#WWDC18

Vision with Core ML

Powerful Computer Vision made easy

Session 717

Frank Doepke, Flat Pixel Enthusiast

Custom image classification

Object recognition

Vision fundamentals

Custom Image classification

Object recognition

Vision fundamentals

Custom Image classification

Object recognition

Vision fundamentals

Custom Image Classification

Worth more than a thousand pictures

The Storyline

Create an app helping shoppers identify items

The Storyline

Create an app helping shoppers identify items

- Train a custom classifier

The Storyline

Create an app helping shoppers identify items

- Train a custom classifier
- Build an iOS app

The Storyline

Create an app helping shoppers identify items

- Train a custom classifier
- Build an iOS app
- Keep an eye on pitfalls

Our Training Regimen

Train using Create ML

Our Training Regimen

Train using Create ML

Take pictures

Our Training Regimen

Train using Create ML

Take pictures

Sort into folders—the folder names are used as labels

Our Training Regimen

Train using Create ML

Take pictures

Sort into folders—the folder names are used as labels

- How much data do I need

Our Training Regimen

Train using Create ML

Take pictures

Sort into folders—the folder names are used as labels

- How much data do I need
  - Minimum of 10 per category but more is better

Our Training Regimen

Train using Create ML

Take pictures

Sort into folders—the folder names are used as labels

- How much data do I need
  - Minimum of 10 per category but more is better
  - Highly imbalanced datasets are hard to train

Our Training Regimen

Train using Create ML

Take pictures

Sort into folders—the folder names are used as labels

- How much data do I need
  - Minimum of 10 per category but more is better
  - Highly imbalanced datasets are hard to train
- Augmentation adds some robustness on top of it but doesn’t replace variety

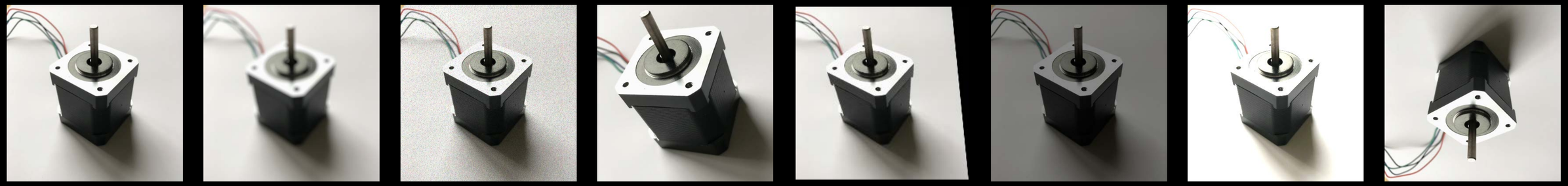
Our Training Regimen

Train using Create ML

Take pictures

Sort into folders—the folder names are used as labels

- How much data do I need
  - Minimum of 10 per category but more is better
  - Highly imbalanced datasets are hard to train
- Augmentation adds some robustness on top of it but doesn’t replace variety



Under the Hood

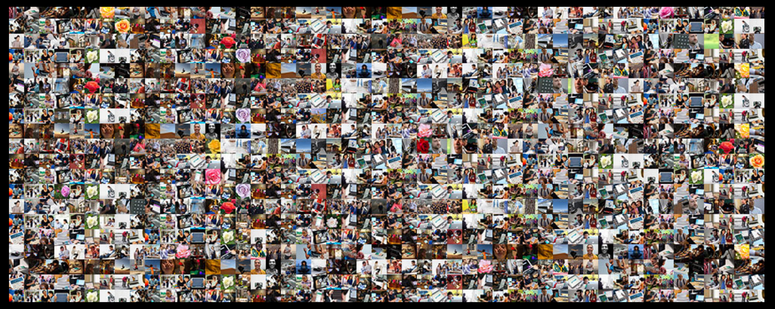
Under the Hood

Transfer Learning

Under the Hood

Transfer Learning

- Starts with a pre-trained model
  - This is the heavy load

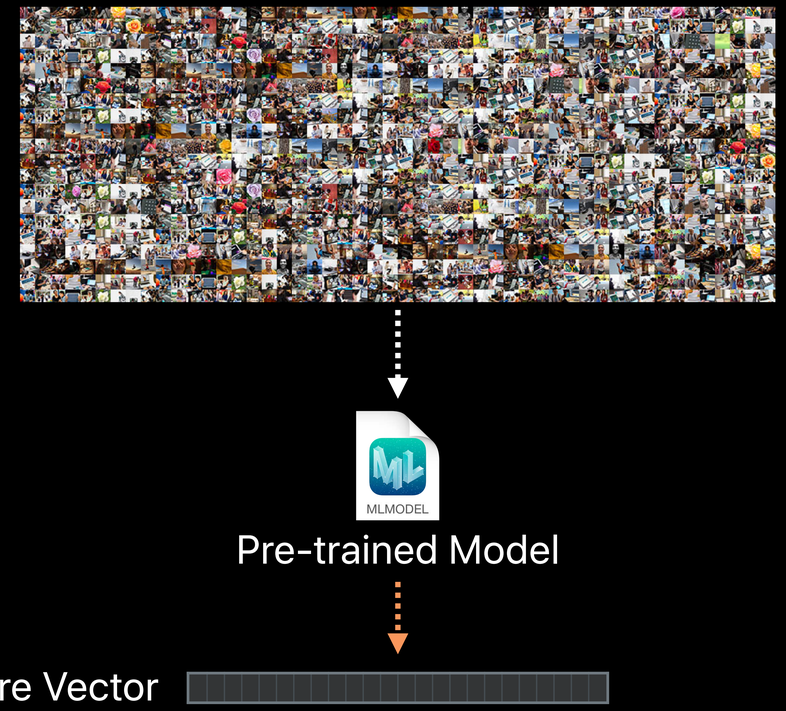


Pre-trained Model

Under the Hood

Transfer Learning

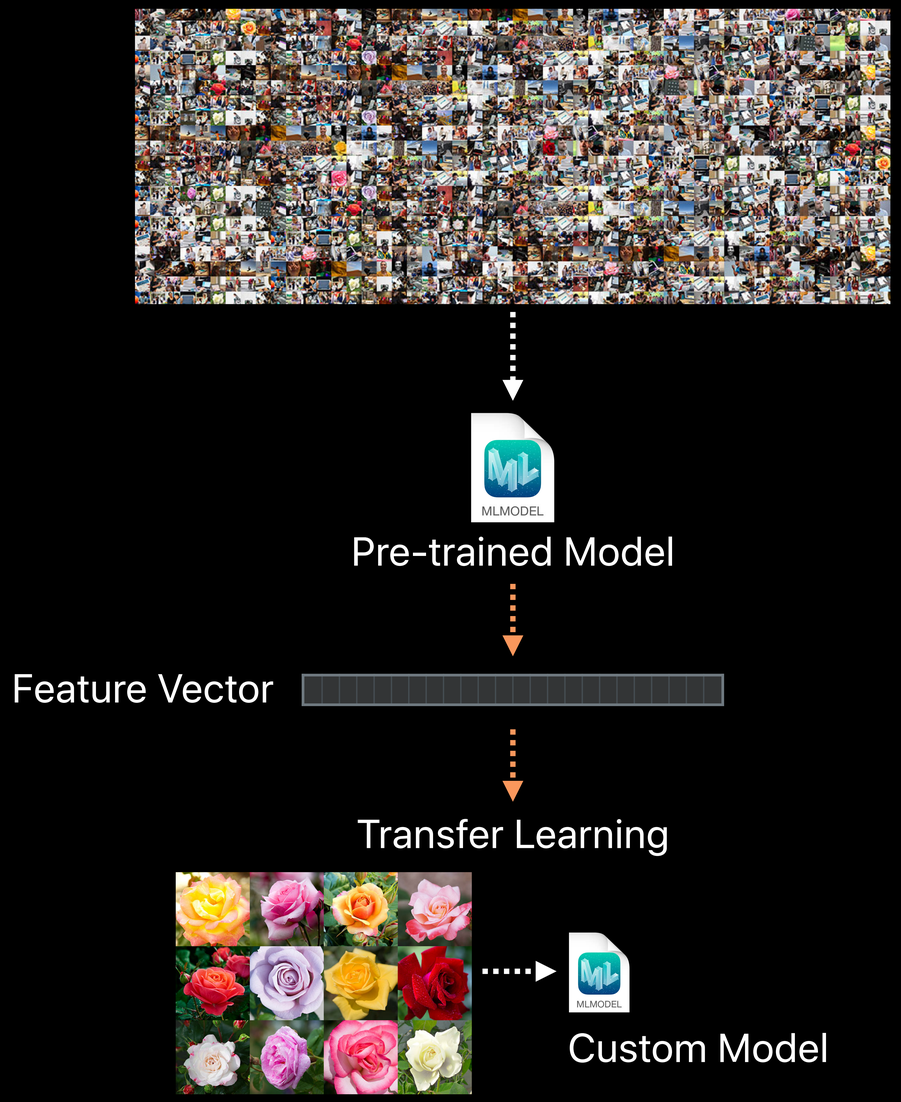
- Starts with a pre-trained model
  - This is the heavy load
- Use that model as a Feature Extractor



Under the Hood

Transfer Learning

- Starts with a pre-trained model
  - This is the heavy load
- Use that model as a Feature Extractor
- Train a last layer as a classifier with your labeled data



Vision FeaturePrint.Scene

Vision FeaturePrint.Scene

Vision Frameworks FeaturePrint for Image Classification

Vision FeaturePrint.Scene

Vision Frameworks FeaturePrint for Image Classification

- Available through ImageClassifier training in Create ML

Vision FeaturePrint.Scene

Vision Frameworks FeaturePrint for Image Classification

- Available through ImageClassifier training in Create ML
- Trained on a very large dataset

Vision FeaturePrint.Scene

Vision Frameworks FeaturePrint for Image Classification

- Available through ImageClassifier training in Create ML
- Trained on a very large dataset
- Capable of predicting over 1000 categories

