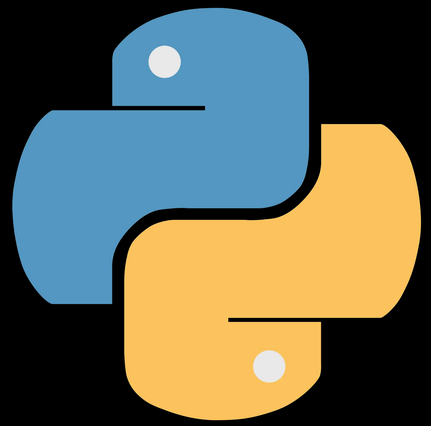
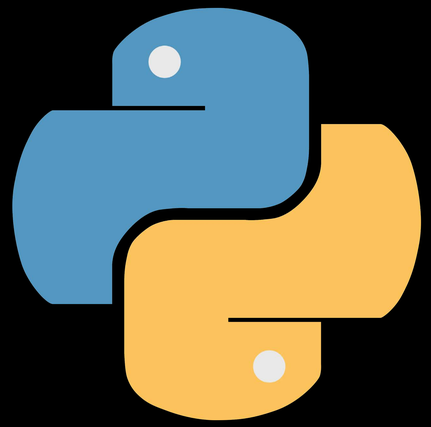
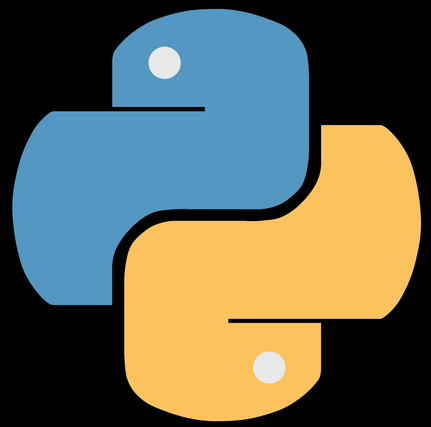
#WWDC19

Drawing Classification and One-Shot Object Detection in Turi Create

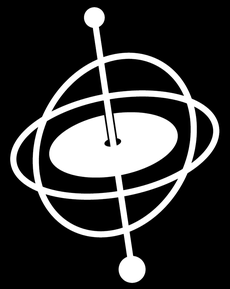
Sam Youtsey, EPM
Shantanu Chhabra, Engineer
Abhishek Pratapa, Engineer

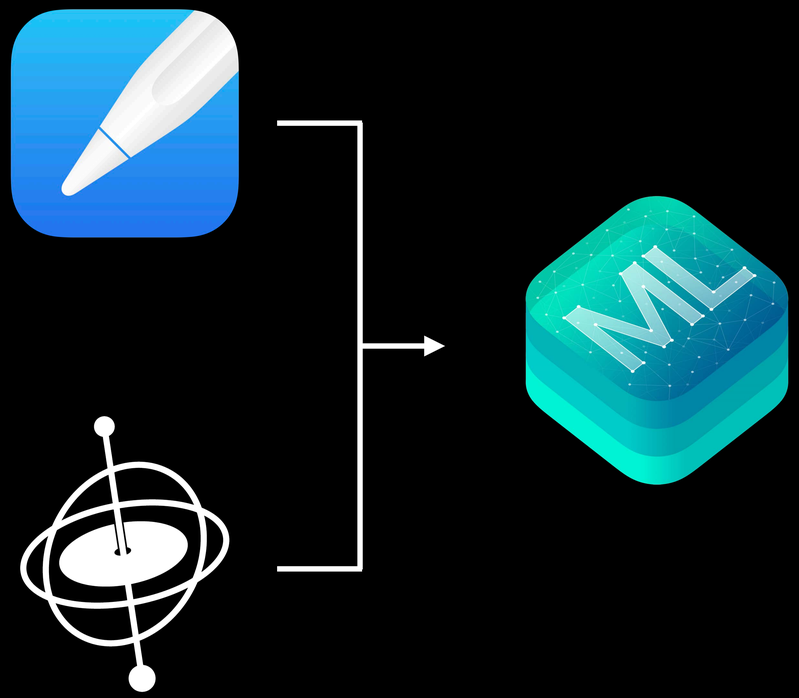


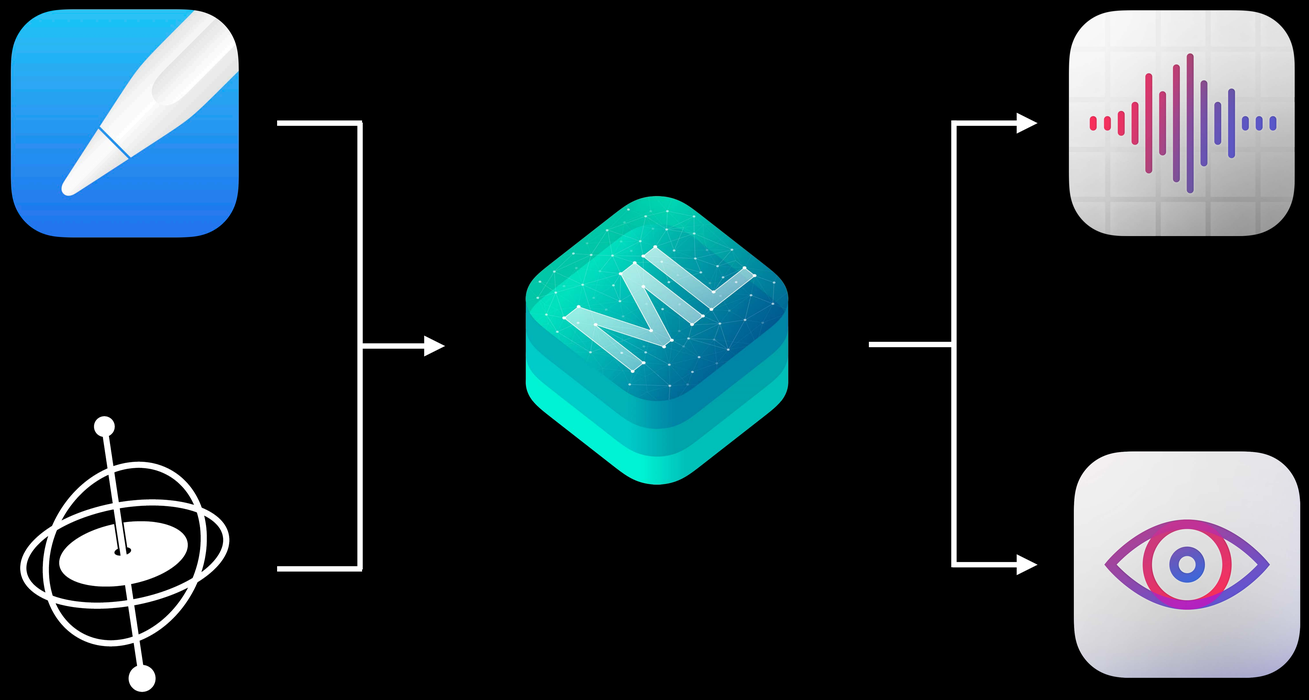


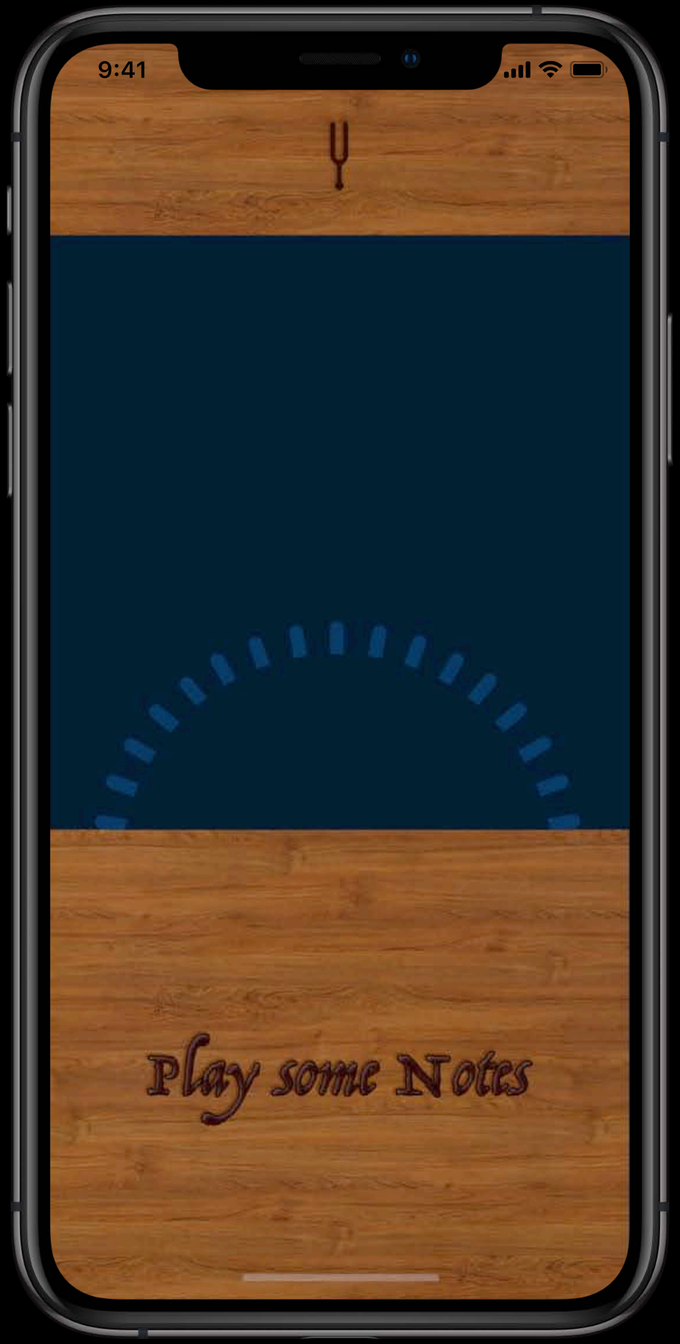


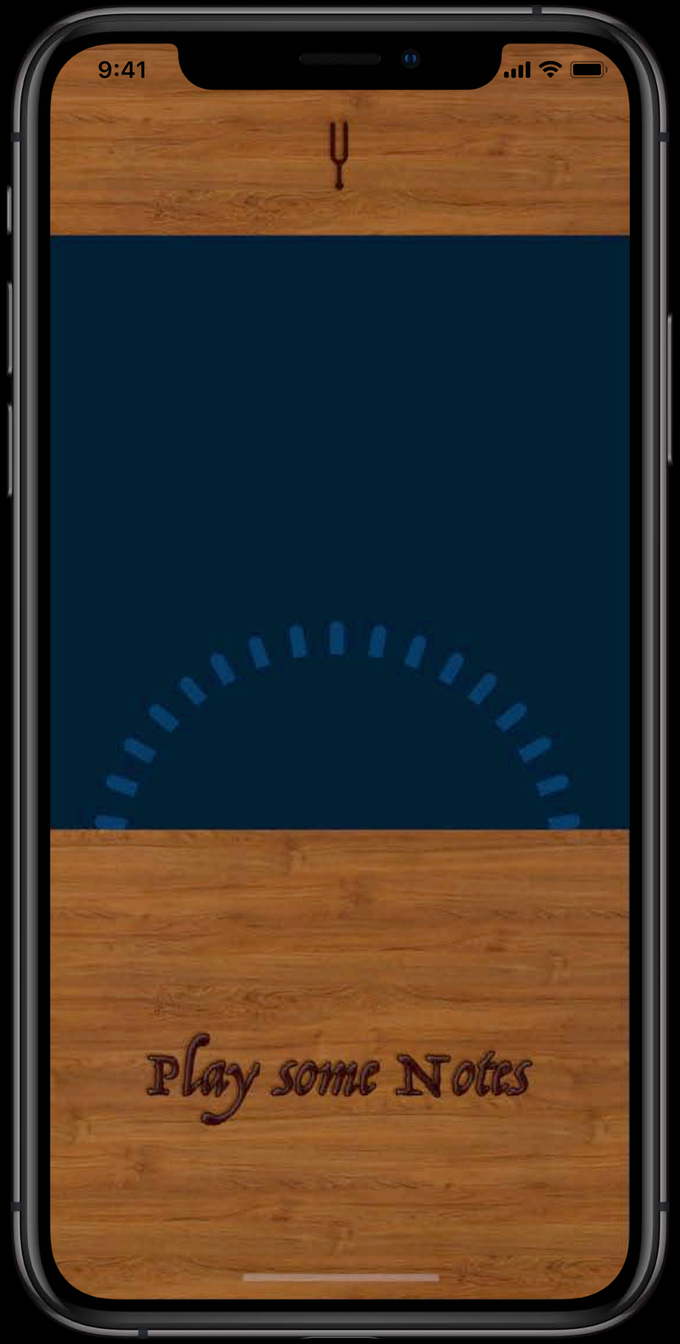
Create Core ML models for your intelligent applications.




