DEPARTAMENTO
DE COMPUTACION

Facultad de Ciencias Exactas y Naturales - UBA

Affective computing

Clasificador del habla

Procesamiento de Señales
Segundo cuatrimestre 2020

Integrante	LU	Correo electrónico
Coy, Camila Paula	33/14	camicoy94@gmail.com
Figarola, Lucas Adriel	953/13	lukas12_alfa56@hotmail.com
Junqueras, Juan	804/16	junquerasjuan@gmail.com

Resumen

Affective computing involucra el área de investigación centrada en la implementación de sistemas y mecanismos capaces de identificar, comprender y elaborar los afectos humanos. Es un campo que abarca la informática, la psicología y las ciencias cognitivas. En este proyecto se realizo un clasificador de emociones a partir del habla. Básicamente consiste en realizar una serie de filtrados y muestreos, para después usar un modelo de Machine Learning para clasificar los audios según su emoción.

Facultad de Ciencias Exactas y Naturales
Universidad de Buenos Aires
Ciudad Universitaria - (Pabellón I/Planta Baja)
Intendente Güiraldes 2160 - C1428EGA
Ciudad Autónoma de Buenos Aires - Rep. Argentina
Tel/Fax: (++54 +11) 4576-3300
http://www.exactas.uba.ar

Índice

1. Introducción

‘‘Computers are beginning to acquire the ability to express and recognize affect, and may soon be given the ability to \have emotions." The essential role of emotion in both human cognition and perception, as demonstrated by recent neurological studies, indicates that affective computers should not only provide better performance in assisting humans, but also might enhance computers’ abilities to make decisions.’’ [1]

Al empezar este proyecto no estábamos muy seguros de que proyecto afrontar, pero a todos nos gustaba la idea de abordar el tema de Affective Computing. Por lo que empezamos mirando varios datasets y clasificando los que nos parecían interesantes. Buscando, encontramos un proyecto que usaba Machine Learning para crear un clasificador de emociones a partir de habla. En una primer instancia se nos ocurrió reproducir dicho proyecto, pero posteriormente optamos por basarnos en el mismo, pero usando un dataset distinto. El dataset elegido fue CREMA-D[2], en el cual los datos están clasificados en muchas más categorías, lo que nos brinda mucha más granularidad, permitiendonos plantearnos nuevas preguntas para guiar la investigación.

[1] Affective Computing, R. W. Picard
[2] www.kaggle.com/ejlok1/cremad

2. Materiales y Métodos

El componente más importante de este trabajo es el dataset CREMA-D, un data set que consiste en 7442 archivos audio de entre 1 a 3 segundos, grabados por 91 actores. Estos audios son de 48 hombres y 43 mujeres, con edades entre los 20 y los 74 años, de diferentes etnias y orígenes (Africanos, Americanos, Asiáticos, Caucásicos, Hispanos y otros sin especificar). Los actores grabaron una selección de 12 frases. Las frases fueron grabadas interpretando seis emociones distintas (enojo, asco, miedo, alegría, neutral y tristeza) y cuatro diferentes intensidades emocionales (bajo, medio, alto e inespecificado). El dataset esta conformado por archivos de audio con extension .wav los cuales fueron bajados de: www.kaggle.com/ejlok1/cremad

Como se comentó, el trabajo se realizó utilizando técnicas de Machine Learning. El clasificador se desarrolló con scripts en Python 3 usando la plataforma Google Colab.

El trabajo original en el cual se basó este proyecto puede encontrarsen aquí:
www.kaggle.com/uldisvalainis/audiomodel

En el trabajo usamos MFCC (Mel Frequency Cepstral Coefficients o Coeficientes Cepstrales en las Frecuencias de Mel) para extraer los features de los audios, en otras pablabras, para identificar los componentes de las señales de audio que son útiles para identificar el contenido linguistico (emocional en este caso), descartando otros aspectos no deseados como el ruido de fondo, etc.

Además se utilizo el modulo Keras, que es un framework de alto nivel para el aprendizaje (Deep Learning), escrito en Python. Keras contiene varias implementaciones de los bloques constructivos de las redes neuronales como por ejemplo los layers, funciones objetivo, funciones de activación, optimizadores matemáticos.

3. Desarrollo

3.1. Exploración de Datos

En esta sección se explica cómo se implementó el proyecto. El trabajo se relata de forma cronológica, para que así se puedan ver las idas y venidas que tuvo el proyecto.

