# Hey Siri：一种用于苹果个人助理的，基于设备DNN的语音触发器 Hey Siri：一种用于苹果个人助理的，基于设备DNN的语音触发器 原文链接： “嘿Siri”功能使用户可以免提调用Siri。 一个很小的语音识别器一直在运行，只听这两个词。 当检测到“嘿Siri”时，其余Siri会将以下语音解析为命令或查询。 “ Hey Siri”检测器使用深度神经网络（DNN）将每个时刻的声音声学模式转换为语音的概率分布。 然后，它使用时间积分过程来计算您说出的短语为“ Hey Siri”的置信度得分。 如果分数足够高，Siri就会醒来。 本文介绍了基础技术。 它主要面向了解机器学习但很少了解语音识别的读者。

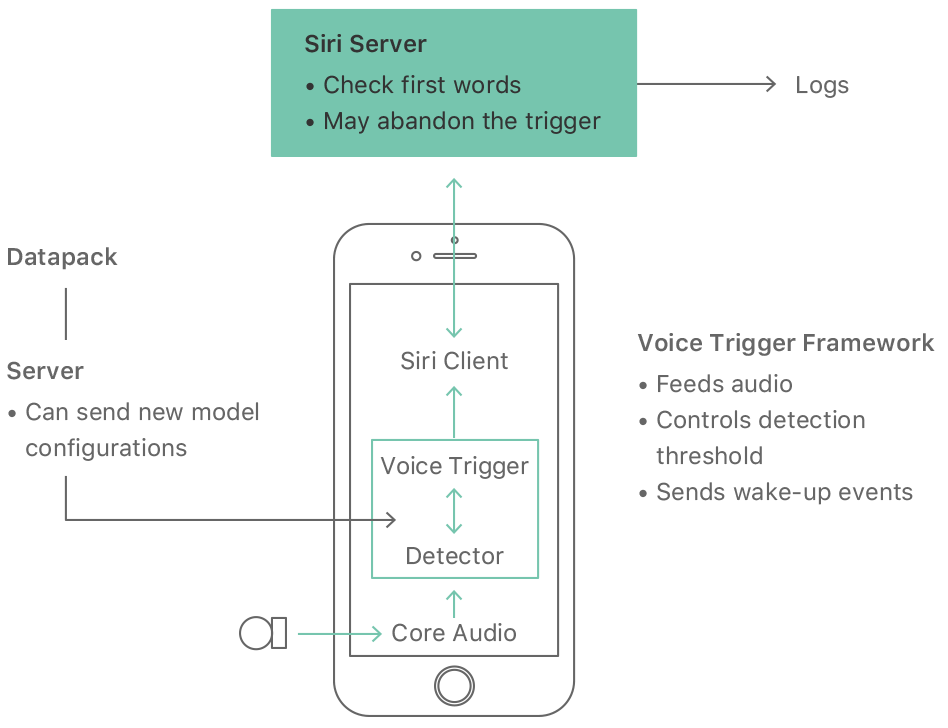
原文链接：[Hey Siri: An On-device DNN-powered Voice Trigger for Apple’s Personal Assistant](https://machinelearning.apple.com/2017/10/01/hey-siri.html?from=hackcv&hmsr=hackcv.com&utm_medium=hackcv.com&utm_source=hackcv.com)

“嘿Siri”功能使用户可以免提调用Siri。 一个很小的语音识别器一直在运行，只听这两个词。 当检测到“嘿Siri”时，其余Siri会将以下语音解析为命令或查询。 “ Hey Siri”检测器使用深度神经网络（DNN）将每个时刻的声音声学模式转换为语音的概率分布。 然后，它使用时间积分过程来计算您说出的短语为“ Hey Siri”的置信度得分。 如果分数足够高，Siri就会醒来。 本文介绍了基础技术。 它主要面向了解机器学习但很少了解语音识别的读者。

# 免提访问Siri 免提访问Siri 要获得Siri的帮助，请说“嘿Siri”。 无需按下按钮，因为“嘿Siri”使Siri免提。 看起来很简单，但是在幕后进行了大量工作以快速有效地唤醒Siri。 硬件，软件和Internet服务可以无缝协作，以提供出色的体验。 图1. iPhone上的Hey Siri流 在忙碌的时候（例如做饭或开车时）或使用Apple Watch时，无需按下按钮就可以使用Siri尤其有用。 如图1所示，整个系统包含几个部分。 Siri的大多数实现是“在云端”，包括主要的自动语音识别，自然语言解释和各种信息服务。 还有一些服务器可以为检测器使用的声学模型提供更新。 本文重点介绍在本地设备（如iPhone或Apple Watch）上运行的部分。 特别是，它专注于检测器：一种专用的语音识别器，始终仅侦听其唤醒短语（在启用了“ Hey Siri”功能的最新iPhone上）。

要获得Siri的帮助，请说“嘿Siri”。 无需按下按钮，因为“嘿Siri”使Siri免提。 看起来很简单，但是在幕后进行了大量工作以快速有效地唤醒Siri。 硬件，软件和Internet服务可以无缝协作，以提供出色的体验。

图1. iPhone上的Hey Siri流

[](https://machinelearning.apple.com/images/journals/hey-siri/HeySiriFlow-1.png)