ML


ML


9:41
Play some Notes


9:41
Play some Notes

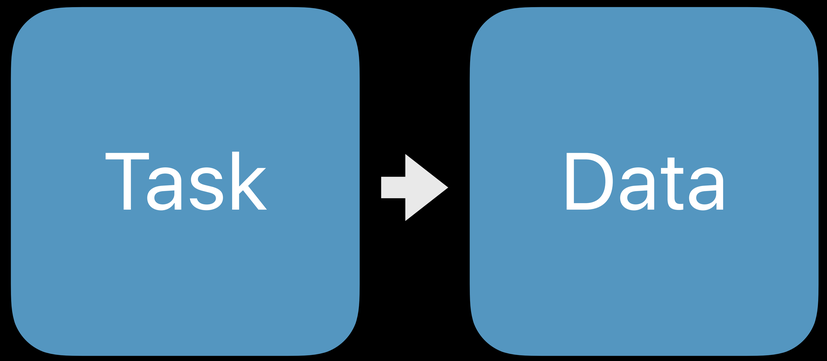

Back
10:09
0


Back
10:09
0

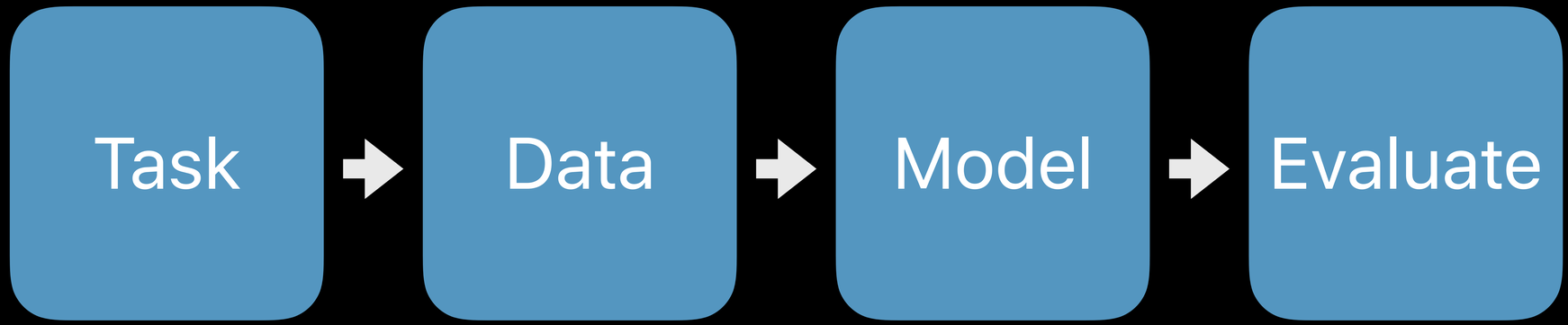

9:41
Detecting …

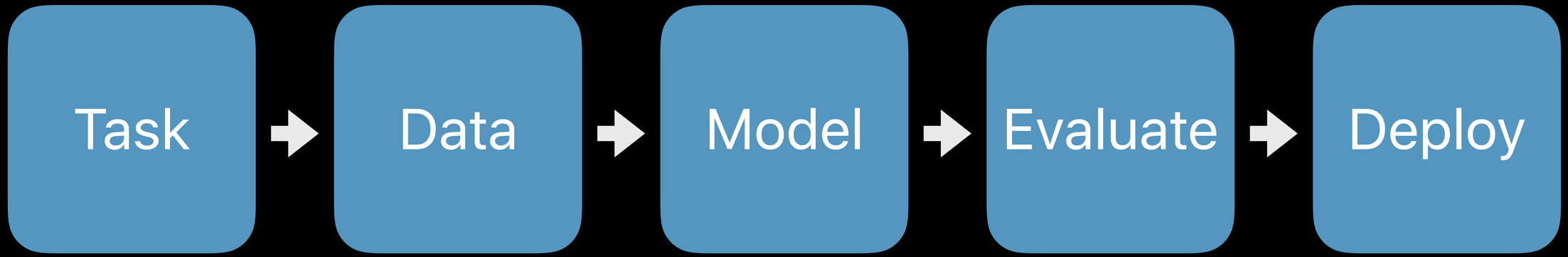

9:41
Detecting …


Task


Task
Data


Task
Data
Model


Task
Data
Model
Evaluate


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

// 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("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")
```

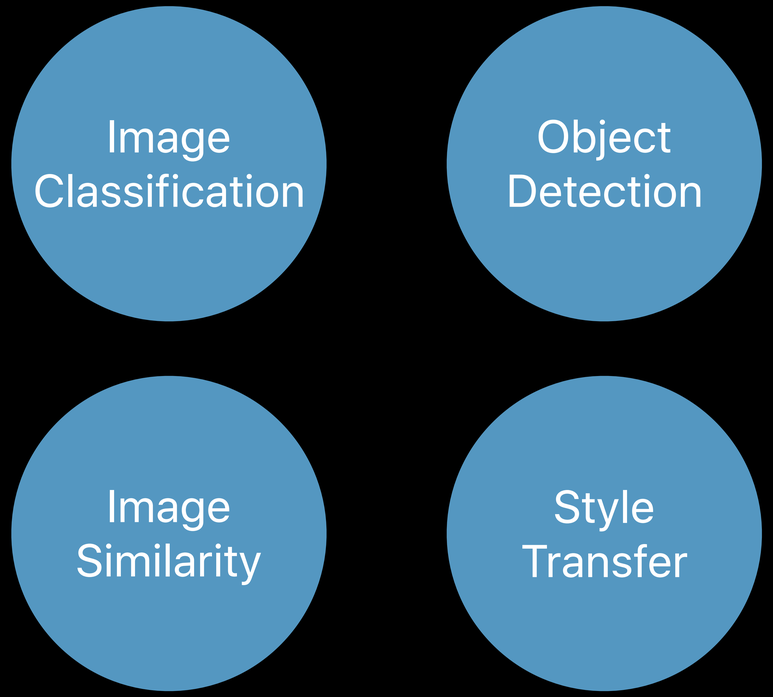


Image
Classification
Object
Detection
Image
Similarity
Style
Transfer

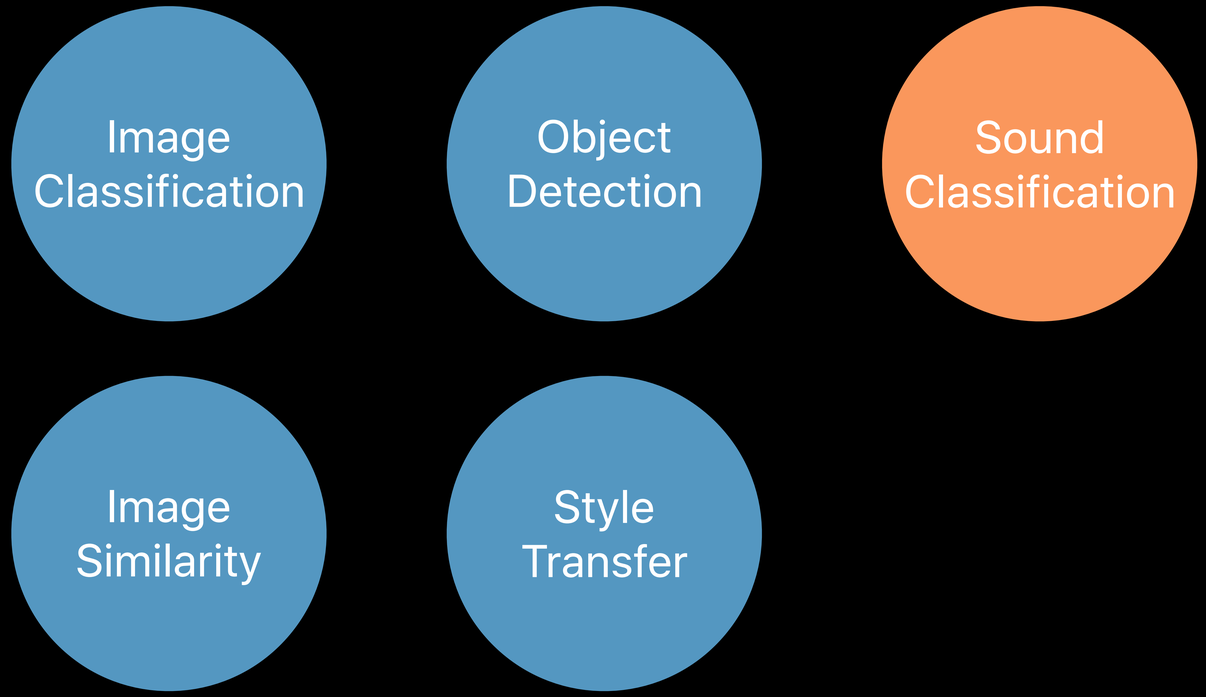


Image Classification
Object Detection
Sound Classification
Image Similarity
Style Transfer

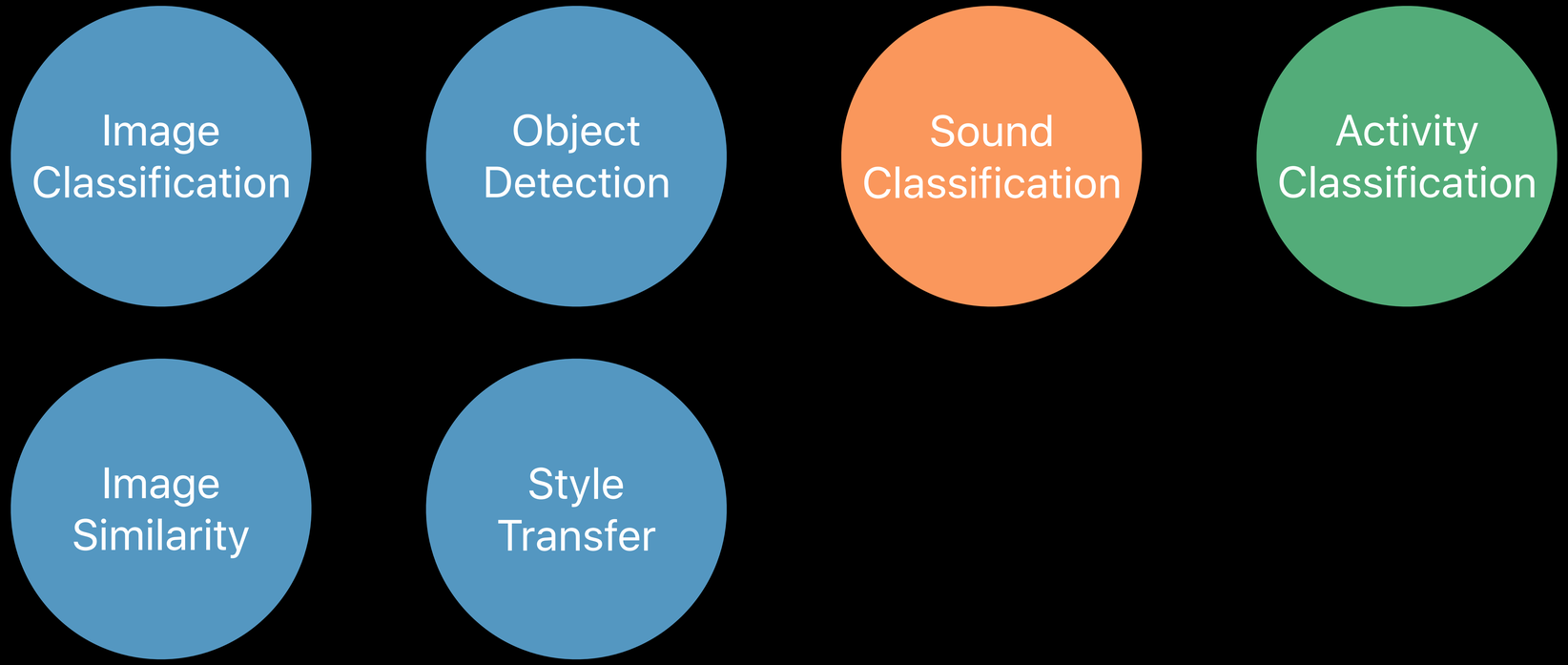


Image Classification
Object Detection
Sound Classification
Activity Classification
Image Similarity
Style Transfer

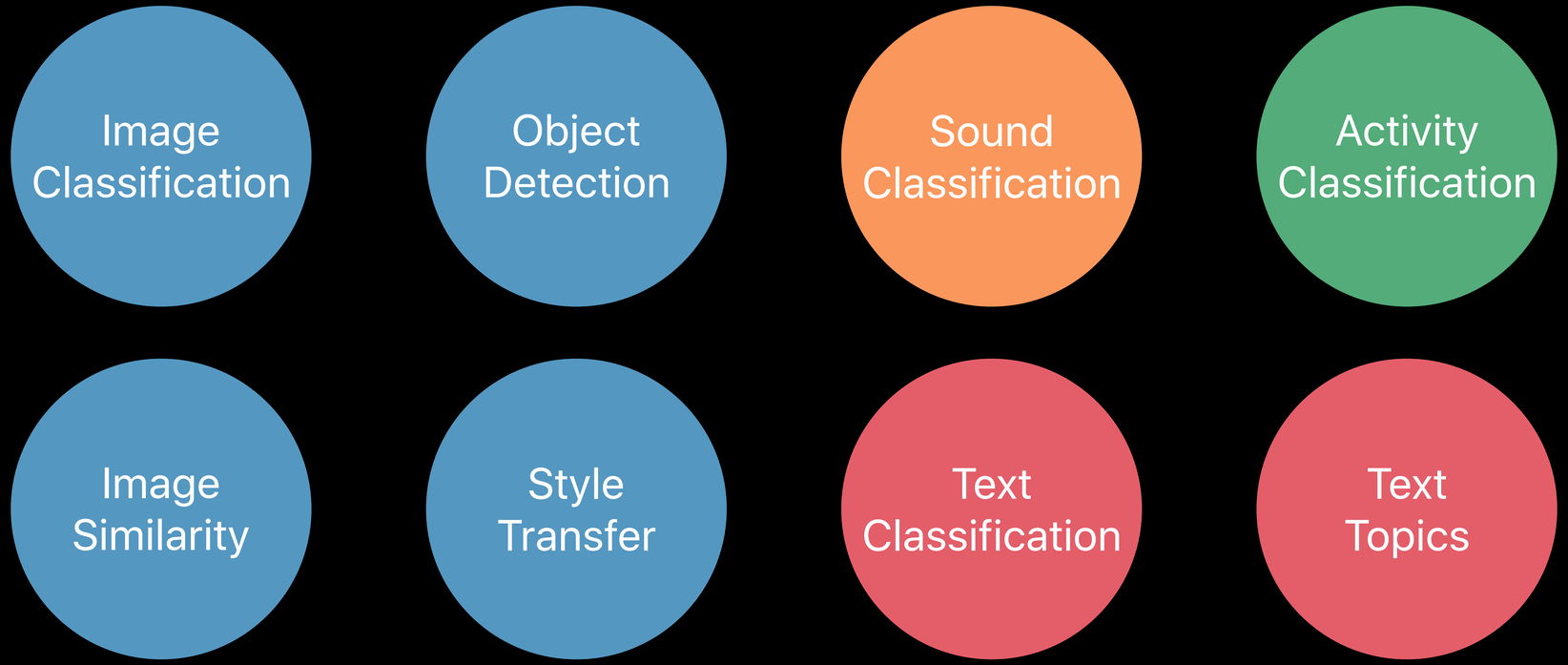


Image Classification
Object Detection
Sound Classification
Activity Classification
Image Similarity
Style Transfer
Text Classification
Text Topics

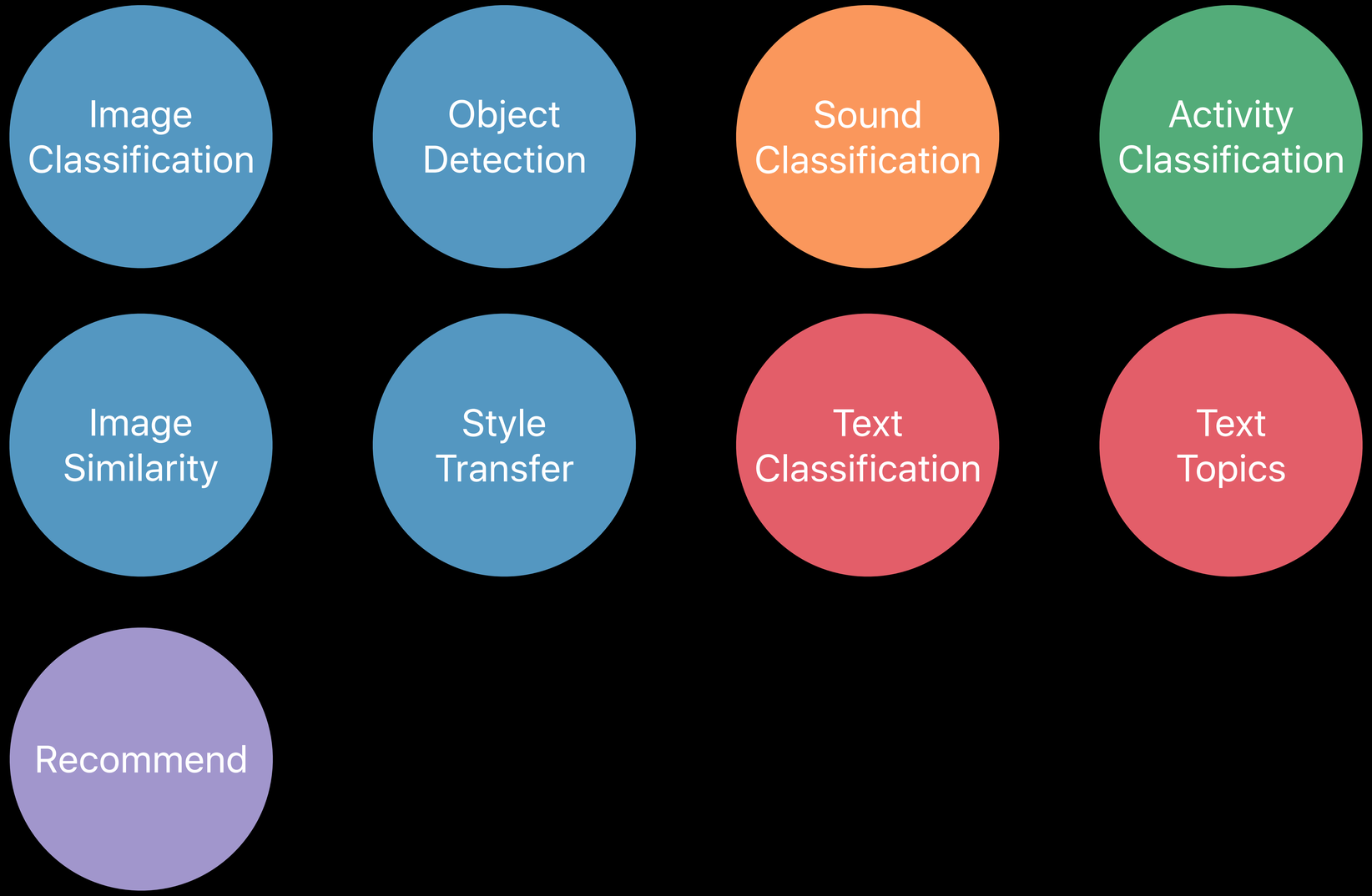
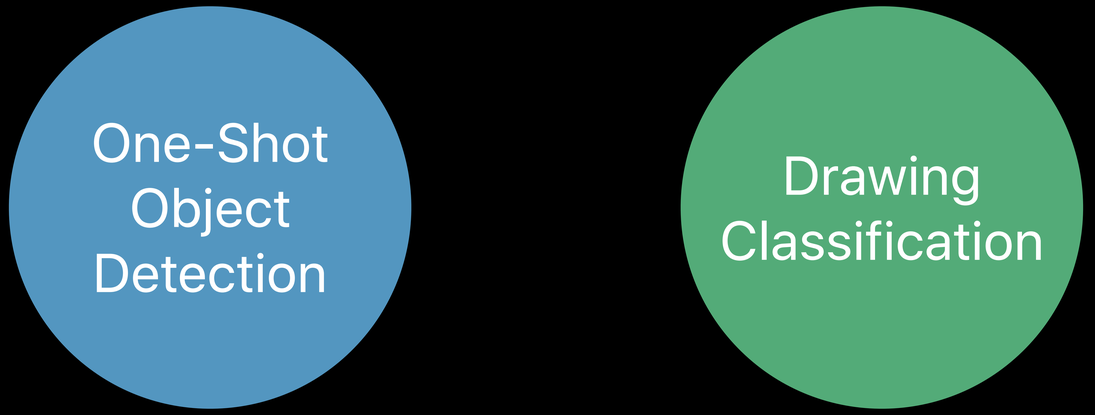


Image Classification
Object Detection
Sound Classification
Activity Classification
Image Similarity
Style Transfer
Text Classification
Text Topics
Recommend


Image Classification
Object Detection
Sound Classification
Activity Classification
Image Similarity
Style Transfer
Text Classification
Text Topics
Recommend
Clustering
Regression
Similarity

New Tasks