En una primer instancia, luego de cargar el dataset se realizó una exploración del mismo, con el objetivo de entender un poco más los datos con los que se realizó el trabajo. Se puede destacar de la exploración que si se toman los audios, se les aplica la Transformada de Fourier, se obtienen las densidades espectrales de potencia de los mismos, a las cuales se les puede aplicar el logaritmo para pasar de amplitud a decibeles, todo esto se puede graficar usando espectrogramas como los que se pueden ver en la figura 1. Lo interesante de estos espectrogramas es que algunos clasificadores de emociones los usan como input a una CNN (convolutional neural networks), este enfoque sería equivalente a pasar las señales de audio a imágenes y hacer renococimiento de imágenes para clasificar.

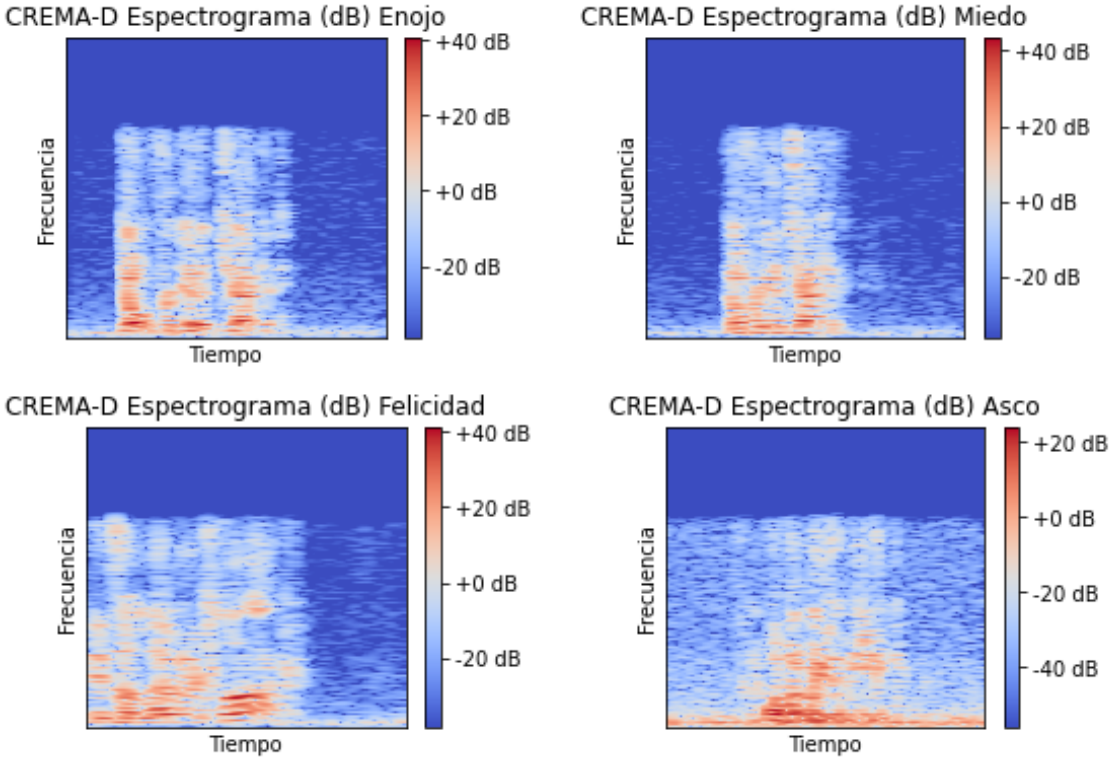

Figura 1: Espectrogramas

3.2. Conversión de los audios

Como se comentó en la sección 2.Metodos y Materiales, en este proyecto los audios originales fueron procesados para obtener los MFCC o features útiles para el reconocimiento de emociones. Puntualmente lo que hace para calcular los MFCC es:

- Dividir el audio en pequeñas muestras, para así asegurar que no se generen muchos cambios en los segmentos a analizar.
- Realizar la transformada de Fourier, para obtener la densidad espectral de potencia.
- Se aplican los filtros de la escala de Mel y el logaritmo a esto, para que se parezcan más a lo que capta el oído humano.
- Aplicar la transformada de coseno discreta, que sirve para comprimir la información en coeficientes.