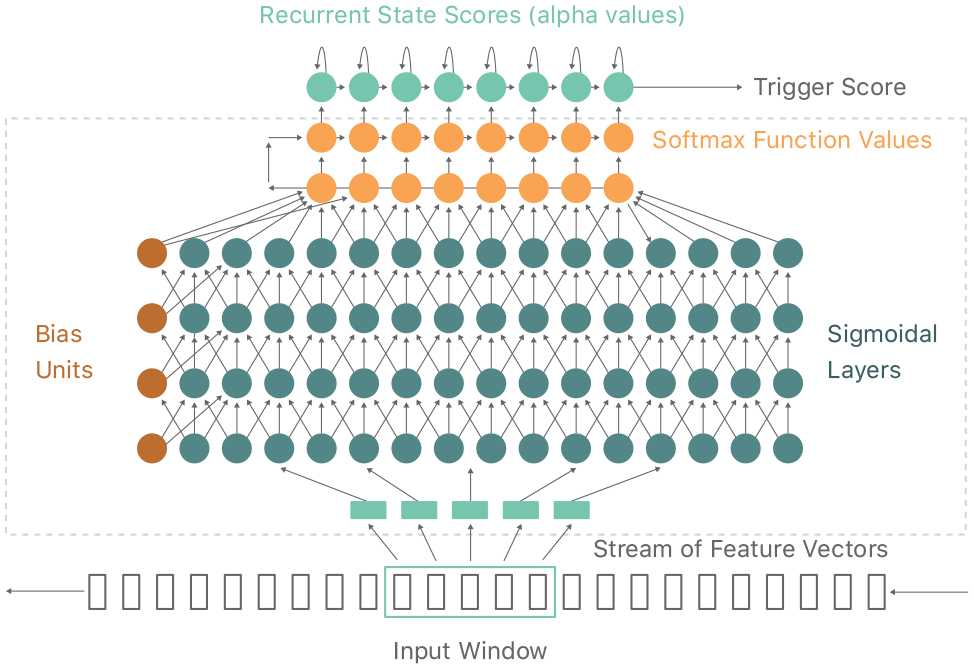
在忙碌的时候（例如做饭或开车时）或使用Apple Watch时，无需按下按钮就可以使用Siri尤其有用。 如图1所示，整个系统包含几个部分。 Siri的大多数实现是“在云端”，包括主要的自动语音识别，自然语言解释和各种信息服务。 还有一些服务器可以为检测器使用的声学模型提供更新。 本文重点介绍在本地设备（如iPhone或Apple Watch）上运行的部分。 特别是，它专注于检测器：一种专用的语音识别器，始终仅侦听其唤醒短语（在启用了“ Hey Siri”功能的最新iPhone上）。

# 侦探：聆听“嘿Siri” 侦探：聆听“嘿Siri” iPhone或Apple Watch中的麦克风将您的声音转换成瞬时波形样本流，速率为每秒16000。频谱分析阶段将波形样本流转换为一系列帧，每个帧描述大约0.01秒的声谱。一次约有20个这些帧（0.2秒的音频）被馈送到声学模型，即深度神经网络（DNN），该网络将每个声学模式转换为一组语音类别的概率分布： “嘿Siri”一词，再加上静音和其他语音，总共约有20种声音类别。参见图2。 DNN主要由矩阵乘法和逻辑非线性组成。每个“隐藏”层都是DNN在对其进行训练以将滤波器组输入转换为声音类的训练过程中发现的中间表示。最终的非线性本质上是一个Softmax函数（也就是一般的对数或归一化指数），但是由于我们需要对数概率，因此实际的数学运算较为简单。

iPhone或Apple Watch中的麦克风将您的声音转换成瞬时波形样本流，速率为每秒16000。频谱分析阶段将波形样本流转换为一系列帧，每个帧描述大约0.01秒的声谱。一次约有20个这些帧（0.2秒的音频）被馈送到声学模型，即深度神经网络（DNN），该网络将每个声学模式转换为一组语音类别的概率分布： “嘿Siri”一词，再加上静音和其他语音，总共约有20种声音类别。参见图2。

DNN主要由矩阵乘法和逻辑非线性组成。每个“隐藏”层都是DNN在对其进行训练以将滤波器组输入转换为声音类的训练过程中发现的中间表示。最终的非线性本质上是一个Softmax函数（也就是一般的对数或归一化指数），但是由于我们需要对数概率，因此实际的数学运算较为简单。

图2.用于检测“嘿Siri”的深度神经网络。隐藏层实际上是完全连接的。顶层执行时间整合。实际的DNN由虚线框指示。  
图2.用于检测“嘿Siri”的深度神经网络。隐藏层实际上是完全连接的。顶层执行时间整合。实际的DNN由虚线框指示。  
我们选择DNN每个隐藏层中的单位数，以适合“ Hey Siri”检测器运行时可用的计算资源。我们使用的网络通常具有五个隐藏层，大小均相同：32、128或192个单元，具体取决于内存和电源限制。在iPhone上，我们使用两个网络-一个用于初始检测，另一个作为辅助检查器。初始检测器使用的单元少于辅助检测器。  
声学模型的输出为每个帧的语音分类提供了分数分布。语音课通常类似于“ / s /的第一部分，后跟高前元音，然后是前元音”。

[](https://machinelearning.apple.com/images/journals/hey-siri/TrainingDNN-1.png)

我们选择DNN每个隐藏层中的单位数，以适合“ Hey Siri”检测器运行时可用的计算资源。我们使用的网络通常具有五个隐藏层，大小均相同：32、128或192个单元，具体取决于内存和电源限制。在iPhone上，我们使用两个网络-一个用于初始检测，另一个作为辅助检查器。初始检测器使用的单元少于辅助检测器。