Vision FeaturePrint.Scene

Vision Frameworks FeaturePrint for Image Classification

- Available through ImageClassifier training in Create ML
- Trained on a very large dataset
- Capable of predicting over 1000 categories
- Powers user facing features in Photos

Vision FeaturePrint.Scene

Vision Frameworks FeaturePrint for Image Classification

- Available through ImageClassifier training in Create ML
- Trained on a very large dataset
- Capable of predicting over 1000 categories
- Powers user facing features in Photos
- Continuous improvement

Vision FeaturePrint.Scene

Vision Frameworks FeaturePrint for Image Classification

- Available through ImageClassifier training in Create ML
- Trained on a very large dataset
- Capable of predicting over 1000 categories
- Powers user facing features in Photos
- Continuous improvement
  - You might want to retrain in the future

Vision FeaturePrint.Scene

Vision FeaturePrint.Scene

Already on device

Vision FeaturePrint.Scene

Already on device

- Smaller disk footprint for your custom model

Vision FeaturePrint.Scene

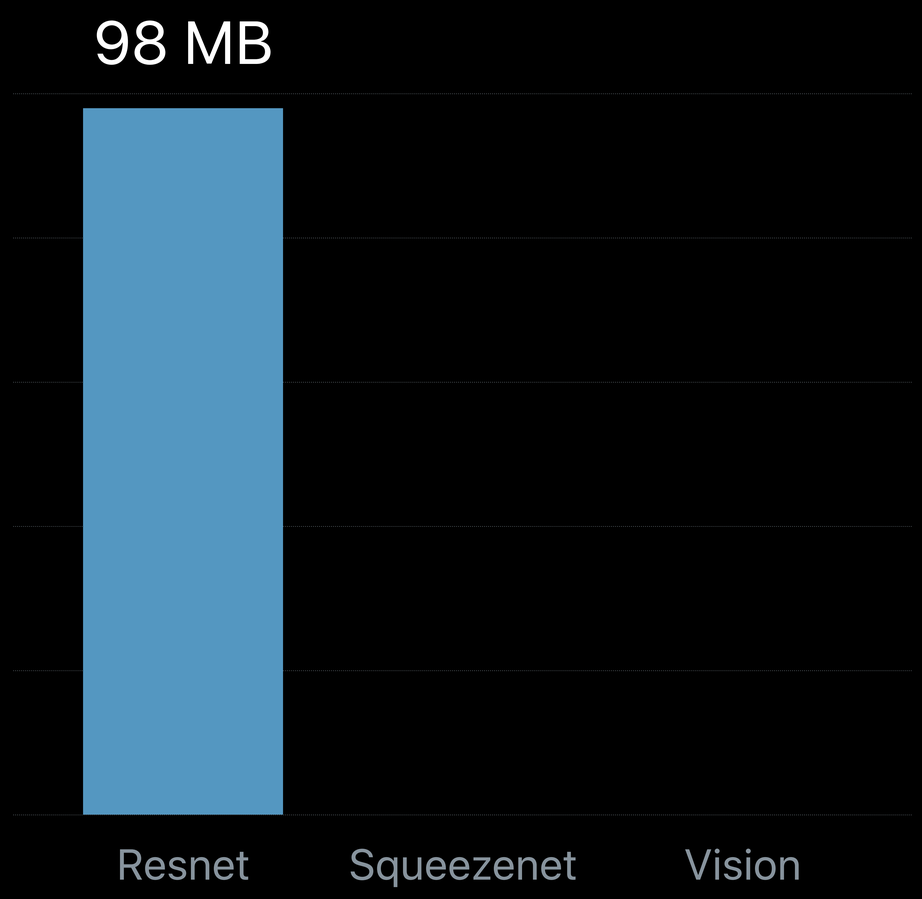
Already on device

- Smaller disk footprint for your custom model

Vision FeaturePrint.Scene

Already on device

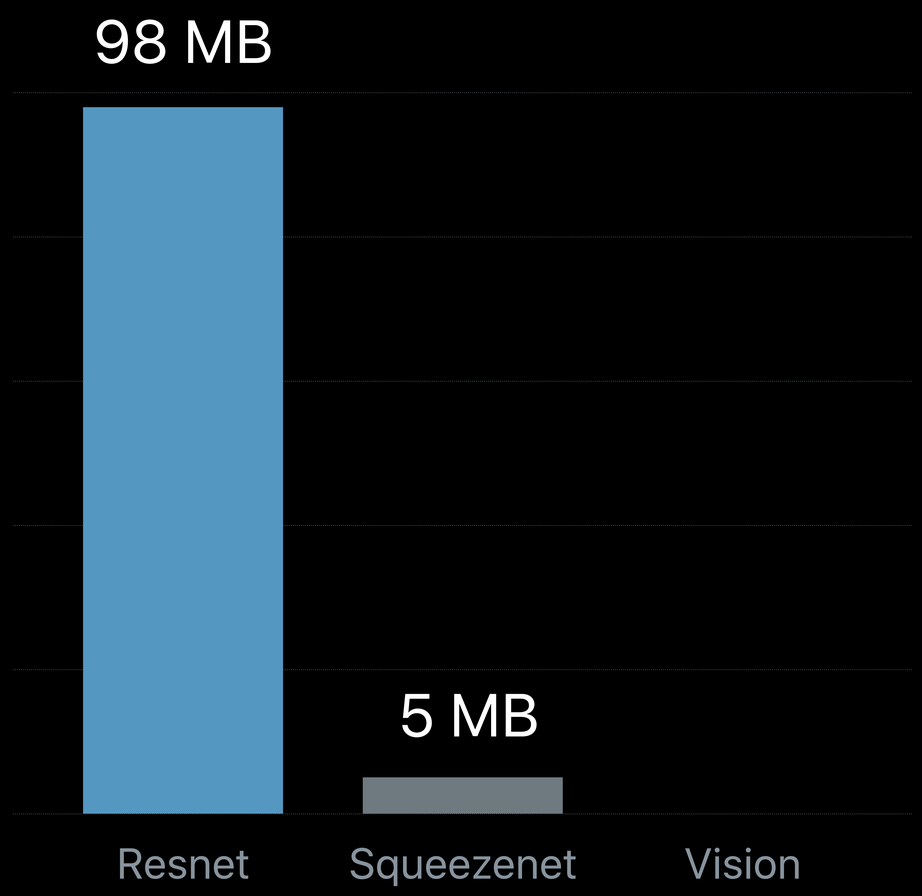
- Smaller disk footprint for your custom model



Vision FeaturePrint.Scene

Already on device

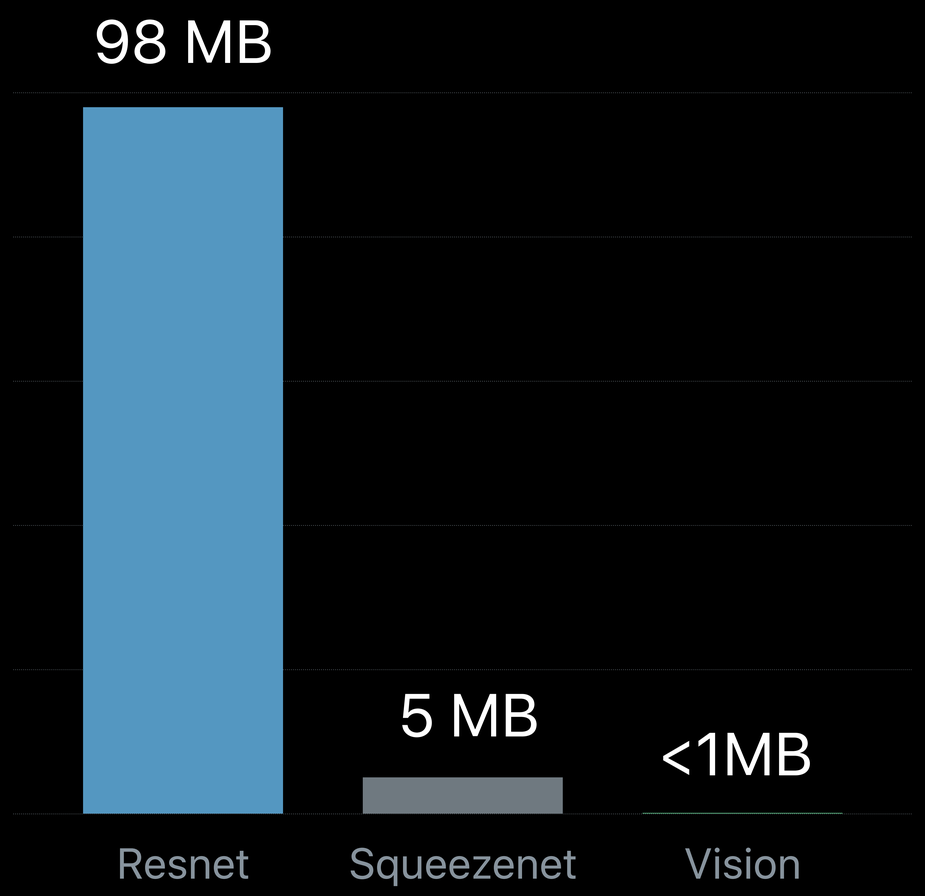
- Smaller disk footprint for your custom model



