Visão Geral de Machine Learning e Visão Computacional para Aplicativos iOS.

CoreML & Vision

Gabriela Bezerra - 2021

CoreML

https://developer.apple.com/documentation/coreml

Framework nativo criado e mantido pela Apple, usado para integrar ***Modelos de Machine Learning*** à uma aplicação feita para as plataformas da Apple.

A partir do dados providos pela aplicação, o modelo pode ser treinado, otimizado, e utilizado para fazer predições, tudo processado localmente no dispositivo.

Modelo

É o resultado da aplicação de um algoritmo de Machine Learning a um conjunto de dados de treinamento.

Na área de *Machine Learning*, modelos são utilizados principalmente para que, ***a partir de uma entrada desconhecida*** (ou seja, não processada durante a fase treinamento do modelo), ***uma predição correta possa ser feita*** e retornada na saída.

Essa predições geralmente são utilizadas para que um sistema seja capaz de realizar autonomamente a tarefa de ***classificação, detecção, e reconhecimento de padrões em dados desconhecidos,***

providos no input.

Modelos para o CoreML

Saiba mais:

New in Apple ML 2020

Para que um modelo seja compatível com o *Core ML*, *é* preciso que ele seja do tipo ***.mlmodel***. Esse tipo de modelo pode ser treinado utilizando umas das seguintes ferramentas: ***Create ML, Metal Performance Shaders, BNNS, ML Compute, e Turi Create***.

Além disso, ***modelos gerados por outras tecnologias podem ser convertidos utilizando o coremltools***, apesar de nem sempre isso ser recomendado. Uma vez que o modelo está implementado em um dispositivo, ele pode ser retreinado e otimizado dentro da aplicação, em tempo de execução, utilizando recursos do próprio *Core ML*.

Vantagens do CoreML

Performance 💪

Privacidade 🔒

De acordo com a documentação oficial da Apple, O *Core ML* ***tem a melhor performance nas plataformas da Apple*** porque otimiza o uso da *CPU*, *GPU*, e *Neural Engine* do dispositivo no qual está executando, além de minimizar o uso de memória *RAM* e consumo de bateria.

A capacidade de executar um modelo de ML localmente no dispositivo também é vantajosa quando vista da perspectiva de ***privacidade e segurança dos dados do usuário***, já que não há necessidade de conexão com servidores externos ao dispositivo nem tráfego de dados através da Internet.

Limitações do CoreML

Exclusivo Apple 🍎

Documentação em Inglês 🇺🇸

O *framework* só está disponível para as plataformas da Apple, que muitas vezes são caras e poucos acessíveis em países como o Brasil, onde grande parte da população utiliza dispositivos Android, seja por preferência, ou por não terem o poder aquisitivo necessário para consumir os produtos e serviços mais recentes da Apple.

Além disso, ***a documentação e a grande maioria dos recursos disponíveis online sobre a tecnologia está em Inglês***, configurando mais uma barreira no aprendizado e uso do framework.

Mesmo assim, **há mercado de desenvolvimento iOS nativo no Brasil, e empresas provendo esse tipo de equipamento a desenvolvedores de todos os níveis que precisarem dele**. Se você conseguir uma oportunidade desse tipo, espero que esse recurso seja útil pra você, e consiga ser um ponto de partida nos seus estudos.

Como Funciona?

O *Core ML* é uma ***camada de abstração*** que faz uso de outros *frameworks* com recursos relacionados a ML de mais baixo nível.

Accelerate

Apple Accelerate Framework Documentation

BNNS

Apple BNNS Library Documentation

Metal Performance Shaders

Apple Metal Performance Shaders Framework Documentation

Accelerate

Apple Accelerate Framework Documentation

Permite ao desenvolvedor fazer ***cálculos matemáticos de larga escala de maneira otimizada*** com alta performance e baixo consumo de energia nas plataformas da Apple;

BNNS

Apple BNNS Library Documentation

Biblioteca do *Accelerate* especialmente otimizada para realizar operações relacionadas a ***implementação, treinamento e inferência de redes neurais.***

