Anime-fy Yourself

Using Core ML

Yono Mittlefehldt — @yonomitt — 2021

Who is Yono?

Co-creator of Gus on the Go

Who is Yono?

Co-creator of Gus on the Go

Freelance software developer
specializing in Computer Vision and iOS

Who is Yono?

Co-creator of Gus on the Go

Freelance software developer
specializing in Computer Vision and iOS

Tutorial writer for raywenderlich.com

Who is Yono?

Co-creator of Gus on the Go

Freelance software developer
specializing in Computer Vision and iOS

Tutorial writer for raywenderlich.com

Currently rewatching all of Futurama
while exercising on a stationary bike

Converting a Model to Core ML

Model

Model

A function

$$f(x, y, z) = w_1x + w_2y + w_3z + b$$

Model

A function

One or more inputs

$$f(\mathbf{x}, \mathbf{y}, \mathbf{z}) = w_1\mathbf{x} + w_2\mathbf{y} + w_3\mathbf{z} + b$$

Model

A function

- One or more inputs
- One or more outputs

$$\mathbf{f(x, y, z)} = w_1x + w_2y + w_3z + b$$

Model

A function

- One or more inputs
- One or more outputs

Parameters

$$f(x, y, z) = w_1x + w_2y + w_3z + b$$

Model

A function

- One or more inputs
- One or more outputs

Parameters

- Weights (multiplied)

$$f(x, y, z) = \mathbf{w}_1 x + \mathbf{w}_2 y + \mathbf{w}_3 z + b$$

Model

A function

- One or more inputs
- One or more outputs

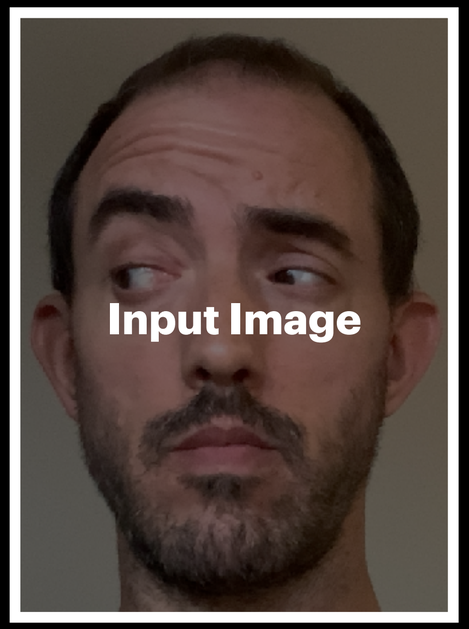
Parameters

- Weights (multiplied)
- Biases (added)

$$f(x, y, z) = w_1x + w_2y + w_3z + \mathbf{b}$$

Model Types

Image Classification

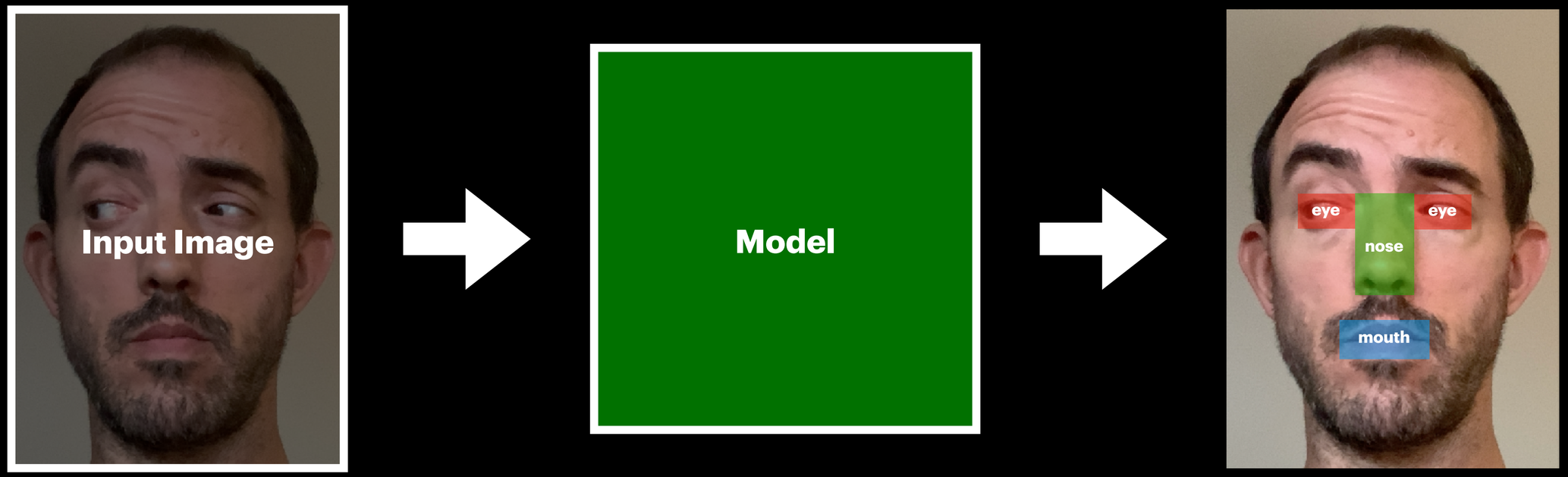


Model

“face”

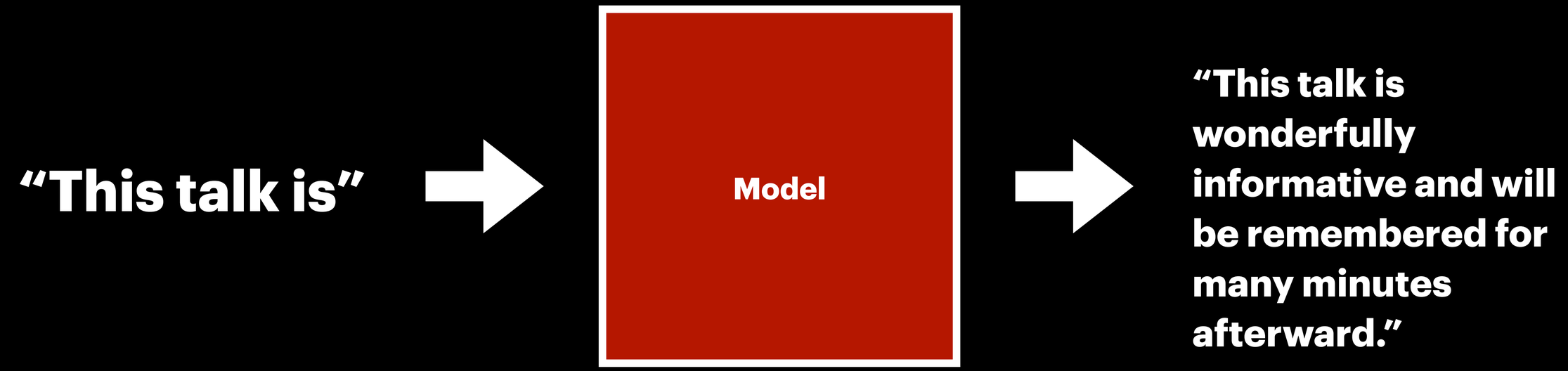
Model Types

Object Detection



Model Types

Text Generation



Models Need Numbers

Encode Inputs and Decode Outputs

Models Need Numbers

Encode Inputs and Decode Outputs

Images can be expressed as multidimensional arrays of pixel values

Models Need Numbers

Encode Inputs and Decode Outputs

Images can be expressed as multidimensional arrays of pixel values

- Separate values for each RGB channel

Models Need Numbers

Encode Inputs and Decode Outputs

Images can be expressed as multidimensional arrays of pixel values

- Separate values for each RGB channel
- Scale is usually important — [0, 255] vs [0.0, 1.0] vs [-1.0, 1.0]

Models Need Numbers

Encode Inputs and Decode Outputs

Images can be expressed as multidimensional arrays of pixel values

- Separate values for each RGB channel
- Scale is usually important — [0, 255] vs [0.0, 1.0] vs [-1.0, 1.0]

Text can be encoded as integers

Models Need Numbers

Encode Inputs and Decode Outputs

Images can be expressed as multidimensional arrays of pixel values

- Separate values for each RGB channel
- Scale is usually important — [0, 255] vs [0.0, 1.0] vs [-1.0, 1.0]

Text can be encoded as integers

- “This”, “this” -> 55
- “talk” -> 743

Models Need Numbers

Encode Inputs and Decode Outputs

Images can be expressed as multidimensional arrays of pixel values

- Separate values for each RGB channel
- Scale is usually important — [0, 255] vs [0.0, 1.0] vs [-1.0, 1.0]

Text can be encoded as integers

- “This”, “this” -> 55
- “talk” -> 743
- These values may be then encoded as one-hot vectors

Frameworks to Create Models

PyTorch

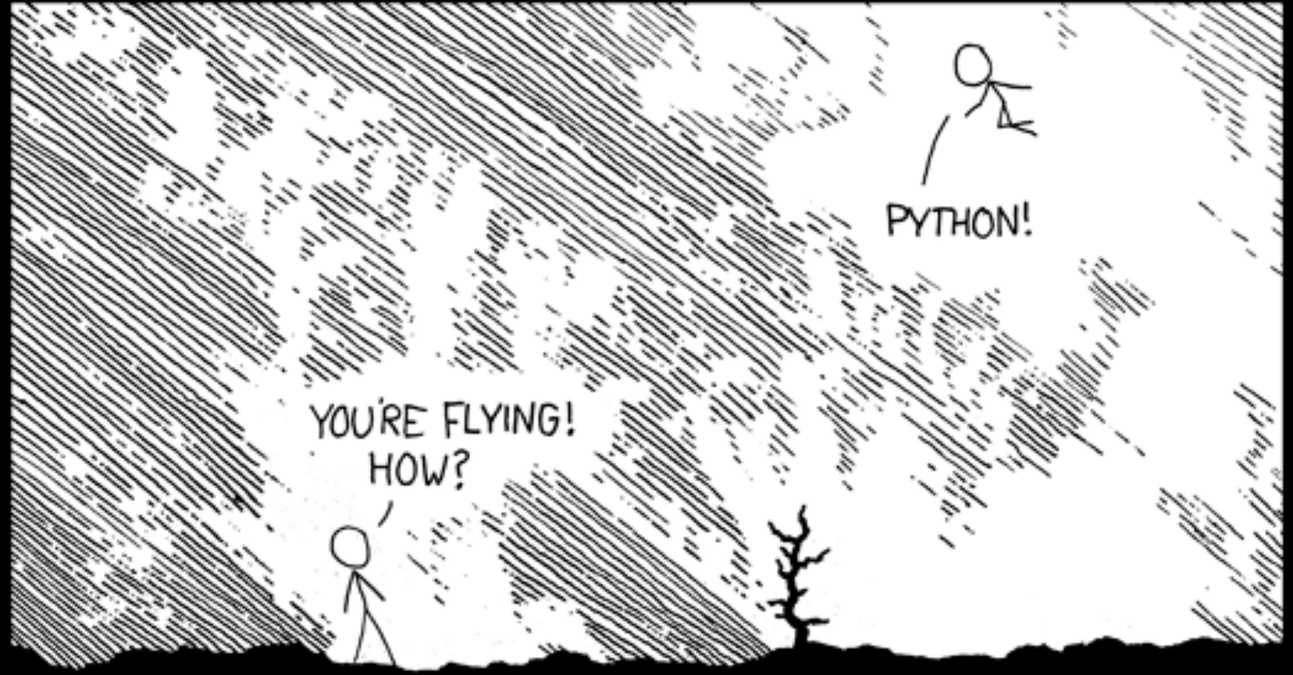
Tensorflow

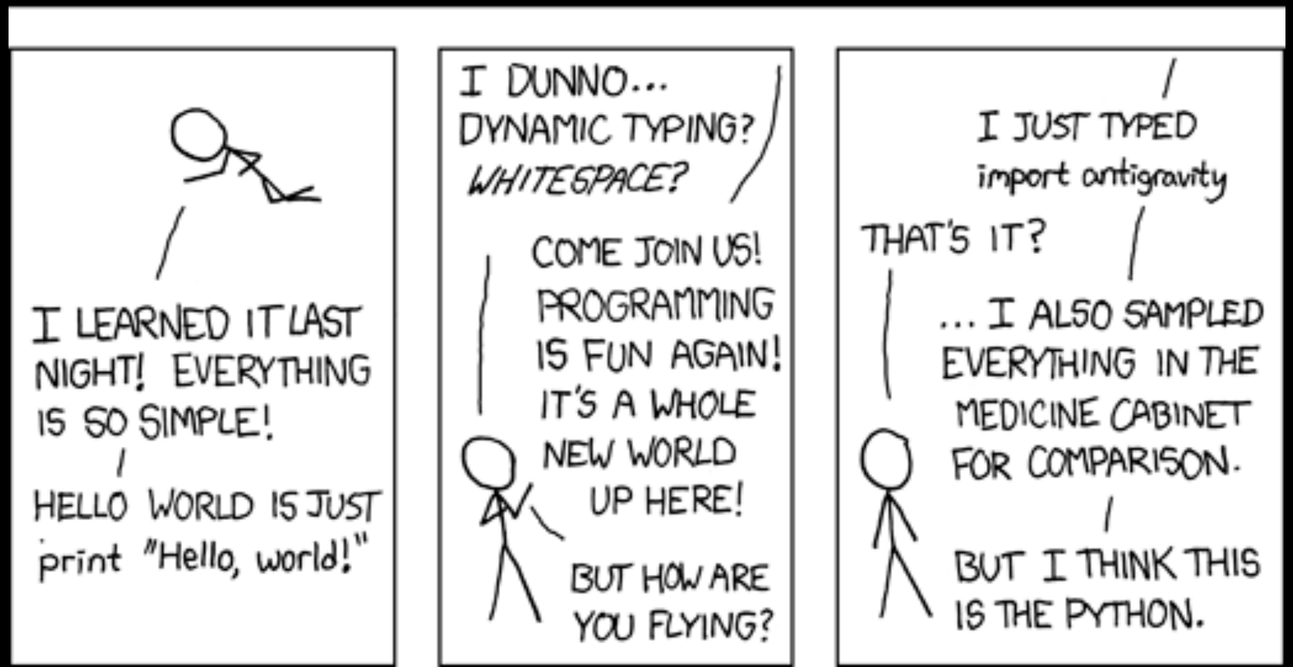
Turi Create / Create ML

...

scikit-learn

Py in PyTorch


PYTHON!
YOU'RE FLYING! HOW?


I LEARNED IT LAST NIGHT! EVERYTHING IS SO SIMPLE!
HELLO WORLD IS JUST print "Hello, world!"
I DUNNO... DYNAMIC TYPING? WHITESPACE?
COME JOIN US! PROGRAMMING IS FUN AGAIN! IT'S A WHOLE NEW WORLD UP HERE!
BUT HOW ARE YOU FLYING?
I JUST TYPED import antigravity
THAT'S IT?
... I ALSO SAMPLED EVERYTHING IN THE MEDICINE CABINET FOR COMPARISON.
BUT I THINK THIS IS THE PYTHON.

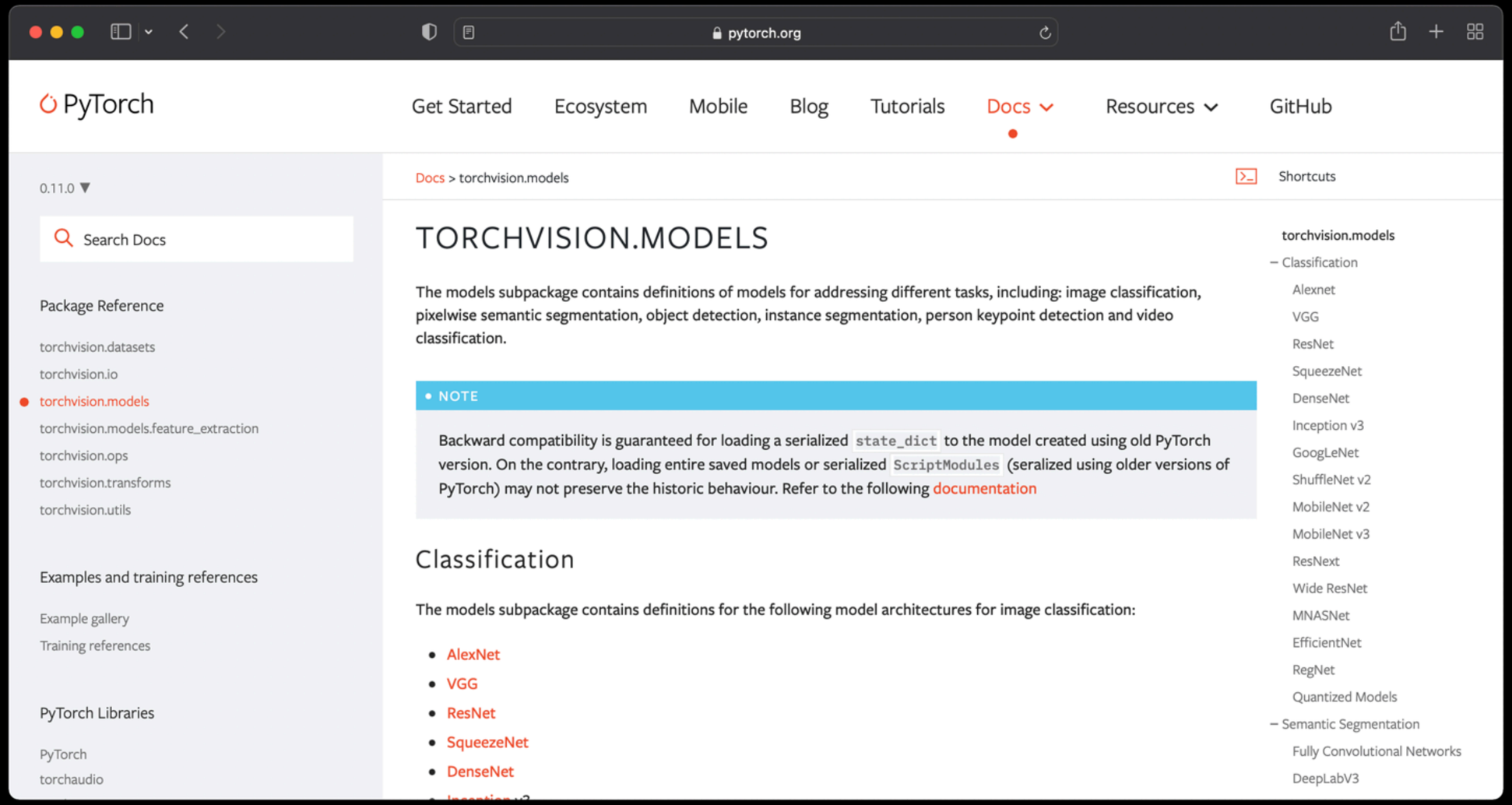
xkcd.com/353/

Hey! What about the alt-text?

