

# TensorFlow Speech Recognition Challenge

PETRA BRČIĆ  
Applied Mathematics  
petrabrcic94@gmail.com

IVAN ČEH  
Computer Science  
ivan.ceh1234@gmail.com

SANDRO LOVNIČKI  
Computer Science  
lovnicki.sandro@gmail.com

June 26, 2018

**Abstract:** We are witnessing the significant development of the blooming relationship between science and technology. Humanity is striving on making everyday life easier and more controllable by the integration of the intelligent machines. We want to control those machine easier, such as by spoken commands. This is where this project comes in and answers that question.

**Keywords:** tensorflow, speech recognition, audio recognition, feature extraction, MFCC.

## CONTENTS

<b>I</b>	<b>Introduction</b>	<b>2</b>
<b>II</b>	<b>Data</b>	<b>2</b>
i	Preprocessing . . . . .	2
<b>III</b>	<b>Machine Learning</b>	<b>3</b>
i	Convolutional Neural Network . . . . .	3
ii	Results . . . . .	3
<b>IV</b>	<b>Different Approaches</b>	<b>3</b>

## I. INTRODUCTION

...for which our code can be found at our GitHub repository<sup>1</sup>.

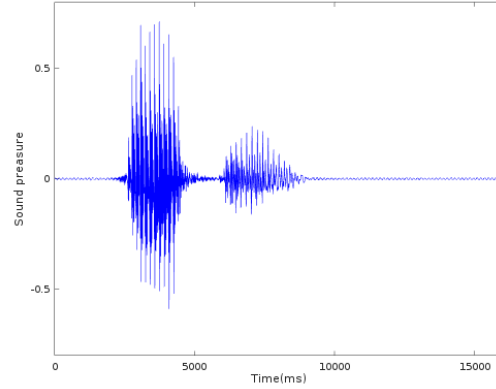
## II. DATA

This project was given by Kaggle as Tensorflow challenge, where the task is to correctly classify the audio recording. Dataset was given by Tensorflow as one second audio clips of different words spoken by different people. Dataset can automatically be downloaded when starting the train part of machine learning solution to the problem. The goal is to have a model that tries to classify a one second audio clip as either silence, an unknown word, "yes", "no", "up", "down", "left", "right", "on", "off", "stop" and "go", even though more words can be found in the dataset and included in the training process. Each of the words is separated into files named as the recordings of the word they present. In order to make these clips as realistic as they would be in real life, the background noise is added in preprocess part of the code. There is a file of few background noises within the dataset that is downloaded. Each audio file is encoded with the id of the person who recorded the word, followed by nohash and a number from 0, specifying the time the same person recorded the word. Apart from the audio files that take up the majority of the data file, testing and validation lists can be found too. Each of those text file contains a list of the audio clips so that dataset can be partitioned in training, validation and test set.

### i. Preprocessing

In order to feed our neural network with the data, we first need to process it. Neural network is fed by the image, so we need to transform the audio file to a spectrogram. Spectrogram is a visual representation of the spectrum of frequencies of sound or other signal as they vary in time. Spectrograms can be generated by an optical spectrometer, a bank of

band-pass filters or by Fourier transform. First, let us observe what the data looks like after reading the audio file.



**Figure 1:** Waveform for audio 'happy'

Creating a spectrogram using FFT is a digital process. Digitally sampled data (which ours is), in time domain, is broken up into chunks, which usually overlap, and Fourier transformed to calculate the magnitude of the frequency spectrum for each chunk. Each chunk then corresponds to a vertical line in the image; a measurement of magnitude versus frequency for a specific moment in time (the midpoint of the chunk). These spectrums or time plots are then laid side by side to form the image or a three-dimensional surface, or slightly overlapped in various ways, i.e. windowing. This process essentially corresponds to computing the squared magnitude of the short-time Fourier transform (STFT) of the signal  $x(t)$  - that is, for a window  $\omega$ ,

$$\text{spectrogram}(t, \omega) = |\text{STFT}(t, \omega)|^2,$$

where

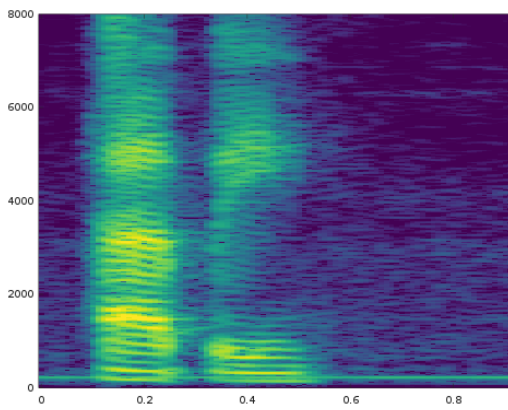
$$\begin{aligned} \text{STFT}\{x[n]\}(m, \omega) &\equiv X(m, \omega) \\ &= \sum_{n=-\infty}^{\infty} x[n] \omega[n - m] e^{-j\omega n}. \end{aligned}$$

The STFT is performed on a computer so it uses the FFT (Fast Fourier Transform), so all the variables are discrete and quantized.

After running our code in octave to generate

<sup>1</sup><https://github.com/Qkvad/SpeechRecognition>

spectrogram yield the figure 2



**Figure 2:** *Spectrogram for audio 'happy'*

2

### III. MACHINE LEARNING

#### i. Convolutional Neural Network

#### ii. Results

### IV. DIFFERENT APPROACHES

#### REFERENCES

- [1] Misha E. Kilmer, Karen Braman, Ning Hao. Third Order Tensors as Operators on Matrices: A Theoretical and Computational Framework with Applications in Imaging
- [2] Misha E. Kilmer, Carla D. Martin. Factorization Strategies for Third-order Tensors