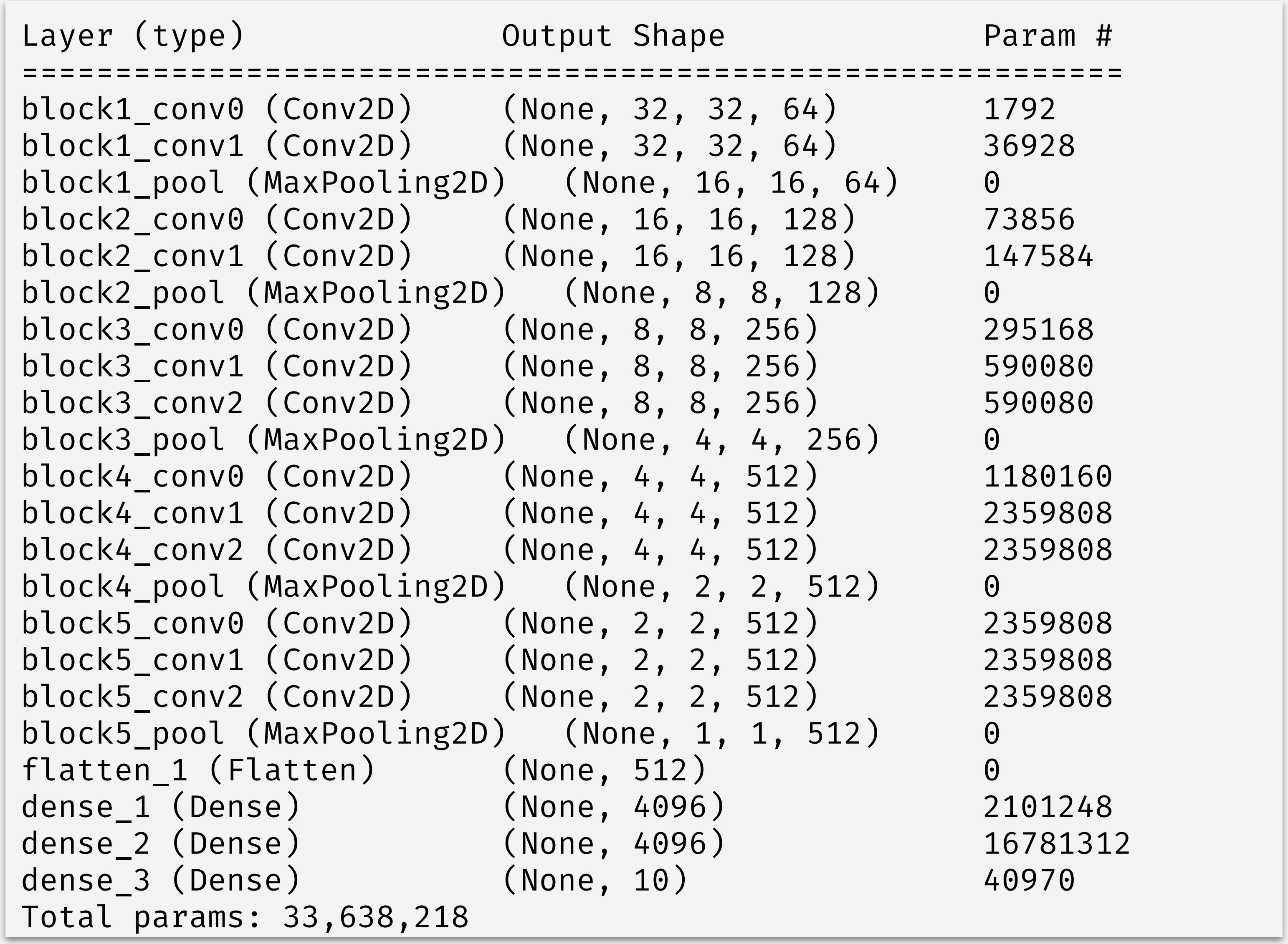
- Relación #feature maps / tamaño feature maps
  - 32x32x64 => 16x16x128 => 8x8x256 => 4x4x512 => 2x2x512
- En Imagenet, similar:
  - Resolución 224x224
  - 224x224x64 => 112x112x128 => 56x56x256 => 23x23x512 => 11x11x512



```
Layer (type)              Output Shape             Param #
===========================================================
block1_conv0 (Conv2D)     (None, 32, 32, 64)       1792
block1_conv1 (Conv2D)     (None, 32, 32, 64)       36928
block1_pool (MaxPooling2D)   (None, 16, 16, 64)    0
block2_conv0 (Conv2D)     (None, 16, 16, 128)      73856
block2_conv1 (Conv2D)     (None, 16, 16, 128)      147584
block2_pool (MaxPooling2D)   (None, 8, 8, 128)     0
block3_conv0 (Conv2D)     (None, 8, 8, 256)        295168
block3_conv1 (Conv2D)     (None, 8, 8, 256)        590080
block3_conv2 (Conv2D)     (None, 8, 8, 256)        590080
block3_pool (MaxPooling2D)   (None, 4, 4, 256)     0
block4_conv0 (Conv2D)     (None, 4, 4, 512)        1180160
block4_conv1 (Conv2D)     (None, 4, 4, 512)        2359808
block4_conv2 (Conv2D)     (None, 4, 4, 512)        2359808
block4_pool (MaxPooling2D)   (None, 2, 2, 512)     0
block5_conv0 (Conv2D)     (None, 2, 2, 512)        2359808
block5_conv1 (Conv2D)     (None, 2, 2, 512)        2359808
block5_conv2 (Conv2D)     (None, 2, 2, 512)        2359808
block5_pool (MaxPooling2D)   (None, 1, 1, 512)     0
flatten_1 (Flatten)       (None, 512)              0
dense_1 (Dense)           (None, 4096)             2101248
dense_2 (Dense)           (None, 4096)             16781312
dense_3 (Dense)           (None, 10)               40970
Total params: 33,638,218