I wrote 20 short programs in Python yesterday. It was wonderful. Perl, I'm leaving you.

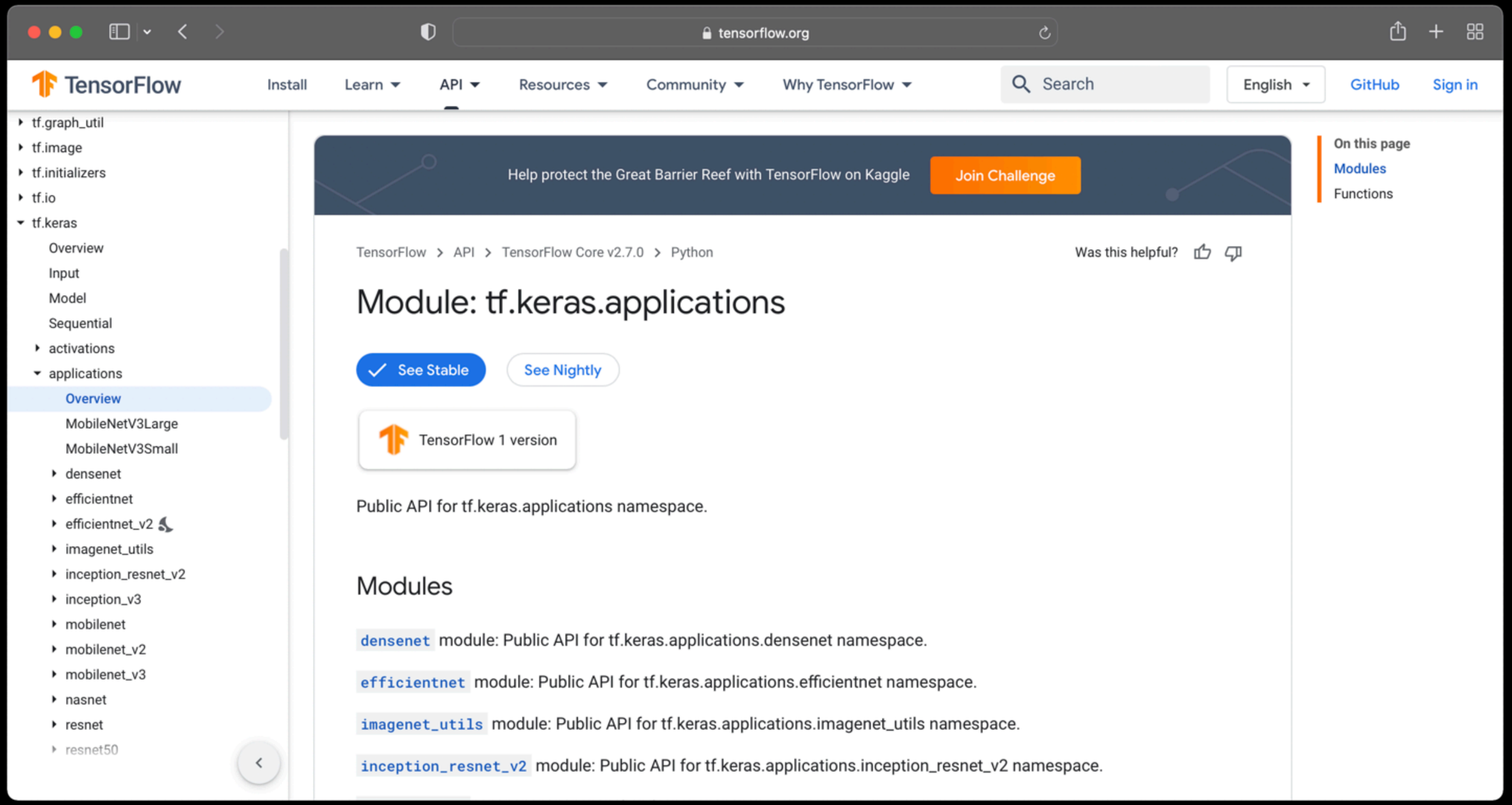
Models

Models - Built In



pytorch.org/vision/stable/models.html

Models - Built In



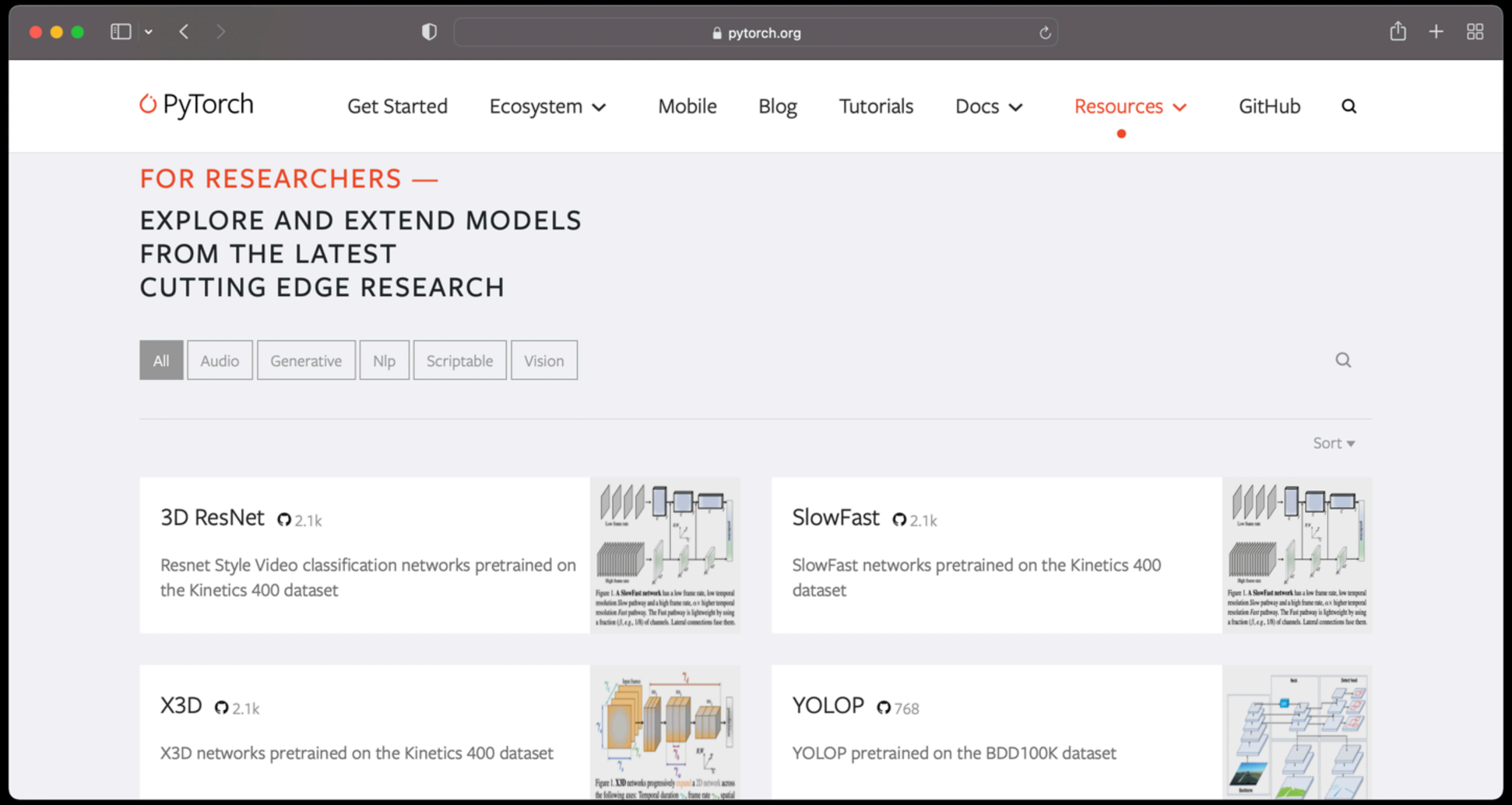
tensorflow.org/api_docs/python/tf/keras/applications

Models - Built In

```
import torchvision.models as models
model = models.efficientnet_b5(pretrained=True)
```

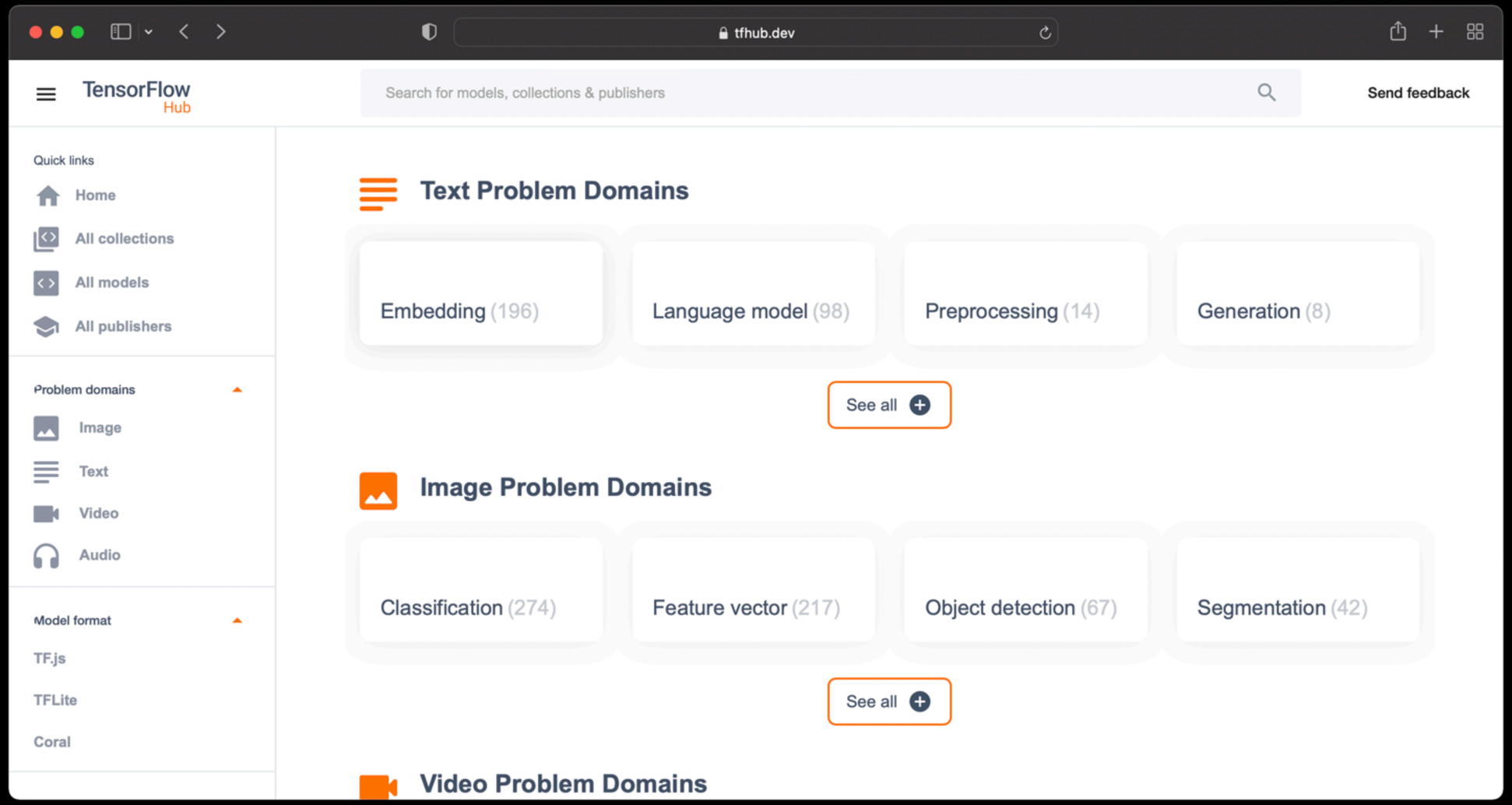
```
import tensorflow as tf
model = tf.keras.applications.efficientnet.EfficientNetB5(weights='imagenet')
```

Models - Hubs



pytorch.org/hub/

Models - Hubs



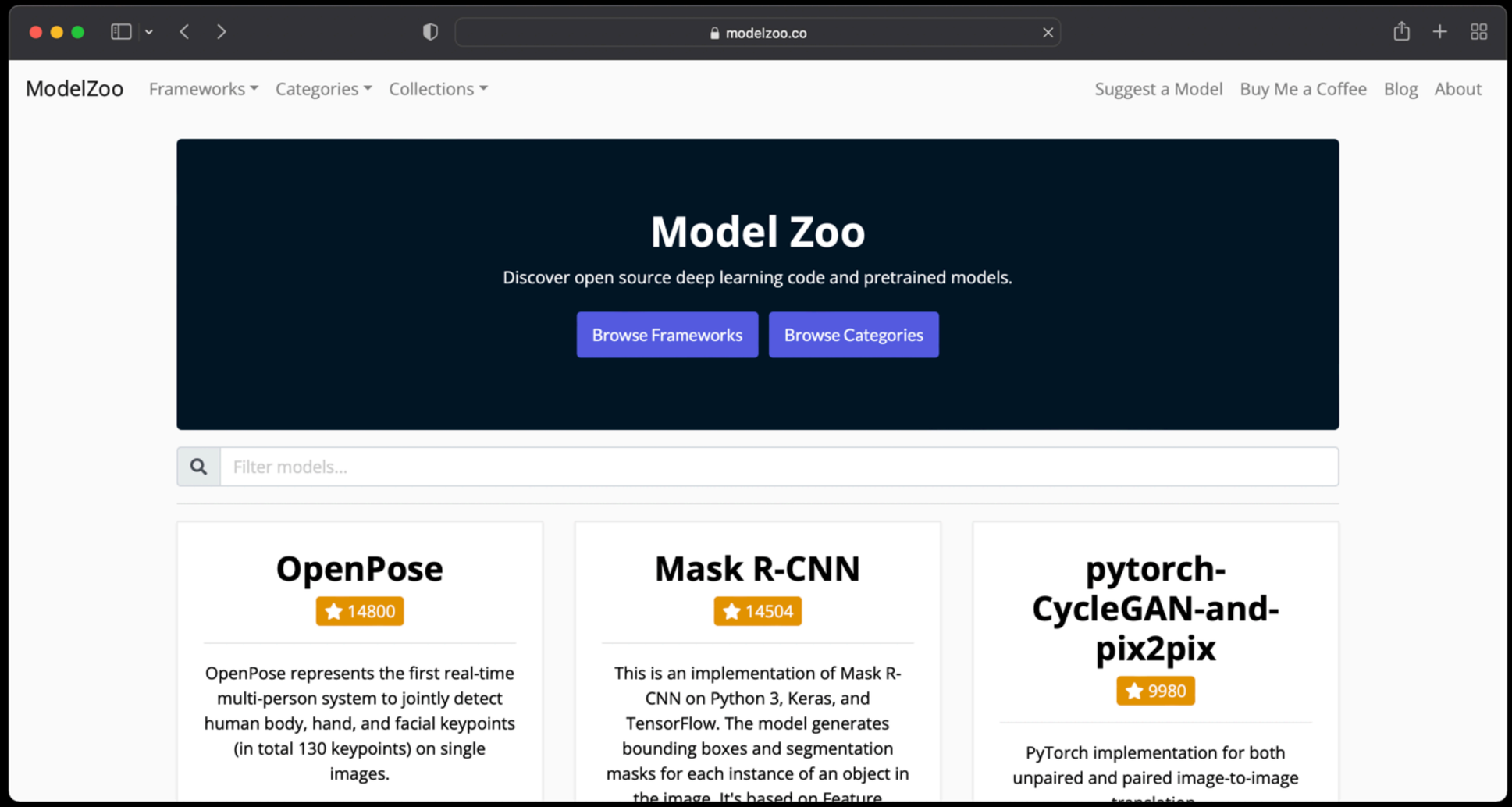
tfhub.dev

Models - Hubs

```
import torch
model = torch.hub.load('ultralytics/yolov5', 'yolov5s', pretrained=True)
```