One-Shot Object Detection

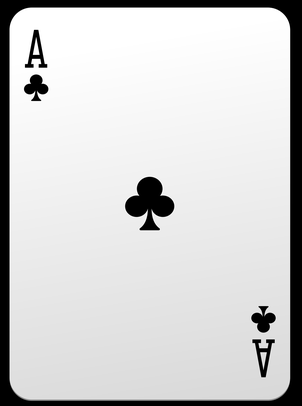

One-Shot
Object
Detection
Drawing
Classification

One-Shot Object Detection


9:41 AM Mon Jun 3
100%
Library
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%


9:41
5
BRAKE
CHECK
x1000 RPM
Airbag Warning
Engine Check
Power Steering


9:41
Founded in 1976
Headquartered in
Cupertino, California


A
A


STOP
ABS

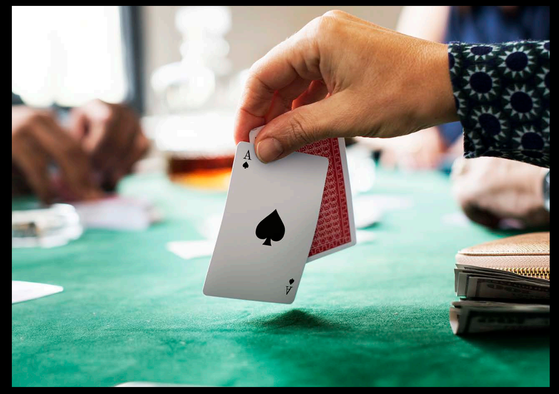

9:41 AM Mon Jun 3
100%
Library
You have a Straight Flush!
Your cards are Ace of Diamonds, King of Diamonds, and Queen of Diamonds
The probability of a Straight Flush is 0.22%

Original Approach



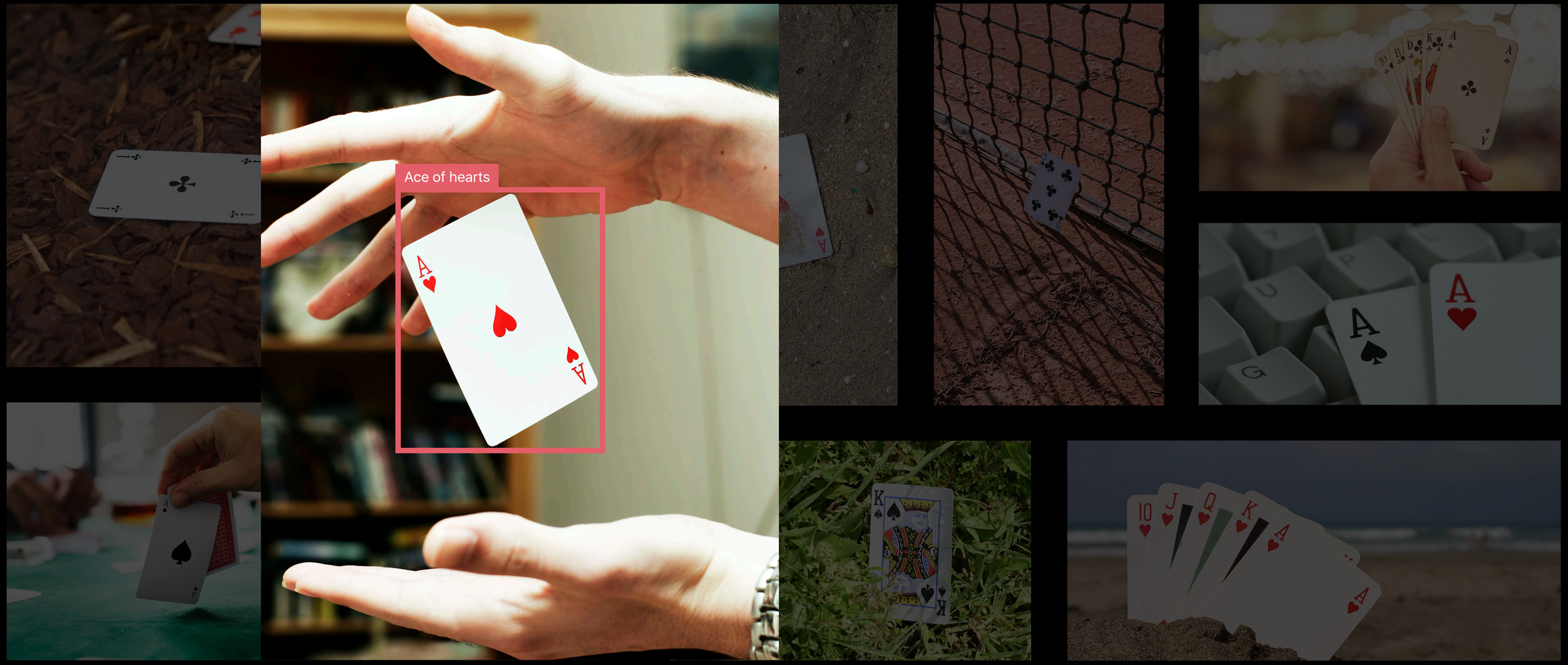







Original Approach

Original Approach

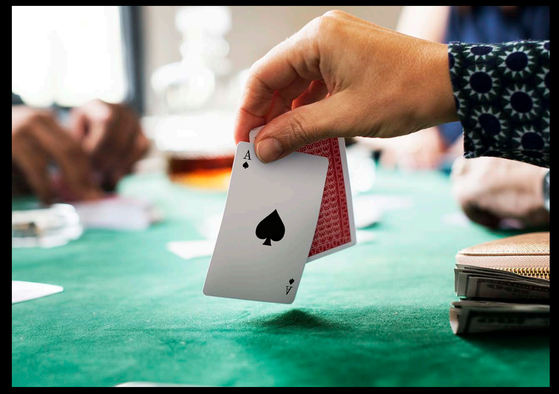


Original Approach







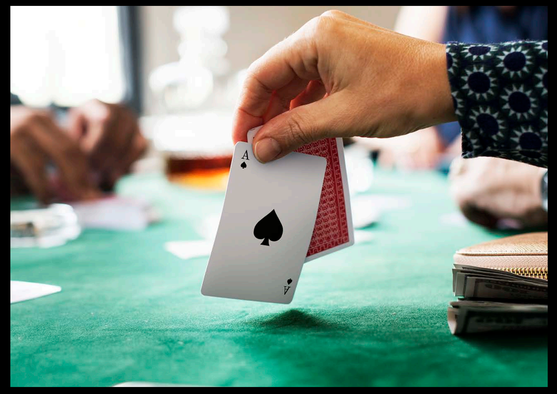



Original Approach

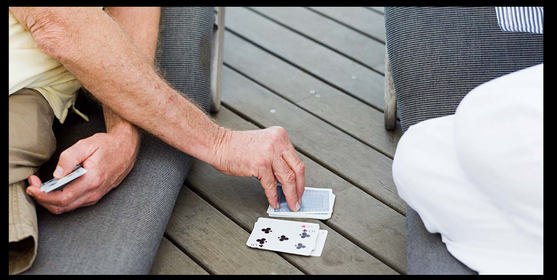




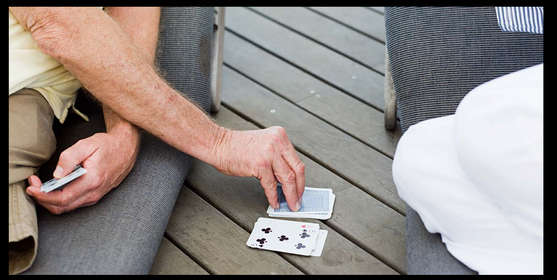





Original Approach



Original Approach

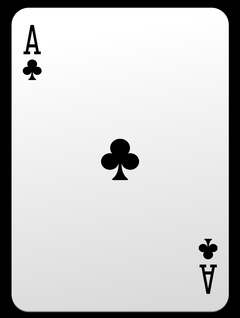
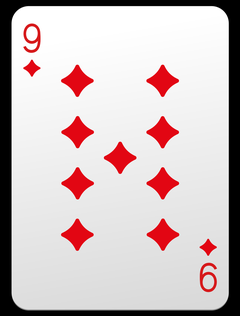


Original Approach

Original Approach

One-Shot

NEW

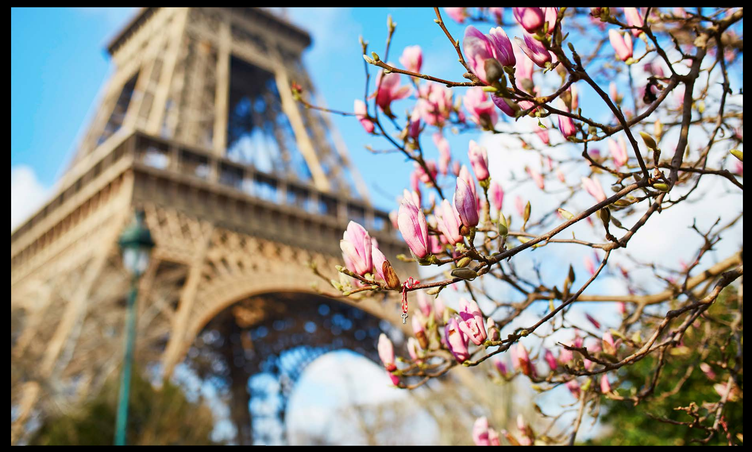
Class	Image
Ace-Clubs	
Nine-Diamonds	
...	...

Synthetic Data Augmentation










Synthetic Data Augmentation










Comparison

One-Shot Object Detection	Traditional Object Detection
2D objects only	2D or 3D objects
Little data required	Data-intensive
No annotation needed	Annotation required
Objects with “regularity”	Any object

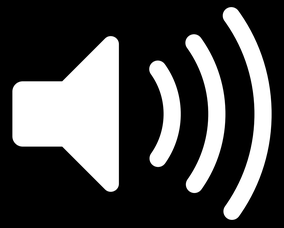
```
import turicreate

