Master en Ciencia de Datos

Escuela Politécnica Superior

Universidad Autónoma de Madrid

Aprendizaje Profundo para Procesamiento de Señales de Audio – APPSA

Práctica 3 – Reconocimiento automático de voz

Guillermo Hoyo Bravo

25 – Abril – 2022

2. Preparando el entorno

El corpus TIMIT es muy especial. Normalmente no se dispone de ningún alineamiento temporal y también son muy poco habituales las transcripciones fonéticas. En corpora más convencionales sólo se dispone de los ficheros .wav y .txt, ¡o incluso sólo los ficheros .wav

PREGUNTAS:

▪ Comprueba el contenido de los ficheros .txt, .wrd y .phn escuchando el audio (.wav) con algunos ejemplos. Puedes hacerlo modificando las partes correspondientes del código del cuaderno de Google Colab. Comenta cómo lo has hecho y lo que has observado.

Comandos Del Cuaderno:

Transcripción Ortográfica Completa .TXT

!cat /content/timit/TIMIT/TRAIN/DR1/FCJF0/SX397.TXT

Resultado: `0 39220 Tim takes Sheila to see movies twice a week.`

Transcripción alineada a nivel de palabra .WRD

!cat /content/timit/TIMIT/TRAIN/DR1/FCJF0/SX397.WRD

Resultado:

2260 5265 she

5265 8920 had

8300 10440 your

…

Transcripción Alineada del .PNH

!cat /content/timit/TIMIT/TRAIN/DR1/FCJF0/SX397.PHN

Resultado:

0 2240 h#

2240 2940 t

2940 4469 ih

4469 5540 m

…

Representación de Audio .WAV

!mkdir -p /content/tmpdata

!sox /content/timit/TIMIT/TRAIN/DR1/FCJF0/SX397.WAV /content/tmpdata/SX397.wav

```
import wave
obj = wave.open('/content/tmpdata/SX397.wav','r')
print( "Number of channels",obj.getnchannels())
print ( "Sample width",obj.getsampwidth())
print ( "Frame rate.",obj.getframerate())
print ("Number of frames",obj.getnframes())
obj.close()
from scipy.io import wavfile
import matplotlib.pyplot as pyplot
import IPython
samplerate, data = wavfile.read('/content/tmpdata/SX397.wav')
#Plot file
pyplot.plot(data)
pyplot.show()
# Play wave (May not workd depending on browser, but you can download and play the file)
IPython.display.Audio('/content/tmpdata/SX397.wav')
```

Resultado:

Figura 1.1. Representación de fichero WAV

Explicación de cada Comando:

Todos los comandos, excepto el de representación del wav, son !cat, que sirve para leer, editar o crear ficheros de texto. En este caso leemos y representamos diferentes ficheros. Los tres ficheros que vamos a ver tienen la siguiente estructura (tiempo inicio, tiempo final, contenido)

Transcripción Ortográfica Completa TXT:

El momento de inicio del audio: 0

La finalización del audio: 39220

La propia frase: `Tim takes Sheila to see movies twice a week.`

Transcripción alineada a nivel de palabra WRD:

El momento de inicio del audio: 2260

La finalización del audio: 5265

La propia frase: she

Transcripción Alineada del PNH:

El momento de inicio del audio: 0

La finalización del audio: 2240

La propia frase: h# (Incluye Silencios h#)

Representación de Áudio:

Number of channels: 1 for mono, 2 for stereo

Sample width: Sample width in bytes

Frame rate: Sampling frequency

Number of frames: Number of audio frames

Plot: **X-axis represent the Number of frames** (divided by the size of the signal the seconds will be represented). At the **Y-axis the Amplitude is represented**

Por temas de amplitud de muestran el resto de los ejemplos de comandos en el **Anexo A**, al final de las preguntas, ya que se entiende que la recién explicación es suficiente. Estos comandos simplemente cambian rutas o direcciones de estor archivos, mostrando sus respectivos resultados de frases y fonemas.

▪ Explora la documentación del corpus TIMIT y trata de encontrar otra información asociada a los hablantes. ¿Puedes imaginar otras aplicaciones de este corpus más allá del entrenamiento y evaluación de sistemas de reconocimiento de voz?

