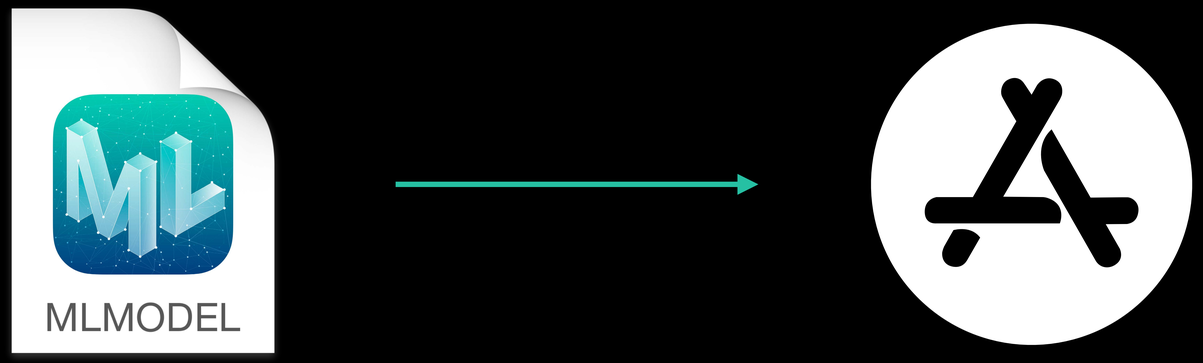
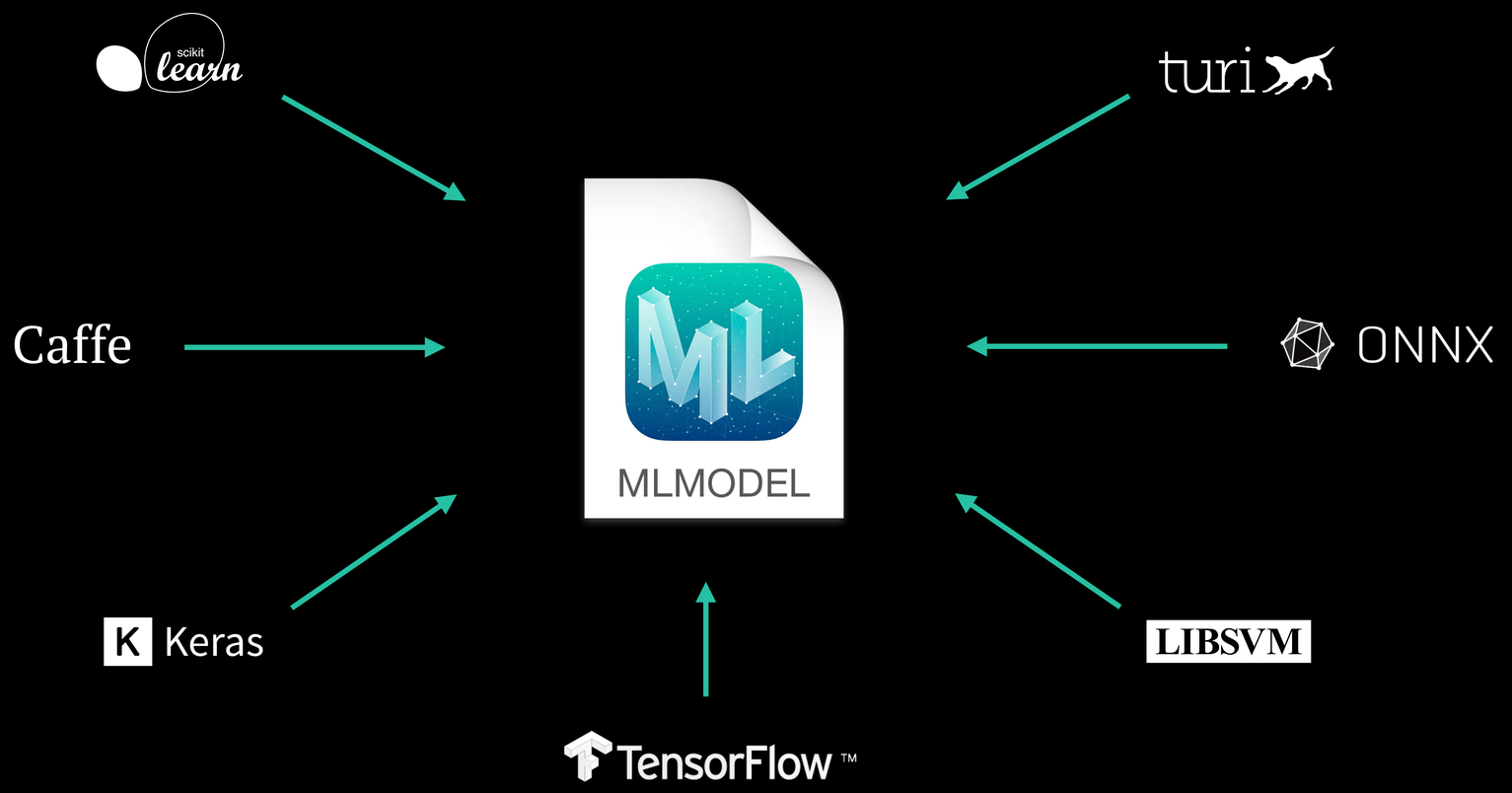
#WWDC19

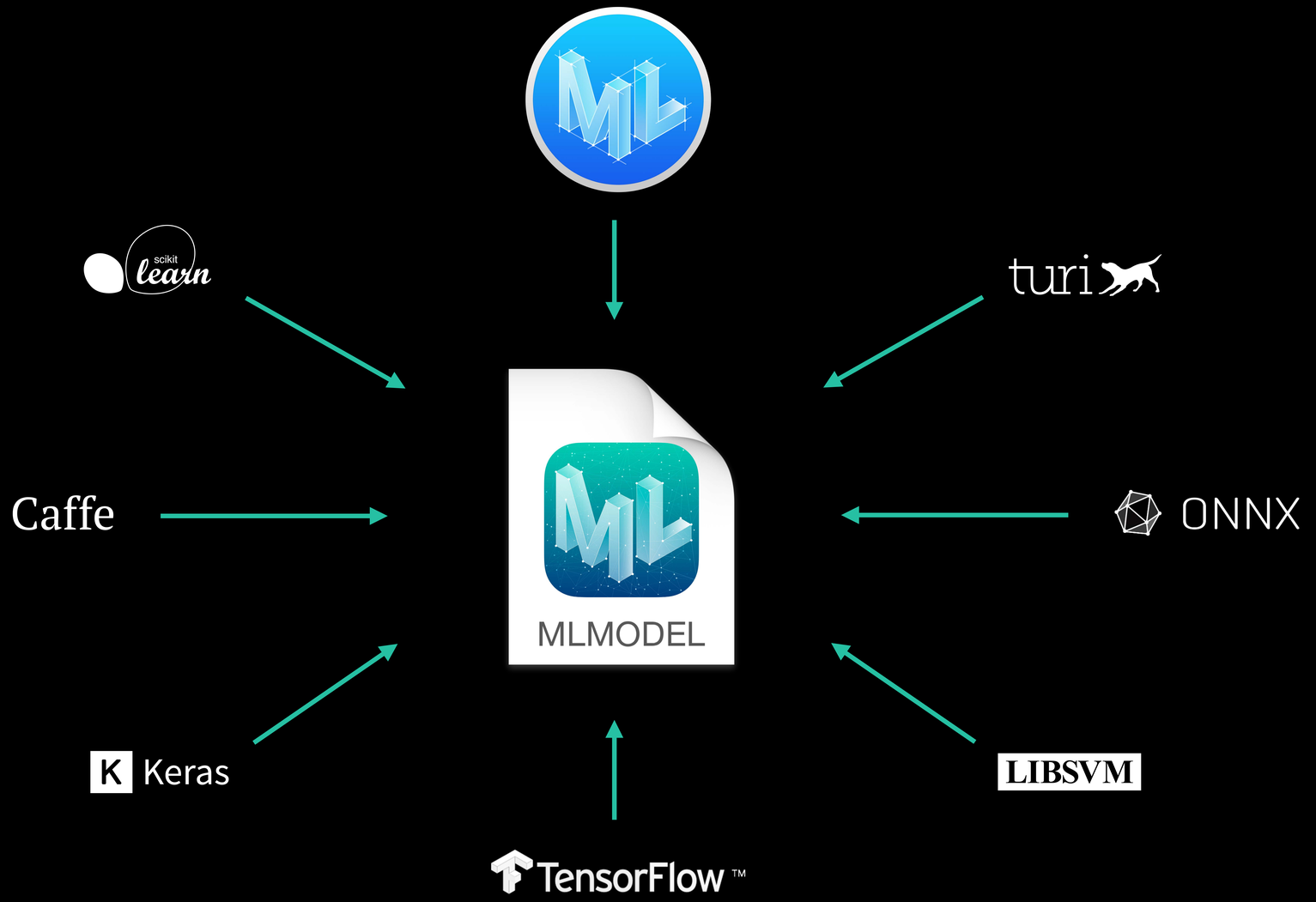
Core ML 3 Framework

Michael Brennan, Core ML
Anil Katti, Core ML
Aseem Wadhwa, Core ML
Allen Lin, Core ML


MLMODEL


MLMODEL


scikit learn
turi
Caffe
ONNX
MLMODEL
K Keras
LIBSVM
TensorFlow ™


scikit learn
turi
Caffe
ONNX
MLMODEL
Keras
LIBSVM
TensorFlow ™

On-Device Model Personalization

Neural Networks

Additional Updates

On-Device Model Personalization

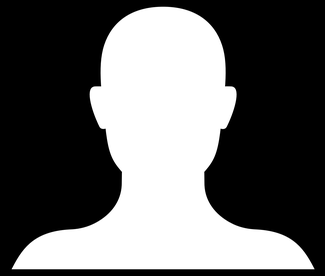
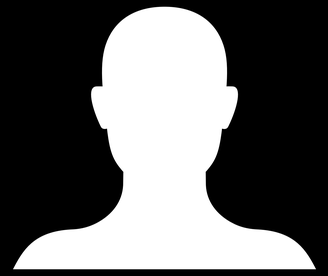
Models On-Device

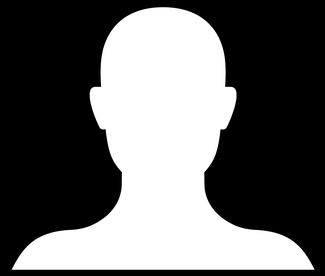
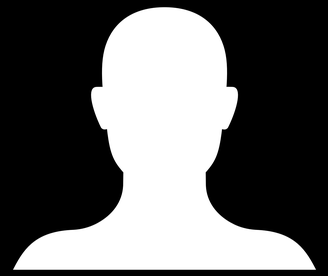
Bundled with or downloaded to app

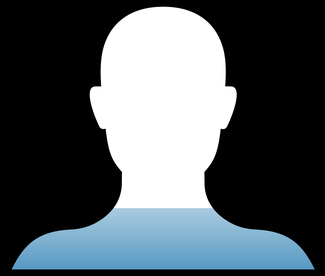
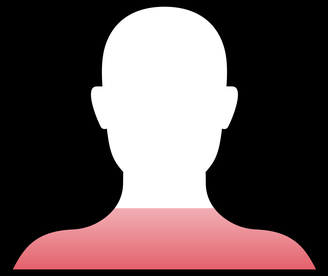
Static assets

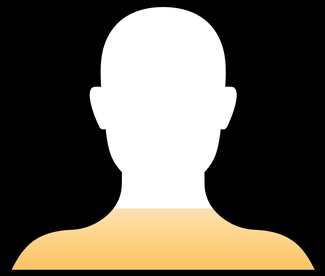
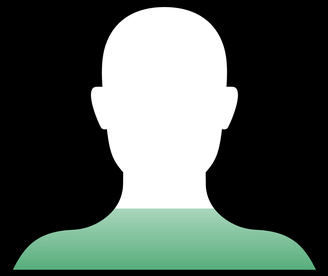
Optimized for prediction

