

Trabajo Final Inteligencia Artificial I – año 2024:

Visión Artificial y reconocimiento de voz

Francisco Castel

Facultad de Ingenieria, UNCUIYO, 13784, `castel.francisco@uncuyo.edu`

Abstract

This work focuses on object recognition and audio processing. For computer vision, four types of vegetables will be recognized by extracting features such as Hu moments and average color. The process includes image preprocessing followed by data clustering using the KMeans algorithm.

For audio recognition, four target words will be identified, with feature extraction based on cepstral coefficients, energy, spectral density, and other relevant parameters. Dimensionality reduction will be applied using UMAP to reduce the number of components, and the KNN algorithm will be used for classification. The importance of dimensionality reduction will be highlighted on audio classification, on the other hand in image recognition, color takes a lot of credit.

Keywords: kmeans, knn, dimensionality, audio, image, clusterization.

Note 1: the following document is in spanish.

Note 2: around 50 % of the redaction is LLM based.

1. Resumen	22
La primera implementación de este trabajo se centra en el reconocimiento de objetos y la segunda en el procesamiento de audio. En el caso de la visión artificial, se reconocerán cuatro tipos de verduras, para los cuales se extraerán características como los momentos de Hu y el color promedio. Este proceso requerirá un preprocesamiento de las imágenes, seguido de una clusterización de los datos utilizando el algoritmo KMeans.	23 24 25 26 27
Por otro lado, en el reconocimiento de voz, se identificarán cuatro palabras clave. La extracción de características incluirá coeficientes cepstrales, energía, densidad espectral y otros parámetros relevantes. Posteriormente, se aplicará una reducción de dimensionalidad usando UMAP, con el objetivo de reducir significativamente la cantidad de componentes. Finalmente, se utilizará el algoritmo KNN para clasificar las nuevas entradas de audio. A lo largo del proceso, se evidenciará la importancia de reducir la dimensionalidad de los datos, y se destacará que, en el caso de las imágenes, los colores juegan un papel crucial en el reconocimiento.	28 29 30 31 32 33 34 35
2. Introducción	36
La visión artificial es una rama de la inteligencia artificial (IA) que busca desarrollar sistemas capaces de interpretar y comprender el mundo visual de manera similar a los seres humanos. Utilizando cámaras, sensores y algoritmos avanzados, la visión artificial permite a las máquinas analizar imágenes y videos para extraer información relevante. Entre las aplicaciones más comunes de esta tecnología se encuentran el reconocimiento de objetos, la segmentación de imágenes, la clasificación de patrones y la detección de anomalías, entre otras. En el campo del reconocimiento de objetos, se busca identificar y clasificar elementos presentes en una imagen, lo cual es fundamental para diversas aplicaciones como la automatización industrial, la robótica y el diagnóstico médico.	37 38 39 40 41 42 43 44 45
Por otro lado, el reconocimiento de voz es una disciplina dentro del procesamiento del lenguaje natural que se enfoca en la capacidad de las máquinas para identificar y comprender el habla humana. A través de técnicas de análisis de señales acústicas, el reconocimiento de voz convierte las ondas sonoras en texto o en comandos que pueden ser interpretados por un sistema computacional. Este campo ha tenido un crecimiento notable gracias a avances en la extracción de características acústicas, como los coeficientes cepstrales, y el uso de algoritmos de clasificación que permiten mejorar la precisión en la interpretación del habla, incluso en entornos ruidosos.	46 47 48 49 50 51 52 53
El objetivo principal de este trabajo es abordar un problema de reconocimiento multimodal, en el que se combinan técnicas de visión artificial y reconocimiento de voz para mejorar la precisión en la identificación de objetos y palabras. En este sentido, se pretende explorar cómo las características visuales, como los momentos de Hu y el color promedio, y las características acústicas, como los coeficientes cepstrales y la energía, pueden ser	54 55 56 57 58

utilizadas de manera conjunta para lograr una clasificación más eficiente y robusta en situaciones reales. Además, se explorarán técnicas de reducción de dimensionalidad que permitan optimizar los modelos, haciendo más eficiente el proceso de clasificación tanto en imágenes como en datos de audio.

3. Especificacion del agente

Para hablar de un agente, primero debemos definir qué tipo de agente es el que se ha desarrollado y qué es lo que consideraremos agente. Podríamos definir dos estrategias para el análisis, la primera es considerar un unico agente que se encarga de la clasificacion de imagenes y de audio, por otro lado, dado que son de funcionamiento independiente pueden considerarse 2 agentes. Ambos son agentes que **aprenden** pero la principal diferencia entre ellos ademas de el metodo sensorial que utilizan es que en el caso de las imagenes el aprendizaje es **no supervisado** y en el caso del agente es **supervisado**. Se eligirá esta ultima propuesta de identificacion de agente, dado que presentan diferencias sustanciales.

3.1. Tabla REAS

Agente	Rendimiento	Entorno	Actuadores	Sensores
Clasificador de imágenes con aprendizaje no supervisado	Precisión: Si la predicción de la verdura es correcta (es decir, la predicción es igual a la categoría real de la verdura).	Entorno controlado con iluminación neutra. Las imágenes se toman siempre con el mismo fondo, y la rotación de las verduras no afecta a la clasificación.	Pantalla de la computadora que muestra la predicción de la verdura al usuario. No hay acción física directa sobre el entorno.	Cámara fotográfica de celular con flash (captura imágenes en formato RGB).
Clasificador de audio, aprendizaje supervisado	Precisión: La precisión con la que el sistema clasifica los audios correctamente, es decir, si la predicción de la palabra es correcta.	El entorno es controlado, ya que el sistema funciona en condiciones de bajo nivel de ruido. No es adecuado para ambientes con alto nivel de ruido.	Pantalla de la computadora que muestra la predicción del audio al usuario. No hay interacción física directa con el entorno.	Micrófono y driver de audio con al menos una tasa de muestreo de 16KHz.

