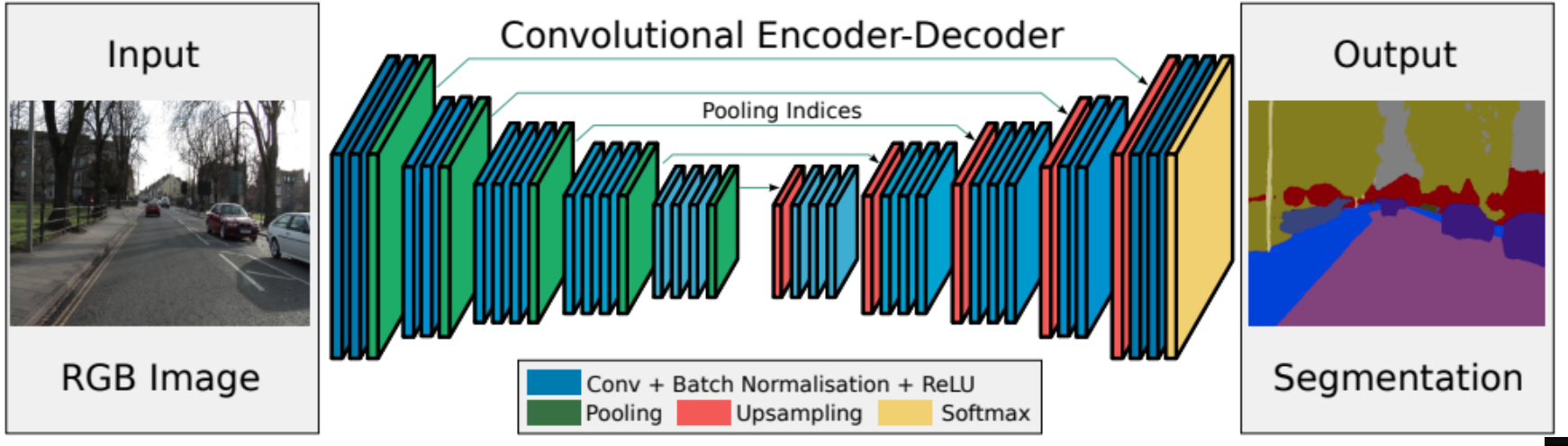
Introduction to Semantic Segmentation

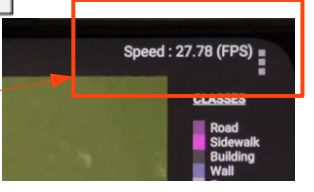
https://arxiv.org/pdf/1511.00561.pdf

SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation

Vijay Badrinarayanan, Alex Kendall, Roberto Cipolla, *Senior Member, IEEE,*



Qualcomm sample video:
https://www.youtube.com/watch?v=hGrJ3zuuvRQ



DeepLab for Dense Pixel Labeling Semantic Image Segmentation

https://github.com/google-research/deeplab2 and older one https://github.com/tensorflow/models/tree/master/research/deeplab

