Introducing Core ML

Gaurav Kapoor, Core ML
Michael Siracusa, Core ML
Lizi Ottens, Core ML
Looking forward to seeing you tonight.

What do you have planned?

Do you want to go for dinner or a movie?

Hello

Hi

Scribble
Why?
Task
Show all images of roses
Task
Show all images of roses

// Use color
if color == "reddish"
Task
Show all images of roses

// Use color
if color == "reddish"

// Use shape
if shape == ???
Task
Show all images of roses

Machine Learning
Training

Offline

+ Labels
Training

Offline

+ Labels

Learning Algorithm
Training

Offline

+ Labels

Learning Algorithm

Model
Inference

Model

Label: Rose
Confidence: 95%
Challenging!
void convolutionLayer(int kernelWidth,
    int kernelHeight,
    int inputFeatureChannels,
    int outputFeatureChannels,
    int strideX,       // cropped from strideX)
    int strideY,       // cropped from strideY)
    int numRows,
    int numCols,
    float* input,
    float* output,
    float* weights,
    int widthPadding,
    int heightPadding,
    float alpha,
    float beta) {

    memset(output, 0, ((numRows - kernelWidth + 2*widthPadding)/2 + 1) * ((numRows - kernelWidth + 2*widthPadding)/2 + 1)*outputFeatureChannels * sizeof(float));

    memset(input, 0, inputFeatureChannels * numRows * numCols * sizeof(float));

    for (int depthInd = 0; depthInd < outputFeatureChannels; depthInd++) {
        // loop over input (color) channels
        for (int colorInd = 0; colorInd < inputFeatureChannels; colorInd++) {
            int numRowsOut = (numRows - kernelWidth + 2*widthPadding)/strideX + 1;
            int numColsOut = (numCols - kernelHeight + 2*heightPadding)/strideY + 1;

            int color = 0;
            // loop over the pixels of the image
            for (int i=0; i < numRowsOut; i++) {
                for(int j=0; j < numColsOut; j++) {

                    int m, n;

                    // loop over this kernel
                    for(int mm=0; m < kernelWidth; m++) {
                        int nn = kernelHeight - 1 - n;
                        int ii = i + (m - kernelWidth/2);
                        int jj = j + (n - kernelHeight/2);

                        if(ii >= 0 && ii < numRows && jj >= 0 && jj < numCols)
                            float weight = weights[nn + mm * kernelHeight + depthInd * kernelHeight * kernelWidth];
                            float value =   input[jj + ii*numCols + colorInd*numCols*inputFeatureChannels] * weight;
                            output[j + i*numColsOut + depthInd*numColsOut*numColsOut] += value;
                    }
                }
            }

            // loop and apply nonlinearity
            for (int i = 0; i < ((numRows - kernelWidth + 2*widthPadding)/strideX + 1) * ((numRows - kernelWidth + 2*widthPadding)/strideY + 1) * outputFeatureChannels;
                i++) {
                output[i] = alpha*tanh(beta*output[i]);
            }
        }
    }
}
Challenges

Correctness
Performance
Energy Efficiency
ML Frameworks
ML Frameworks

Your app
ML Frameworks

Your app

Domain Specific Frameworks

NEW
Vision
NLP
ML Frameworks

Your app

Domain Specific Frameworks
- Vision
- NLP

ML Framework
- Core ML

NEW
ML Frameworks

Domain Specific Frameworks
- NEW Vision
- NEW NLP

ML Framework
- Core ML

ML Performance Primitives
- Accelerate
- MPS

Your app
Vision Framework

Accelerate

MPS

Core ML

NLP

Your app
Vision Framework

<table>
<thead>
<tr>
<th>Your app</th>
<th>MPS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Vision</td>
<td>NLP</td>
</tr>
<tr>
<td>Core ML</td>
<td></td>
</tr>
<tr>
<td>Accelerate</td>
<td>MPS</td>
</tr>
</tbody>
</table>

Object Tracking
Vision Framework

Object Tracking

Face Detection
Natural Language Processing
Pablo y yo ya regresamos de nuestras vacaciones en Finlandia.