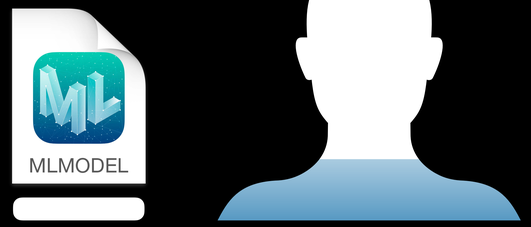

MLMODEL

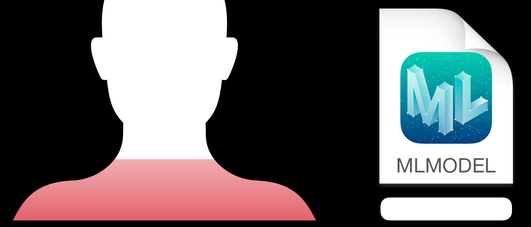



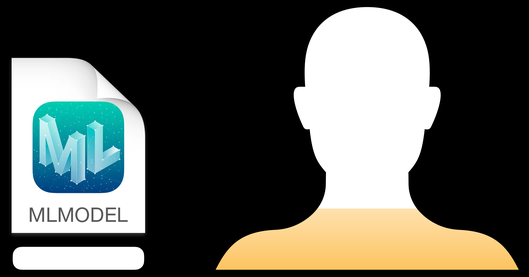
MLMODEL



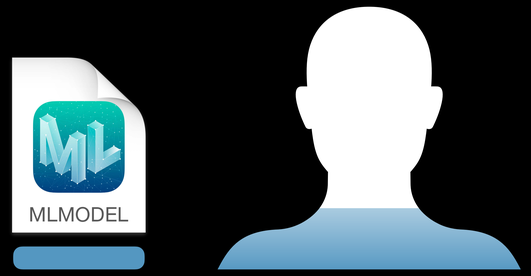
MLMODEL



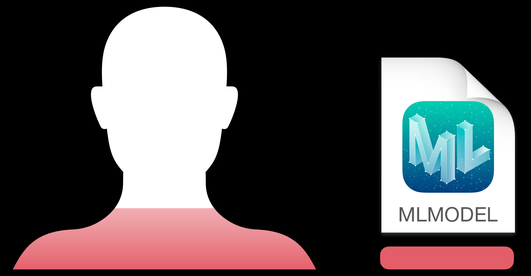

MLMODEL

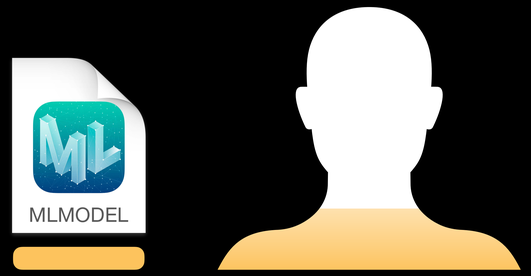

MLMODEL


MLMODEL

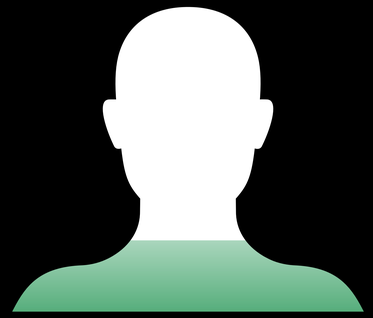

MLMODEL

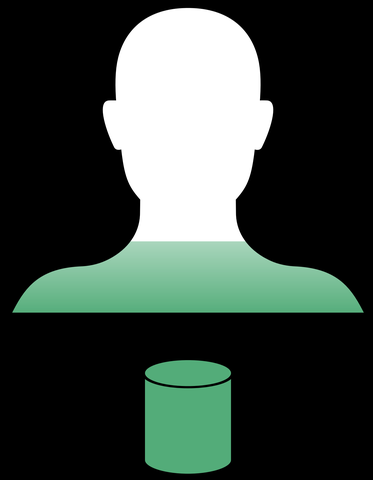

MLMODEL

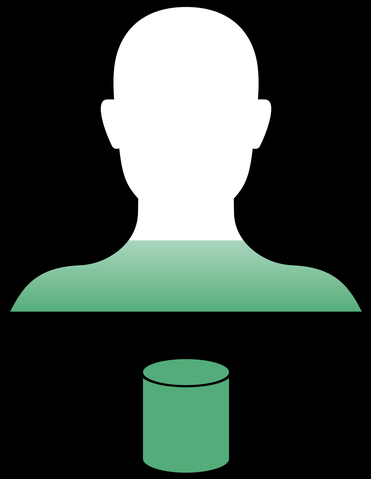

MLMODEL

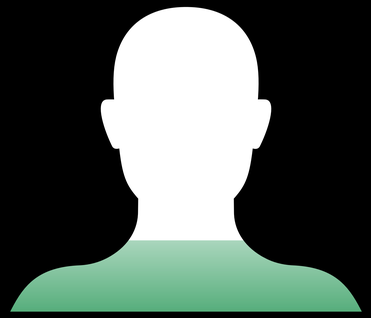

MLMODEL


MLMODEL

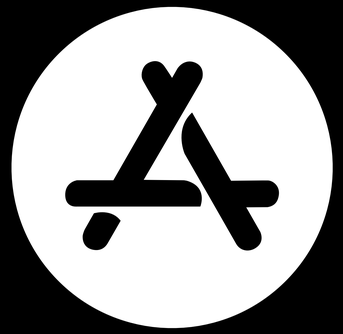


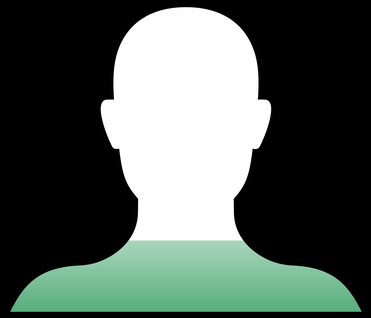






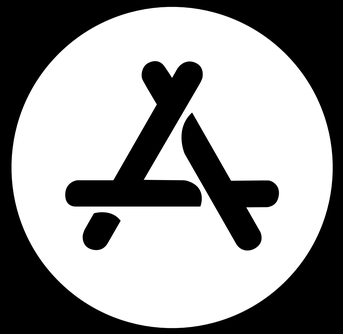

MLMODEL

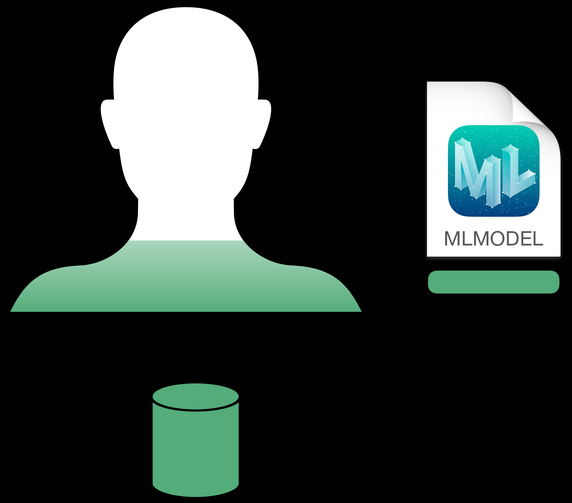





MLMODEL


MLMODEL




MLMODEL

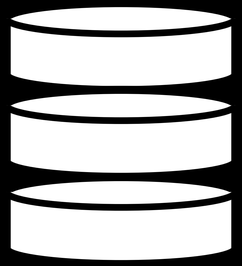
On-Device

On-Device

Privacy

On-Device

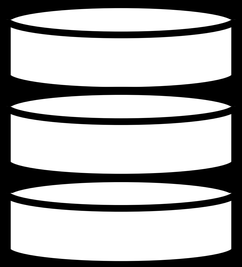
Privacy



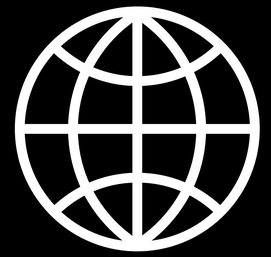
No server

On-Device

Privacy



No server

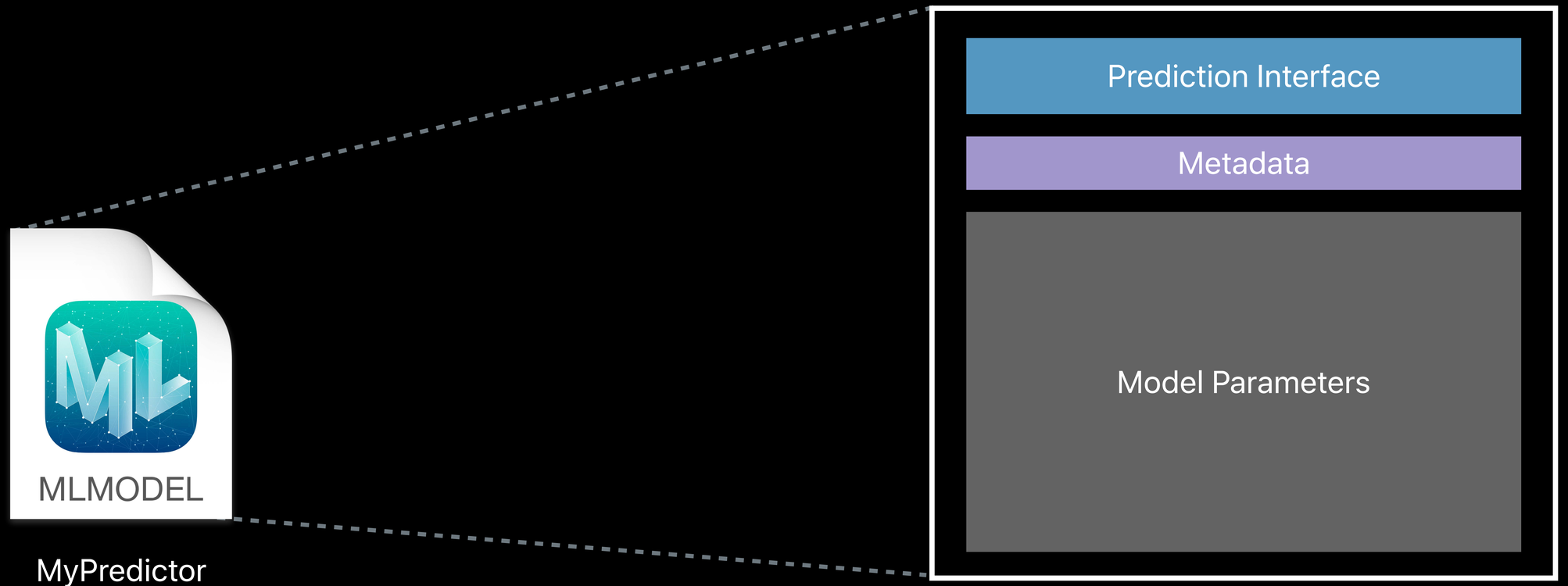


Available

Model Anatomy

MyPredictor

Model Anatomy



Updatable Model Anatomy

NEW

MyUpdatablePredictor

Prediction Interface

Update Interface

Metadata

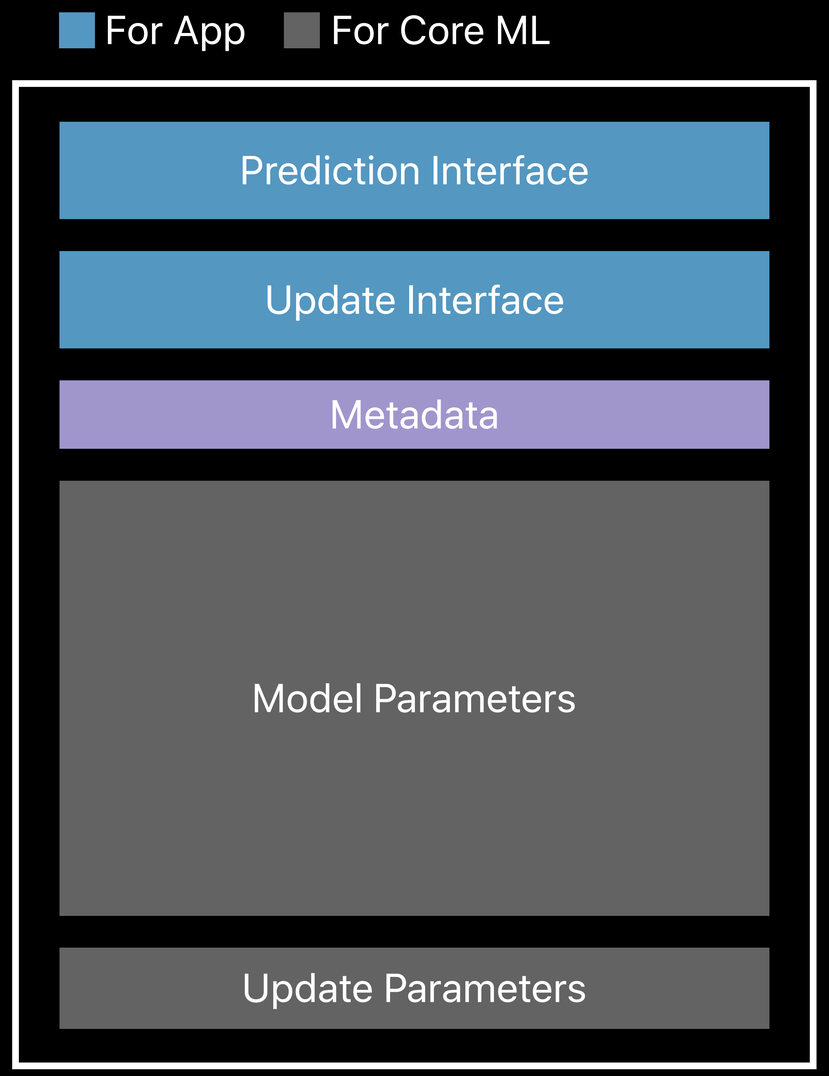
Model Parameters

Update Parameters

Encapsulation

Functionality accessed via Interface

All parameter values defined in model



Prediction

Update

NEW

Supported Model Types

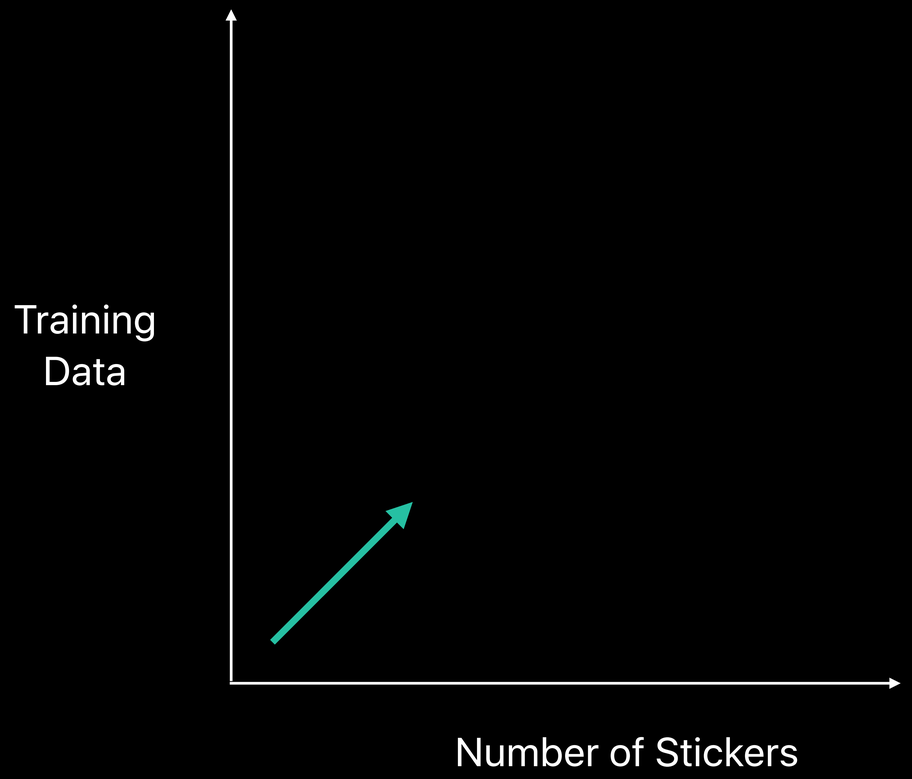
Nearest Neighbor Classifiers

Neural Networks

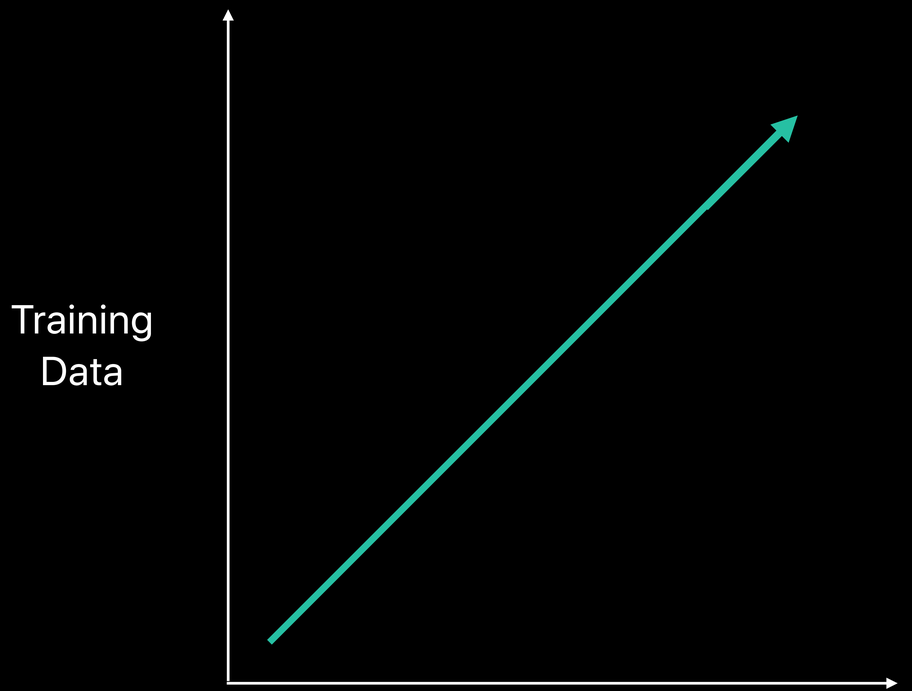
Pipeline Models

Use pre-trained sticker model?

