**VARIOUS NOTES FOR PAPER AND PRESENTATION**

[**Abstract** 2](#_Toc46998982)

[**1.** **Introduction** 3](#_Toc46998983)

[**2.** **Related work** 4](#_Toc46998984)

[**Processing pipeline** 4](#_Toc46998985)

[**Proposed End-to-End Architecture** 4](#_Toc46998986)

[**Preprocessing** 8](#_Toc46998987)

[**Dataset** 8](#_Toc46998988)

**Abstract**

We explore the application of ~~deep residual learning~~ and ~~dilated convolutions~~ to the keyword spotting task.

The proposed model establishes a new state-of-the-art accuracy of ~~94.1%~~ on Google Speech Commands dataset V2 (for the 20-commands recognition task), while still keeping a small footprint of only ~~202K~~ trainable parameters.

**Index Terms**: human voice, command recognition, deep learning, keyword spotting

1. **Introduction**

The goal of keyword spotting is to detect a relatively small set of predefined keywords in a stream of user utterances, usually in the context of an intelligent agent on a mobile phone or a consumer “smart home” device. Such a capability complements full automatic speech recognition, which is typically performed in the cloud. Because cloud-based interpretation of speech input requires transferring audio recordings from the user’s device, there are significant privacy implications. Therefore, on-device keyword spotting has two main uses: First, recognition of common commands such as “on” and “off” as well as other frequent words such as “yes” and “no” can be accomplished directly on the user’s device, thereby sidestepping any potential privacy concerns. Second, keyword spotting can be used to detect “command triggers” such as “hey Siri”, which provide explicit cues for interactions directed at the device. It is additionally desirable that such models have a small footprint (for example, measured in the number of model parameters) so they can be deployed on low power and performance-limited devices.

In recent years, neural networks have been shown to provide effective solutions to the small-footprint keyword spotting problem. Research typically focuses on a tradeoff between achieving high detection accuracy and having a small footprint. Compact models are usually variants derived from a full model that sacrifice accuracy for a smaller model footprint, often via some form of sparsification.

In this work, we focus on convolutional neural networks (CNNs), a class of models that has been successfully applied to small-footprint keyword spotting in recent years. In particular, we explore the use of depthwise separable convolutions and 1D CNN.

To summarize in this paper we will:

* Try different audio features (raw data, 80 Mels, 40 MFCC and 40 MFCC with deltas and delta-deltas)
* train different type of neural networks (1D CNN on raw data and DSConv CNN on features)
* Comparing the speed of prediction and the number of parameters of the models

The report is structured as follows. In Section II we present the current state of the art in the speech recognition field, in Section III we show our approach in order to tackle the problem and in Section IV we explain the preprocessing techniques used. In Section V we describe the various architectures we used, in Section VI we report their results and in Section VII we make some extra considerations on them and on some possible developments and future improvements.

1. **Related work**

The first system similar to a modern ASR was built in the 1952 by researchers at Bell laboratories and was able to recognize numerical digits from speech using formants of the input audio. These are a concentration of the acoustic energy around a particular frequency in the input file wave. For the next thirty years, various researchers developed devices capable of recognize vowels and consonants using different types of features like phonemes and keep taking incremental steps forward, until the introduction, in the mid 1980s of the Hidden Markov Models (HMM). This approach represented a significant shift from simple pattern recognition methods, based on templates and a spectral distance measure, to a statistical method for speech processing

*L. R. Rabiner, “A tutorial on hidden markov models and selected applications in speech recognition,” Proceedings of the IEEE, vol. 77, pp. 257–286, Feb 1989.*

and was possible due to the incredible advances in the computer computational power during these years.

But in recent times, the HMMs were challenged by the introduction of Deep Learning and several architecture that works well with these type of problems like Convolutional Neural Networks (CNN) due to their use of weight-sharing and the convolution operation, which is shift-invariant in the data representation domain, and Recurrent Neural Networks (RNN) because of their ability to store information.

