

[Vision](#) / CoreMLRequest

Structure

CoreMLRequest

An image-analysis request that uses a Core ML model to process images.

iOS 18.0+ | iPadOS 18.0+ | macOS 15.0+ | tvOS 18.0+ | visionOS 2.0+

```
struct CoreMLRequest
```

Overview

The results array of a [Core ML](#)-based image-analysis request contain a different observation type, depending on the kind of `MLModel` object you use:

- If the model predicts a single feature and the model's [MLModelDescription](#) object has a non-`nil` value for [predictedFeatureName](#), then Vision treats the model as a classifier. The results are [ClassificationObservation](#) objects.
- If the model's outputs include at least one output with a feature type of [MLFeatureType.image](#), Vision treats that model as an image-to-image model. The results are [PixelBufferObservation](#) objects.
- Otherwise, Vision treats the model as a general predictor model. The results are [CoreMLFeatureValueObservation](#) objects.

Note

Vision forwards all confidence values from Core ML models as-is and doesn't normalize them to `[0, 1]`.

Topics

Creating a request

```
init(model: CoreMLModelContainer, CoreMLRequest.Revision?)
```

Creates a Core ML request.

Getting the revision

```
let revision: CoreMLRequest.Revision
```

The algorithm or implementation the request uses.

```
static let supportedRevisions: [CoreMLRequest.Revision]
```

The collection of revisions the request supports.

```
enum Revision
```

A type that describes the algorithm or implementation that the request performs.

Inspecting a request

```
var supportedIdentifiers: [String]?
```

The classification identifiers supported by the request.

```
let modelContainer: CoreMLModelContainer
```

The model to base the image analysis request on.

```
struct CoreMLModelContainer
```

A model container to use with an image-analysis request.

```
enum ComputeStage
```

Types that represent the compute stage.

```
var cropAndScaleAction: ImageCropAndScaleAction
```

An optional setting that tells the Vision algorithm how to scale an input image.

```
enum ImageCropAndScaleAction
```

A scale to apply to an input image before performing a request.

Performing a request

```
func perform(on: URL, orientation: CGImagePropertyOrientation?) async  
throws -> Self.Result
```

Performs the request on an image URL and produces observations.

Required Default implementations provided.

```
func perform(on: Data, orientation: CGImagePropertyOrientation?) async  
throws -> Self.Result
```

Performs the request on image data and produces observations.

Required Default implementations provided.

```
func perform(on: CGImage, orientation: CGImagePropertyOrientation?)  
async throws -> Self.Result
```

Performs the request on a Core Graphics image and produces observations.

Required Default implementations provided.

```
func perform(on: CVPixelBuffer, orientation: CGImagePropertyOrientation  
?) async throws -> Self.Result
```

Performs the request on a pixel buffer and produces observations.

Required Default implementations provided.

```
func perform(on: CMSampleBuffer, orientation: CGImageProperty  
Orientation?) async throws -> Self.Result
```

Performs the request on a Core Media buffer and produces observations.

Required Default implementations provided.

```
func perform(on: CIImage, orientation: CGImagePropertyOrientation?)  
async throws -> Self.Result
```

Performs the request on a Core Image image and produces observations.

Required Default implementations provided.

Relationships

Conforms To

CustomStringConvertible

Equatable

Hashable
ImageProcessingRequest
Sendable
SendableMetatype
VisionRequest

See Also

Machine learning image analysis

`struct CoreMLFeatureValueObservation`

An object that represents a collection of key-value information that a Core ML image-analysis request produces.

`struct ClassificationObservation`

An object that represents classification information that an image-analysis request produces.

`struct PixelBufferObservation`

An object that represents an image that an image-analysis request produces.