声学模型的输出为每个帧的语音分类提供了分数分布。语音课通常类似于“ / s /的第一部分，后跟高前元音，然后是前元音”。

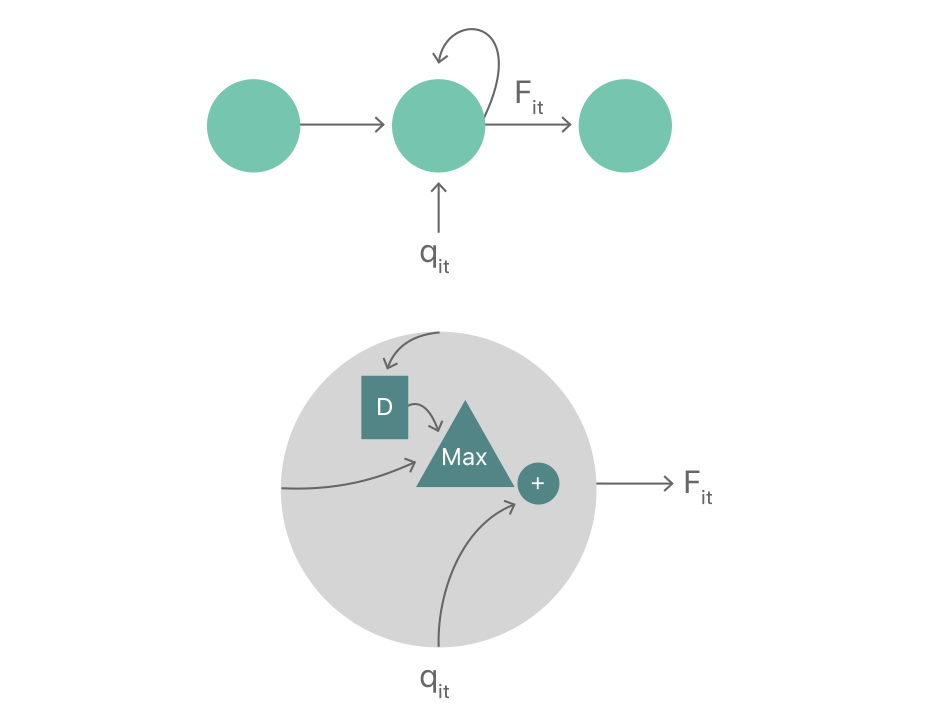
如果声学模型的输出在目标短语的正确序列中较高，我们想检测“ Hey Siri”。为了对每个帧产生单个分数，我们会随着时间的推移按有效顺序累积这些局部值。这在图2的最后（顶层）中表示为循环网络，该循环网络具有到同一单元的连接以及依次连接到下一个的单元。每个单元内都有一个最大操作和一个附加操作：  
如果声学模型的输出在目标短语的正确序列中较高，我们想检测“ Hey Siri”。为了对每个帧产生单个分数，我们会随着时间的推移按有效顺序累积这些局部值。这在图2的最后（顶层）中表示为循环网络，该循环网络具有到同一单元的连接以及依次连接到下一个的单元。每个单元内都有一个最大操作和一个附加操作：  
Fi,t=max{si+Fi,t−1,mi−1+Fi−1,t−1}+qi,tFi,t=max{si+Fi,t-1,mi-1+Fi-1,t-1}+qi,t  
where  
Fi,t 是模型状态i的累计得分  
qi,t 是声学模型的输出-在给定时间t周围的声学模式的情况下，与ith状态相关的语音类的对数得分

Fi,t=max{si+Fi,t−1,mi−1+Fi−1,t−1}+qi,tFi,t=max{si+Fi,t-1,mi-1+Fi-1,t-1}+qi,t

where

* *Fi,t* 是模型状态*i*的累计得分
* *qi,t* 是声学模型的输出-在给定时间*t*周围的声学模式的情况下，与*ith*状态相关的语音类的对数得分
* si 是与保持状态相关的成本i  
  si 是与保持状态相关的成本i  
  mi 是从状态移动的代价 i  
  si和 mi均基于对训练数据中带有相关标签的分段持续时间的分析。 （此过程是动态编程的应用程序，可以基于关于隐马尔可夫模型（HMM）的思想派生。）  
  图3.方程的直观描述  
  每个累积得分Fi，t与状态的先前帧的标记相关联，如最大操作的决策序列所给定的。每帧的最终分数是Fi，t，其中词组的最后状态是状态I，并且在导致该分数的帧序列中有N帧。 （N可以通过追溯最大决策序列来找到，但实际上是通过向前传播自路径进入短语的第一状态以来的帧数来完成的。）
* *mi* 是从状态移动的代价 *i*
* *si*和 *mi*均基于对训练数据中带有相关标签的分段持续时间的分析。 （此过程是动态编程的应用程序，可以基于关于隐马尔可夫模型（HMM）的思想派生。）