Training Data

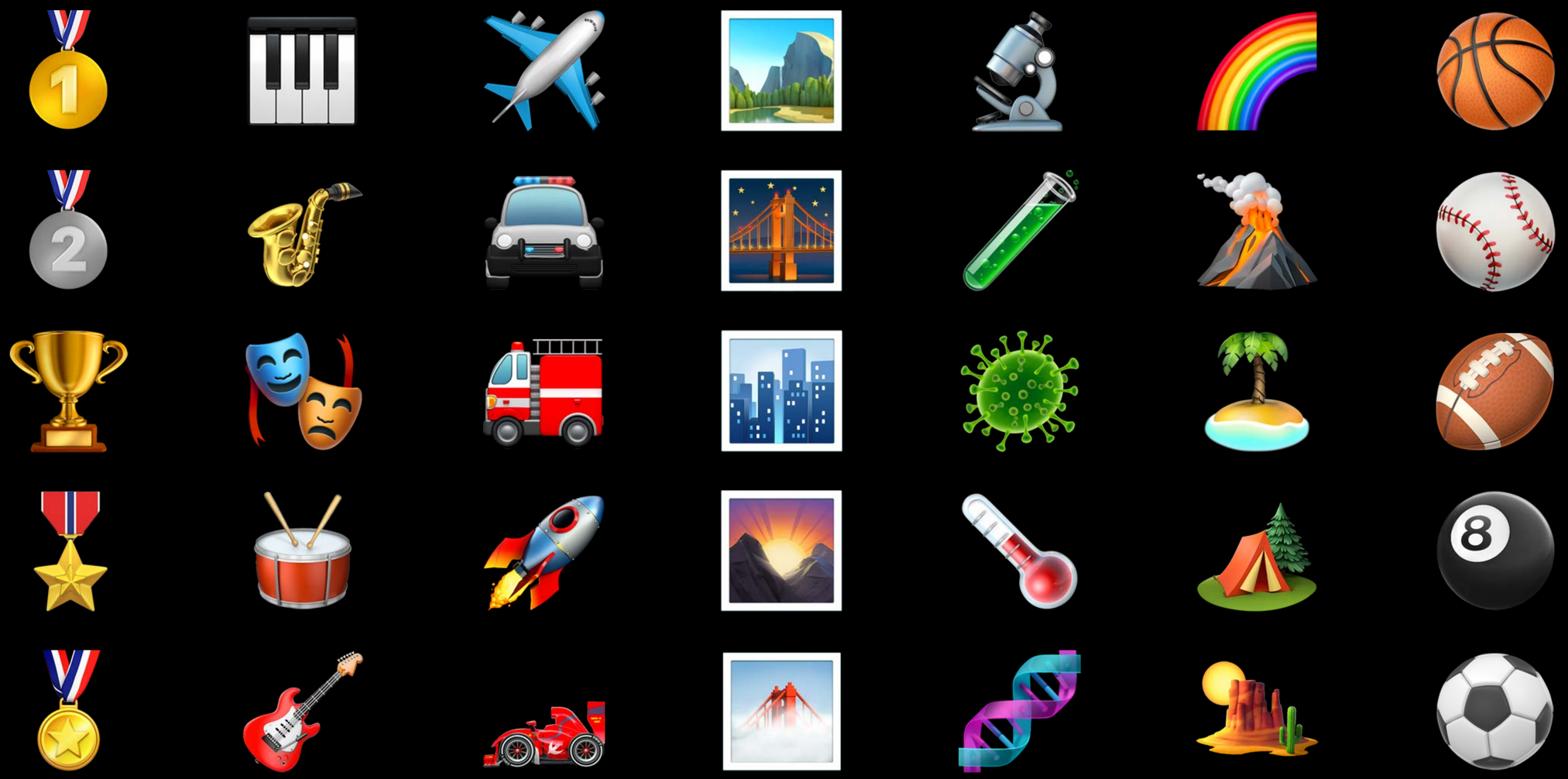


Training Data



Scale for New Stickers

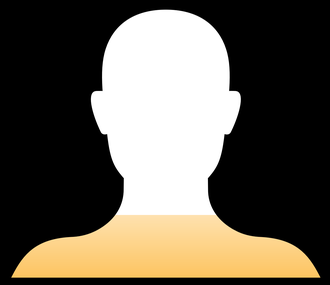
Scale for New Stickers

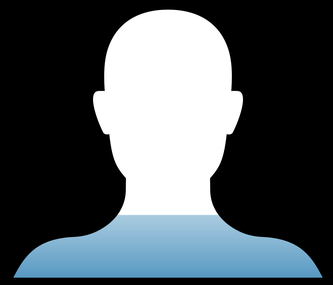


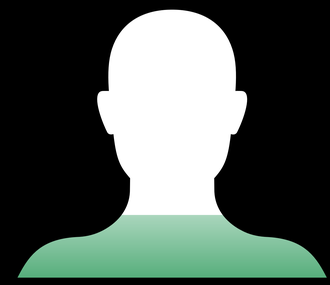
User Differences

User Differences

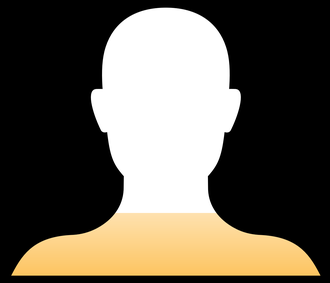
User Differences

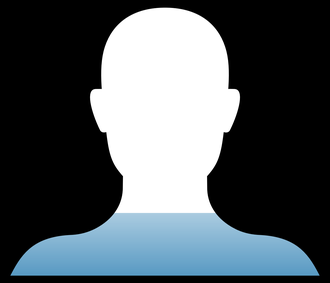


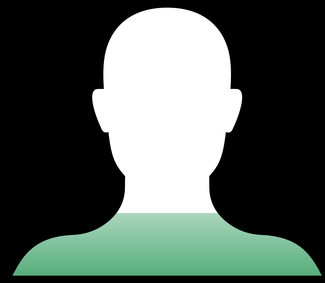




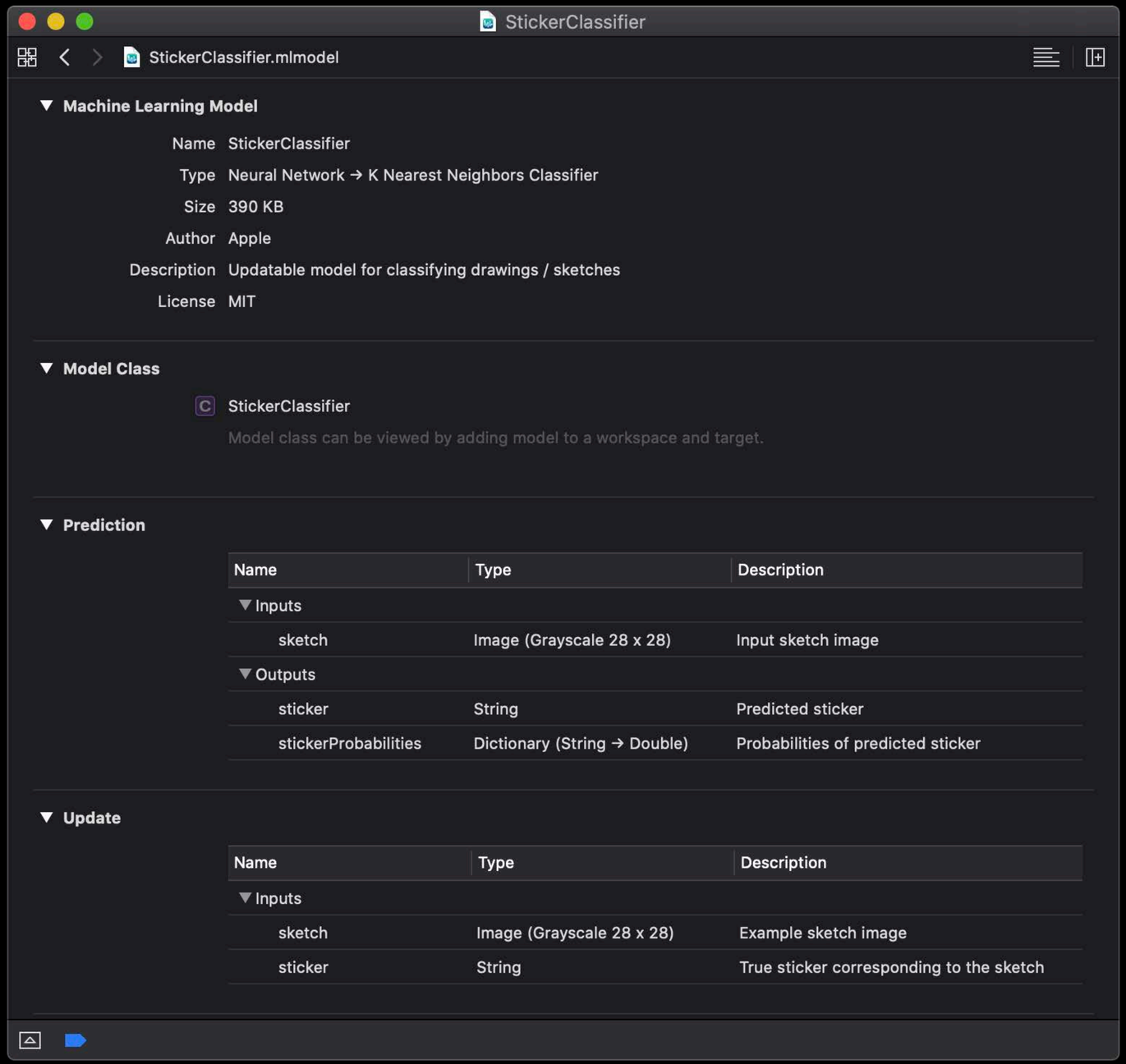
User Differences

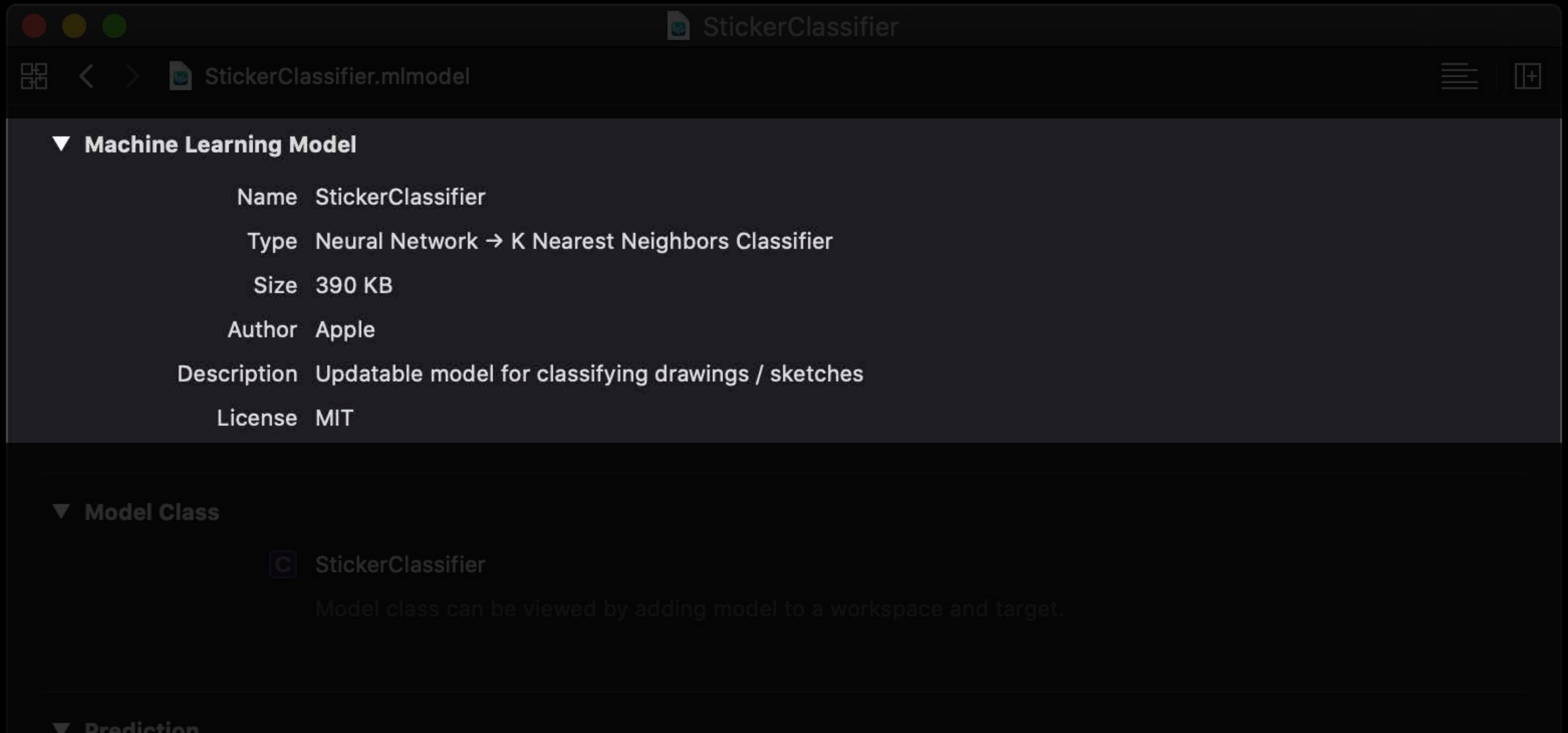


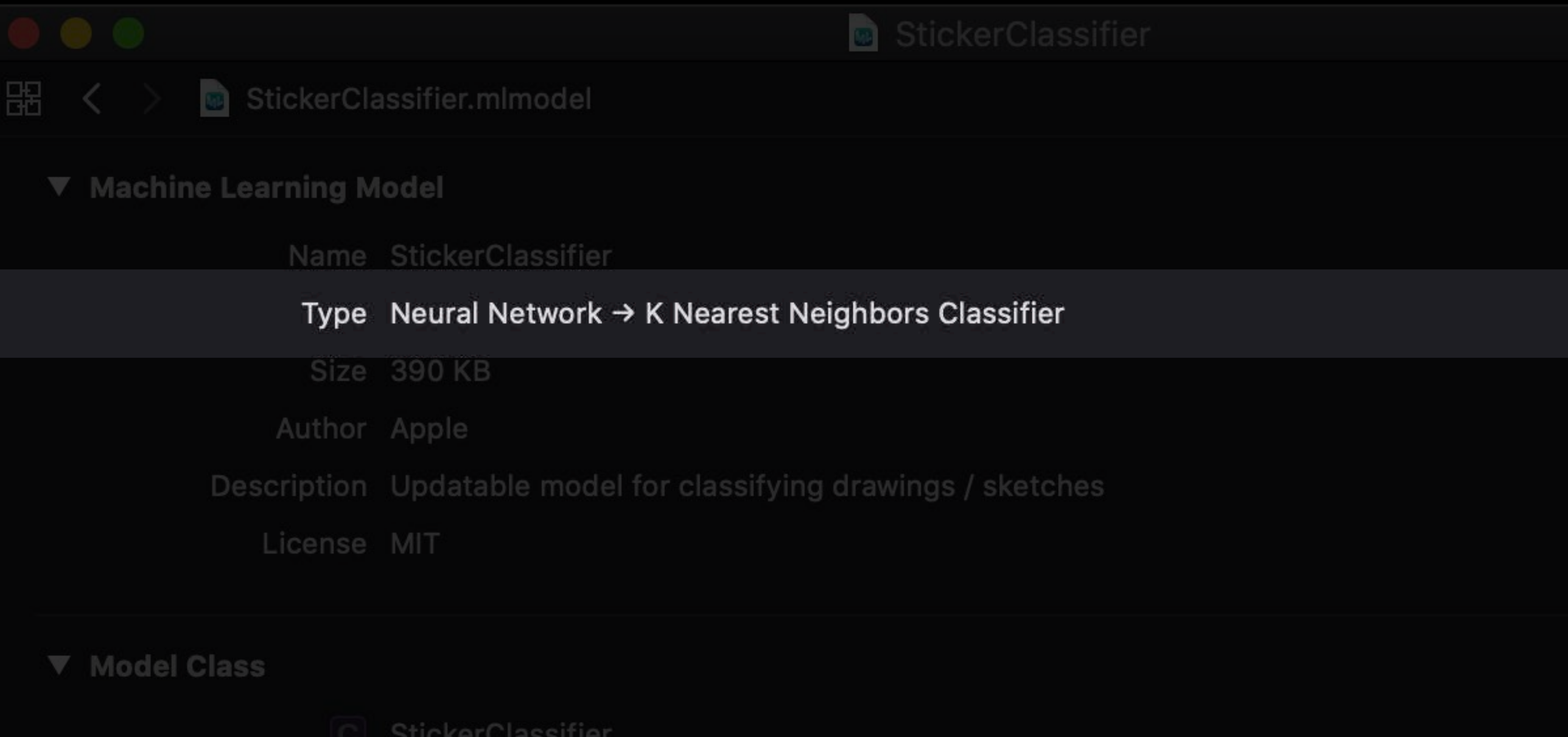




User can train the model


StickerClassifier
StickerClassifier.mlmodel
Machine Learning Model
Name StickerClassifier
Type Neural Network → K Nearest Neighbors Classifier
Size 390 KB
Author Apple
Description Updatable model for classifying drawings / sketches
License MIT
Model Class
StickerClassifier
Model class can be viewed by adding model to a workspace and target.
Prediction
Name | Type | Description
Inputs
sketch | Image (Grayscale 28 x 28) | Input sketch image
Outputs
sticker | String | Predicted sticker
stickerProbabilities | Dictionary (String → Double) | Probabilities of predicted sticker
Update
Name | Type | Description
Inputs
sketch | Image (Grayscale 28 x 28) | Example sketch image
sticker | String | True sticker corresponding to the sketch


StickerClassifier
StickerClassifier.mlmodel
Machine Learning Model
Name StickerClassifier
Type Neural Network → K Nearest Neighbors Classifier
Size 390 KB
Author Apple
Description Updatable model for classifying drawings / sketches
License MIT
Model Class
StickerClassifier
Model class can be viewed by adding model to a workspace and target.


StickerClassifier
StickerClassifier.mlmodel
Machine Learning Model
Name StickerClassifier
Type Neural Network → K Nearest Neighbors Classifier
Size 390 KB
Author Apple
Description Updatable model for classifying drawings / sketches
License MIT
Model Class

License MIT

Model Class

StickerClassifier

Model class can be viewed by adding model to a workspace and target.

Prediction

Name	Type	Description
Inputs		
sketch	Image (Grayscale 28 x 28)	Input sketch image
Outputs		
sticker	String	Predicted sticker
stickerProbabilities	Dictionary (String → Double)	Probabilities of predicted sticker

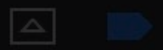
Update

Name	Type	Description
Inputs		
sketch	Image (Grayscale 28 x 28)	Example sketch image

Name	Type	Description
▼ Inputs		
sketch	Image (Grayscale 28 x 28)	Input sketch image
▼ Outputs		
sticker	String	Predicted sticker
stickerProbabilities	Dictionary (String → Double)	Probabilities of predicted sticker

▼ **Update**

Name	Type	Description
▼ Inputs		
sketch	Image (Grayscale 28 x 28)	Example sketch image
sticker	String	True sticker corresponding to the sketch



```
// Auto Generated Classes

// Model class
class StickerClassifier

// Prediction input
class StickerClassifierInput

// Prediction output
class StickerClassifierOutput
```

```
// Auto Generated Classes

// Model class
class StickerClassifier

// Prediction input
class StickerClassifierInput

// Prediction output
class StickerClassifierOutput

// Training input
class StickerClassifierTrainingInput
```

```
// Training input
class StickerClassifierTrainingInput
```

```
// Training input
class StickerClassifierTrainingInput : MLFeatureProvider {

    /// Example sketch image as grayscale image buffer, 28 pixels wide by 28 pixels high
    var sketch: CVPixelBuffer

    /// True sticker corresponding to the sketch as string value
    var sticker: String

    ...
}
```

```swift
// 1. Get updatable model URL
let bundle = Bundle(for: StickerClassifier.self)
let updatableModelURL = bundle.url(forResource: "StickerClassifier",
                                   withExtension: "mlmodelc")!

// 2. Prepare training data
let trainingData = prepareTrainingData(from: trainingSamples)

// 3. Kick off update task
let updateTask = try MLUpdateTask(forModelAt: updatableModelURL,
                                  trainingData: trainingData,
                                  configuration: configuration,
                                  completionHandler: {
                                      context in
                                      // Use updated model for predictions
                                      self.stickerClassifier.model = context.model
                                  })
updateTask.resume()
```

```swift
// 1. Get updatable model URL
let bundle = Bundle(for: StickerClassifier.self)
let updatableModelURL = bundle.url(forResource: "StickerClassifier",
                                   withExtension: "mlmodelc")!

// 2. Prepare training data
let trainingData = prepareTrainingData(from: trainingSamples)

// 3. Kick off update task
let updateTask = try MLUpdateTask(forModelAt: updatableModelURL,
                                  trainingData: trainingData,
                                  configuration: configuration,
                                  completionHandler: {
                                      context in
                                      // Use updated model for predictions
                                      self.stickerClassifier.model = context.model
                                  })
updateTask.resume()
```

```swift
// 1. Get updatable model URL
let bundle = Bundle(for: StickerClassifier.self)
let updatableModelURL = bundle.url(forResource: "StickerClassifier",
                                   withExtension: "mlmodelc")!

// 2. Prepare training data
let trainingData = prepareTrainingData(from: trainingSamples)

// 3. Kick off update task
let updateTask = try MLUpdateTask(forModelAt: updatableModelURL,
                                  trainingData: trainingData,
                                  configuration: configuration,
                                  completionHandler: {
                                      context in
                                      // Use updated model for predictions
                                      self.stickerClassifier.model = context.model
                                  })
updateTask.resume()
```

```swift
// 1. Get updatable model URL
let bundle = Bundle(for: StickerClassifier.self)
let updatableModelURL = bundle.url(forResource: "StickerClassifier",
                                   withExtension: "mlmodelc")!

// 2. Prepare training data
let trainingData = prepareTrainingData(from: trainingSamples)

// 3. Kick off update task
let updateTask = try MLUpdateTask(forModelAt: updatableModelURL,
                                  trainingData: trainingData,
                                  configuration: configuration,
                                  completionHandler: {
                                      context in
                                      // Use updated model for predictions
                                      self.stickerClassifier.model = context.model
                                  })
updateTask.resume()
```

```swift
// 1. Get updatable model URL
let bundle = Bundle(for: StickerClassifier.self)
let updatableModelURL = bundle.url(forResource: "StickerClassifier",
                                   withExtension: "mlmodelc")!

// 2. Prepare training data
let trainingData = prepareTrainingData(from: trainingSamples)

// 3. Kick off update task
let updateTask = try MLUpdateTask(forModelAt: updatableModelURL,
                                  trainingData: trainingData,
                                  configuration: configuration,
                                  completionHandler: {
                                      context in
                                      // Use updated model for predictions
                                      self.stickerClassifier.model = context.model
                                  })

updateTask.resume()
```