Cuadro 1: Tabla REAS para el Agente de Clasificación de Imágenes

3.2. Propiedades del entorno

Para la definición de las propiedades del entorno, dado a lo visto en la tabla REAS, se comparten bastantes características del mismo. Por lo que, como se explicará a continuación, se pueden definir las mismas propiedades del entorno para ambos agentes.

Determinístico vs. Estocástico: El entorno es **determinístico** porque las acciones del agente tienen un resultado predecible y fijo. Es decir, siempre que el agente reciba la misma entrada (ya sea una imagen o un archivo de audio), el resultado de la clasificación será el mismo. No hay incertidumbre ni aleatoriedad en la interacción del agente con el entorno, ya que los procesos de clasificación se basan en características bien definidas de las entradas.

Totalmente Observable vs. Parcialmente Observable: El entorno es **totalmente observable** porque el agente tiene acceso completo a la información necesaria para realizar la clasificación. El agente no necesita inferir información adicional fuera de los datos proporcionados (ya sea la imagen completa o el archivo de audio). Las entradas son claras y completas para el agente, lo que le permite tomar decisiones precisas sin tener que adivinar o estimar valores no observados.

Secuencial vs. Episódico: El entorno es **secuencial** porque la clasificación de cada imagen o audio depende del procesamiento de la entrada en un orden específico. Cada decisión del agente influye en el futuro, ya que el agente debe aprender de las entradas anteriores para mejorar sus predicciones. Esto implica que el agente no puede clasificar de manera independiente cada entrada sin tener en cuenta el contexto de las decisiones previas.

Estático vs. Dinámico: El entorno es **estático** porque las condiciones en las que el agente realiza la clasificación no cambian mientras está procesando una entrada. Las imágenes y los audios se toman previamente y no cambian durante el proceso de clasificación. El agente puede tomar su decisión sin preocuparse de que el entorno modifique las entradas.

Discreto vs. Continuo: El entorno es **discreto** porque tanto las imágenes como los audios se representan en unidades discretas. Las imágenes son divididas en píxeles, y los audios se transforman en coeficientes discretos (por ejemplo, coeficientes cepstrales). Además, las decisiones del agente (como la clasificación de las imágenes y audios) también son discretas, ya que el agente selecciona una categoría específica.

Agente Único vs. Multiagente: El entorno es de **agente único** porque en este proyecto solo se cuenta con un agente encargado de la clasificación de imágenes o audios. No hay interacción con otros agentes que afecte el desempeño del agente en el proceso de clasificación.

4. Diseño de los agentes

A continuación se presentan los algoritmos e implementaciones necesarias para la construcción de los agentes previamente especificados.

4.1. Clasificador de imagenes

112

En el caso de este agente, se consigna por parte de la catedra la utilización del conocido algoritmo KMeans, incluyendo la clusterización y luego la distancia media a cada centroide formado.

Algorithm 1 Algoritmo K-Means

Require: Conjunto de datos $X = \{x_1, x_2, \dots, x_m\}$, número de clústeres k , número máximo de iteraciones T

Ensure: Etiquetas de los clústeres $C = \{c_1, c_2, \dots, c_m\}$ y centros de los clústeres $M = \{m_1, m_2, \dots, m_k\}$

```
1: Inicializar aleatoriamente los  $k$  centros  $m_1, m_2, \dots, m_k$ 
2:  $t \leftarrow 0$ 
3: repeat
4:   for cada punto  $x_i \in X$  do
5:     Asignar  $x_i$  al clúster  $c_i$  tal que  $c_i = \arg \min_j \|x_i - m_j\|^2$ 
6:   end for
7:   for cada clúster  $j = 1, 2, \dots, k$  do
8:     Actualizar el centro del clúster  $m_j \leftarrow \frac{1}{|C_j|} \sum_{x_i \in C_j} x_i$ 
9:   end for
10:   $t \leftarrow t + 1$ 
11: until la convergencia o  $t \geq T$ 
```

115

Estrategia de base de datos

116

Para la base de datos se optó por armar la propia con imágenes capturadas con la cámara con la que se realizarán las entradas al programa, esto básicamente para no entrar en inconvenientes de sensores distintos y diferentes representación de los colores y luz captada. La base de datos consta de 63 imágenes donde la cantidad de imágenes por verdura esta casi repartida uniformemente.

121

Preprocesamiento de imágenes

122

El pilar del procesamiento de imágenes de este trabajo es la librería `opencv-python`¹ la cual ofrece varios filtros y varias funciones de extracción de características. La aplicación de los filtros en orden es la siguiente:

125

■ Filtro 1: Desenfoque Gaussiano

126

- **Descripción:** Este filtro aplica un desenfoque gaussiano para suavizar la imagen y reducir el ruido de alta frecuencia. El objetivo es eliminar detalles irrelevantes para mejorar la precisión de los siguientes pasos.

129

- **Parámetros:**

130

- (13, 13): Tamaño del núcleo del filtro, de 13×13 píxeles.

