

## **Trabajo Final Inteligencia Artificial I año 2024:**

### **Visión Artificial y reconocimiento de voz**

# Trabajo Final

Ingeniería en Mecatrónica – Inteligencia Artificial I – Año 2024

---

[1. Resumen](#)

[2. Introducción](#)

[3. Especificación del agente](#)

[3.1 Tipo de agente](#)

[3.2 Tabla REAS](#)

[3.3 Entorno](#)

[4. Diseño del proyecto](#)

[4.1 Identificación de imágenes](#)

[4.1.1 Base de datos](#)

[4.1.2 Preprocesamiento y segmentación](#)

[4.1.3 Extracción de características](#)

[4.1.4 Aplicación de Kmeans](#)

[4.2 Identificación de audios](#)

[4.2.1 Base de datos](#)

[4.2.2 Preprocesamiento y segmentación](#)

[4.2.3 Aplicación de Knn](#)

## 1. Resumen

Este proyecto aborda el desarrollo de un sistema de clasificación dual que combina reconocimiento de voz y visión artificial para la identificación automática de verduras en un entorno industrial. El sistema implementa dos algoritmos principales: K-Nearest Neighbors (KNN) para el reconocimiento de comandos de voz ("papa", "berenjena", "zanahoria", "camote") y K-means para la clasificación de imágenes de las verduras. La base de datos de entrenamiento incluye 88 muestras de audio de 5 personas diferentes y 70 imágenes de las verduras en distintas posiciones y condiciones de iluminación. Se desarrolló una interfaz gráfica que permite la captura de audio y clasificación de imágenes en tiempo real.

Las pruebas realizadas demostraron una precisión sobresaliente en reconocimiento de voz, con solo un error en la clasificación de "camote". El sistema de visión artificial mostró una clara separación entre clases utilizando características de color en el espacio LAB con resultados perfectos. Los resultados confirman que la combinación de estos algoritmos, junto con un adecuado preprocesamiento de señales y extracción de características, permite crear un sistema robusto y eficiente para la clasificación automática de verduras.

## 2. Introducción

La visión artificial y el reconocimiento de voz son tecnologías que permiten a los sistemas interpretar el entorno mediante señales visuales y auditivas. La visión artificial consiste en el procesamiento de imágenes digitales para reconocer y clasificar objetos. Su objetivo es dividir una imagen en segmentos relevantes, extraer características (como bordes, contornos o texturas) y emplear algoritmos de clasificación para identificar los objetos. Esto se logra mediante técnicas de segmentación y algoritmos como K-means, que permite agrupar los píxeles de una imagen en distintas clases.

El reconocimiento de voz, por otro lado, convierte palabras habladas en datos digitales que los sistemas pueden interpretar. Mediante la extracción de características del sonido, como coeficientes cepstrales (MFCC), y el uso de algoritmos de clasificación como KNN, es posible identificar palabras específicas.

El objetivo principal es lograr una identificación precisa y robusta de las verduras, independientemente de variaciones en la iluminación o posición, permitiendo así la automatización eficiente del proceso de clasificación en un entorno industrial simulado.

## 3. Especificación del agente

**Agente:** Es cualquier cosa capaz de percibir su medioambiente con la ayuda de sensores y actuar en ese medio utilizando actuadores

### 3.1 Tipo de agente

El sistema implementado es un **agente que aprende** ya que posee la capacidad de mejorar su rendimiento a partir de la experiencia adquirida durante la fase de entrenamiento.

### 3.2 Tabla REAS

Agente	Medidas de rendimiento	Entorno	Actuadores	Sensores
El software de control del brazo robótico está diseñado para reconocer comandos de voz e imágenes, permitiendo seleccionar y colocar la verdura solicitada de manera eficiente.	Precisión y consistencia en el reconocimiento de voz e imágenes. Tiempo de respuesta del sistema.	Cinta transportadora con verduras. Zona de trabajo del brazo robótico. Condiciones de iluminación y de contaminación auditiva.	Pantalla de la computadora.	Micrófono para captar comandos de voz. Cámara para tomar fotos de las verduras. Sensor de posición

### 3.3 Entorno

Totalmente Observable: Ya que los sensores del agente proporcionan toda la información relevante necesaria para tomar decisiones óptimas. Esto implica que el agente tiene acceso a todos los datos que describen completamente el estado del entorno en cada momento, lo cual elimina la incertidumbre y permite responder de forma precisa y eficaz en función de la información recibida.

Determinista: Porque ante las mismas entradas (audio o imagen) y bajo las mismas condiciones, el agente siempre tomará las mismas decisiones de clasificación.

Episódico: Porque cada interacción del agente consiste en percibir una verdura y realizar una acción de manera independiente. Los episodios son autónomos y no dependen de acciones anteriores.

# Trabajo Final

Ingeniería en Mecatrónica – Inteligencia Artificial I – Año 2024

Estático: El entorno es estático porque ni las imágenes de las verduras ni las señales de voz cambian mientras el agente procesa la información. El agente analiza cada entrada sin preocuparse por cambios en el entorno mientras actúa.

Discreto: Porque tanto las percepciones como las acciones están claramente definidas y limitadas. El agente sólo recibe comandos de voz específicos (como “papa” o “zanahoria”) y percibe imágenes de un conjunto finito de verduras.

Agente individual: Un solo agente software controla el brazo robótico y toma decisiones basado en la información recibida.

## 4. Diseño del proyecto

### 4.1 Identificación de imágenes

#### 4.1.1 Base de datos

Las imágenes fueron tomadas con un smartphone, utilizando el flash del mismo. Se utilizaron distintas posiciones para las verduras, todas a la misma distancia y con la misma iluminación. Se tomaron 20 fotos por cada verdura (con dos verduras por clase). y luego se realizó una revisión de las fotos en la cual se eliminaron algunas fotos mal tomadas. Una vez obtenida la base de datos, se pasaron las imágenes a la computadora para poder trabajar con éstas.