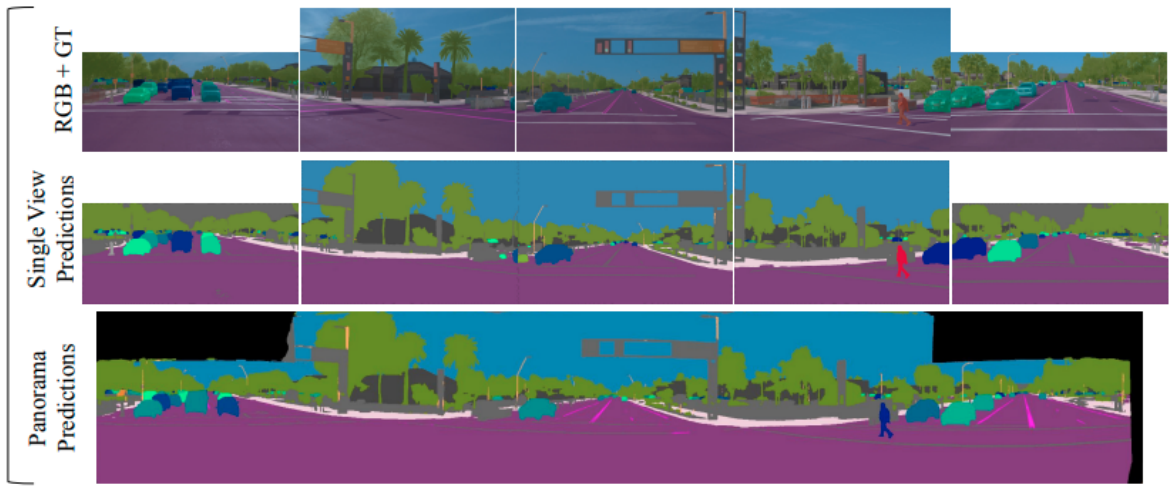
DeepLab2: (1) a TensorFlow library for deep labeling for a unified and state-of-the-art TensorFlow codebase for dense pixel labeling, including, but not limited to semantic segmentation, instance segmentation, panoptic segmentation, depth estimation, or even video panoptic segmentation.
(2) Deep labeling assigns a predicted value for each pixel.

Waymo Open Dataset: Panoramic Video Panoptic Segmentation

Waymo Open Dataset: Panoramic Video Panoptic Segmentation

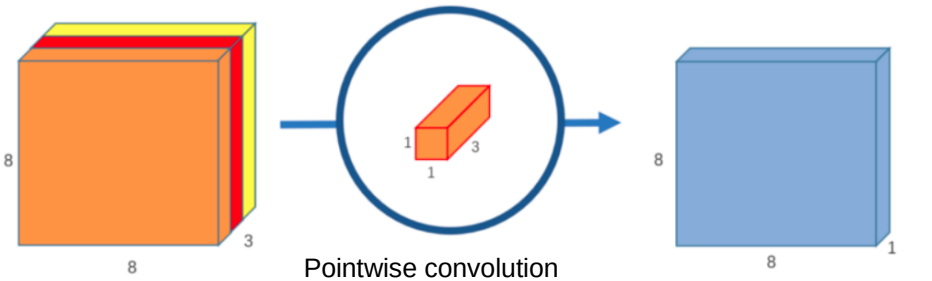
Jieru Mei[1*] Alex Zihao Zhu[2] Xinchen Yan[2] Hang Yan[2]
Siyuan Qiao[3] Yukun Zhu[3] Liang-Chieh Chen[3]
Henrik Kretzschmar[2] Dragomir Anguelov[2]

[1]Johns Hopkins University [2]Waymo LLC [3]Google Research



Semantic Decoder Python

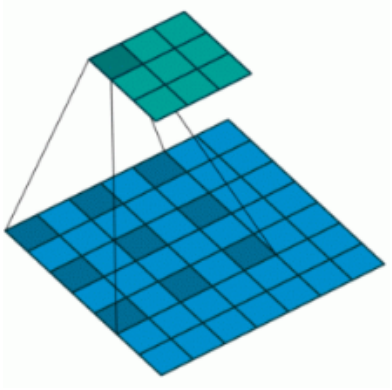
1. Pointwise convolution, e.g., 1x1xk convolution;
2. Depthwise convolution: Atrous convolution;



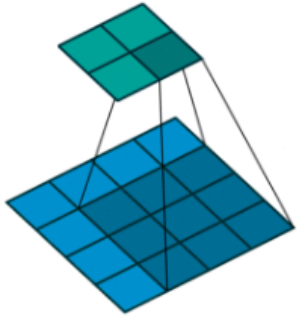
Pointwise convolution

Depthwise convolution

https://www.analyticsvidhya.com/blog/2019/02/tutorial-semantic-segmentation-google-deeplab/



Dilated Convolution



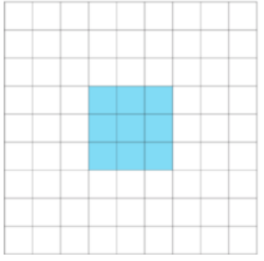
Standard Convolution

Atrous convolutions

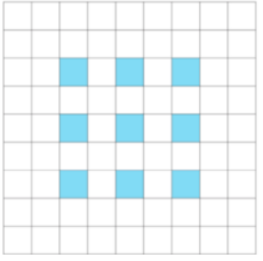
$$y[i] = \sum_{k} x[i + r \cdot k] w[k]$$

DeepLab uses atrous convolution with rates 6, 12 and 18.