Para poder hacer el clasificador, primero fue necesario convertir los archivos wav en un array de tuplas de MFCC y la distinción de la emoción a la cual pertenece el audio, para que así pudiéramos usarlos en nuestra red neuronal.

convert():
(newDataBase) ← emotionDicc;
{ item1 (integer) "Enojo" ← 1, item2 (integer) "Tristeza" ← 2, item3 (integer) "Miedo" ← 3,
item4 (integer) "Felicidad" ← 4, item5 (integer) "Neutral" ← 5, }
list_wav ← get_all_files_wav();
for *i* ← 0 **to** *length(list_wav)* **do**
    mfcc ← transform_into_MFCC(list_wav(i));
    emotion ← get_emotion(list_wav(i));
    num_emotion ← emotionDicc(emotion);
    arr ← (mfcc,num_emotion);
    add(list,arr);
end

Algoritmo 1: Pseudocódigo conversión de audio a MFCC

En la figura 2 se puede ver el ploteo de los espectrogramas de cuatro de los audios convertidos a MFCC.

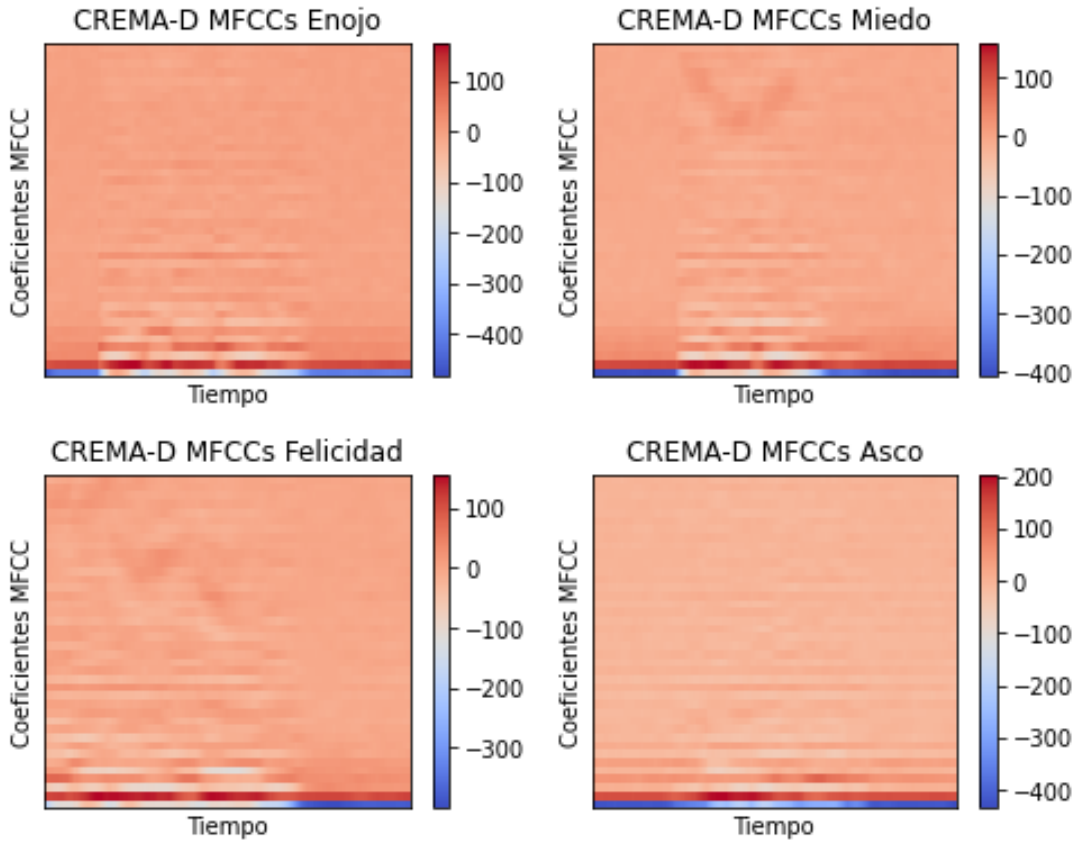

Figura 2: Espectrogramas de los MFCC

Una vez cargados los datos y convertidos los wav a MFCC, iniciamos la construcción del clasificador.