131

¹Más detalles en Bradski, 2000

○ 0: Desviación estándar, que se calcula automáticamente.	132
■ Filtro 2: Conversión a escala de grises	133
● Descripción: Convierte la imagen de color (BGR) a una imagen en escala de grises. Este paso es importante para reducir la complejidad y trabajar solo con las intensidades de píxeles, sin la información de color.	134 135 136
■ Filtro 3: Umbralización Adaptativa	137
● Descripción: Se aplica un umbral adaptativo para binarizar la imagen. El valor del umbral varía localmente en función de la vecindad de cada píxel, lo que permite manejar mejor las variaciones de iluminación en la imagen.	138 139 140
● Parámetros:	141
○ 255: Valor máximo que se asignará a los píxeles que superen el umbral.	142
○ <code>cv2.ADAPTIVE_THRESH_MEAN_C</code> : Método adaptativo basado en la media de los píxeles vecinos.	143 144
○ <code>cv2.THRESH_BINARY_INV</code> : Tipo de umbral binario invertido. Los píxeles debajo del umbral se asignan a blanco (255) y los que lo superan a negro (0).	145 146 147
○ 31: Tamaño de la vecindad usada para calcular el umbral de cada píxel.	148
○ 6: Constante que se resta del valor medio para el cálculo del umbral adaptativo.	149 150
■ Filtro 4: Operación morfológica (dilatación)	151
● Descripción: Este filtro realiza una dilatación morfológica sobre la imagen binarizada. La dilatación expande las áreas blancas y conecta regiones cercanas, lo que facilita la detección de contornos y áreas de interés.	152 153 154
● Parámetros:	155
○ <code>cv2.MORPH_CROSS</code> : Tipo de estructura del elemento morfológico, en este caso una cruz (<code>MORPH_CROSS</code>).	156 157
○ (11, 11): Tamaño del núcleo de dilatación de 11×11 píxeles.	158
Debe notarse como prácticamente la imagen queda igual, esto es por la alta resolución de la misma, aun así el filtro gaussiano limpia exitosamente el ruido de fondo.	159 160



Figura 1: Imagen original



Figura 2: Imagen luego de aplicar gaussian blur



Figura 3: Imagen binarizada sin filtrado previo

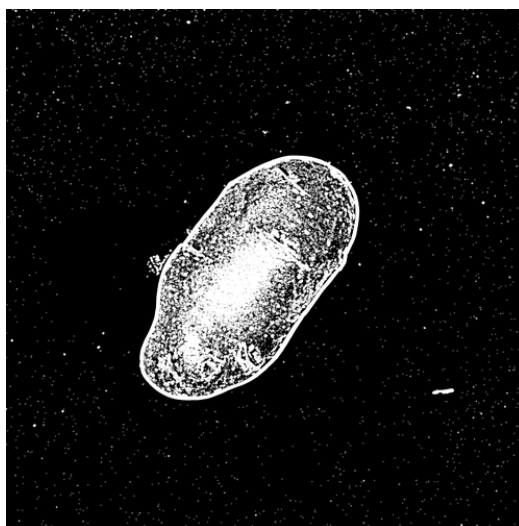


Figura 4: Filtro morfológico sin filtrado previo

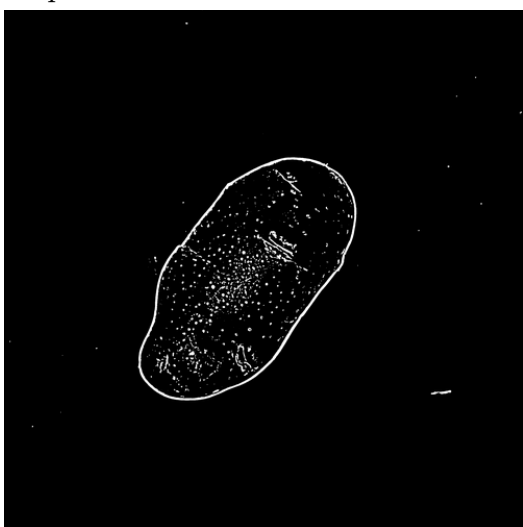


Figura 5: Imagen binarizada con blur aplicado



Figura 6: Filtro morfológico con blur aplicado

Extracción de características 161

Una vez que se han aplicado los filtros, se procede con la extracción de características 162 de la imagen procesada. En esta sección se detallan los pasos utilizados para obtener las 163 características. 164

■ Detección de contornos 165

- **Descripción:** Se identifican los contornos en la imagen utilizando el algoritmo 166 `cv2.findContours`. Este proceso permite detectar las fronteras de los objetos 167 presentes en la imagen. 168

- **Parámetros:** 169

- `cv2.RETR_EXTERNAL`: Modo de recuperación de contornos que solo extrae 170 los contornos externos de los objetos. 171
- `cv2.CHAIN_APPROX_SIMPLE`: Método de aproximación de contornos que 172 almacena solo los puntos extremos de los segmentos de contorno. 173

- **Proceso:** 174

- Se detectan los contornos presentes en la imagen procesada. 175
- Se selecciona el contorno con el área más grande, que generalmente repre- 176 senta el objeto de interés en la imagen. 177

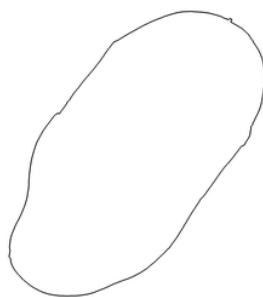


Figura 7: Contorno capturado

■ Cálculo de momentos de Hu 178

- **Descripción:** Los momentos de Hu [Wikipedia contributors, 2024] son inva- 179 riantes a transformaciones geométricas (traslación, rotación y escala). Se uti- 180 lizan para describir la forma del contorno y se calculan a partir del contorno 181 más grande detectado. 182

• Proceso:	183
◦ Se calculan el segundo y el tercer momento de Hu utilizando <code>cv2.moments</code>	184
y <code>cv2.HuMoments</code> .	185
■ Cálculo del color promedio	186
• Descripción: El color promedio dentro del área delimitada por el contorno	187
detectado se calcula como una característica adicional.	188
• Proceso:	189
◦ Se utiliza una máscara generada a partir del contorno para extraer los	190
píxeles dentro de la región de interés.	191
◦ Se aumenta la saturación y el brillo de la imagen dentro de la mascara con	192
factores de 1.03 y 1.7 respectivamente.	193
◦ Se calcula el valor promedio de los canales de color (Rojo, Verde, Azul) de	194
los píxeles en esa región.	195
■ Almacenamiento y visualización de características	196
• Descripción: Los momentos de Hu y el color promedio se guardan en un	197
archivo CSV para su posterior análisis o clasificación.	198
• Proceso:	199
◦ Los momentos de Hu seleccionados y el color promedio se almacenan en	200
un archivo de tipo CSV con encabezados apropiados.	201
◦ Se visualiza la imagen procesada, destacando los contornos y mostrando	202
la región de interés.	203
Dimensionalidad y PCA	204
Para el caso de las imágenes las dimensiones iniciales son 5, dado que se tienen:	205
■ 3er Momento de Hu	206
■ 4to Momento de Hu	207
■ Media del color Rojo	208
■ Media del color Verde	209
■ Media del color Azul	210
Es importante aclarar que de forma matemática el algoritmo KMeans utiliza la distancia	211
Euclidiana para la clusterización, la misma pierde sentido en más de 3 dimensiones, pero,	212
aun así funciona de forma aceptable dado que solo se esta excediendo por 2 la dimensio-	213
nalidad recomendada. De todas maneras, se experimento con dos métodos de reducción	214