Recap

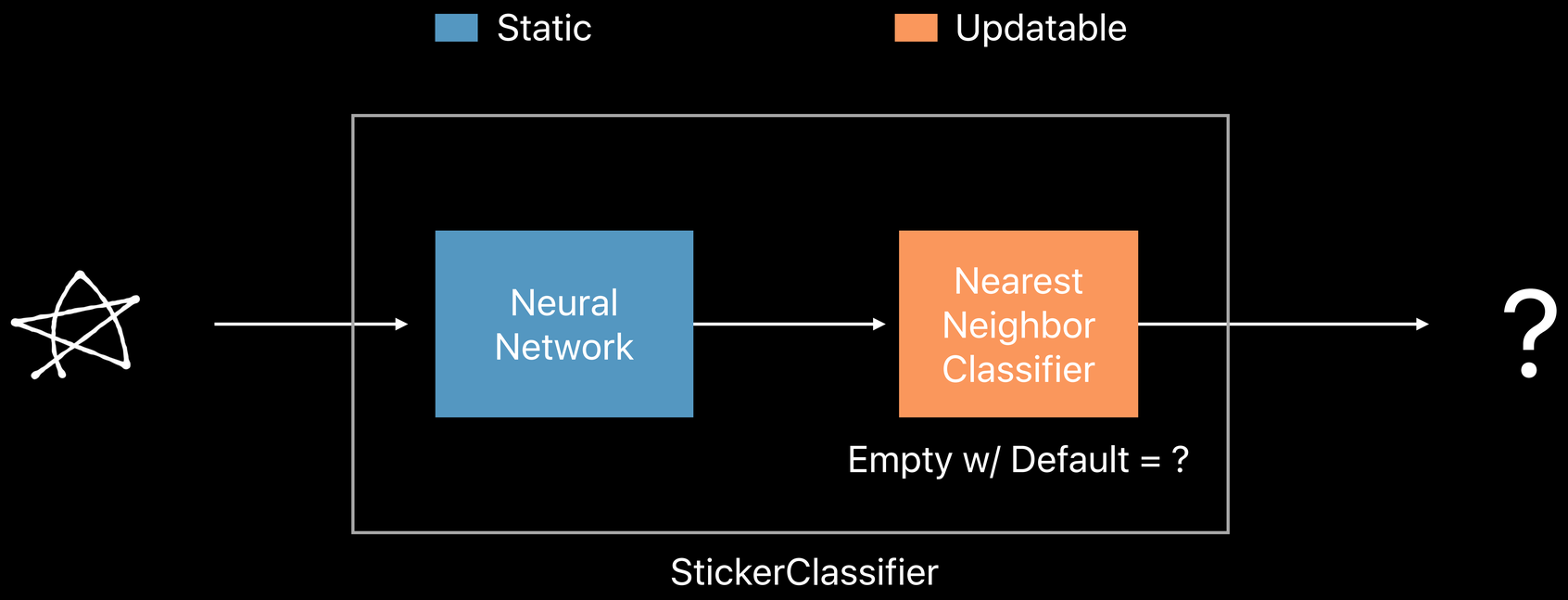
Recap

StickerClassifier

Recap

StickerClassifier

Initial Model



Update Task

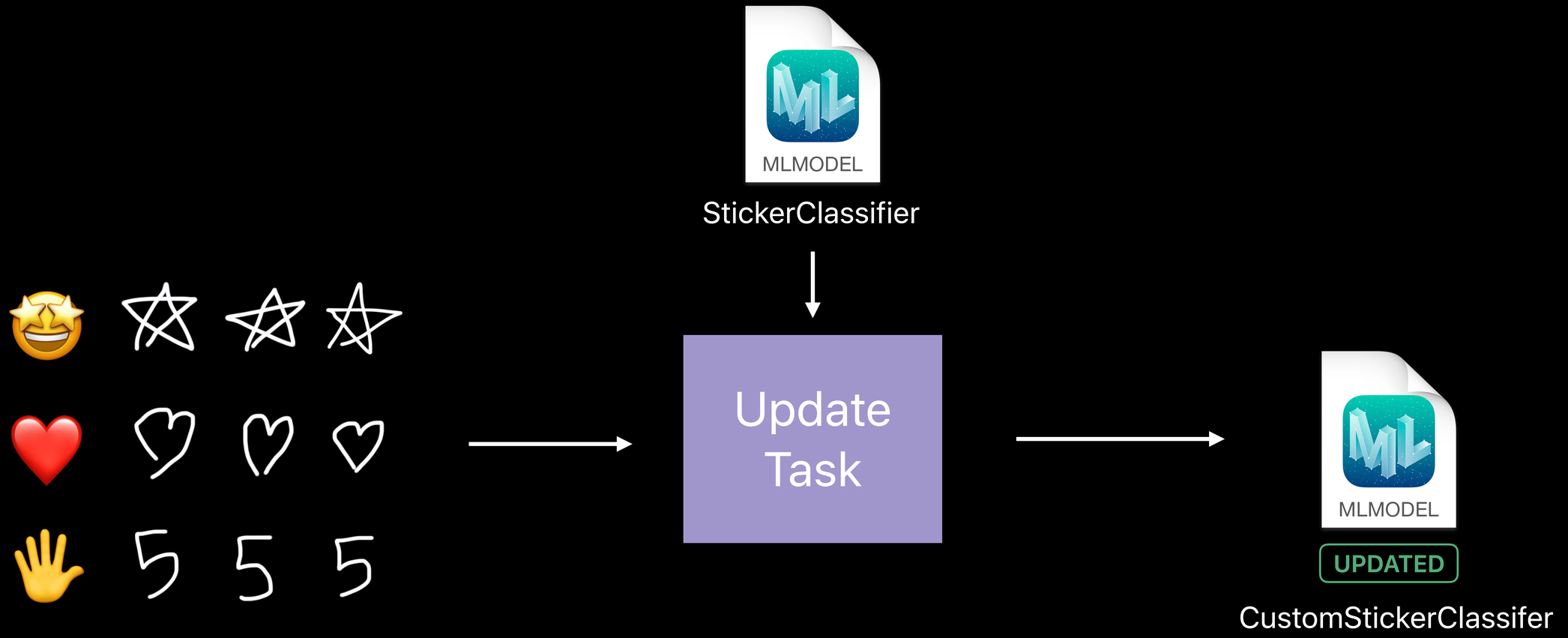
StickerClassifier

Update Task

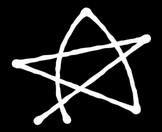
UPDATED

CustomStickerClassifer

Update Task


MLMODEL
StickerClassifier
Update Task
MLMODEL
UPDATED
CustomStickerClassifer

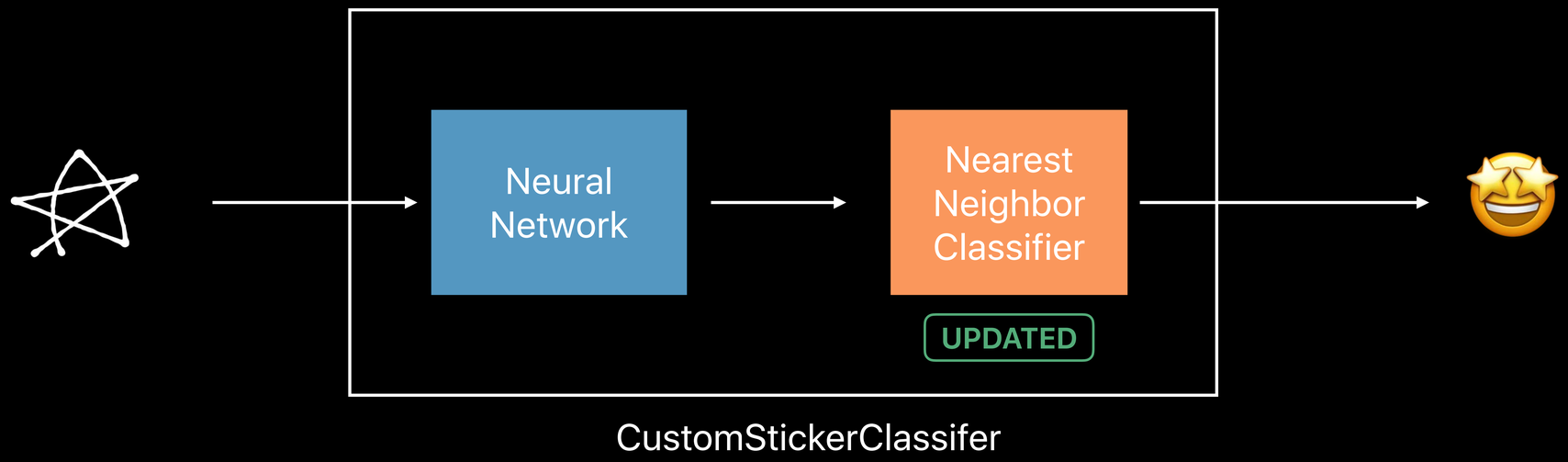
Predictions with Updated Model



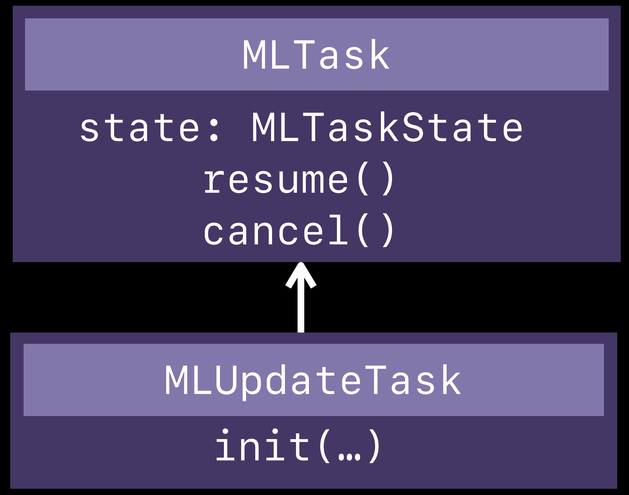
CustomStickerClassifer

Predictions with Updated Model

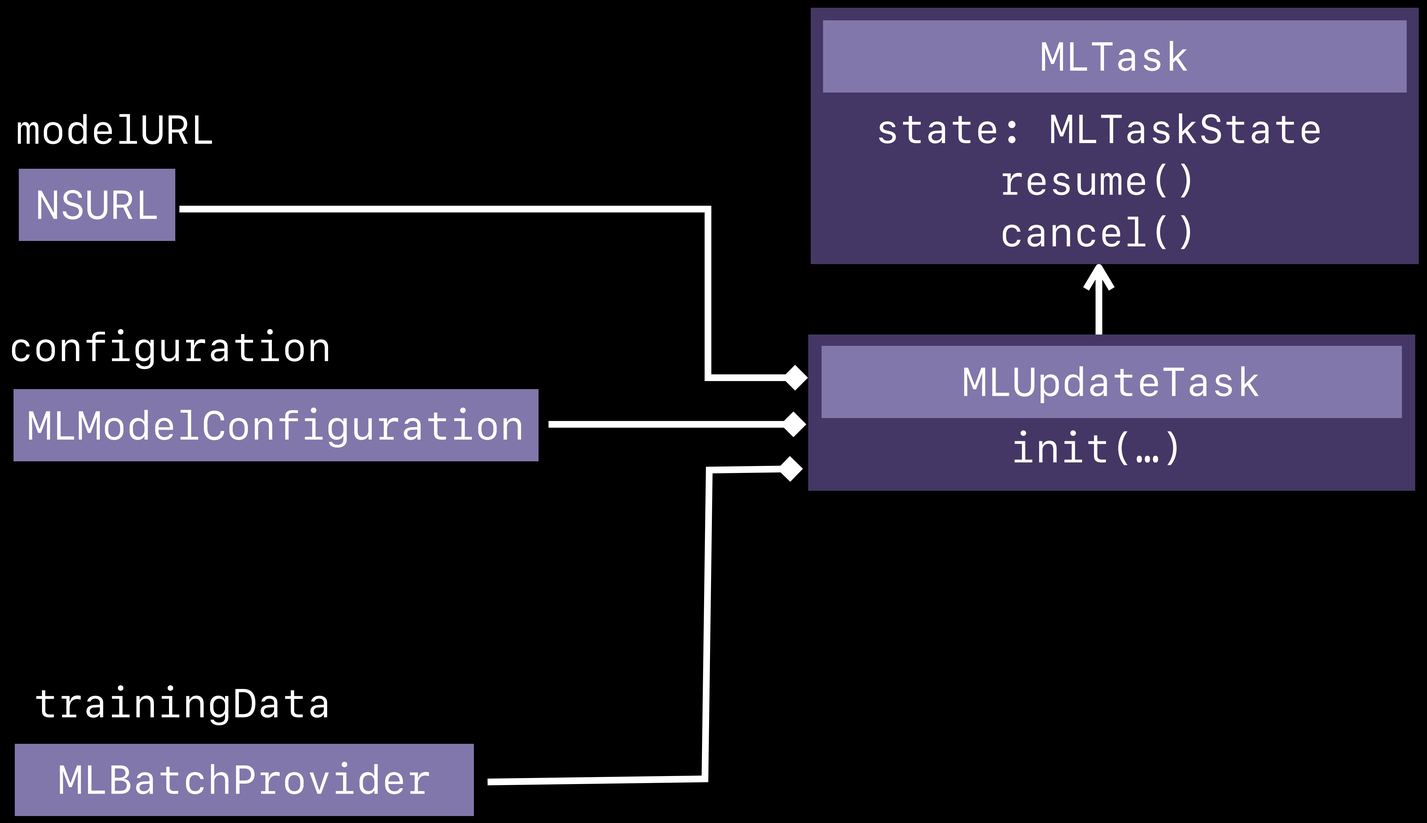
Predictions with Updated Model



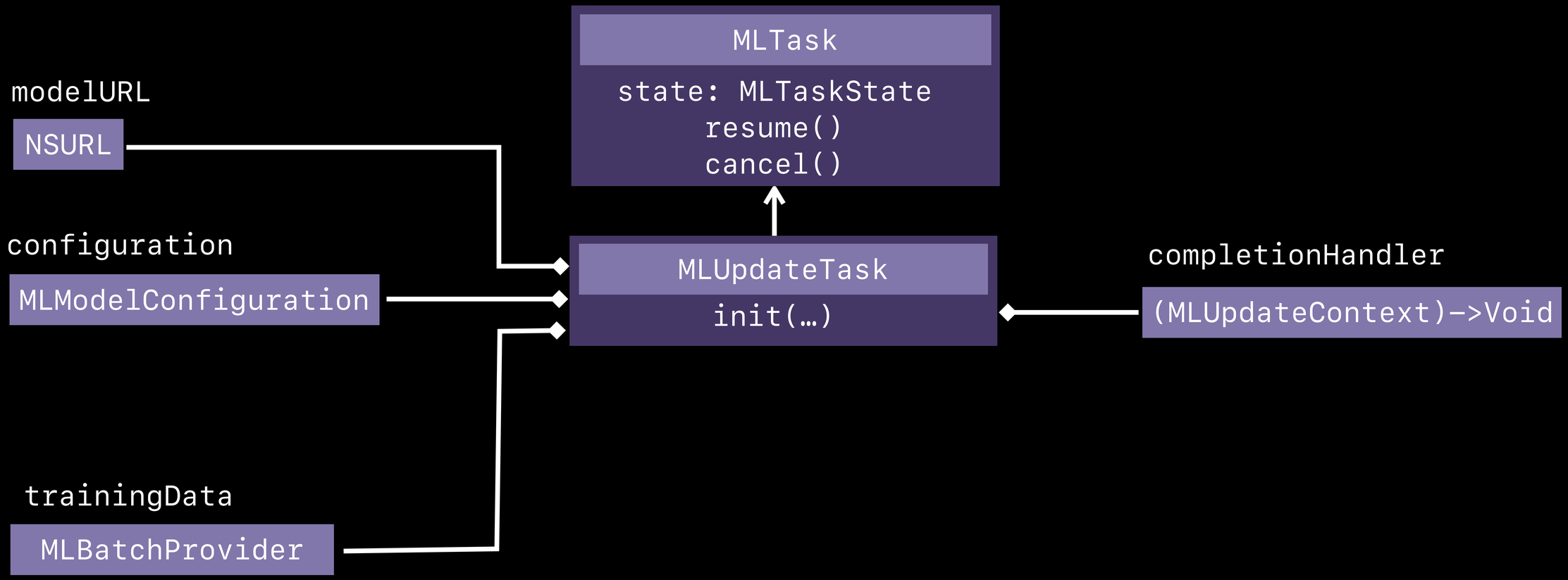
Update Tasks



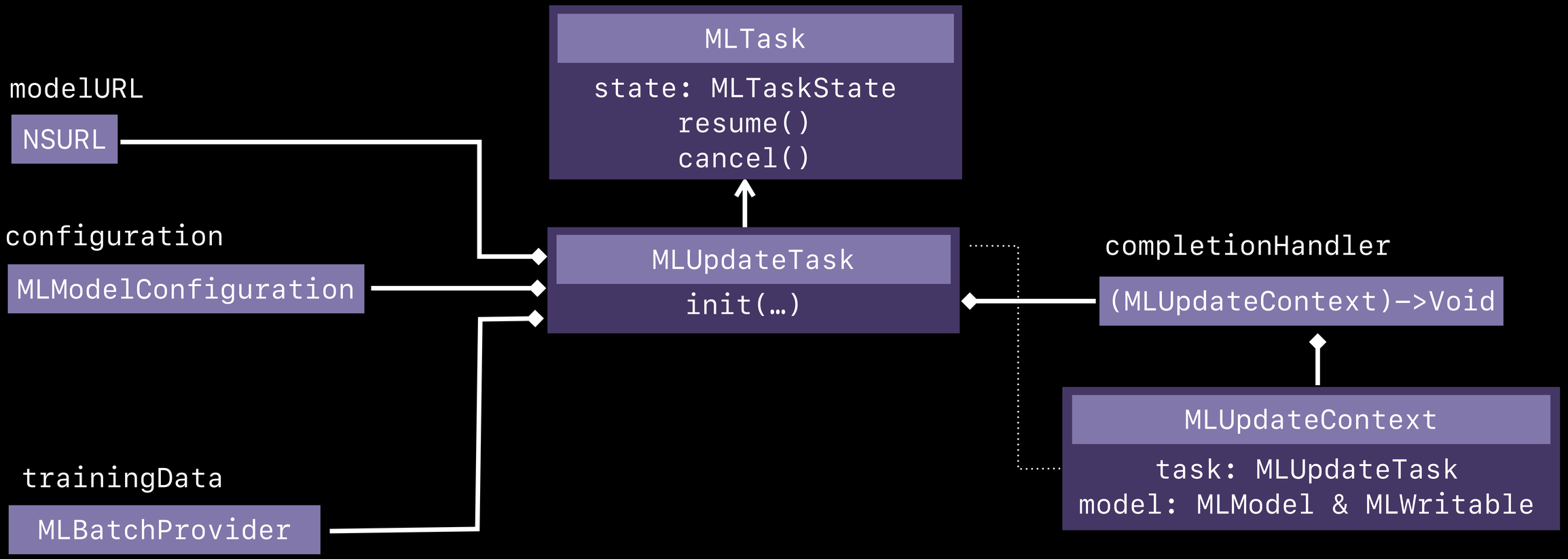
Update Tasks



Update Tasks



Update Tasks

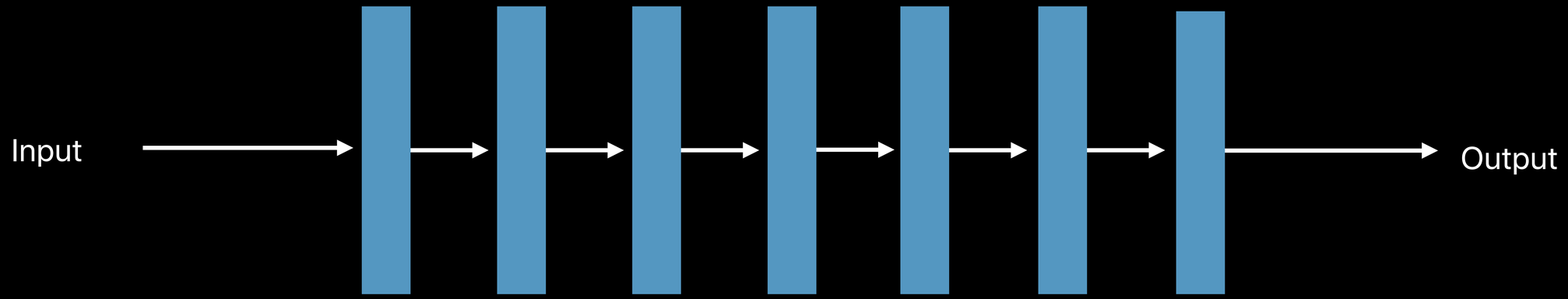


```
// Updating your MLModel

let updateTask = try MLUpdateTask(forModelAt: updatableModelURL,
                                  trainingData: trainingData,
                                  configuration: configuration,
                                  completionHandler: {
                                      context in
                                      computeAccuracy(forModel: context.model)
                                      context.model.write(to: updatedModelURL)
                                      self.stickerClassifier.model = context.model
                                  })
```

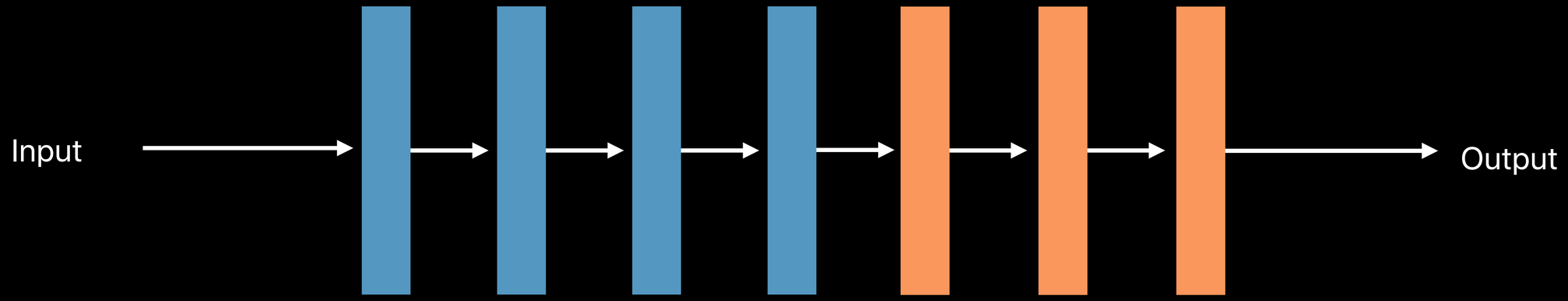
```
// Updating your MLModel

let updateTask = try MLUpdateTask(forModelAt: updatableModelURL,
                                  trainingData: trainingData,
                                  configuration: configuration,
                                  completionHandler: {
                                      context in
                                      computeAccuracy(forModel: context.model)
                                      context.model.write(to: updatedModelURL)
                                      self.stickerClassifier.model = context.model
                                  })
```

Updatable Neural Networks

Updatable Neural Networks



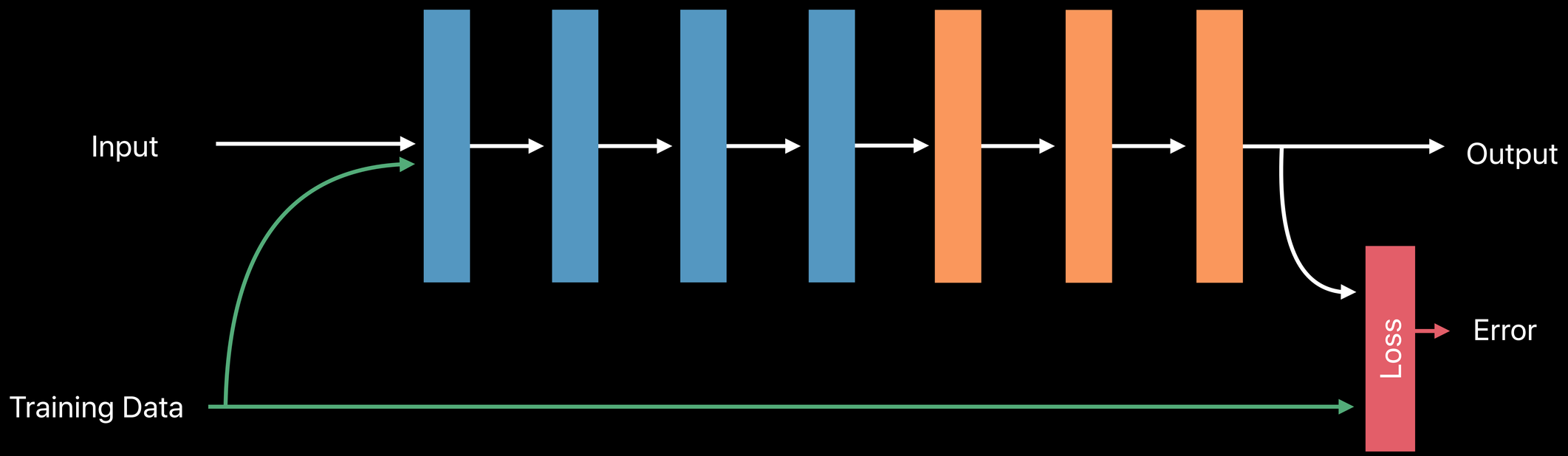
Updatable Neural Networks



Static

Updatable

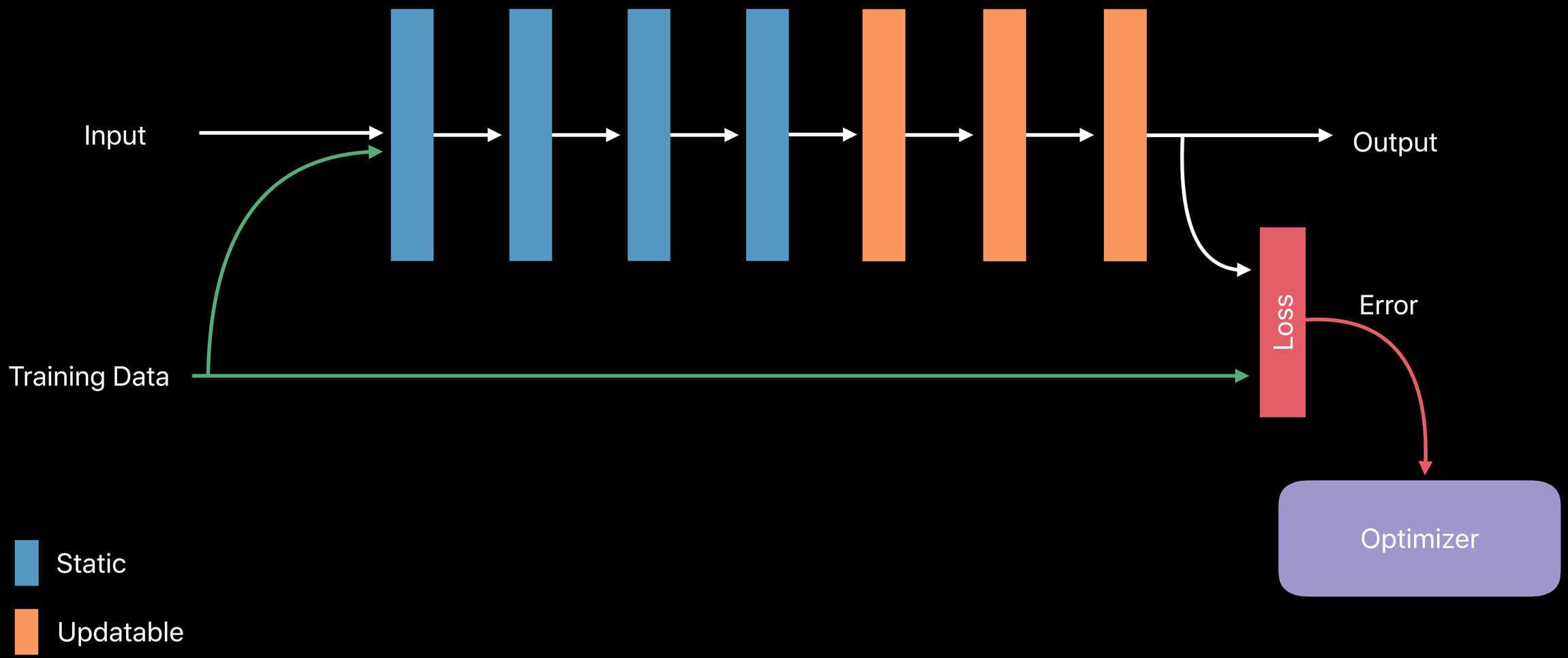
Updatable Neural Networks



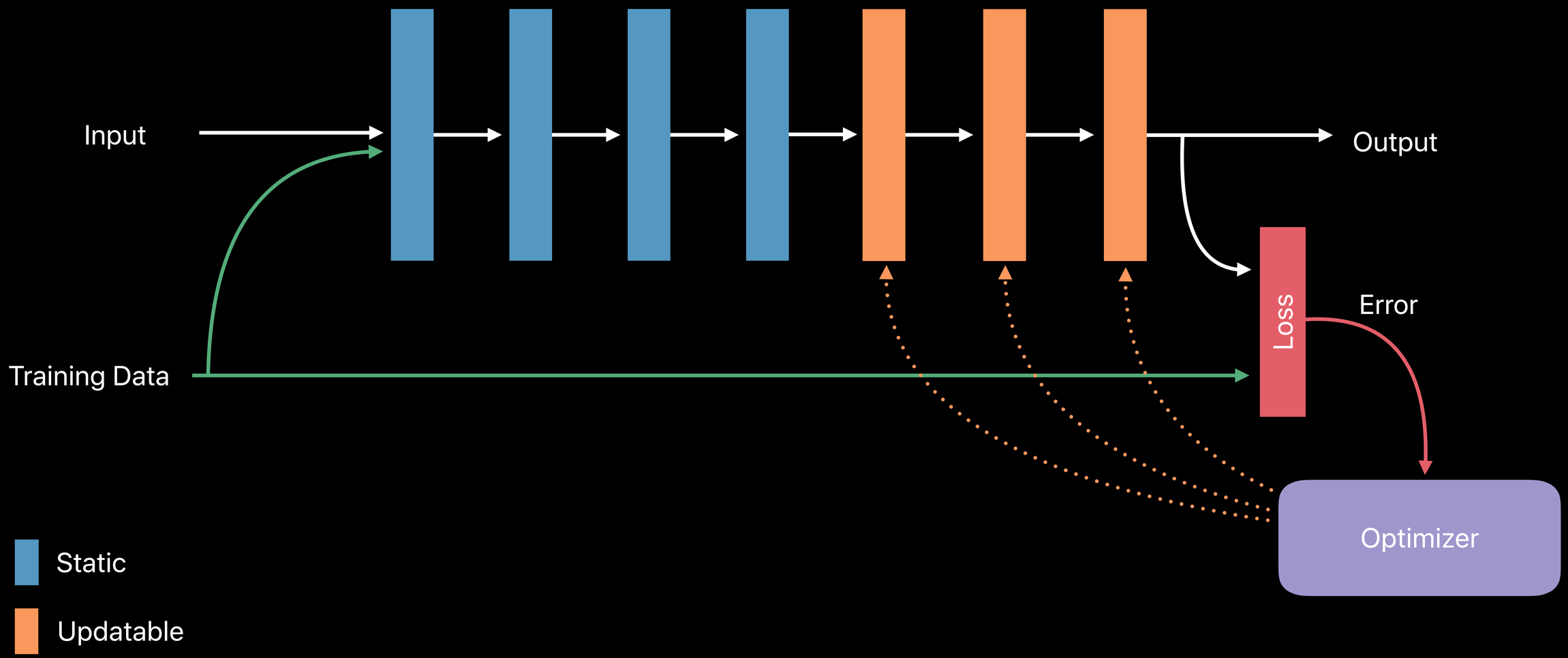
Static

Updatable

Updatable Neural Networks



Updatable Neural Networks



Updatable Neural Networks

Updatable Layers

Convolution

Fully Connected

Losses

Categorical Cross Entropy

Mean Squared Error

Optimizers

Stochastic Gradient Descent

Adam

*Can backprop through many more

Neural Network Optimizer Parameters

Encapsulated in model

Neural Network Optimizer Parameters

Encapsulated in model

Optionally changed at runtime

Neural Network Optimizer Parameters

Encapsulated in model

Optionally changed at runtime

```
MLModelConfiguration.updateParameters : [MLParameterKey : Any]
```