3.3. Construyendo el clasificador

El primer paso (luego de convertir todos los archivos .wav) consistió en dividir el dataset en dos conjuntos: el de entrenamiento y el test. El primero como su nombre lo indica, se utilizó para entrenar al modelo de Machine Learning (más adelante explicaremos como se contstruyó este modelo). El segundo

conjunto (el de testing) se utilizó para evaluar (con distintas métricas) cómo se estaba comportando nuestro modelo.

Posteriormente construímos un red neuronal muy simple: una imput layer, tres capas internas y la output layer. A pesar de ser un modelo de clasificación muy simple, aparentemente serviría para poder clasificar los audios según su emoción.

Una vez construído el modelo, lo entrenamos para que clasifique los audios según 5 categorías de emociones: enojo, tristeza , miedo, felicidad y neutral(o sin emoción).

Durante el desarrollo de este trabajo se entrenó varias veces al modelo corriendo la función fit, ajustando los parámetros batch size y epochs, buscando conseguir los mejores resultados, estos serán detallados en la sección 4

3.4. Alternativas de clasificación

Una de las alternativas de clasificación era que las categorías se distinguiesen no sólo por emoción, sino también por intensidad. De esta manera miedo-alto, sería una categoría distinta a miedo-medio. Cruzando las seis emociones con las cuatro intensidades, el modelo tendría que ser capaz de clasificar las señales de audio en 24 categorías. Por cuestiones implementativas, no se pudo realizar esta vía de experimentación.

Como quizás se haya notado, CREMA-D cuenta con seis categorías de emociones, sin embargo nuestro modelo clasifica en cinco, en particular se ignora la categoría “asco”. Los audios que entran bajo esta categoría no fueron usados durante el entrenamiento ni la evaluación. Las causas por las cuales se ignoró están ligadas a una segunda alternativa de clasificación que exploramos durante el desarrollo del trabajo.

Buscando otras formas de mejorar los resultados del clasificador de emociones (que no sean seguir entrenando al modelo con otros parámetros), decidimos intentar con otro grupo de categorías. La idea era clasificar los audios según la emoción y el género del hablante. La idea detrás de esto era que quizás el modelo confundía algunos estados de ánimo según el género del hablante. Según este enfoque de investigación, tendríamos 12 categorías (dos por cada emoción), pero por razones implementativas, no podíamos hacer funcionar el modelo con seis emociones, había que eliminar una. Evaluando los resultados de la experimentación que clasificaba únicamente con las seis emociones, notamos que “asco” era la emoción que peor clasifcaba, confundiendolá frecuentemente con “neutral” y al escuchar los audios, nuestra percepción subjetiva fue que los audios etiquetados como “asco” sonaban muy parecidos a los “neutral”, por lo que decidimos eliminar la categoría “asco” tanto de los entrenamientos como de las evaluaciones. Pudimos entonces correr el modelo (los resultados serán expuestos y analizados en la sección 4. Resultados).

El objetivo de buscar alternativas de categorías, era mejorar la capacidad de clasificación del modelo. Para saber si esta había mejorado o no, era necesario comparar los resultados del modelo que clasificaba según emoción contra el que clasificaba según emoción y género. Como uno clasificaba según seis emociones y el otro según cinco, para poder como compararlos, se optó por eliminar la emoción que peor era clasificada: “asco”.

Una vez que se decidió eliminar la emoción asco, los modelos eran comparables. Se volvió a correr el clasificador que distinguía exclusivamente según emociones (enojo, asco, miedo, alegría y neutral). La comparación de los resultados de ambos modelos están en la sección 4.Resultados, de este informe.

4. Resultados

En esta sección se analizarán los resultados de los clasificadores. Esto lo haremos principalmente con las métricas de accuracy(que es el porcentaje de aciertos totales del modelo), precision(por cada categoría, es la relación entre los que predijo correctamente y el total de los que predijo), recall(por cada categoría, la relación entre los que predijo correctamente con el total de los que había), f1(es una combinación entre la precision y el recall) y matriz de confusión(una matriz donde cada fila representa la clase real y cada columna representa la predicción, pudiendo observar donde el modelo se confunde).

4.1. Clasificador de emociones

Al terminar el proyecto nuestro clasificador llego a una accuracy de $0,52$. Para entender un poco más cómo clasifica y dónde suele equivocarse, realizamos una matriz de confusión con los resultados de correr el modelo sobre los datos de testeo (habíamos partido el data set en dos: una parte para el entrenamiento y la otra para el testeo).