El corpus TIMIT ha sido creado para proveer información asociada al habla para la evaluación de conocimiento acústico-fonético y para el desarrollo y evaluación de sistemas de reconocimiento automático de habla. El corpus contiene 6300 Frases, 10 por cada locutor, lo que hacen un total de 630 locutores, de 8 diferentes dialectos.

La Información asociada al hablante que se ha encontrado es la referente en el fichero **spkrinfo.txt** o en la url https://catalog.ldc.upenn.edu/docs/LDC93S1/SPKRINFO.TXT :

Resumiéndolo, se encuentra una tabla para cada locutor con la siguiente información:

- Sexo
- Dialecto (New England, Northern, ...)
- Uso de los datos (Train/Test)
- Dia de Grabación
- Cumpleaños

- Altura
- Raza (español-americano, Oriental...)
- Nivel de educación (Instituto, Bachillerato...)
- Comentarios

Ideas de otras aplicaciones para este corpus pueden ser:

- Reconocimiento de sentimiento según la cadencia: Esta demostrado científicamente que, si alguien se encuentra feliz, este habla más rápido y con más variaciones en su voz. Mientras que si una persona esta triste o deprimida, la cadencia es menor, igual que las variaciones de su voz.
- Se podría analizar, por ejemplo, la diferencia de los fonemas generados por las personas con diferentes niveles de educación. Algo muy interesante ya que seguro se encuentran patrones en la pronunciación, y con suerte en la cadencia, aunque también es posible que no y puede ser precipitado asumirlo. De igual modo, también se podría realizar el estudio de las mismas regiones, por razas.
- Otra idea podría ser la de buscar patrones de voz entre gente de misma estatura. Tal vez usuarios que miden igual tengan algún rasgo similar.

3. Uso de ESPnet en Bash

En esta sección usaremos ESPnet en Bash para entrenar y evaluar un sistema STT usando redes neuronales end-to-end y el corpus TIMIT.

3.1. Estructura de directorios y recetas estilo KALDI

Explora la estructura de directorios de las recetas de ESPNet (que siguen la estructura de directorios de KALDI) y contesta las siguientes cuestiones. PREGUNTAS:

▪ ¿Qué fichero o ficheros deberías cambiar para modificar cada una de las siguientes configuraciones del reconocedor de voz?

o La estructura de las DNNs empleadas.

Modificar la arquitectura de la red Neuronal: espnet/egs/timit/asr1/conf/train.yaml

o El mapeo secuencia a secuencia empleado.

Modificar parámetros de decodificación (reconocimiento): espnet/egs/timit /asr1/run.sh

El parámetro que estamos buscando es: trans_type.

▪ Comparando los resultados obtenidos con reconocimiento de fonemas y de caracteres con TIMIT, ¿qué modalidad consideras que es más precisa?

Los resultados se pueden observar guardados en el fichero Results.md. Este fichero contiene tablas con la siguiente información. N.º de ficheros, N.º de palabras, % tokens correctos, % Tokens sustituidos, % Tokens Borrados, % Tokens Insertados, Tasa de error en %. Todo ello según se evalúan Palabras, fonemas o caracteres, con las diferentes estructuras utilizadas.

Para Fonemas

	VGGBGRU – GRU	**BGRUP – LSTM**	**BGRUP - GRU**
Correct	82.1	82.7	82.5
Error	21.4	20.2	20.5

Para Caracteres

	VGGBGRU – LSTM	**VGGBGRUP – GRU**	**BGRUP - LSTM**
Correct	69,6	72,0	72,8
Error	38,2	34,0	33,7

Con estas tablas es fácil identificar que estructura y método funciona mejor. Este es el de fonemas, y se debe a que los fonemas describen a la perfección la plenitud de una estructura, a diferencia de los caracteres, una “c” en español puede representar el sonido “k” si viene seguido de una “a”, o el sonido “c” si viene seguido de una “e”. Esto es difuso para el modelo, lo que hace que funcione peor.

Sería mucho mejor comparar los métodos por fonemas y por palabras, ya que ambos describen a la perfección las estructuras, facilitando trabajo al modelo.

Aun así, en ambos casos (fonemas/caracteres), funciona mejor el modo BGRUP – LSTM.

3.2. Preparación de los datos (Etapas 0 - 2)

Ejecuta las etapas de preparación de datos y contesta las siguientes cuestiones.