// 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")
```

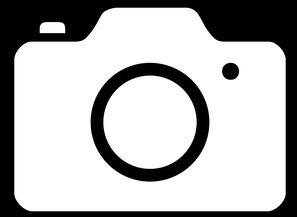
Demo

Three card poker

Shantanu Chhabra, Engineer

Drawing Classification

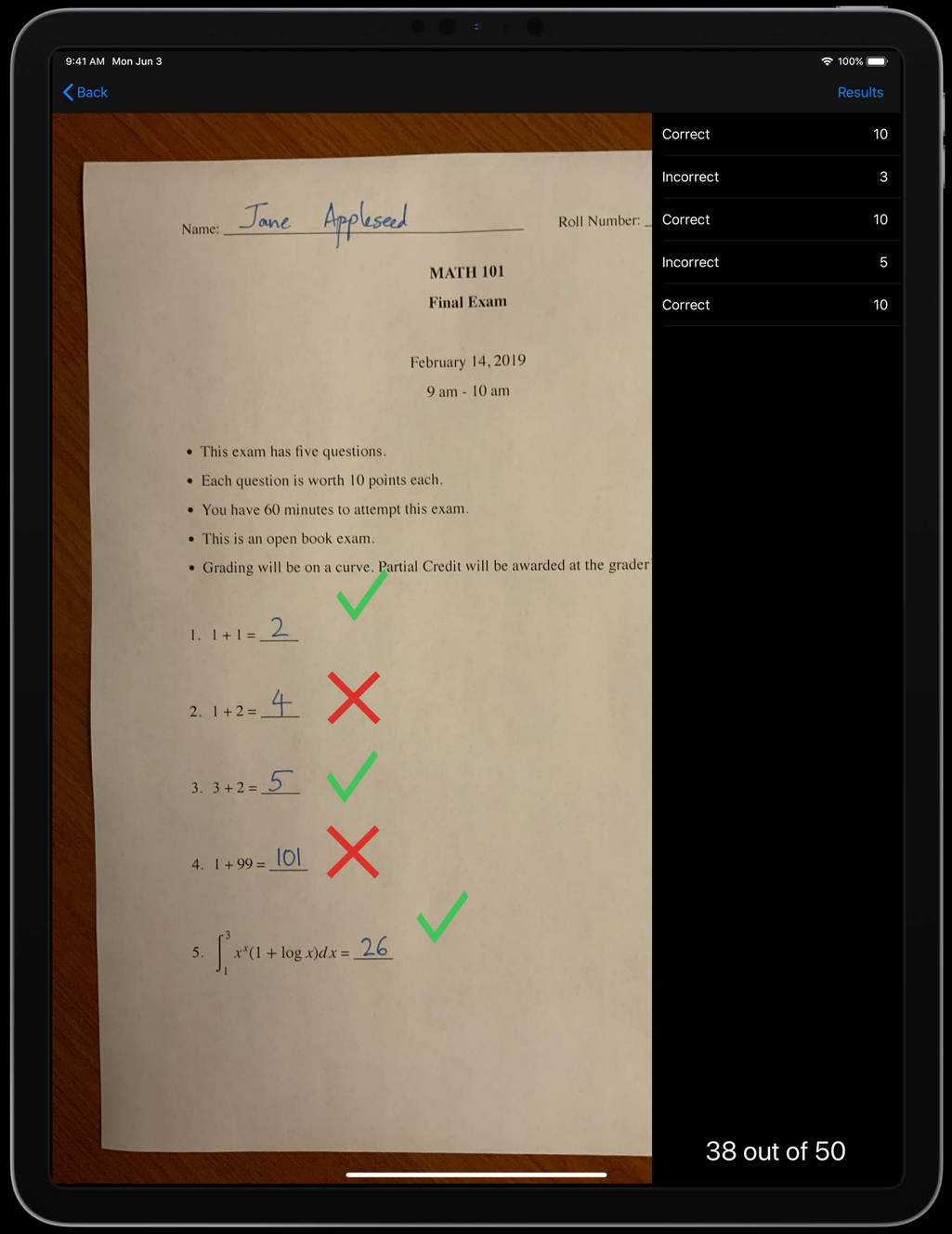





Pencil


HAPPY
birthday


HAPPY
birthday


9:41 AM Mon Jun 3
100%
Back
Results
Name: Jane Appleseed
Roll Number:
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
1. 1 + 1 = 2
2. 1 + 2 = 4
3. 3 + 2 = 5
4. 1 + 99 = 101
5. $\int_1^3 x^x(1+\log x)dx$ = 26
Correct 10
Incorrect 3
Correct 10
Incorrect 5
Correct 10
38 out of 50

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")
```