Our 3 neural networks are inspired by [7], [8] and [9]. In [7] the authors apply 1D CNN on raw signal for environment sound classification, we slightly modified the structure to adapt it to our key word spotting task. In [8] the….

As in [8] our first CNN presents a first layer that scans the input allowing the kernel to move along the timedimension (x-axis), while the other dimension of the kernel covers the entire coefficients dimension (y-axis). The second neural network instead has a number of convolutional layers fixed but each of them covers only a smaller portion of the input in both dimensions, such as in [7]. The development of recurrent neural networks and of attentions models increased performance on multiple tasks [9] [10], especially those related to long sequence to sequence models.

These models are extremely powerful ways to understand what parts of the input are being used by the neural network to predict outputs. In the case of acoustic models, Connectionist Temporal Classification loss shows good performance in English and Mandarin speech to text tasks [11]. Results using raw waveform without any Fourier analysis have also been investigated [12].

**Processing pipeline**

……

<https://www.researchgate.net/publication/332553888_End-to-End_Environmental_Sound_Classification_using_a_1D_Convolutional_Neural_Network>

**Proposed End-to-End Architecture**

The aim of the proposed end-to-end architecture is to handle audio signals of variable lengths, learning directly from the audio signal, a discriminative representation that achieves a good classiﬁcation performance on diﬀerent speech commands.

2.1. Variable Audio Length

One of the challenges of using 1D CNNs in audio processing is that the length of the input sample must be ﬁxed but the sound captured from the environment may have various duration. Therefore, it is necessary to adapt a CNN to be used with audio signals of diﬀerent lengths. Moreover, a CNN must be used for continuous prediction of input audio signals of environmental sounds. One way to circumvent this constraint imposed by the CNN input layer is to split the audio signal into several frames of ﬁxed length using a sliding window of 1 second width.

Several other conﬁgurations can also be derived from subtle modiﬁcations of the base model (shown in Figure 2) to adapt it to shorter or longer audio inputs, as shown in Table 1. This implies modifying the number of convolutional layers as well as the number and the dimension of ﬁlters and the stride. However, for long contiguous audio recordings, instead of increasing the input dimension of the network, which also implies increasing the number of parameters, and consequently its complexity, it is preferable to split the audio waveform into shorter frames.

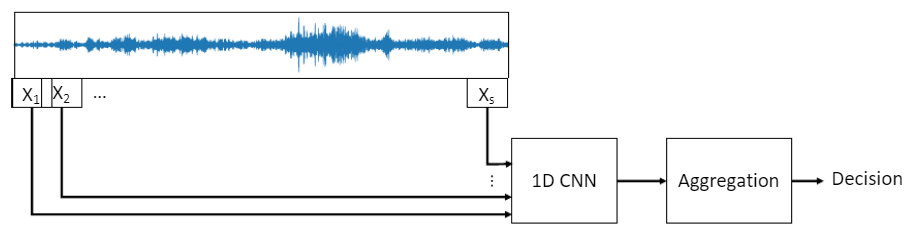
In this way, we keep the network compact and it can process audio waveforms of any length.

In the case of multiclass classiﬁcation, the number of neurons of the output layer is the number of classes. Using softmax as the activation function for the output layer, each output neuron indicates the membership degree of the input samples for each class. During the training process, the parameters of the network are adjusted according to the back-propagated classiﬁcation error and the parameters of the network are optimized to minimize an appropriate loss function (Goodfellow et al., 2016).

**Aggregation of Audio Frames**

In the case where the input audio waveform X is split into S frames denoted asX1, X2, . . . , XS, during the classiﬁcation we need to aggregate the CNN predictions to come up to a decision on X, as illustrated in Figure 4. For such an aim, diﬀerent fusion rules can be used to reach a ﬁnal decision, such as the majority vote or the sum rule, which are denoted in Equations 5 and 6 respectively.