Neural Network Optimizer Parameters

Encapsulated in model

Optionally changed at runtime

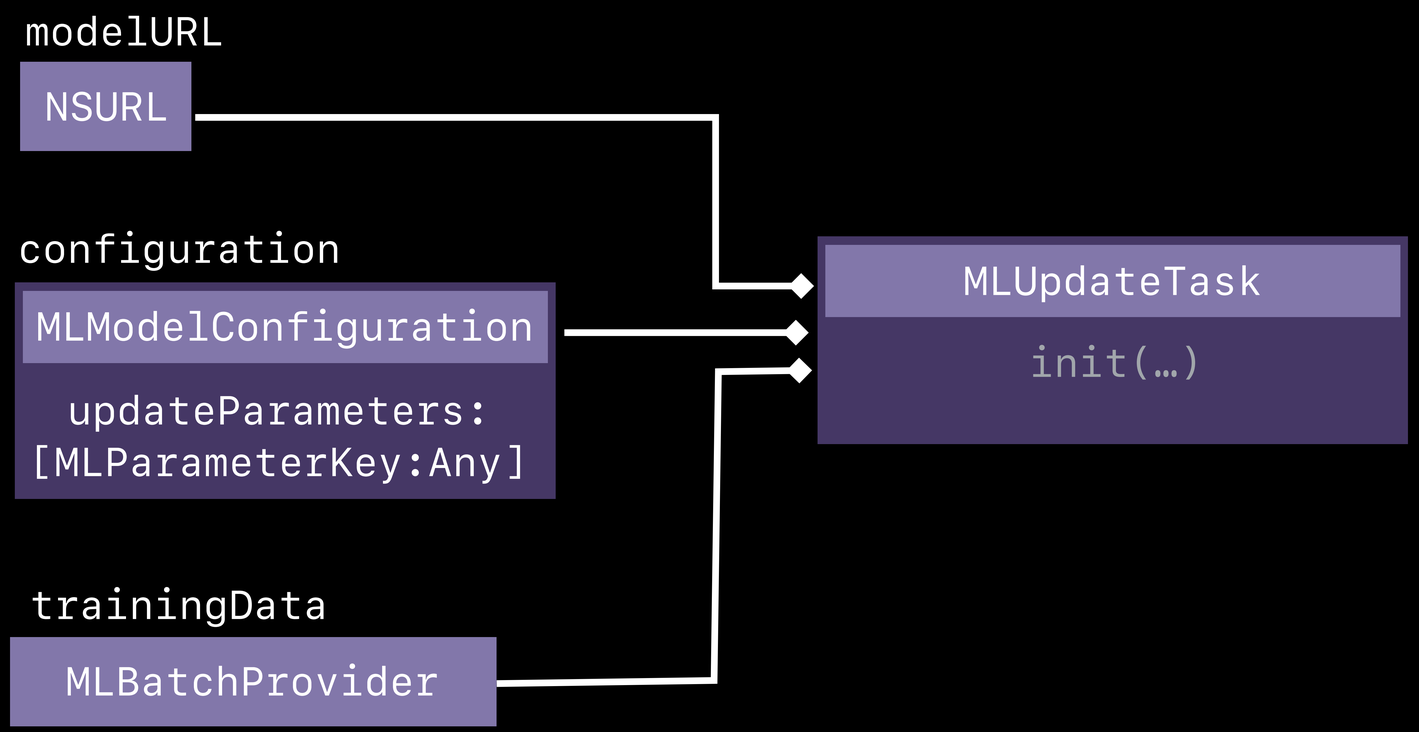
```
MLModelConfiguration.updateParameters : [MLParameterKey : Any]

MLParameterKey
```

.epochs	.learningRate	.eps	
.miniBatchSize	.momentum	.beta1	.beta2

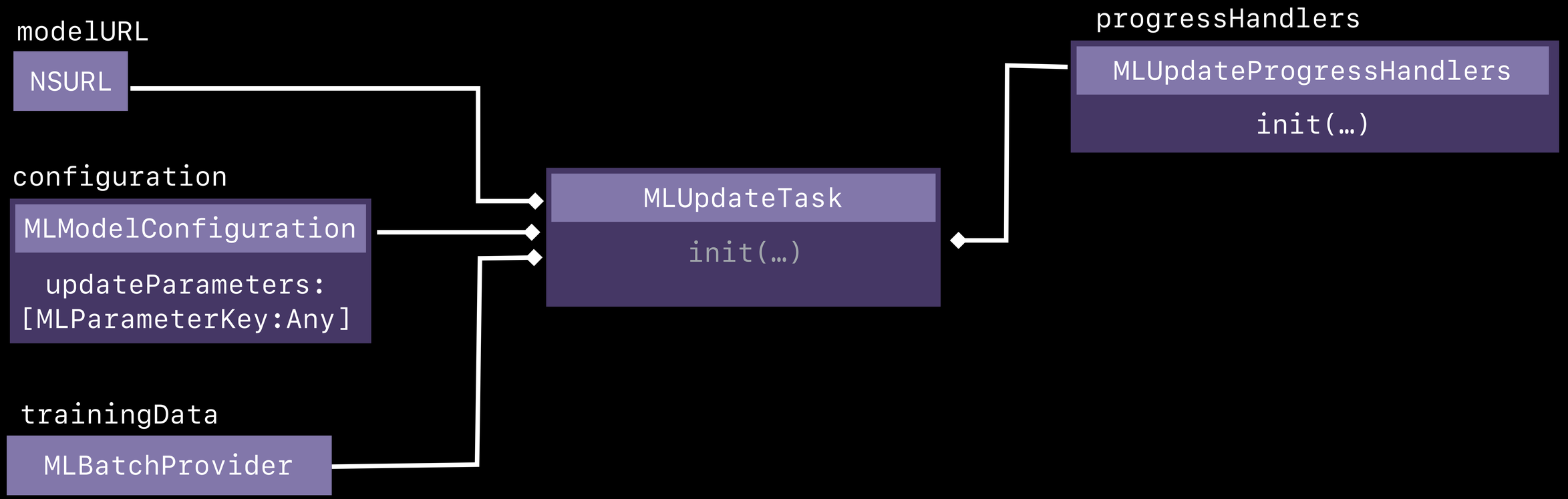
Parameters in Xcode

Parameters in Xcode

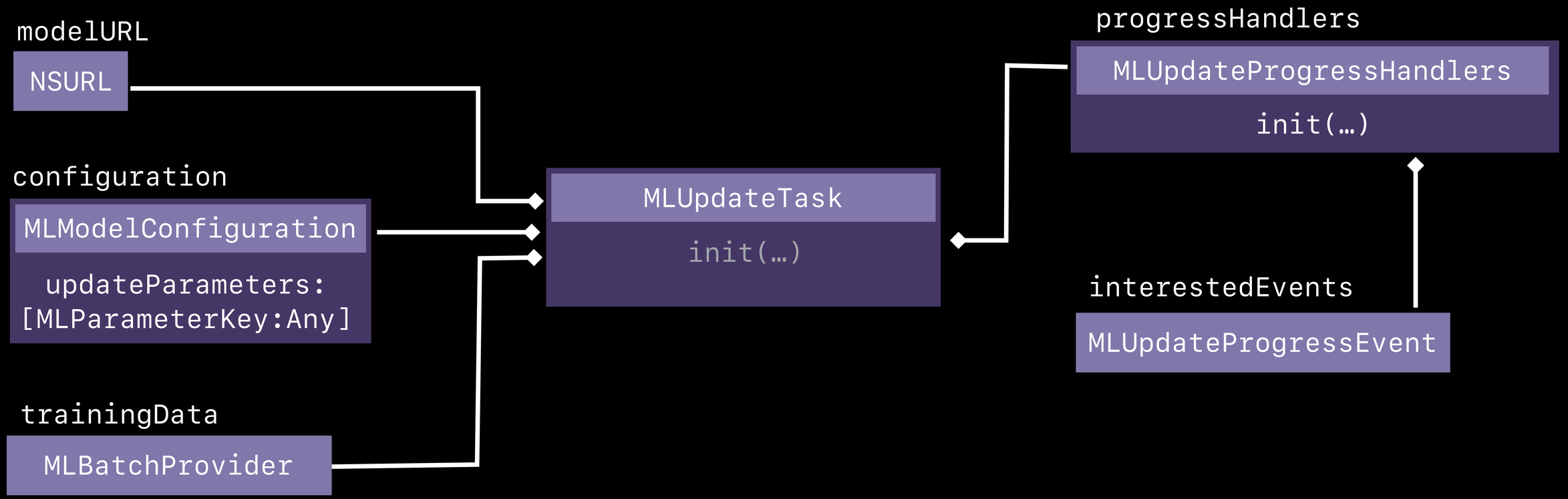
Progress Handlers and Control



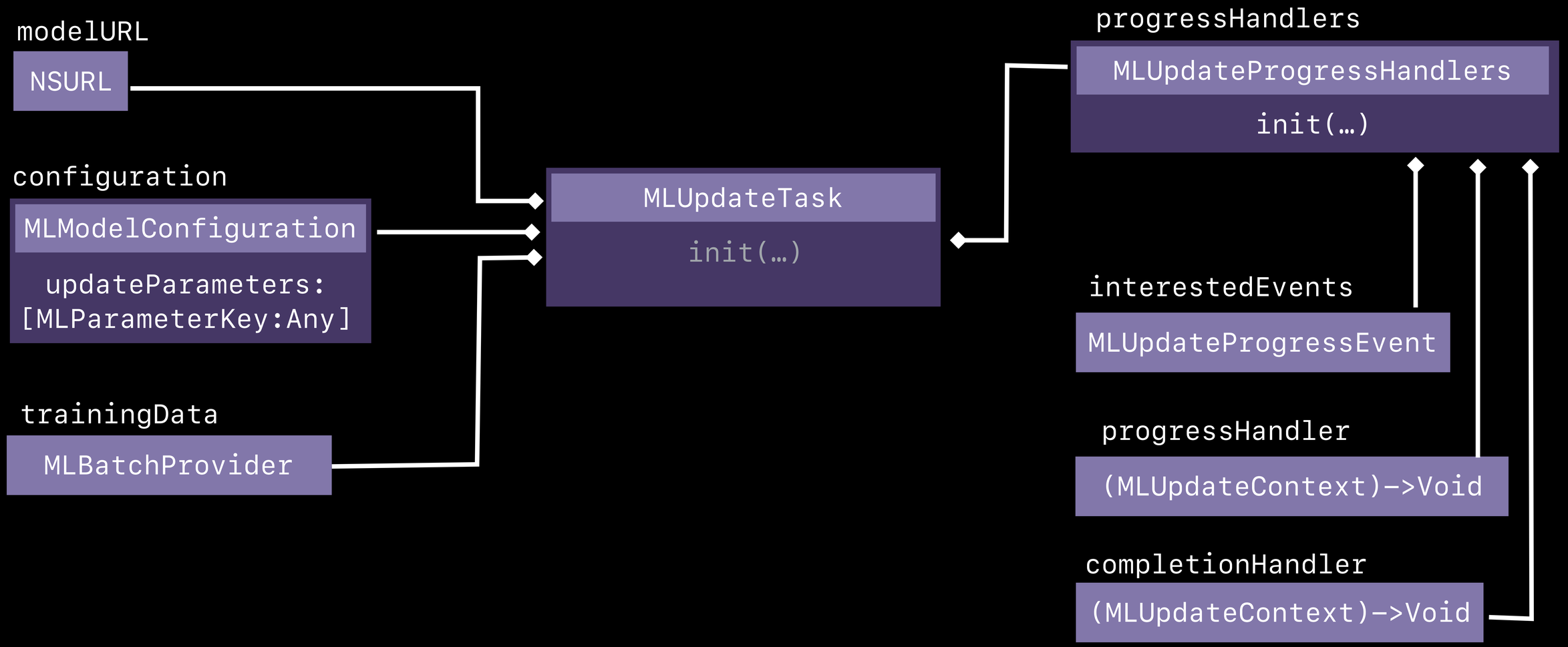
Progress Handlers and Control



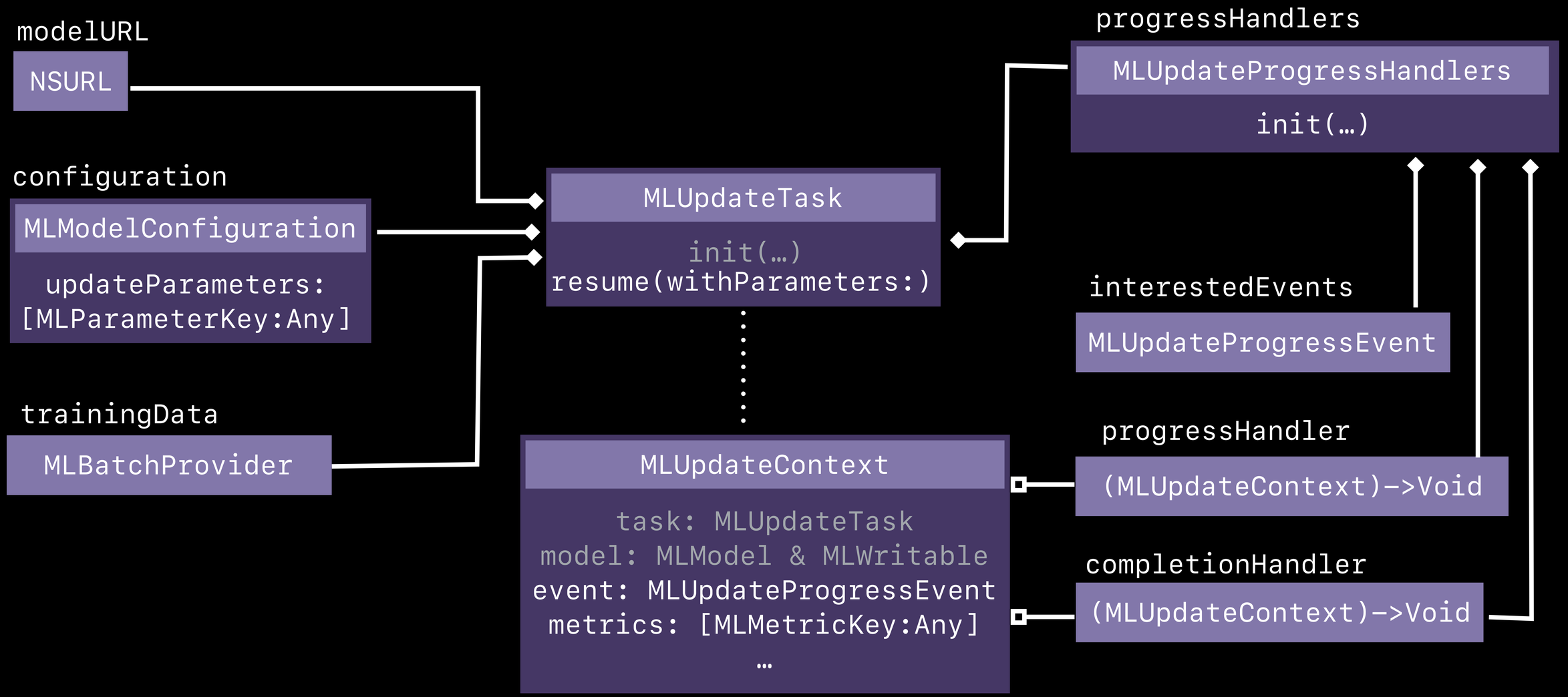
Progress Handlers and Control



Progress Handlers and Control



Progress Handlers and Control



```
// Updating your MLModel



let handlers = try MLUpdateProgressHandlers(forEvents: [.trainingBegan, .epochEnd],
                                            progressHandler: {
                                                context in
                                                computeAccuracy(forModel: context.model)
                                            },
                                            completionHandler: completionHandler)


let updateTask = try MLUpdateTask(forModelAt: updatableModelURL,
                                  trainingData: trainingData,
                                  configuration: configuration,
                                  progressHandlers: handlers)
```

```
// Updating your MLModel
```

```
let handlers = try MLUpdateProgressHandlers(forEvents: [.trainingBegan, .epochEnd],
                                            progressHandler: {
                                                context in
                                                computeAccuracy(forModel: context.model)
                                            },
                                            completionHandler: completionHandler)
```

```
let updateTask = try MLUpdateTask(forModelAt: updatableModelURL,
                                  trainingData: trainingData,
                                  configuration: configuration,
                                  progressHandlers: handlers)
```

```
// Updating your MLModel



let handlers = try MLUpdateProgressHandlers(forEvents: [.trainingBegan, .epochEnd],
                                            progressHandler: {
                                                context in
                                                computeAccuracy(forModel: context.model)
                                            },
                                            completionHandler: completionHandler)

let updateTask = try MLUpdateTask(forModelAt: updatableModelURL,
                                  trainingData: trainingData,
                                  configuration: configuration,
                                  progressHandlers: handlers)
```

```
// Updating your MLModel



let handlers = try MLUpdateProgressHandlers(forEvents: [.trainingBegan, .epochEnd],
                                            progressHandler: {
                                                context in
                                                computeAccuracy(forModel: context.model)
                                            },
                                            completionHandler: completionHandler)


let updateTask = try MLUpdateTask(forModelAt: updatableModelURL,
                                  trainingData: trainingData,
                                  configuration: configuration,
                                  progressHandlers: handlers)
```

Core ML in the Background

NEW

Core ML in the Background

NEW

BackgroundTasks Framework

`BGTaskScheduler`

`BGProcessingTaskRequest`

Core ML in the Background

NEW

BackgroundTasks Framework

`BGTaskScheduler`

`BGProcessingTaskRequest`

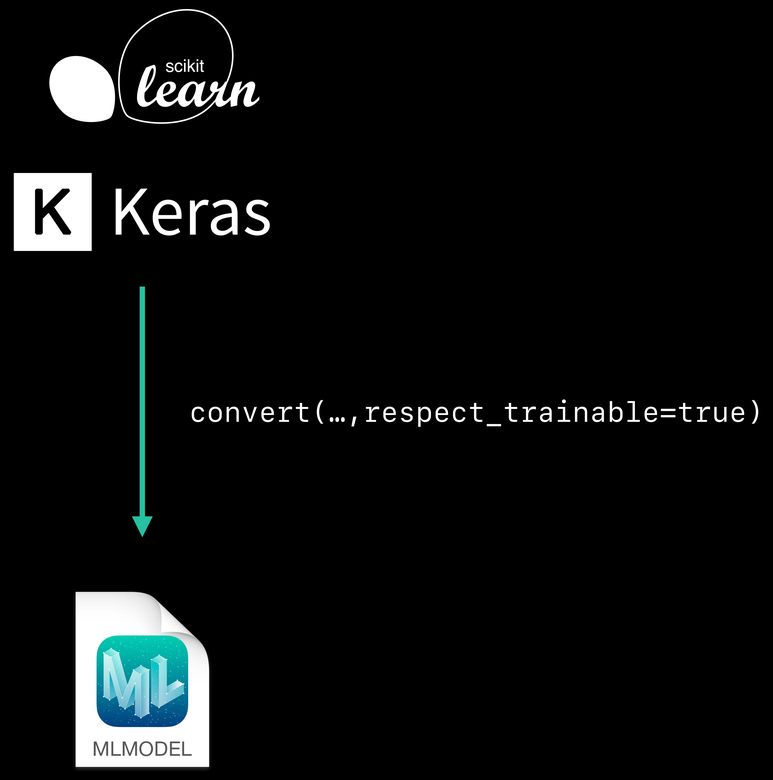
Advances in App Background Execution — Wednesday, 10:00

Getting an Updatable Model

Core ML Tools

Getting an Updatable Model

Core ML Tools



Getting an Updatable Model

Core ML Tools

Model Personalization Summary

Personalize experiences

On-device

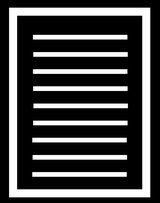
Flexible update task APIs

Neural Networks

Aseem Wadhwa, Core ML

Segmentation

Text Summarization



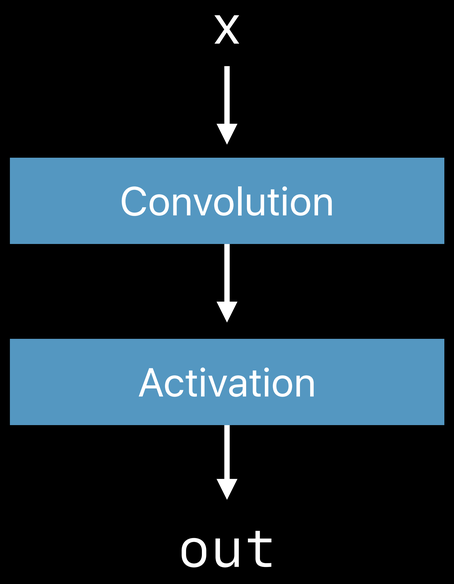
headline

Speech-to-Text

“Neural Networks are awesome!”