Vision FeaturePrint.Scene

Already on device

- Smaller disk footprint for your custom model

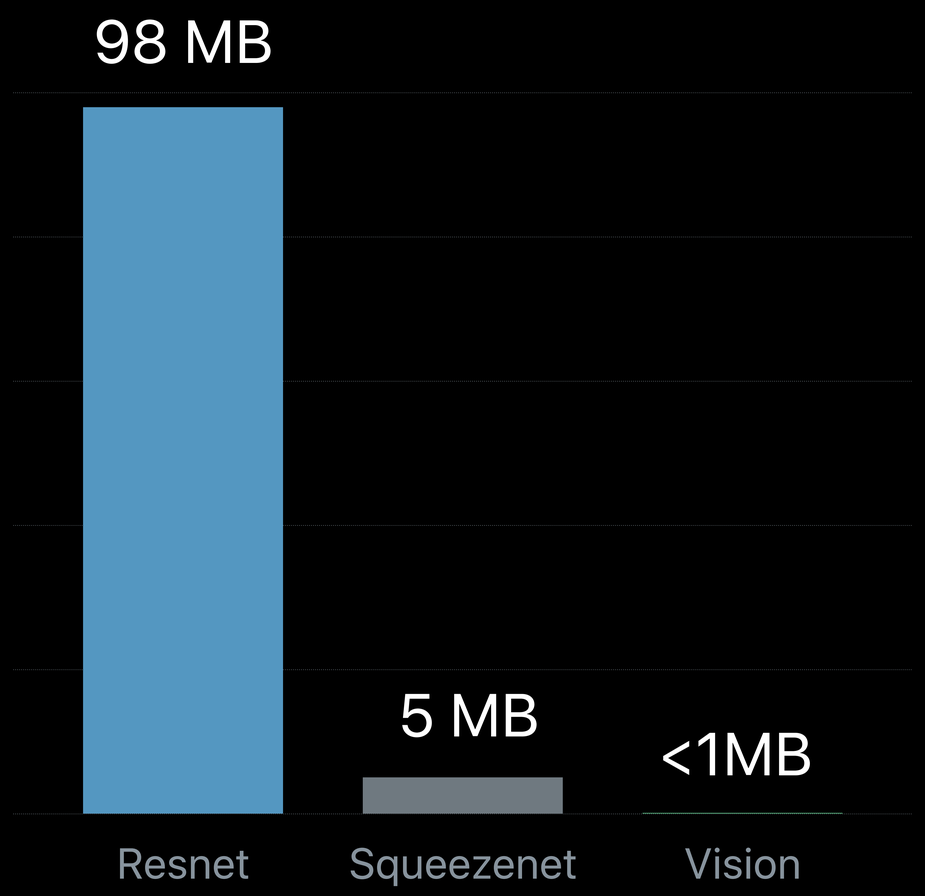


Vision FeaturePrint.Scene

Already on device

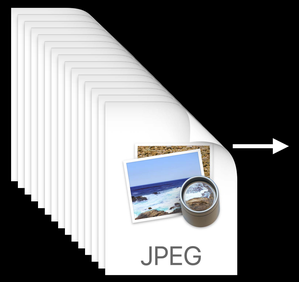
- Smaller disk footprint for your custom model

Optimized for Apple devices



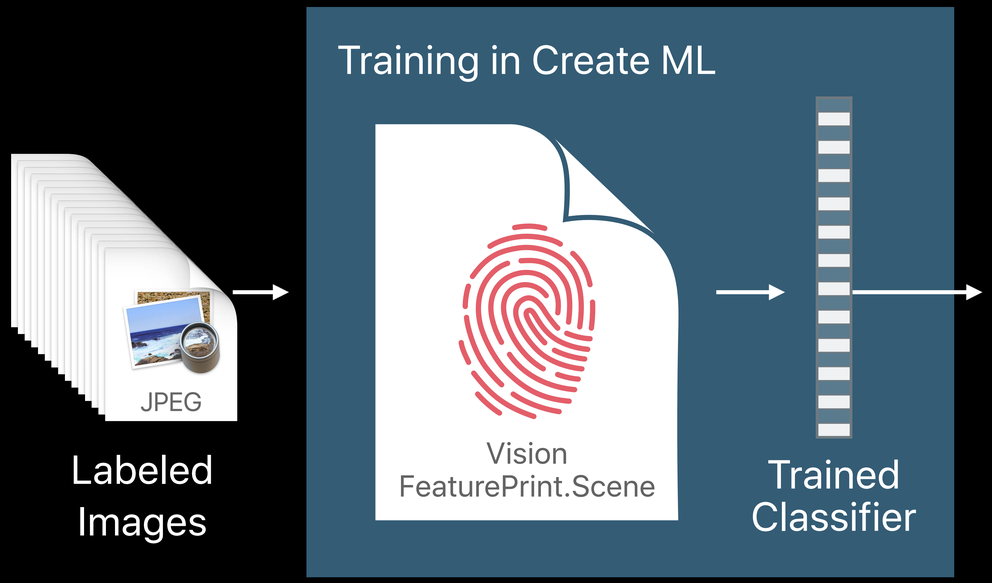
Vision FeaturePrint.Scene

Vision FeaturePrint.Scene

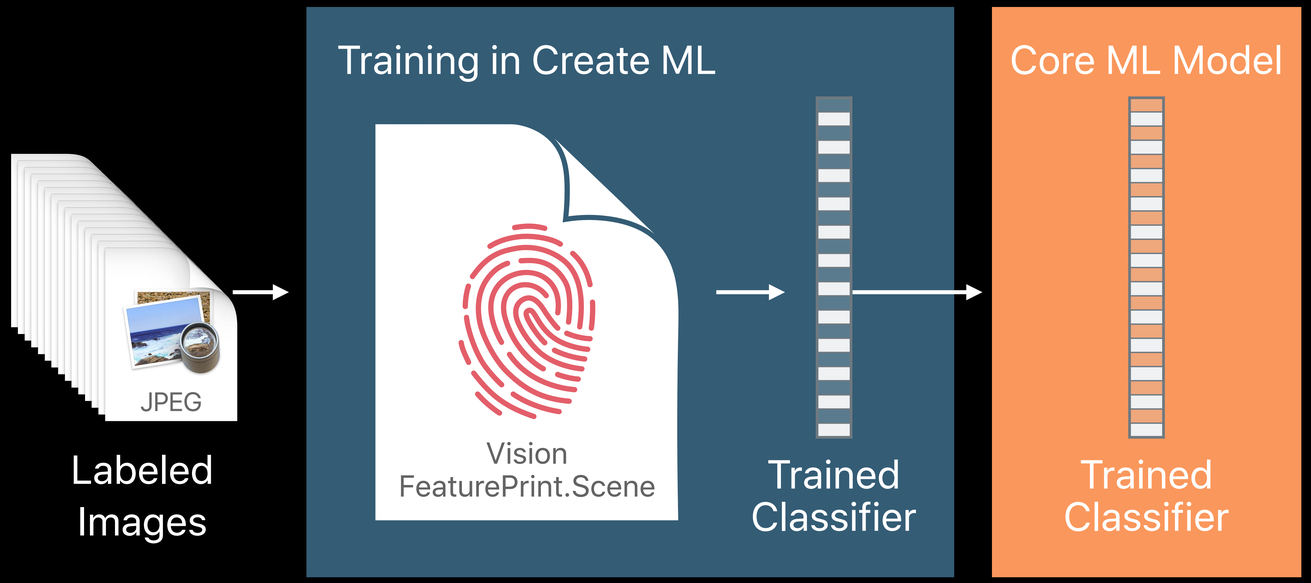


Labeled
Images

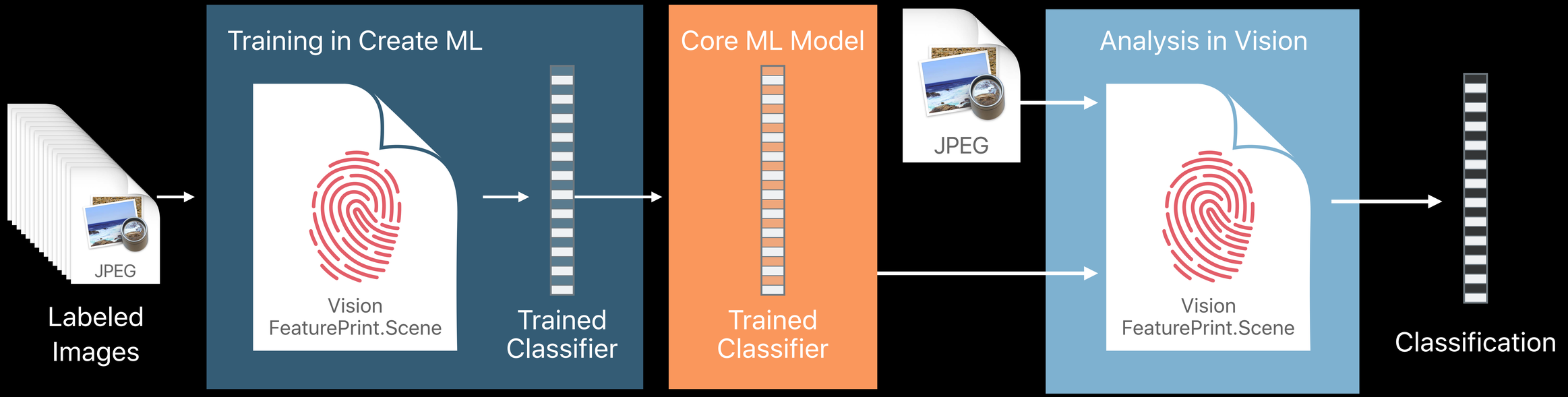
Vision FeaturePrint.Scene



Vision FeaturePrint.Scene



Vision FeaturePrint.Scene



Refining the App

Only classify when needed

Refining the App

Only classify when needed

Don’t run expensive tasks when not needed

Refining the App

Only classify when needed

Don’t run expensive tasks when not needed

Am I holding still?

Refining the App

Only classify when needed

Don’t run expensive tasks when not needed

Am I holding still?

Using registration

- Cheap and fast
- Camera holds still
- Subject is not moving

Refining the App

Only classify when needed

Don’t run expensive tasks when not needed

Am I holding still?