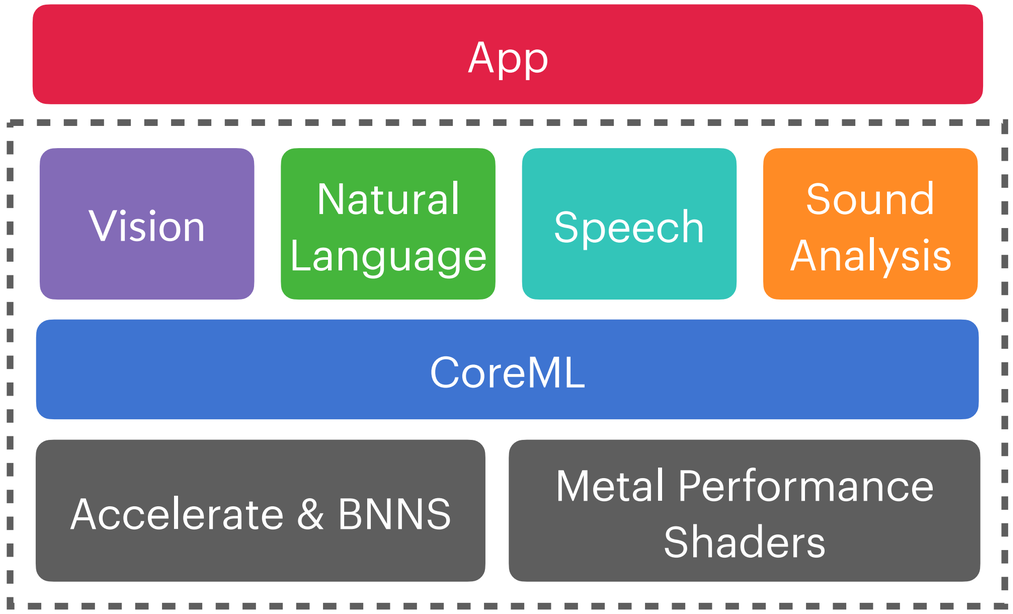
Metal Performance Shaders

Apple Metal Performance Shaders Framework Documentation

Principal ***framework de processamento de imagens*** da Apple, altamente otimizado para fazer operações utilizando as *GPUs* dos dispositivos que são compatíveis com ele.

O *CoreML* faz o uso desse *framework* durante o processamento de imagens com modelos relacionados a Visão Computacional.

Frameworks baseados no CoreML



Para facilitar as implementações mais comuns de Machine Learning em aplicativos iOS, existem **frameworks especializados** em determinadas subareas, cada um resolvendo um problema específico, e funcionando como uma abstração do CoreML. São eles:

Vision

https://developer.apple.com/documentation/vision

Natural Language

https://developer.apple.com/documentation/naturallanguage

Speech

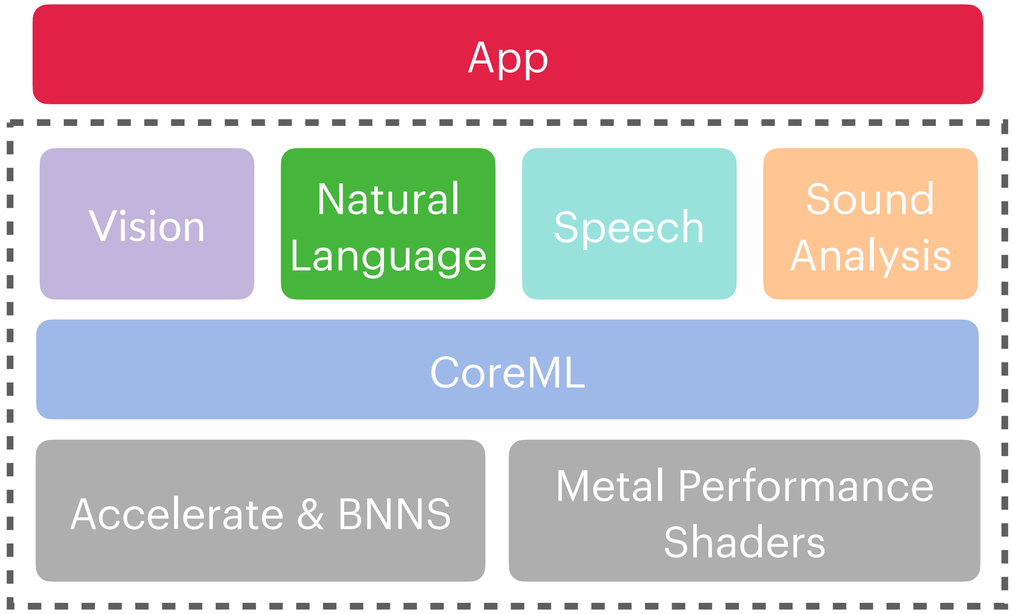
https://developer.apple.com/documentation/speech