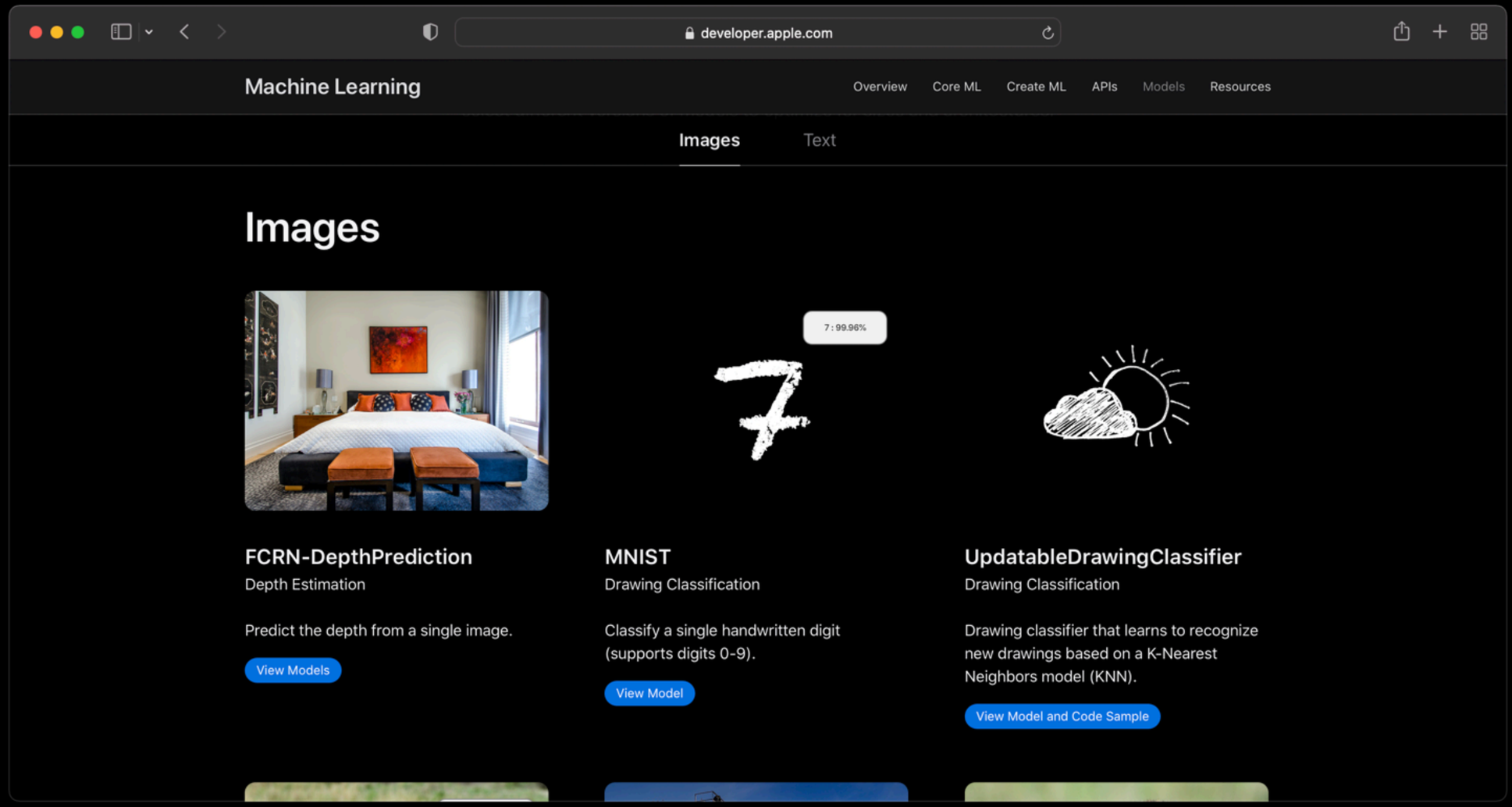
```
import tensorflow_hub as tfhub
model = tfhub.load('https://tfhub.dev/tensorflow/efficientnet/b5/classification/1')
```

Models - Model Zoos



modelzoo.co

Models - Model Zoos



developer.apple.com/machine-learning/models/


YOU GET A MODEL
AND YOU GET A MODEL, AND YOU...
imgflip.com

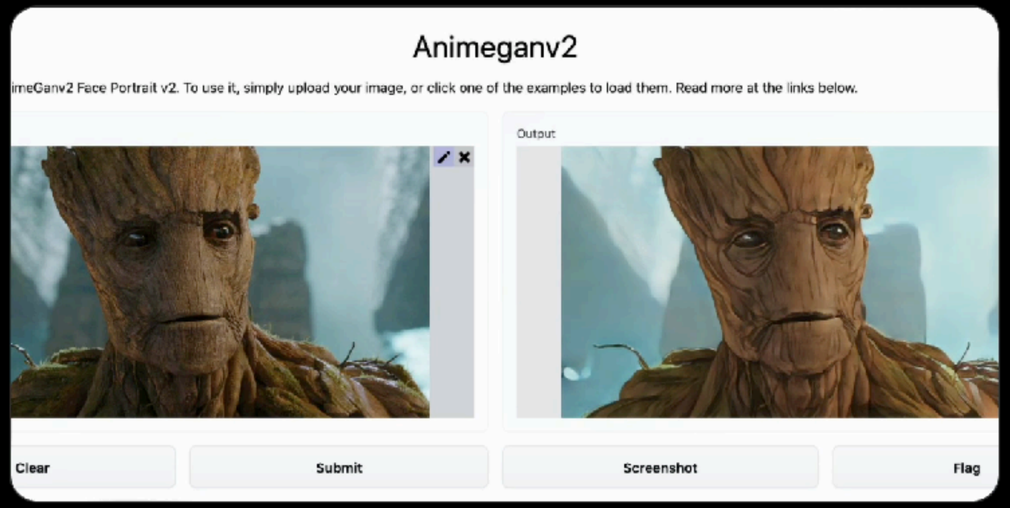
Models

Models - Twitter?

← Thread

AK
@ak92501

.@Gradio Demo for AnimeGANv2 Face Portrait v2 now on @huggingface Spaces
demo: huggingface.co/spaces/akhaliq...
github: github.com/bryandlee/anim...



6:13 PM · Nov 6, 2021 · Twitter Web App

349 Retweets **351** Quote Tweets **2,022** Likes

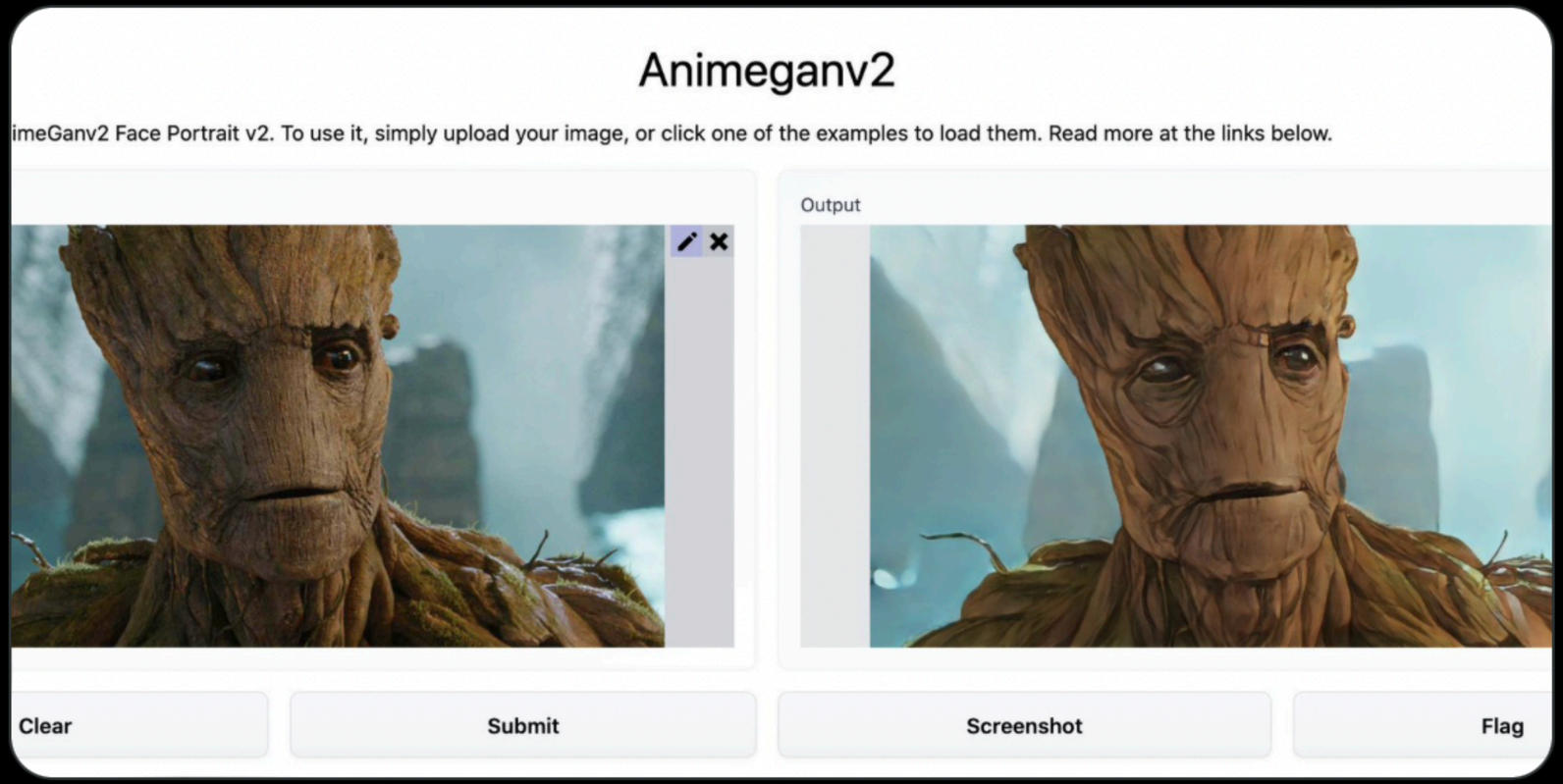
Tweet your reply

Reply

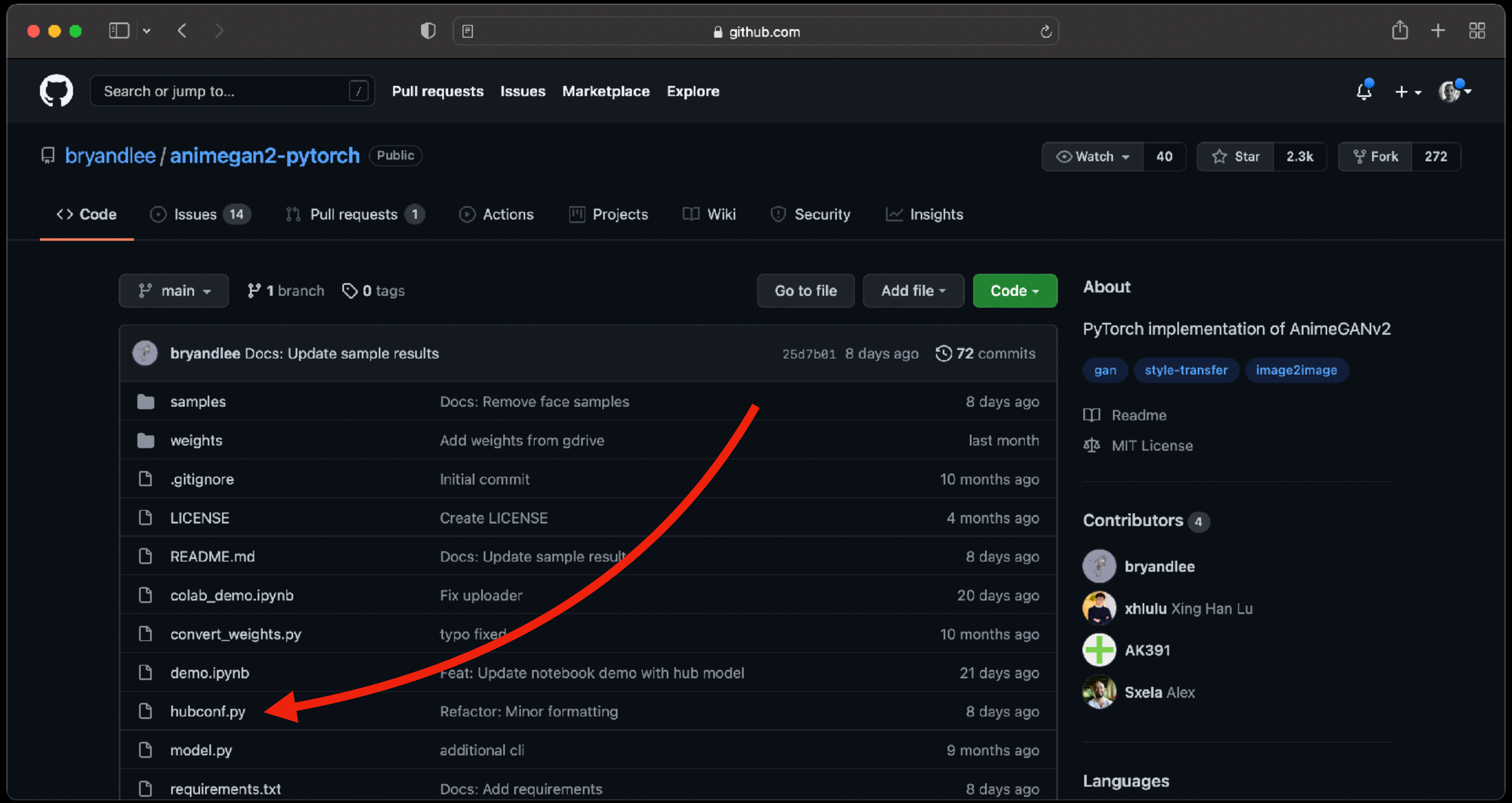
AK @ak92501 · Nov 6
Replying to @ak92501

AK
@ak92501

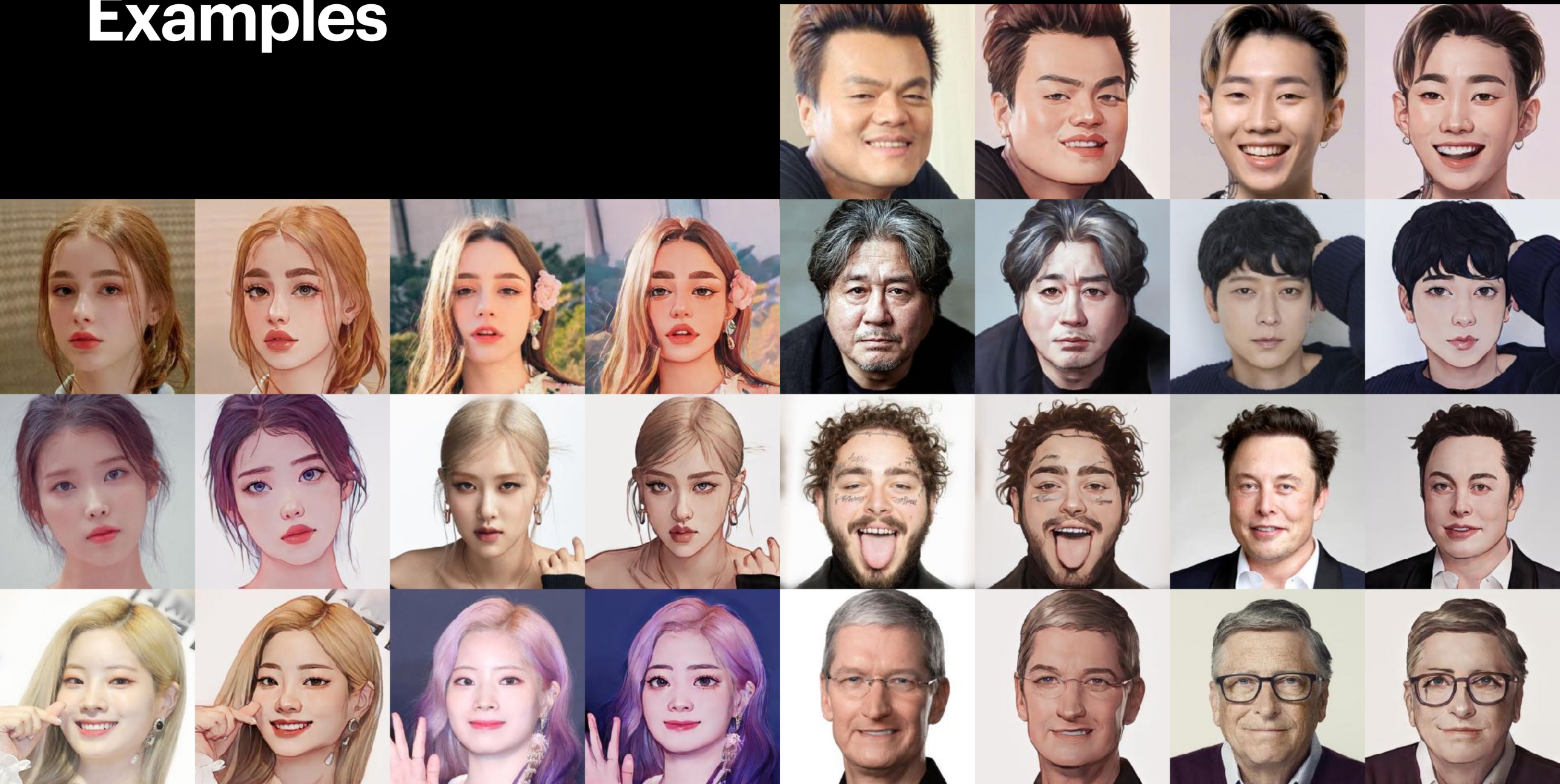
.@Gradio Demo for AnimeGANv2 Face Portrait v2 now on @huggingface Spaces
demo: huggingface.co/spaces/akhaliq...
github: github.com/bryandlee/anim...