de dimensionalidad, el primero, PCA 8 muestra una separación aunque aceptable, no sufi- 215
 ciente para el algoritmo dado que se pretende trabajar con una dimensionalidad reducida, 216
 luego se encuentra UMAP 9 que muestra una notable separación de los datos, obteniendo 217
 ventajas en la clusterización por KMeans y en la eficiencia del modelo. 218

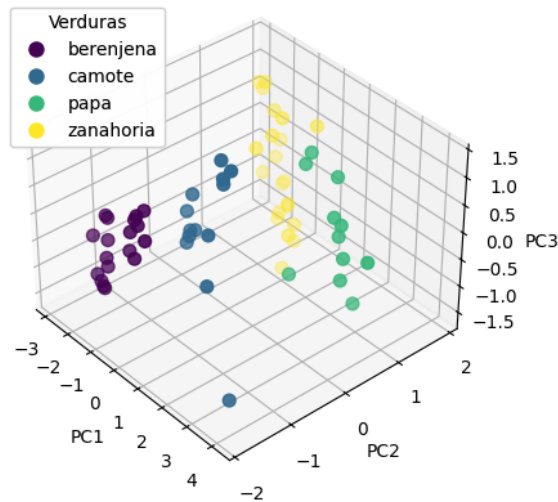


Figura 8: PCA (3 componentes) realizado a las 5 componentes iniciales

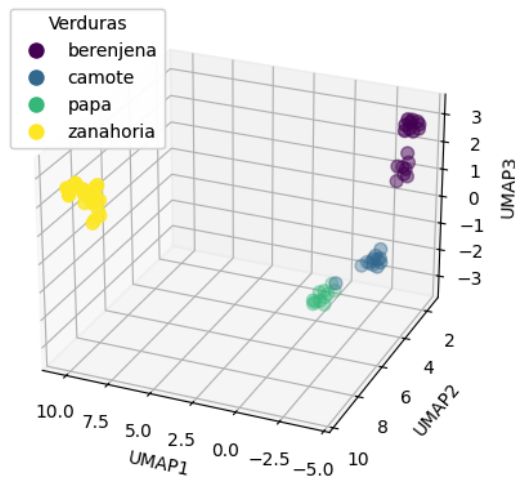


Figura 9: UMAP (3 componentes) realizado a las 5 componentes iniciales

4.2. Clasificador de audios

En el caso del clasificador de audios, al tratarse de una agente que aprende de forma 220
 supervisada, la cátedra consigna el uso del conocido algoritmo KNN. 221
 Se eligió la particularidad de utilizar siempre un valor de K impar, para no resolver 222
 innecesarios empates. 223

Para el calculo de la distancia se utiliza la distancia de Minkowski con un valor de p igual a 2, dando como resultado la distancia Euclidiana en n-dimensiones. La distancia de Minkowski entre dos puntos $x = (x_1, x_2, \dots, x_n)$ y $y = (y_1, y_2, \dots, y_n)$ en un espacio n -dimensional es:

$$d(x, y) = \left(\sum_{i=1}^n |x_i - y_i|^p \right)^{\frac{1}{p}}$$

Algorithm 2 Algoritmo de K-Nearest Neighbors (KNN)

Require: $X = \{x_1, x_2, \dots, x_n\}$: conjunto de entrenamiento con n ejemplos.

1:

Require: $Y = \{y_1, y_2, \dots, y_n\}$: etiquetas correspondientes al conjunto de entrenamiento.

2:

Require: k : número de vecinos más cercanos a considerar.

3:

Require: x_{query} : ejemplo de consulta (punto de datos para clasificar).

Ensure: Predicción de clase para x_{query} .

4: **Paso 1:** Calcular la distancia entre x_{query} y cada punto de entrenamiento x_i .

5:

6: **Paso 2:** Ordenar los puntos de entrenamiento en función de las distancias calculadas.

7:

8: **Paso 3:** Seleccionar los k vecinos más cercanos.

9:

10: **Paso 4:** Obtener las etiquetas de los k vecinos seleccionados: y_1, y_2, \dots, y_k .

11:

12: **Paso 5:** Realizar una votación para determinar la clase mayoritaria entre las etiquetas y_1, y_2, \dots, y_k .

13:

14: **Paso 6:** Devolver la clase mayoritaria como la predicción para x_{query} .