Sound Analysis

https://developer.apple.com/documentation/soundanalysis

Natural Language

https://developer.apple.com/documentation/naturallanguage

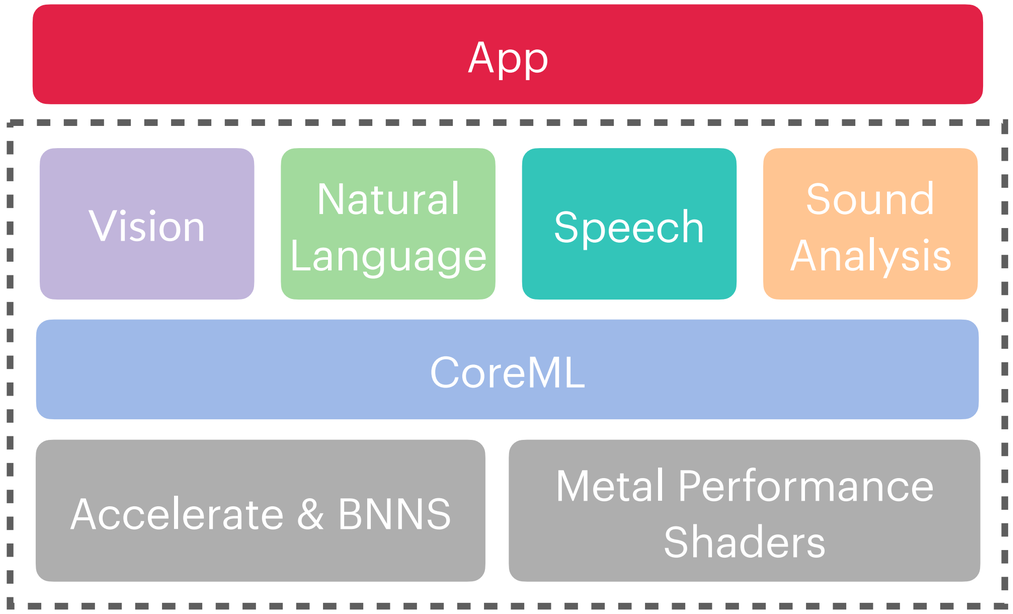


Framework de **Processamento Natural de Linguagem (NLP) para extração de metadados em textos de diversas linguagens.**

Esses metadados podem ser interpretados dentro do contexto do aplicativo como determinada emoção do usuário, ou o quanto ele ficou satisfeito ou não em uma determinada avaliação de produto, etc.

Speech

https://developer.apple.com/documentation/speech



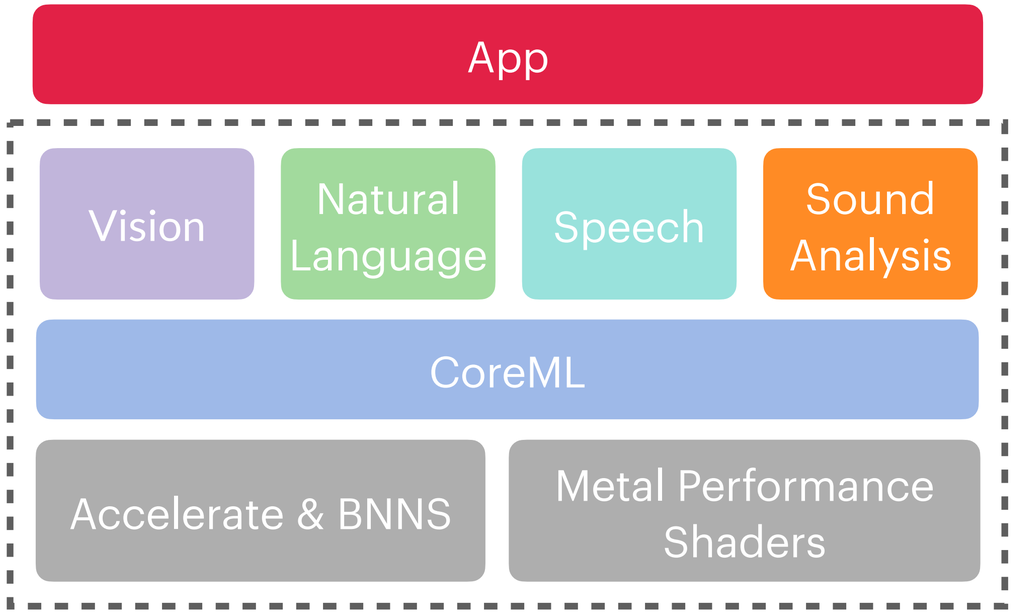
Framework de **Reconhecimento de Fala** em Audios.