Inside a Neural Network Model

Inside a Neural Network Model



Graph

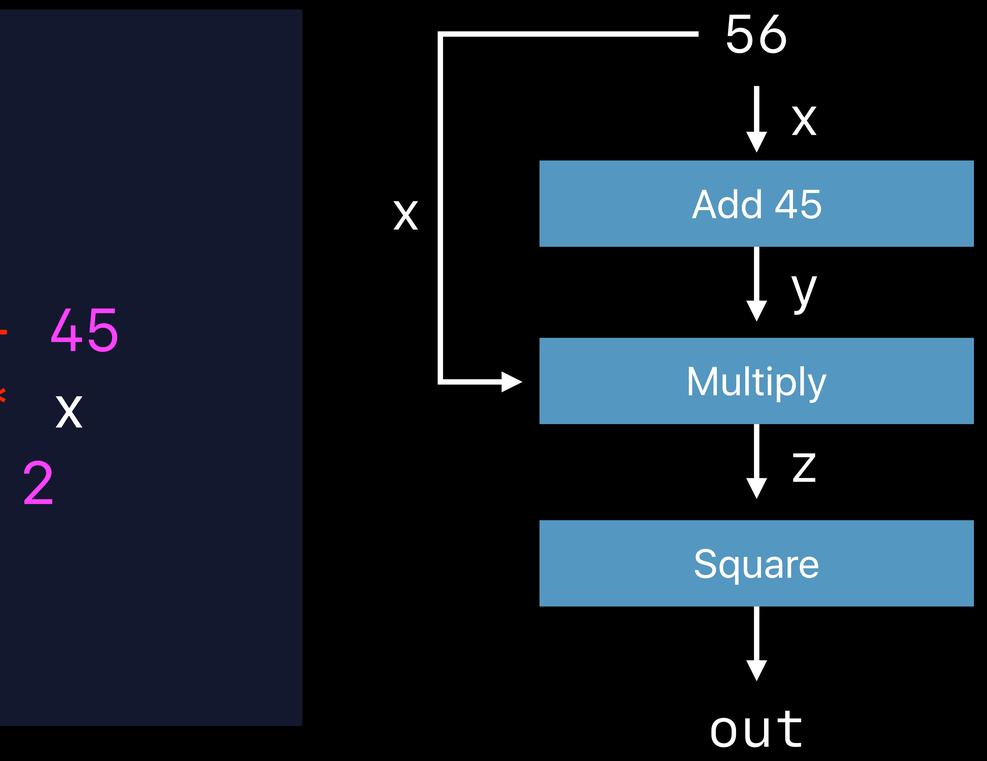
Graph ⟷ Code

Code Snippet

```
x = 56
y = x + 45
z = y * x
out = z ^ 2
```

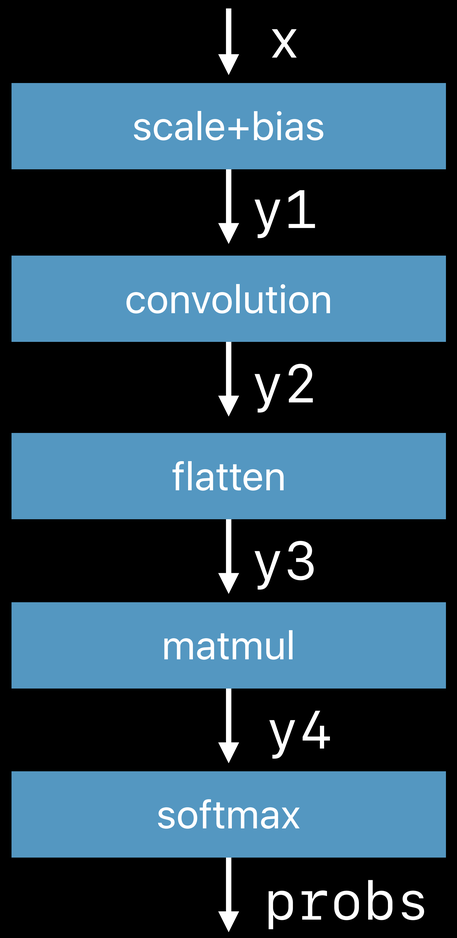
Code as Graph

```
x = 56
y = x + 45
z = y * x
out = z ^ 2
```

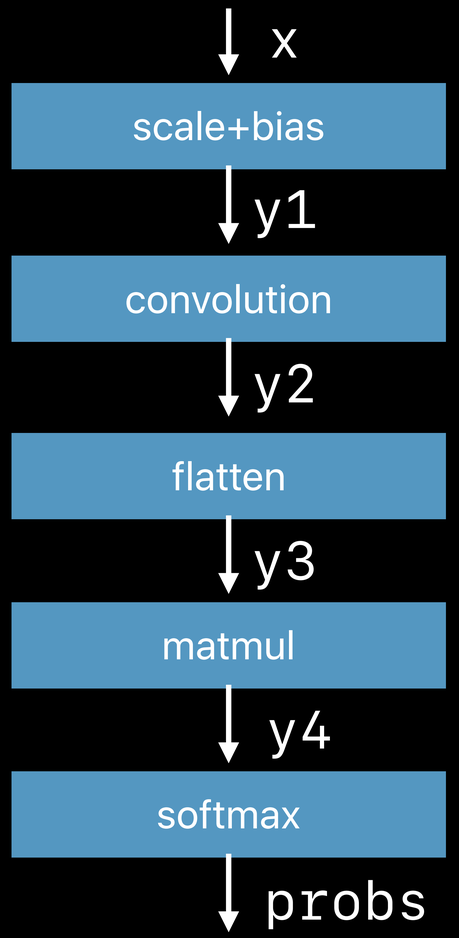


Neural Network as Code or Graph

```
x = MultiArray(3,100,100)
y1 = x/255.0 - 1
y2 = convolution(y1)
y3 = flatten(y2)
y4 = matmul(y3)
probs = softmax(y4)
```

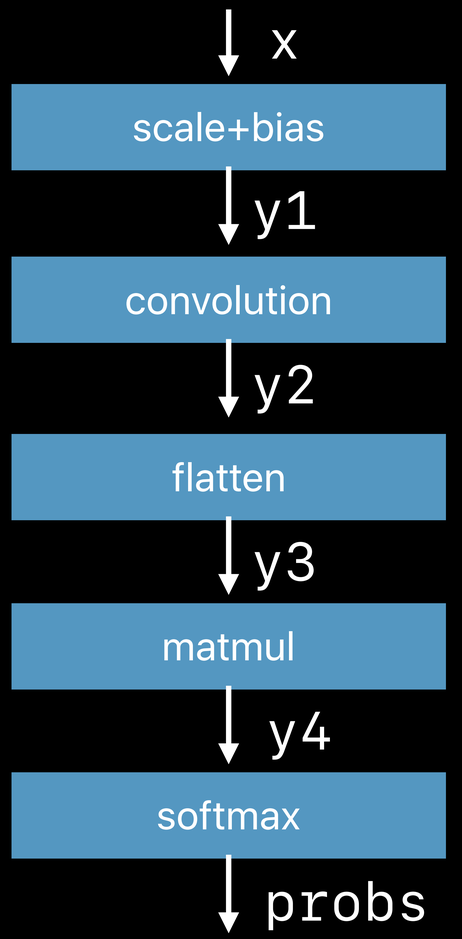


Neural Network Graph



Neural Network Graph

Multi-dimensional variables

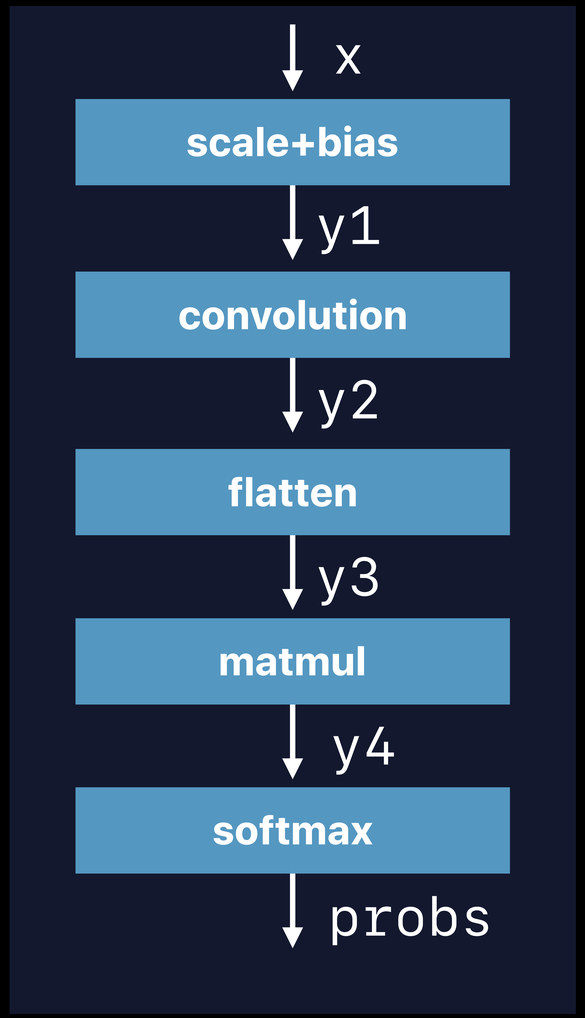


Shape	Rank
(3,100,100)	3
(3,100,100)	3
(32,20,20)	3
(12800)	1
(5)	1
(5)	1

Neural Network Graph

Multi-dimensional variables

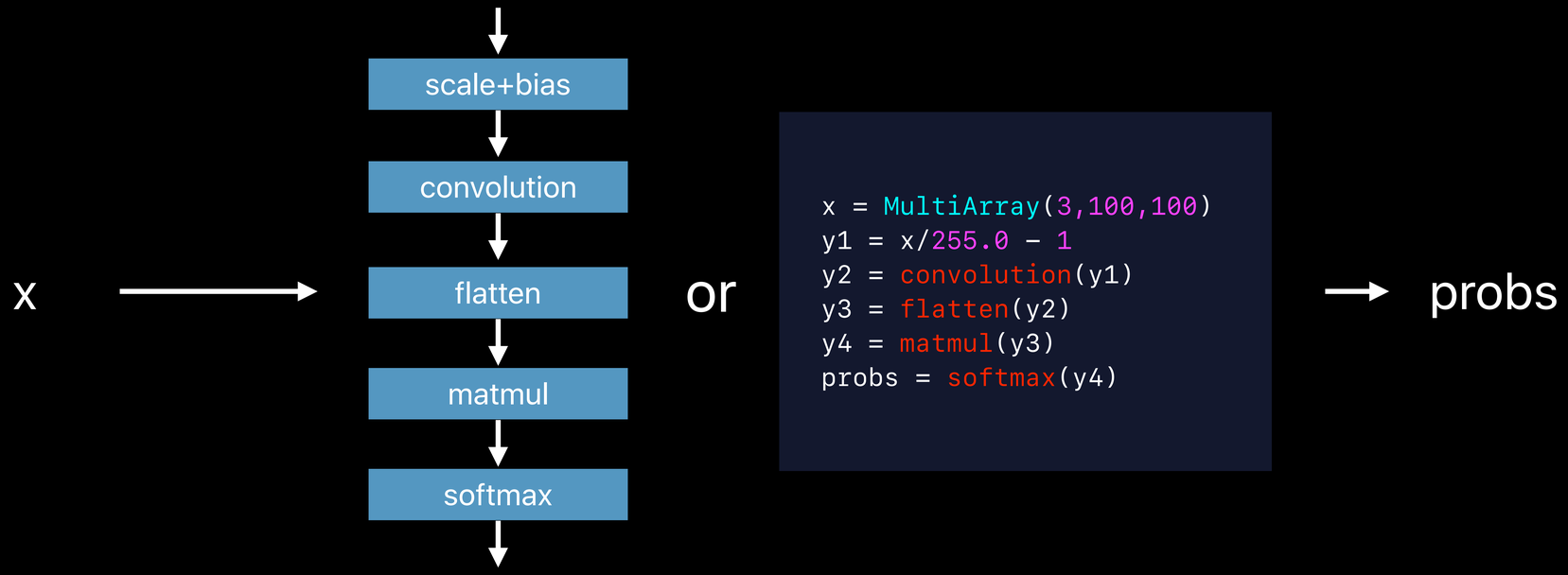
Specialized math functions



Shape	Rank
(3,100,100)	3
(3,100,100)	3
(32,20,20)	3
(12800)	1
(5)	1
(5)	1

Neural Network

Memory and compute intensive



Core ML Model

Efficient, Portable

Core ML 2

Acyclic graphs ⟷ Straight-line code

Core ML 2

Acyclic graphs ⟷ Straight-line code

40 layer types

```
Convolution
Pooling
Fully connected
Reshape
Crop
Pad
Batchnorm
LSTMs
GRU
ResizeBilinear
CropAndResize
UnaryFunctions
Upsample
…
```

What’s new?

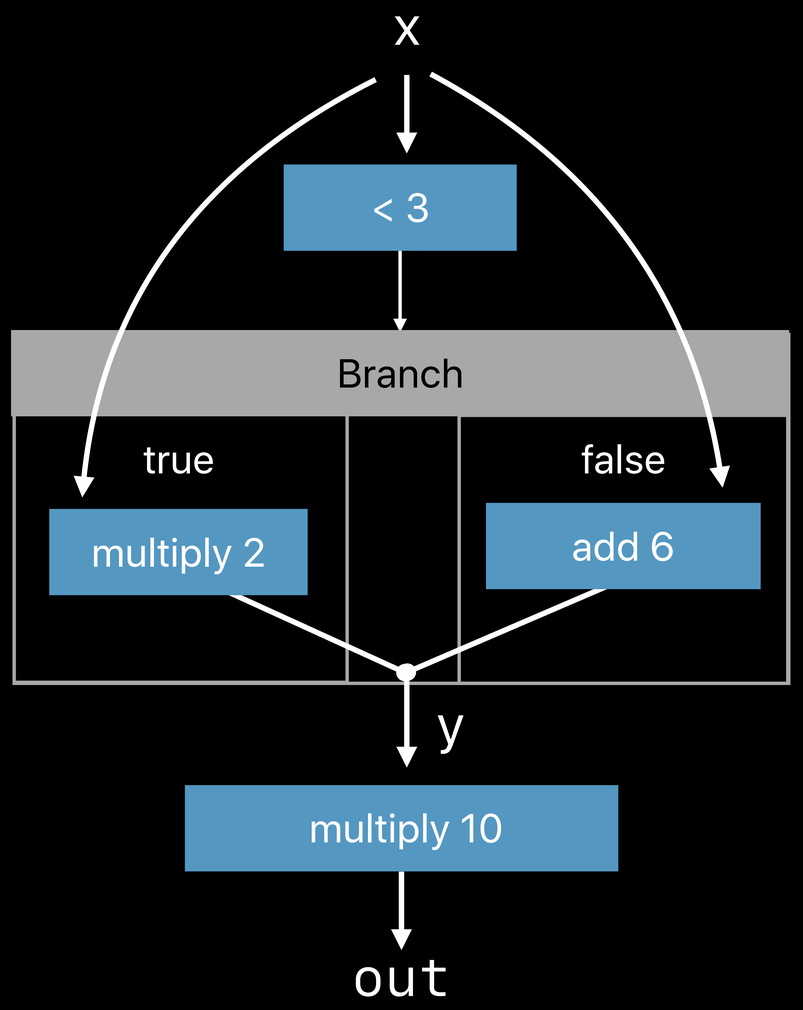
Beyond straight-line code

Control Flow

Branch

NEW

```
if x > 3:
  y = x * 2
else:
  y = x + 6
out = 10 * y
```

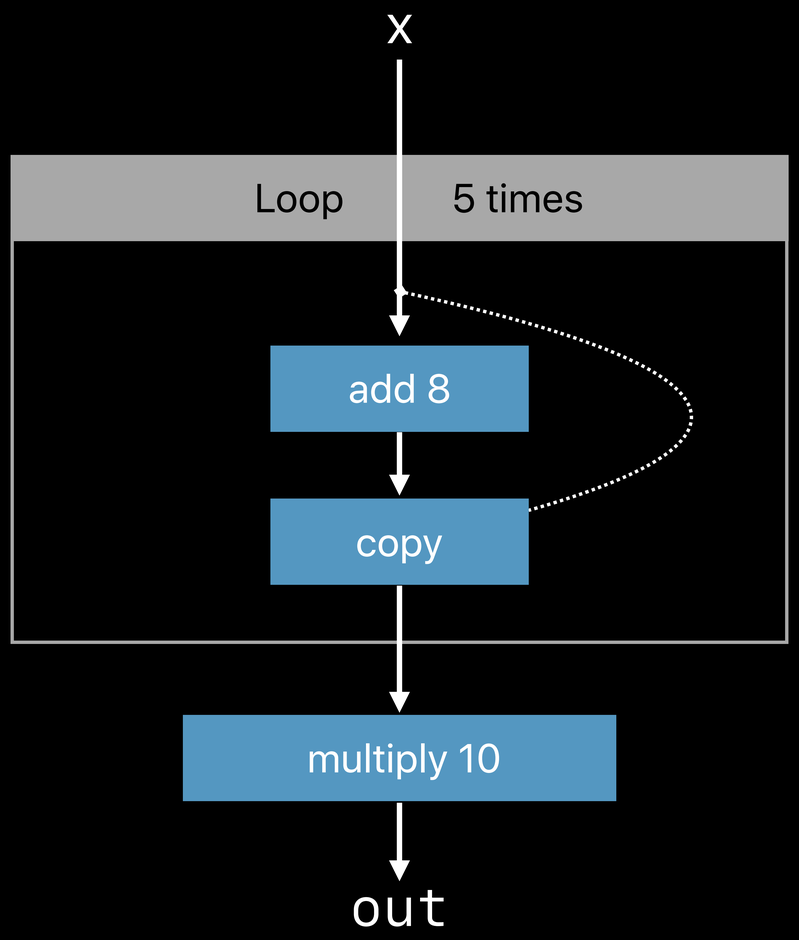


Control Flow

Loop

NEW

```
for i in 1 to 5:
    x = x + 8
out = 10 * x
```



Dynamic Variables

NEW

Dynamic memory allocation

```
y = x * 5
z = malloc(y)
for i in 0..<y:
  z[i] = x+i
out = z
```

“Dynamic“ layers

fillDynamic

rangeDynamic

sliceDynamic

...