#### 4.1.2 Preprocesamiento y segmentación

- 1) Carga de la Imagen: La imagen se carga en formato RGB (azul, verde, rojo) utilizando OpenCV.
- 2) Conversión a Espacio de Color LAB: La imagen se convierte del espacio de color RGB al espacio de color LAB. El espacio LAB es útil para análisis de color porque separa la información de luminosidad (L) de la información de color (a y b). Esta conversión es crucial ya que el espacio LAB es perceptualmente uniforme y menos sensible a cambios de iluminación
- 3) Segmentación del Objeto: La imagen se convierte al espacio de color HSV (matiz, saturación, valor), y se crea una máscara que separa el objeto de interés del fondos. La función `cv2.inRange()` crea una máscara binaria, donde los píxeles dentro del rango se vuelven blancos (255) y los demás negros (0). Luego, con `bitwise_not` se invierten los valores, haciendo que el objeto de interés esté en blanco y el fondo en negro.
- 4) Operaciones morfológicas: se aplican operaciones morfológicas para mejorar la máscara. `MORPH CLOSE` ayuda a cerrar agujeros pequeños en el objeto segmentado, y `MORPH OPEN` elimina pequeños puntos de ruido en el fondo. Estas operaciones son esenciales para limpiar la segmentación y hacerla más precisa.

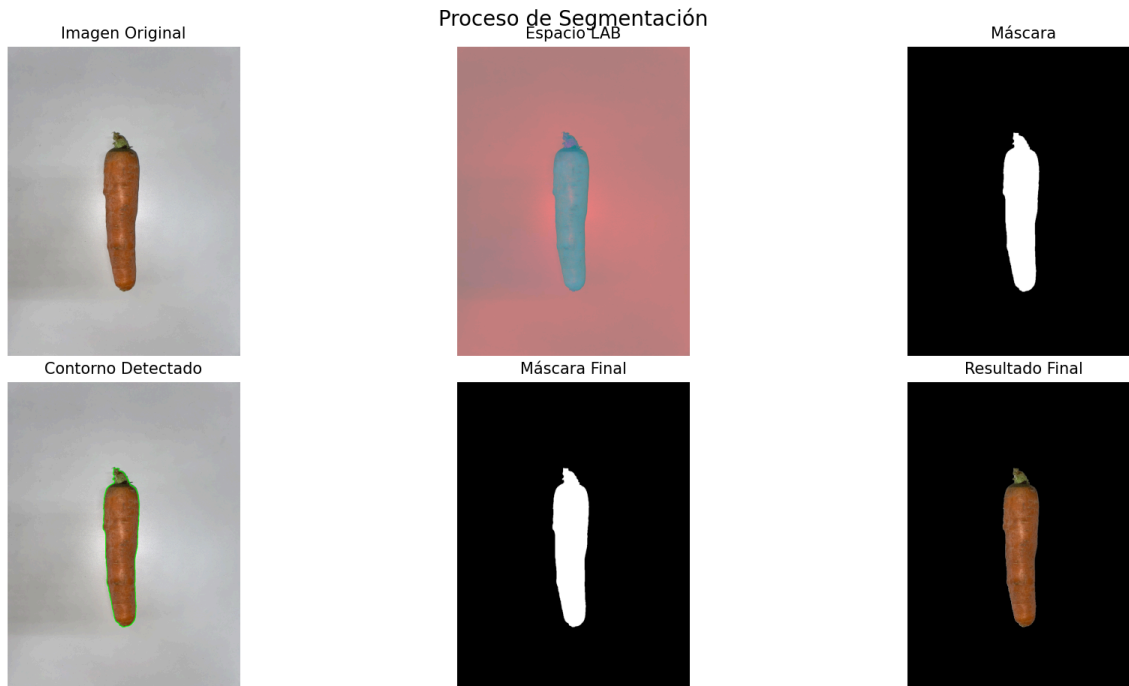
Juan Francisco Huertas Coppo L:12620

Profesora Titular: Dra. Selva S. Rivera

# Trabajo Final

Ingeniería en Mecatrónica – Inteligencia Artificial I – Año 2024

- 5) Detección de Contornos: Se detectan los contornos en la máscara. El contorno más grande se considera el objeto principal de interés (la verdura). `cv2.findContours()` detecta los bordes del objeto, y luego `max()` selecciona el contorno con el área más grande.
- 6) Creación de Máscara Final del Objeto: Se crea una imagen en blanco del mismo tamaño que la original y se dibuja el contorno del objeto en blanco, rellenando el área del objeto. Esto produce una máscara final donde solo el objeto de interés es visible y el resto es negro.



## 4.1.3 Extracción de características

A la hora de clasificar correctamente las verduras, necesité encontrar la combinación de características que mejor identifica a cada verdura, para esto desarrollé un programa que generaba combinaciones iterativas de diferentes tríos de características. Este programa calculaba y visualizaba cada combinación, permitiéndole analizar cómo se agrupaban los datos en el espacio de características.

Al observar los resultados, pude identificar las combinaciones específicas de características que maximizan la separación visual entre los clusters, logrando una representación clara y distintiva de cada grupo. Este proceso iterativo de prueba y observación fue esencial para identificar la mejor configuración de características, asegurando una visualización óptima de los clusters en el análisis.

A modo de observación, descubrí que otra terna que representaba muy bien los clusters era la de perímetro, circularidad y escala de rojo.

Juan Francisco Huertas Coppo L:12620  
Profesora Titular: Dra. Selva S. Rivera

# Trabajo Final

Ingeniería en Mecatrónica – Inteligencia Artificial I – Año 2024

A través de este proceso iterativo, identifiqué que la combinación de características  $L$ ,  $a$ , y  $b$  era la más efectiva para representar los clusters de manera clara y precisa. En esta terna:

- $L$  representa la luminosidad o el brillo de la imagen.
- $a$  indica la variación de verde a rojo.
- $b$  refleja la variación de azul a amarillo.

## 4.1.4 Aplicación de Kmeans

### 4.1.4.1 Kmeans tradicional

El algoritmo K-means es una técnica de agrupamiento no supervisado que particiona  $n$  observaciones en  $k$  grupos. En su implementación tradicional:

1. **Inicialización:**
  - Se seleccionan  $k$  puntos aleatorios como centroides iniciales
  - Cada centroide representa el "centro" de un cluster
2. **Iteración Principal:**
  - **Asignación:** Cada punto se asigna al centroide más cercano usando distancia euclidiana
  - **Actualización:** Los centroides se recalculan como el promedio de todos los puntos asignados
  - Este proceso se repite hasta la convergencia o un número máximo de iteraciones
3. **Convergencia:**
  - Se alcanza cuando los centroides no cambian significativamente entre iteraciones
  - O cuando se alcanza el número máximo de iteraciones establecido

### 4.1.4.1 Kmeans personalizado

En nuestro sistema, hemos adaptado K-means para la clasificación supervisada de verduras con las siguientes modificaciones clave:

1. **Inicialización Informada:** En lugar de inicialización aleatoria, calculamos los centroides iniciales basados en el promedio de las muestras de cada clase
2. **Cálculo de Distancias:** Calcula distancias euclidianas entre muestras y centroides utilizando vectorización de NumPy.

