Training Sound Classification Models in Create ML

Dan Klingler, Audio
Sound Classification
Sound Classification
Sound Classification

Guitar

Drums
Sound Classification

Nature

City
Create ML
Training Data
Training Data

Guitar
- guitar0.caf
- guitar1.wav
- guitar2.aiff

Drums
- drums0.aiff
- drums1.caf
- drums2.m4a

Background
- noise.mp3
- dog.wav
- car.caf
Considerations

Audio files
Considerations

Audio files

sounds.caf
Considerations

Audio files

sounds.caf
Considerations

Audio files

sounds.caf
Considerations

Audio files
Considerations

Audio files

sounds.caf
Considerations

Audio files

- drums.caf
- guitar.caf
- background.caf
Considerations

Audio files

- drums.caf
- guitar.caf
- background.caf
Considerations

Audio files
Considerations

Audio files

Ensure data matches real-world audio environments
Considerations

Audio files

Ensure data matches real-world audio environments

Consider device microphone processing
Considerations

Audio files

Ensure data matches real-world audio environments

Consider device microphone processing

Be aware of model architecture
Sound Analysis Framework
Sound Analysis Framework

New high-level framework for analyzing sound
Sound Analysis Framework

New high-level framework for analyzing sound
Uses Core ML sound classifier models
New high-level framework for analyzing sound

Uses Core ML sound classifier models

Handles common audio operations internally
Sound Analysis Framework

Label | Confidence
--- | ---
Guitar | 0.93
Background | 0.05
Drums | 0.02
Sound Analysis Framework

- Application
- Sound Analysis

Channel Mapping → Sample Rate Conversion → Audio Buffering → ML Model

<table>
<thead>
<tr>
<th>Label</th>
<th>Confidence</th>
</tr>
</thead>
<tbody>
<tr>
<td>Guitar</td>
<td>0.93</td>
</tr>
<tr>
<td>Background</td>
<td>0.05</td>
</tr>
<tr>
<td>Drums</td>
<td>0.02</td>
</tr>
</tbody>
</table>
Sound Analysis Framework

Application

Sound Analysis

Channel Mapping → Sample Rate Conversion → Audio Buffering → MLMODEL

Label | Confidence
---|---
Guitar | 0.93
Background | 0.05
Drums | 0.02
Sound Analysis Framework

Channel Mapping ➞ Sample Rate Conversion ➞ Audio Buffering ➞ MLMODEL

Label          Confidence
Guitar  ➞  0.93
Background ➞  0.05
Drums   ➞  0.02
Sound Analysis Framework

Application

Sound Analysis

Channel Mapping  Sample Rate Conversion  Audio Buffering

Label  Confidence
Guitar  ——>  0.93
Background ——>  0.05
Drums  ——>  0.02
<table>
<thead>
<tr>
<th>Label</th>
<th>Confidence</th>
</tr>
</thead>
<tbody>
<tr>
<td>Guitar</td>
<td>0.93</td>
</tr>
<tr>
<td>Background</td>
<td>0.05</td>
</tr>
<tr>
<td>Drums</td>
<td>0.02</td>
</tr>
</tbody>
</table>
Sound Analysis Framework

Application

Sound Analysis

Channel Mapping → Sample Rate Conversion → Audio Buffering → MLMODEL

<table>
<thead>
<tr>
<th>Label</th>
<th>Confidence</th>
</tr>
</thead>
<tbody>
<tr>
<td>Guitar</td>
<td>0.93</td>
</tr>
<tr>
<td>Background</td>
<td>0.05</td>
</tr>
<tr>
<td>Drums</td>
<td>0.02</td>
</tr>
</tbody>
</table>
Results

<table>
<thead>
<tr>
<th>Label</th>
<th>Confidence</th>
</tr>
</thead>
<tbody>
<tr>
<td>Guitar</td>
<td>0.93</td>
</tr>
<tr>
<td>Background</td>
<td>0.05</td>
</tr>
<tr>
<td>Drums</td>
<td>0.02</td>
</tr>
</tbody>
</table>
## Results

<table>
<thead>
<tr>
<th>Label</th>
<th>Confidence</th>
</tr>
</thead>
<tbody>
<tr>
<td>Guitar</td>
<td>0.95</td>
</tr>
<tr>
<td>Background</td>
<td>0.03</td>
</tr>
<tr>
<td>Drums</td>
<td>0.02</td>
</tr>
</tbody>
</table>

![Waveform](waveform.png)

Time (seconds)
## Results

<table>
<thead>
<tr>
<th>Label</th>
<th>Confidence</th>
</tr>
</thead>
<tbody>
<tr>
<td>Guitar</td>
<td>0.91</td>
</tr>
<tr>
<td>Background</td>
<td>0.05</td>
</tr>
<tr>
<td>Drums</td>
<td>0.04</td>
</tr>
</tbody>
</table>

![Waveform with labeled sections](image)
### Results

<table>
<thead>
<tr>
<th>Time (seconds)</th>
<th>Label</th>
<th>Confidence</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Drums</td>
<td>0.60</td>
</tr>
<tr>
<td></td>
<td>Guitar</td>
<td>0.22</td>
</tr>
<tr>
<td></td>
<td>Background</td>
<td>0.18</td>
</tr>
</tbody>
</table>
Results

