Advances in Speech Recognition

Neha Agrawal, Software Engineering
What is Speech Recognition?

Live or Pre-Recorded Audio → SFSpeech Recognizer → Transcribed Text

Speech Recognition API

WWDC 2016
What’s New?
Support for macOS
Support for macOS

Supports both AppKit and iPad apps
Support for macOS

Supports both AppKit and iPad apps

50+ languages supported
Support for macOS

Supports both AppKit and iPad apps

50+ languages supported

Requires privacy approval
Support for macOS

Supports both AppKit and iPad apps

50+ languages supported

Requires privacy approval

Must have Siri enabled
On-Device Speech Recognition
On-Device Speech Recognition

Speech is private, stays on-device
On-Device Speech Recognition

Speech is private, stays on-device

Network connection not required
On-Device Speech Recognition

Speech is private, stays on-device
Network connection not required
No cellular data consumption
<table>
<thead>
<tr>
<th>On-Device Speech Recognition</th>
<th>Server</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Accuracy</strong></td>
<td></td>
</tr>
<tr>
<td>Best</td>
<td>Good</td>
</tr>
<tr>
<td>1 minute max audio duration</td>
<td>None</td>
</tr>
<tr>
<td>Limited requests per day</td>
<td>None</td>
</tr>
<tr>
<td>50+</td>
<td>10+</td>
</tr>
</tbody>
</table>
On-Device Device Support

- iPhone 6s and later
- iPad (5th generation) and later
- All
// Recognizing pre-recorded audio

guard let recognizer = SFSpeechRecognizer() else {
    // Not supported for device's locale
    return
}

if !recognizer.isAvailable {
    // Not available right now
    return
}

let request = SFSpeechURLRecognitionRequest(url: url)
// Recognizing pre-recorded audio

guard let recognizer = SFSpeechRecognizer() else {
    // Not supported for device's locale
    return
}

if !recognizer.isAvailable {
    // Not available right now
    return
}

let request = SFSpeechURLRecognitionRequest(url: url)
// Recognizing pre-recorded audio

guard let recognizer = SFSpeechRecognizer() else {
    // Not supported for device's locale
    return
}
if !recognizer.isAvailable {
    // Not available right now
    return
}
let request = SFSpeechURLRecognitionRequest(url: url)
// Recognizing pre-recorded audio

guard let recognizer = SFSpeechRecognizer() else {
    // Not supported for device's locale
    return
}

if !recognizer.isAvailable {
    // Not available right now
    return
}

let request = SFSpeechURLRecognitionRequest(url: url)
// Recognizing pre-recorded audio

guard let recognizer = SFSpeechRecognizer() else {
    // Not supported for device's locale
    return
}

if !recognizer.isAvailable {
    // Not available right now
    return
}

let request = SFSpeechURLRecognitionRequest(url: url)

if speechRecognizer.supportsOnDeviceRecognition {
    request.requiresOnDeviceRecognition = true
}
// Recognizing pre-recorded audio

guard let recognizer = SFSpeechRecognizer() else {
    // Not supported for device's locale
    return
}

if !recognizer.isAvailable {
    // Not available right now
    return
}

let request = SFSpeechURLRecognitionRequest(url: url)

if speechRecognizer.supportsOnDeviceRecognition {
    request.requiresOnDeviceRecognition = true
}
Speech Recognition Results

Transcription

Alternative interpretations

Confidence levels

Timing information
Speech Recognition Results
Speech Recognition Results

Speaking rate
Speech Recognition Results

Speaking rate

Average pause duration
Speech Recognition Results

Speaking rate
Average pause duration
Voice Analytics features
Voice Analytics Features

Jitter

Measures variation in pitch

\[ Jitter = \frac{\text{Delta} 1 - \text{Delta} 2}{\text{mean}} \]
Voice Analytics Features

Shimmer

Measures variation in amplitude

\[
\text{Shimmer} = \frac{\text{Delta1} - \text{Delta2}}{\text{mean}}
\]
Voice Analytics Features
Voice Analytics Features

Audio with normal jitter and shimmer
Voice Analytics Features

Audio with normal jitter and shimmer

Audio with high jitter and shimmer
Voice Analytics Features

Pitch
Measures frequency characteristics of voice

Voicing
Identifies voiced regions in speech
// Printing new results when recognizing pre-recorded audio
let request = SFSpeechURLRecognitionRequest(url:url)
recognizer.recognitionTask(with: request) { (result, error) in
    guard let result = result else {
        // handle error
        return
    }
    if result.isFinal {
        let formattedString = result.bestTranscription.formattedString
        let speakingRate = result.bestTranscription.speakingRate
        let averagePauseDuration = result.bestTranscription.averagePauseDuration
    }
}
// Printing new results when recognizing pre-recorded audio
let request = SFSpeechURLRecognitionRequest(url:url)
recognizer.recognitionTask(with: request) { (result, error) in
    guard let result = result else {
        // handle error
        return
    }
    if result.isFinal {
        let formattedString = result.bestTranscription.formattedString
        let speakingRate = result.bestTranscription.speakingRate
        let averagePauseDuration = result.bestTranscription.averagePauseDuration
    }
}
// Printing new results when recognizing pre-recorded audio

if result.isFinal {
    let formattedString = result.bestTranscription.formattedString
    let speakingRate = result.bestTranscription.speakingRate
    let averagePauseDuration = result.bestTranscription.averagePauseDuration

    for segment in recognitionResult.bestTranscription.segments {
        let jitter = segment.voiceAnalytics?.jitter.acousticFeatureValuePerFrame
        let shimmer = segment.voiceAnalytics?.shimmer.acousticFeatureValuePerFrame
        let pitch = segment.voiceAnalytics?.pitch.acousticFeatureValuePerFrame
        let voicing = segment.voiceAnalytics?.voicing.acousticFeatureValuePerFrame
    }
}
// Printing new results when recognizing pre-recorded audio

if result.isFinal {
    let formattedString = result.bestTranscription.formattedString
    let speakingRate = result.bestTranscription.speakingRate
    let averagePauseDuration = result.bestTranscription.averagePauseDuration

    for segment in recognitionResult.bestTranscription.segments {
        let jitter = segment.voiceAnalytics?.jitter.acousticFeatureValuePerFrame
        let shimmer = segment.voiceAnalytics?.shimmer.acousticFeatureValuePerFrame
        let pitch = segment.voiceAnalytics?.pitch.acousticFeatureValuePerFrame
        let voicing = segment.voiceAnalytics?.voicing.acousticFeatureValuePerFrame
    }
}
Summary

Access to speech recognition in macOS

Run speech recognition on-device

Access to rich voice analytics features
More Information

developer.apple.com/wwdc19/256