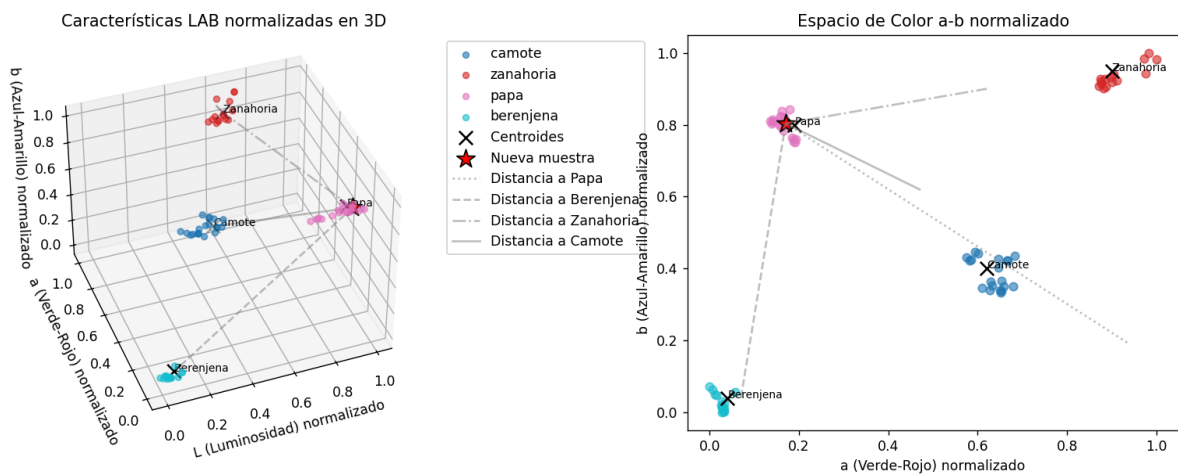
Juan Francisco Huertas Coppo L:12620  
Profesora Titular: Dra. Selva S. Rivera

# Trabajo Final

Ingeniería en Mecatrónica – Inteligencia Artificial I – Año 2024

## 3. Proceso de Entrenamiento: Entrenamiento supervisado con evaluación de precisión

En resumen, se calculan centroides fijos una sola vez usando las etiquetas conocidas, garantizando una clasificación consistente. Se emplea supervisión inicializando los centroides con las etiquetas reales, donde cada centroide representa una clase de verdura y la precisión se evalúa con las etiquetas verdaderas. Para la predicción, las nuevas muestras se asignan al centroide más cercano. Las características se normalizan para igualar el peso de las dimensiones (L, a, b) y no pierde precisión ante variaciones de iluminación. Esta combinación híbrida aprovecha la simplicidad del K-means y elementos supervisados, creando un clasificador eficiente y robusto para identificar verduras en LAB.



## 4.2 Identificación de audios

### 4.2.1 Base de datos

Para crear la base de datos de audios, desarrollé un código que recopila y organiza los registros de todas las personas necesarias para el proyecto. Este código permite grabar y almacenar los audios de cada individuo de forma estructurada, asegurando que cada muestra quede correctamente etiquetada y almacenada en la base de datos. Esto facilitó el proceso de acceso y análisis de los datos de entrenamiento en las siguientes etapas del proyecto

### 4.2.2 Preprocesamiento

#### 4.2.2.1 Etapas del Preprocesamiento

El preprocesamiento de audio es crucial para obtener señales limpias y estandarizadas antes de la extracción de características. En tu implementación, se realizan las siguientes etapas clave:



# Trabajo Final

---

Ingeniería en Mecatrónica – Inteligencia Artificial I – Año 2024

## 1. Normalización inicial:

Conversión a formato float32 y escalado a rango  $[-1, 1]$ . Y conversión a mono si la entrada es estéreo, lo que ayuda a manejar diferentes formatos de entrada de manera uniforme

## 2. Resampleo:

Estandarización a 44.1 kHz (frecuencia de muestreo objetivo), para asegurar consistencia en el procesamiento independiente del dispositivo de grabación

## 3. Filtrado Pasa-Banda:

Rango de 80 Hz a 6.5 kHz, optimizado para el rango de voz humana, nos ayuda a eliminar ruido de baja y alta frecuencia

## 4. Eliminación de Silencios:

Detección de segmentos activos mediante energía, con un umbral adaptativo basado en RMS. solo se retiene el segmento más significativo.