228

Estrategia de base de datos	229
Para el caso de la base de datos de los audios se opto por un enfoque inclinado a tener una gran cantidad de datos (alrededor de 380 muestras), la justificación para esto es el hecho de que la idea inicial era que no se limitara a ciertas voces, ya sea por el timbre de la voz dado el sexo o cualquier otra característica adicional, en pocas palabras una gran base de datos implica robustez ya que, al fin y al cabo, el modelo se entrena con mas ruido y es mas propenso a 'aprender' patrones, aunque también implica un gran costo computacional para el procesamiento de las muestras.	230 231 232 233 234 235 236
Preprocesamiento del audio	237
En este proyecto, el audio pasa por una serie de etapas de preprocesamiento antes de ser utilizado para la clasificación de palabras. A continuación se describen los pasos seguidos en el filtrado y la normalización del audio:	238 239 240
■ Pre-énfasis:	241
<ul style="list-style-type: none"> • La etapa de pre-énfasis tiene como objetivo resaltar las altas frecuencias del audio para mejorar la discriminación de las características. Esto es particularmente útil en el procesamiento de señales de voz, ya que ayuda a compensar la atenuación de frecuencias altas en el canal de transmisión. • Se aplica un filtro de pre-énfasis con un coeficiente de 0.99 usando la función <code>preemphasis</code> de <code>librosa</code>². Esto amplifica las frecuencias altas y prepara el audio para las siguientes etapas de filtrado. 	242 243 244 245 246 247 248
■ Filtro paso banda dinámico:	249
<ul style="list-style-type: none"> • El audio pasa por un filtro paso banda dinámico para eliminar las frecuencias fuera de un rango útil para la clasificación de las palabras. Este filtro ayuda a reducir el ruido en frecuencias no relevantes para la tarea de clasificación y mejora la calidad de las características extraídas. • Se configura un filtro de Butterworth de orden 4, con un corte bajo en 50 Hz y un corte alto en 8000 Hz. El corte superior se ajusta dinámicamente para no superar la frecuencia de Nyquist, que depende de la tasa de muestreo <code>sr</code> del audio. • El filtro se aplica usando la función <code>signal.butter</code> para crear los coeficientes del filtro y <code>signal.filtfilt</code> para aplicar el filtro al audio pre-énfasis. Este paso elimina frecuencias no deseadas, especialmente por debajo de 50 Hz y por encima de 8000 Hz. 	250 251 252 253 254 255 256 257 258 259 260 261

²Mas detalles en McFee et al., 2015

- **Reducción de ruido:** 262
 - La reducción de ruido se lleva a cabo utilizando un enfoque basado en la 263
descomposición de la transformada de Fourier (STFT) y un umbral de rui- 264
do adaptativo. Este paso permite eliminar componentes no deseadas del audio 265
que corresponden a ruido ambiental o interferencias. 266
 - Primero, se calcula la STFT del audio filtrado y se obtiene el espectro de 267
magnitudes S . Se calcula un umbral de ruido basado en la media del espectro, 268
multiplicado por un factor de 1.5. Luego, se aplica una máscara para reducir 269
las componentes espectrales que están por debajo de este umbral. 270
 - Esta máscara se ajusta mediante el uso de un filtro de vecindad mediana para 271
refinar la estimación de las componentes útiles. La fase se mantiene intacta y 272
se reconstruye la señal limpia mediante la inversa de la STFT (`istft`). 273
- **Normalización de Loudness:** 274
 - La normalización de la loudness (volumen percibido) se realiza para garantizar 275
que el nivel de volumen del audio esté dentro de un rango objetivo estándar. 276
Esto es especialmente importante en tareas de clasificación, ya que los modelos 277
pueden verse afectados por variaciones en el volumen de las grabaciones. 278
 - Se utiliza el medidor de loudness de `pyn.Meter` para calcular el `loudness_actual` 279
del audio filtrado. Luego, se ajusta el volumen del audio para que el nivel de 280
loudness coincida con un valor objetivo de -23 LUFS (Loudness Units Full 281
Scale), que es el estándar para audio de calidad. 282
 - La función `pyn.normalize.loudness` ajusta la loudness del audio, asegu- 283
rando que el volumen final sea adecuado para su posterior procesamiento y 284
clasificación. 285

Extracción de características 286

Se extraen varias características acústicas del audio para su clasificación. Estas carac- 287
terísticas se calculan tanto para cada segmento del audio como para el archivo de audio 288
completo. Es importante aclarar que la extracción de estas se realiza mediante la conocida 289
librería para tratamiento de señales `librosa`, disponible para Python. 290

- **13 MFCCs (Mel-Frequency Cepstral Coefficients):** 291
 - Para cada segmento del audio, se calculan los 13 coeficientes **MFCC**, que repre- 292
sentan las características espectrales más importantes de la señal. La media de 293
cada uno de los 13 coeficientes se calcula a lo largo de todo el segmento, lo que 294
proporciona una descripción compacta y estable de las características acústicas 295
de la palabra en ese segmento. 296

- Estos coeficientes son cruciales para capturar las variaciones espectrales que permiten diferenciar las 4 palabras. Cada palabra tiene un patrón acústico único que puede ser descrito por sus coeficientes MFCC. Al calcular la media, se obtienen representaciones robustas de cómo las características espectrales de la palabra se comportan en el tiempo.
- **Estadísticas adicionales de los 13 MFCCs:**
 - Además de la media, se calculan tres estadísticas adicionales para cada uno de los 13 coeficientes MFCC:
 - **Valor máximo:** El valor máximo de cada uno de los coeficientes MFCC proporciona información sobre los picos espectrales dentro del segmento de audio. Las palabras pueden tener características distintivas que se manifiestan como picos espectrales en ciertas frecuencias, y capturar estos máximos puede ser útil para diferenciarlas.
 - **Valor mínimo:** El valor mínimo de los coeficientes MFCC resalta las frecuencias menos prominentes de la palabra, lo que puede ayudar a identificar las partes de la palabra que tienen una presencia espectral más baja o sutil, y también a distinguir entre palabras con un patrón espectral menos intenso.
 - **Desviación estándar:** La desviación estándar captura la variabilidad de los coeficientes MFCC dentro del segmento. Una baja desviación estándar indica una señal más constante, mientras que una alta desviación estándar puede indicar una mayor variabilidad en las características espectrales, lo cual es importante para distinguir palabras que tienen variabilidad en su pronunciación.
- **RMS (Root Mean Square):**
 - El valor RMS se calcula para cada segmento de audio y mide la amplitud promedio de la señal. Este valor es crucial para determinar la "intensidad" general de la señal. Las 4 palabras pueden tener diferentes niveles de intensidad acústica (por ejemplo, algunas pueden ser pronunciadas con más énfasis o mayor volumen). El cálculo del RMS ayuda a capturar esta variabilidad y es útil para identificar patrones relacionados con la pronunciación de las palabras.
- **Duración del Audio:**
 - La duración del audio es una característica única que mide el tiempo total de la grabación, en segundos. En este caso, la duración es relevante porque las 4 palabras a clasificar tienen una duración aproximada consistente, y la duración total del archivo de audio puede ayudar a identificar patrones de tiempo que están asociados con las palabras específicas. Aunque no se calcula por segmento,

la duración total de la grabación puede servir como una referencia útil para el 334
 modelo, especialmente si se tiene en cuenta la variabilidad de la duración de 335
 las palabras en diferentes entornos acústicos o de pronunciación. 336