Using registration

- Cheap and fast
- Camera holds still
- Subject is not moving

```
VNTranslationalImageRegistrationRequest
```

Refining the App

Only classify when needed

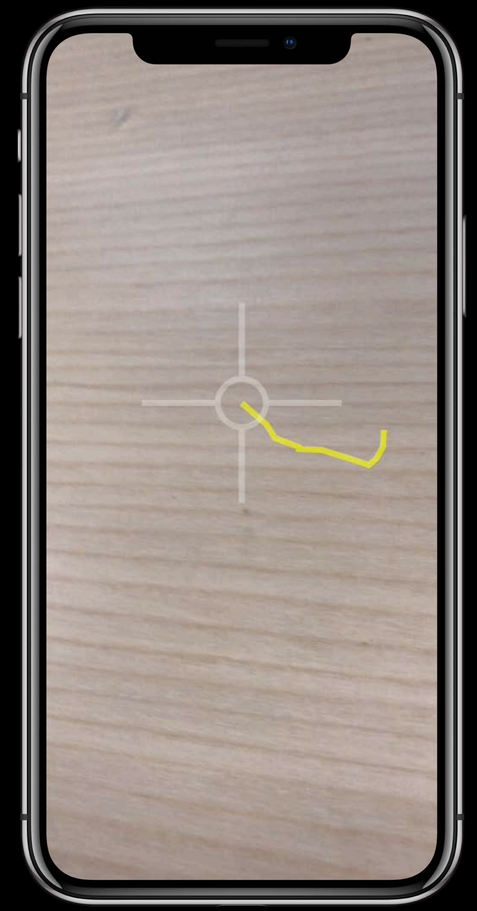
Don’t run expensive tasks when not needed

Am I holding still?

Using registration

- Cheap and fast
- Camera holds still
- Subject is not moving

```
VNTranslationalImageRegistrationRequest
```



Refining the App

Always have a backup plan

Refining the App

Always have a backup plan

Classifications can be wrong

Refining the App

Always have a backup plan

Classifications can be wrong

Even when confidence is high > plan for it

Refining the App

Always have a backup plan

Classifications can be wrong

Even when confidence is high > plan for it

Alternative identification

- Barcode reading

Demo

Build the RobotShop

Recap

Recap

Using Registration for Scene Stability

Recap

Using Registration for Scene Stability

Use the `VNSequenceRequestHandler` with `VNTranslationalImageRegistrationRequest`

Recap

Using Registration for Scene Stability

Use the `VNSequenceRequestHandler` with `VNTranslationalImageRegistrationRequest`

Compare against previous frame:

```
sequenceRequestHandler.perform([request], on: previousBuffer!)
```

Recap

Using Registration for Scene Stability

Use the `VNSequenceRequestHandler` with `VNTranslationalImageRegistrationRequest`

Compare against previous frame:

```
sequenceRequestHandler.perform([request], on: previousBuffer!)
```

Registration is returned as pixels in the `alignmentObservation.alignmentTransform`

Recap

Recap

Analyze only when scene is stable

Recap

Analyze only when scene is stable

Create an `VNImageRequestHandler` for the current frame and pass in the orientation

Recap

Analyze only when scene is stable

Create an `VNImageRequestHandler` for the current frame and pass in the orientation

Perform Barcode and Image Classification together

```
try imageRequestHandler.perform([barcodeDetection, imageClassification])
```

Recap

Recap

Manage your buffers

Recap

Manage your buffers

Some Vision requests can take longer

Recap

Manage your buffers

Some Vision requests can take longer

Perform longer task asynchronously

Recap

Manage your buffers

Some Vision requests can take longer

Perform longer task asynchronously

Do not queue up more buffers than the camera can provide

- We only operate with a one deep queue in this example

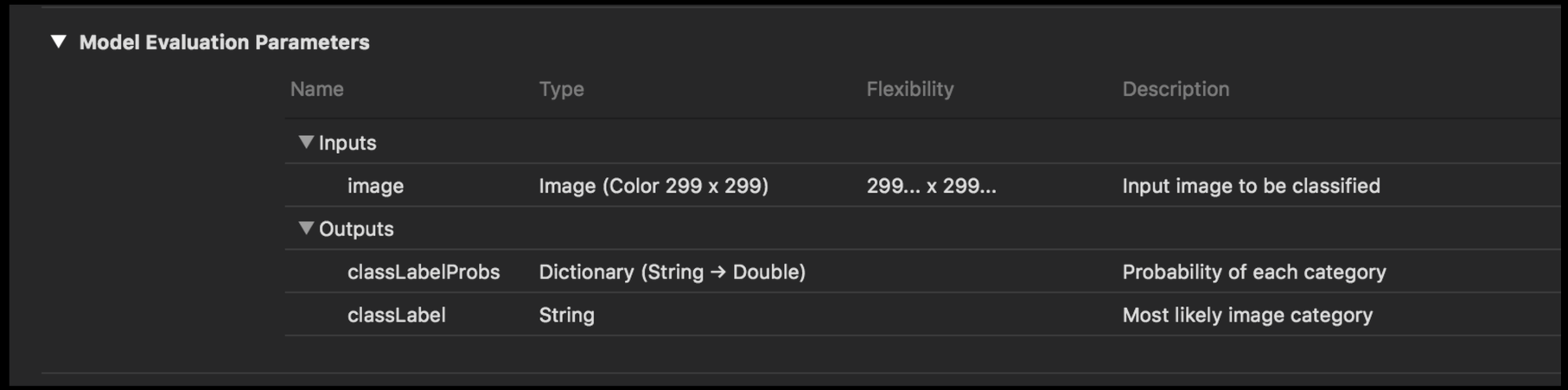
Recap

Recap

Why not use just Core ML?

Recap

Why not use just Core ML?



Model Evaluation Parameters

Name	Type	Flexibility	Description
Inputs			
image	Image (Color 299 x 299)	299... x 299...	Input image to be classified
Outputs			
classLabelProbs	Dictionary (String → Double)		Probability of each category
classLabel	String		Most likely image category

Recap

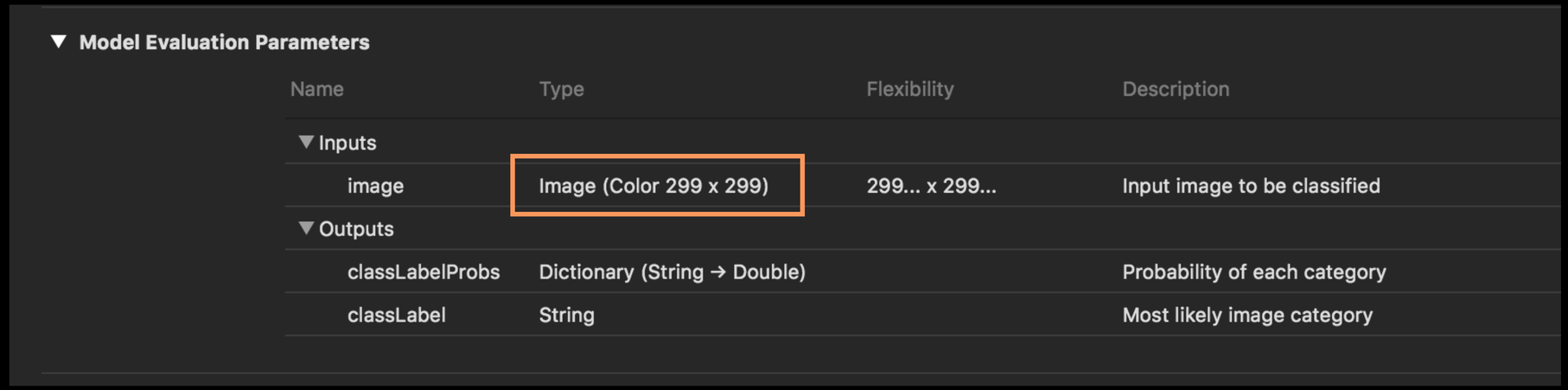
Why not use just Core ML?

Model Evaluation Parameters

Name	Type	Flexibility	Description
Inputs			
image	Image (Color 299 x 299)	299... x 299...	Input image to be classified
Outputs			
classLabelProbs	Dictionary (String → Double)		Probability of each category
classLabel	String		Most likely image category

Recap

Why not use just Core ML?



Model Evaluation Parameters

Name	Type	Flexibility	Description
▼ Inputs			
image	Image (Color 299 x 299)	299... x 299...	Input image to be classified
▼ Outputs			
classLabelProbs	Dictionary (String → Double)		Probability of each category
classLabel	String		Most likely image category

Vision does all the scaling and color conversion for you

Object Recognition

I spy with my little eye

What and How

YOLO (You Only Look Once)

Fast Object Detection and Classification

- Label and Bounding Box
- Finds multiple and different objects

Train for custom objects

- Training is more involved than ImageClassifier

What and How

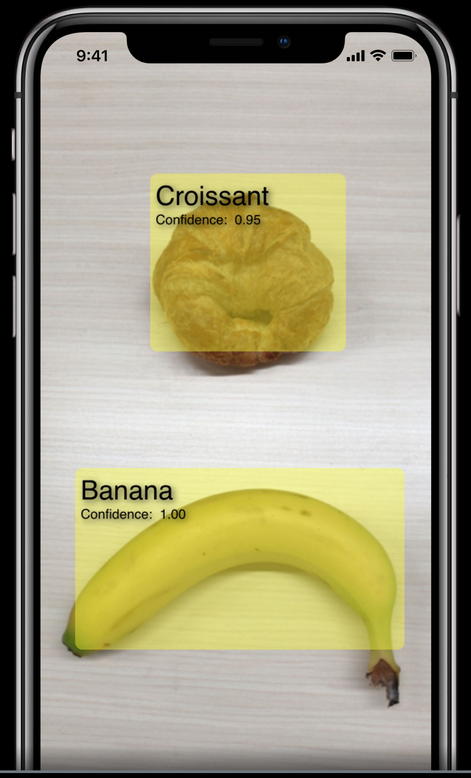
YOLO (You Only Look Once)

Fast Object Detection and Classification

- Label and Bounding Box
- Finds multiple and different objects

Train for custom objects

- Training is more involved than ImageClassifier



Demo

Where is my breakfast

VNRecognizedObjectObservation

NEW

VNRecognizedObjectObservation

NEW

Result of a `VNCoreMLModelRequest`

VNRecognizedObjectObservation

NEW

Result of a `VNCoreMLModelRequest`

New observation subclass `VNRecognizedObjectObservation`

VNRecognizedObjectObservation

NEW

Result of a `VNCoreMLModelRequest`

New observation subclass `VNRecognizedObjectObservation`

YOLO based models made easy

```
let mlModel = try MLModel(contentsOf: modelURL)
let visionModel = try VNCoreMLModel(for: mlModel)
let objectRecognition = VNCoreMLRequest(model: visionModel,
                                        completionHandler: { (request, error) in
    guard let results = request.results else { return }

    for case let foundObject as VNRecognizedObjectObservation in results {
        let bestLabel = foundObject.labels.first! // Label with highest confidence
        let objectBounds = foundObject.boundingBox

        // Use the computed values.
        print(bestLabel.identifier, bestLabel.confidence, objectBounds)
    }
})
```