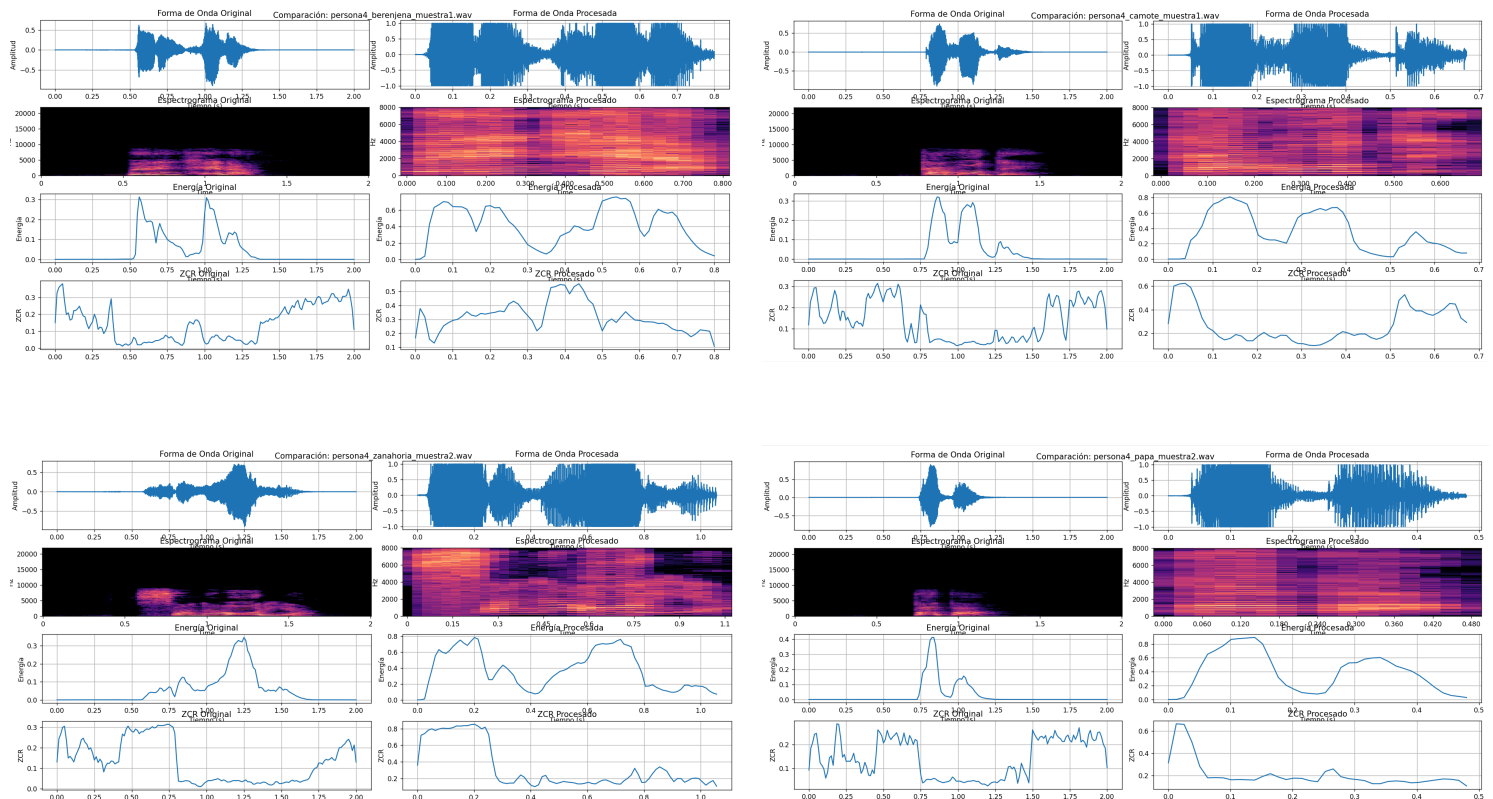
## 5. Normalización Final:

Ajuste de amplitud al 90% del máximo, lo que nos ayuda a evitar saturación mientras se mantiene buena relación señal-ruido

Cabe mencionar que para encontrar la mejor configuración para el preprocesamiento, se utilizaron algoritmos que permiten encontrar los mejores valores para configurar el preprocesador

# Trabajo Final

Ingeniería en Mecatrónica – Inteligencia Artificial I – Año 2024



## 4.2.2.2 Antes y después del procesamiento de los audios

## 4.1.3 Extracción de características

La extracción de características de señales de audio es un paso fundamental en el procesamiento de voz, donde se busca transformar la señal acústica en un conjunto de medidas numéricas que capturen la esencia de lo que se está diciendo. Es crucial porque reduce la dimensionalidad de los datos y extrae información relevante que permite al sistema discriminar entre diferentes palabras.

En este proyecto, el proceso de extracción se realiza en dos niveles:

1. Segmentación temporal: La señal se divide en segmentos de igual duración, permitiendo capturar la evolución temporal del sonido.
2. Extracción por segmento: Para cada segmento se calculan múltiples características que capturan diferentes aspectos de la señal:

Características Totales Extraídas:

- 13 MFCCs (media y std) = 26 características
- 13 MFCC deltas (media y std) = 26 características
- 13 MFCC delta2 (media y std) = 26 características

Juan Francisco Huertas Coppo L:12620  
Profesora Titular: Dra. Selva S. Rivera

# Trabajo Final

Ingeniería en Mecatrónica – Inteligencia Artificial I – Año 2024

- Características espectrales (centroide, ancho banda, rolloff) con media y std = 6 características
- ZCR (media, max, std) = 3 características
- Amplitud y envolvente (5 métricas)

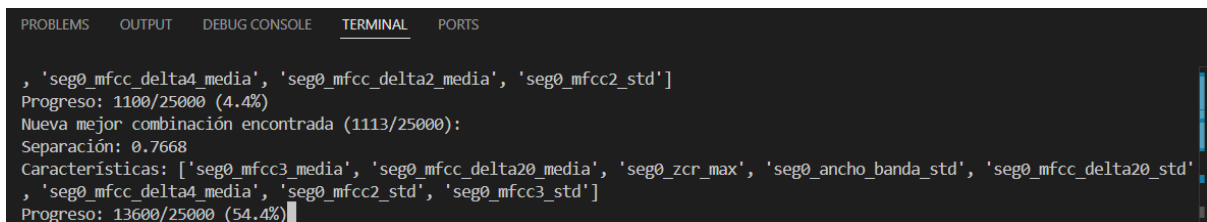
Lo que nos deja un total de 92 características. De las cuales solo se utilizaron 8, las cuales se identificaron mediante un proceso sistemático de selección de características, utilizando técnicas como análisis de correlación y algoritmos de importancia de características.

Características Seleccionadas para Clasificación:

- mfcc3\_media: Coeficiente MFCC #3, relacionado con la forma del tracto vocal
- ancho\_banda\_std: Variabilidad en la distribución de frecuencias
- zcr\_max: Pico máximo de cruces por cero, útil para detectar consonantes
- mfcc\_delta20\_media: Tasa de cambio del MFCC #20, captura transiciones
- mfcc2\_std: Variabilidad del segundo coeficiente MFCC
- mfcc5\_media: Coeficiente MFCC #5, características tonales
- mfcc\_delta5\_std: Variabilidad en la tasa de cambio del MFCC #5
- mfcc\_delta3\_std: Variabilidad en la tasa de cambio del MFCC #3

Estas 8 características fueron seleccionadas por su capacidad discriminativa entre las diferentes palabras del vocabulario.