When there are K classes, we generate K values and them for an audio input, we choose the class with the maximum yi value.



It is to computational expensive to use the model for each frame. So we can use a VAD (Voice Activity Detection) or Silence filter to let the model process just the frames that contains voice.

This component must be very compact, low computational expensive and very fast, like just ignoring anything under 40db, it can be a piece of software or even a little piece hardware like some home device are using.

the proposed architecture shown in Figure 2 is made of four convolutional layers, possibly interlaced with max pooling layers, followed by two fully connected layers and an output layer.

input an array of 16,000 dimensions, which represents 1-second of audio sampled at 16kHz.

The proposed 1D CNN has large receptive fields in the first convolutional layers since it is assumed that the first layers should have a more global view of the audio signal.

The output of the last pooling layer for all feature maps is flattened and used as input to a fully connected layer.

In order to reduce the over-fitting, batch normalization is applied after the activation function of each convolution layer.

after the last pooling layer, there are two fully connected layers with 128 and 64 neurons respectively on which a drop-out is applied with a probability of 0.25 for both layers.

By the use of the architecture shown in Figure 2, it is possible to omit a signal processing module because the network is powerful enough to extract relevant low-level and high-level information from the audio waveform.

**Separable convolutions**

Separable convolution uses less parameters, less memory and less computations than regular convolutional layers, and in general they even perform better.

There are two types of Separable convolution: Spatial and depthwise.

Spatial Separable Convolutions divides a kernel into two, smaller kernels. E.g. divide a 3x3 kernel into a 3x1 and 1x3 kernel. So, instead of doing one convolution with 9 multiplications, we do two convolutions with 3 multiplications each (6 in total) to achieve the same effect. With less multiplications, computational complexity goes down, and the network is able to run and train faster.

The problem with Spatial Separable Convolutions is that not all kernels can be “separated” into two, smaller kernels. So during training, since of all the possible kernels the network could adopt, it can only end up using one of the tiny portion that can be separated into two smaller kernels.

Depthwise Separable Convolutions uses kernels that cannot be “factored” into two smaller kernels.

It deals not just with the spatial dimensions, but also with the depth dimension (number of channels).

Splits a kernel into 2 separate kernels that do two convolutions:

* the depthwise convolution, which result will be the input for:
* the pointwise (1x1) convolution.

Depthwise Convolution: Each n×n×1 kernel iterates 1 channel of the input image

**Datasets**:

* 10 cmd
* 20 cmd + unknown

**Preprocessing**:

* No. Directly train on the 16000-element vector
* Mel spectrogram with 80mels
* 40 MFCC
* 40MFCC +delta +deltadelta
* Solo 12 MFCC????? Non si puo fare DSConv

**Architectures:**

* Raw 1D CNN
* DSConv
* ATTNN

**Metrics**:

* Accuracy (in a problem where there is a large class imbalance, a model can predict the value of the majority class for all predictions and achieve a high classification accuracy. So, further performance measures are needed such as F1 score and Brier score, but since in our dataset the classes are balanced we can still use Accuracy)
* Prediction speed (ms)

Comparison:

* DEEP RESIDUAL LEARNING FOR SMALL-FOOTPRINT KEYWORD SPOTTING

**Preprocessing**

The Mel frequency cepstral coefficients (MFCCs) of a signal are a small set of features (usually about 10–20) which concisely describe the overall shape of a spectral envelope. It models the characteristics of the human voice.

**Dataset**

The core words are "Yes", "No", "Up", "Down", "Left", "Right", "On", "Off", "Stop", "Go", "Zero", "One", "Two", "Three", "Four", "Five", "Six", "Seven", "Eight", and "Nine".

To help distinguish unrecognized words, there are also ten auxiliary words, which most speakers only said once.

These include "Bed", "Bird", "Cat", "Dog", "Happy", "House", "Marvin", "Sheila", "Tree", and "Wow".