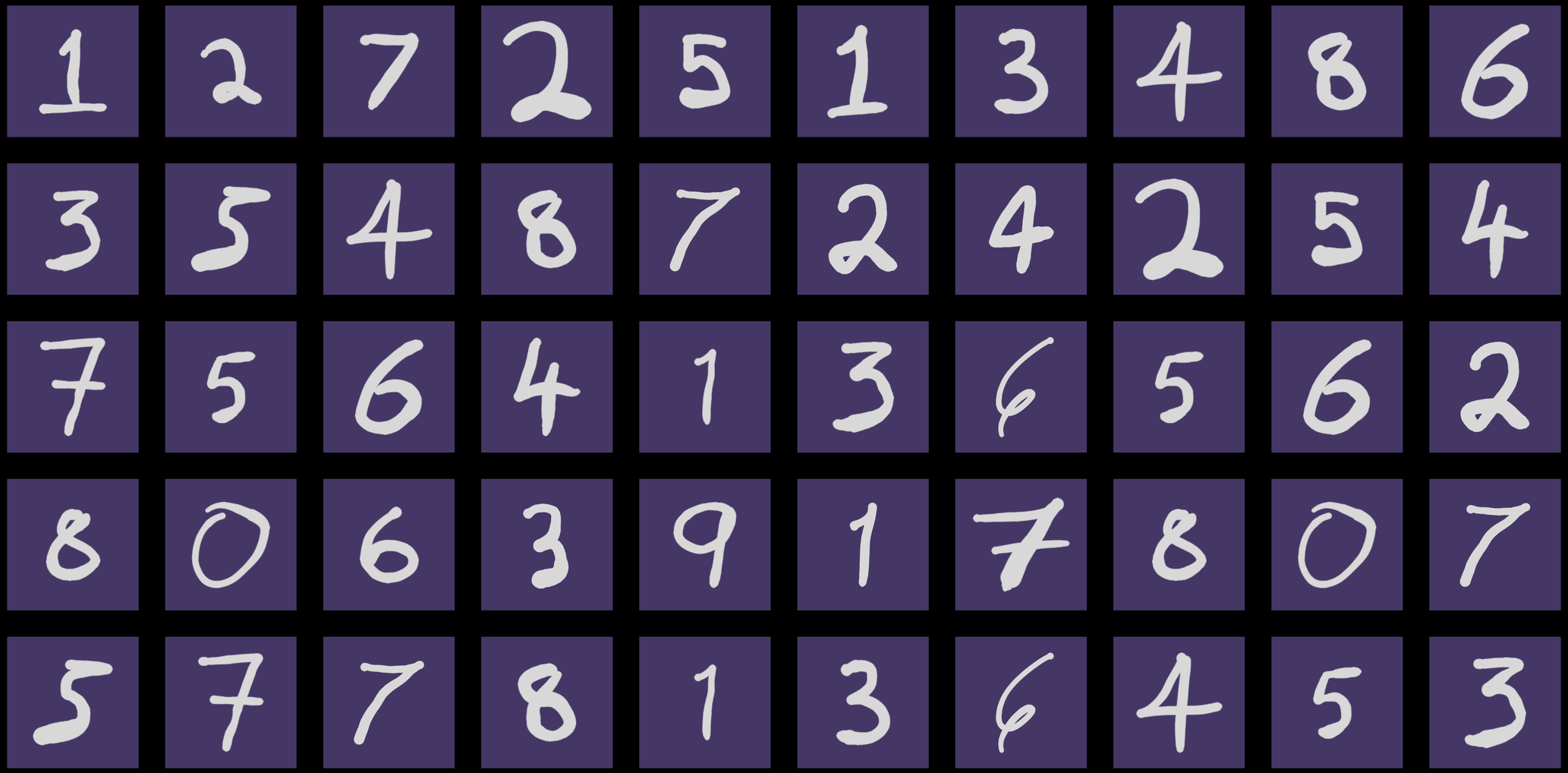
Model Creation

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

State of the art

Model less than
500 kB on-device

GPU acceleration




MLMODEL

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)
```

NEW

```
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)
```

NEW

```
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)
```

NEW

```
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)
```

NEW

```
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

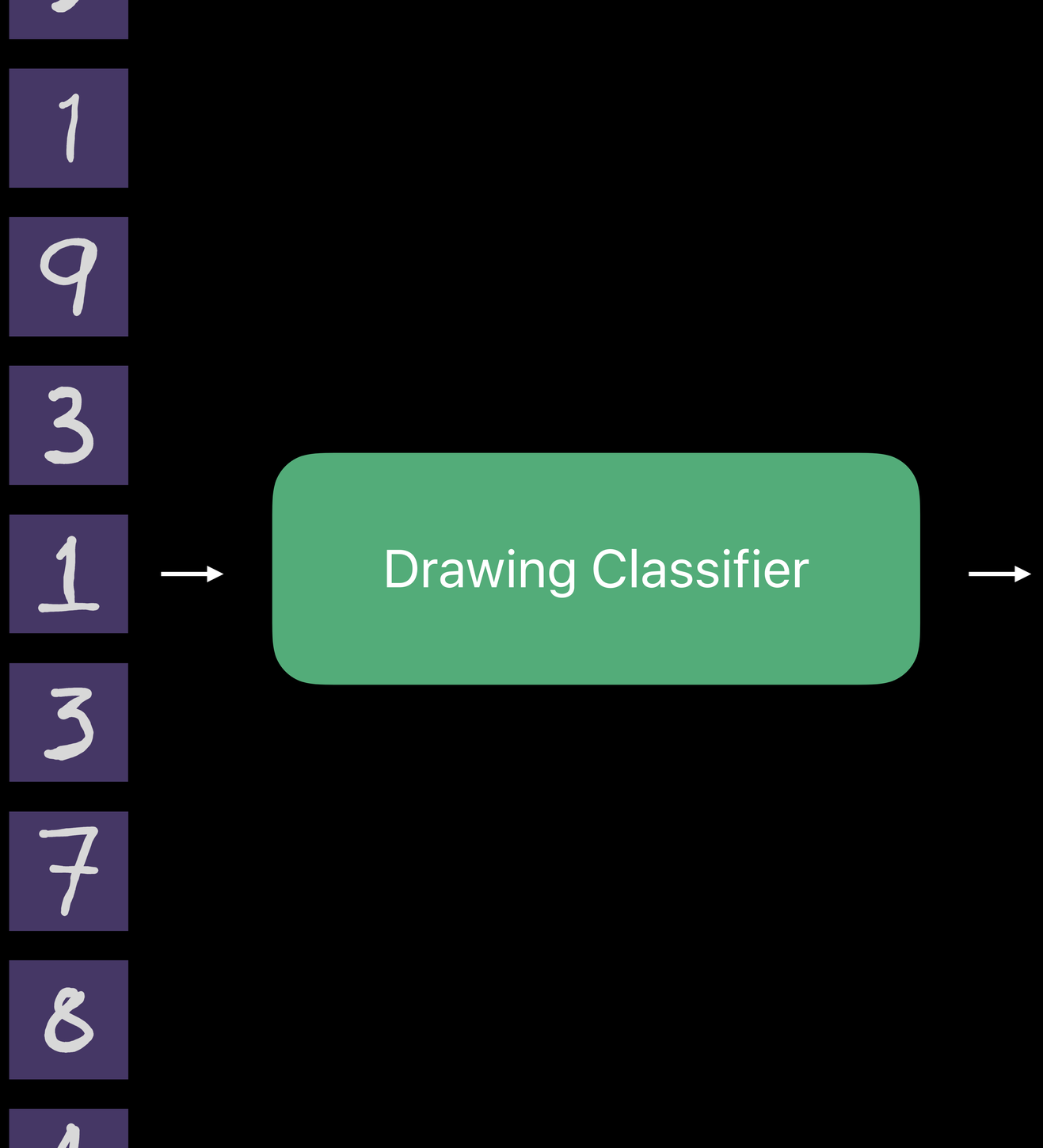
Exported to Core ML

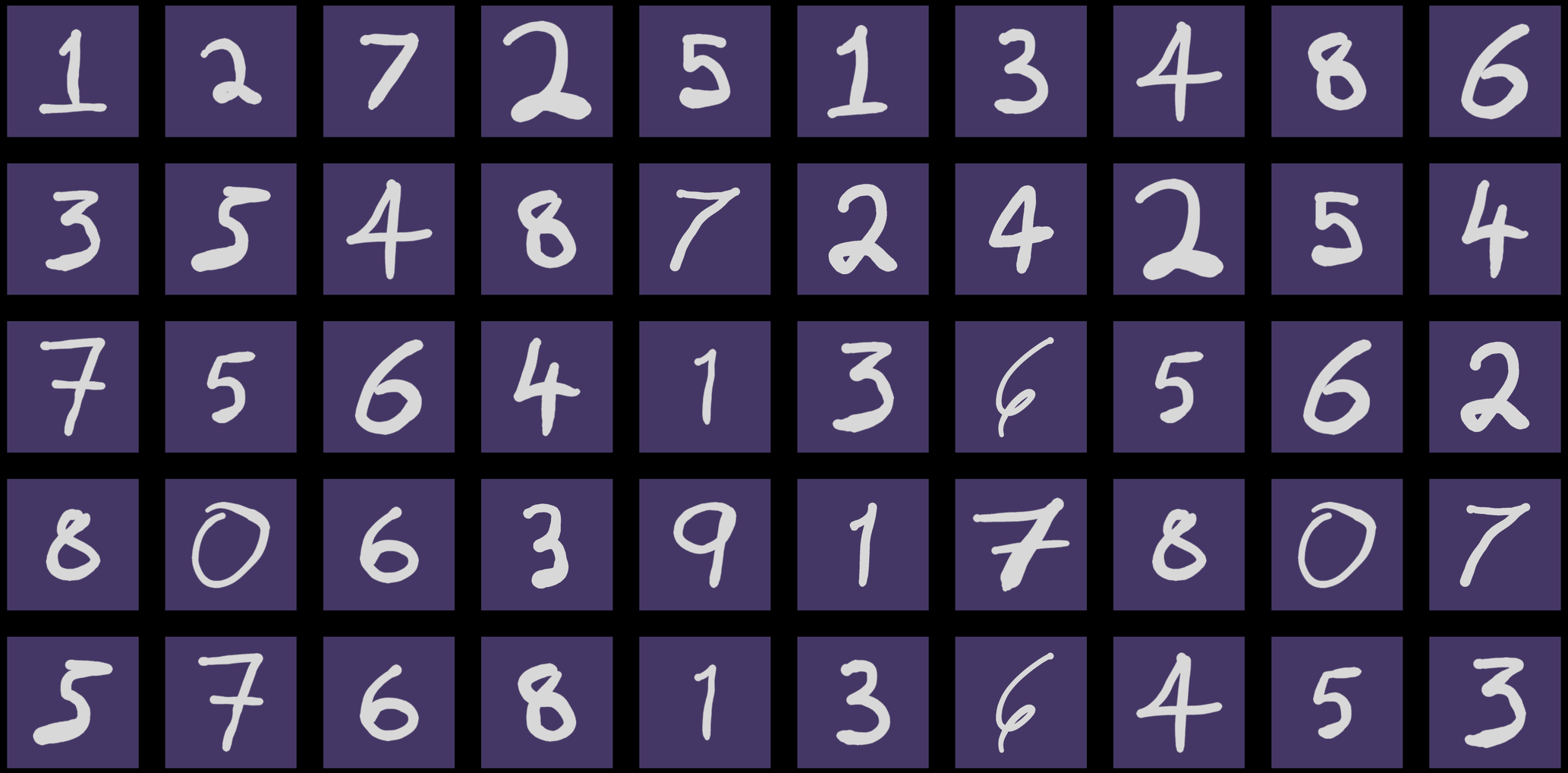
Deployed using PencilKit, Vision