Combinador.py en funcionamiento:



```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
, 'seg0_mfcc_delta4_media', 'seg0_mfcc_delta20_media', 'seg0_mfcc2_std']
Progreso: 1100/25000 (4.4%)
Nueva mejor combinación encontrada (1113/25000):
Separación: 0.7668
Características: ['seg0_mfcc3_media', 'seg0_mfcc_delta20_media', 'seg0_zcr_max', 'seg0_ancho_banda_std', 'seg0_mfcc_delta20_std',
, 'seg0_mfcc_delta4_media', 'seg0_mfcc2_std', 'seg0_mfcc3_std']
Progreso: 13600/25000 (54.4%)
```

## 4.2.3 Aplicación de Knn

El algoritmo de clasificación funciona en dos etapas:

### 1. Primera Etapa (Clasificación General):

Analiza solo el primer segmento (de tres) del audio. Utiliza las 8 características clave. Encuentra los K vecinos más cercanos ( $k=7$  en este caso), para luego predecir la palabra (papa, camote, berenjena o zanahoria)

### 2. Segunda Etapa (Verificación Papa/Camote):

Juan Francisco Huertas Coppo L:12620  
Profesora Titular: Dra. Selva S. Rivera

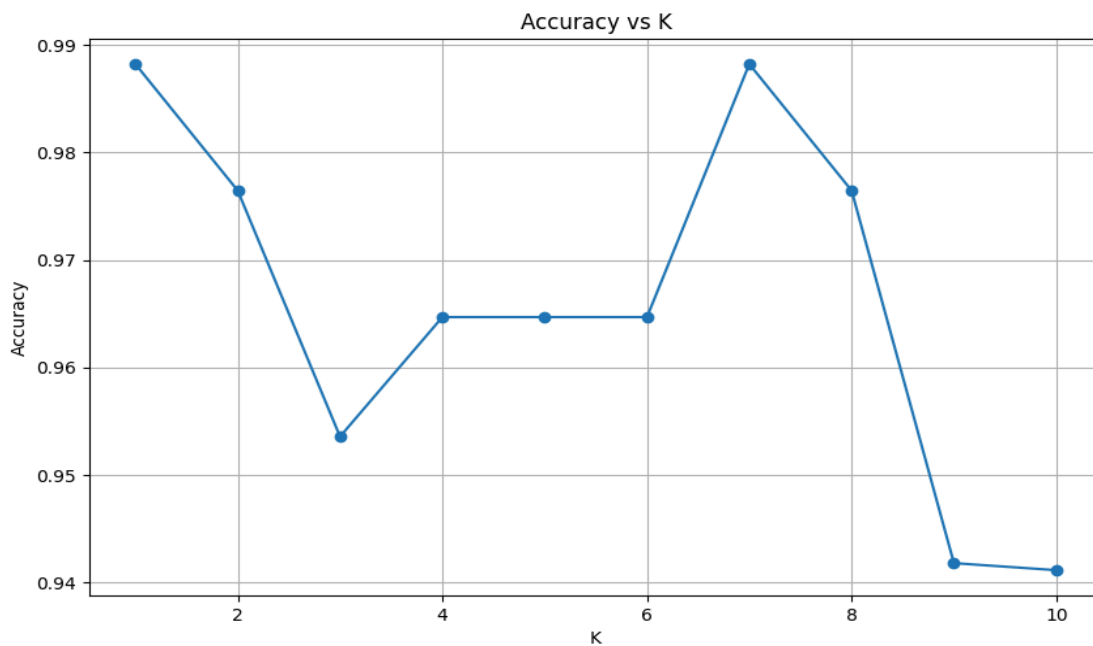
# Trabajo Final

Ingeniería en Mecatrónica – Inteligencia Artificial I – Año 2024

Solo si la predicción inicial fue "papa" o "camote", analiza el audio completo (sin segmentar) usando las mismas 8 características. Nueva búsqueda de 7 vecinos más cercanos. Y finalmente realiza la clasificación final entre papa y camote

Esta estrategia de dos etapas mejora la precisión en la distinción entre papa y camote, que son las palabras más similares fonéticamente si analizamos el primer tercio de los audios. El sistema visualiza cada etapa mediante PCA 3D para mostrar la distribución de las muestras y los vecinos encontrados. La reducción de dimensionalidad con PCA ayuda a optimizar el proceso, haciendo que el modelo sea más rápido y efectivo al reducir el ruido en los datos, facilitando la tarea de identificación por KNN

Durante la fase de desarrollo se realizaron pruebas sistemáticas variando el número de vecinos (k) para optimizar la clasificación. Los mejores resultados se obtuvieron con  $k=1$  y  $k=7$ :



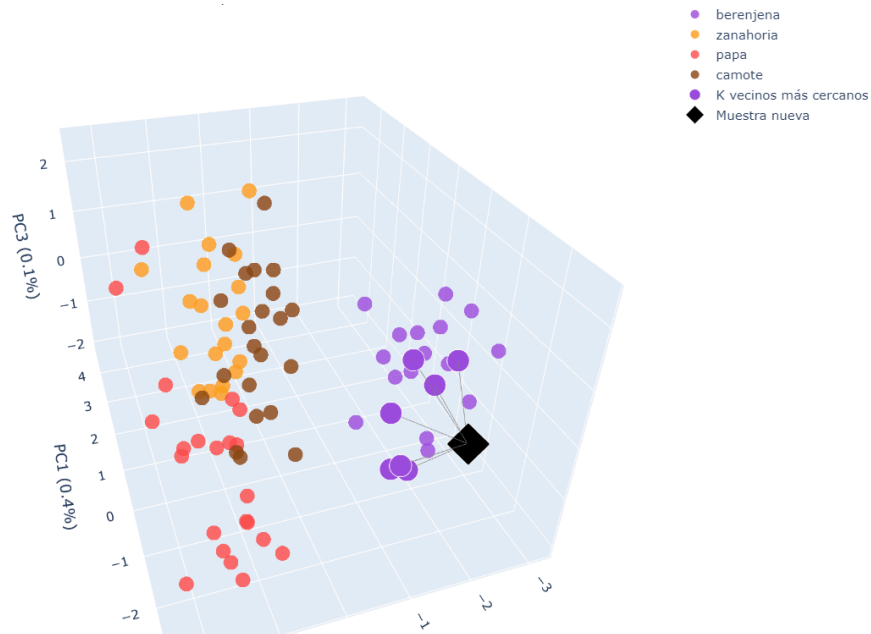
Juan Francisco Huertas Coppo L:12620  
Profesora Titular: Dra. Selva S. Rivera