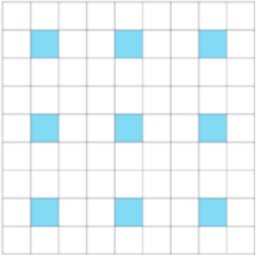
r=1



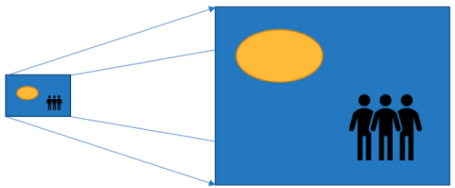
r=2



r=3



Convolution Up-sampling



1

Nearest Neighbor

1	2
3	4

Input: 2 x 2

1	1	2	2
1	1	2	2
3	3	4	4
3	3	4	4

Output: 4 x 4

3

Bi-Linear Interpolation

10	20
30	40

2x2

2x

10	12	17	20
15	17	22	25
25	27	32	35
30	32	37	40

4x4

https://naokishibuya.medium.com/up-sampling-with-transposed-convolution-9ae4f2df52d0

3

"Bed of Nails"

1	2
3	4

Input: 2 x 2

1	0	2	0
0	0	0	0
3	0	4	0
0	0	0	0

Output: 4 x 4

4

Max Pooling
Remember which element was max!

1	2	6	3
3	5	2	1
1	2	2	1
7	3	4	8

Input: 4 x 4

5	6
7	8

Output: 2 x 2

Rest of the network

Max Unpooling
Use positions from pooling layer

1	2
3	4

Input: 2 x 2

0	0	2	0
0	1	0	0
0	0	0	0
3	0	0	4

Output: 4 x 4

https://towardsdatascience.com/transposed-convolution-demystified-84ca81b4baba

Transposed Convolution Up-sampling

Credit of the example illustration:
https://towardsdatascience.com/transposed-convolution-demystified-84ca81b4baba

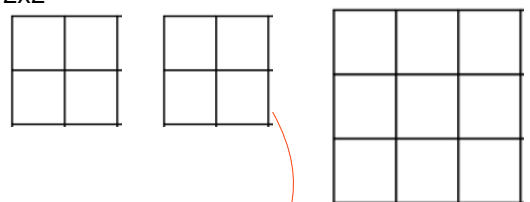
https://naokishibuya.medium.com/up-sampling-with-transposed-convolution-9ae4f2df52d0

