What’s New in Core ML
Part one

Michael Siracusa, Core ML
Francesco Rossi, Core ML
Bill March, Core ML
“Keep up the great work Cate!”
Jazz
Functionality Encoded in a Model
Learned From Data

Neural Network  Tree Ensemble
Linear Model  Support Vector Machine

Beach
Xcode

Machine Learning Model
- Name: SceneClassifier
- Type: Neural Network Classifier
- Size: 5.4 MB
- Description: Identify type of scene captured by an image
- Author: Johnny Appleseed
- License: MIT

Model Class
- SceneClassifier
  - Automatically generated Swift model class

Model Evaluation Parameters
- **Inputs**
  - Input: Image (Color 227 x 227)
  - Description: Input image to classify

- **Outputs**
  - sceneType: String
    - Description: Most likely type of scene captured in image
  - sceneTypeProbs: Dictionary (String → Double)
    - Description: Probability of each scene type
let model = SceneClassifier()
let prediction = try model.prediction(input: image)
return prediction.sceneType
let model = SceneClassifier()

let prediction = try model.prediction(input: image)

return prediction.sceneType
let model = SceneClassifier()

let prediction = try model.prediction(input: image)

return prediction.sceneType
let model = SceneClassifier()

let prediction = try model.prediction(input: image)

return prediction.sceneType
Integrated Vision

VNCoreMLRequest

Natural Language

NLModel
What’s New?
What's New?
Part One

Model Size
Part One

Model Size

Performance
Part One

Model Size
Performance
Customization
Model Size
Model Size
Performance
Customization
Models on Device
Models on Device

Privacy
Models on Device

Privacy

Speed
Models on Device

Privacy  Speed  No Server
Models on Device

- Privacy
- Speed
- No Server
- Available
Models on Device

Privacy  Speed  No Server  Available

Need to store machine learning models on device
Awesomeness
Awesomeness 😄
More Awesomeness
More Awesomeness

App Size: +100 MB
More Awesomeness

App Size: +100 MB
Even More Awesomeness

App Size: +250 MB
Download on Demand
Compile on device

More Awesomeness
Initial Install Size: Unchanged
Download on Demand

Compile on device

More Awesomeness

Initial Install Size: Unchanged
On demand: +250 MB
Download on Demand
Compile on device

More Awesomeness

Initial Install Size: **Unchanged**
On demand: +250 MB
Reducing Model Size
Reducing Model Size

Smaller bundle
Reducing Model Size

Smaller bundle

Smaller/faster downloads
Reducing Model Size

Smaller bundle

Smaller/faster downloads

Reduced runtime memory usage
Core ML App Size

Factors
Core ML App Size
Factors

Number of Models
Core ML App Size
Factors

Number of Models $\times$ Number of Weights
Core ML App Size
Factors

Number of Models \times Number of Weights \times Size of Weight
Core ML App Size
Factors

Number of Models \times Number of Weights \times Size of Weight
Options for Neural Networks
Options for Neural Networks

- Float
- 32-bit

iOS 11
Options for Neural Networks

- iOS 11: Float 32-bit
- iOS 11.2: Half 16-bit
Options for Neural Networks

- **iOS 11**: Float 32-bit
- **iOS 11.2**: Half 16-bit
- **iOS 12**: Quantized (NEW)
Weight Quantization

Billions of possible values

Min Subset of Weights Max

32-bit float
Weight Quantization

Billions of possible values

Subset of Weights

Min

32-bit float

Max

256 values
Weight Quantization

Billions of possible values

Subset of Weights

Min

Max

32-bit float

8-bit index

256 values
Weight Quantization

Min → Subset of Weights → Max

32-bit float
Weight Quantization

Min

Subset of Weights

Max

32-bit float

8 values
Weight Quantization

- Min
- Subset of Weights
- Max

32-bit float

8 values
3-bit index
Weight Quantization

- Linear
- Uniform distribution

Subset of Weights
- 8 values
- 3-bit index
- 32-bit float

Min

Max

Uniform distribution
Weight Quantization

Min Max Subset of Weights

32-bit float

8 values

3-bit index
Weight Quantization

Lookup Table
Non-uniform distribution

Min

Subset of Weights

Max

32-bit float

8 values
3-bit index
Resnet50
8 bit Quantization

25 Million Weights
32-bit Float
103 MB

25 Million Weights
8-bit UInt8
26 MB
Resnet50
< 8 bit quantization

25 Million Weights
32-bit Float
103 MB

25 Million Weights
8-bit UInt8
26 MB

25 Million Weights
4-bit UInt4
14 MB
Resnet50
< 8 bit quantization

25 Million Weights
32-bit Float
103 MB

25 Million Weights
8-bit UInt8
26 MB

25 Million Weights
4-bit UInt4
14 MB

All the way down to 1-bit
Obtaining Quantized Models
Obtaining Quantized Models

Post-training quantization

32-bit → Core ML Tools → Quantized
Obtaining Quantized Models

Post-training quantization

Train quantized
- From scratch or re-training
- Then convert quantized models to Core ML
Accuracy Tradeoff