# Trabajo Final

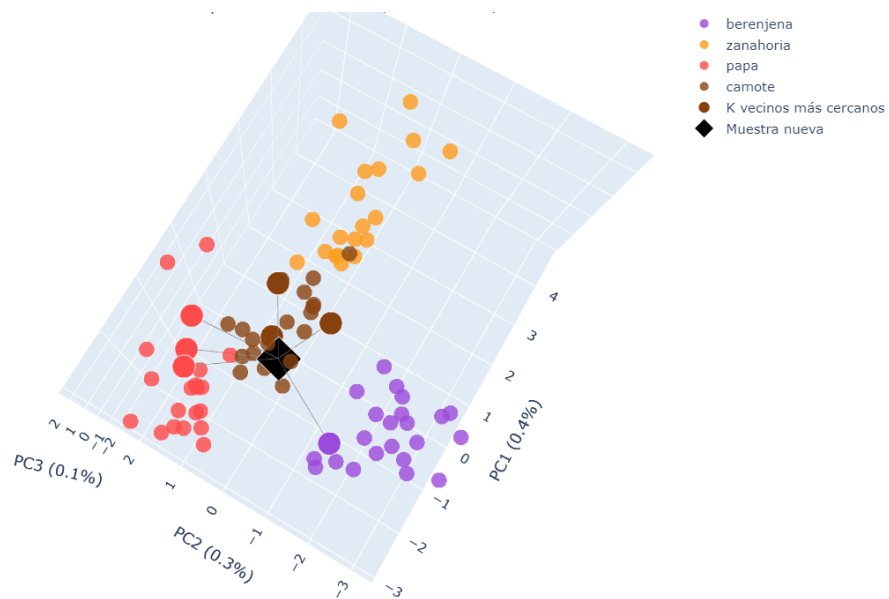
Ingeniería en Mecatrónica – Inteligencia Artificial I – Año 2024

Se adjuntan algunos resultado mostrados en PCA para un mejor entendimiento del funcionamiento:

Usando la palabra berenjena como muestra:



Y ahora usando la palabra camote con la segunda verificación:

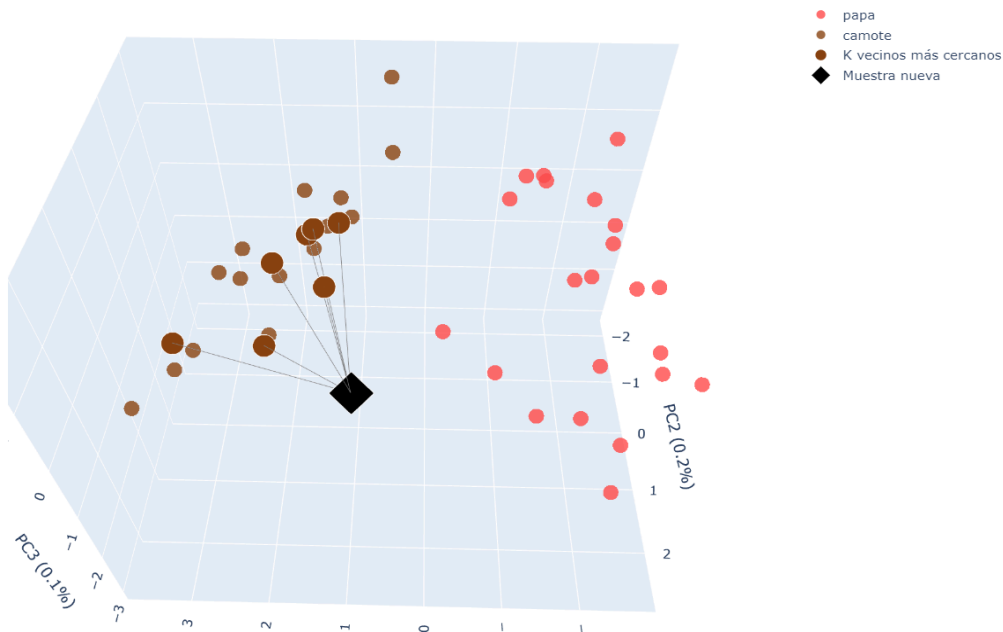


Juan Francisco Huertas Coppo L:12620  
Profesora Titular: Dra. Selva S. Rivera

# Trabajo Final

Ingeniería en Mecatrónica – Inteligencia Artificial I – Año 2024

Desempeño entre papa y camote:



## 5. Resultados

### 5.1 Análisis del Dataset de Clasificación de Verduras

#### 5.1.1 Dataset de Audio

El dataset de audio está compuesto por grabaciones de 5 personas diferentes pronunciando nombres de verduras, con 4 voces femeninas y 1 voz masculina. En total se recolectaron 88 grabaciones distribuidas de la siguiente manera:

- 28 grabaciones de "berenjena"
- 26 grabaciones de "camote"
- 27 grabaciones de "papa"
- 26 grabaciones de "zanahoria"

#### 5.1.2 Dataset de Imágenes

Para el entrenamiento del clasificador de imágenes se utilizó un dataset de 70 imágenes distribuidas de la siguiente manera:

- 20 imágenes de camote
- 20 imágenes de papa
- 15 imágenes de zanahoria
- 15 imágenes de berenjena

Juan Francisco Huertas Coppo L:12620  
Profesora Titular: Dra. Selva S. Rivera

# Trabajo Final

Ingeniería en Mecatrónica – Inteligencia Artificial I – Año 2024

La distribución balanceada en ambos datasets ayuda a minimizar el sesgo en el entrenamiento de los modelos de clasificación. Las grabaciones de audio incluyen variedad en la pronunciación debido a las diferentes voces y géneros, mientras que las imágenes presentan diversidad en los ángulos de captura.