Tracking

Tracking

Tracking is faster and smoother than re-detection

Tracking

Tracking is faster and smoother than re-detection

- Use tracking to follow a detected object

Tracking

Tracking is faster and smoother than re-detection

- Use tracking to follow a detected object
- Tracking is a lighter algorithm

Tracking

Tracking is faster and smoother than re-detection

- Use tracking to follow a detected object
- Tracking is a lighter algorithm
- Applies temporal smoothing

Tracking

Tracking is faster and smoother than re-detection

- Use tracking to follow a detected object
- Tracking is a lighter algorithm
- Applies temporal smoothing

Vision Fundamentals

The tripod to computer vision

Image Orientation

Image Orientation

Not all algorithms are orientation agnostic

Image Orientation

Not all algorithms are orientation agnostic

Images are not always upright

- EXIF orientation defines what is upright
- When using a URL as input Vision reads the EXIF orientation from file

Image Orientation

Not all algorithms are orientation agnostic

Images are not always upright

- EXIF orientation defines what is upright
- When using a URL as input Vision reads the EXIF orientation from file

Live from a capture feed

Image Orientation

Not all algorithms are orientation agnostic

Images are not always upright

- EXIF orientation defines what is upright
- When using a URL as input Vision reads the EXIF orientation from file

Live from a capture feed

- Orientation has to be inferred from `UIDevice.current.orientation`

Image Orientation

Not all algorithms are orientation agnostic

Images are not always upright

- EXIF orientation defines what is upright
- When using a URL as input Vision reads the EXIF orientation from file

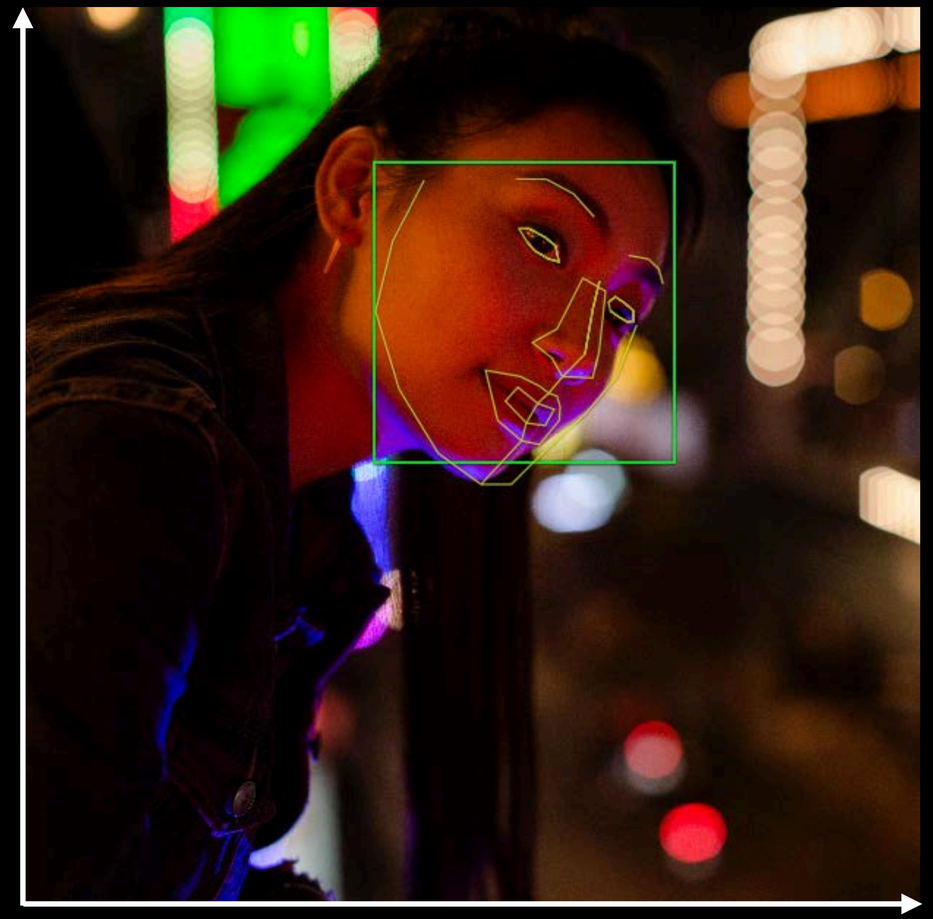
Live from a capture feed

- Orientation has to be inferred from `UIDevice.current.orientation`
- Needs to be mapped to a `CGImagePropertyOrientation`