图3.方程的直观描述

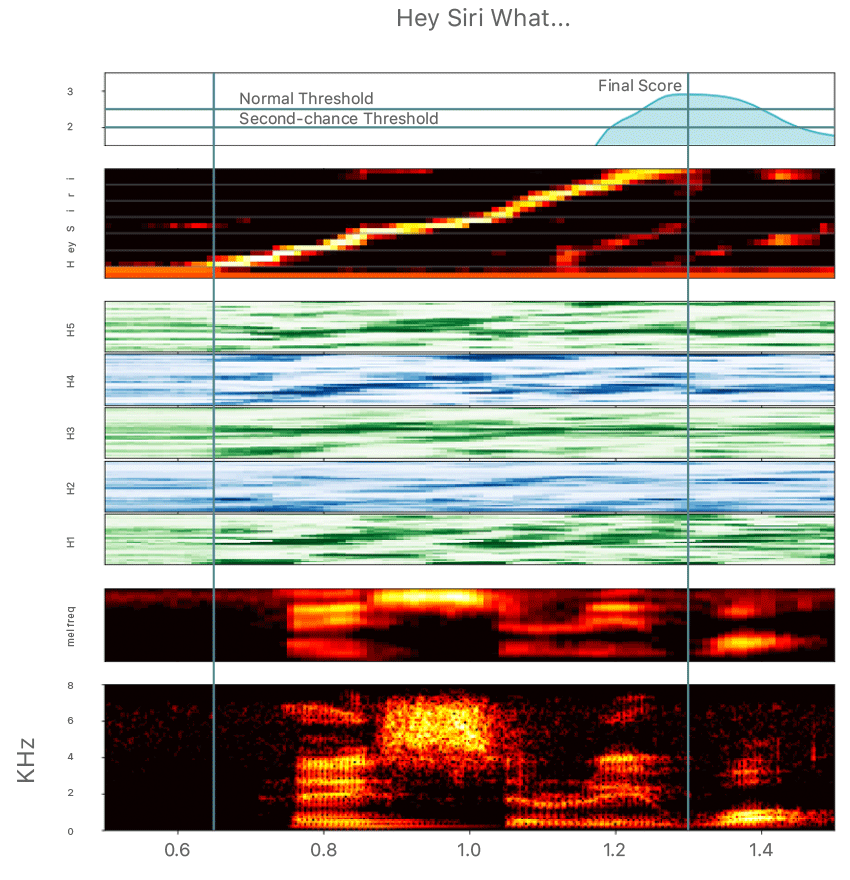
[](https://machinelearning.apple.com/images/journals/hey-siri/equation-1.png)

每个累积得分*Fi，t*与状态的先前帧的标记相关联，如最大操作的决策序列所给定的。每帧的最终分数是*Fi，t*，其中词组的最后状态是状态I，并且在导致该分数的帧序列中有N帧。 （N可以通过追溯最大决策序列来找到，但实际上是通过向前传播自路径进入短语的第一状态以来的帧数来完成的。）

“ Hey Siri”探测器中的几乎所有计算都在声学模型中进行。时间积分计算相对便宜，因此在评估大小或计算资源时我们将其忽略。  
“ Hey Siri”探测器中的几乎所有计算都在声学模型中进行。时间积分计算相对便宜，因此在评估大小或计算资源时我们将其忽略。  
通过查看图4，您可以更好地了解检测器的工作原理，该图显示了假设使用最小DNN的各个阶段的声信号。最底部是麦克风波形的频谱图。在这种情况下，有人说“嘿Siri，什么……”。较亮的部分是该短语中最响亮的部分。 Hey Siri模式位于垂直蓝线之间。  
图4.穿过探测器的声学模式  
从底部向上的第二个水平条显示了使用梅尔滤波器组分析相同波形的结果，该滤波器组基于感知测量值赋予了频率权重。这种转换也使声谱图中可见的细节变得平滑，这归因于声道激励的精细结构：是随机的（如/ s /），还是周期性的（在此处视为垂直条纹）。

通过查看图4，您可以更好地了解检测器的工作原理，该图显示了假设使用最小DNN的各个阶段的声信号。最底部是麦克风波形的频谱图。在这种情况下，有人说“嘿Siri，什么……”。较亮的部分是该短语中最响亮的部分。 Hey Siri模式位于垂直蓝线之间。

图4.穿过探测器的声学模式

[](https://machinelearning.apple.com/images/journals/hey-siri/hswr-1.png)

从底部向上的第二个水平条显示了使用梅尔滤波器组分析相同波形的结果，该滤波器组基于感知测量值赋予了频率权重。这种转换也使声谱图中可见的细节变得平滑，这归因于声道激励的精细结构：是随机的（如/ s /），还是周期性的（在此处视为垂直条纹）。

标记为H1到H5的绿色和蓝色交替水平条显示了五个隐藏层中每个层的单位的数值（激活）。为该图安排了每层中的32个隐藏单元，以便将具有相似输出的单元放在一起。  
标记为H1到H5的绿色和蓝色交替水平条显示了五个隐藏层中每个层的单位的数值（激活）。为该图安排了每层中的32个隐藏单元，以便将具有相似输出的单元放在一起。  
下一个条带（带有黄色对角线）显示了声学模型的输出。在每一帧中，短语中每个位置都有一个输出，另外还有一个输出用于静音和其他语音。根据等式1，将沿明亮的对角线的局部分数相加即可得到最终分数，显示在顶部。请注意，在整个短语进入系统后，分数便上升到一个峰值。  
我们将分数与阈值进行比较，以决定是否激活Siri。实际上，阈值不是固定值。我们提供了一些灵活性，可以使在困难条件下更轻松地激活Siri，而不会显着增加错误激活的次数。有一个主要阈值或正常阈值，以及一个通常不会触发Siri的较低阈值。如果分数超过了下限阈值，但没有超过上限阈值，则可能是我们错过了真正的“ Hey Siri”事件。当分数在此范围内时，系统会进入更敏感的状态几秒钟，因此，即使用户不花力气就重复该短语，Siri也会触发。该第二次机会机制显着提高了系统的可用性，而又不会过多地提高误报率，因为它只是在短时间内处于这种超敏感状态。 （我们稍后讨论测试和调整的准确性。）