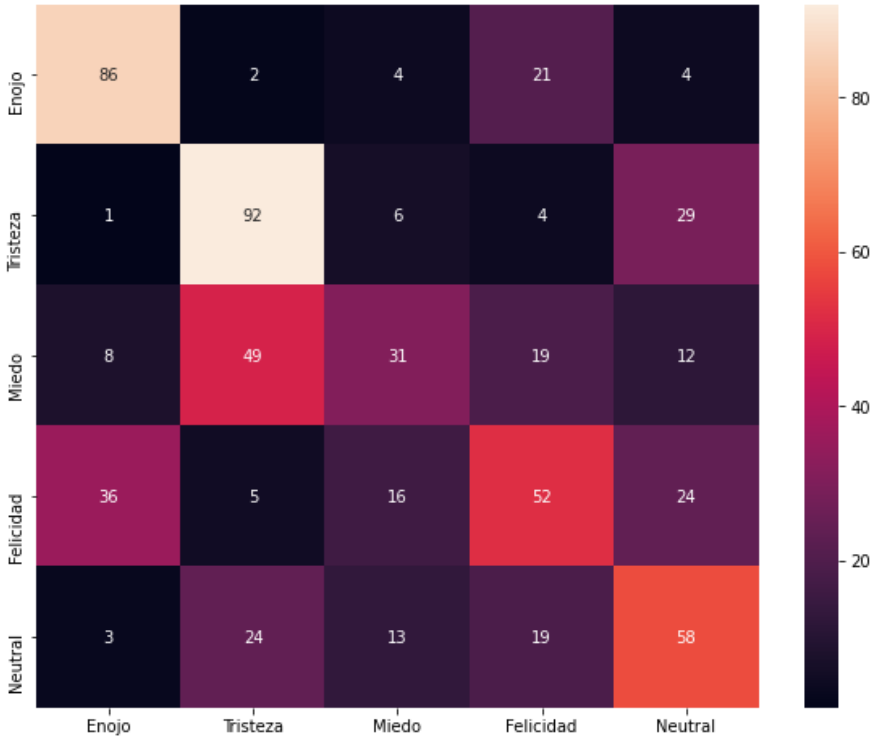

Figura 3: Matriz de Confusión

Como se puede ver en la figura 3, las emociones que mejor clasifica nuestro modelo son la “tristeza” y el “enojo”. Esto también se ve en la figura 4 donde muestra que tanto la precision, recall y f1 son bastante más altos que el del resto de las emociones. Con estos datos, se observa que el clasificador identifica mejor “enojo” que “tristeza”. El miedo es la emoción que más confunde el modelo ya que el recall es muy bajo y con la matriz se ve que lo hace más con la tristeza que con el resto de la emociones. Al escuchar los audios de miedo es difícil hasta para un humano distinguir estas dos emociones.

	precision	recall	f1-score	support
Enojo	0.64	0.74	0.69	117
Tristeza	0.53	0.70	0.61	132
Miedo	0.44	0.26	0.33	119
Felicidad	0.45	0.39	0.42	133
Neutral	0.46	0.50	0.48	117
accuracy			0.52	618
macro avg	0.51	0.52	0.50	618
weighted avg	0.50	0.52	0.50	618

Figura 4: Métricas del modelo

4.2. Clasificador de emociones y género

Otro de los caminos fue generar un clasificador que clasifique según la emoción y según si la voz es femenina o masculina. Aunque creíamos que iba a haber una mejoría esto no paso, ya que el accuracy del modelo bajo a $0,47$. También optamos por mirar la matriz de confusión.