PREGUNTAS:

▪ ¿Cómo se denominan los tres subconjuntos de TIMIT que se usan? ¿Cuántas frases tiene cada uno de ellos?

train_set=train_nodev

train_dev=train_dev

recog_set="train_dev test"

▪ ¿Qué son las características denominadas “FBANK”?

FBANK or Filter Bank is a system that divides the input signal into a set of analysis signals, each of which corresponds to a diffenter refion in the sprectrum.

The analysis filter bank divides an input signal to different subbands with different frequency spectra

Las características Filterbank se extraen de una señal de audio como input. Esta, pasa por el banko de filtros que separa la señal de entrada en diferentes componentes, cada uno de los cuales lleva consigo una frecuencia o una banda de frequencias de la señal original, extrayendo caracteristicas. Es similar a MFCC.

▪ ¿Qué significan las siglas “CMVN” que aparecen en los logs de cálculo de las características?

Cepstral mean and variance normalization

...

3.3. Entrenamiento de las redes neuronales (Etapas 3 - 4)

Ejecuta las etapas de entrenamiento de las redes neuronales (¡tardará aproximadamente 20 minutos!). Mientras se entrenan las redes, trata de contestar a las siguientes cuestiones:

PREGUNTAS:

▪ Analiza el fichero train.yalm y trata de identificar el parámetro que deberías modificar para cambiar:

- o El número de capas en el encoder.

  elayers: 5

- El tipo de encoder.

  etype: bgrup

- El número de unidades en cada capa del encoder.

  eunits: 320

- El tipo de atención en el decoder.

  atype: coverage_location

- El número de unidades en el decoder.

  dunits: 300

Se puede usar Tensorboard para analizar gráficamente la evolución del entrenamiento (aunque necesitarás que el entrenamiento haya terminado completamente antes). Echa un vistazo a los distintos gráficos de Tensorboard y contesta a las siguientes cuestiones:

PREGUNTAS:

▪ ¿Puedes identificar los parámetros correspondientes a CTC y atención?

▪ Comprueba la evolución de los diferentes parámetros y compara la evolución en entrenamiento y validación. ¿Consideras que el entrenamiento ha funcionado correctamente?

Un método alternativo para comprobar la evolución del entrenamiento es emplear los gráficos que genera ESPnet durante el proceso de entrenamiento.

PREGUNTAS:

▪ ¿Puede observar la diferencia en la evolución de las funciones de coste asociadas a CTC y atención? ¿Puede explicarlo?