Vision Coordinate System

Vision Coordinate System

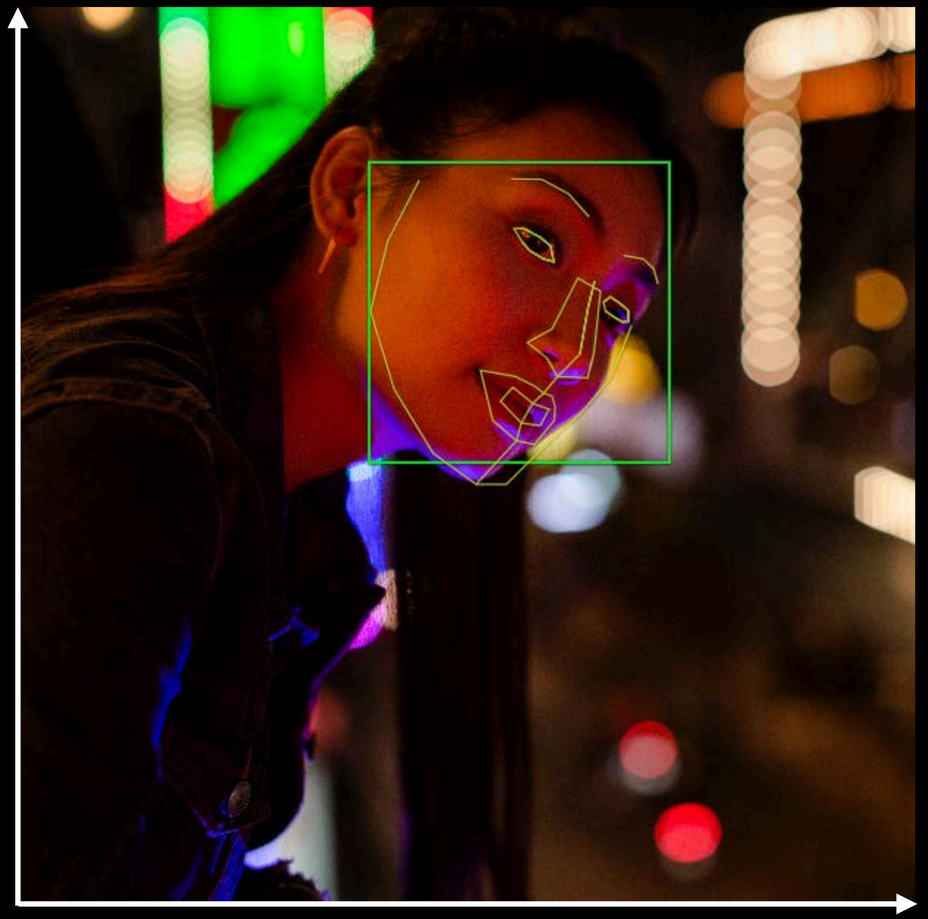
Origin is at the lower-left corner



Vision Coordinate System

Origin is at the lower-left corner

All processing is in relation to the image in upright coordinates

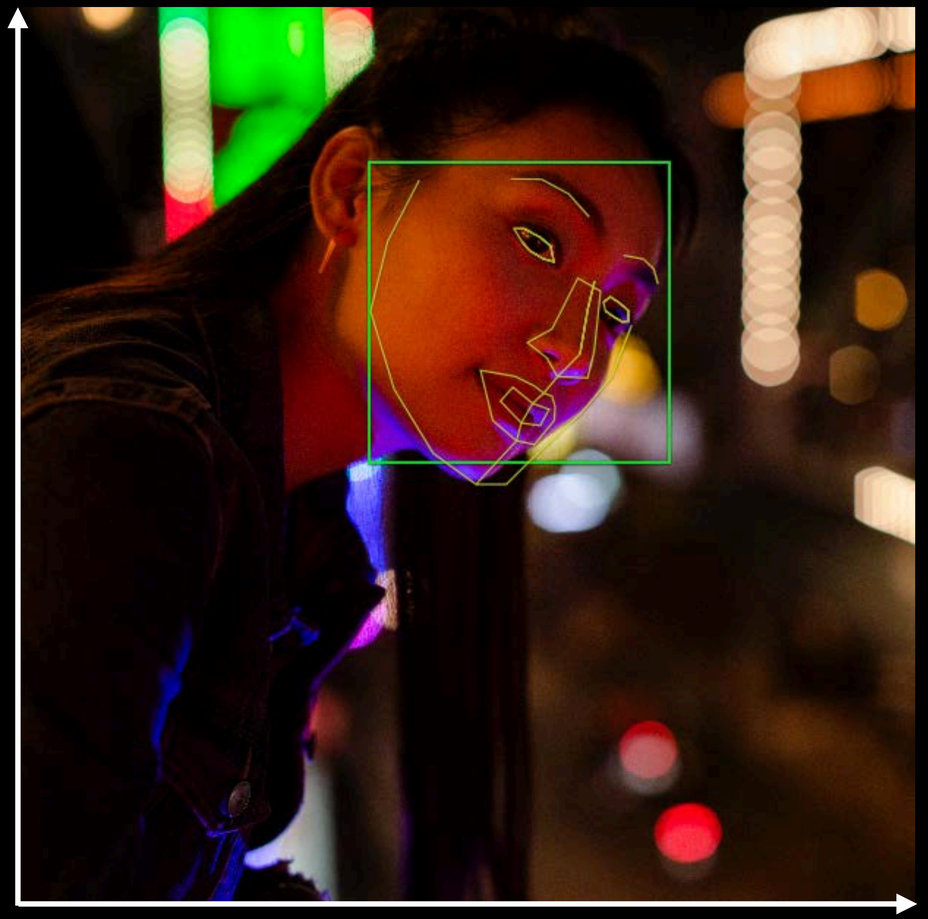


Vision Coordinate System

Origin is at the lower-left corner

All processing is in relation to the image in upright coordinates

Normalized coordinates



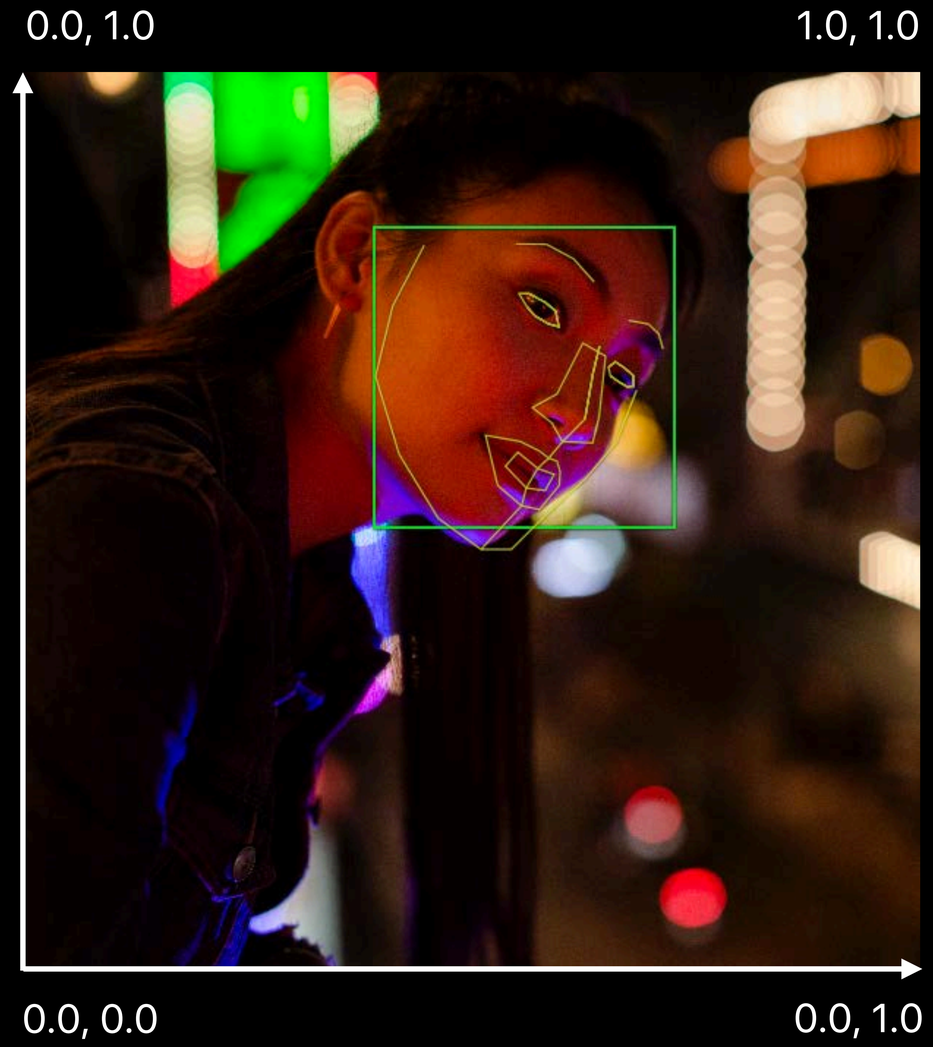
Vision Coordinate System

Origin is at the lower-left corner

All processing is in relation to the image in upright coordinates

Normalized coordinates

- 0.0 to 1.0



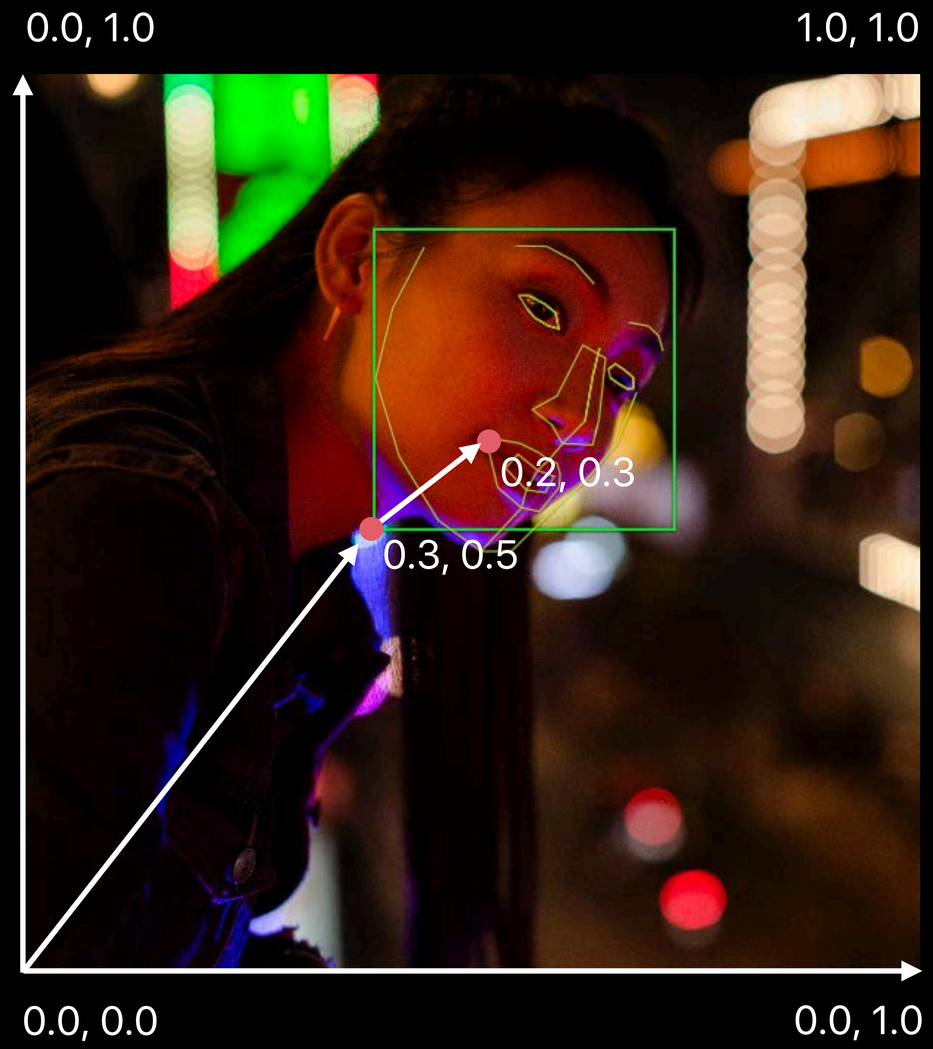
Vision Coordinate System

Origin is at the lower-left corner

All processing is in relation to the image in upright coordinates

Normalized coordinates

- 0.0 to 1.0
- Landmarks are relative to the face rectangle



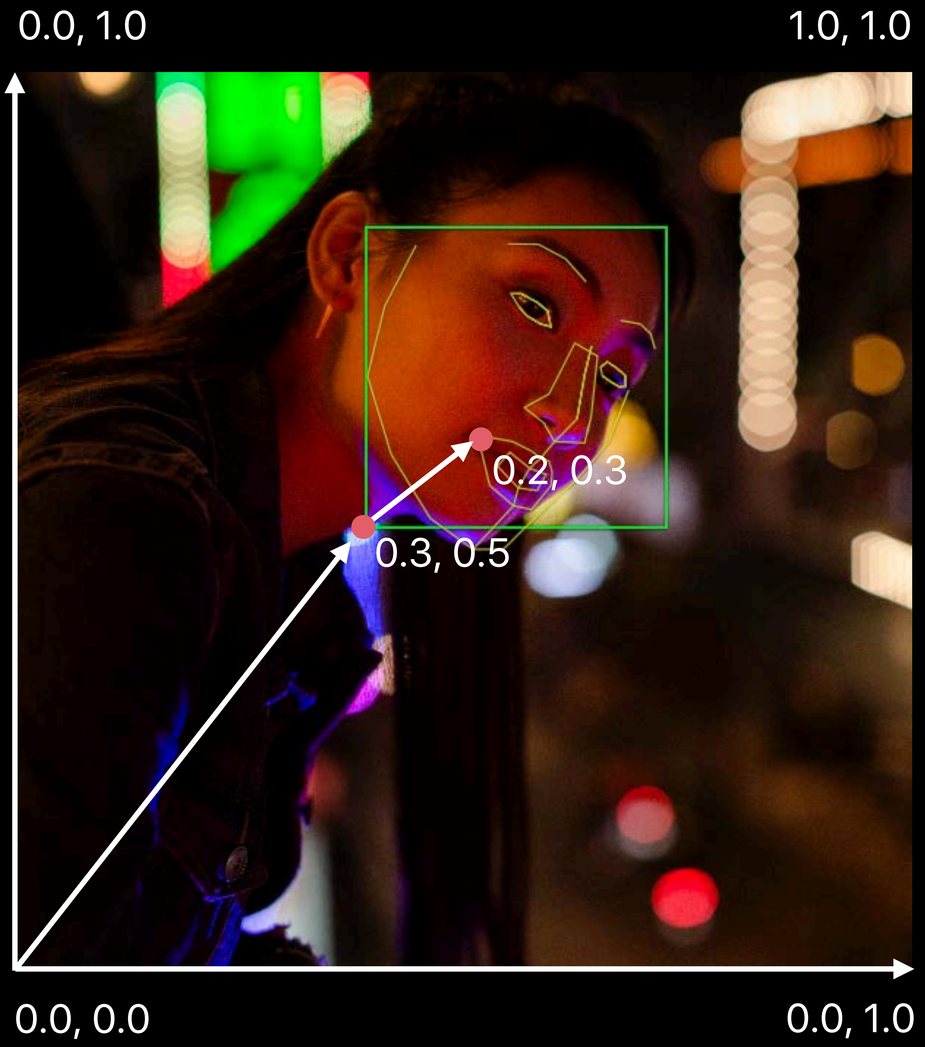
Vision Coordinate System

Origin is at the lower-left corner

All processing is in relation to the image in upright coordinates

Normalized coordinates

- 0.0 to 1.0
- Landmarks are relative to the face rectangle
- `VNUtils.h` provides conversion utils into image coordinates like `VNImageRectForNormalizedRect`