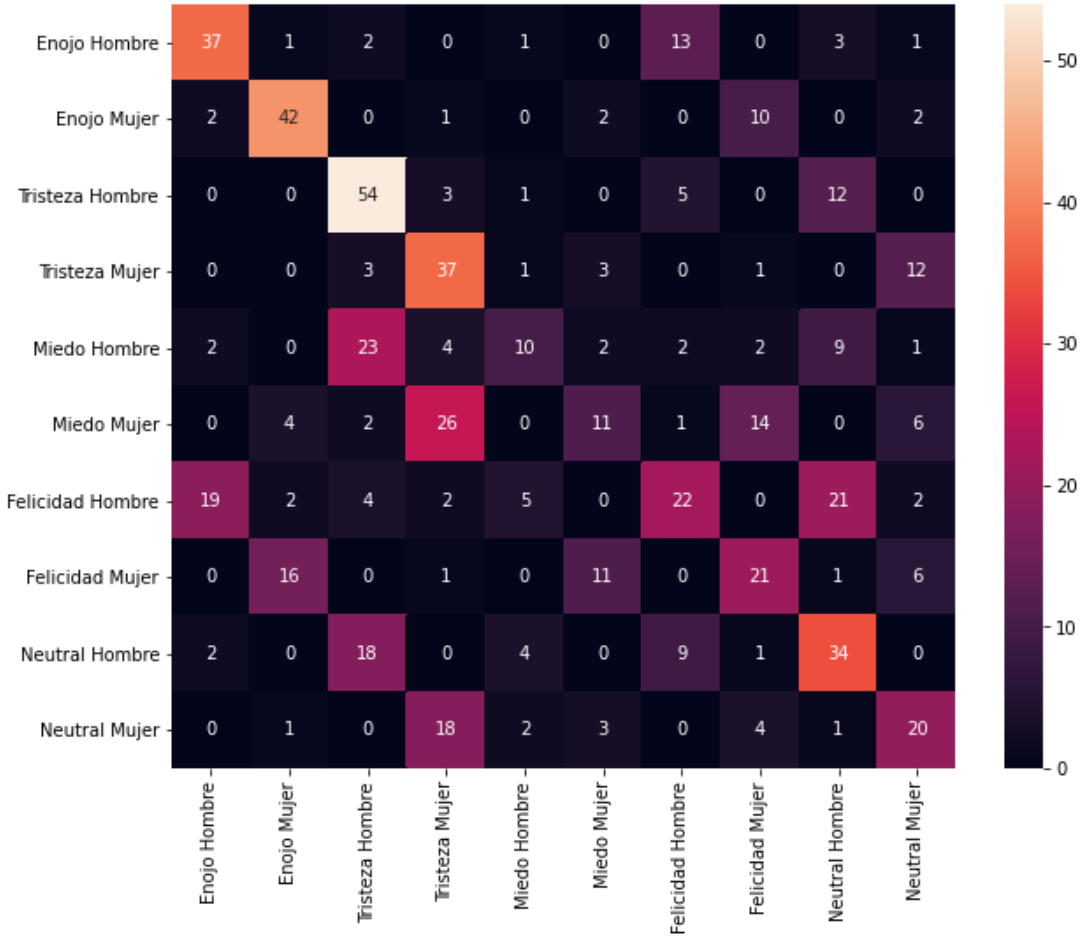

Figura 5: Matriz de Confusión

Con la matriz de la figura 5 pudimos ver que las voces de mujeres y hombre no se confunden mucho entre sí y que diferenciarlos entre género no modifico las confusiones que se generaban entre emociones. Este se ve mejor en la figura 6 ya que, mirando f1, las dos categorías de "enojo" son mejores que las dos de "tristeza" y las dos categorías de miedo siguen siendo las que peores números muestran. La única mejora

que se puede observar es que el recall de la “tristeza” aumento e incluso supero a “enojo” al separar el género

	precision	recall	f1-score	support
Enojo Hombre	0.60	0.64	0.62	58
Enojo Mujer	0.64	0.71	0.67	59
Tristeza Hombre	0.51	0.72	0.60	75
Tristeza Mujer	0.40	0.65	0.50	57
Miedo Hombre	0.42	0.18	0.25	55
Miedo Mujer	0.34	0.17	0.23	64
Felicidad Hombre	0.42	0.29	0.34	77
Felicidad Mujer	0.40	0.38	0.39	56
Neutral Hombre	0.42	0.50	0.46	68
Neutral Mujer	0.40	0.41	0.40	49
accuracy			0.47	618
macro avg	0.45	0.46	0.45	618
weighted avg	0.45	0.47	0.45	618

Figura 6: Métricas del modelo

Para poder comparar los resultados entre los dos modelos generamos una matriz de confusión con el género colapsado(las cuatro casillas que pertenecen a una emoción sumadas en una sola), quedando así la figura 7. Al comparar esta nueva matriz con la de la figura 3, se ve que el segundo modelo es peor que el primero y que las confusiones son parecidas.

