Drawing Classification and One-Shot Object Detection in Turi Create

Sam Youtsey, EPM
Shantanu Chhabra, Engineer
Abhishek Pratapa, Engineer
Create Core ML models for your intelligent applications.
Task → Data → Model → Evaluate → Deploy
import turicreate

// Load data
data = turicreate.SFrame("ImageData.sframe")
train, test = data.random_split(0.8)

// Create a model
model = turicreate.object_detector.create(train, "labels")

// Evaluate the model
metrics = model.evaluate(test)

// Export for deployment
model.export_coreml("MyDetector.mlmodel")
import turicreate

data = turicreate.SFrame("ImageData.sframe")
train, test = data.random_split(0.8)

model = turicreate.object_detector.create(train, "labels")

metrics = model.evaluate(test)

model.export_coreml("MyDetector.mlmodel")
import turicreate

// Load data
data = turicreate.SFrame("ImageData.sframe")
train, test = data.random_split(0.8)

// Create a model
model = turicreate.object_detector.create(train, "labels")

// Evaluate the model
metrics = model.evaluate(test)

// Export for deployment
model.export_coreml("MyDetector.mlmodel")
import turicreate

// Load data
data = turicreate.SFrame("ImageData.sframe")
train, test = data.random_split(0.8)

// Create a model
model = turicreate.object_detector.create(train, "labels")

// Evaluate the model
metrics = model.evaluate(test)

// Export for deployment
model.export_coreml("MyDetector.mlmodel")
import turicreate

// Load data
data = turicreate.SFrame("ImageData.sframe")
train, test = data.random_split(0.8)

// Create a model
model = turicreate.object_detector.create(train, "labels")

// Evaluate the model
metrics = model.evaluate(test)

// Export for deployment
model.export_coreml("MyDetector.mlmodel")
import turicreate

// Load data
data = turicreate.SFrame("ImageData.sframe")
train, test = data.random_split(0.8)

// Create a model
model = turicreate.object_detector.create(train, "labels")

// Evaluate the model
metrics = model.evaluate(test)

// Export for deployment
model.export_coreml("MyDetector.mlmodel")
import turicreate

// Load data
data = turicreate.SFrame("AudioData.sframe")
train, test = data.random_split(0.8)

// Create a model
model = turicreate.sound_classifier.create(train, "labels")

// Evaluate the model
metrics = model.evaluate(test)

// Export for deployment
model.export_coreml("MySoundClassifier.mlmodel")
import turicreate

// Load data
data = turicreate.SFrame("ActivityData.sframe")
train, test = data.random_split(0.8)

// Create a model
model = turicreate.activity_classifier.create(train, "labels")

// Evaluate the model
metrics = model.evaluate(test)

// Export for deployment
model.export_coreml("MyActivityClassifier.mlmodel")
New Tasks
One-Shot Object Detection
One-Shot Object Detection

Drawing Classification
One-Shot Object Detection
You have a Straight Flush!
Your cards are Ace of Diamonds, King of Diamonds, and Queen of Diamonds
The probability of a Straight Flush is 0.22%
Founded in 1976

Headquartered in
Cupertino, California
You have a Straight Flush!
Your cards are Ace of Diamonds, King of Diamonds, and Queen of Diamonds
The probability of a Straight Flush is 0.22%
Original Approach
Original Approach
Original Approach

Ace of hearts
Original Approach
Original Approach
Original Approach
Original Approach
<table>
<thead>
<tr>
<th>Class</th>
<th>Image</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ace-Clubs</td>
<td><img src="image1.png" alt="Ace-Clubs Card" /></td>
</tr>
<tr>
<td>Nine-Diamonds</td>
<td><img src="image2.png" alt="Nine-Diamonds Card" /></td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>
Synthetic Data Augmentation
Synthetic Data Augmentation
## Comparison