Com ele é possível **gerar um texto** a partir de uma entrada de audio. Para detectar o "significado" dessa fala, é necessário utilizar outras tecnologias para processar o texto reconhecido pelo Speech, como por exemplo, os recursos que o Natural Language fornece.

Sound Analysis

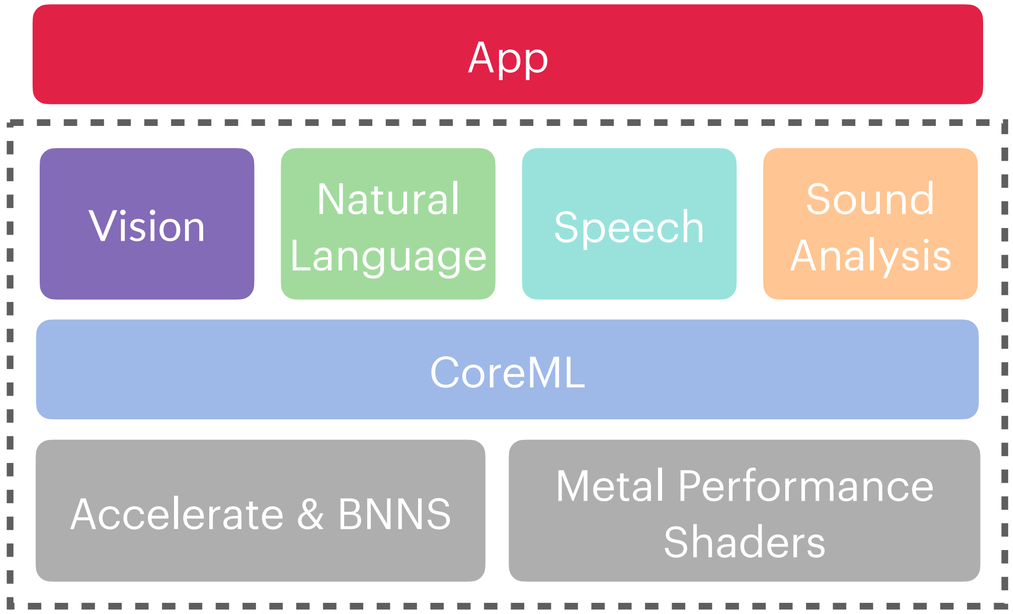
https://developer.apple.com/documentation/soundanalysis

Framework especializado em **processar áudio e encontrar padrões** sobre um contexto específico como "aplausos" ou “gargalhadas”.



Vision

https://developer.apple.com/documentation/vision



Framework especializado em ***Visão Computacional***, com o qual é possível utilizar modelos pré-treinados e métodos para **classificar imagens, realizar a detecção de rosto (com *face landmarks)*, texto, corpo humano, posição corporal, posição das mãos**, entre outros padrões e objetos.

Como o Vision Funciona?

Explore Computer Vision APIs - WWDC2020

De acordo com o vídeo Explore Computer Vision APIs da WWDC20, o funcionamento do Vision pode ser entendido em 3 partes:

Tarefa – **VNRequest** que pode ser de vários tipos, dependendo do modelo utilizado. Ela é configurada com Modelo e Callback.

Maquinário – Handler que executa a request para determinado Input e gera os Resultados. Pode ser de uma imagem **ImageRequestHandler** ou sequencia de imagens **SequenceRequestHandler**.

Resultado – **VNObsevartion** que pode ser de vários tipos e representar vários aspectos da imagem, dependendo do modelo utilizado. Essas são as chamadas features extraídas da imagem processada.

Com o Vision, e possível fazer a extração de Features de uma imagem, mas não é possível classificá-las. Para isso é necessário uso do CoreML.

Vision na prática: Explore você mesmo!

Repositório com o Projeto:

https://github.com/GabrielaBezerra/VisionCoreMLExample

Existe um mundo enorme de possibilidades para *Machine Learning* e Visão Computacional utilizando somente os frameworks da Apple CoreML e Vision.

Os frameworks são responsáveis por executar os modelos, processar input e outputs, e facilitam bastante o trabalho de desenvolvedores que querem utilizar diversos modelos em aplicações tanto para a construção de aplicativos quanto para a realização de pesquisas científicas no contexto Mobile.

Como prova de conceito e recurso de experimentação, você pode utilizar esse projeto de exemplo implementando o uso de Vision para extração de features de imagens e CoreML com modelos **.mlmodel** retirados do site oficial da Apple para classificar essas features e descobrir o que é a imagem e quais detectar quais objetos estão nela.

Sinta-se livre para explorá-lo!