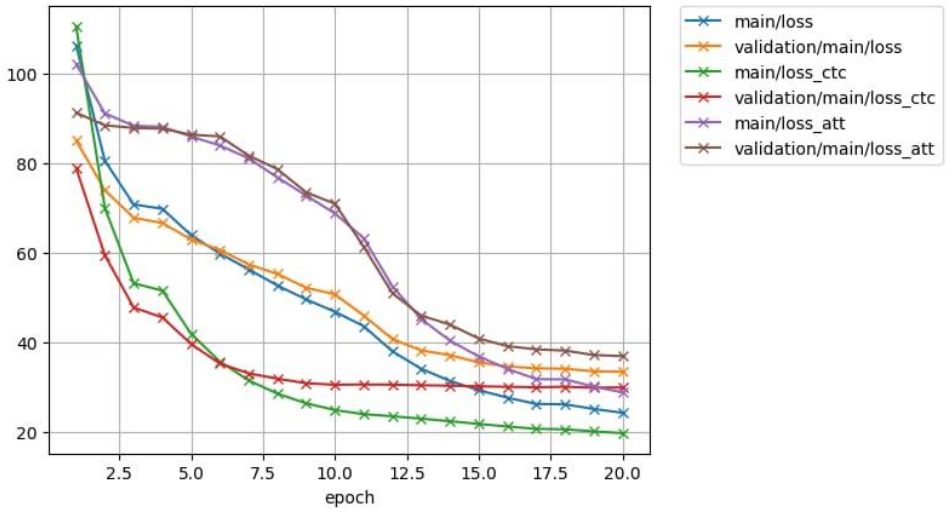

Figura 2.1. Loss Functions

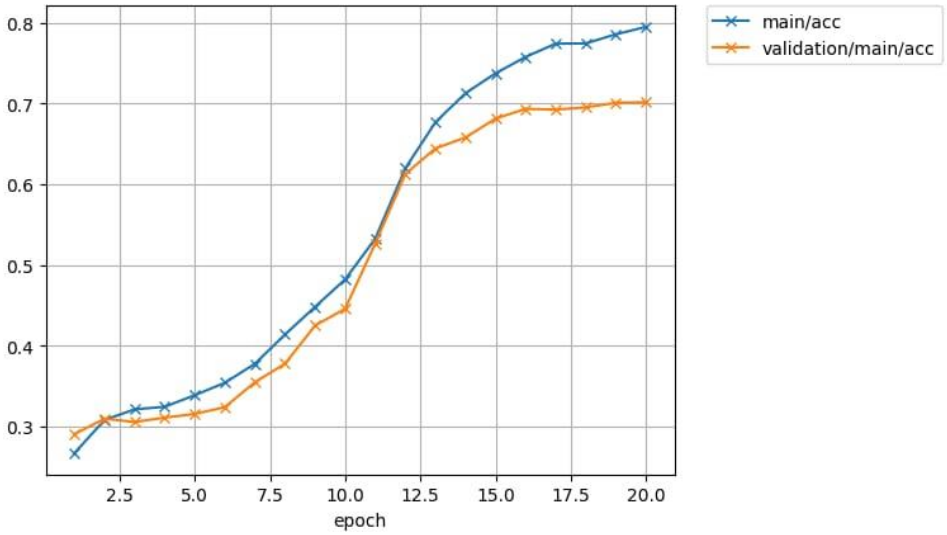

Figura 2.2. Validación Accuracy

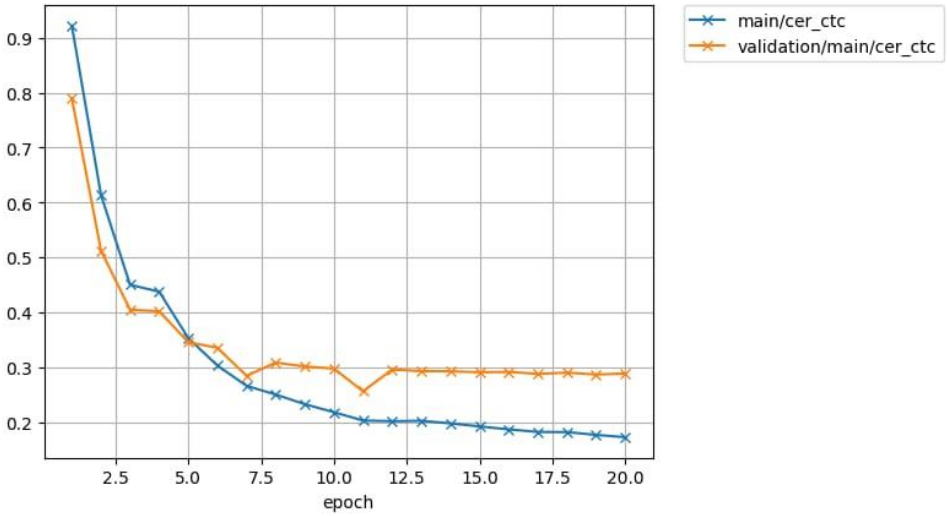

Figura 2.3. Validación PER

▪ ¿Cuál es (aproximadamente) el PER final en entrenamiento y validación? ¿Por qué se dan estas diferencias?

El per en train es menor a 0,2, mientras que el per en train es aproximadamente 0,3. Las diferencias se dan porque además de que en la fase de train siempre se suelen obtener mejores resultados, los resultados de la fase de test se mantienen estables mientras que los de train siguen mejorando.

Se puede decir que el modelo no sobreajusta ya que la accuracy no decae, ni el PER empieza a aumentar obteniendo peores resutlados.

3.4. Reconocimiento (decodificación) y evaluación (Etapa 5)

Explora el formato del fichero decode.yaml y contesta a las siguientes cuestiones:

PREGUNTAS:

▪ ¿Qué es el parámetro “beam-size”?

Determina el número de las mejores soluciones parciales se van a evaluar. El parámetro Beam size limita el número de candidatos a coger como input para el decoder. Consiste del argoritmo greedy seach

▪ ¿Qué es el parámetro “ctc-weigth”? Ahora ejecuta la fase de reconocimiento (etapa 5) y contesta las siguientes cuestiones:

Es un parámetro utilizado para cambiar entre los modelos y modo de: atención, CTC, e híbrida.