<table>
<thead>
<tr>
<th>One-Shot Object Detection</th>
<th>Traditional Object Detection</th>
</tr>
</thead>
<tbody>
<tr>
<td>2D objects only</td>
<td>2D or 3D objects</td>
</tr>
<tr>
<td>Little data required</td>
<td>Data-intensive</td>
</tr>
<tr>
<td>No annotation needed</td>
<td>Annotation required</td>
</tr>
<tr>
<td>Objects with “regularity”</td>
<td>Any object</td>
</tr>
</tbody>
</table>
import turicreate

// Load data
train = turicreate.SFrame("ImageData.sframe")

// Create a model
model = turicreate.one_shot_object_detector.create(train, "labels")

// Evaluate the model
test = turicreate.SFrame("TestData.sframe")
predictions = model.predict(test)

// Export for deployment
model.export_coreml("MyDetector.mlmodel")
Demo

Three card poker

Shantanu Chhabra, Engineer
Drawing Classification
MATH 101
Final Exam
February 14, 2019
9 am - 10 am

• This exam has five questions.
• Each question is worth 10 points each.
• You have 60 minutes to attempt this exam.
• This is an open book exam.
• Grading will be on a curve. Partial Credit will be awarded at the grader's discretion.

1. $1 + 1 = \boxed{2}$ 
2. $2 + 2 = \boxed{4}$
3. $3 + 2 = \boxed{5}$
4. $4 + 5 = \boxed{10}$
5. $\frac{3^2}{3} = \boxed{3}$

$\int \frac{1}{x} + \log x \, dx = 2x$
Bitmap Drawings
Bitmap Drawings
drawing_stroke = [ ]
drawing_stroke = [
    {"x": 1.0, "y": 1.5}, {"x": 3.0, "y": 4.5}, {"x": 5.5, "y": 8.5}, {"x": 10, "y": 4.0} ...
]
Model Creation

// Create a model
model = turicreate.drawing_classifier.create(train, "labels")
Model Creation

// Create a model
model = turicreate.drawing_classifier.create(train, "labels")

- State of the art
- Model less than 500 kB on-device
- GPU acceleration
More accurate models

Train on as few as 30 drawings per class
import turicreate

// Load data
data = turicreate.SFrame("drawing_data.sframe")
train, test = data.random_split(0.8)

// Create a model
model = turicreate.drawing_classifier.create(train, "labels")

// Evaluate the model
metrics = model.evaluate(test)

// Export for deployment
model.export_coreml("MyModel.mlmodel")
Demo
Grading app
let image = canvas.drawing.image(in: canvas.frame, scale: 1.0)

let cropped = image.cgImage!.cropping(to: canvas.drawing.bounds)

let model = try! VNCoreMLModel(for: DrawingClassifier().model)
let request = VNCoreMLRequest(model: model, completionHandler: { (request, error) in
    if let _ = error {
        return
    } else {
        self.processClassifications(for: request)
    }
})
let image = canvas.drawing.image(in: canvas.frame, scale: 1.0)

let cropped = image.cgImage!.cropping(to: canvas.drawing.bounds)

let model = try! VNCoreMLModel(for: DrawingClassifier().model)
let request = VNCoreMLRequest(model: model, completionHandler: { (request, error) in
    if let _ = error {
        return
    } else {
        self.processClassifications(for: request)
    }
})
let image = canvas.drawing.image(in: canvas.frame, scale: 1.0)

let cropped = image.cgImage!.cropping(to: canvas.drawing.bounds)

let model = try! VNCoreMLModel(for: DrawingClassifier().model)
let request = VNCoreMLRequest(model: model, completionHandler: { (request, error) in
  if let _ = error {
    return
  } else {
    self.processClassifications(for: request)
  }
})
let image = canvas.drawing.image(in: canvas.frame, scale: 1.0)

let cropped = image.cgImage!.cropping(to: canvas.drawing.bounds)

let model = try! VNCoreMLModel(for: DrawingClassifier().model)
let request = VNCoreMLRequest(model: model, completionHandler: { (request, error) in
    if let _ = error {
        return
    } else {
        self.processClassifications(for: request)
    }
})
Demo Recap
Demo Recap