下一个条带（带有黄色对角线）显示了声学模型的输出。在每一帧中，短语中每个位置都有一个输出，另外还有一个输出用于静音和其他语音。根据等式1，将沿明亮的对角线的局部分数相加即可得到最终分数，显示在顶部。请注意，在整个短语进入系统后，分数便上升到一个峰值。

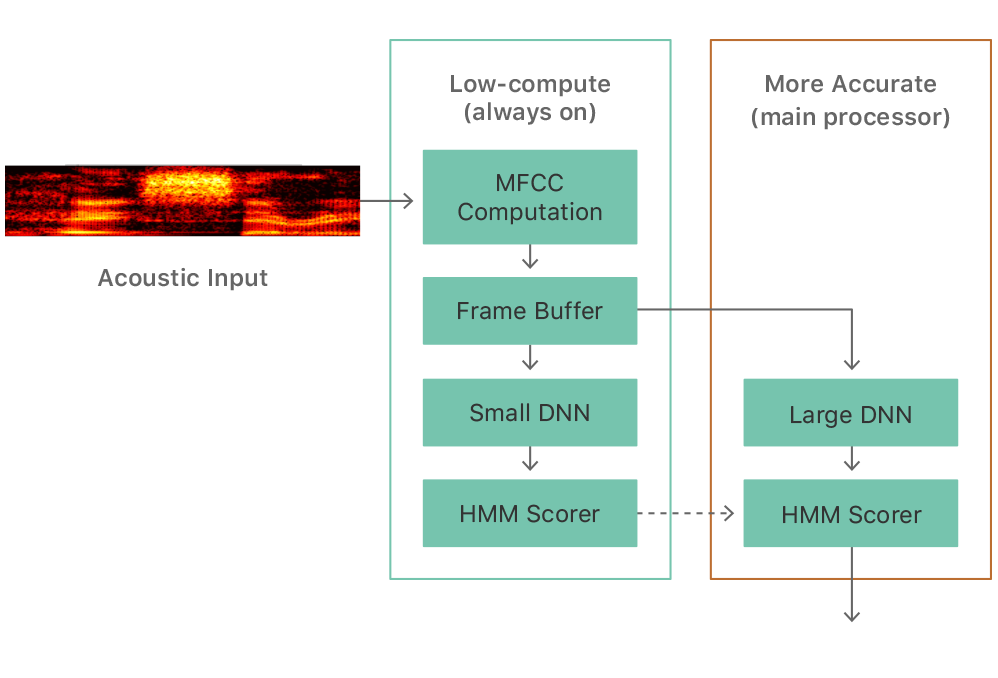
我们将分数与阈值进行比较，以决定是否激活Siri。实际上，阈值不是固定值。我们提供了一些灵活性，可以使在困难条件下更轻松地激活Siri，而不会显着增加错误激活的次数。有一个主要阈值或正常阈值，以及一个通常不会触发Siri的较低阈值。如果分数超过了下限阈值，但没有超过上限阈值，则可能是我们错过了真正的“ Hey Siri”事件。当分数在此范围内时，系统会进入更敏感的状态几秒钟，因此，即使用户不花力气就重复该短语，Siri也会触发。该第二次机会机制显着提高了系统的可用性，而又不会过多地提高误报率，因为它只是在短时间内处于这种超敏感状态。 （我们稍后讨论测试和调整的准确性。）

# 响应能力和功率：两次通过检测 响应能力和功率：两次通过检测 “ Hey Siri”检测器不仅必须准确，而且还必须快速且不会对电池寿命产生重大影响。 我们还需要最小化内存使用和处理器需求，尤其是高峰处理器需求。 为避免整日运行主处理器只是为了听触发语，iPhone的Always On处理器（AOP）（小型，低功耗辅助处理器，即嵌入式运动协处理器）可以访问麦克风信号（在 6S及更高版本）。 我们使用AOP有限的有限处理能力中的一小部分来运行具有较小声学模型（DNN）的探测器。 当分数超过阈值时，运动协处理器唤醒主处理器，该主处理器使用较大的DNN分析信号。 在具有AOP支持的第一个版本中，第一个检测器使用具有5层32个隐藏单元的DNN，第二个检测器使用5层192个隐藏单元。

“ Hey Siri”检测器不仅必须准确，而且还必须快速且不会对电池寿命产生重大影响。 我们还需要最小化内存使用和处理器需求，尤其是高峰处理器需求。

为避免整日运行主处理器只是为了听触发语，iPhone的Always On处理器（AOP）（小型，低功耗辅助处理器，即嵌入式运动协处理器）可以访问麦克风信号（在 6S及更高版本）。 我们使用AOP有限的有限处理能力中的一小部分来运行具有较小声学模型（DNN）的探测器。 当分数超过阈值时，运动协处理器唤醒主处理器，该主处理器使用较大的DNN分析信号。 在具有AOP支持的第一个版本中，第一个检测器使用具有5层32个隐藏单元的DNN，第二个检测器使用5层192个隐藏单元。

图5.两遍检测  
图5.两遍检测  
由于电池要小得多，Apple Watch面临一些特殊的挑战。 Apple Watch使用单次通过的“ Hey Siri”检测器，其声学模型的大小介于其他iOS设备的第一次和第二次通过之间。 “ Hey Siri”检测器仅在手表运动协处理器检测到手腕抬起手势（这会打开屏幕）时运行。 那时，WatchOS有很多工作要做-开机，准备屏幕等。因此，系统仅在相当有限的计算预算中分配一小部分（〜5％）给“ Hey Siri”。 及时开始音频捕获以捕获触发短语的开始是一个挑战，因此我们在初始化检测器的方式中考虑了可能的截断。