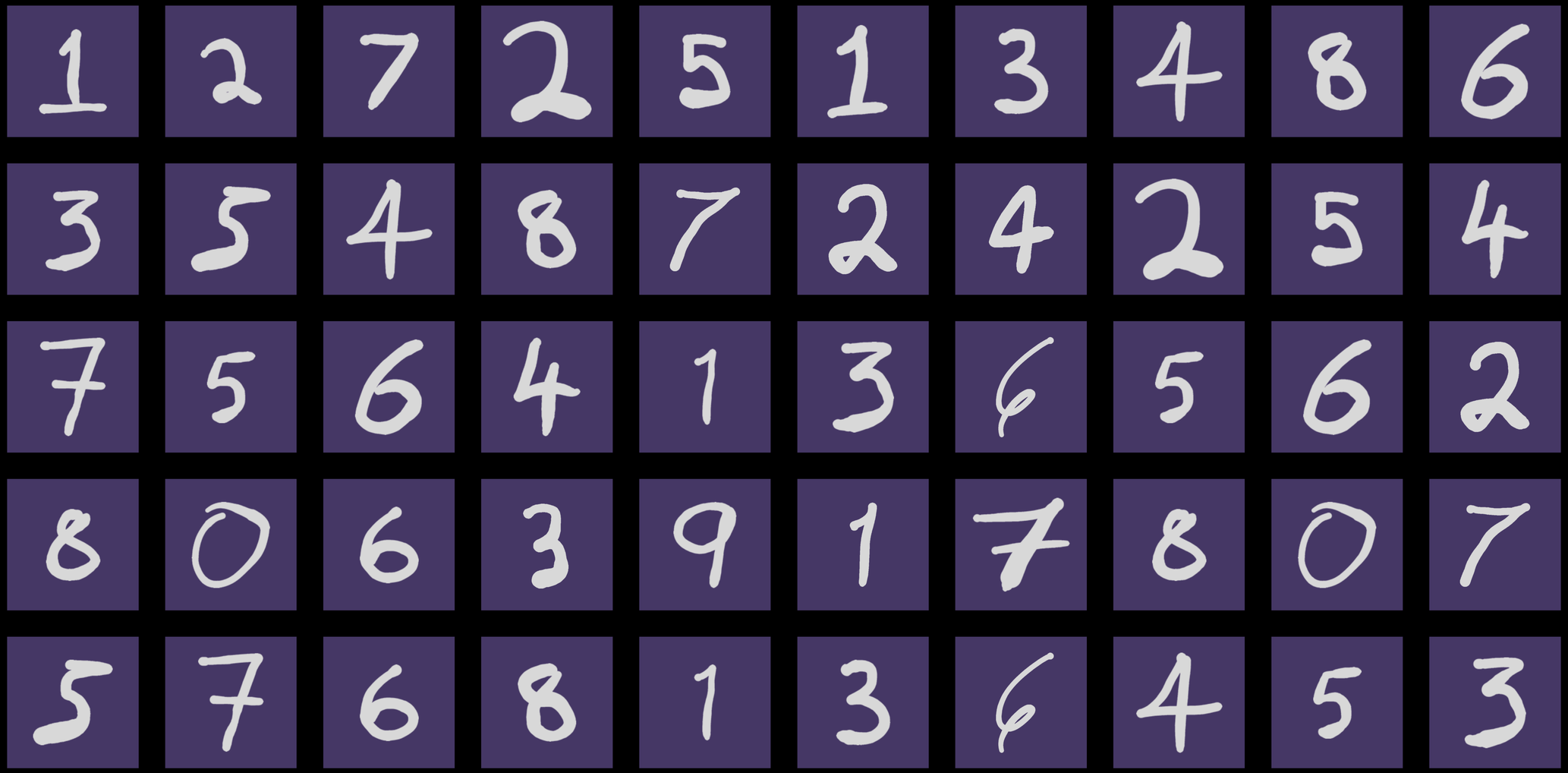
Interactive Model Evaluation

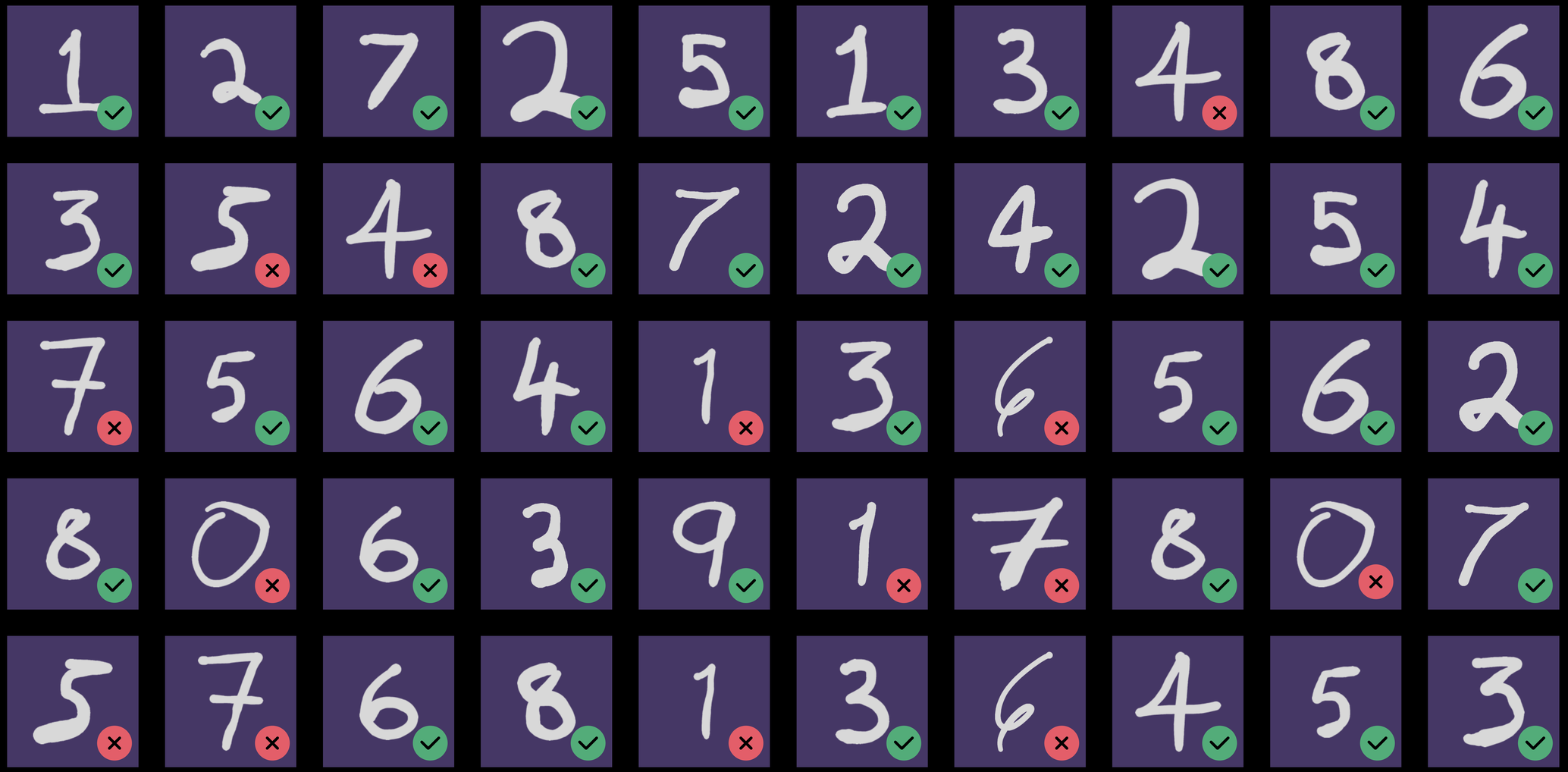
Abhishek Pratapa, Engineer

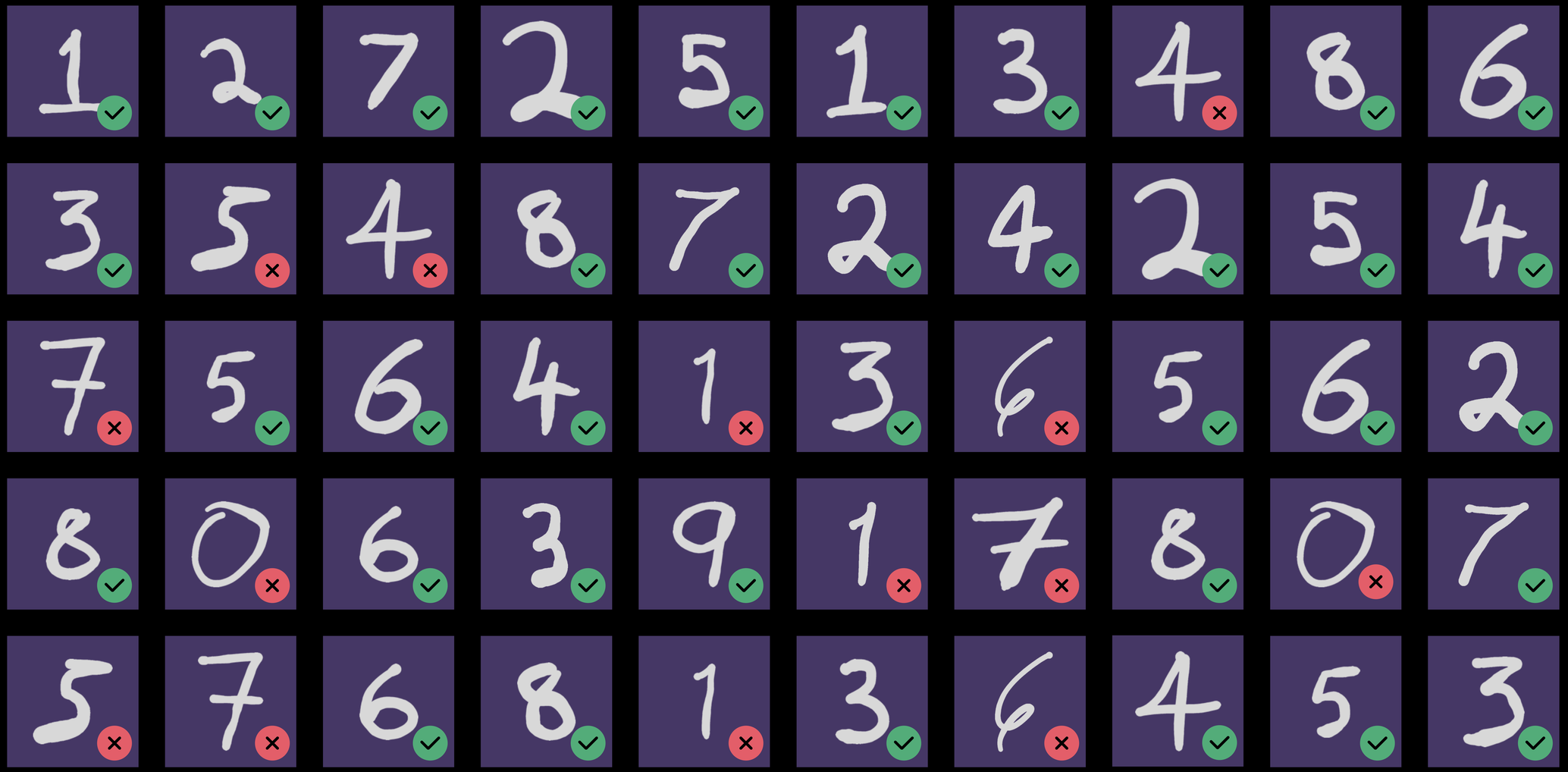
```
import turicreate

// 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






1 2 7 2 5 1 3 4 8 6
3 5 4 8 7 2 4 2 5 4
7 5 6 4 1 3 6 5 6 2
8 0 6 3 9 1 7 8 0 7
5 7 6 8 1 3 6 4 5 3


1 2 7 2 5 1 3 4 8 6
3 5 4 8 7 2 4 2 5 4
7 5 6 4 1 3 6 5 6 2
8 0 6 3 9 1 7 8 0 7
5 7 6 8 1 3 6 4 5 3

Correct

Incorrect


Accuracy
80%


Accuracy
80%
F1 Score
77%
Precision
85%
Recall
65%

```
metrics = model.evaluate( test )
```

```
metrics = model.evaluate(  )
```

= model.evaluate()