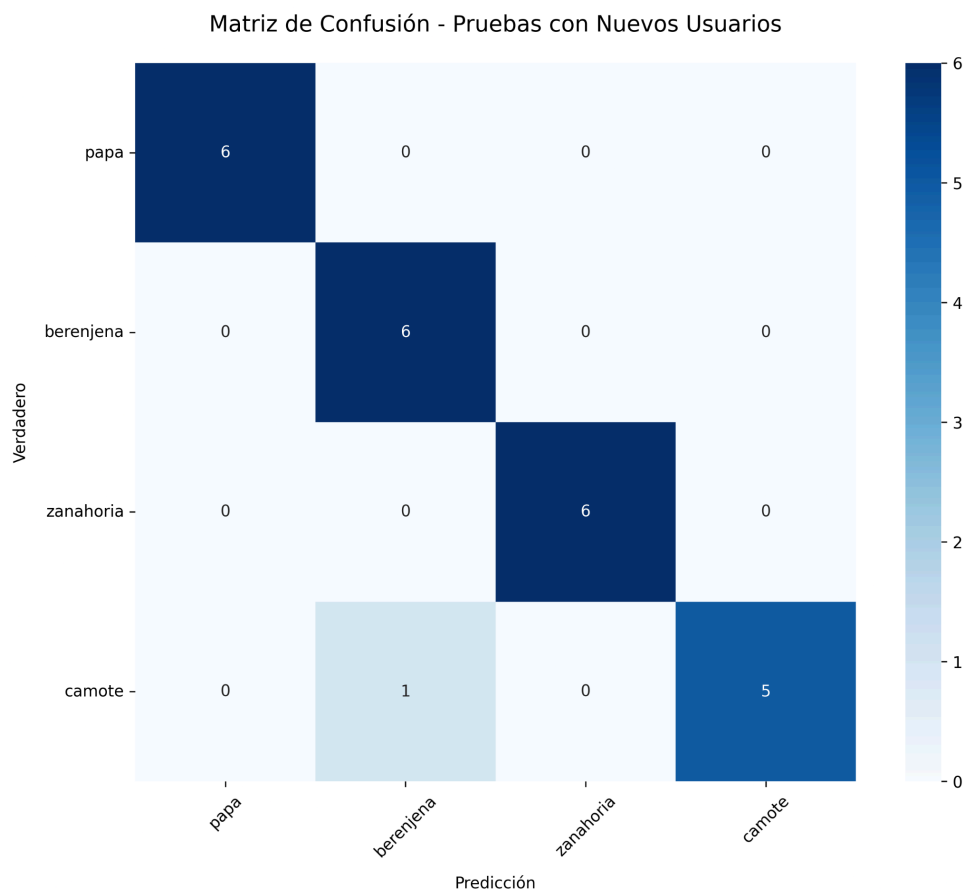
## 5.2 Pruebas con nuevas imágenes y usuarios

### 5.2.1 Pruebas de audio

Se realizaron pruebas del sistema con 6 personas que no formaban parte de la base de datos de entrenamiento. Cada participante grabó los nombres de las cuatro verduras (papa, berenjena, zanahoria y camote), sumando un total de 24 clasificaciones.

Los resultados fueron sobresalientes, con una precisión del 95.83% (23 clasificaciones correctas de 24). El sistema mostró un rendimiento casi perfecto, con un único error de clasificación: en uno de los casos, cuando un participante pronunció "camote", el sistema lo clasificó como "berenjena". Para todos los demás casos (23 grabaciones), el sistema clasificó correctamente las verduras pronunciadas.

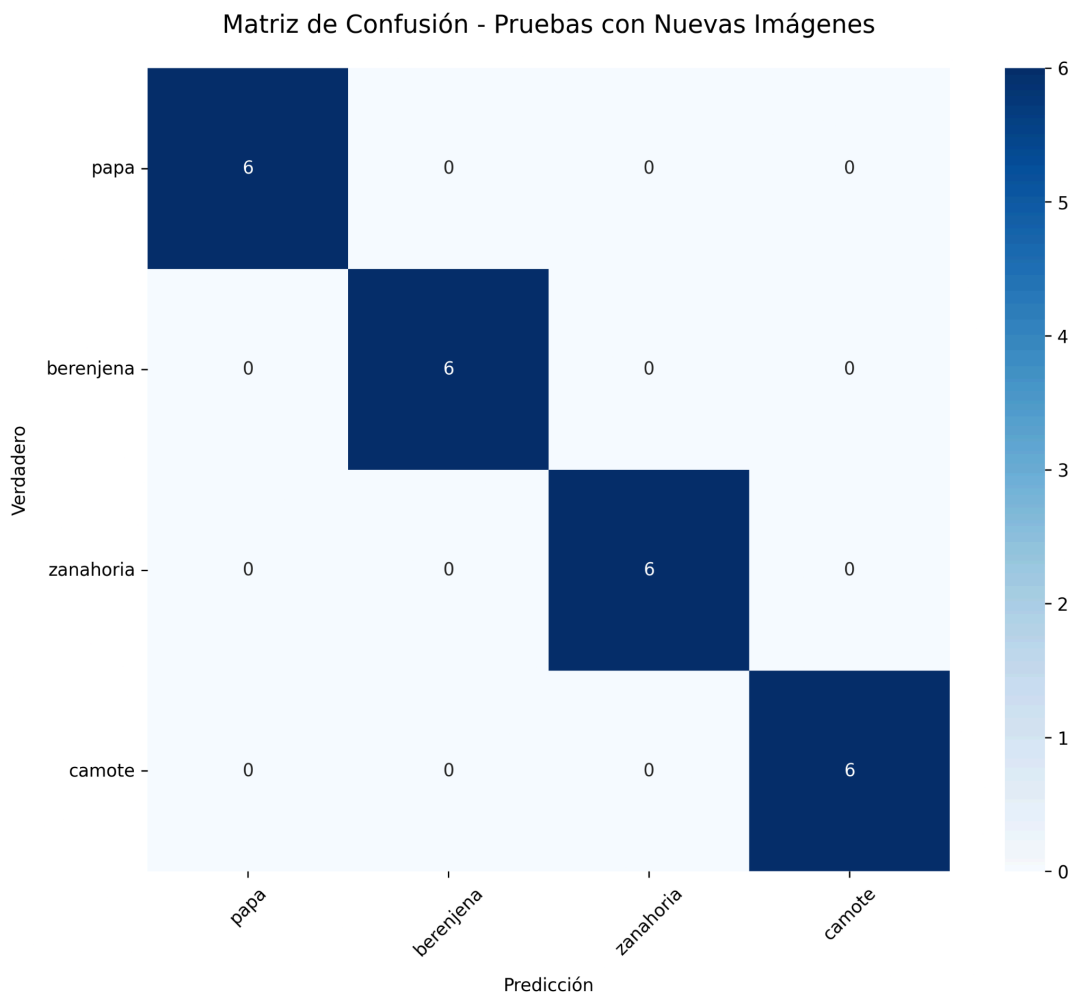
La matriz de confusión generada muestra claramente este comportamiento, donde solo se observa un error en la clasificación de camote, manteniendo una precisión perfecta para las demás categorías.