[](https://machinelearning.apple.com/images/journals/hey-siri/TwoPassDetector-1.png)

由于电池要小得多，Apple Watch面临一些特殊的挑战。 Apple Watch使用单次通过的“ Hey Siri”检测器，其声学模型的大小介于其他iOS设备的第一次和第二次通过之间。 “ Hey Siri”检测器仅在手表运动协处理器检测到手腕抬起手势（这会打开屏幕）时运行。 那时，WatchOS有很多工作要做-开机，准备屏幕等。因此，系统仅在相当有限的计算预算中分配一小部分（〜5％）给“ Hey Siri”。 及时开始音频捕获以捕获触发短语的开始是一个挑战，因此我们在初始化检测器的方式中考虑了可能的截断。

# “嘿Siri”个性化 “嘿Siri”个性化 我们设计了永远在线的“ Hey Siri”探测器，以使附近的任何人说出触发短语时都能做出响应。为了减少错误触发的烦恼，我们邀请用户参加简短的注册会议。在注册过程中，用户说出五个短语，每个短语都以“ Hey Siri”开头。我们将这些示例保存在设备上。 我们将任何可能的新“嘿Siri”语音与存储的示例进行比较，如下所示。 （第二遍）检测器通过获取与每种状态对齐的帧的平均值，生成用于将声学模式转换为固定长度矢量的时序信息。单独的，经过特殊训练的DNN将此向量转换为“扬声器空间”，根据设计，该空间中来自同一扬声器的模式趋于接近，而来自不同扬声器的模式趋于彼此分离。我们将距离与在注册期间创建的参考图案的距离与另一个阈值进行比较，以确定触发检测器的声音是否很可能是注册用户说的“ Hey Siri”。

我们设计了永远在线的“ Hey Siri”探测器，以使附近的任何人说出触发短语时都能做出响应。为了减少错误触发的烦恼，我们邀请用户参加简短的注册会议。在注册过程中，用户说出五个短语，每个短语都以“ Hey Siri”开头。我们将这些示例保存在设备上。

我们将任何可能的新“嘿Siri”语音与存储的示例进行比较，如下所示。 （第二遍）检测器通过获取与每种状态对齐的帧的平均值，生成用于将声学模式转换为固定长度矢量的时序信息。单独的，经过特殊训练的DNN将此向量转换为“扬声器空间”，根据设计，该空间中来自同一扬声器的模式趋于接近，而来自不同扬声器的模式趋于彼此分离。我们将距离与在注册期间创建的参考图案的距离与另一个阈值进行比较，以确定触发检测器的声音是否很可能是注册用户说的“ Hey Siri”。

此过程不仅降低了另一个人说“嘿Siri”将触发iPhone的可能性，而且还降低了其他类似发音的短语触发Siri的速度。  
此过程不仅降低了另一个人说“嘿Siri”将触发iPhone的可能性，而且还降低了其他类似发音的短语触发Siri的速度。  
进一步检查  
如果iPhone上的各个阶段通过，波形将到达Siri服务器。 如果主语音识别器听到的声音不是“ Hey Siri”（例如“ Hey严重”），则服务器将取消信号发送到电话以使其重新进入睡眠状态，如图1所示。 在设备上运行主识别器的精简版，以提早进行额外检查。  
声学模型：训练  
DNN声学模型是“ Hey Siri”探测器的核心。因此，让我们看一下我们如何训练它。在提供Hey Siri功能之前，一小部分用户会在请求开始时说“ Hey Siri”，方法是先按下按钮。我们在美国英语探测器模型的初始训练中使用了此类“嘿Siri”语音。我们还包括用于训练主要语音识别器的一般语音示例。在这两种情况下，我们都在训练短语上使用了自动转录。 Siri团队成员检查了转录的子集的准确性。

# 进一步检查

如果iPhone上的各个阶段通过，波形将到达Siri服务器。 如果主语音识别器听到的声音不是“ Hey Siri”（例如“ Hey严重”），则服务器将取消信号发送到电话以使其重新进入睡眠状态，如图1所示。 在设备上运行主识别器的精简版，以提早进行额外检查。

# 声学模型：训练

DNN声学模型是“ Hey Siri”探测器的核心。因此，让我们看一下我们如何训练它。在提供Hey Siri功能之前，一小部分用户会在请求开始时说“ Hey Siri”，方法是先按下按钮。我们在美国英语探测器模型的初始训练中使用了此类“嘿Siri”语音。我们还包括用于训练主要语音识别器的一般语音示例。在这两种情况下，我们都在训练短语上使用了自动转录。 Siri团队成员检查了转录的子集的准确性。

我们为“ Hey Siri”短语创建了特定于语言的语音规范。在美式英语中，我们有两种变体，“ Siri”中的第一个元音不同-一个在“ Serious”中，另一个在“ Syria”中。我们还试图应对两个词之间的短暂中断，尤其是这个短语通常用逗号写：“嘿，Siri。”每个注音符号会产生三个语音类别（开始，中间和结尾），每个类别都有自己的声学模型输出。  
我们为“ Hey Siri”短语创建了特定于语言的语音规范。在美式英语中，我们有两种变体，“ Siri”中的第一个元音不同-一个在“ Serious”中，另一个在“ Syria”中。我们还试图应对两个词之间的短暂中断，尤其是这个短语通常用逗号写：“嘿，Siri。”每个注音符号会产生三个语音类别（开始，中间和结尾），每个类别都有自己的声学模型输出。  
我们使用了语料库来训练DNN，主要Siri识别器为此提供了每帧声音类别的标签。主识别器使用了数千种声音类别，但是只需要大约20个声音类别就可以解释目标短语（包括初始静音），而一个大的类别则需要处理其他所有类别。训练过程仅根据本地声音模式，尝试为标有相关状态和电话的帧生成接近1的DNN输出。训练过程使用标准的反向传播和随机梯度下降来调整权重。我们已经使用了各种神经网络培训软件工具包，包括Theano，Tensorflow和Kaldi。