6:13 PM · Nov 6, 2021 · Twitter Web App


github.com
Search or jump to...
Pull requests
Issues
Marketplace
Explore
bryandlee / animegan2-pytorch
Public
Watch 40
Star 2.3k
Fork 272
Code
Issues 14
Pull requests 1
Actions
Projects
Wiki
Security
Insights
main
1 branch
0 tags
Go to file
Add file
Code
bryandlee Docs: Update sample results
25d7b01 8 days ago
72 commits
samples Docs: Remove face samples 8 days ago
weights Add weights from gdrive last month
.gitignore Initial commit 10 months ago
LICENSE Create LICENSE 4 months ago
README.md Docs: Update sample result 8 days ago
colab_demo.ipynb Fix uploader 20 days ago
convert_weights.py typo fixed 10 months ago
demo.ipynb Feat: Update notebook demo with hub model 21 days ago
hubconf.py Refactor: Minor formatting 8 days ago
model.py additional cli 9 months ago
requirements.txt Docs: Add requirements 8 days ago
About
PyTorch implementation of AnimeGANv2
gan
style-transfer
image2image
Readme
MIT License
Contributors 4
bryandlee
xhlulu Xing Han Lu
AK391
Sxela Alex
Languages

Examples



But we don’t want a PyTorch model!

We want a Core ML model...

Core ML Tools

Convert models from popular frameworks to Core ML using Python

- PyTorch
- Tensorflow
- scikit-learn

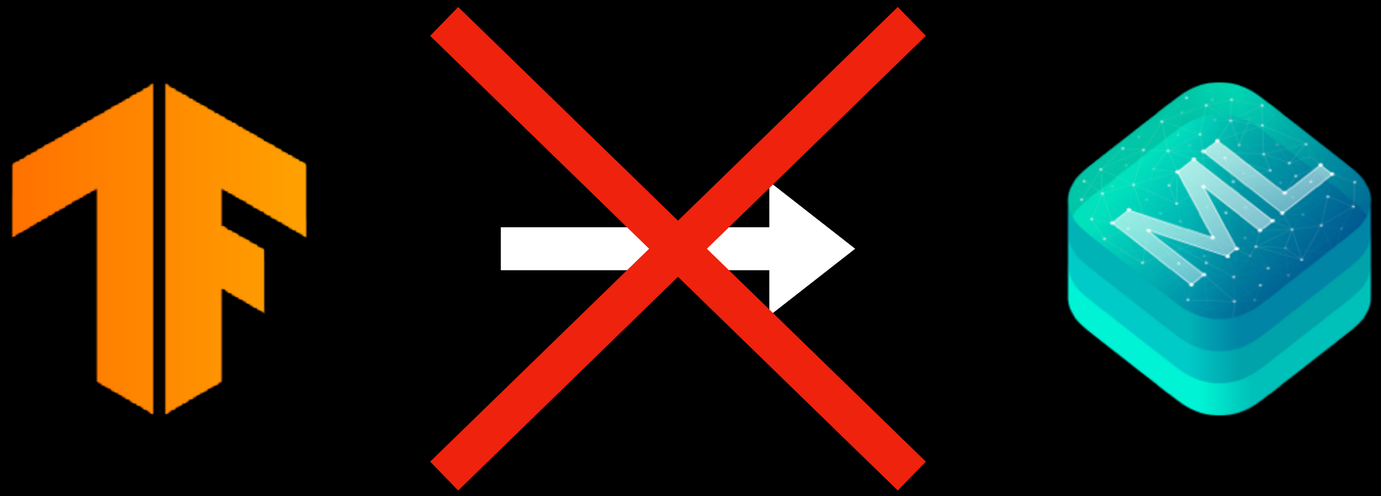
Read, write, and optimize Core ML models

Core ML Tools

Long, long ago

Core ML Tools

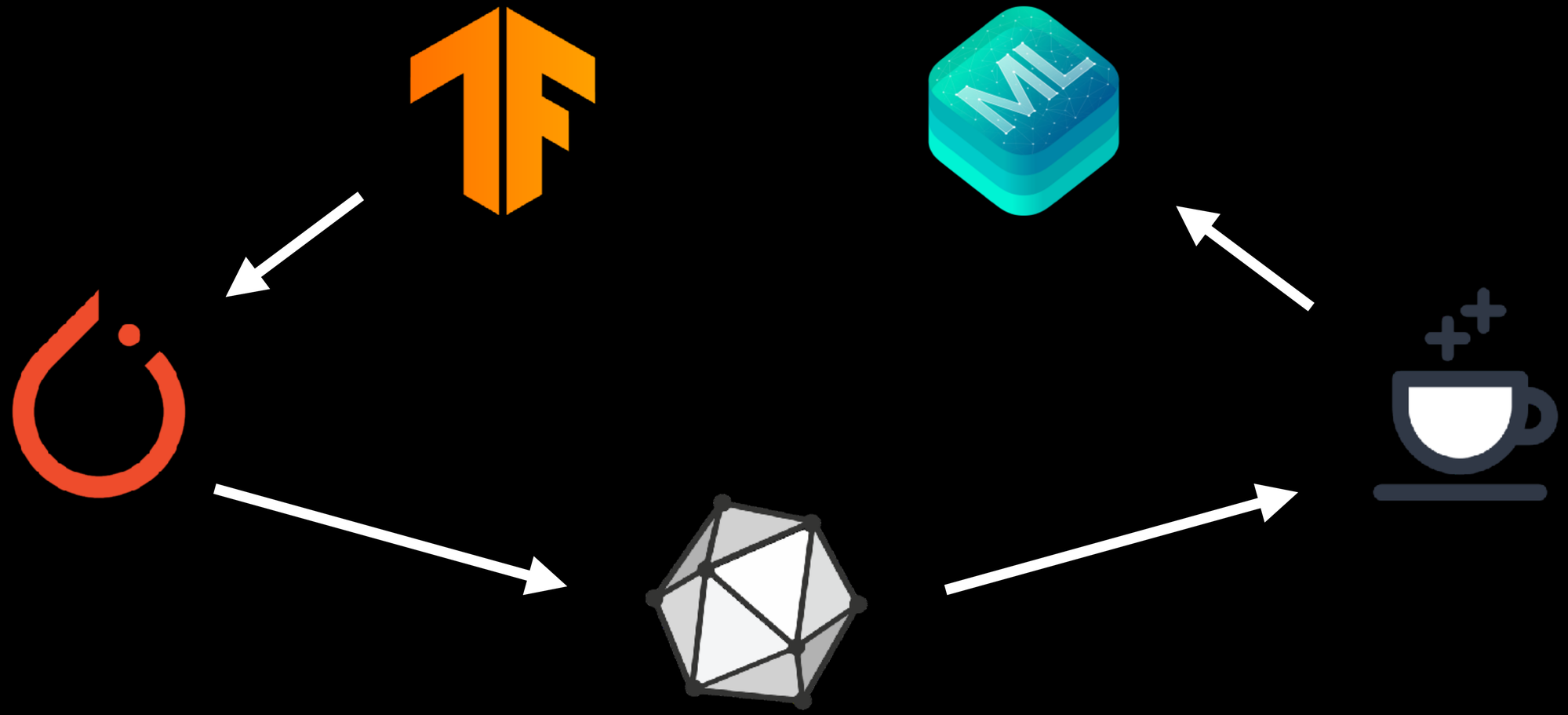
Long, long ago (2019)



EfficientNet-B5

Core ML Tools

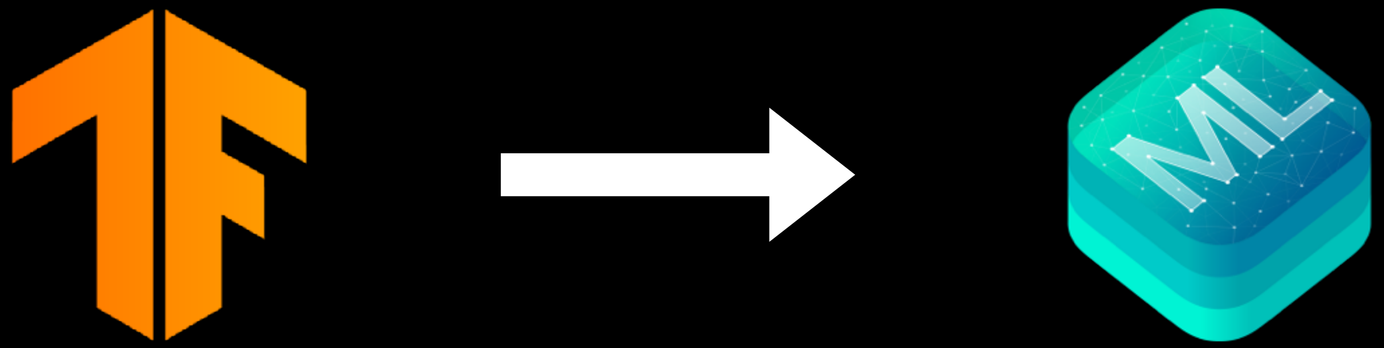
Long, long ago (2019)



EfficientNet-B5

Core ML Tools

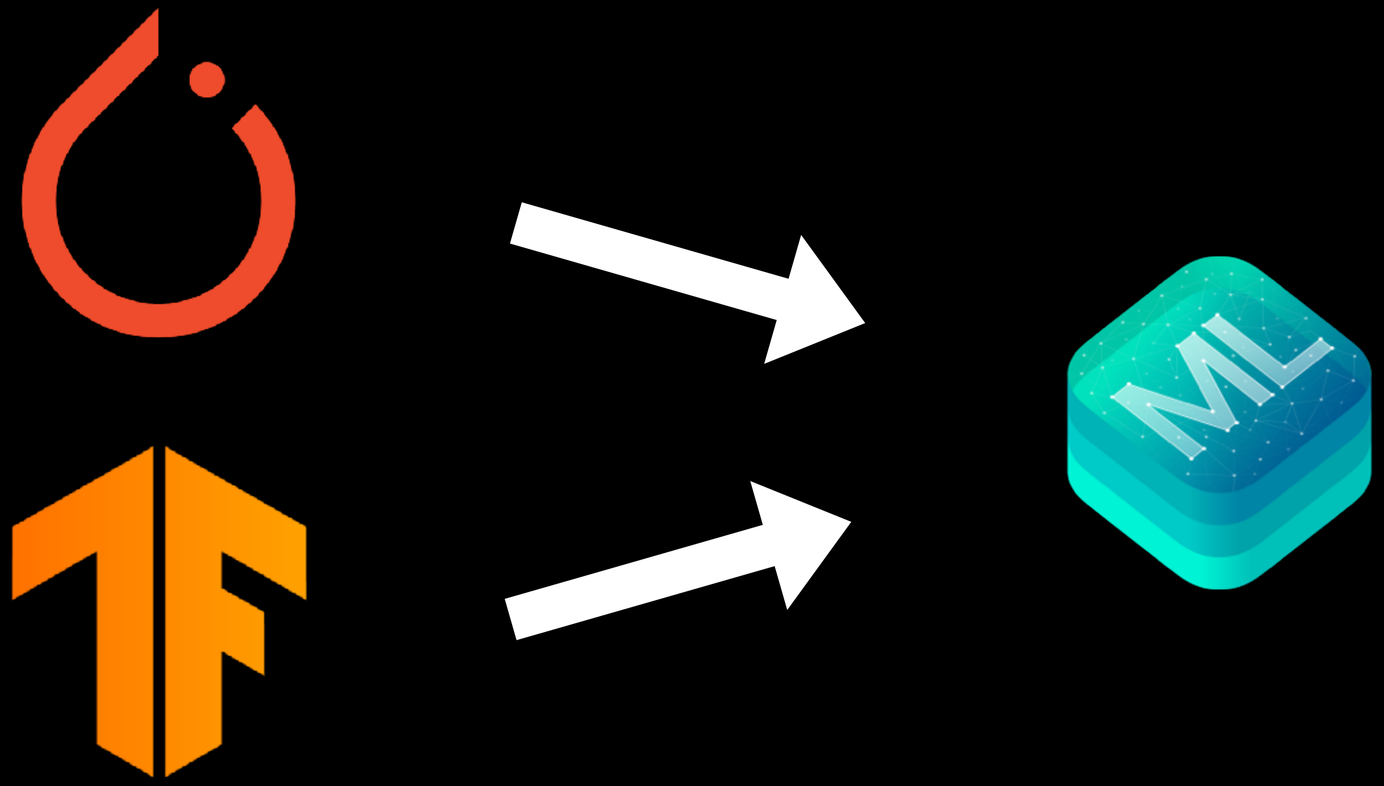
Today



EfficientNet-B5

Core ML Tools

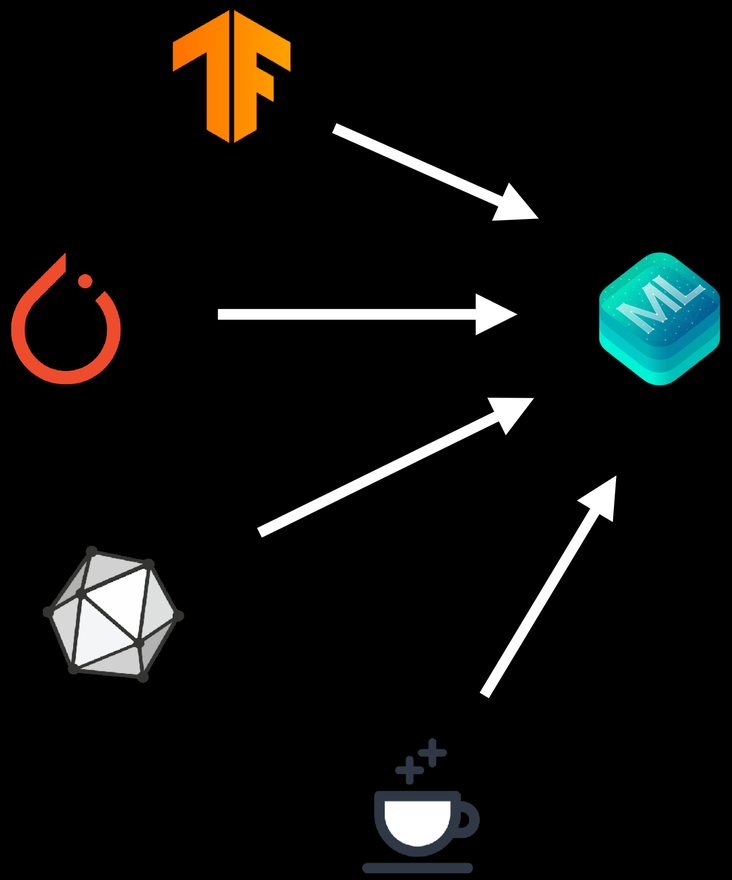
Today



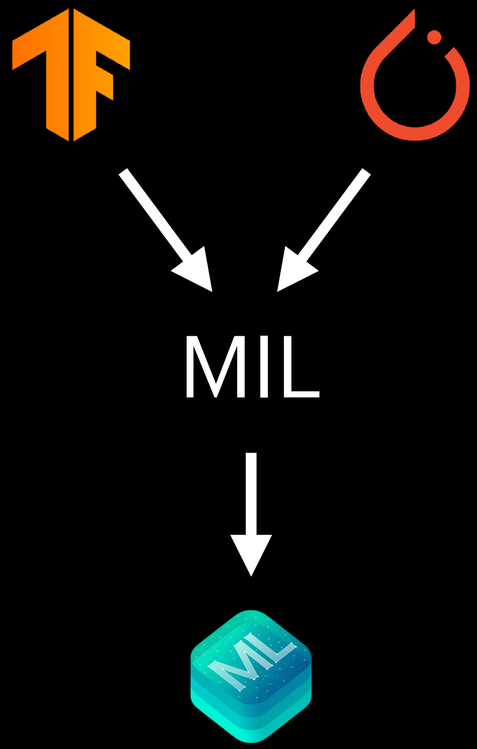
EfficientNet-B5

What changed?