```

Resumen

- Red muy grande
- 33M parámetros
    - Tarda en entrenar
- Se utiliza mucho como parte de otros modelos
- Diseño en bloques: nueva forma de pensar las redes
- Disponible en Keras keras.applications.vgg16.VGG16()



```
Layer (type)              Output Shape              Param #
===========================================================
block1_conv0 (Conv2D)     (None, 32, 32, 64)        1792
block1_conv1 (Conv2D)     (None, 32, 32, 64)        36928
block1_pool (MaxPooling2D)   (None, 16, 16, 64)     0
block2_conv0 (Conv2D)     (None, 16, 16, 128)       73856
block2_conv1 (Conv2D)     (None, 16, 16, 128)       147584
block2_pool (MaxPooling2D)   (None, 8, 8, 128)      0
block3_conv0 (Conv2D)     (None, 8, 8, 256)         295168
block3_conv1 (Conv2D)     (None, 8, 8, 256)         590080
block3_conv2 (Conv2D)     (None, 8, 8, 256)         590080
block3_pool (MaxPooling2D)   (None, 4, 4, 256)      0
block4_conv0 (Conv2D)     (None, 4, 4, 512)         1180160
block4_conv1 (Conv2D)     (None, 4, 4, 512)         2359808
block4_conv2 (Conv2D)     (None, 4, 4, 512)         2359808
block4_pool (MaxPooling2D)   (None, 2, 2, 512)      0
block5_conv0 (Conv2D)     (None, 2, 2, 512)         2359808
block5_conv1 (Conv2D)     (None, 2, 2, 512)         2359808
block5_conv2 (Conv2D)     (None, 2, 2, 512)         2359808
block5_pool (MaxPooling2D)   (None, 1, 1, 512)      0
flatten_1 (Flatten)       (None, 512)               0
dense_1 (Dense)           (None, 4096)              2101248
dense_2 (Dense)           (None, 4096)              16781312
dense_3 (Dense)           (None, 10)                40970
Total params: 33,638,218
```