Machine Learning on Apple Devices 📱

~

Who: Jarrod Parkes
What: DevSpace 2017
When: Oct 14th, 2017 @ 9:30 AM CST
Where: Von Braun Center, Huntsville, AL, USA 🇺🇸
Why: Swift + CoreML 🚀!

~

Credits: *Meghan Kane*

Hey, I'm Jarrod 👋

- I build iOS/Swift courses at Udacity

- I'm not a ML expert

- BUT, you don't have to be to use Core ML 🎉!

Talk Outline 📋

- What is Core ML? 📦

- How to use it 🔨

- Core ML + Vision 👁️

- Using custom models ⭐

How Does it Work? 📦

- Core ML utilizes machine learning models

- Ex: neural networks

  - "models mimicking the brain"

  - "sophisticated statistical models"

  - "predictors"

- Models are specialized for specific problems

- [How Models are Made](#)

Training and Core ML 📦

- Core ML uses trained models (.coreml)

- Training happens elsewhere

- Tools for training: TensorFlow, Keras, Caffe, …

- [DevSpace session on TensorFlow](#)

Trained Models 📦

- Apple provides [sample models](#)

  - All models are concerned with image or scene classification

- Pros

  - Faster, computationally inexpensive

  - Works offline

- Cons

  - Models cannot adapt to new information

  - Models can be large* *(VGG is 553 MB, SqueezeNet is 5 MB)*

The CoreML Framework 📦

- New with iOS 11

- Built on top of existing frameworks:

    - Metal Performance Shaders (GPU)

    - Accelerate (CPU)

- Creates easy-to-use "black box" interfaces! 😎

Make a Predicition 🔨

- Links to code available at end of presentation*

```swift
// create model object (for image classification)
let mobileNet = MobileNet()

// input is an image (as a pixel buffer)
let pixelBuffer = Converter.toPixelBuffer(image: image)

// simple interface to transform image into a prediction
if let prediction = try? self.mobileNet.prediction(image: pixelBuffer) {
    return prediction.classLabel
}
```

DEMO: First Core ML App 🔨

Improving with Vision 👁️

```swift
// can we get rid of this?
let pixelBuffer = Converter.toPixelBuffer(image: image)
```

- ^ Can we get rid of this?
  - Yes! With the Vision framework
- Provides a wrapper for image-based Core ML models

Setup a "Vision Model" 👁️

```swift
// create vision model
guard let visionModel = try? VNCoreMLModel(for: mobileNet.model) else {
    fatalError("Unable to convert to Vision Core ML Model")
}

// create request "use this model, then pass results to..."
let visionRequest = VNCoreMLRequest(
    model: visionModel,
    completionHandler: self.handleResults
)

// create handler "processes requests"
let handler = VNImageRequestHandler(
    cgImage: image.cgImage,
    orientation: image.orientation
)
```

Perform Inference 👁️

```swift
// start processing requests on a background thread
DispatchQueue.global(qos: .userInitiated).async {
    do {
        try handler.perform([visionRequest])
    } catch {
        print("Error performing classification")
    }
}
```

Handle Results 👁️

```swift
// handle the results!
func handleResults(for request: VNRequest, error: Error?) {

    // switch back to main thread
    DispatchQueue.main.async {

        // ensure there are results
        guard let classifications = request.results,
            let topClassification = classifications.first else {
                print("nothing recognized")
        }

        // do something with results!
        let imageClass = topClassification.identifier
    }
}
```

DEMO: Core ML + Vision 👁️

Better Vision 👁️

- One step further? Real-time object classification

  - Uses Vision and AVFoundation frameworks

Capture Output 👁

```swift
func captureOutput(_ output: AVCaptureOutput,
    didOutput sampleBuffer: CMSampleBuffer,
    from connection: AVCaptureConnection) {

    guard let buffer = CMSampleBufferGetImageBuffer(sampleBuffer) else {
        return
    }

    // create handler (OLD WAY)
    // let handler = VNImageRequestHandler(cgImage: image.cgImage,
    //                                     orientation: image.orientation)

    // create handler "processes requests"
    let handler = VNImageRequestHandler(cvPixelBuffer: buffer,
        orientation: .upMirrored, options: requestOptions)
```

Capture Output 👁

```swift
    // handler created...

    // "use this model, then pass results to..."
    //
    // let visionRequest = VNCoreMLRequest(model: model,
    //                              completionHandler: handleResults)
    //
    // let visionRequests: [VNRequest] = [visionRequest]

    // start processing requests!
    do {
      try handler.perform(self.visionRequests)
    } catch {
      print(error)
    }
}
```

DEMO: Real-Time Object Classification 👁

[Video Link](#)

Using a Custom Model ⭐

- Can you do something other than image classification?

- Yes, a few examples...

  ○ Recommendations (Netflix, Amazon, etc.)

  ○ Image style transfer

  ○ Custom glasses from facial scans (Topology Eyewear)

- Many things can be built on top of image classification

  ○ Skin cancer detector

How? coremltools ⭐

- A python tool for creating and testing Core ML models
- Can convert popular formats into Core ML models
    - Keras
    - Caffe
    - Tensowflow
    - Xgboost
    - scikit-learn
    - libSVM

Model Converstion ⭐

```python
from keras.models import load_model
import coremltools

# load keras models
model = load_model('food101-model.hdf5')

# create core ml model
coreml_model = coremltools.converters.keras.convert(
    model,
    input_names=['image'],
    output_names=['confidence'],
    class_labels='labels.txt'
)

# set metadata...
coreml_model.author = 'Udacity'
```

Test and Save Model ⭐

```python
from PIL import Image

# get image input
bibimbap = Image.open('bibimbap.jpg')

# perform inference
coreml_model.predict({'image' : bibimbap})

# check outputs, verify top classification...

# save model
coreml_model.save('Food101Net.mlmodel')
```

DEMO: Using a Custom Model ⭐

Review

- Core ML, "on-device" machine intelligence! 🎉

- Easy-to-use interface

- Supports *many* custom models

- Used extensively by Apple apps, and more to come... ⏰

Questions ✋?

Slides available @ **jarrodparkes.com**

Thank You 🙏!

- [Code Samples](#)

~

- [Apple: Machine Learning](#)

- [Apple Docs: Core ML](#) and [coremltools](#)

- [Clarifai](#) (new SDK for "on-device training")

- [Udacity: ML Nanodegree Program](#)

- [Coursera: Machine Learning (Andrew Ng)](#)

DevSpace would like to thank our sponsors