Ctc-weight = 0.0 es el modelo de atención.

Ctc-weight = 1.0 es el modelo CTC

Cualquier cosa entre medias es el porcentaje de cada uno de los dos modelos para formar el modelo híbrido.

PREGUNTAS:

▪ ¿Por qué aparecen dos resultados distintos? ¿Qué significan? ¿Se comportan como esperabas?

Los resultados vienen de:

exp/train_nodev_pytorch_train/decode_test_decode/results.txt

exp/train_nodev_pytorch_train/decode_train_dev_decode/result.txt

Esto significa que existen unos resultados para la parte de train, y otros para la parte de test. Los resultados son bastante buenos, y si, se comporta como se esperaba, ya que el train es mejor que el test.

3.5. Comprobación de resultados de evaluación

Los resultados se pueden analizar con distinto nivel de detalle. Conteste las siguientes cuestiones:

PREGUNTAS:

▪ ¿Puede calcular manualmente el PER para el alineamiento obtenido con el siguiente comando? ¿Cuál es el PER para este fichero?

!tail -n 7 espnet/egs/timit/asr1/exp/train_nodev_pytorch_train/decode_test_decode/result.txt

El PER o phone error rate se calcula con la siguiente formula:

$$PER = \frac{S+D+I}{S+D+I+C}$$

D = Eliminaciones

S = Sustituciones

I = Inserciones

C = Correctos

De esta manera,

4. Extensiones OPCIONALES

Hay una serie de posibles extensiones opcionales de esta práctica (pero se puede obtener la máxima puntuación sin necesidad de ninguna de estas extensiones). Se recomienda que estas extensiones se consideren única y exclusivamente para la práctica final.

Las primeras tres extensiones opcionales no cambian la tarea, sino sólo el modelo o las características:

• Cambiar la topología de las redes encoder y/o decoder (añadiendo o eliminando capas, haciendo capas más anchas, o más estrechas, en términos de unidades, etc.).

• Cambiar la función de mapeo de secuencia a secuencia (usando sólo CTC, sólo atención, modelos Transformer, etc.).

• Cambiar las características de entrada (aunque esto requiere un buen conocimiento de Kaldi y puede resultar bastante complejo – puedes echar un vistazo en http://kaldi-asr.org).

Cualquiera de las anteriores extensiones opcionales, o cualquier combinación de las mismas, dará lugar a un PER diferente en la misma tarea, por lo que el resultado será directamente comparable con el resultado obtenido en la práctica original.

Las siguientes modificaciones opcionales implican cambiar la tarea, y por tanto los resultados ya no serán comparables con el sistema original propuesto en la práctica. Además, estas tareas requieren un grado de desarrollo bastante superior a las anteriores, por lo que se recomienda abordarlas sólo después de tener resuelta la práctica final de forma más sencilla.

• Cambiar el cuaderno de Google Colab para entrenar y evaluar un STT basado en caracteres, en lugar de basado en fonemas.

• Intentar reconocer ficheros con tu voz.

• Intentar ejecutar una receta distinta (hay varias recetas que se basan en datos abiertos). Deberás escoger un corpus lo suficientemente pequeño como para que el entrenamiento se pueda ejecutar en Google Colab.

• Ejecutar una receta para entrenar el problema inverso, un Text-To-Speech como en este tutorial: https://colab.research.google.com/github/espnet/interspeech2019-tutorial/blob/master/notebooks/interspeech2019_tts/interspeech2019_tts.ipynb

Anexos:

Anexo A: Ejemplos Adicionales de Comandas y ficheros .TXT, .WRD, .PHN, .WAV

Ejemplos con Comandos adicionales:

Diferente Frase – Mismo Locutor

!cat /content/timit/TIMIT/TRAIN/DR1/FCJF0/SX37.TXT

```
Resultado: 0 36250 Critical equipment needs proper maintenance.
```

!cat /content/timit/TIMIT/TRAIN/DR1/FCJF0/SX37.WRD

2260 7895 critical

7895 15190 equipment

13781 18101 needs

…

!cat /content/timit/TIMIT/TRAIN/DR1/FCJF0/SX37.PHN

0 2260 h#

2260 2920 k

2920 3266 r

3266 3960 ih

…

Representación wav:

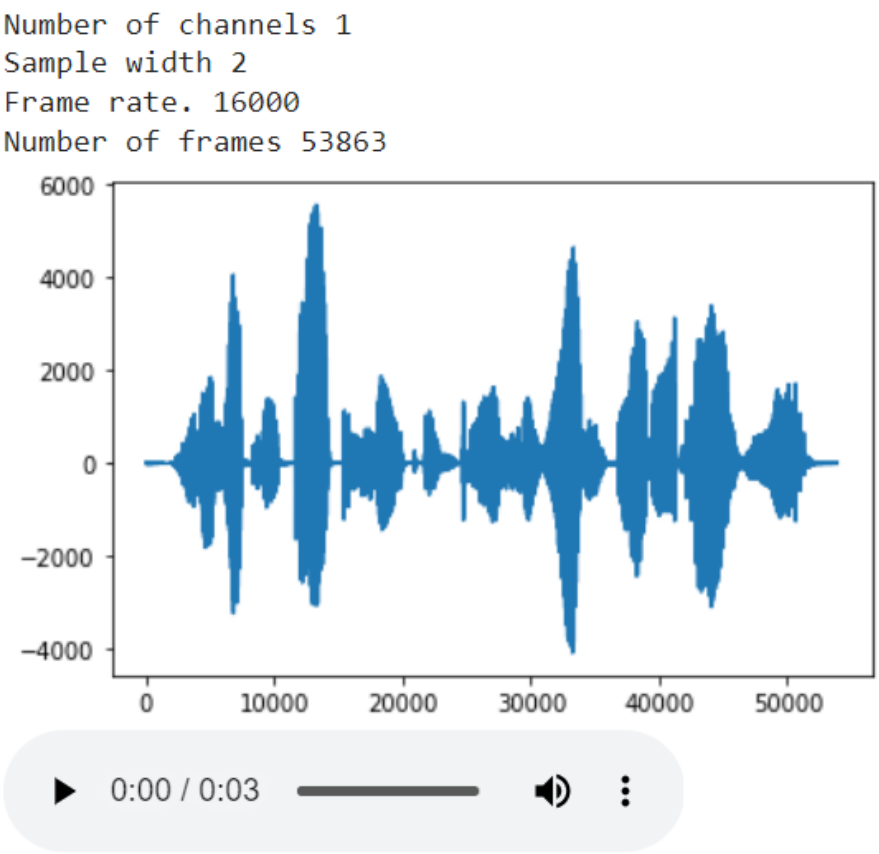

Figura 1.2. Representación de fichero WAV

Diferente Locutor

!cat /content/timit/TIMIT/TRAIN/DR1/FDAW0/SA1.TXT

```
0 53556 She had your dark suit in greasy wash water all year.
```

!cat /content/timit/TIMIT/TRAIN/DR1/FDAW0/SA1.WRD

2230 5353 she

5353 9400 had

9400 10920 your

…

!cat /content/timit/TIMIT/TRAIN/DR1/FDAW0/SA1.PHN

0 2230 h#

2230 4263 sh

4263 5353 iy

5353 6224 hv

...

Representación wav:

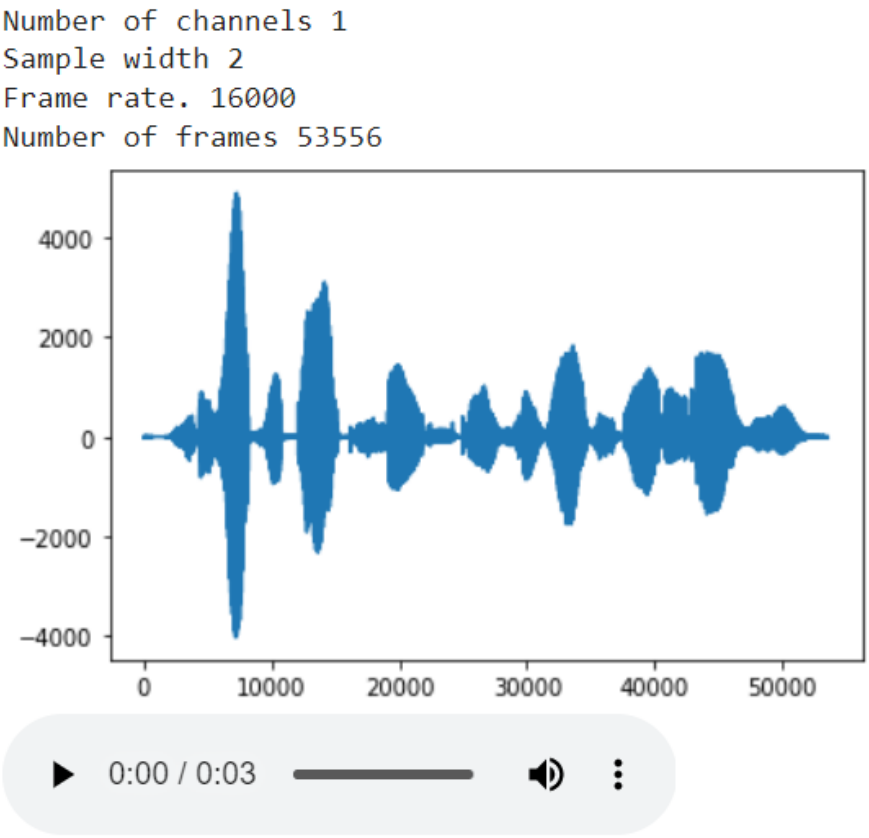

Figura 1.3. Representación de fichero WAV

Diferente Región Díalectica:

!cat /content/timit/TIMIT/TRAIN/DR2/FAEM0/SA1.TXT

0 53863 She had your dark suit in greasy wash water all year.

!cat /content/timit/TIMIT/TRAIN/DR2/FAEM0/SA1.WRD

2260 5265 she

5265 8920 had

8300 10440 your

...

!cat /content/timit/TIMIT/TRAIN/DR2/FAEM0/SA1.PHN

0 2260 h#

2260 4070 sh

4070 5265 iy

5265 6349 hv

...

Representación wav:

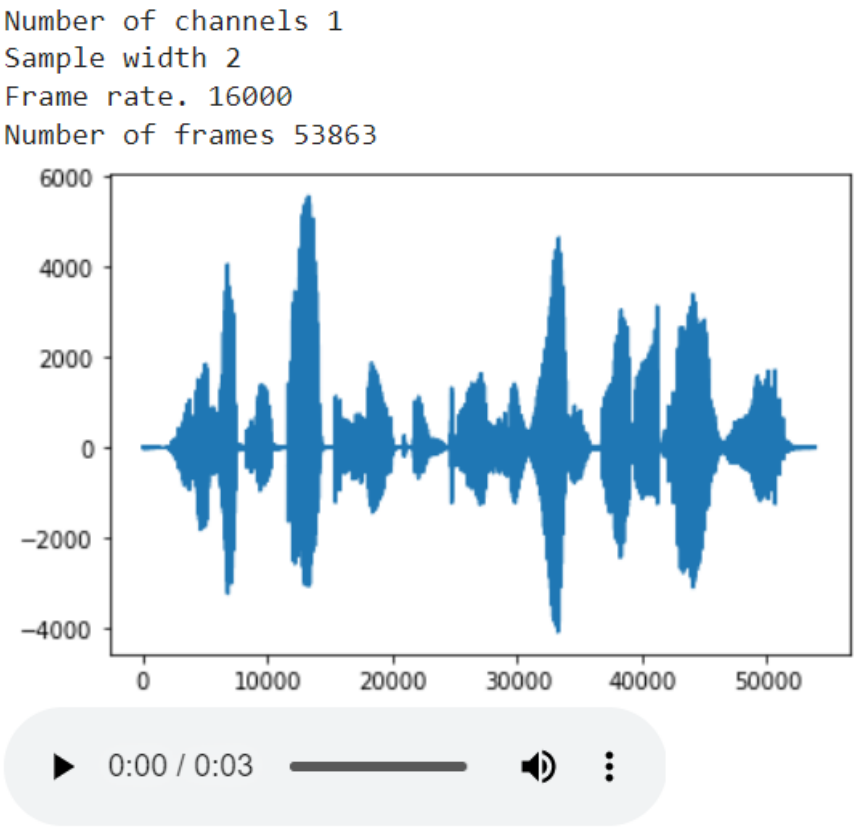

Figura 1.4. Representación de fichero WAV

FIN