Sin embargo, hay una mejoría en “tristeza”, ya que logra acertar algunos mas. Esto se ve mejor al comparar la precision y el recall entre las dos matrices. Usando las figuras 4 y 8 se observa que en general todas las métricas son peores en el segundo modelo, con la excepción del recall de la “tristeza” que mejora yendo de 0.70 a 0.73, lo que significa que confundió menos esta emoción en el modelo que identifica emoción y género

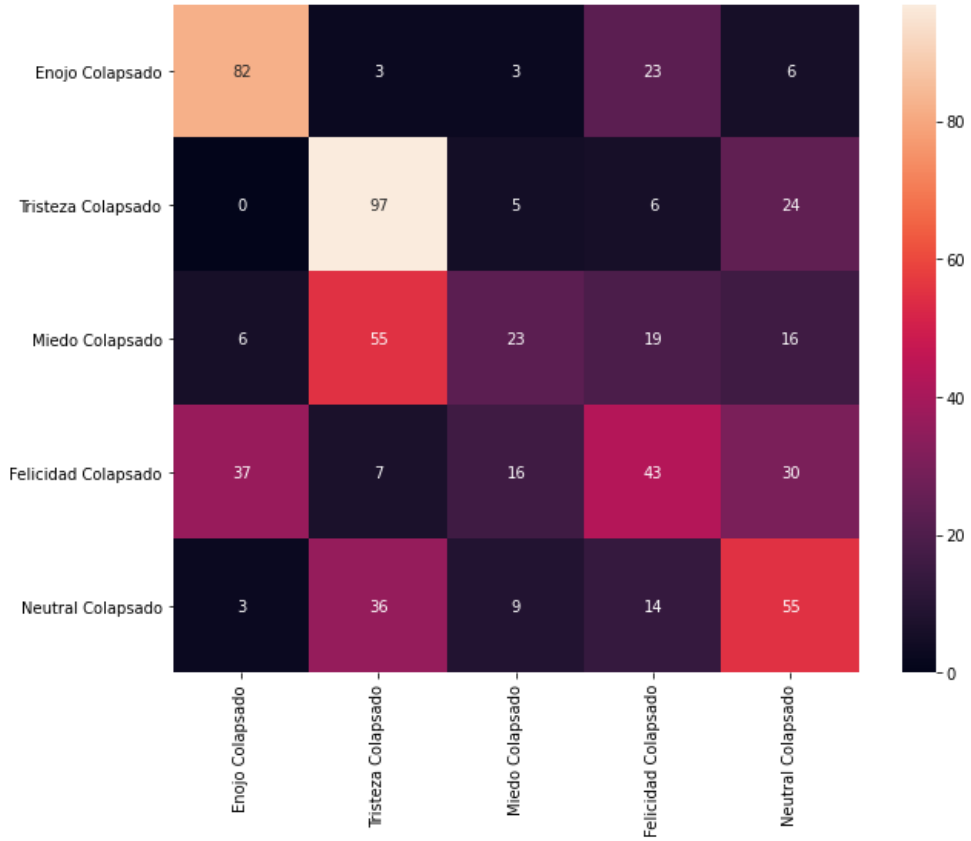

Figura 7: Matriz de confusión colapsada

	presicion	recall
Enojo Colapsado	0.64	0.70
Tristeza Colapsado	0.49	0.73
Miedo Colapsado	0.41	0.19
Felicidad Colapsado	0.41	0.32
Neutral Colapsado	0.42	0.47

Figura 8: Métricas del colapso

5. Conclusión

Es sorprendente como un modelo tan simple (en comparación, un modelo de 5 capas es simple), es capaz -luego de procesar las señales de audio con MFCC para obtener los atributos importantes para trabajar con el habla- de clasificar la emoción de quien habla a partir de audios tan cortos.

Otra grata sorpresa fue que el modelo es lo suficientemente inteligente para clasificar las emociones independientemente del sexo de quien habla.

Nos hubiera gustado realizar un clasificador que utilizara los espectrogramas(imágenes), para ver si así se obtienen mejores resultados que con los audios. También se nos había ocurrido mirar que pasaba si restringimos las edades, ya que las personas mayores hablan de forma diferente a los adultos.

Referencias

[1] AudioModel Kaggle URL: https://www.kaggle.com/uldisvalainis/audiomodel.

[2] CREMA-D URL: https://www.kaggle.com/ejlok1/cremad.

[3] Affective Computing, R. W. Picard URL: https://affect.media.mit.edu/pdfs/95.picard.pdf.