2019



2021

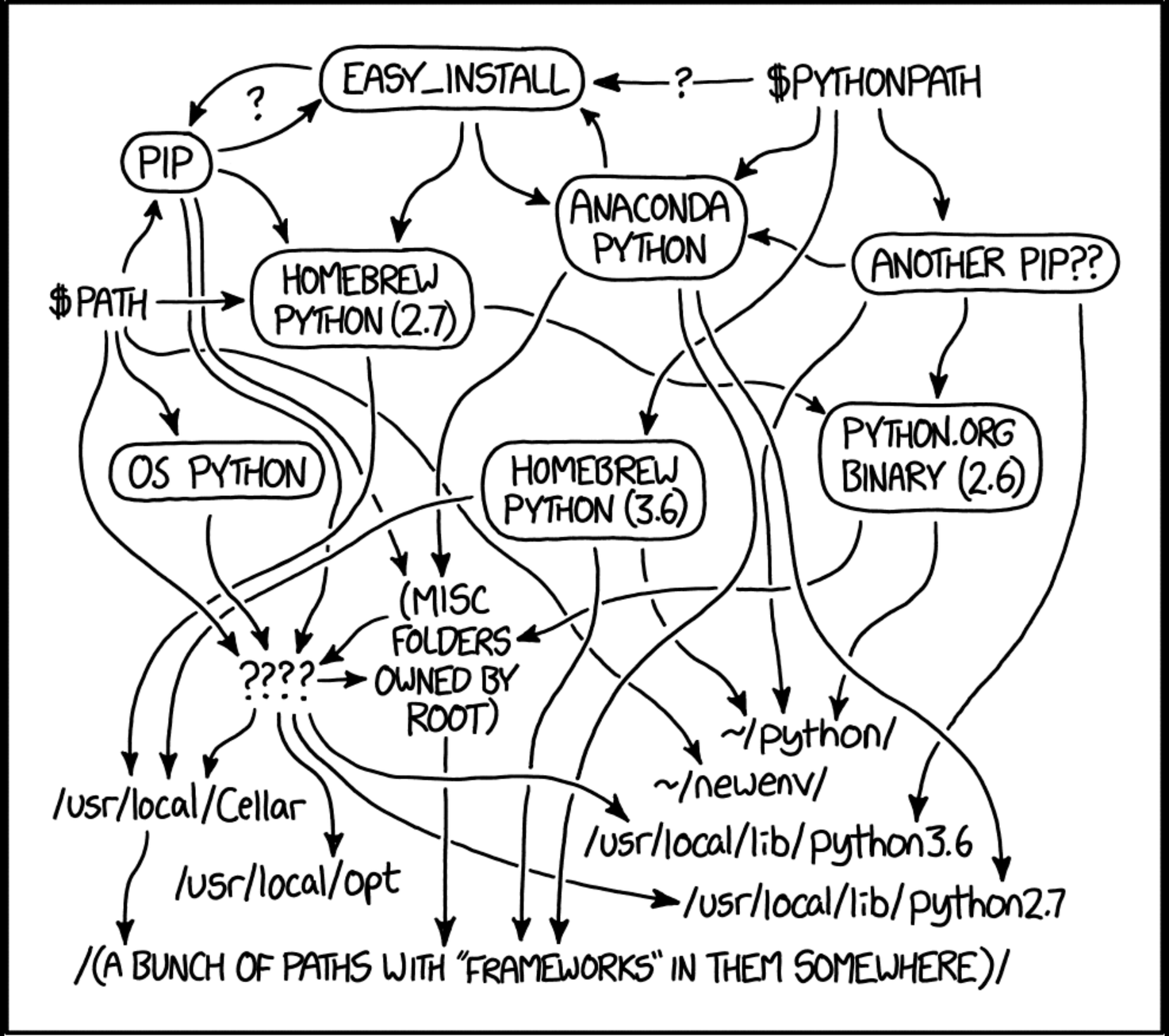


New-ish Layer Used in AnimeGANv2

Group Norm

Implemented in MIL ops

Python Environment



MY PYTHON ENVIRONMENT HAS BECOME SO DEGRADED THAT MY LAPTOP HAS BEEN DECLARED A SUPERFUND SITE.

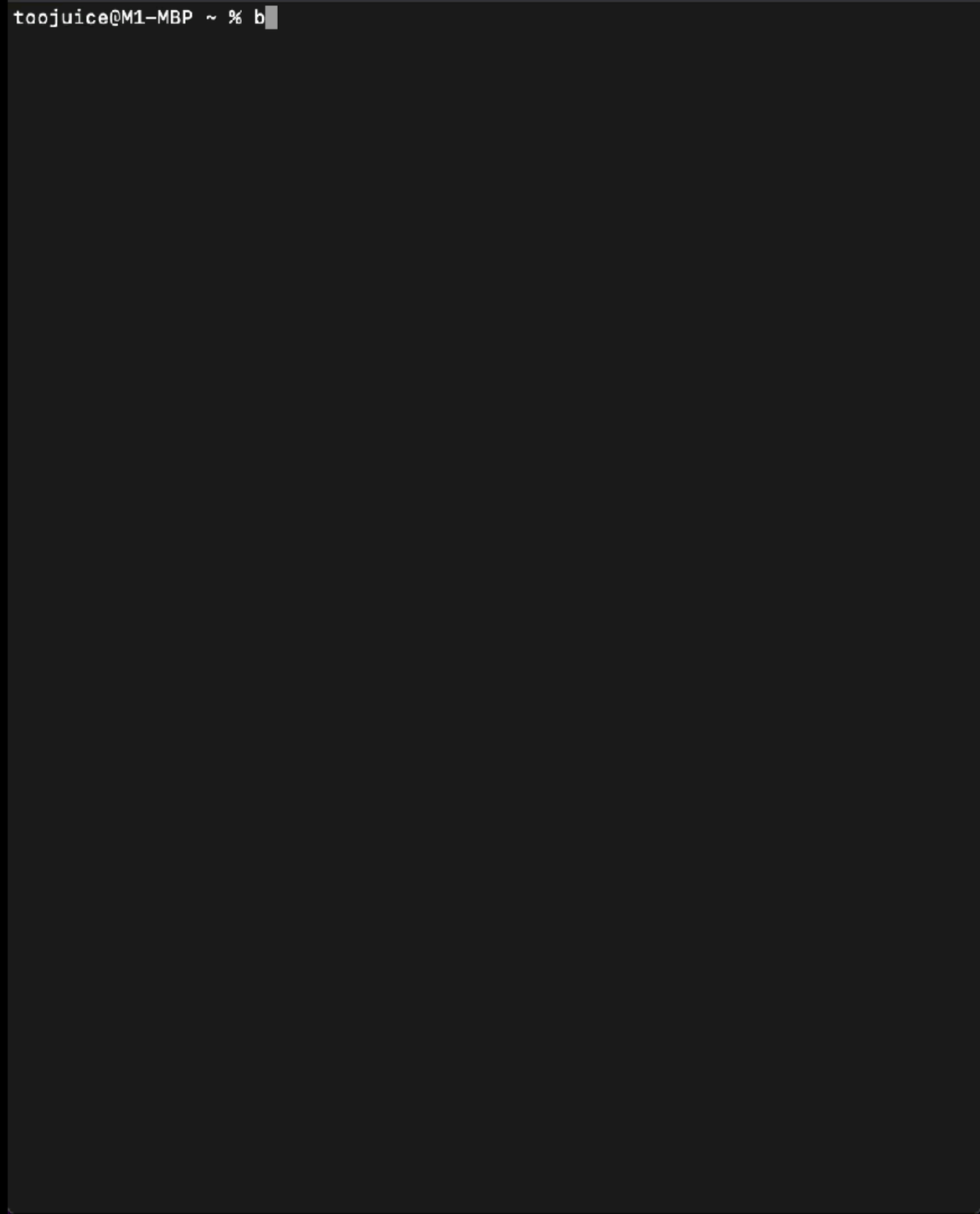
xkcd.com/1987/

The Python environmental protection agency wants to seal it in a cement chamber, with pictorial messages to future civilizations warning them about the danger of using sudo to install random Python packages.

Conda

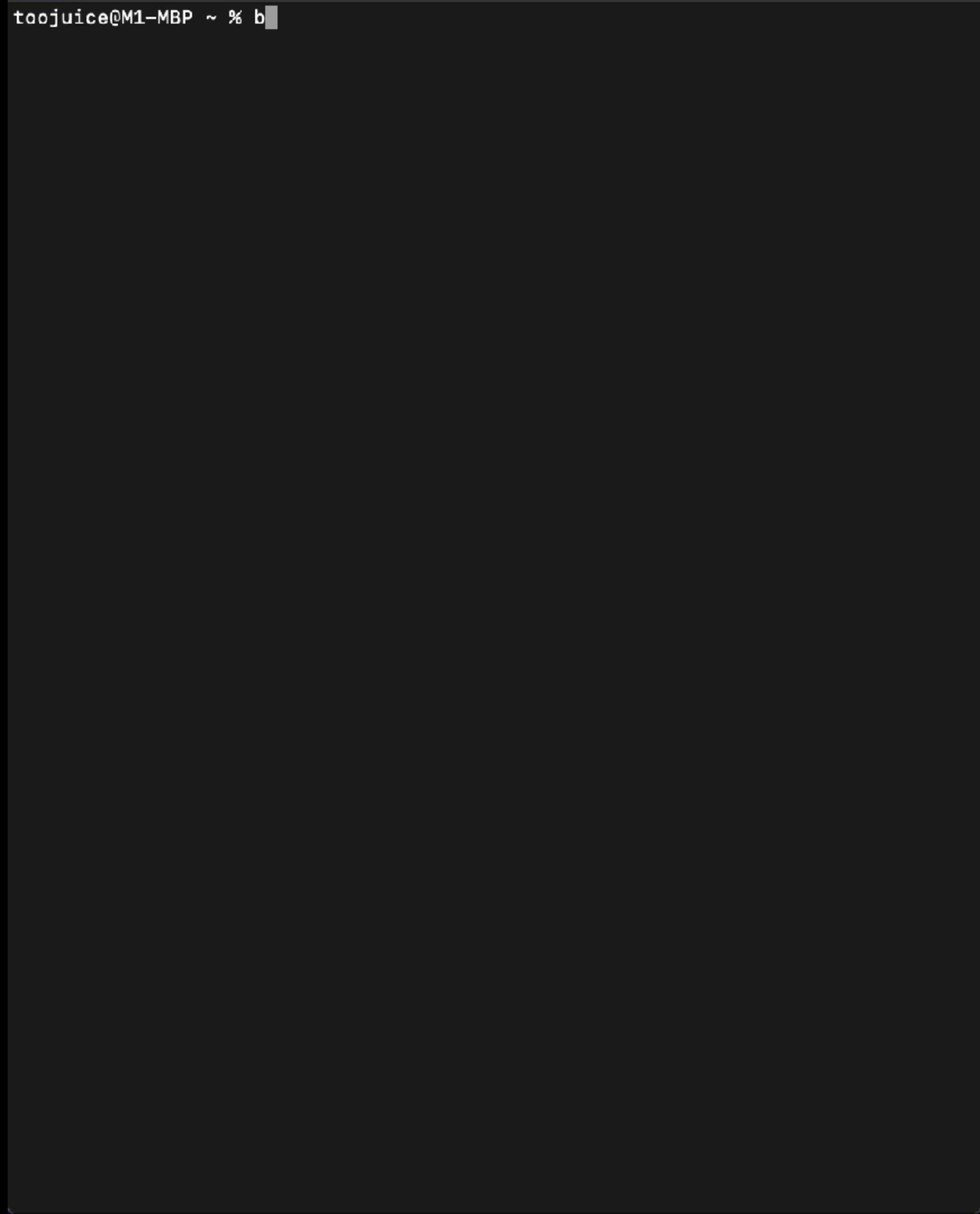
Conda Setup

```
> brew install miniforge
```