我们使用了语料库来训练DNN，主要Siri识别器为此提供了每帧声音类别的标签。主识别器使用了数千种声音类别，但是只需要大约20个声音类别就可以解释目标短语（包括初始静音），而一个大的类别则需要处理其他所有类别。训练过程仅根据本地声音模式，尝试为标有相关状态和电话的帧生成接近1的DNN输出。训练过程使用标准的反向传播和随机梯度下降来调整权重。我们已经使用了各种神经网络培训软件工具包，包括Theano，Tensorflow和Kaldi。

该训练过程会给出电话的概率估计值，并根据给定的本地声学观测值进行状态估计，但是这些估计值包括训练集中的电话频率（先验先验），这可能很不均匀，并且与频率无关。在使用检测器的情况下，因此我们在使用声学模型输出之前先验先验。  
该训练过程会给出电话的概率估计值，并根据给定的本地声学观测值进行状态估计，但是这些估计值包括训练集中的电话频率（先验先验），这可能很不均匀，并且与频率无关。在使用检测器的情况下，因此我们在使用声学模型输出之前先验先验。  
训练一个模型大约需要一天的时间，并且通常在任何一次训练中都有几个模型。我们通常会训练三个版本：用于运动协处理器的第一遍的小模型，用于第二遍的大尺寸的模型和用于Apple Watch的中型模型。  
“ Hey Siri”可以使用Siri支持的所有语言，但是“ Hey Siri”不一定是开始监听Siri的短语。例如，说法语的用户需要说“ Dis Siri”，而说韩语的用户要说“ Siri야”（听起来像“ Siri Ya”。）在俄语中是“приветSiri”（听起来像“ Privet Siri”），以及泰文中的“หวัดดีSiri”。 （类似“ Wadi Siri”的声音。）

训练一个模型大约需要一天的时间，并且通常在任何一次训练中都有几个模型。我们通常会训练三个版本：用于运动协处理器的第一遍的小模型，用于第二遍的大尺寸的模型和用于Apple Watch的中型模型。

“ Hey Siri”可以使用Siri支持的所有语言，但是“ Hey Siri”不一定是开始监听Siri的短语。例如，说法语的用户需要说“ Dis Siri”，而说韩语的用户要说“ Siri야”（听起来像“ Siri Ya”。）在俄语中是“приветSiri”（听起来像“ Privet Siri”），以及泰文中的“หวัดดีSiri”。 （类似“ Wadi Siri”的声音。）

## 测试与微调 测试与微调 理想的检测器会在用户说“嘿Siri”时触发，而在其他时间不触发。我们用两种误差来描述探测器的精度：在错误的时间触发和在正确的时间触发失败。错误接受率（FAR或错误警报率）是每小时错误激活的次数（或两次激活之间的平均小时数），错误拒绝率（FRR）是失败的尝试激活的比例。 （请注意，我们用于测量FAR的单位与用于FRR的单位不同。甚至尺寸也不同。因此，没有错误率相等的概念。） 对于给定的模型，我们可以通过更改激活阈值来更改两种错误之间的平衡。图6显示了两种尺寸的早期开发模型的折衷示例。更改阈值将沿着曲线移动。

理想的检测器会在用户说“嘿Siri”时触发，而在其他时间不触发。我们用两种误差来描述探测器的精度：在错误的时间触发和在正确的时间触发失败。错误接受率（FAR或错误警报率）是每小时错误激活的次数（或两次激活之间的平均小时数），错误拒绝率（FRR）是失败的尝试激活的比例。 （请注意，我们用于测量FAR的单位与用于FRR的单位不同。甚至尺寸也不同。因此，没有错误率相等的概念。）

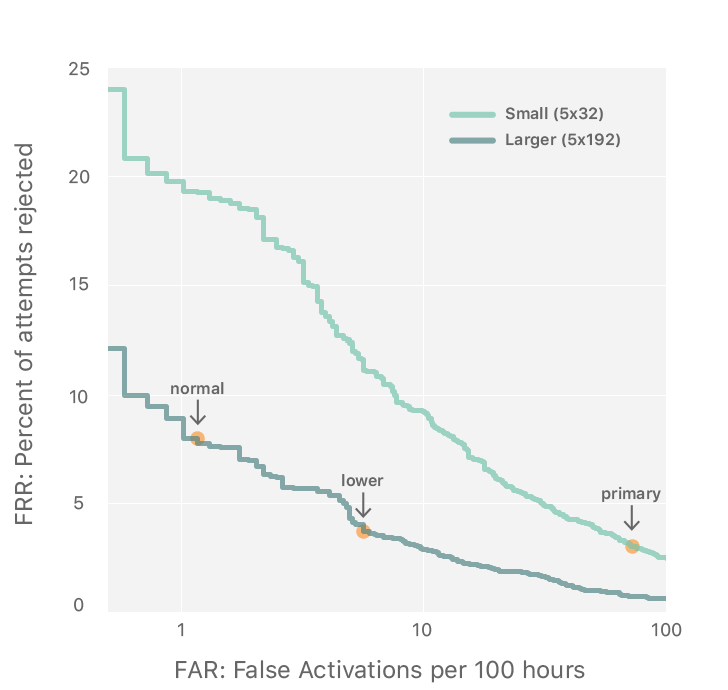
对于给定的模型，我们可以通过更改激活阈值来更改两种错误之间的平衡。图6显示了两种尺寸的早期开发模型的折衷示例。更改阈值将沿着曲线移动。