Language: Spanish
Natural Language Processing

Language Identification

Pablo y yo ya regresamos de nuestras vacaciones en Finlandia.

Language: Spanish

Place Identified: Finland

Named Entity Recognition
Core ML
Core ML

Music Tagging
Core ML

Music Tagging

Image Captioning

Dog playing in a soccer field
Accelerate and MPS

High performance math

Inference for custom ML models
Run on Device
Run on Device

User Privacy
Run on Device

User Privacy

Data Cost
Run on Device

- User Privacy
- Data Cost
- Server Cost
Run on Device

User Privacy

Data Cost

Server Cost

Always Available

24/7
ML Frameworks

Domain Specific Frameworks
- Vision
- NLP

ML Framework
- Core ML

ML Performance Primitives
- Accelerate
- MPS

Your app
Core ML

Michael Siracusa, Core ML
Overview
Overview
Models
Overview
Models
Development Flow
Overview

Models

Development Flow
Focus on the experience you are trying to enable
Simple
Simple

Unified inference API
Xcode integration
Performant

Fine tuned inference engines
Built onAccelerate and Metal
Compatible

Public model format
Support for popular training libraries
Overview
Models
Development Flow
Model

Function learned from data

Observed inputs

Predicts outputs

photo: flowerType:
Underlying Function

- Sentiment Analysis
- Handwriting Recognition
- Translation
- Scene Classification
- Style Transfer
- Music Tagging
- Predicting Text
Underlying Function

Sentiment Analysis
That was totally awesome Leo!

Handwriting Recognition
7 → 7

Translation
I love you mom → 사랑해 엄마

Scene Classification
Beach

Style Transfer

Music Tagging
Rock

Predicting Text
Do you know the way to San Jose
<table>
<thead>
<tr>
<th>Model Types</th>
<th>Sentiment Analysis</th>
<th>Handwriting Recognition</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Translation</td>
<td>Scene Classification</td>
</tr>
<tr>
<td></td>
<td>Music Tagging</td>
<td>Style Transfer</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Predicting Text</td>
</tr>
</tbody>
</table>
Focus on Use Cases

- Sentiment Analysis
- Translation
- Scene Classification
- Music Tagging

- Handwriting Recognition
- Style Transfer
- Predicting Text
Core ML Model

Single document

Public format
Where do models come from?
Sample Models

Core ML models

Ready to use

Task specific

Explore!

Places205-GoogLeNet
Detects the scene of an image from 205 categories such as an airport terminal, bedroom, forest, coast, and more.

View original model details

Download Core ML Model
File size: 24.8 MB

ResNet50
Detects the dominant objects present in an image from a set of 1000 categories such as trees, animals, food, vehicles, people, and more.

View original model details

Download Core ML Model
File size: 102.6 MB
Tap Into ML Community

Thriving communities
Popular ML libraries
Many models
Convert to Core ML

Core ML Tools

£python£
Convert to Core ML

Caffe  Keras
dxlc  XGBoost  scikit-learn  turi  LIBSVM

Core ML Tools

python
Convert to Core ML

Caffe  K  Keras

dxlc  XGBoost  scikit

turi  LIBSVM

Core ML Tools

don

python

MLMODEL
Convert to Core ML

Caffe  Keras  Core ML Tools  Open Source

dmlc XGBoost  scikit learn  python  ML MODEL
Convert to Core ML

Caffe → Keras → Core ML Tools → Open Source → MLMODEL

CoreML in Depth
Hall 3
Thursday 9:00AM
Overview
Models
Development Flow
Model as Code
Model as Code
Model as Code
Model as Code

Xcode

MLMODEL

SWIFT

Your App

exec
Model as Code
Development Flow

Lizi Ottens, Core ML
Getting the Model
Getting the Model
Getting the Model

Caffe + Keras + XGBoost + turi + LIBSVM
Getting the Model

+ Caffe
  + dm/nc
  + XGBoost
  + turi

Keras

Convert

LIBSVM

MLMODEL
Demo

Image based flower identifier
Demo Recap

Image based flower identifier
Demo Recap

Xcode integration

**Machine Learning Model**
- Name: FlowerClassifier
- Type: Neural Network Classifier
- Size: 41.6 MB
- Author: Lizi Ottens
- License: MIT
- Description: Identify the type of flower present in an image.