Conda Setup

```
> brew install miniforge
> conda init zsh
```

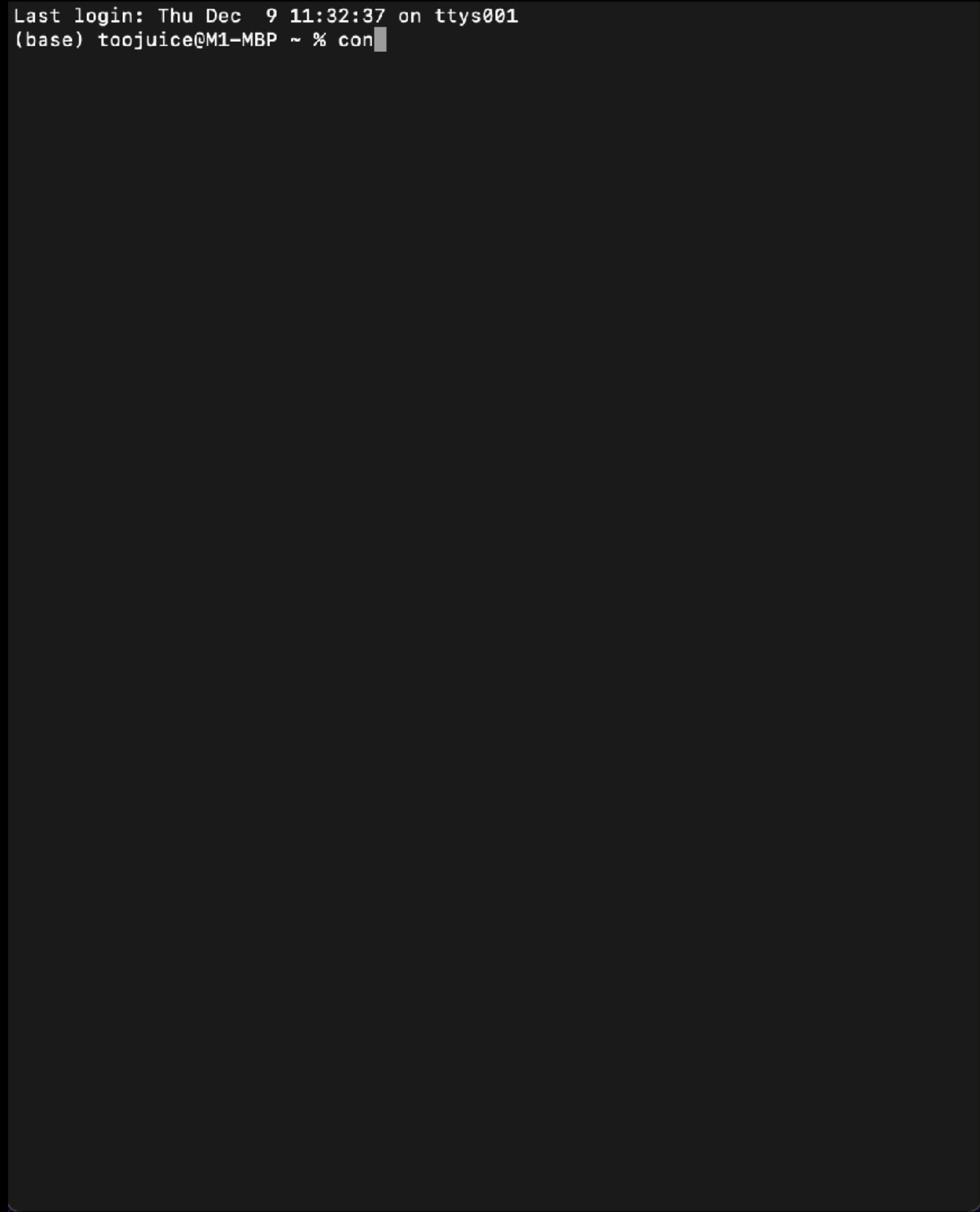


Conda Setup

restart shell or Terminal.app

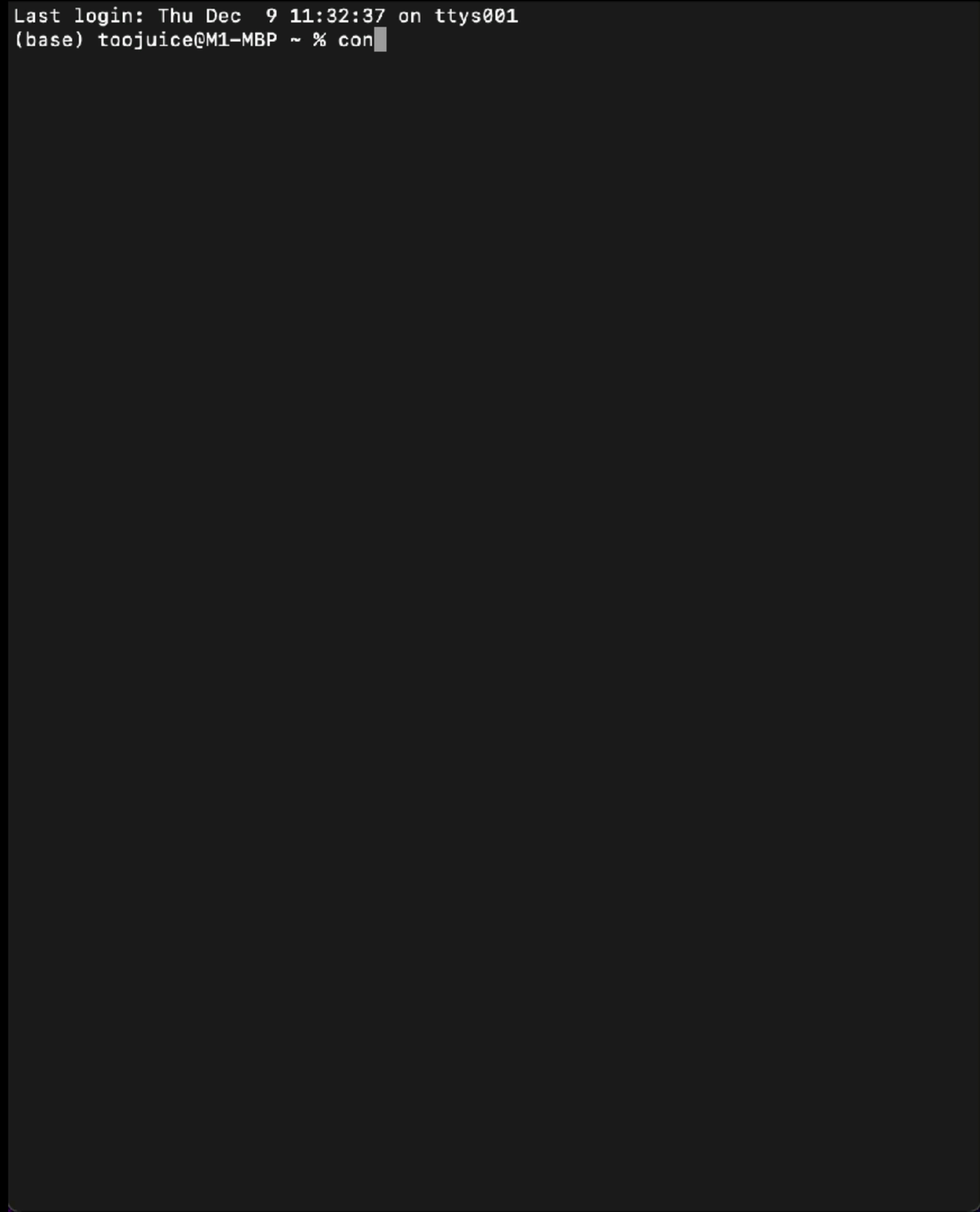
Conda Env Setup

```
> conda create --name coreml python=3.9
```



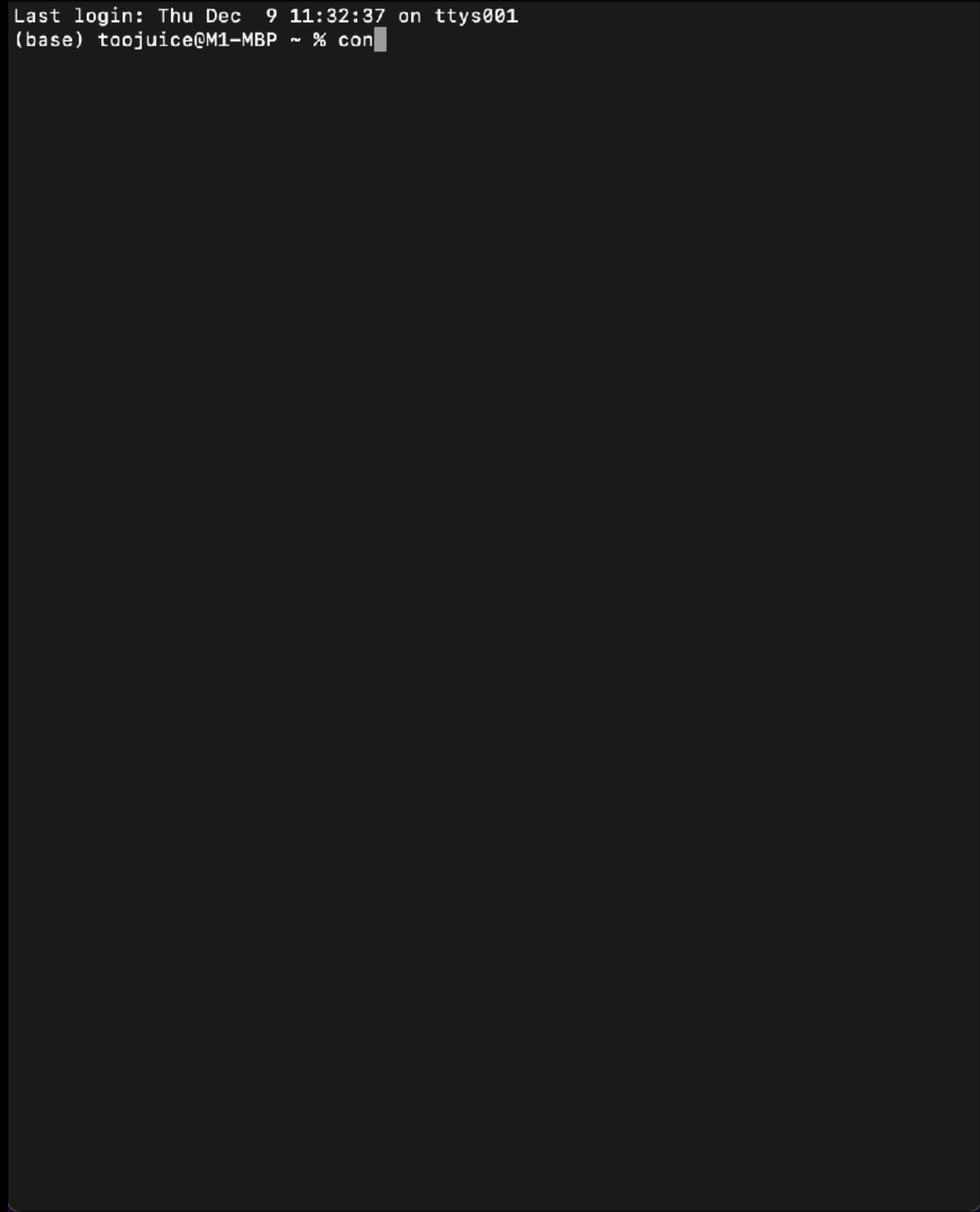
Conda Env Setup

```
> conda create --name coreml python=3.9
> conda activate coreml
```



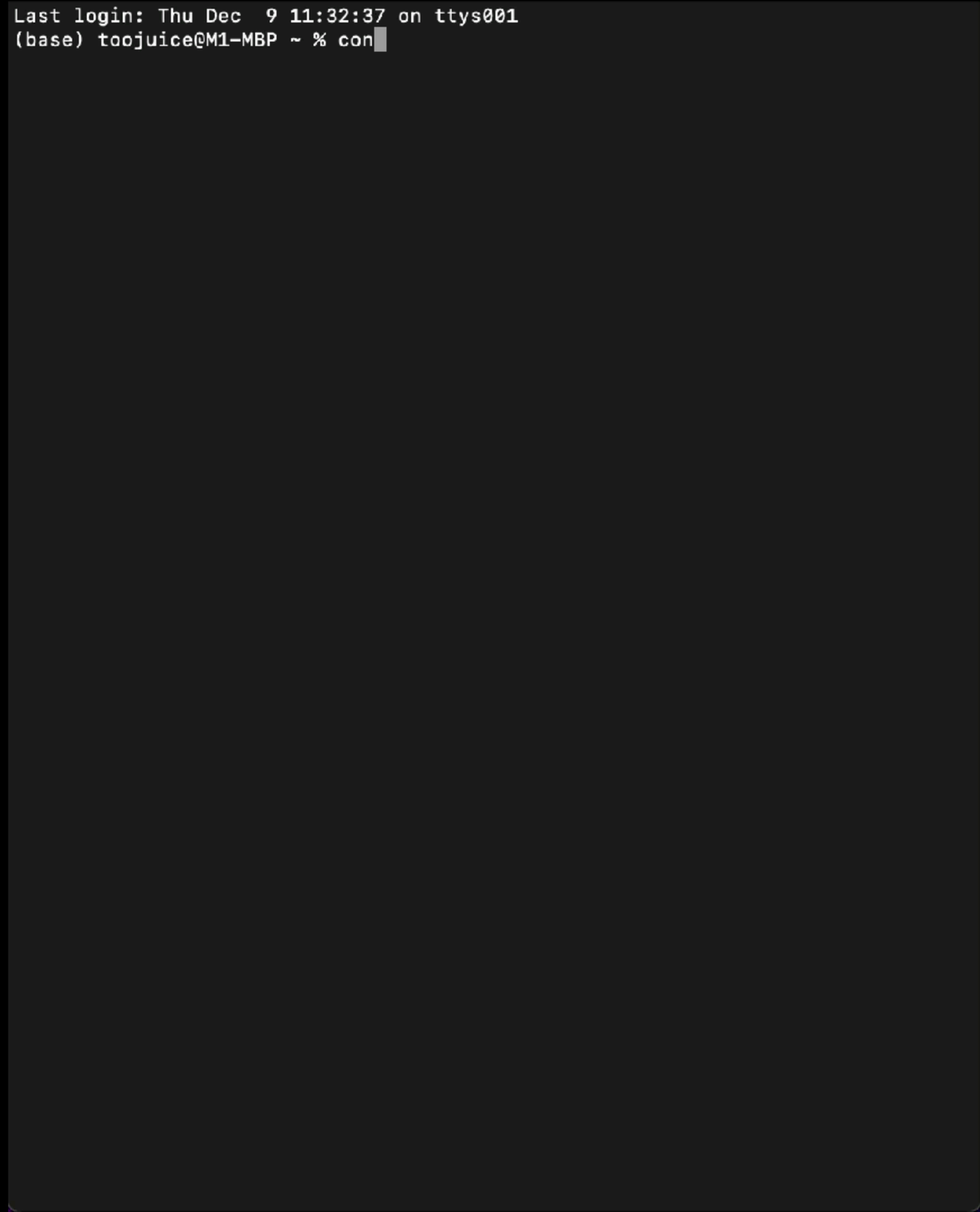
Conda Env Setup

```
> conda create --name coreml python=3.9
> conda activate coreml
> conda install pytorch torchvision
```



Conda Env Setup

```
> conda create --name coreml python=3.9
> conda activate coreml
> conda install pytorch torchvision
> pip install coremltools
```



Conda Env Setup

```
> conda create --name coreml python=3.9
> conda activate coreml
> conda install pytorch torchvision
> pip install coremltools
> conda install jupyterlab
```

* optional but cool!

Conda Deactivate

When you’re done using the environment

```
> conda deactivate
```

Other Handy Conda Commands

```
> conda list
```

Other Handy Conda Commands

```
> conda update <package>
```

Other Handy Conda Commands

```
> conda remove <package(s)>
```

Other Handy Conda Commands

```
> conda env list
```

Other Handy Conda Commands

```
> conda env remove --name <env_name>
```

Conda Cheat Sheet

https://docs.conda.io/projects/conda/en/4.6.0/_downloads/52a95608c49671267e40c689e0bc00ca/conda-cheatsheet.pdf

Check Python Path and Version

>

Check Python Path and Version

```
> which python
```

Check Python Path and Version

```
> which python
/opt/homebrew/Caskroom/miniforge/base/envs/coreml/bin/python
>
```

Check Python Path and Version

```
> which python
/opt/homebrew/Caskroom/miniforge/base/envs/coreml/bin/python
> python --version
```

Check Python Path and Version

```
> which python
/opt/homebrew/Caskroom/miniforge/base/envs/coreml/bin/python
> python --version
Python 3.9.7
```

Check Library Versions

```
>>> import torch
```

Check Library Versions

```
>>> import torch
>>> torch.__version__
```

Check Library Versions

```
>>> import torch
>>> torch.__version__
‘1.10.0’
>>> import coremltools
```

Check Library Versions

```
>>> import torch
>>> torch.__version__
‘1.10.0’
>>> import coremltools
>>> coremltools.__version__
```

Check Library Versions

```
>>> import torch
>>> torch.__version__
‘1.10.0’
>>> import coremltools
>>> coremltools.__version__
‘5.1.0’
```

Conversion Script

https://github.com/yonomitt/Anime-fy-Yourself

```
import torch
import coremltools as ct
```

```
Python 3.9.7 | packaged by conda-forge | (default, Sep 29 2021, 19:24:02)
[Clang 11.1.0 ] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

```
Python 3.9.7 | packaged by conda-forge | (default, Sep 29 2021, 19:24:02)
[Clang 11.1.0 ] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> import torch
```

```
Python 3.9.7 | packaged by conda-forge | (default, Sep 29 2021, 19:24:02)
[Clang 11.1.0 ] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> import torch
>>> import coremltools as ct
```

```
Python 3.9.7 | packaged by conda-forge | (default, Sep 29 2021, 19:24:02)
[Clang 11.1.0 ] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> import torch
>>> import coremltools as ct
WARNING:root:Torch version 1.10.0 has not been tested with coremltools. You may run into
unexpected errors. Torch 1.9.1 is the most recent version that has been tested.
>>>
```

```
import torch
import coremltools as ct
```

```
import torch
import coremltools as ct
import sys
```

```
import torch
import coremltools as ct
import sys

# check for command line arguments
if len(sys.argv) < 2:
    print(f'USAGE: {sys.argv[0]} <image size>’)
    sys.exit(-1)
```

```
import torch
import coremltools as ct
import sys

# check for command line arguments
if len(sys.argv) < 2:
    print(f'USAGE: {sys.argv[0]} <image size>’)
    sys.exit(-1)

# store the desired image size
SIZE = int(sys.argv[1])
```

```
# load the AnimeGAN v2 model using the pretrained Face Portrait v2 weights
model = torch.hub.load("bryandlee/animegan2-pytorch:main", “generator",
                       pretrained=“face_paint_512_v2")
```

```
# load the AnimeGAN v2 model using the pretrained Face Portrait v2 weights
model = torch.hub.load("bryandlee/animegan2-pytorch:main", “generator",
                       pretrained=“face_paint_512_v2")
model.eval()
```

```
# load the AnimeGAN v2 model using the pretrained Face Portrait v2 weights
model = torch.hub.load("bryandlee/animegan2-pytorch:main", “generator",
                       pretrained=“face_paint_512_v2")
model.eval()

# create a dummy input for the model trace
# dimensions are (batch size, channels (RGB), width, height)
dummy_in = torch.rand(1, 3, SIZE, SIZE)
```

```
# load the AnimeGAN v2 model using the pretrained Face Portrait v2 weights
model = torch.hub.load("bryandlee/animegan2-pytorch:main", “generator",
                       pretrained=“face_paint_512_v2")
model.eval()

# create a dummy input for the model trace
# dimensions are (batch size, channels (RGB), width, height)
dummy_in = torch.rand(1, 3, SIZE, SIZE)

# run the model's forward method and record the tensor operations
traced_model = torch.jit.trace(model, dummy_in)
```

```
# load the AnimeGAN v2 model using the pretrained Face Portrait v2 weights
model = torch.hub.load("bryandlee/animegan2-pytorch:main", “generator",
                       pretrained=“face_paint_512_v2")
model.eval()

# create a dummy input for the model trace
# dimensions are (batch size, channels (RGB), width, height)
dummy_in = torch.rand(1, 3, SIZE, SIZE)