Tratamiento de dimensionalidad 337

Este agente se caracteriza por tener que lidiar con un conjunto de datos de gran 338
 dimension a comparación del agente anterior, finalmente se obtiene un conjunto de datos 339
 de 209 dimensiones. La realidad es que aunque KNN podría llegar a funcionar aun teniendo 340
 tantas componentes, la probabilidad de encontrar vecinos **realmente cercanos** es cada 341
 vez menor. Se tratará en el apéndice más detalles sobre esto. Primero se propuso la 342
 utilización de PCA al igual que en el clasificador de imágenes, funcionando correctamente 343
 cuando se reducían las dimensiones a 15, esto reducía bastante el costo computacional pero 344
 aun así la separación de de los clusters no era del todo convincente. Por lo que mediante 345
 una serie de prompts a ChatGPT, se propuso la utilización de UMAP (Uniform Manifold 346
 Approximation and Projection) el cual logró una considerable separación de clusters a 347
 comparación de PCA, además de logrando reducir la dimensionalidad del problema a solo 348
 3, por lo tanto, finalmente las distancias son realizadas en 3 dimensiones y el algoritmo 349
 no pierde su sentido geométrico. 350

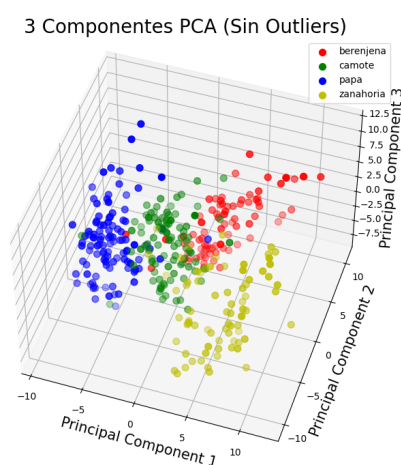


Figura 10: Datos luego de aplicar PCA en 3 dimensiones.

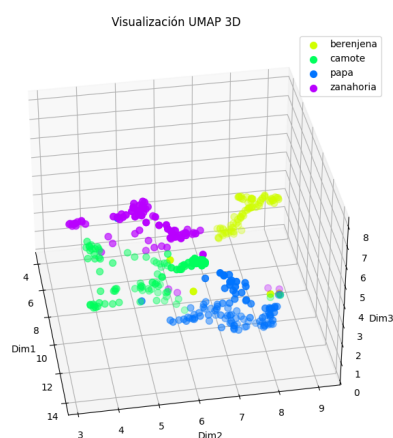


Figura 11: Datos luego de aplicar UMAP en 3 dimensiones.

Figura 12: Comparación entre PCA y UMAP aplicados a los datos en 3 dimensiones.

Es notable la filtración al ruido que realiza UMAP, aun así es importante destacar 351
 que aunque este algoritmo obtiene grandes resultados en terminos de redimensionalidad, 352
 no sería posible si el conjunto de datos inicial de n-dimensiones no formara subespacios, 353
 es decir que si la información original no respetara alguna relación UMAP no tiene la 354
 capacidad de generarla. 355

4.3. Eficiencia de los agentes

356

Para el clasificador de audios la estrategia de comprobación de eficiencia fue la de a 357
partir de la base de datos original, entrenar al algoritmo con un 70 % de la información, 358
se utilizaba el 30 % restante para como muestras de prueba. Este enfoque no siempre 359
presentaba resultados coherentes frente a la prueba de campo cuando un ser humano 360
debía grabar una nueva voz y pasarla como muestra al sistema, aun así era un indicativo 361
de si se estaba empeorando o no la estrategia de características. Aun así a veces ya sea 362
por overfitting o por la cantidad tan grande de datos que se tienen arrojaba eficiencias 363
superiores al 90 % aun cuando el sistema mostraba grandes deficiencias en pruebas de 364
campo. 365

Finalmente se termino obteniendo una eficiencia del 94 % que al menos si representa las 366
pruebas de campo. 367

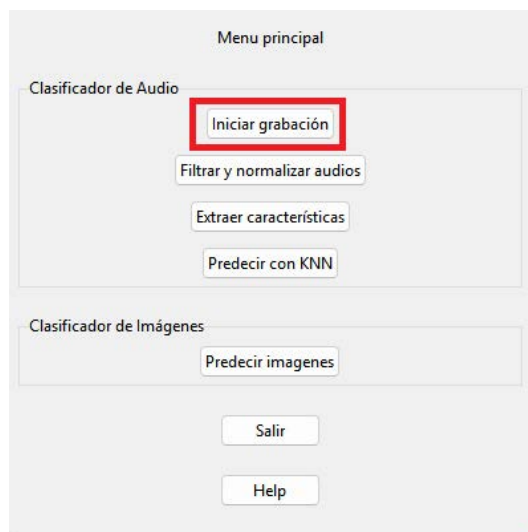
5. Repositorio del proyecto

368

Se adjunta el link al repositorio que contiene toda la información respecto al código 369
fuente, y el código fuente en si mismo. 370