Model dependent

Use case dependent

Active area of research
Accuracy Tradeoff

Model dependent

Use case dependent

Active area of research

Check Your Quantized Model Accuracy
Accuracy Tradeoff

Model dependent

Use case dependent

Active area of research

Check Your Quantized Model Accuracy

With test data and metric relevant for your app
Demo
Demo Summary

4 Models
Demo Summary

4 Models

6.7 MB each
32-bit

26.8 MB
Demo Summary

4 Models

- 6.7 MB each 32-bit
- 857 KB each 4-bit

Total:
- 26.8 MB
- 3.4 MB
Factors

- Number of Models
- Number of Weights
- Size of Weight
Factors

Number of Models \times \text{Number of Weights} \times \text{Size of Weight}
Number of Models

Function A

Function B

Function C
Number of Models

Function A

Function B and C
Number of Models

Function A

Function B and C

Multi-task models
Example: See Turi Create session
Number of Models

Function A

Function B and C

- Multi-task models
  Example: See Turi Create session
- Flexible shapes and sizes
Number of Models

Function A

Function B and C
- Multi-task models
  - Example: See Turi Create session
- Flexible shapes and sizes
Style Transfer Model

Input size defined by the model

GlassSD.mlmodel

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>image</td>
<td>Image (Color 640 x 480)</td>
<td>Input image to stylize</td>
</tr>
<tr>
<td>stylized</td>
<td>Image (Color 640 x 480)</td>
<td>Stylized image</td>
</tr>
</tbody>
</table>

Same style for different sizes??
Input Size Mismatch

1920 x 1440

Glass.mlmodel
640 x 480
Input Size Mismatch

1920 x 1440

Glass.mlmodel
640 x 480
Input Size Mismatch

1920 x 1440

Glass.mlmodel
640 x 480
Input Size Mismatch

1920 x 1440

Glass.mlmodel
640 x 480
Model for High Resolution Input
Model for High Resolution Input

1920 x 1440

Glass.mlmodel
640 x 480

GlassHD.mlmodel
1920 x 1440
Model Per Resolution

640 x 480
Glass.mlmodel

1920 x 1440
GlassHD.mlmodel
Model Per Resolution

640 x 480
Glass.mlmodel

1920 x 1440
GlassHD.mlmodel

2x Size
One Flexible Model

640 x 480 or 1920 x 1440

GlassFlexible.mlmodel
## Combine Using Flexible Image Sizes

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Flexibility</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>▼ Inputs</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>image</td>
<td>Image (Color 640 x 480)</td>
<td>640 x 480</td>
<td>1920 x 1440</td>
</tr>
<tr>
<td>▼ Outputs</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>stylized</td>
<td>Image (Color 640 x 480)</td>
<td>640 x 480</td>
<td>1920 x 1440</td>
</tr>
</tbody>
</table>

GlassFlexible.mlmodel
## Combine Using Flexible Image Sizes

**GlassFlexible.mlmodel**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Inputs</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>image</td>
<td>Image (Color 640 x 480)</td>
<td>Input image to stylize</td>
</tr>
<tr>
<td><strong>Outputs</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>stylized</td>
<td>Image (Color 640 x 480)</td>
<td>Stylized image</td>
</tr>
</tbody>
</table>

**Flexibility**

<table>
<thead>
<tr>
<th>640 x 480</th>
<th>1920 x 1440</th>
</tr>
</thead>
</table>
Combine Using Flexible Image Sizes

GlassFlexible.mlmodel

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inputs</td>
<td>image</td>
<td>Input image to stylize</td>
</tr>
<tr>
<td></td>
<td>640 x 480</td>
<td>1920 x 1440</td>
</tr>
<tr>
<td>Outputs</td>
<td>stylized</td>
<td>Stylized image</td>
</tr>
<tr>
<td></td>
<td>640 x 480</td>
<td>1920 x 1440</td>
</tr>
</tbody>
</table>

One model
No redundant code
Faster model switching times
Flexibility Options
Flexibility Options

Size Range Per Dimension

• Flexibility
Flexibility Options

Size Range Per Dimension
• Flexibility

Enumerated
• Better performance
• Smaller testing surface
Which Models are Flexible?
Which Models are Flexible?

Fully Convolutional Neural Networks
• Image processing
• Object detection

Core ML Tools can check for you!
Model Size

Number of Models \times Number of Weights \times Size of Weight

Flexible Sizes \times Quantization
Model Size

Number of Models \times Number of Weights \times Size of Weight

Flexible Sizes \times Broad ML Support \times Quantization
Performance and Customization
Model Size
Performance
Customization
// Loop over inputs

for i in 0..<modelInputs.count {
    modelOutputs[i] = model.prediction(from: modelInputs[i], options: options)
}

// Loop over inputs

for i in 0..<modelInputs.count {
    modelOutputs[i] = model.prediction(from: modelInputs[i], options: options)
}
modelOutputs = model.predictions(from: modelInputs, options: options)
Model Size
Performance
Customization
Any Horses?
Any Horses?

~2007

Hope computers can answer some day, but for now—just ask someone.
Any Horses?