# run the model's forward method and record the tensor operations
traced_model = torch.jit.trace(model, dummy_in)

# save the traced model (TorchScript) to a file
traced_model.save("animegan_v2.pt")
```

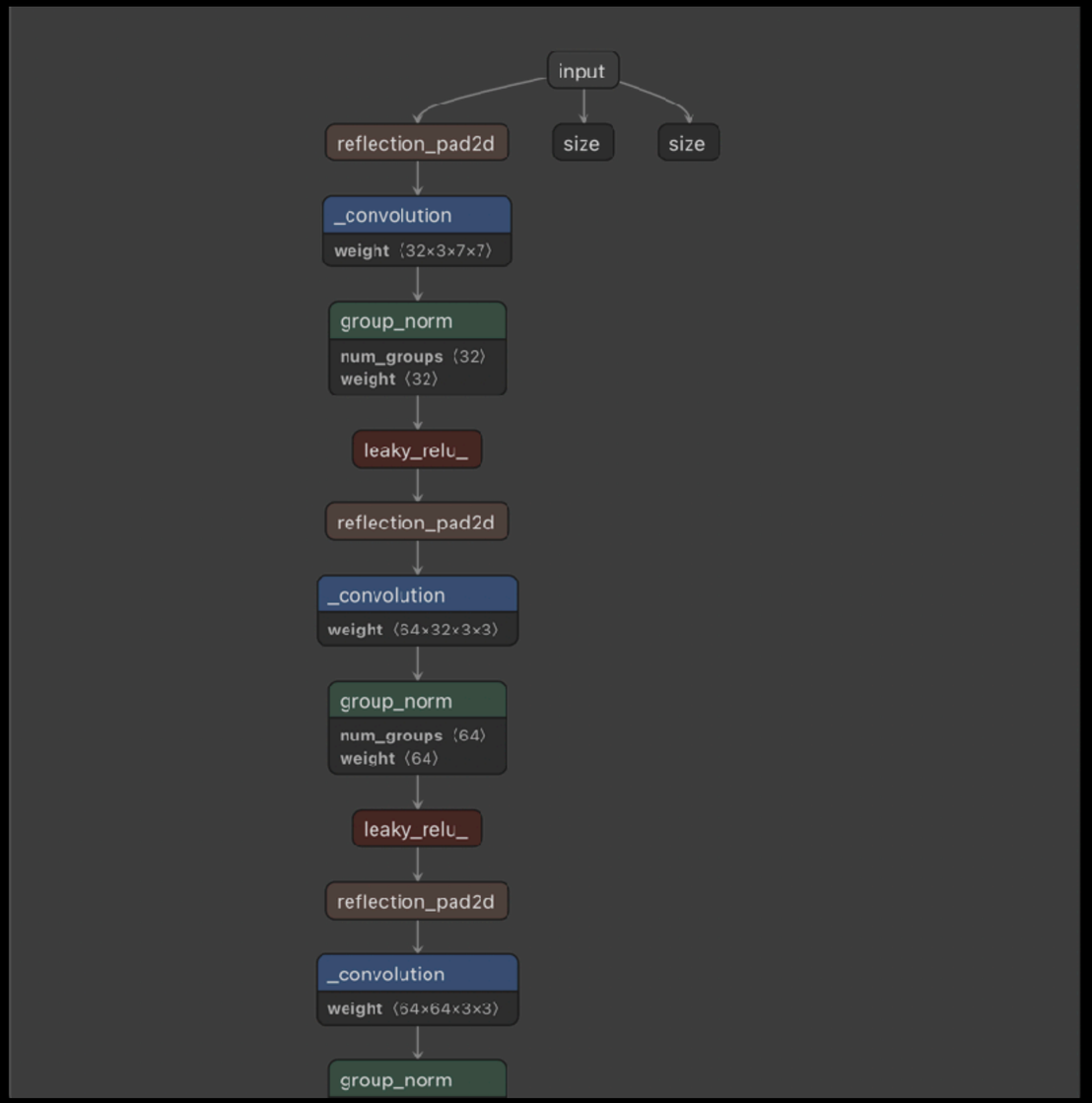
```
# define the CoreML inputs
input = ct.TensorType(name="input", shape=dummy_in.shape)
```

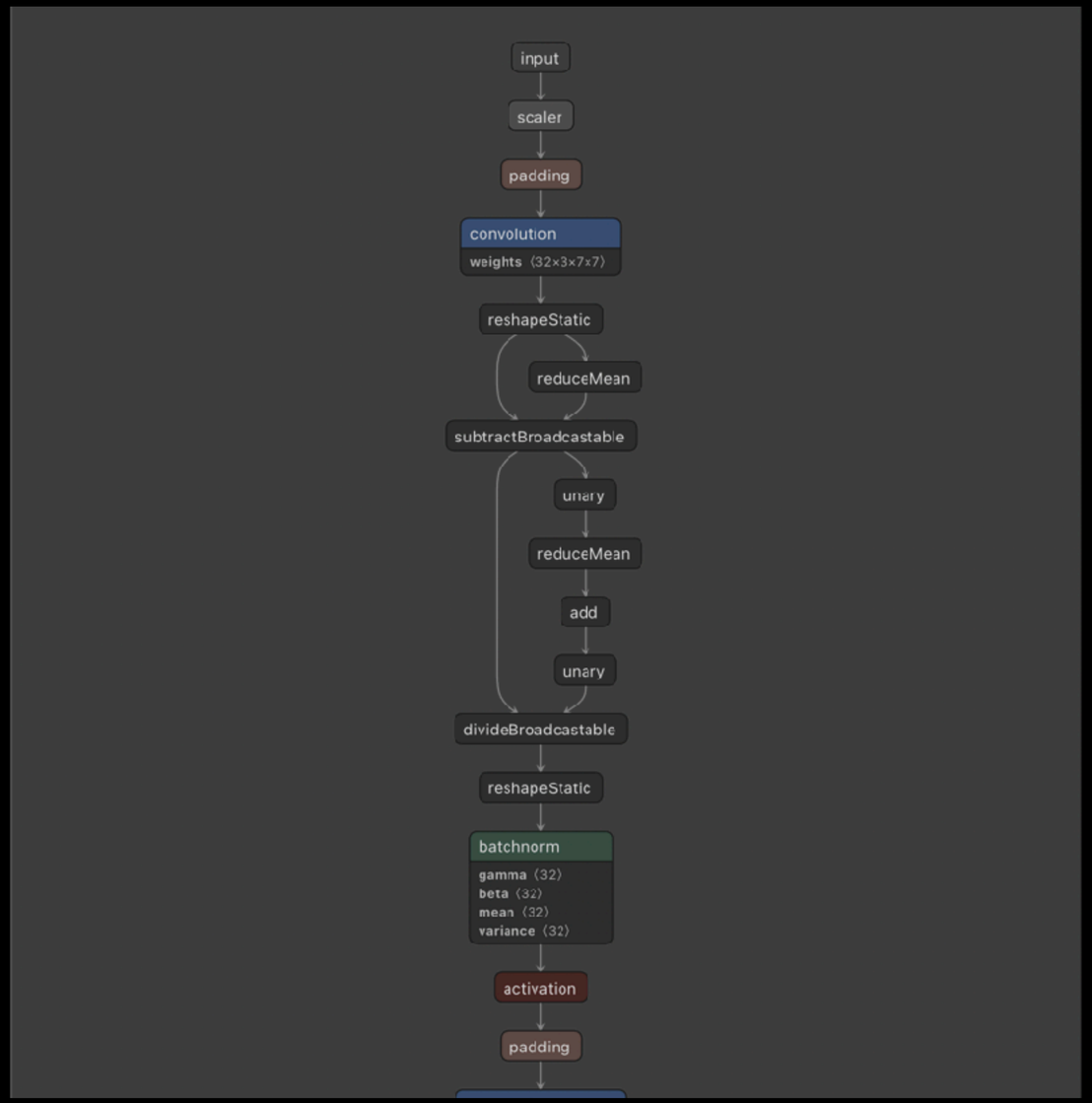
```
# define the CoreML inputs
input = ct.TensorType(name="input", shape=dummy_in.shape)

# convert the TorchScript to an MLModel
mlmodel = ct.convert("animegan_v2.pt", inputs=[input])
```

```
# define the CoreML inputs
input = ct.TensorType(name="input", shape=dummy_in.shape)

# convert the TorchScript to an MLModel
mlmodel = ct.convert("animegan_v2.pt", inputs=[input])

# save the model
mlmodel.save(f"AnimeGANv2_{SIZE}.mlmodel")
```

THIS SLIDE INTENTIONALLY LEFT BLANK

Netron


input
reflection_pad2d
size
size
_convolution
weight ⟨32×3×7×7⟩
group_norm
num_groups ⟨32⟩
weight ⟨32⟩
leaky_relu_
reflection_pad2d
_convolution
weight ⟨64×32×3×3⟩
group_norm
num_groups ⟨64⟩
weight ⟨64⟩
leaky_relu_
reflection_pad2d
_convolution
weight ⟨64×64×3×3⟩
group_norm

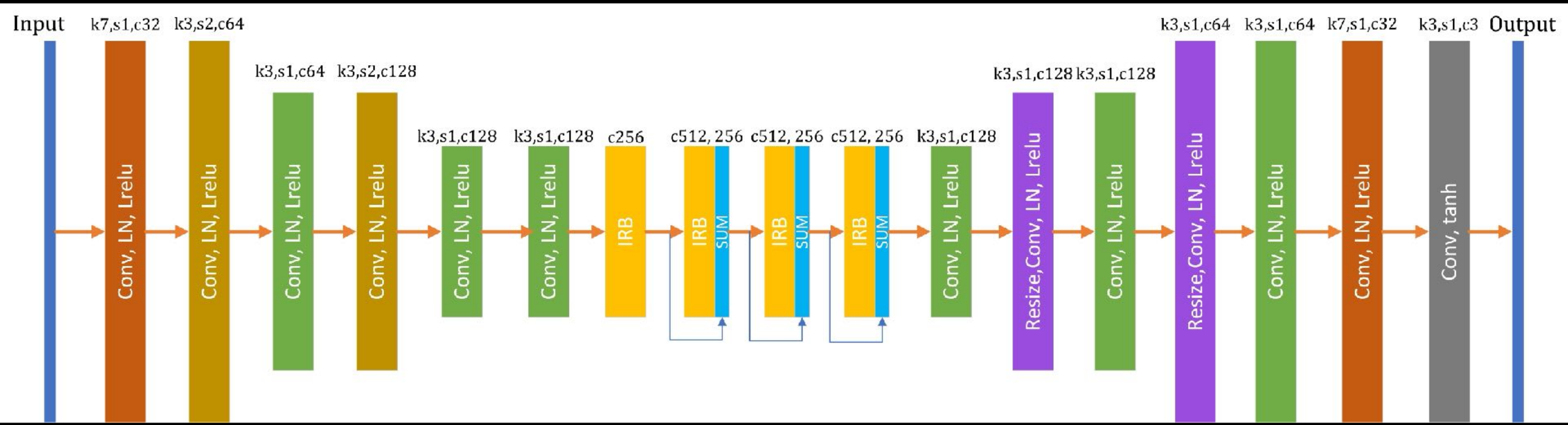

input
scaler
padding
convolution
weights ⟨32×3×7×7⟩
reshapeStatic
reduceMean
subtractBroadcastable
unary
reduceMean
add
unary
divideBroadcastable
reshapeStatic
batchnorm
gamma ⟨32⟩
beta ⟨32⟩
mean ⟨32⟩
variance ⟨32⟩
activation
padding

Trace vs Script

Trace uses a sample input and traces which operations are performed on the data

Script will follow all branches of control and conditional operations

AnimeGAN v2


Input
k7,s1,c32
k3,s2,c64
k3,s1,c64
k3,s2,c128
k3,s1,c128
k3,s1,c128
c256
c512, 256
c512, 256
c512, 256
k3,s1,c128
k3,s1,c128
k3,s1,c128
k3,s1,c64
k3,s1,c64
k7,s1,c32
k3,s1,c3
Output
Conv, LN, Lrelu
IRB
SUM
Resize,Conv, LN, Lrelu
Conv, tanh

Fixing the Input Type

```
# define the CoreML inputs
input = ct.TensorType(name="input", shape=dummy_in.shape)

# convert the TorchScript to an MLModel
mlmodel = ct.convert("animegan_v2.pt", inputs=[input])

# save the model
mlmodel.save(f"AnimeGANv2_{SIZE}.mlmodel")
```

```
# define the CoreML inputs
input = ct.TensorType(name="input", shape=dummy_in.shape)



# convert the TorchScript to an MLModel
mlmodel = ct.convert("animegan_v2.pt", inputs=[input])

# save the model
mlmodel.save(f"AnimeGANv2_{SIZE}.mlmodel")
```

```
# define the CoreML inputs
input = ct.ImageType(name="input", shape=dummy_in.shape, scale=1./127.5,
                     bias=[-1.0, -1.0, -1.0], color_layout='RGB', channel_first=True)

# convert the TorchScript to an MLModel
mlmodel = ct.convert("animegan_v2.pt", inputs=[input])

# save the model
mlmodel.save(f”AnimeGANv2_{SIZE}.mlmodel")
```

THIS SLIDE INTENTIONALLY LEFT BLANK

Can we do better?

Model Editing

Changing the Output Type

```
import coremltools.proto.FeatureTypes_pb2 as ft

from coremltools import utils
```

```
import coremltools.proto.FeatureTypes_pb2 as ft
from coremltools import utils

# get a deep copy of the model's protobuf specification
spec = mlmodel.get_spec()
```

```
import coremltools.proto.FeatureTypes_pb2 as ft
from coremltools import utils

# get a deep copy of the model's protobuf specification
spec = mlmodel.get_spec()

# make the output an image type
output = spec.description.output[0]
```

```
import coremltools.proto.FeatureTypes_pb2 as ft
from coremltools import utils

# get a deep copy of the model's protobuf specification
spec = mlmodel.get_spec()

# make the output an image type
output = spec.description.output[0]
output.type.imageType.colorSpace = ft.ImageFeatureType.RGB
```

```
import coremltools.proto.FeatureTypes_pb2 as ft
from coremltools import utils

# get a deep copy of the model's protobuf specification
spec = mlmodel.get_spec()

# make the output an image type
output = spec.description.output[0]
output.type.imageType.colorSpace = ft.ImageFeatureType.RGB
output.type.imageType.width = SIZE
```

```
import coremltools.proto.FeatureTypes_pb2 as ft
from coremltools import utils

# get a deep copy of the model's protobuf specification
spec = mlmodel.get_spec()

# make the output an image type
output = spec.description.output[0]
output.type.imageType.colorSpace = ft.ImageFeatureType.RGB
output.type.imageType.width = SIZE
output.type.imageType.height = SIZE
```

```
import coremltools.proto.FeatureTypes_pb2 as ft
from coremltools import utils

# get a deep copy of the model's protobuf specification
spec = mlmodel.get_spec()

# make the output an image type
output = spec.description.output[0]
output.type.imageType.colorSpace = ft.ImageFeatureType.RGB
output.type.imageType.width = SIZE
output.type.imageType.height = SIZE

# save the CoreML model
utils.save_spec(spec, f"AnimeGANv2_{SIZE}.mlmodel")
```

THIS SLIDE INTENTIONALLY LEFT BLANK

Scaling the Output

```
# get the neural network
nn = get_nn(spec)
```

```
# get the neural network
nn = get_nn(spec)


# add a new layer to the end of the neural network
unscaled_output_layer = nn.layers[-1]
```

```
# get the neural network
nn = get_nn(spec)

# add a new layer to the end of the neural network
unscaled_output_layer = nn.layers[-1]
scale_layer = nn.layers.add()
```

```
# get the neural network
nn = get_nn(spec)

# add a new layer to the end of the neural network
unscaled_output_layer = nn.layers[-1]
scale_layer = nn.layers.add()

# set it's name
scale_layer.name = "scale_output"
```

```
# make the new layer a linear activation function,
# which scales the output from [-1, 1] to [0, 255]
# y = alpha * x + beta
scale_layer.activation.linear.alpha = 127.5
scale_layer.activation.linear.beta = 127.5
```

```
# make the new layer a linear activation function,
# which scales the output from [-1, 1] to [0, 255]
# y = alpha * x + beta
scale_layer.activation.linear.alpha = 127.5
scale_layer.activation.linear.beta = 127.5

# connect the new layer to the output
scale_layer.output.append(unscaled_output_layer.output[0])
```

```
# make the new layer a linear activation function,
# which scales the output from [-1, 1] to [0, 255]
# y = alpha * x + beta
scale_layer.activation.linear.alpha = 127.5
scale_layer.activation.linear.beta = 127.5

# connect the new layer to the output
scale_layer.output.append(unscaled_output_layer.output[0])

# connect the old last layer's output to the new scale layer's input
unscaled_output_layer.output[0] = "scale_layer_input"
scale_layer.input.append(unscaled_output_layer.output[0])
```

```
# save the CoreML model
utils.save_spec(spec, f"AnimeGANv2_{SIZE}.mlmodel")
```

THIS SLIDE INTENTIONALLY LEFT BLANK

Renaming the Output

```
# rename the output

output.name = "output"
```

```
# rename the output
output.name = "output"
output_layer = nn.layers[-1]
```

```
# rename the output
output.name = "output"
output_layer = nn.layers[-1]
output_layer.output[0] = output.name
```

```
# rename the output
output.name = "output"
output_layer = nn.layers[-1]
output_layer.output[0] = output.name

# save the CoreML model
utils.save_spec(spec, f"AnimeGANv2_{SIZE}.mlmodel")
```

THIS SLIDE INTENTIONALLY LEFT BLANK

Quantizing the Parameters