<https://github.com/fcastel2002/Trabajo-final-IA1---Castel> 371

6. Ejemplo de uso de la aplicación



6. Conclusiones

372

Son varias las conclusiones extraídas del proyecto presentado, la primera es que para el clasificador de imágenes, existen bastantes limitantes al momento de diseñar un modelo robusto y a prueba de variaciones en el entorno, sobre todo por las particularidades del sensor que presenta el agente, al fin y al cabo extraer los colores de una verdura es un proceso que aunque eficiente si se realiza siempre en el mismo entorno, muy susceptible a variaciones de iluminación, por suerte los momentos de Hu impiden de cierta forma confundir notablemente dos verduras que tengan colores similares, así se presentarían enormes dificultades con el agente diseñado si se pretendiera realizar la clasificación de verduras con colores similares o directamente iguales. Por otro lado en el caso del clasificador de audio se obtuvieron conclusiones mucho más enriquecedoras, primero se tiene la muy notable influencia de los coeficientes cepstrales, los cuales diferencian sustancialmente a las palabras (es de esperarse dado que estos coeficientes se calculan en escala MEL, la misma escala para obtener los Espectrogramas MEL [13], estos son comunmente utilizados en la clasificación de audios con redes neuronales en formato imagen), luego fue interesante el descubrimiento de que la tasa de cruces por cero no estaba arrojando valores relevantes más bien estaba añadiendo ruido al conjunto de datos.

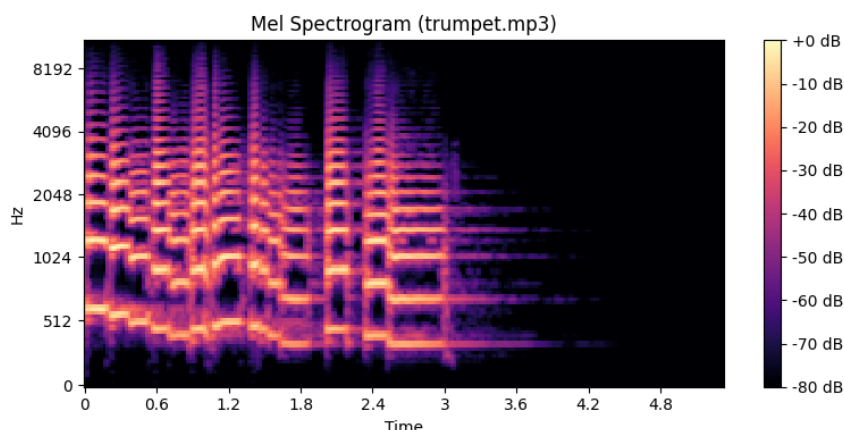


Figura 13: Imagen tomada de <https://notesbylex.com/melspectrogram>

388

Generalización de los datos

389

Clasificador de audios

390

A mitad de proyecto se pretendió hacer más robusto el modelo mediante una normalización del tono a través la extracción de la frecuencia relacionada al mismo F0, para realizar una normalización en todos los audios, así, se lograría que el sistema de reconocimiento funcionara tanto como para voces graves como para voces agudas. Este enfoque trajo mas problemas que soluciones dado que el proceso de normalización de tono no es preciso y es complejo que los audios queden con el mismo tono. Finalmente se decidió por

396

expandir enormemente la base de datos de aproximadamente 50 audios a 380. Además se
llegó a la conclusión de que los coeficientes que se estaban extrayendo no eran sumamente
dependientes del tono de los audios, por lo que tener audios de distinto tono no genero
una dispersión notable.

Clasificador de imágenes

Al principio se propuso utilizar los 7 Momentos de Hu, sin los colores promedios,
lo que realmente no generaba ningún tipo de patrón distinguible en los datos, luego se
implemento la obtención del color promedio pero esto no solucionaba lo anterior, llegando
a la conclusión que la utilización de los 7 Momentos generaba ruido no despreciable en
el conjunto de muestras, por lo que, realizando pruebas, se decidió quedarse solo con
2 Momentos de Hu, la realidad de las pruebas mostraba despreciable a la selección de
momentos específicos, por lo tanto la elección de estos dos, es arbitraria.

Uso de IA generativa

Para este proyecto se utilizaron dos principales IA generativas, la primera, ChatGPT,
el cual fue de gran ayuda al momento de iniciar el proyecto y consultar sobre las caracte-
rísticas que se suelen extraer a las imágenes y a los audios, también sobre métodos de
reducción de dimensionalidad, por otro lado proporcionó el código inicial de los algorit-
mos, la segunda IA generativa que se utilizó fue Github Copilot, en su versión de Edits
actualmente disponible para Visual Studio Code, la misma fue de suma ayuda al momento
de realizar código, ya que en base a las propuestas de ChatGPT se realizaba el prompt
correspondiente a Copilot, el cual conocía las librerías y los métodos de ellas. Los mode-
los LLM utilizados fueron GPT 4o, o1-mini y o1-preview para prompts que implicaban
grandes modificaciones sobre todo en código.

Un claro ejemplo de un prompt que aumentó notablemente la eficiencia de los agentes fue
a partir de mencionar que aunque la varianza de las características extraídas actualmen-
te eran no correspondientes con características que son redundantes, y además también
parecía mostrar cierta diferencia para cada etiqueta el diagrama de PCA no mostraba
suficiente separación de las distintas etiquetas, este fue el precursor de que la IA gene-
rativa propusiera utilizar otras técnicas de re dimensionamiento de la información, como
por ejemplo UMAP, esta implementación fue clave en el proyecto, además hizo que se
dedicara extensamente a investigar en este área.

Creo que para este proyecto el uso de este tipo de herramientas fue fundamental para
comprender más que es lo que se estaba haciendo, al fin y al cabo la utilización de la
misma depende de el uso que se le dé y la conciencia con la que se realice.