**Model Class**
- FlowerClassifier (Swift generated source)

**Model Evaluation Parameters**

<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>flowerImage</td>
<td>Image&lt;RGB,227,227&gt;</td>
<td>Input image of a flower</td>
</tr>
<tr>
<td><strong>outputs</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>flowerType</td>
<td>String</td>
<td>Most likely flower type in image</td>
</tr>
<tr>
<td>flowerTypeProbs</td>
<td>Dictionary&lt;String,Double&gt;</td>
<td>Probability of each flower type</td>
</tr>
</tbody>
</table>
Demo Recap

Xcode integration

Machine Learning Model

- Name: FlowerClassifier
- Type: Neural Network Classifier
- Size: 41.6 MB
- Author: Lizi Ottens
- License: MIT
- Description: Identify the type of flower present in an image.

Model Class

- FlowerClassifier (Swift generated source)

Model Evaluation Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>flowerImage</td>
<td>Image&lt;RGB,227,227&gt;</td>
<td>Input image of a flower</td>
</tr>
<tr>
<td>flowerType</td>
<td>String</td>
<td>Most likely flower type in image</td>
</tr>
<tr>
<td>flowerTypeProbs</td>
<td>Dictionary&lt;String,Double&gt;</td>
<td>Probability of each flower type</td>
</tr>
</tbody>
</table>
## Demo Recap

### Xcode integration

### Machine Learning Model

- **Name**: FlowerClassifier
- **Type**: Neural Network Classifier
- **Size**: 41.6 MB
- **Author**: Lizi Ottens
- **License**: MIT
- **Description**: Identify the type of flower present in an image.

### Model Class

- **FlowerClassifier (Swift generated source)**

### Model Evaluation Parameters

<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>flowerImage</td>
<td>Image&lt;RGB,227,227&gt;</td>
<td>Input image of a flower</td>
</tr>
<tr>
<td><strong>outputs</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>flowerType</td>
<td>String</td>
<td>Most likely flower type in image</td>
</tr>
<tr>
<td>flowerTypeProbs</td>
<td>Dictionary&lt;String,Double&gt;</td>
<td>Probability of each flower type</td>
</tr>
</tbody>
</table>
Demo Recap

Xcode integration

Machine Learning Model
- Name: FlowerClassifier
- Type: Neural Network Classifier
- Size: 41.6 MB
- Author: Lizi Ottens
- License: MIT
- Description: Identify the type of flower present in an image.

Model Class
- FlowerClassifier (Swift generated source)

Model Evaluation Parameters
- Name: inputs
  - flowerImage: Image<RGB,227,227>: Input image of a flower
- Name: outputs
  - flowerType: String: Most likely flower type in image
  - flowerTypeProbs: Dictionary<String,Double>: Probability of each flower type
Demo Recap
Simple usage

```swift
let flowerModel = FlowerClassifier()
if let prediction = try? flowerModel.prediction(flowerImage: image) {
    return prediction.flowerType
}
```
Demo Recap

Simple usage

```swift
let flowerModel = FlowerClassifier()
if let prediction = try? flowerModel.prediction(flowerImage: image) {
    return prediction.flowerType
}
```

Type of model abstracted
Demo Recap
Simple usage

```swift
let flowerModel = FlowerClassifier()
if let prediction = try? flowerModel.prediction(flowerImage: image) {
    return prediction.flowerType
}
```