1. Consider a 2x2 encoded feature map which needs to be upsampled to a 3x3 feature map.

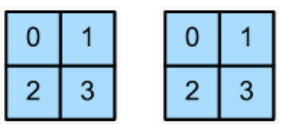
Input image Feature map: 2x2

Kernel 2x2

Upsampled output image: 3x3

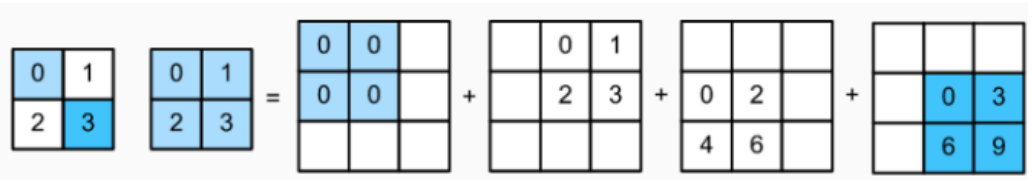


kernel of size 2x2 with unit stride and zero paddin

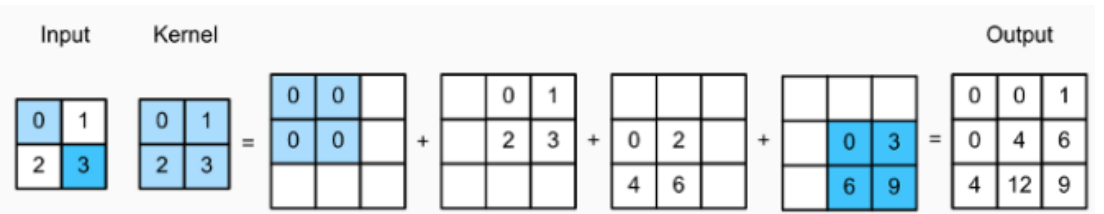


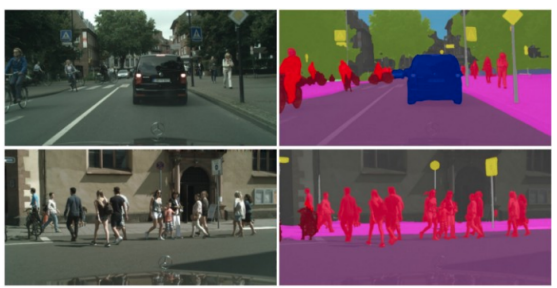
Step 1. Feature map and the kernel

Step 2. Transposed convolution for each pixel in the feature map



Step 3. Add output at each pixel location together to form upsampled image





Transposed Convolution Up-sampling with Python

Python Reference Code (Untested)

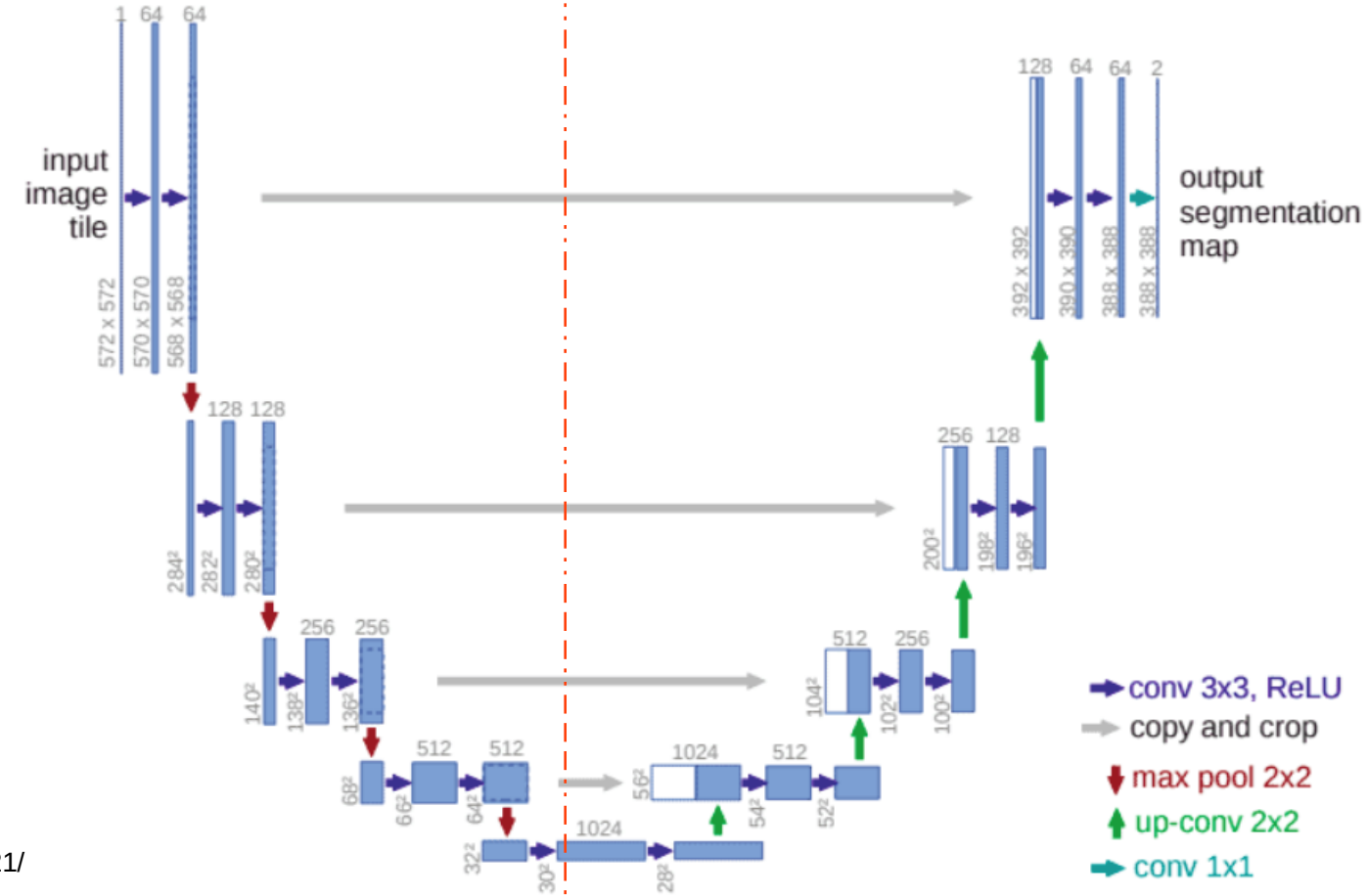
```
def apply_conv(data, kernel, conv):
"""
Args:
data (NDArray): input data.
kernel (NDArray): convolution's kernel parameters.
conv (Block): convolutional layer.
Returns:
NDArray: output data (after applying convolution).
"""
# add dimensions for batch and channels if necessary
while data.ndim < len(conv.weight.shape):
data = data.expand_dims(0)
# add dimensions for channels and in_channels if necessary
while kernel.ndim < len(conv.weight.shape):
kernel = kernel.expand_dims(0)
# check if transpose convolution
if type(conv).__name__.endswith("Transpose"):
in_channel_idx = 0
else:
in_channel_idx = 1
# initialize and set weight
conv._in_channels = kernel.shape[in_channel_idx]
conv.initialize()
conv.weight.set_data(kernel)
return conv(data)
```