```
= model.evaluate(  )
```


= model.evaluate( )

accuracy: 80%
precision: 85%
recall: 65%
f1_score: 77%
...

= model.evaluate()


5

Accuracy

85%

Accuracy

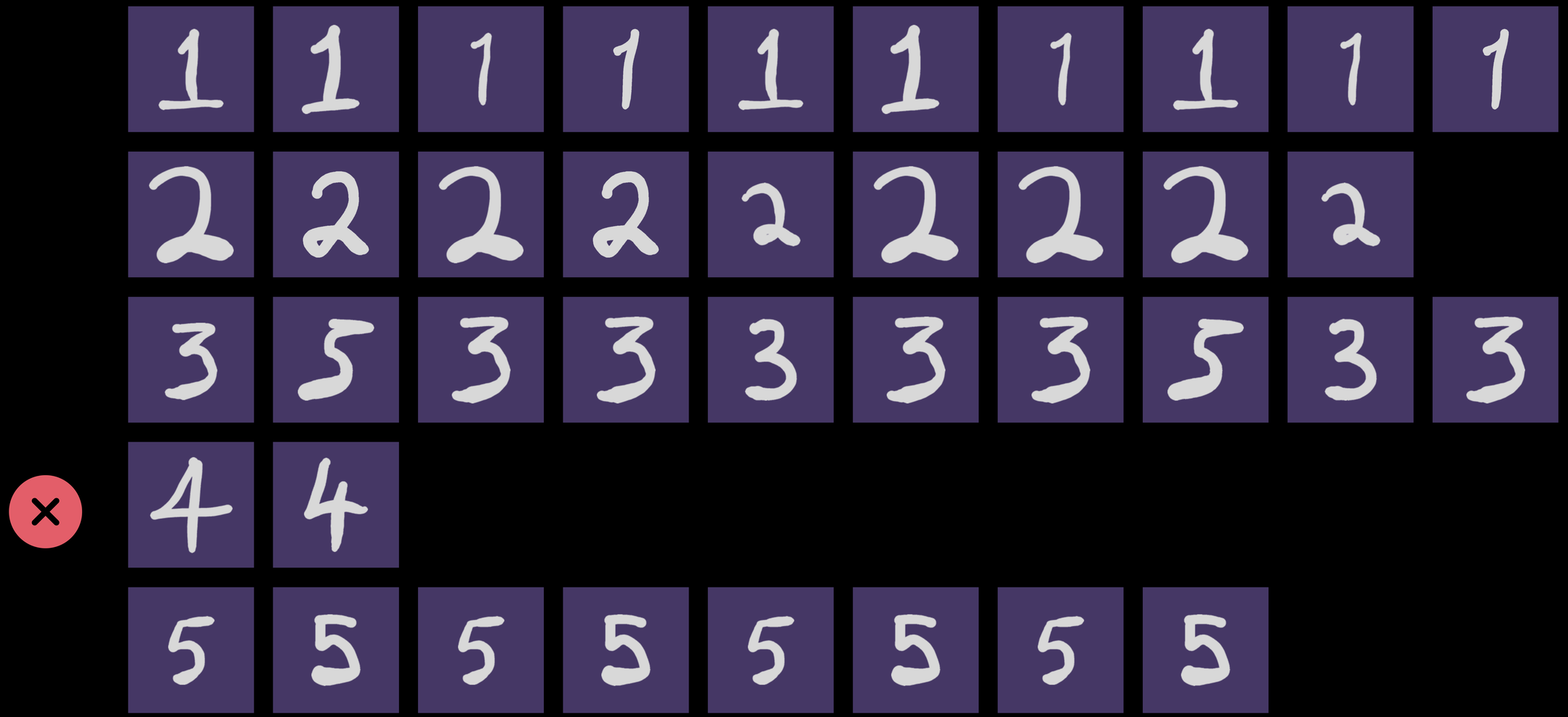
85%

Accuracy

85%


1
Accuracy
93%
2
Accuracy
85%
3
Accuracy
98%
4
Accuracy
23%
5
Accuracy
94%


1
Accuracy
93%
2
Accuracy
85%
3
Accuracy
98%
4
Accuracy
23%
5
Accuracy
94%


1 1 1 1 1 1 1 1 1 1
2 2 2 2 2 2 2 2 2
3 5 3 3 3 3 3 5 3 3
4 4
5 5 5 5 5 5 5 5

Predicted: 1
Annotation: 7

Predicted: 1
Annotation: 7

Predicted: 4
Annotation: 1

Predicted: 3
Annotation: 7

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

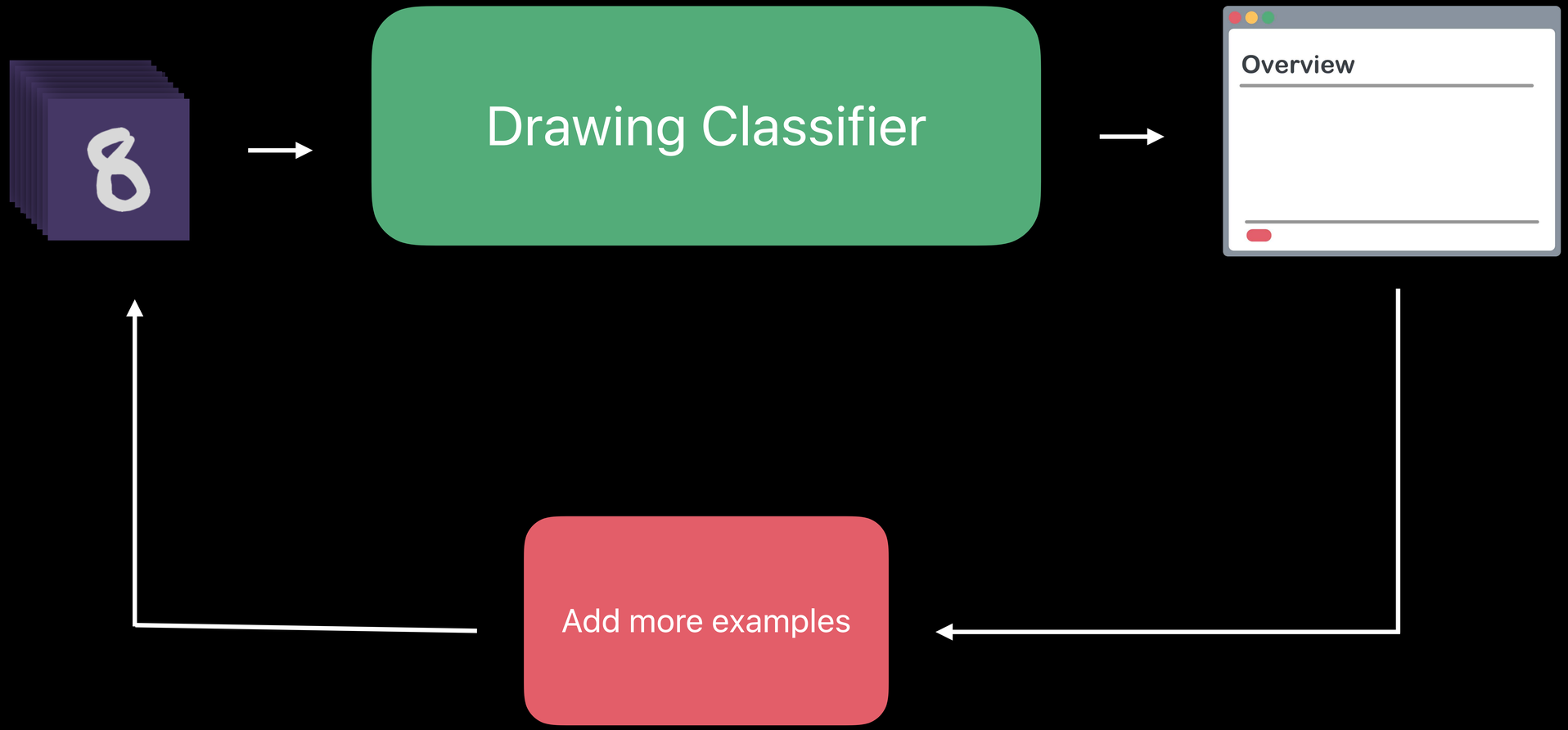

8

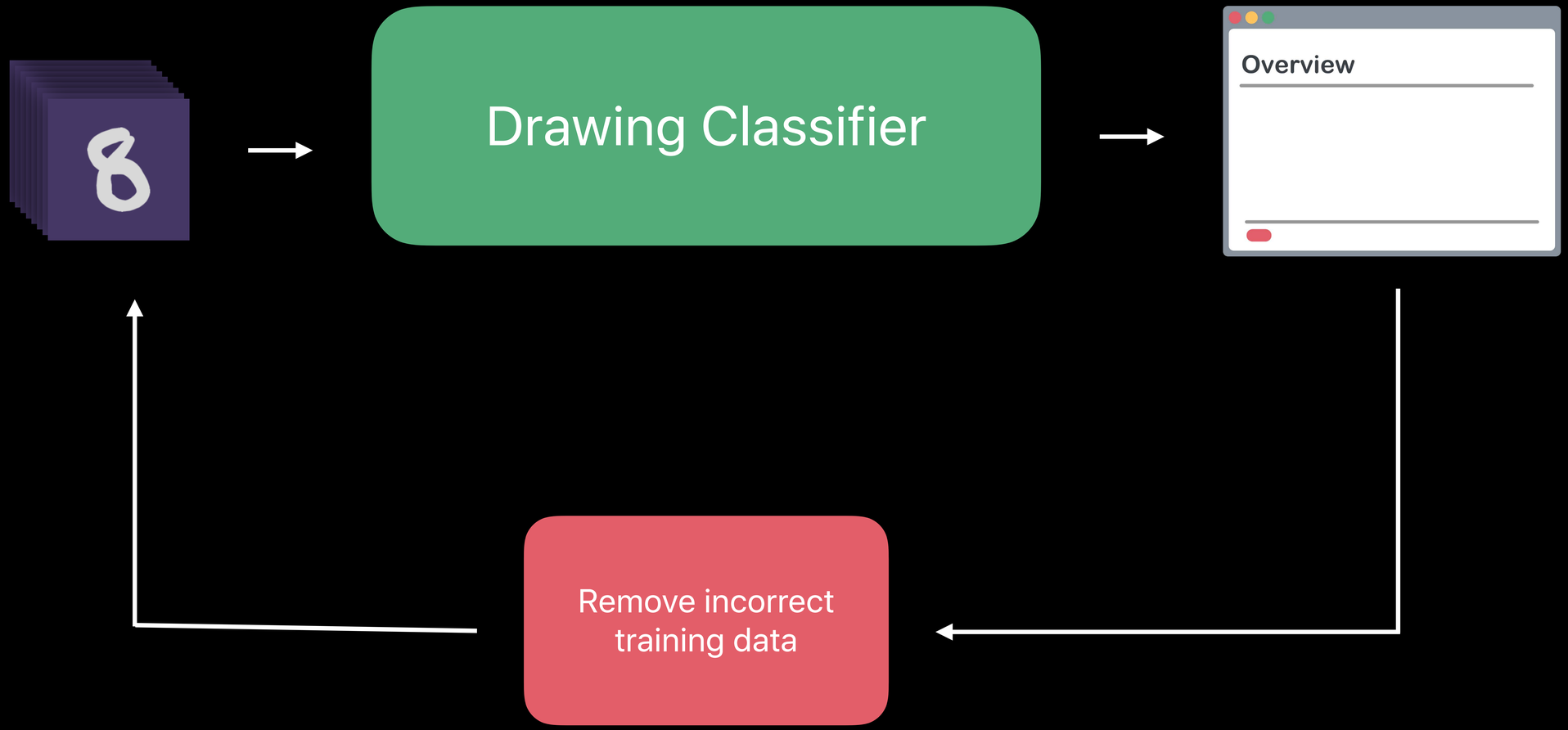
Drawing Classifier

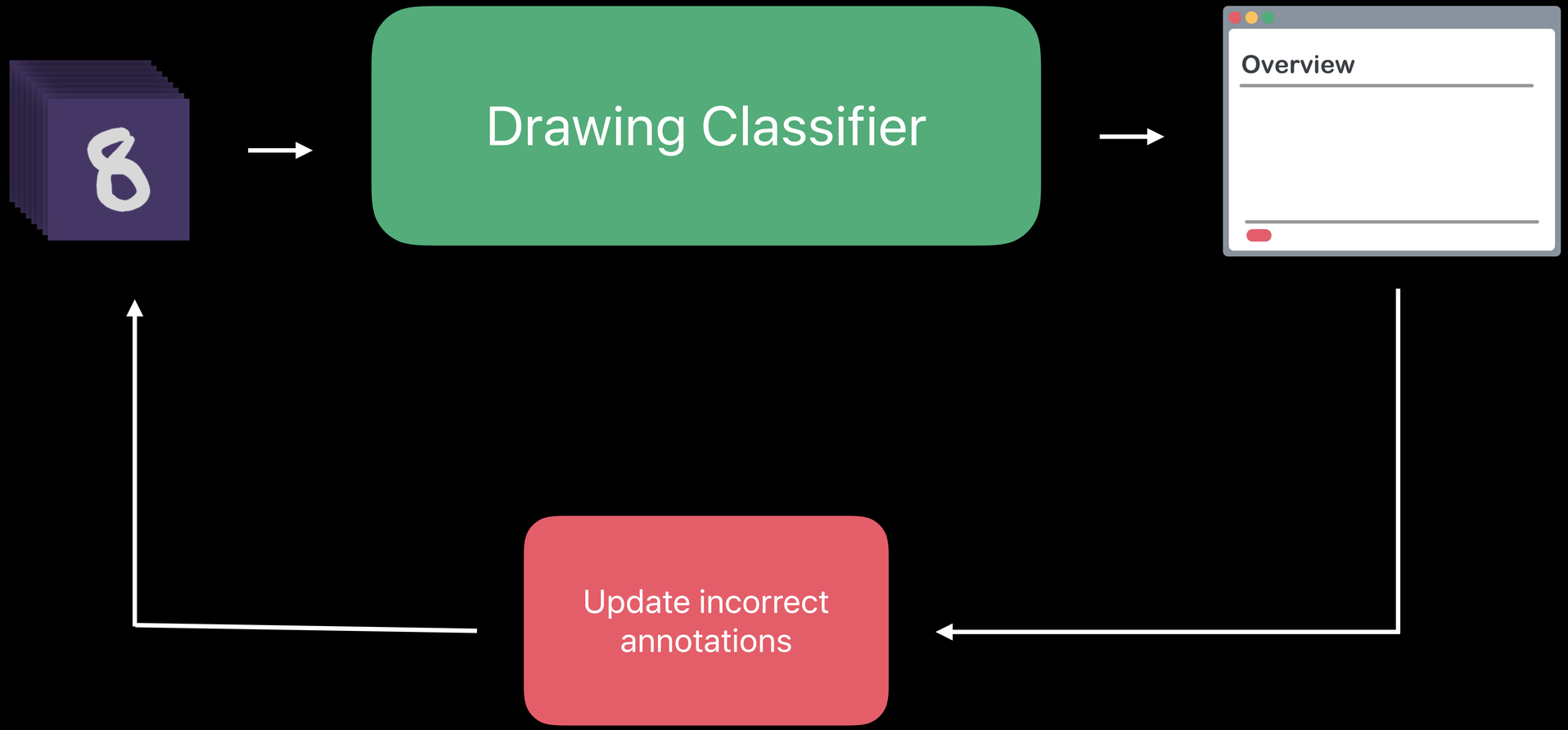

Drawing Classifier
Overview


Drawing Classifier
Overview

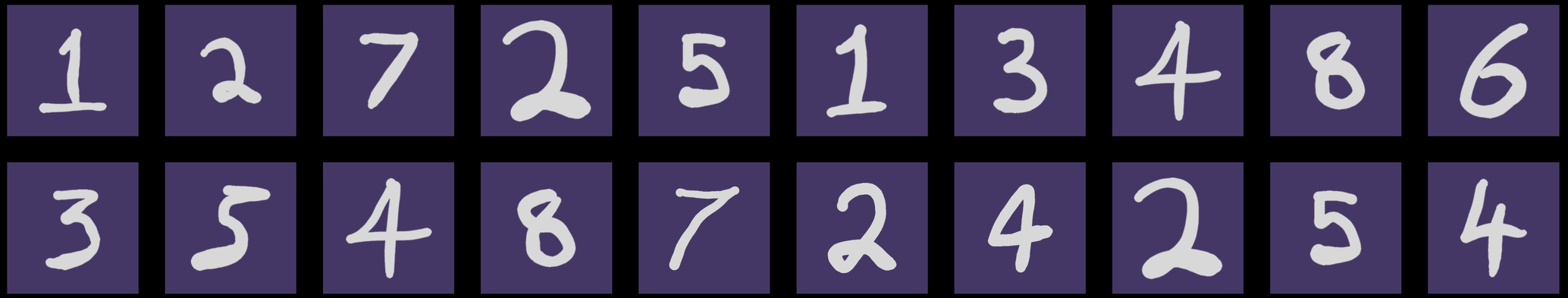

Drawing Classifier
Overview


Drawing Classifier
Overview
Add more examples


Drawing Classifier
Overview
Remove incorrect
training data

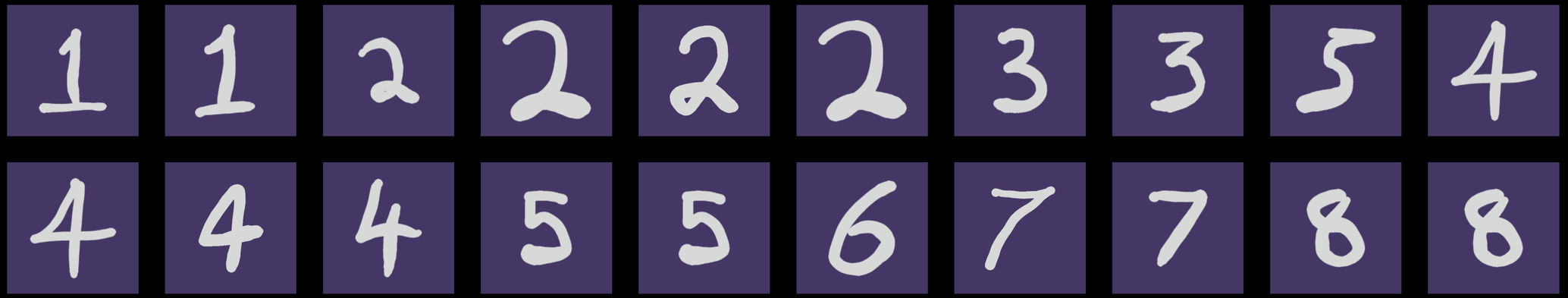

Drawing Classifier
Overview
Update incorrect annotations