Confidence Score

Confidence Score

A lot of algorithm can express how certain they are about the results

Confidence Score

A lot of algorithm can express how certain they are about the results

Confidence is expressed between 0.0 (low) and 1.0 (highest)

Confidence Score

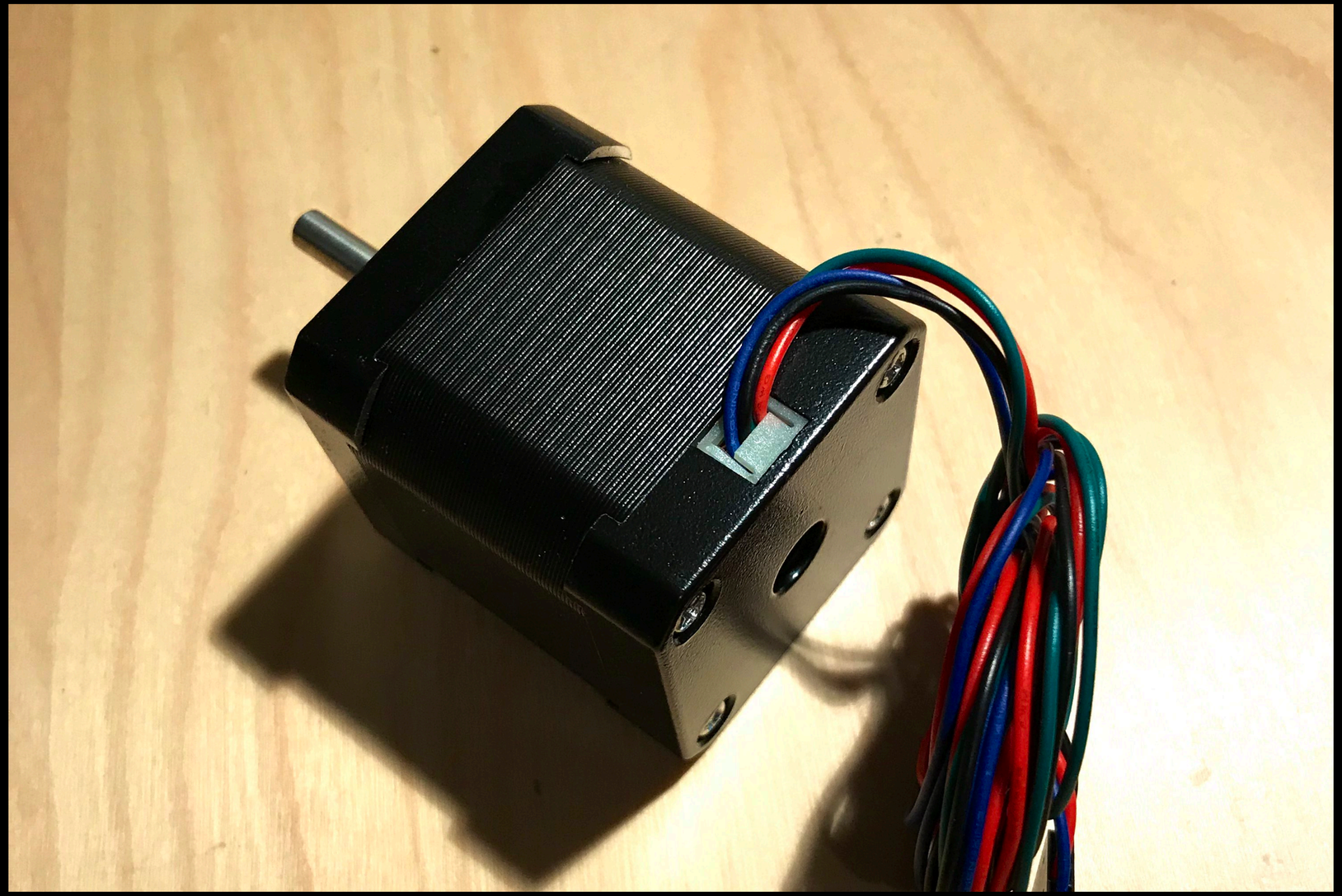
A lot of algorithm can express how certain they are about the results

Confidence is expressed between 0.0 (low) and 1.0 (highest)

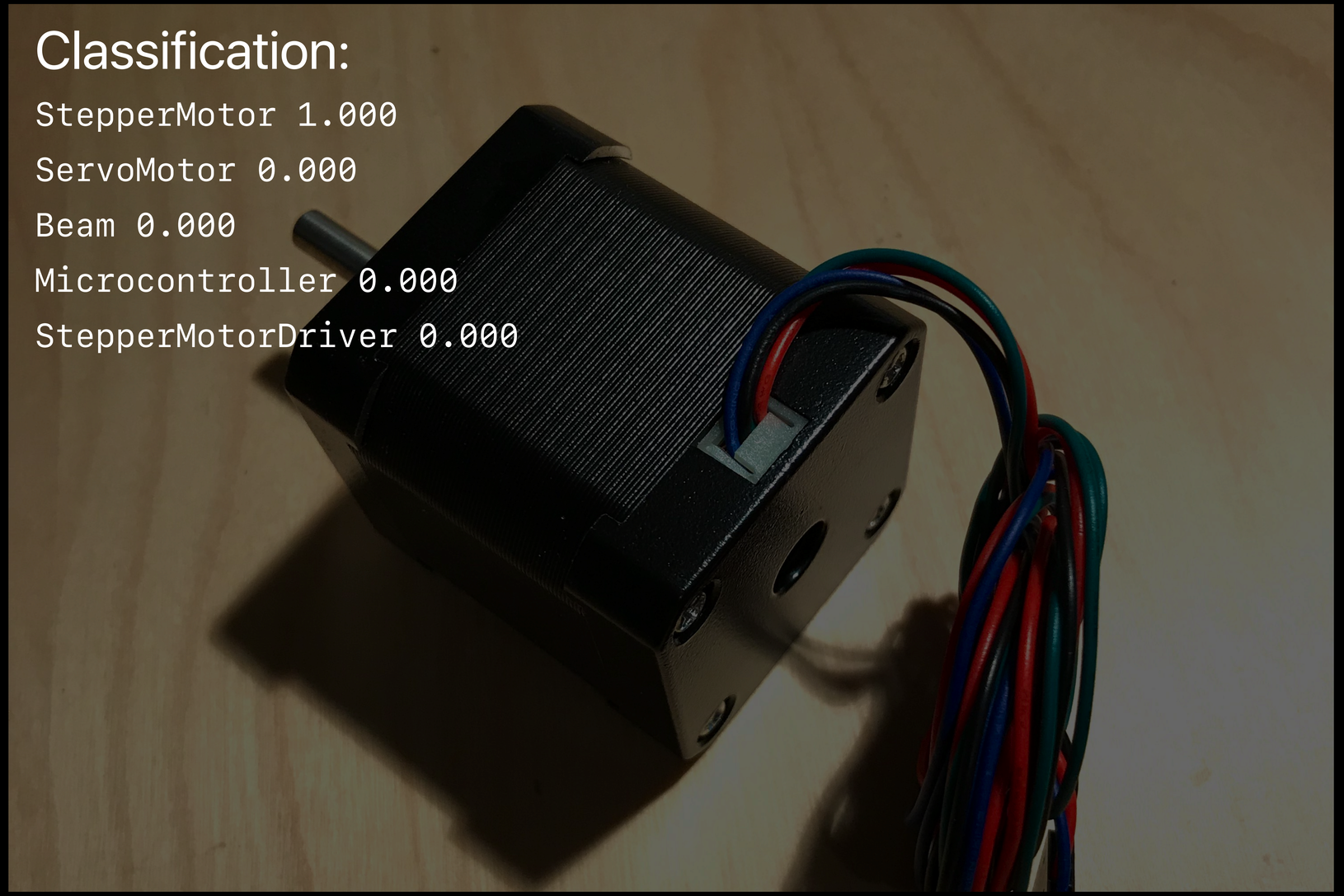
The scale is not uniform across request types