<table>
<thead>
<tr>
<th>Label</th>
<th>Confidence</th>
</tr>
</thead>
<tbody>
<tr>
<td>Drums</td>
<td>0.98</td>
</tr>
<tr>
<td>Background</td>
<td>0.01</td>
</tr>
<tr>
<td>Guitar</td>
<td>0.01</td>
</tr>
<tr>
<td>Label</td>
<td>Confidence</td>
</tr>
<tr>
<td>----------------</td>
<td>------------</td>
</tr>
<tr>
<td>Drums</td>
<td>0.96</td>
</tr>
<tr>
<td>Guitar</td>
<td>0.03</td>
</tr>
<tr>
<td>Background</td>
<td>0.01</td>
</tr>
</tbody>
</table>
Sound Analysis Framework
// Create a file analyzer
let fileAnalyzer = try SNAudioFileAnalyzer(url: URL(fileURLWithPath: "/Users/demo/sound.caf"))

// Create the request with your MLModel
let request = try SNClassifySoundRequest(mlModel: MySoundClassifier().model)

// Add the request to the analyzer
try fileAnalyzer.add(request, withObserver: self)

// Analyze the file
fileAnalyzer.analyze()
// Create a file analyzer
let fileAnalyzer = try SNAudioFileAnalyzer(url: URL(fileURLWithPath: "~/Users/demo/sound.caf"))

// Create the request with your MLModel
let request = try SNClassifySoundRequest(mlModel: MySoundClassifier().model)

// Add the request to the analyzer
try fileAnalyzer.add(request, withObserver: self)

// Analyze the file
fileAnalyzer.analyze()
// Create a file analyzer
let fileAnalyzer = try SNAudioFileAnalyzer(url: URL(fileURLWithPath: "/Users/demo/sound.caf"))

// Create the request with your MLModel
let request = try SNClassifySoundRequest(mlModel: MySoundClassifier().model)

// Add the request to the analyzer
try fileAnalyzer.add(request, withObserver: self)

// Analyze the file
fileAnalyzer.analyze()
// Create a file analyzer
let fileAnalyzer = try SNAudioFileAnalyzer(url: URL(fileURLWithPath: "~/Users/demo/sound.caf"))

// Create the request with your MLModel
let request = try SNClassifySoundRequest(mlModel: MySoundClassifier().model)

// Add the request to the analyzer
try fileAnalyzer.add(request, withObserver: self)

// Analyze the file
fileAnalyzer.analyze()
// Create a file analyzer
let fileAnalyzer = try SNAudioFileAnalyzer(url: URL(fileURLWithPath: "/Users/demo/sound.caf"))

// Create the request with your MLModel
let request = try SNClassifySoundRequest(mlModel: MySoundClassifier().model)

// Add the request to the analyzer
try fileAnalyzer.add(request, withObserver: self)

// Analyze the file
fileAnalyzer.analyze()
extension MyResultsObserver : SNResultsObserving {

    func request(_ request: SNRequest, didProduce result: SNResult) {
        guard let classificationResult = result as? SNClassificationResult else { return }
        let topClassification = classificationResult.classifications.first
        let timeRange = classificationResult.timeRange
        // Handle result
    }

    func request(_ request: SNRequest, didFailWithError error: Error) {
        // Handle error
    }

    func requestDidComplete(_ request: SNRequest) {
        // Handle successful end of analysis
    }
}
extension MyResultsObserver : SNResultsObserving {

    func request(_ request: SNRequest, didProduce result: SNResult) {
        guard let classificationResult = result as? SNClassificationResult else { return }
        let topClassification = classificationResult.classifications.first
        let timeRange = classificationResult.timeRange
        // Handle result
    }

    func request(_ request: SNRequest, didFailWithError error: Error) {
        // Handle error
    }

    func requestDidComplete(_ request: SNRequest) {
        // Handle successful end of analysis
    }
}

extension MyResultsObserver: SNResultsObserving {

    func request(_ request: SNRequest, didProduce result: SNResult) {
        guard let classificationResult = result as? SNClassificationResult else { return }
        let topClassification = classificationResult.classifications.first
        let timeRange = classificationResult.timeRange
        // Handle result
    }

    func request(_ request: SNRequest, didFailWithError error: Error) {
        // Handle error
    }

    func requestDidComplete(_ request: SNRequest) {
        // Handle successful end of analysis
    }
}
extension MyResultsObserver : SNResultsObserving {

    func request(_ request: SNRequest, didProduce result: SNResult) {
        guard let classificationResult = result as? SNClassificationResult else { return }
        let topClassification = classificationResult.classifications.first
        let timeRange = classificationResult.timeRange
        // Handle result
    }

    func request(_ request: SNRequest, didFailWithError error: Error) {
        // Handle error
    }

    func requestDidComplete(_ request: SNRequest) {
        // Handle successful end of analysis
    }
}
extension MyResultsObserver : SNResultsObserving {

    func request(_ request: SNRequest, didProduce result: SNResult) {
        guard let classificationResult = result as SNClassificationResult else { return }
        let topClassification = classificationResult.classifications.first
        let timeRange = classificationResult.timeRange
        // Handle result
    }

    func request(_ request: SNRequest, didFailWithError error: Error) {
        // Handle error
    }

    func requestDidComplete(_ request: SNRequest) {
        // Handle successful end of analysis
    }
}
Recap
Recap

Train a sound classifier in Create ML using your own audio data
Recap

Train a sound classifier in Create ML using your own audio data
Run your model on-device using Sound Analysis

<table>
<thead>
<tr>
<th>Label</th>
<th>Confidence</th>
</tr>
</thead>
<tbody>
<tr>
<td>Guitar</td>
<td>0.93</td>
</tr>
<tr>
<td>Background</td>
<td>0.05</td>
</tr>
<tr>
<td>Drums</td>
<td>0.02</td>
</tr>
</tbody>
</table>
## Create ML for Activity, Text, and Recommendations

- **Thursday, 2:00**

## Machine Learning Labs

- **Daily**

More Information

developer.apple.com/wwdc19/425