Rapidly Evolving Field

Active research

Dynamic Neural Networks

New operators

100+ Layer Additions in Core ML 3

NEW

100+ Layer Additions in Core ML 3

NEW

Control Flow	Elementwise	Random Distributions	Logical
Loop, Continue, Break, Branch ...	Clip, Floor, Round Trignometric, Sign, Broadcastable Add, Multiply ...	Uniform, Normal, Bernoulli ...	AND, OR, XOR, NOT

Boolean	Tensor Manipulation	Masking	Dynamic
Equal, Greater, Less ...	Tile, Stack, Gather, Scatter, Split ...	MatrixBand, LowerTriangle ...	Range, Pad, Fill, Slice ...

Neural Networks in Core ML 3

Control flow

Dynamic shapes

Expanded generalized operations

New layers

Resnet
Inceptionv1
ELMo
Mobilenet
Nasnet
VGG
PSPNet
ULMFit
SSD mobilenet
Squeezenet

Style transfer
YOLO
Densenet
FCN
BERT
MT-DNN
Inceptionv2
SSD Lite
Deeplabv3
WaveGlow

Tacotron
WaveRNN
Open AI GPT
Inceptionv3
VGGish
Xception
Transformer
LAS
Seq2seq
RNN-T

BiDAF
GPT
VoxelNet
Mask R-CNN
DeepConvLSTM
Resnet50
Inceptionv4
Deep Speech
GPT2
Wavenet

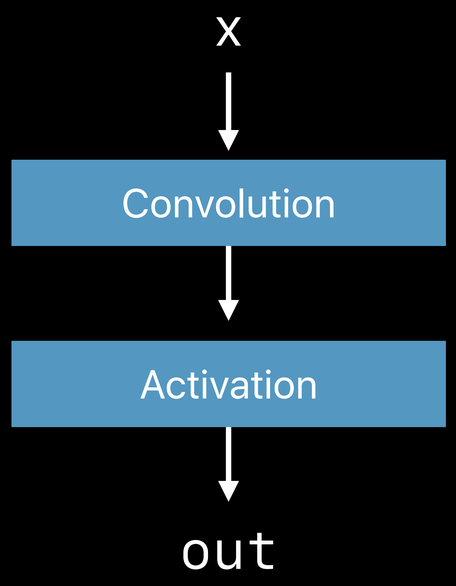
How to Build ML Models

Open Protobuf Specification

Converters

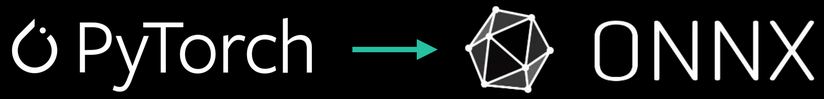
Specifying Model in Code

e.g. Core ML Tools




```
# Build
builder = NeuralNetworkBuilder(input_features, output_features)

builder.add_convolution('conv','data','conv',W=…,params=…)
builder.add_activation('relu','conv', 'output', 'RELU')

# Save
coremltools.utils.save_spec(builder.spec,'conv_relu.mlmodel')
```

Converters



Conversion Bumps

Conversion Bumps

"NotImplementedError: Unsupported Ops of type: Tile"

"Slice doesn't handle begin=[0,0,0,0]"

"Unhandled strided slice case"

"AssertionError: Concat axis case not handled"

"Concat not supported along batch axis"

"NotImplementedError: Unsupported ONNX ops of type: RandomNormal"

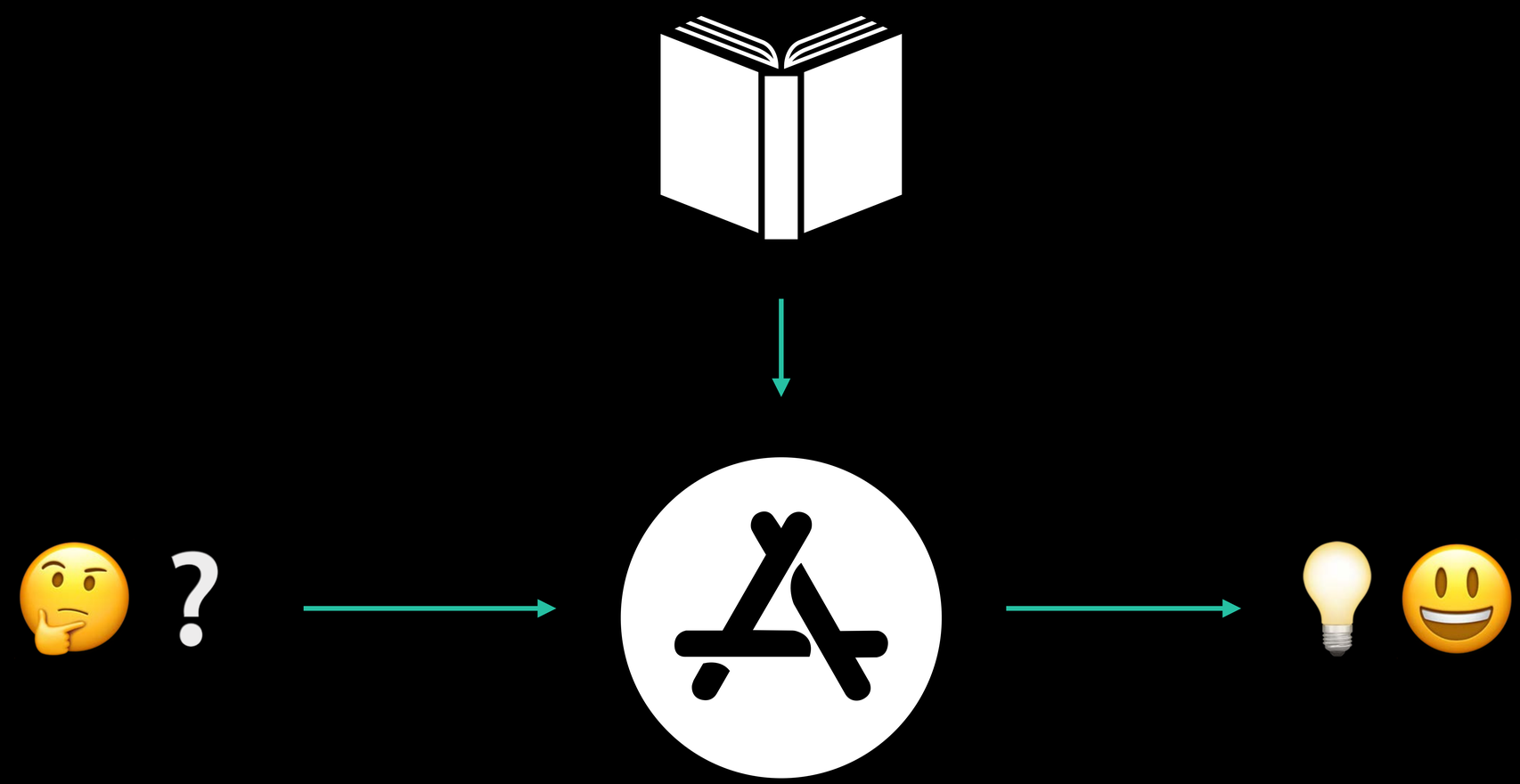
etc

Conversion Bumps

Smooth road ahead

Question and Answer App

Allen Lin, Core ML



Demo

BERT Model

Devlin, Jacob, et al. "Bert: Pre-training of deep bidirectional transformers for language understanding." arXiv preprint arXiv:1810.04805 (2018).

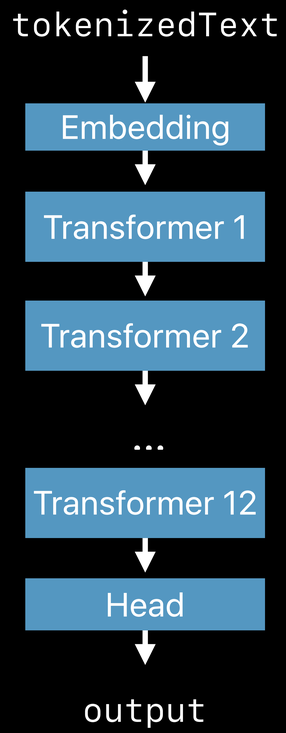
BERT Model

Natural language understanding

Devlin, Jacob, et al. "Bert: Pre-training of deep bidirectional transformers for language understanding." arXiv preprint arXiv:1810.04805 (2018).

BERT Model

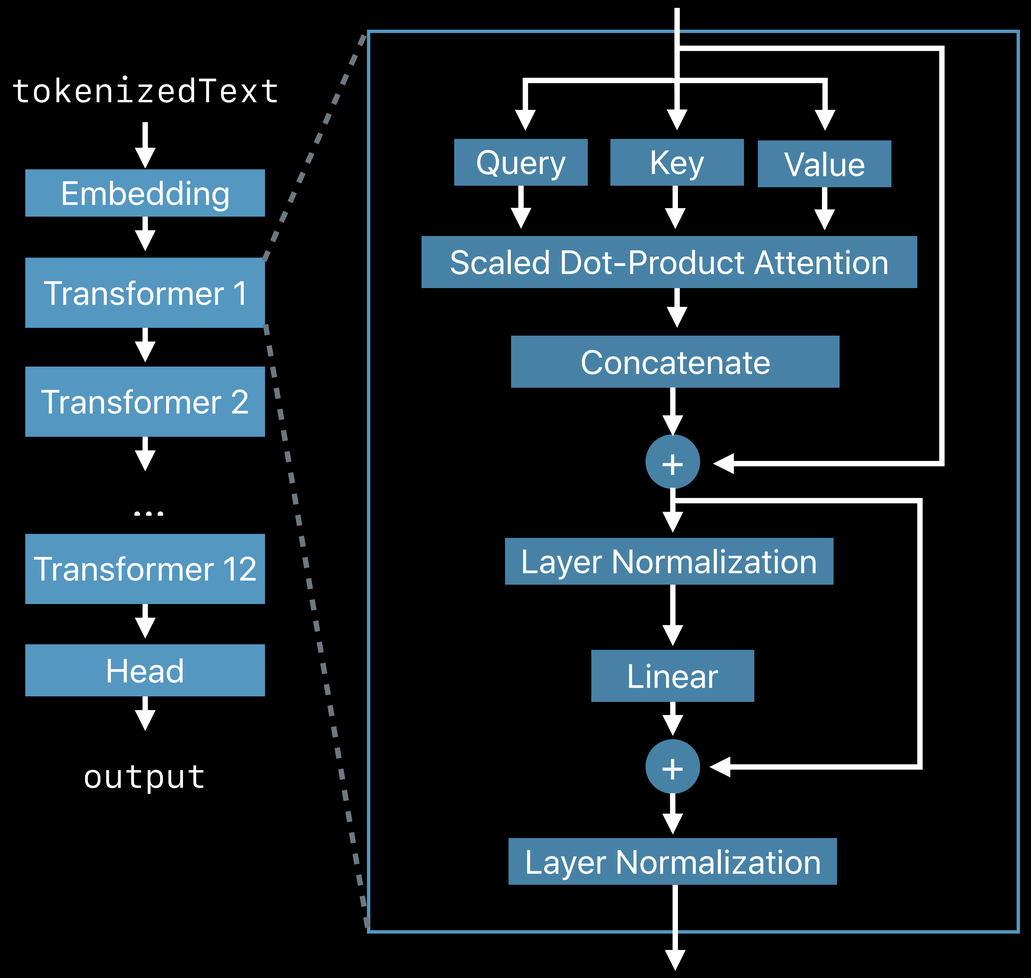
Natural language understanding



Devlin, Jacob, et al. "Bert: Pre-training of deep bidirectional transformers for language understanding." arXiv preprint arXiv:1810.04805 (2018).

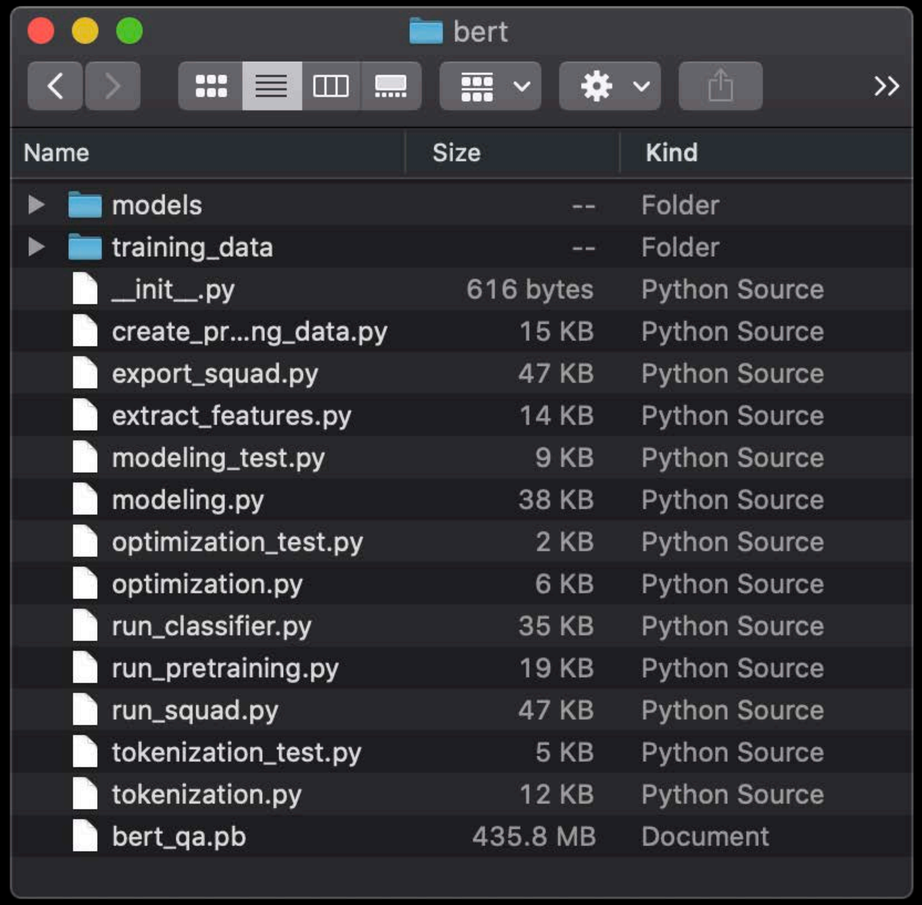
BERT Model

Natural language understanding

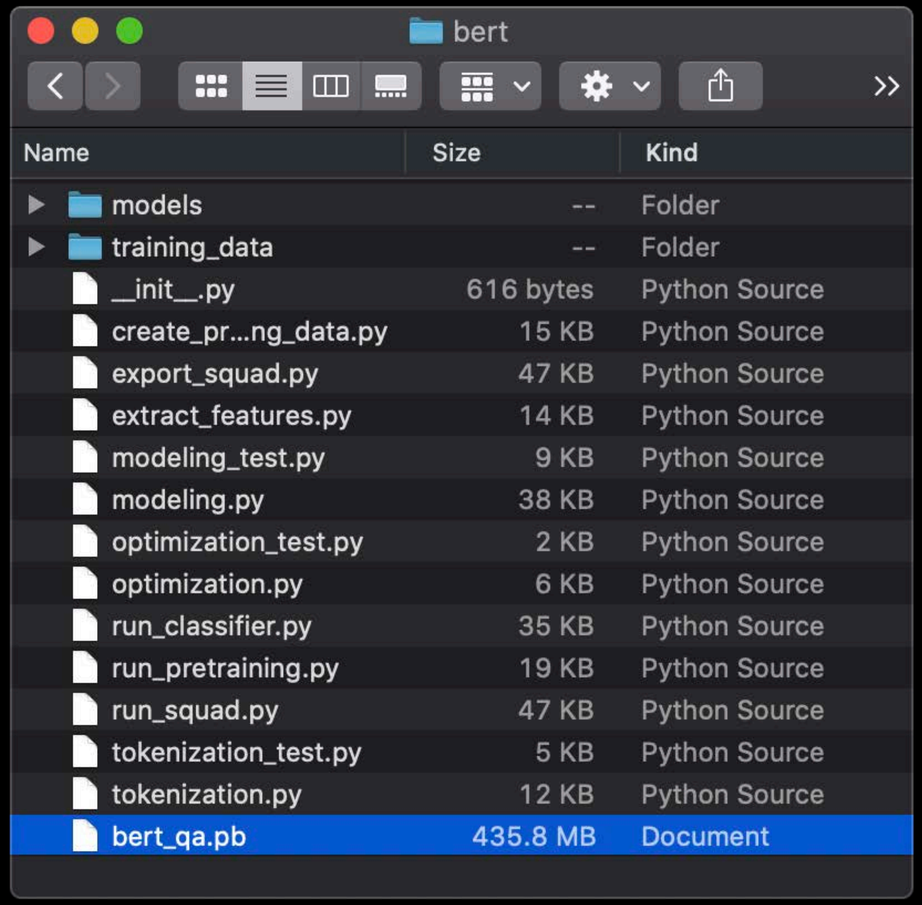


Devlin, Jacob, et al. "Bert: Pre-training of deep bidirectional transformers for language understanding." arXiv preprint arXiv:1810.04805 (2018).