Classification Example

Classification Example



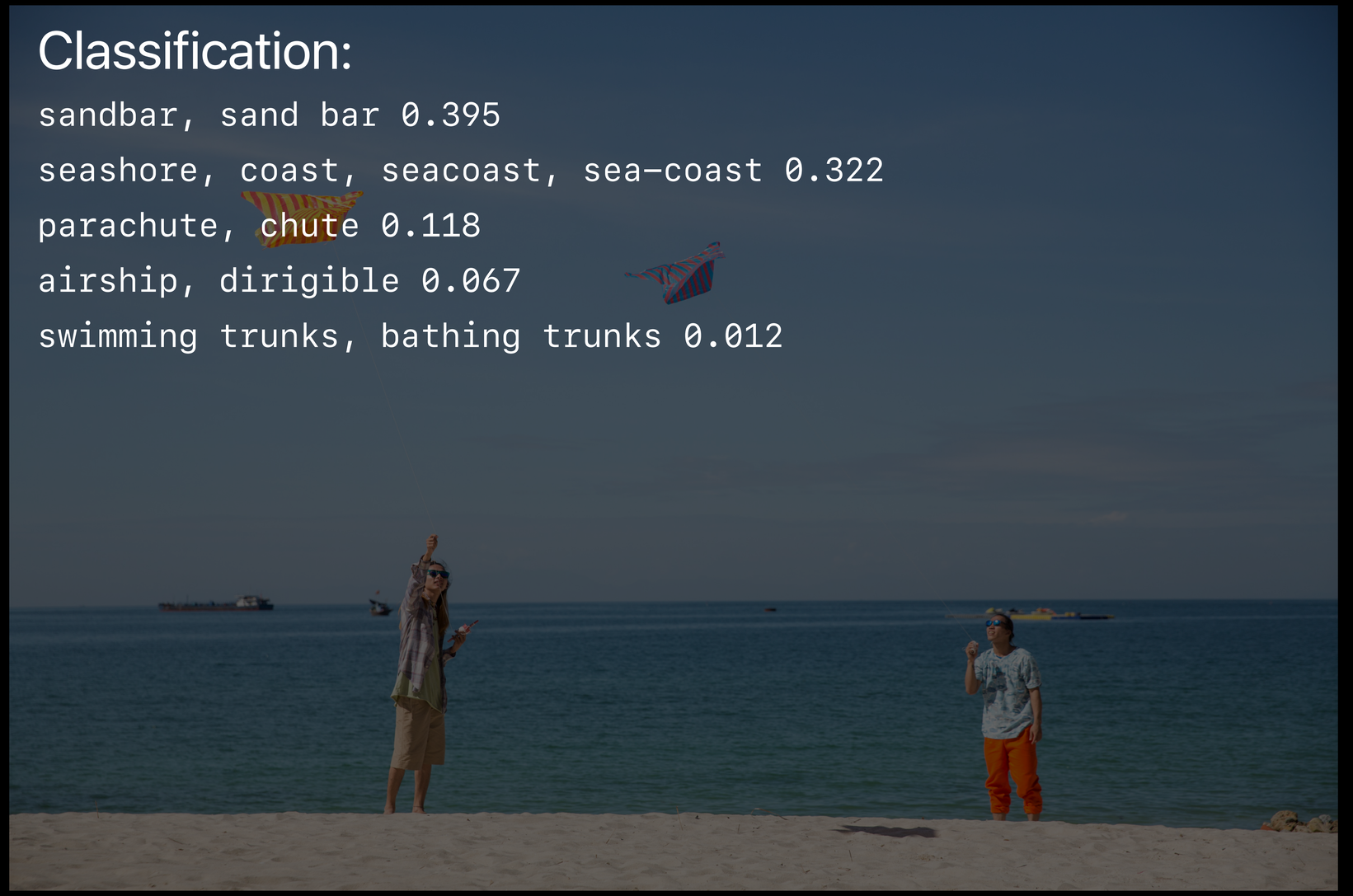
Classification Example



Classification Example

Classification Example

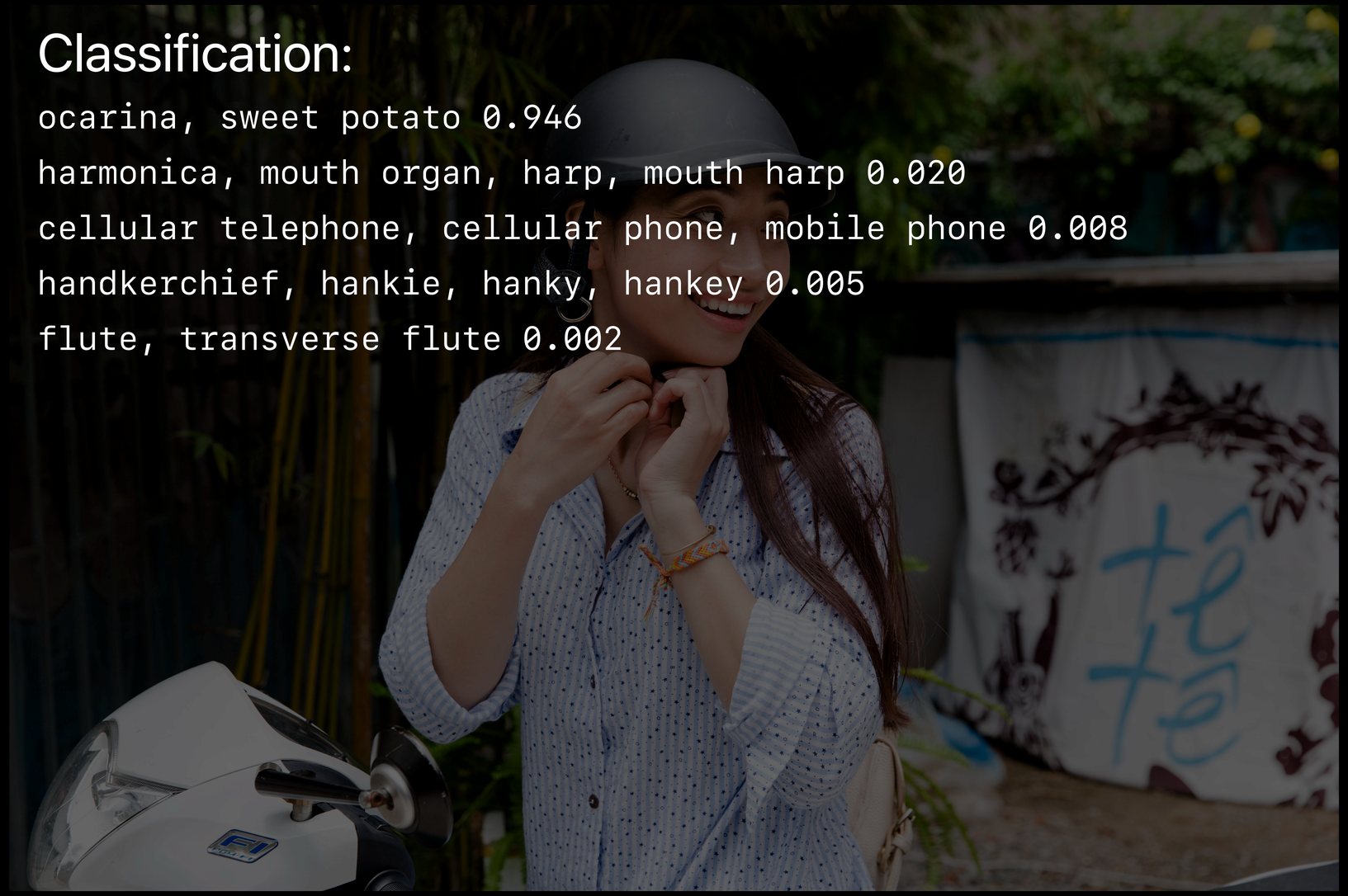
Classification Example



Classification Example

Classification Example

Classification Example



Classification Example

Classification Example

```swift
//
//  VisionObjectRecognitionViewController.swift
//  VisionCamObjectRecognition-Swift
//
//  Created by Frank Doepke on 5/24/17.
//  Copyright © 2018 Apple. All rights reserved.
//

import UIKit
import AVFoundation
import Vision

class VisionObjectRecognitionViewController: ViewController {

    private var detectionOverlay: CALayer! = nil

    // Vision parts
    private var requests = [VNRequest]()

    @discardableResult
    func setupVision() -> NSError? {
        // Setup Vision parts
        let error: NSError! = nil

        guard let modelURL = Bundle.main.url(forResource: "ObjectDetector", withExtension: "mlmodelc") else {
            return NSError(domain: "VisionObjectRecognitionViewController", code: -1, userInfo: [NSLocalizedDescriptionKey: "Model file is missing"])
        }
        do {
            let visionModel = try VNCoreMLModel(for: MLModel(contentsOf: modelURL))
            let objectRecognition = VNCoreMLRequest(model: visionModel, completionHandler: { (request, error) in
                DispatchQueue.main.async(execute: {
                    // perform all the UI updates on the main queue
                    if let results = request.results {
                        self.drawVisionRequestResults(results)
                    }
                })
            })
            self.requests = [objectRecognition]
        } catch let error as NSError {
            print("Model loading went wrong: \(error)")
        }

        return error
    }
```

Classification Example

Classification:
web site, website, internet site, site 0.994
monitor 0.002
screen, CRT screen 0.002
desktop computer 0.000
hand-held computer, hand-held microcomputer 0.000

```
//
//  VisionObjectRecognitionViewController.swift
//
//  Created by Frank Doepke on 5/24/17.
//  Copyright © 2018 Apple Inc. All rights reserved.
//

import AVFoundation
import Vision

class VisionObjectRecognitionViewController: ViewController {

    private var detectionOverlay: CALayer!

    // Vision parts
    private var requests = [VNRequest]()

    @discardableResult
    func setupVision() -> NSError? {
        // Setup Vision parts
        let error: NSError! = nil

        guard let modelURL = Bundle.main.url(forResource: "ObjectDetector", withExtension: "mlmodelc") else {
            return NSError(domain: "VisionObjectRecognitionViewController", code: -1, userInfo: [NSLocalizedDescriptionKey: "Model file is missing"])
        }
        do {
            let visionModel = try VNCoreMLModel(for: MLModel(contentsOf: modelURL))
            let objectRecognition = VNCoreMLRequest(model: visionModel, completionHandler: { (request, error) in
                DispatchQueue.main.async(execute: {
                    // perform all the UI updates on the main queue
                    if let results = request.results {
                        self.drawVisionRequestResults(results)
                    }
                })
            })
            self.requests = [objectRecognition]
        } catch let error as NSError {
            print("Model loading went wrong: \(error)")
        }

        return error
    }
```

Confidence Score Conclusions

Confidence Score Conclusions

Does 1.0 mean it is certainly correct?

- It fulfilled the criteria of the algorithm but our perception can differ

Confidence Score Conclusions

Does 1.0 mean it is certainly correct?

- It fulfilled the criteria of the algorithm but our perception can differ

Where the threshold is depends on the use case

- Labeling requires high confidence—observe how your classifier behaves
- Search might want to include lower confidence scores as they are probable

More Information

https://developer.apple.com/wwdc18/717

Vision Lab	Technology Lab 11	Friday 3:00PM


WWDC18