在开发过程中，我们尝试通过使用大型测试集来估计系统的准确性，该测试集的收集和准备工作相当昂贵，但必不可少。有“正”数据和“负”数据。 “阳性”数据确实包含目标短语。您可能会认为我们可以使用“ Hey Siri”系统收集到的语音，但是该系统无法捕获失败触发的尝试，因此我们希望改进该系统以尽可能多地包含此类失败尝试。  
在开发过程中，我们尝试通过使用大型测试集来估计系统的准确性，该测试集的收集和准备工作相当昂贵，但必不可少。有“正”数据和“负”数据。 “阳性”数据确实包含目标短语。您可能会认为我们可以使用“ Hey Siri”系统收集到的语音，但是该系统无法捕获失败触发的尝试，因此我们希望改进该系统以尽可能多地包含此类失败尝试。  
最初，我们使用了某些用户在按下“主页”按钮时所说的“嘿Siri”的发音，但这些用户并不是想引起Siri的注意（该按钮可以这样做），并且麦克风一定会伸手可及，而我们也希望“嘿Siri”在整个房间中工作。我们专门以各种语言在各种条件下（例如，在厨房（近处和远处），汽车，卧室和餐厅中）用每种语言的母语进行录音。

最初，我们使用了某些用户在按下“主页”按钮时所说的“嘿Siri”的发音，但这些用户并不是想引起Siri的注意（该按钮可以这样做），并且麦克风一定会伸手可及，而我们也希望“嘿Siri”在整个房间中工作。我们专门以各种语言在各种条件下（例如，在厨房（近处和远处），汽车，卧室和餐厅中）用每种语言的母语进行录音。

我们使用“负”数据测试错误激活（和错误唤醒）。数据代表来自各种来源的数千小时的录音，包括播客和多种语言对Siri的非“ Hey Siri”输入，既可以表示背景声音（尤其是语音），也可以表示用户可能对另一个人说的短语人。我们需要大量数据，因为我们正试图估计低至每周一次的误报率。 （如果在否定数据中出现了目标短语，则将其标记为这样，这样我们就不会将对它们的响应计为错误。）  
我们使用“负”数据测试错误激活（和错误唤醒）。数据代表来自各种来源的数千小时的录音，包括播客和多种语言对Siri的非“ Hey Siri”输入，既可以表示背景声音（尤其是语音），也可以表示用户可能对另一个人说的短语人。我们需要大量数据，因为我们正试图估计低至每周一次的误报率。 （如果在否定数据中出现了目标短语，则将其标记为这样，这样我们就不会将对它们的响应计为错误。）  
图6.检测器精度。小型和大型DNN的检测阈值之间的权衡  
调优很大程度上取决于决定使用什么阈值。在图6中，较大模型的较低权衡曲线上的两个点显示了可能的正常和第二次机会阈值。较小（首遍）模型的工作点将在右侧。这些曲线仅适用于检测器的两个阶段，不包括个性化阶段或后续检查。

图6.检测器精度。小型和大型DNN的检测阈值之间的权衡

[](https://machinelearning.apple.com/images/journals/hey-siri/GraphFRR-FAR-1.png)

调优很大程度上取决于决定使用什么阈值。在图6中，较大模型的较低权衡曲线上的两个点显示了可能的正常和第二次机会阈值。较小（首遍）模型的工作点将在右侧。这些曲线仅适用于检测器的两个阶段，不包括个性化阶段或后续检查。

尽管我们相信看起来在测试集上表现更好的模型可能确实更好，但是将离线测试结果转换为对用户体验的有用预测是非常困难的。因此，除了前面描述的离线测量结果之外，我们还估计了误报警率（当Siri打开而用户未说“嘿Siri”时）和冒名顶替者接受率（当Siri打开时，除训练探测器的用户以外的其他人时）通过在最新的iOS设备和Apple Watch上从生产数据中抽样，每周说“嘿Siri”。这不会给我们拒绝率（当系统无法响应有效的“ Hey Siri”时），但是我们可以根据仅在有效阈值之上的激活比例以及仅次于阈值事件的采样来估计拒绝率由开发人员携带的设备上。  
尽管我们相信看起来在测试集上表现更好的模型可能确实更好，但是将离线测试结果转换为对用户体验的有用预测是非常困难的。因此，除了前面描述的离线测量结果之外，我们还估计了误报警率（当Siri打开而用户未说“嘿Siri”时）和冒名顶替者接受率（当Siri打开时，除训练探测器的用户以外的其他人时）通过在最新的iOS设备和Apple Watch上从生产数据中抽样，每周说“嘿Siri”。这不会给我们拒绝率（当系统无法响应有效的“ Hey Siri”时），但是我们可以根据仅在有效阈值之上的激活比例以及仅次于阈值事件的采样来估计拒绝率由开发人员携带的设备上。

我们通过使用此处介绍的方法的变体进行培训和测试，不断评估和改进“ Hey Siri”及其支持的模型。我们以多种不同的语言进行培训，并在各种条件下进行测试。

下次您说“嘿Siri”时，您可能会想到使对该短语做出响应的所有操作，但我们希望它“有效”！