Loaded drawings and annotations
Demo Recap

Loaded drawings and annotations

Interactively explored data
Demo Recap

Loaded drawings and annotations

Interactively explored data

Trained a model
Demo Recap

Loaded drawings and annotations

Interactively explored data

Trained a model

Exported to Core ML
Demo Recap

Loaded drawings and annotations
Interactively explored data
Trained a model
Exported to Core ML
Deployed using PencilKit, Vision
Interactive Model Evaluation

Abhishek Pratapa, Engineer
import turicreate

// Load data
data = turicreate.SFrame("data.sframe")
train, test = data.random_split(0.8)

// Create a model
model = turicreate.drawing_classifier.create(train, "labels")

// Evaluate the model
metrics = model.evaluate(test)

// Export for deployment
model.export_coreml("MyModel.mlmodel")
Drawing Classifier
Correct

Incorrect
Accuracy 80%
metrics = model.evaluate(test)
metrics = model.evaluate(8)
= model.evaluate()}
= model.evaluate()
= model.evaluate(8)
accuracy: 80%
precision: 85%
recall: 65%
f1_score: 77%

= model.evaluate()
Accuracy 85%
Accuracy 85%
Predicted: 1
Annotation: 7

Predicted: 4
Annotation: 1
Predicted: 3
Annotation: 7

Predicted: 4
Annotation: 1
import turicreate

// Load data
data = turicreate.SFrame("data.sframe")
train, test = data.random_split(0.8)

// Create a model
model = turicreate.drawing_classifier.create(train, "labels")

// Evaluate the model
metrics = model.evaluate(test)
metrics.explore()

// Export for deployment
model.export_coreml("MyModel.mlmodel")
Demo

Model evaluation
Demo Recap
Demo Recap

Interactively visualize model predictions
Demo Recap

- Interactively visualize model predictions
- Easily spot data/annotation errors
Demo Recap

Interactively visualize model predictions

Easily spot data/annotation errors

Identify systematic biases
Demo Recap

Interactively visualize model predictions
Easily spot data/annotation errors
Identify systematic biases
Drawing Classifier
Drawing Classifier

- Add more examples
- Remove incorrect training data

Overview
Drawing Classifier

Add more examples

Remove incorrect training data

Update incorrect annotations

Overview

Update incorrect annotations
data = tc.SFrame("data.sframe")
data = data.sort("labels")

Remove incorrect training data
Remove incorrect training data

data = data.filter_by([1, 2, 3], "labels")
data = data.append(more_data)
Remove incorrect training data

Add more examples

Update incorrect annotations

data = tc.drawing_classifier.annotate(data)
import turicreate

// Load data
data = turicreate.SFrame("data.sframe")
annotated_data = turicreate.drawing_classifier.annotate(data)

// Create a model
train, test = annotated_data.random_split(0.8)
model = turicreate.drawing_classifier.create(train, "labels")

// Evaluate the model
metrics = model.evaluate(test)

// Export for deployment
model.export_coreml("MyModel.mlmodel")
Summary
You have a Straight Flush!
Your cards are Ace of Diamonds, King of Diamonds, and Queen of Diamonds
The probability of a Straight Flush is 0.22%
MATH 101
Final Exam

February 14, 2019
9 am - 10 am

- This exam has five questions.
- Each question is worth 10 points each.
- You have 60 minutes to attempt this exam.
- This is an open book exam.
- Grading will be on a curve. Partial Credit will be awarded at the grader's discretion.

1. \( 1 + 1 = 2 \) [✓]
2. \( 3 + 2 = 4 \) [✗]
3. \( 3 \times 2 = 5 \) [✓]
4. \( 1 + 90 = 101 \) [✗]
5. \( e^\pi + \log(\pi) = \frac{26}{7} \) [✓]

38 out of 50
Install Latest Release

Updating to latest Turi Create is easy

Simply use `pip install` from Terminal

> pip install turicreate
More Information

developer.apple.com/wwdc19/420

<table>
<thead>
<tr>
<th>Introducing PencilKit</th>
<th>WWDC 2019</th>
</tr>
</thead>
<tbody>
<tr>
<td>Vision and Core ML</td>
<td>WWDC 2018</td>
</tr>
</tbody>
</table>