https://medium.com/apache-mxnet/transposed-convolutions-explained-with-ms-excel-52d13030c7e8

Unet For Semantic Segmentation

https://towardsdatascience.com/understanding-semantic-segmentation-with-unet-6be4f42d4b47

Unet For Semantic Segmentation

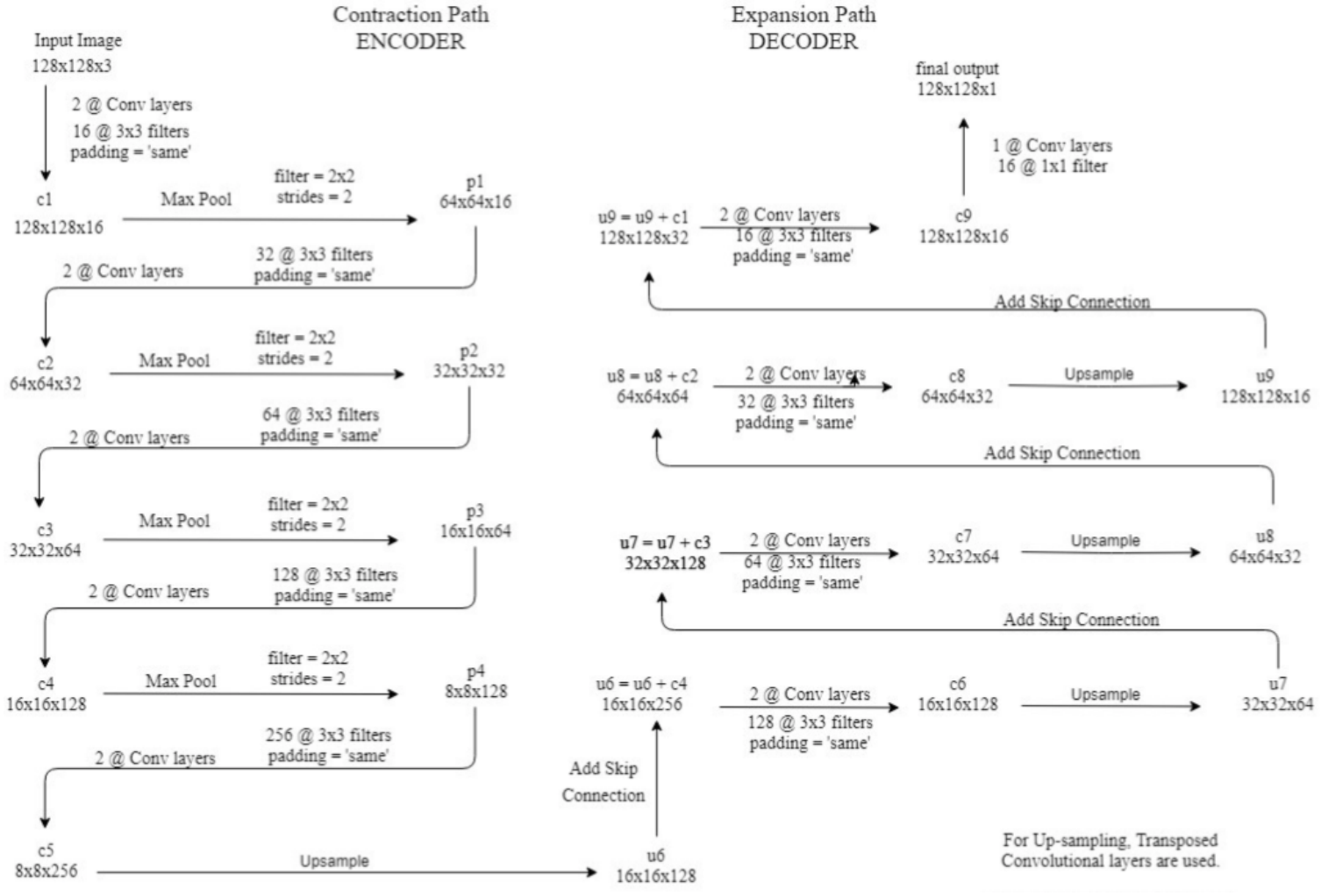


https://www.analyticsvidhya.com/blog/2019/02/tutorial-semantic-segmentation-google-deeplab/

Code sample and tutorial:
https://pyimagesearch.com/2022/02/21/u-net-image-segmentation-in-keras/

Unet For Semantic Segmentation

https://towardsdatascience.com/understanding-semantic-segmentation-with-unet-6be4f42d4b47



GIPHY and Other Tools for Annotation of Images for Semantic Segmentation

https://giphy.com/gifs/R0dnXaKJowlR2yL5CG

Labelbox

Supervisely

Fritz AI

RectLabel

Anolytics

Playment

Appen

Scale.ai

https://cnvrg.io/semantic-segmentation/

Six (6) Useful Image Segmentation Datasets And Python Usage

https://cnvrg.io/semantic-segmentation/

coco:

```
import tensorflow_datasets as tfds
(X_train, X_test), ds_info = tfds.load(
    'coco',
    split=['train', 'test'],
    shuffle_files=True,
    as_supervised=True,
    with_info=True,
)
```

waymo:

```
import tensorflow_datasets as tfds
(X_train, X_test), ds_info = tfds.load(
    â€˜waymo_open_datasetâ€™,
    split=['train', 'test'],
    shuffle_files=True,
    as_supervised=True,
    with_info=True,
)
```

PASCAL:

```
import tensorflow_datasets as tfds
(X_train, X_test), ds_info = tfds.load(
    â€˜vocâ€™,
    split=['train', 'test'],
    shuffle_files=True,
    as_supervised=True,
    with_info=True,
)
```