Type of model abstracted

Input/output strongly typed
class FlowerClassifierInput {
    var flowerImage: CVPixelBuffer
}

class FlowerClassifierOutput {
    let flowerType: String
    let flowerTypeProbs: [String: Double]
}

class FlowerClassifier {
    convenience init()
    func prediction(flowerImage: CVPixelBuffer) throws -> FlowerClassifierOutput
}
class FlowerClassifierInput {
    var flowerImage: CVPixelBuffer
}

class FlowerClassifierOutput {
    let flowerType: String
    let flowerTypeProbs: [String: Double]
}

class FlowerClassifier {
    convenience init()
    func prediction(flowerImage: CVPixelBuffer) throws -> FlowerClassifierOutput
}
class FlowerClassifierInput {
    var flowerImage: CVPixelBuffer
}

class FlowerClassifierOutput {
    let flowerType: String
    let flowerTypeProbs: [String: Double]
}

class FlowerClassifier {
    convenience init()
    func prediction(flowerImage: CVPixelBuffer) throws -> FlowerClassifierOutput
}
class FlowerClassifierInput {
    var flowerImage: CVPixelBuffer
}

class FlowerClassifierOutput {
    let flowerType: String
    let flowerTypeProbs: [String: Double]
}

class FlowerClassifier {
    convenience init()
    func prediction(flowerImage: CVPixelBuffer) throws -> FlowerClassifierOutput
}
class FlowerClassifier {
    convenience init()
    func prediction(flowerImage: CVPixelBuffer) throws -> FlowerClassifierOutput
}
class FlowerClassifier {
    convenience init()

    func prediction(flowerImage: CVPixelBuffer) throws -> FlowerClassifierOutput
}

More Advanced
Underlying API
class FlowerClassifier {
    convenience init()

    let model: MLMModel

    func prediction(flowerImage: CVPixelBuffer) throws -> FlowerClassifierOutput
}

Programmatic access to model for power users
class MLModel {
    var modelDescription: MLModelDescription
    func prediction(from input: MLFeatureProvider) throws -> MLFeatureProvider
}

MLModel
```swift
class MLModel {
    var modelDescription: MLModelDescription

    func prediction(from input: MLFeatureProvider) throws -> MLFeatureProvider
}
```

Access to model description
MLModel

class MLModel {
    var modelDescription: MLModelDescription
    func prediction(from input: MLFeatureProvider) throws -> MLFeatureProvider
}

Access to model description
Flexibility in how input is provided
Behind the Scenes
Model compilation
Behind the Scenes
Model compilation
Behind the Scenes
Model compilation

Quick initialization
Behind the Scenes

Model compilation

Quick initialization

Optimized prediction
Model Goals
Model Goals

Reduce size
Model Goals

Reduce size

Improve accuracy
Model Goals

Reduce size
Improve accuracy
Decrease prediction times
Summary
Summary

Machine learning frameworks
Summary

Machine learning frameworks
Core ML
Summary

Machine learning frameworks

Core ML

Development flow in action
More Information

https://developer.apple.com/wwdc17/703
<table>
<thead>
<tr>
<th>Session</th>
<th>Hall</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>Natural Language Processing and your Apps</td>
<td>Hall 3</td>
<td>Wednesday 9:00AM</td>
</tr>
<tr>
<td>Vision Framework: Building on Core ML</td>
<td>Hall 2</td>
<td>Wednesday 3:10PM</td>
</tr>
<tr>
<td>Core ML in depth</td>
<td>Hall 3</td>
<td>Thursday 9:00AM</td>
</tr>
<tr>
<td>Accelerate and Sparse Solvers</td>
<td>Executive Ballroom</td>
<td>Thursday 10:00AM</td>
</tr>
<tr>
<td>Using Metal 2 for Compute</td>
<td>Grand Ballroom A</td>
<td>Thursday 4:10PM</td>
</tr>
<tr>
<td>Lab Name</td>
<td>Location</td>
<td>Time</td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td>----------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>Core ML and Natural Language Processing Lab</td>
<td>Technology Lab D</td>
<td>Thu 11:00AM-3:30PM</td>
</tr>
<tr>
<td>Core ML &amp; Natural Language Processing Lab</td>
<td>Technology Lab D</td>
<td>Fri 1:50AM-4:00PM</td>
</tr>
</tbody>
</table>