Converting to Core ML

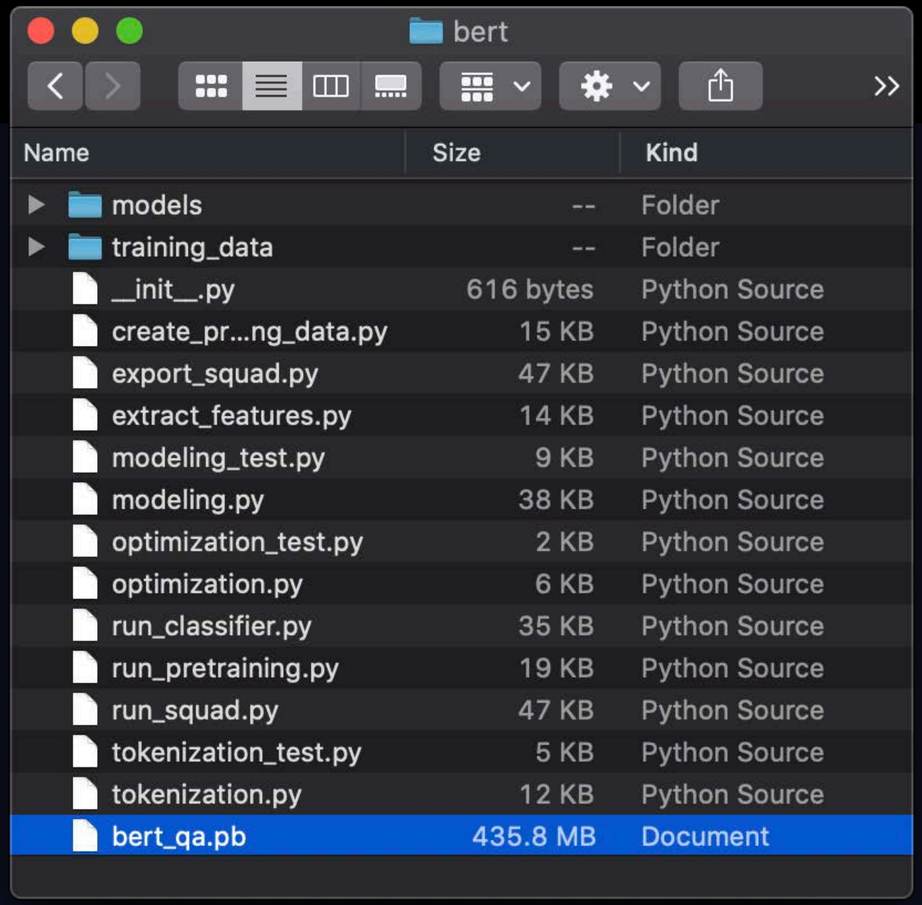


Converting to Core ML

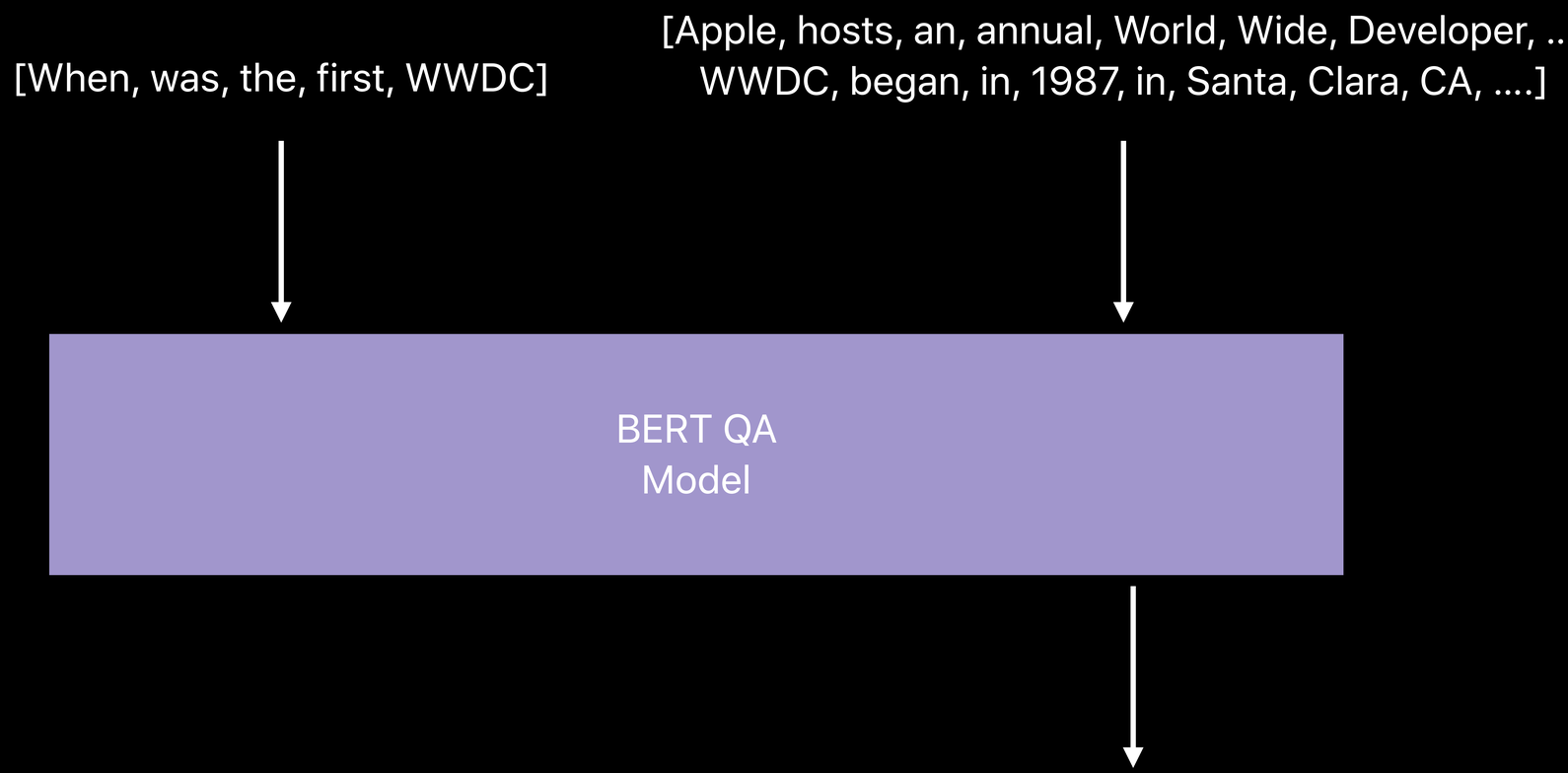


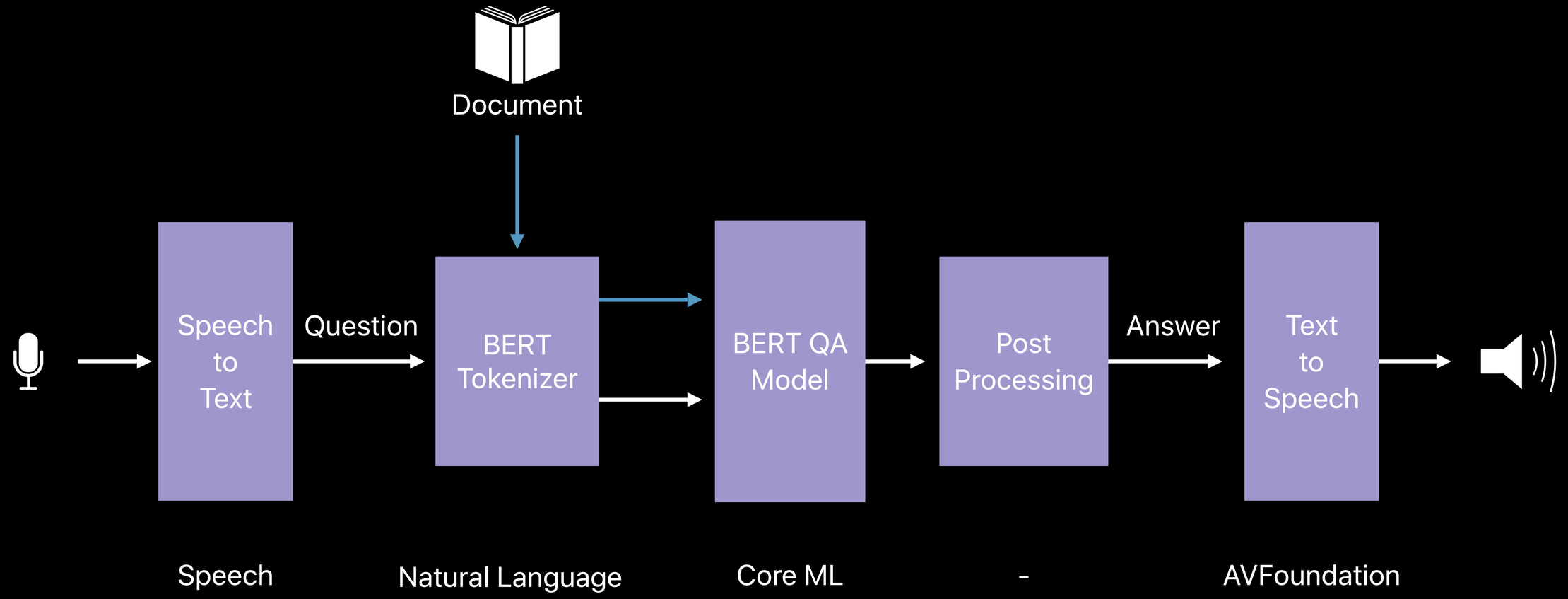
Converting to Core ML

```
import tfcoreml
mlmodel = tfcoreml.convert('./bert_qa.pb')
mlmodel.save('BertQA.mlmodel')
```



Question and Answer Model



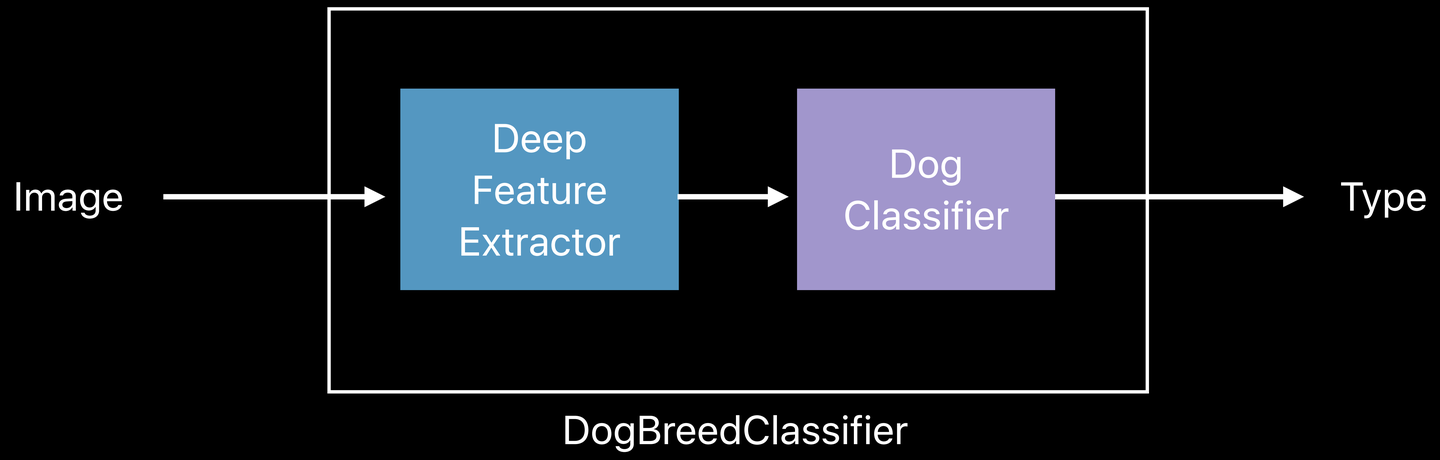

Document
Speech to Text
Question
BERT Tokenizer
BERT QA Model
Post Processing
Answer
Text to Speech
Speech
Natural Language
Core ML
-
AVFoundation

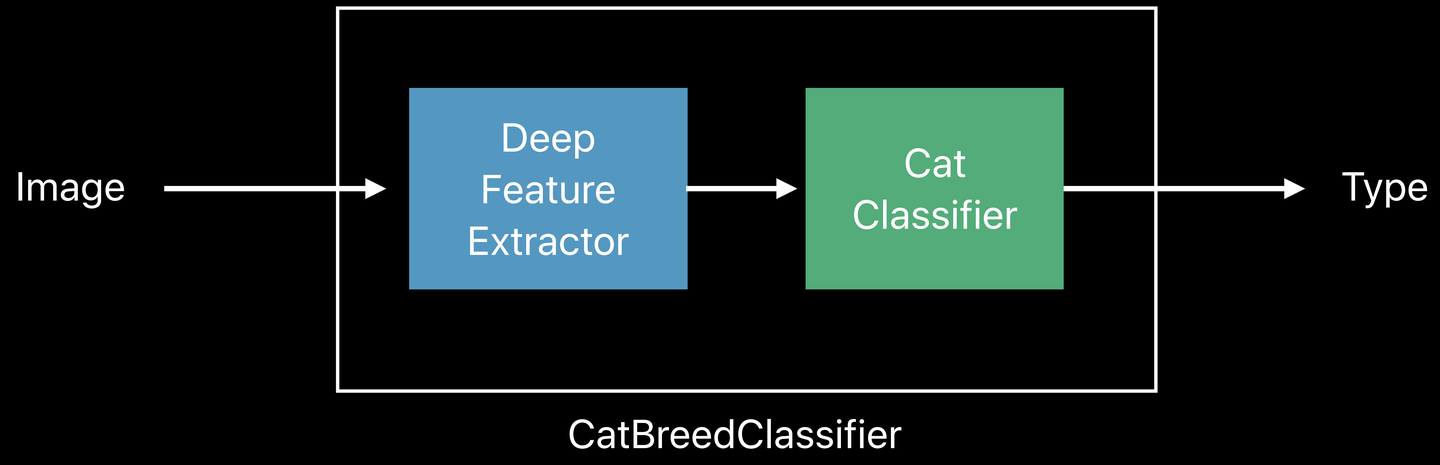
Additional Updates

Aseem Wadhwa, Core ML

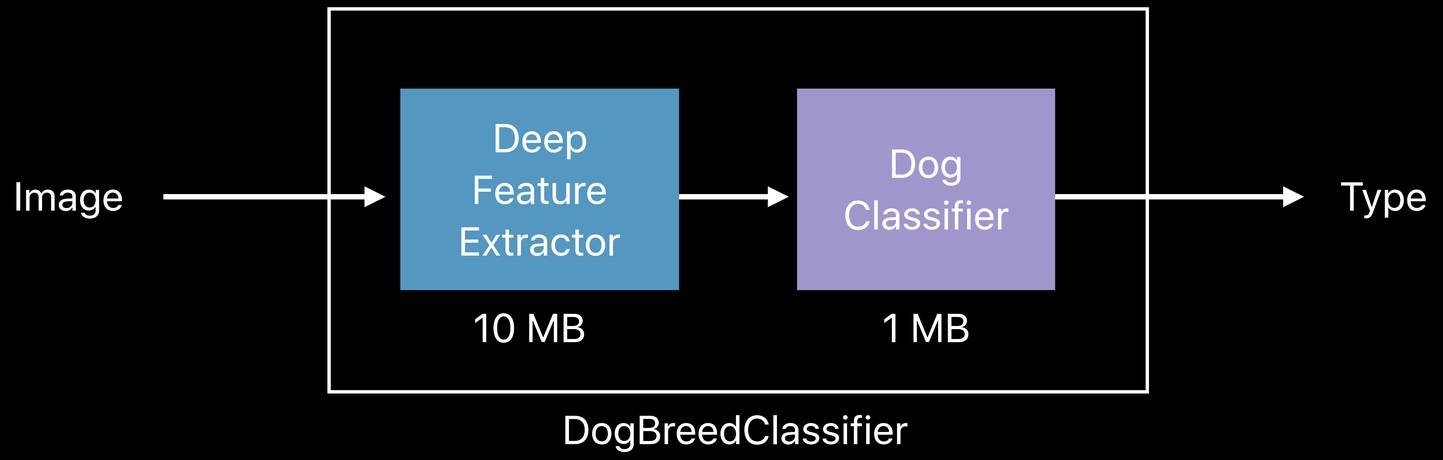
Multiple Models

Duplicated Submodels

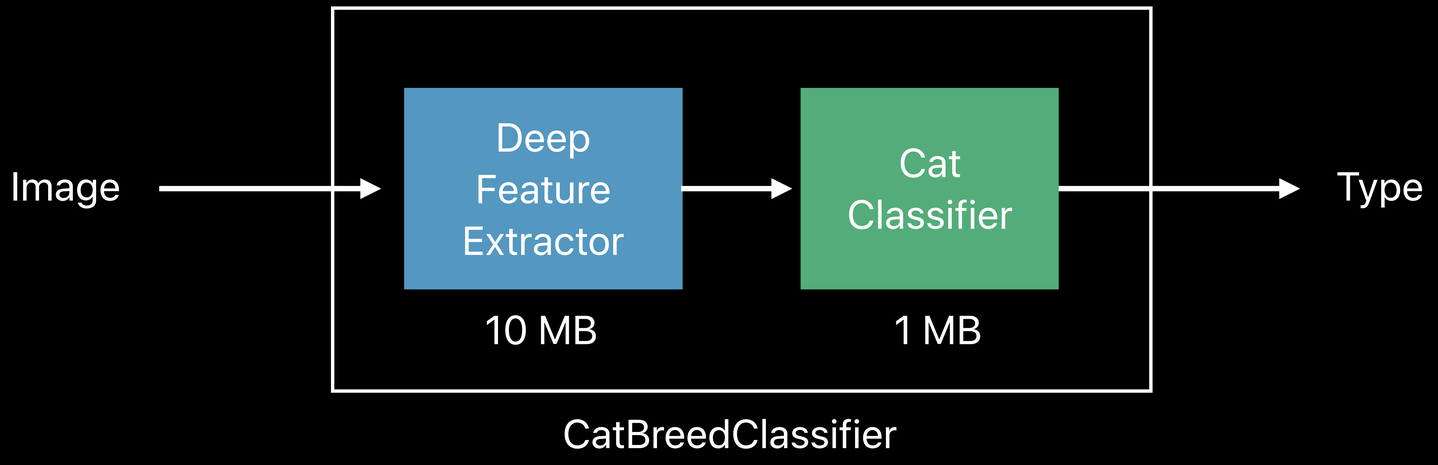




Duplicated Submodels

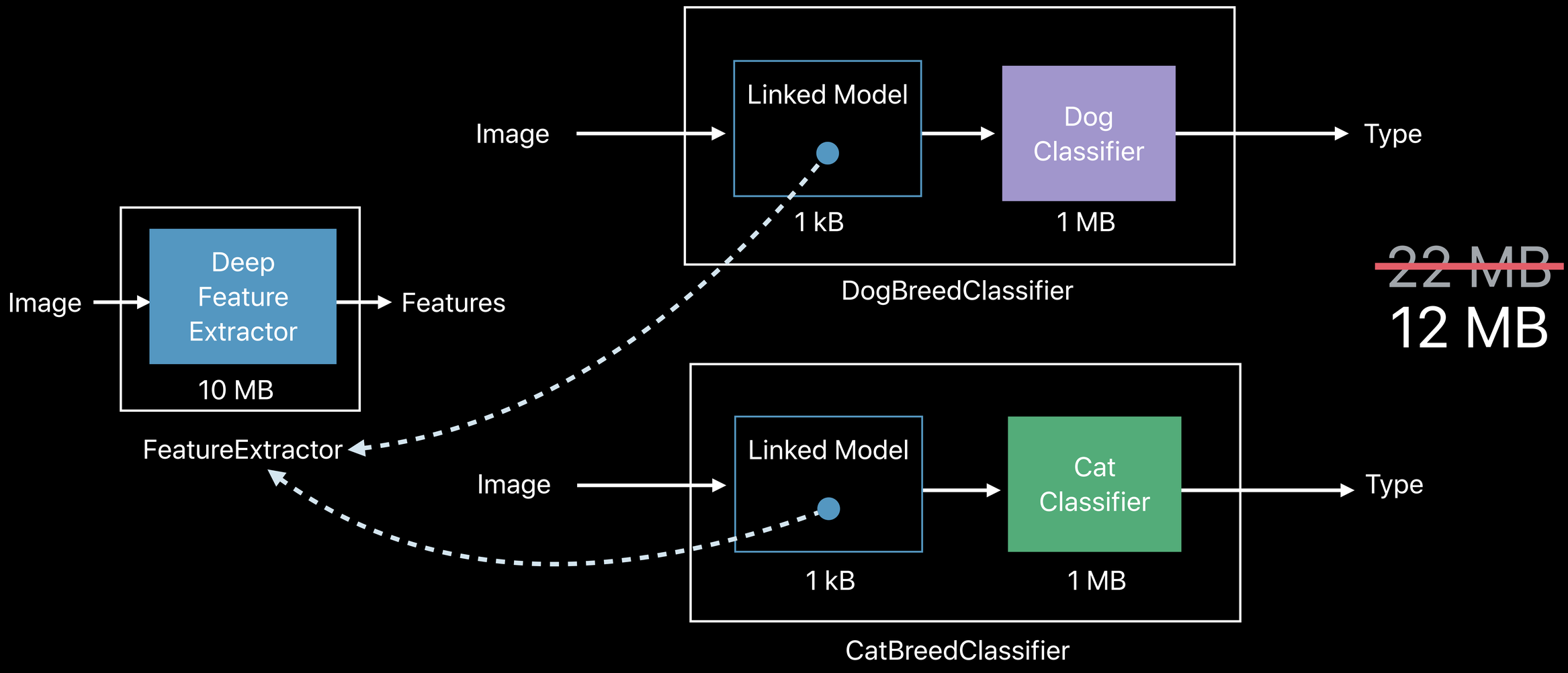


22 MB



Linked Model

NEW



~~22 MB~~
12 MB

Linked Model

NEW

Dynamic linking using

- Compiled filename
- Search path

Useful for pipelines and updatable models

Image Features

Name	Type	Description
▼ Inputs		
image	Image (Color 224 x 224)	Input image to be classified

```
// Input Image to be classified
var image : CVPixelBuffer
```

Image Features

Name	Type	Description
▼ Inputs		
image	Image (Color 224 x 224)	Input image to be classified

```
// Input Image to be classified
var image : CVPixelBuffer
```

What if you have

photo2112.png
1024 x 768

or `CGImage` ?

Image Features

Name	Type	Description
▼ Inputs		
image	Image (Color 224 x 224)	Input image to be classified

```
// Input Image to be classified
var image : CVPixelBuffer
```

What if you have

photo2112.png
1024 x 768

or `CGImage` ?

Vision Framework

`VNCoreMLRequest`

MLFeatureValue

Image Extension

NEW

Automatic scaling and format conversion

```
init(imageAt: URL,
  constraint: MLImageConstraint,
     options: [MLFeatureValue.ImageOption : Any]?) throws
```

```
init(cgImage: CGImage,
  constraint: MLImageConstraint,
     options: [MLFeatureValue.ImageOption : Any]?) throws
```

MLModelConfiguration

Optionally constrain compute units:

```
var computeUnits : MLComputeUnits
```

Case	Compute Unit Set	
`.all`	**Neural Engine**, GPU, CPU	← Default
`.cpuAndGPU`	CPU, GPU	
`.cpuOnly`	CPU	

MLModelConfiguration

GPU specific options

NEW

```
var preferredMetalDevice : MTLDevice
var allowLowPrecisionAccumulationOnGPU : Bool
```

MLModelConfiguration

GPU specific options

NEW

```
var preferredMetalDevice : MTLDevice
var allowLowPrecisionAccumulationOnGPU : Bool
```

Summary

Summary

Personalize experiences

Summary

Personalize experiences

Recent advances in neural networks architectures

Summary

Personalize experiences

Recent advances in neural networks architectures

More options and flexibility

More Information

developer.apple.com/wwdc19/704

Introducing PencilKit	Wednesday, 3:00
Creating Great Apps Using Core ML and ARKit	Thursday, 10:00
Drawing Classification and One-Shot Object Detection in Turi Create	Friday, 10:00


WWDC19