Juan Francisco Huertas Coppo L:12620  
Profesora Titular: Dra. Selva S. Rivera

## 5.2.1 Pruebas de imágenes

Se realizó una evaluación adicional del sistema utilizando un conjunto de prueba compuesto por 24 imágenes nuevas (6 fotos por cada tipo de verdura). Las fotografías se



tomaron considerando diferentes variables para probar la robustez del sistema:

- Variaciones en el tipo/forma de cada verdura
- Diferentes condiciones de iluminación

Juan Francisco Huertas Coppo L:12620  
Profesora Titular: Dra. Selva S. Rivera



- Distintas distancias de captura

El clasificador de imágenes demostró un rendimiento excepcional, alcanzando una precisión del 100% al clasificar correctamente las 24 imágenes de prueba.

## 5.3 Resultados Generales

Los resultados obtenidos en la fase de prueba demuestran un rendimiento excepcional del sistema de clasificación dual. En la clasificación por audio, con 24 pruebas realizadas por 6 personas diferentes no incluidas en el dataset de entrenamiento, se alcanzó una precisión del 95.83%. Por su parte, el clasificador de imágenes mostró un rendimiento perfecto, con una precisión del 100% en las 24 nuevas imágenes de prueba (6 por cada verdura), incluso con variaciones en iluminación, distancia y características de las verduras.

Considerando ambos sistemas, se alcanzó una precisión global del 97.92% (47 clasificaciones correctas de 48 totales), lo que demuestra la robustez y confiabilidad del sistema.

Estos resultados sugieren que el sistema es altamente viable para su implementación en un entorno real, donde las condiciones de captura tanto de audio como de imagen pueden ser aún más controladas mediante un diseño apropiado.

## 6. Código

La implementación completa del sistema se realizó en Python 3.9, un lenguaje de programación versátil y ampliamente utilizado en el campo de la inteligencia artificial. Para el desarrollo se utilizaron diversas librerías especializadas que facilitaron el procesamiento de señales de audio, manipulación de imágenes, análisis de datos y creación de interfaces gráficas. Las principales librerías externas utilizadas fueron:

- Procesamiento de Audio:
  - sounddevice: Grabación de audio en tiempo real
  - librosa: Procesamiento y análisis de señales de audio
  - scipy.io wavfile: Lectura/escritura de archivos WAV
  - queue: Manejo de buffer de audio
- Procesamiento de Imágenes:
  - cv2 (OpenCV): Procesamiento y manipulación de imágenes
  - PIL: Manejo de imágenes en la interfaz
  - skimage: Extracción de características de textura
- Análisis de Datos:
  - numpy: Operaciones matemáticas y manejo de arrays
  - pandas: Manipulación y análisis de datos
  - scikit-learn: Métricas de evaluación
- Visualización:

Juan Francisco Huertas Coppo L:12620  
Profesora Titular: Dra. Selva S. Rivera

# Trabajo Final

Ingeniería en Mecatrónica – Inteligencia Artificial I – Año 2024

- matplotlib: Gráficos y visualizaciones
- seaborn: Visualización de datos estadísticos
- plotly: Gráficos interactivos 3D
- Interfaz Gráfica:
  - tkinter: Creación de la interfaz de usuario

## 6. Conclusiones

El proyecto logró implementar exitosamente un sistema de clasificación dual que combina reconocimiento de voz y visión artificial, alcanzando una precisión sobresaliente en ambas modalidades. Las pruebas con usuarios nuevos demostraron una tasa de acierto del 95.83% en reconocimiento de voz, mientras que el sistema de visión artificial mostró gran robustez en la clasificación de imágenes mediante la extracción de características en el espacio de color LAB.

La implementación demostró que algoritmos relativamente simples como KNN y K-means pueden ofrecer resultados excepcionales cuando se combinan con una adecuada selección y procesamiento de características. Aunque es importante recalcar que K-means, no se implementa habitualmente con la mejora que hemos hecho, por lo que para problemas de este tipo, resulta más efectivo usar KNN.

Como mejoras futuras, se sugiere expandir la base de datos de entrenamiento para incluir mayor variabilidad en las condiciones de captura de imagen y grabación de audio. También sería valioso implementar técnicas de preprocesamiento más robustas para manejar diferentes condiciones de iluminación y ruido ambiental. Se podría trabajar mejorando aún más los casos en los que aparecen distintas clases de vecinos en KNN.

El sistema actual sienta las bases para una implementación práctica en un entorno industrial de clasificación automatizada de verduras.

## 7. Bibliografía y/o referencia

Procesamiento de imágenes

▶ Image Processing with OpenCV and Python

Procesamiento de audios y sus señales

▶ Audio Signal Processing for Machine Learning

Extracción de características:

[https://joserzapata.github.io/courses/mineria-audio/extraccion\\_caracteristicas/](https://joserzapata.github.io/courses/mineria-audio/extraccion_caracteristicas/)

Qué son los clusters:

<https://www.ibm.com/topics/clustering>




Juan Francisco Huertas Coppo L:12620  
Profesora Titular: Dra. Selva S. Rivera

# Trabajo Final

---

Ingeniería en Mecatrónica – Inteligencia Artificial I – Año 2024

KNN:

   KNN ALGORITMO en Español (K VECINOS MÁS CERCANOS) de CLASIFIC...

IA generativa:

<https://claude.ai/>

## 8. Repositorio de Github del proyecto

<https://github.com/Arguur/IA-1>

Juan Francisco Huertas Coppo L:12620  
Profesora Titular: Dra. Selva S. Rivera