```
from coremltools.models.neural_network import quantization_utils
```

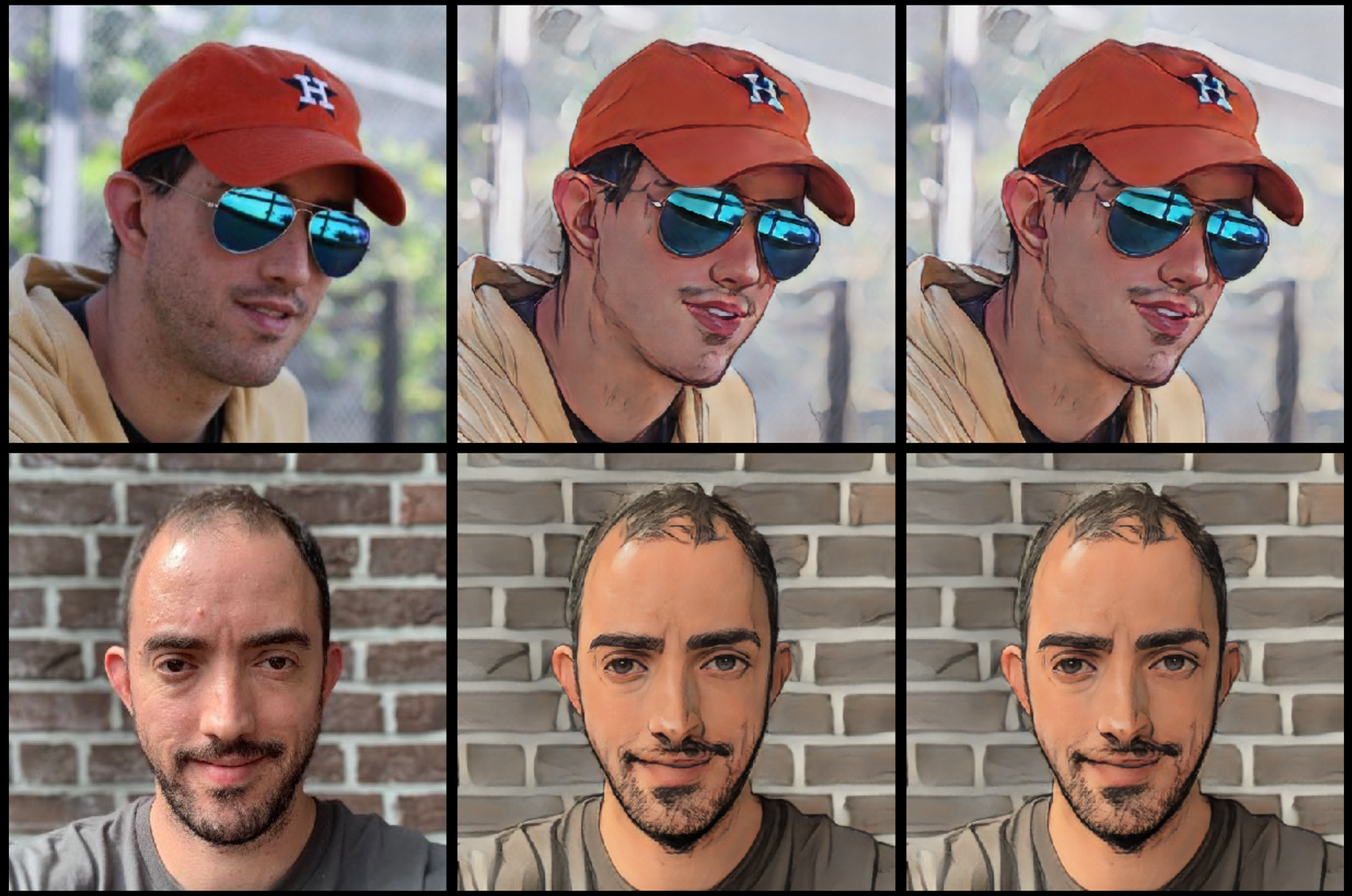
```
from coremltools.models.neural_network import quantization_utils


...
```

```
from coremltools.models.neural_network import quantization_utils


...


# convert the TorchScript to an MLModel
mlmodel = ct.convert("animegan_v2.pt", inputs=[input])
```

```
from coremltools.models.neural_network import quantization_utils


...


# convert the TorchScript to an MLModel
mlmodel = ct.convert("animegan_v2.pt", inputs=[input])


# convert the weights from fp32 -> fp16
mlmodel = quantization_utils.quantize_weights(mlmodel, nbits=16)
```

THIS SLIDE INTENTIONALLY LEFT BLANK

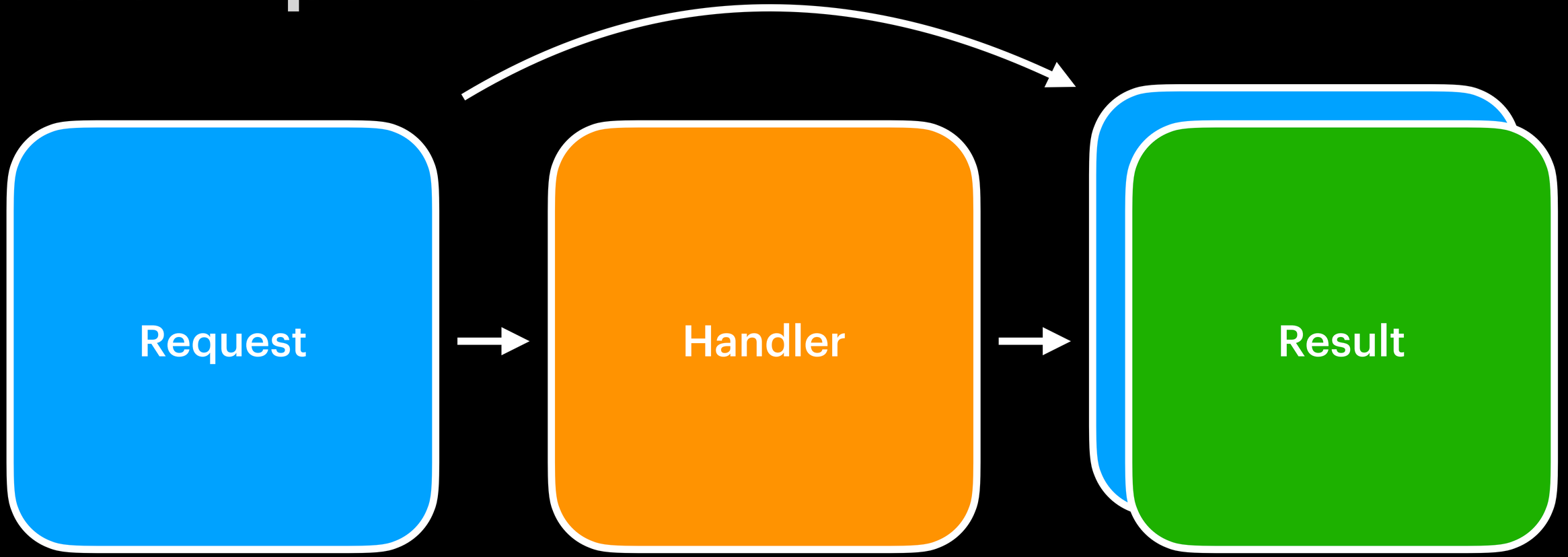
Comparing Outputs



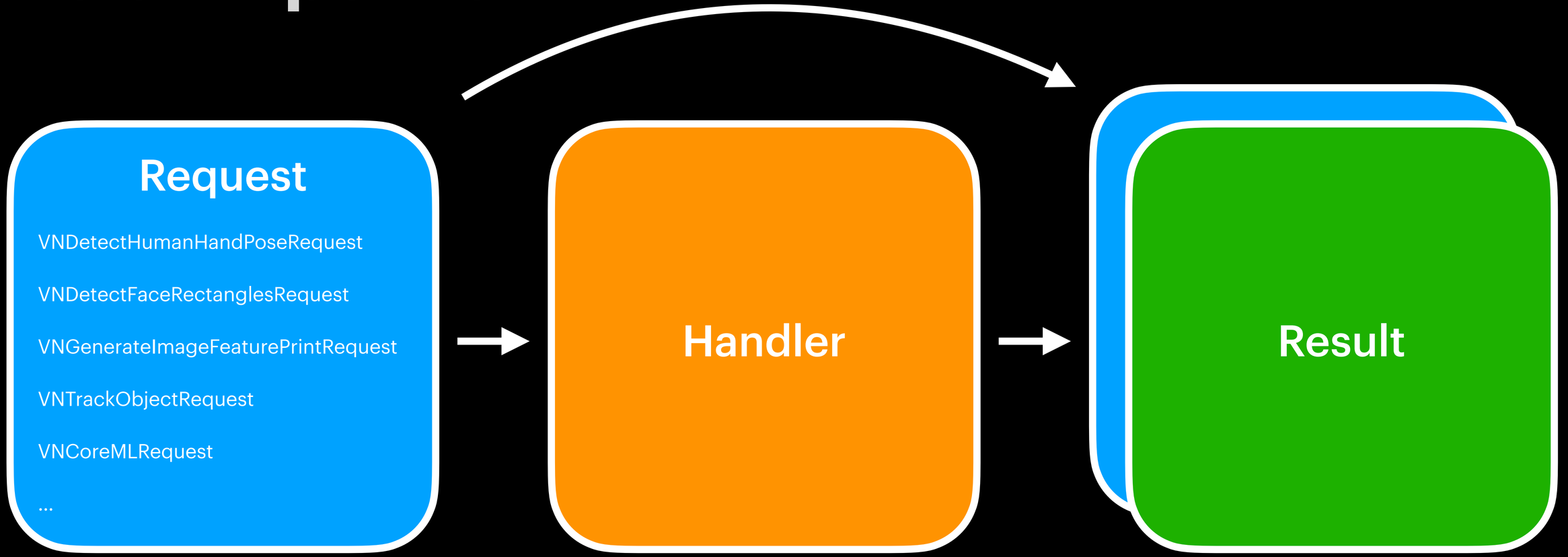
iOS Integration

Vision Pipeline

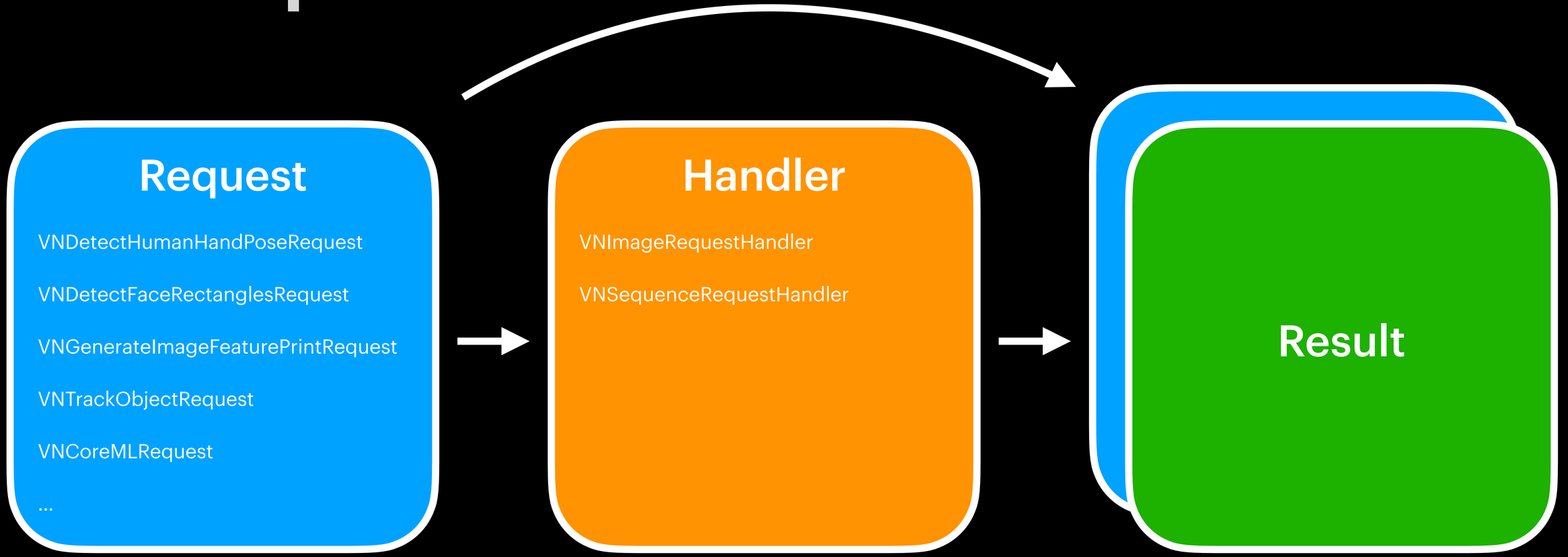
Vision Pipeline



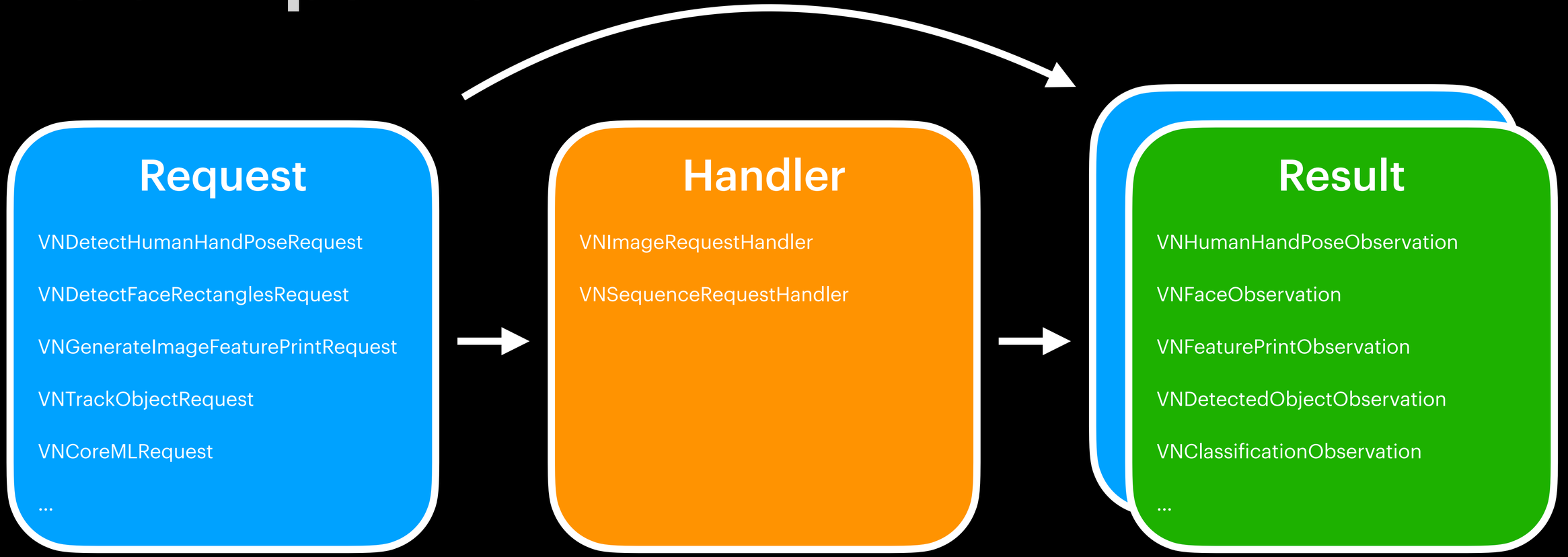
Vision Pipeline



Vision Pipeline



Vision Pipeline



```
private let request: VNRequest = {









}()
```

```
private let request: VNRequest = {
    let configuration = MLModelConfiguration()









}()
```

```
private let request: VNRequest = {
    let configuration = MLModelConfiguration()
    configuration.computeUnits = .all
    configuration.allowLowPrecisionAccumulationOnGPU = true




}()
```

```
private let request: VNRequest = {
    let configuration = MLModelConfiguration()
    configuration.computeUnits = .all
    configuration.allowLowPrecisionAccumulationOnGPU = true

    let model = try! AnimeGANv2_512(configuration: configuration).model




}()
```

```
private let request: VNRequest = {
    let configuration = MLModelConfiguration()
    configuration.computeUnits = .all
    configuration.allowLowPrecisionAccumulationOnGPU = true

    let model = try! AnimeGANv2_512(configuration: configuration).model

    let req = VNCoreMLRequest(model: try! VNCoreMLModel(for: model))



}()
```

```
private let request: VNRequest = {
    let configuration = MLModelConfiguration()
    configuration.computeUnits = .all
    configuration.allowLowPrecisionAccumulationOnGPU = true

    let model = try! AnimeGANv2_512(configuration: configuration).model

    let req = VNCoreMLRequest(model: try! VNCoreMLModel(for: model))
    req.imageCropAndScaleOption = .centerCrop


}()
```

```
private let request: VNRequest = {
    let configuration = MLModelConfiguration()
    configuration.computeUnits = .all
    configuration.allowLowPrecisionAccumulationOnGPU = true

    let model = try! AnimeGANv2_512(configuration: configuration).model

    let req = VNCoreMLRequest(model: try! VNCoreMLModel(for: model))
    req.imageCropAndScaleOption = .centerCrop

    return req
}()
```

```
let handler = VNImageRequestHandler(cvPixelBuffer: image, options: [:])
```

```
let handler = VNImageRequestHandler(cvPixelBuffer: image, options: [:])


let animeRequest = try! handler.perform([request])
```

```
let handler = VNImageRequestHandler(cvPixelBuffer: image, options: [:])

let animeRequest = try! handler.perform([request])

guard let results = animeRequest.results as? [VNPixelBufferObservation],
      let result = results.first else {
          return nil
}
```

```
let handler = VNImageRequestHandler(cvPixelBuffer: image, options: [:])

let animeRequest = try! handler.perform([request])

guard let results = animeRequest.results as? [VNPixelBufferObservation],
      let result = results.first else {
          return nil
}

return CIImage(cvPixelBuffer: result.pixelBuffer)
```

THIS SLIDE INTENTIONALLY LEFT BLANK

https://github.com/yonomitt/Anime-fy-Yourself

Thank You

@yonomitt

https://github.com/yonomitt/Anime-fy-Yourself