```
data = tc.SFrame("data.sframe")
```



Remove incorrect training data

```
data = data.sort("labels")
```



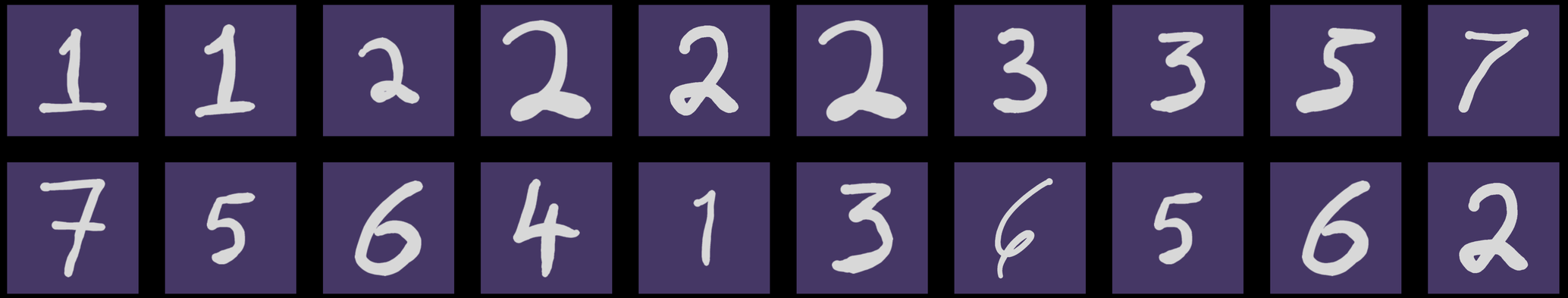
Remove incorrect training data

```
data = data.filter_by([1, 2, 3], "labels")
```

Remove incorrect training data

Add more examples

```
data = data.append(more_data)
```



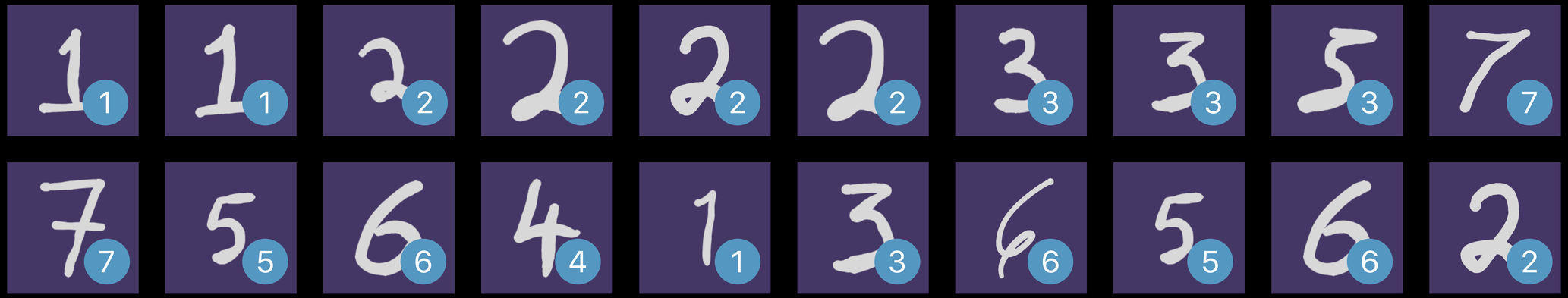
NEW

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")
```

Total: 1831

Hide Annotated image

Labels

+ Add New Label

Jump to image

Extracting Features: 7%

Total: 1831

Hide Annotated image

Labels

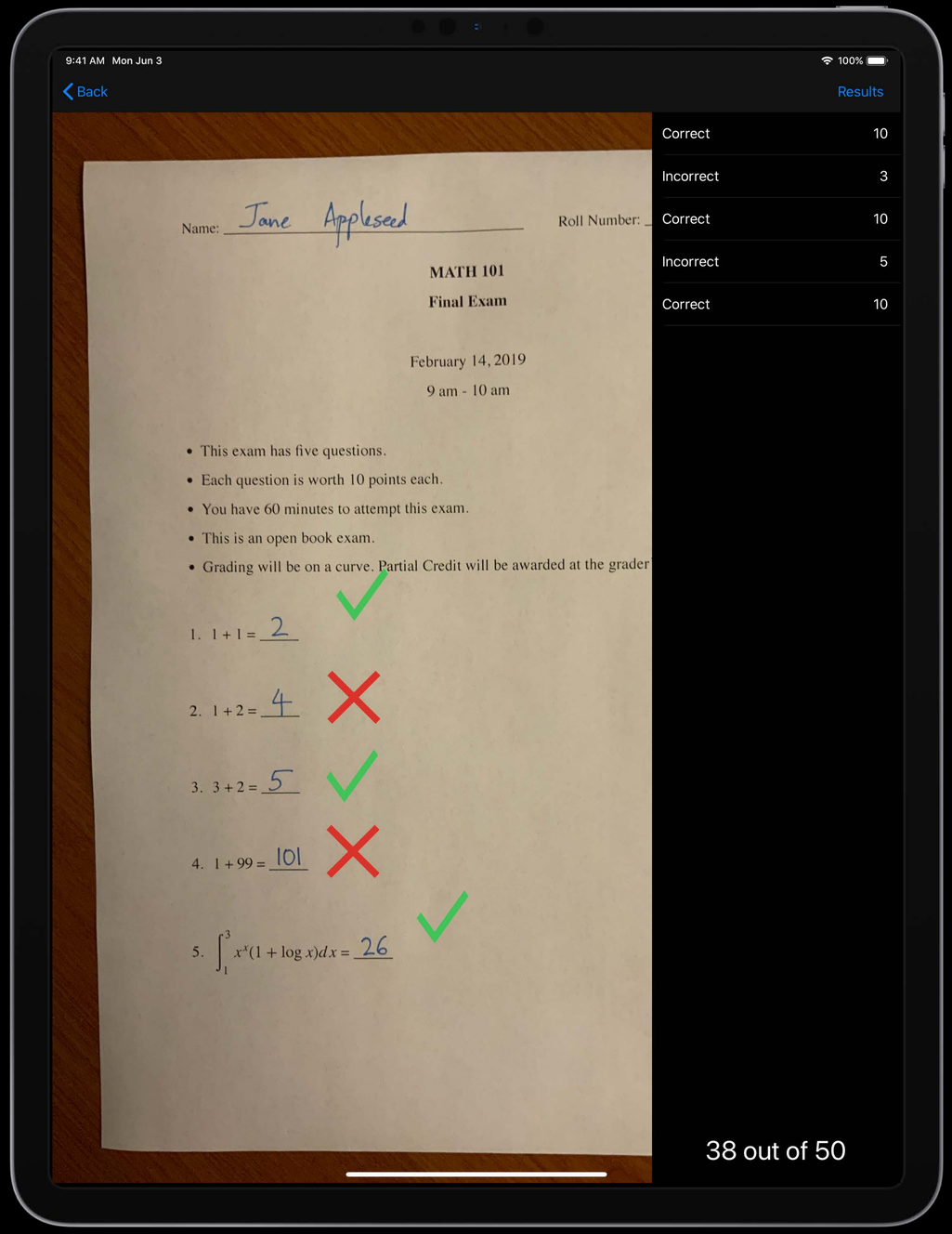
+ Add New Label

Jump to image

Extracting Features: 7%

Summary


9:41 AM Mon Jun 3
100%
Library
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%


9:41 AM Mon Jun 3
100%
Back
Results
Name: Jane Appleseed
Roll Number:
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
1. 1 + 1 = 2
2. 1 + 2 = 4
3. 3 + 2 = 5
4. 1 + 99 = 101
5. $\int_1^3 x^x(1 + \log x)dx$ = 26
Correct 10
Incorrect 3
Correct 10
Incorrect 5
Correct 10
38 out of 50

Overview

Model	Number of Iterations	Accuracy	Total Tested
Drawing Classifier	100	93.06%	1831

CLASS NAME	EXAMPLE IMAGES	ACCURACY	NUM TEST
-8		0%	102
-3		93%	73
-2		94%	72
-9		95%	58
-7		95%	85
not_equals		98%	95
question_mark		98%	58
-6		98%	63
-1		99%	94
contradiction		99%	104
cross		99%	109
-5		99%	111

127 Errors

Total: 1831

Hide Annotated image

Jump to image

Labels

ten 5

six 3 Apply

check 4 Apply

+ Add New Label

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

Introducing PencilKit	WWDC 2019
Vision and Core ML	WWDC 2018


WWDC19