Referencias

- Bradski, G. (2000). The OpenCV Library. *Dr. Dobb's Journal of Software Tools*. 431
- McFee, B., Raffel, C., Liang, D., Ellis, D. P. W., McVicar, M., Battenberg, E., & Nieto, 433
O. (2015). librosa: Audio and Music Signal Analysis in Python [Accedido: 18 de 434
diciembre de 2024]. <https://doi.org/10.5281/zenodo.3961370> 435
- Miner, H. D. (2023). *K-NN o la influencia de los vecinos* [Accedido: 23 de noviembre de 436
2024]. <https://healthdataminer.com/data-mining/k-nn-o-la-influencia-de-los-vecinos/> 437
438
- UNCUYO, I. (2024). *Clase 8: Percepción* [Accedido: 14 de noviembre de 2024]. [https://aulaabierta.ingenieria.uncuyo.edu.ar/pluginfile.php/44301/mod_resource/](https://aulaabierta.ingenieria.uncuyo.edu.ar/pluginfile.php/44301/mod_resource/content/9/Percepci%C3%B3n%202024.pdf) 439
[content/9/Percepci%C3%B3n%202024.pdf](https://aulaabierta.ingenieria.uncuyo.edu.ar/pluginfile.php/44301/mod_resource/content/9/Percepci%C3%B3n%202024.pdf) 440
441
- University, C. (2022a). CS 4780 Lecture Notes: Dimensionality Reduction [Accedido: 20 442
de noviembre de 2024]. [https://www.cs.cornell.edu/courses/cs4780/2022sp/notes/](https://www.cs.cornell.edu/courses/cs4780/2022sp/notes/LectureNotes03.html) 443
[LectureNotes03.html](https://www.cs.cornell.edu/courses/cs4780/2022sp/notes/LectureNotes03.html) 444
- University, C. (2022b). CS 4780 Lecture: Dimensionality Reduction and PCA [Accedido: 445
20 de noviembre de 2024]. <https://www.youtube.com/watch?v=oymtGlGdT-k> 446
- Wikipedia contributors. (2024). Image moment — Wikipedia, The Free Encyclopedia 447
[Accedido: 14 de diciembre de 2024]. [https://en.wikipedia.org/wiki/Image_](https://en.wikipedia.org/wiki/Image_moment) 448
[moment](https://en.wikipedia.org/wiki/Image_moment) 449

Apéndice A: Reducción de Dimensionalidad con UMAP 450

En este proyecto, se empleó la técnica de reducción de dimensionalidad UMAP (Proyección Uniforme Aproximada y Manifold) para transformar los datos originales a un espacio de menor dimensionalidad. Este paso fue esencial para mejorar la eficiencia computacional y facilitar la interpretación de los resultados en ambos agentes (no supervisado y supervisado). 451 452 453 454 455

La técnica PCA (Análisis de Componentes Principales) fue considerada inicialmente; sin embargo, debido a su naturaleza lineal, no logró filtrar adecuadamente el ruido presente en los datos, lo que motivó la elección de UMAP como una alternativa más robusta. 456 457 458

A.1 UMAP (Uniform Manifold Approximation and Projection) 459

UMAP es una técnica de reducción de dimensionalidad no lineal que: 460

- Conserva las relaciones locales entre los puntos de datos, asegurando que puntos cercanos en el espacio original permanezcan cercanos en el espacio reducido. 461 462
- Se basa en la teoría de grafos y manifolds para modelar relaciones complejas entre los puntos, siendo especialmente útil en conjuntos de datos con ruido o estructuras no lineales. 463 464 465
- Permite una reducción eficaz incluso en espacios de alta dimensionalidad, manteniendo patrones significativos en los datos. 466 467

A.2 Aplicación de UMAP en el Proyecto 468

UMAP fue utilizado en ambos agentes para reducir la dimensionalidad, con los siguientes resultados: 469 470

- **Agente No Supervisado:** 471
 - Los datos originales tenían 5 dimensiones. 472
 - UMAP redujo los datos a un espacio de 3 dimensiones, lo que permitió visualizar y explorar mejor los patrones intrínsecos de los datos no etiquetados. 473 474
 - Este paso fue crucial para mejorar el rendimiento del agente, ya que las relaciones clave en los datos fueron destacadas mientras que el ruido fue minimizado. 475 476
- **Agente Supervisado:** 477
 - Los datos originales tenían 209 dimensiones, lo que representaba un desafío debido a la alta dimensionalidad y al ruido inherente. 478 479
 - UMAP redujo estos datos a 3 dimensiones, conservando las relaciones relevantes para la tarea de clasificación supervisada. 480 481

- Esta reducción ayudó a evitar problemas relacionados con la "maldición de la dimensionalidad" mejoró la eficiencia del modelo, tanto en el entrenamiento como en la predicción.

A.3 Importancia de la Reducción de Dimensionalidad con UMAP

La elección de UMAP como técnica de reducción de dimensionalidad fue crítica para el éxito de este proyecto debido a las siguientes ventajas:

- **Manejo del ruido:** UMAP filtró eficazmente el ruido presente en los datos originales, algo que no se logró con PCA.
- **Eficiencia computacional:** Trabajar con datos reducidos a 3 dimensiones permitió una reducción significativa en el tiempo y los recursos computacionales necesarios para entrenar los modelos.
- **Mejora de la visualización:** La reducción a 3 dimensiones facilitó la interpretación de los patrones en los datos, especialmente en el agente no supervisado.
- **Preservación de relaciones relevantes:** UMAP conservó las estructuras locales y globales importantes en los datos, asegurando que las transformaciones no degradaran la calidad de la información.

A.4 Conclusión

La implementación de UMAP fue determinante para este proyecto, permitiendo una reducción de dimensionalidad eficiente y efectiva en ambos agentes. Al reducir el espacio de 5 a 3 dimensiones para el agente no supervisado y de 209 a 3 dimensiones para el agente supervisado, se logró mejorar tanto la interpretabilidad como el rendimiento de los modelos, lo que demuestra la importancia de utilizar técnicas avanzadas de reducción de dimensionalidad en aplicaciones prácticas.