~2012
Wow—Computers can answer this!
Cutting-edge deep learning research is exciting.
Any Horses?

2018
Train a model with Create ML—the UI is great!
There ARE horses!
Custom Layers

There ARE horses!
There ARE horses!
@objc(AAPLCustomHorseLayer)
class AAPLCustomHorseLayer: MLCustomLayer {
    ...
}

Custom Layers
iOS 11.2
public protocol MLCustomLayer {

    public init(parameters: [String : Any]) throws

    public func setWeightData(_ weights: [Data]) throws

    public func outputShapes(forInputShapes: [[NSNumber]]) throws -> [[NSNumber]]

    public func evaluate(inputs: [MLMultiArray], outputs: [MLMultiArray]) throws

}
public protocol MLCustomLayer {

    public init(parameters: [String : Any]) throws

    public func setWeightData(_ weights: [Data]) throws

    public func outputShapes(forInputShapes: [[NSNumber]]) throws -> [[NSNumber]]

    public func evaluate(inputs: [MLMultiArray], outputs: [MLMultiArray]) throws

}
public protocol MLCustomLayer {

    public init(parameters: [String : Any]) throws

    public func setWeightData(_ weights: [Data]) throws

    public func outputShapes(forInputShapes: [[NSNumber]]) throws -> [[NSNumber]]

    public func evaluate(inputs: [MLMultiArray], outputs: [MLMultiArray]) throws

}
public protocol MLCustomLayer {

    public init(parameters: [String : Any]) throws

    public func setWeightData(_ weights: [Data]) throws

    public func outputShapes(forInputShapes: [[NSNumber]]) throws -> [[NSNumber]]

    public func evaluate(inputs: [MLMultiArray], outputs: [MLMultiArray]) throws
}
public protocol MLCustomLayer {

    public init(parameters: [String : Any]) throws

    public func setWeightData(_ weights: [Data]) throws

    public func outputShapes(forInputShapes: [[NSNumber]]) throws -> [[NSNumber]]

    public func evaluate(inputs: [MLMultiArray], outputs: [MLMultiArray]) throws
}

// Optional method for performance

optional public func encode(commandBuffer: MTLCommandBuffer,
    inputs: [MTLTexture],
    outputs: [MTLTexture]) throws
Custom Layers

Neural Network Only

MultiArray Inputs → MultiArray Outputs
"Isn’t my ring closed yet?"

"Not yet, but you can do it!"
Custom Models

@objc(AAPLCustomModel)
class AAPLCustomModel: MLCustomModel {
    ...
}

MLMODEL

SWIFT
public protocol MLCustomModel {

    public init(modelDescription: MLModelDescription, parameters: [String : Any]) throws

    public func prediction(from: MLFeatureProvider, options: MLPredictionOptions) throws -> MLFeatureProvider

    optional public func predictions(from: MLBatchProvider, options: MLPredictionOptions) throws -> MLBatchProvider

}
public protocol MLCustomModel {

public init(modelDescription: MLModelDescription, parameters: [String : Any]) throws

public func prediction(from: MLFeatureProvider, options: MLPredictionOptions) throws -> MLFeatureProvider

optional public func predictions(from: MLBatchProvider, options: MLPredictionOptions) throws -> MLBatchProvider

}
public protocol MLCustomModel {

    public init(modelDescription: MLModelDescription, parameters: [String : Any]) throws

    public func prediction(from: MLFeatureProvider,
                             options: MLPredictionOptions) throws -> MLFeatureProvider

    optional public func predictions(from: MLBatchProvider,
                                      options: MLPredictionOptions) throws -> MLBatchProvider

}
public protocol MLCustomModel {

    public init(modelDescription: MLModelDescription, parameters: [String : Any]) throws

    public func prediction(from: MLFeatureProvider, options: MLPredictionOptions) throws -> MLFeatureProvider

    optional public func predictions(from: MLBatchProvider, options: MLPredictionOptions) throws -> MLBatchProvider

}
Customization in Xcode

Machine Learning Model
- Name: SuperCustomized
- Type: Pipeline
- Size: 637 bytes
- Author: Jane Appleseed
- Description: A model containing both a custom layer and a custom model.
- License: MIT

Model Class

Model Evaluation Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Flexibility</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input</td>
<td>Image (Color 840 x 480)</td>
<td>640 x 480</td>
<td>480 x 640</td>
</tr>
</tbody>
</table>

Outputs
- output | Image (Color 840 x 480)   | 640 x 480 | 480 x 640 | Resulting output image |

Dependencies

- Custom Layers
  - AAIPICustomLayer: A custom neural network layer.
- Custom Models
  - AAIPICustomModel: A custom post-processing step expressed as an MLCustomModel.
Customization in Xcode

- **Custom Layers**
  - AAPLCustomLayer: A custom neural network layer.

- **Custom Models**
  - AAPLCustomModel: A custom post-processing step expressed as an MLCustomModel.
Customization

Options

Custom Layers

MultiArrays → MultiArrays

Custom Models

Any → Custom Models → Any
And more...
Core ML 2
Smaller. Faster. Customizable.
More Information

https://developer.apple.com/wwdc18/708