Benemérita Universidad Autónoma de Puebla

Facultad de Ciencias Físico Matemáticas

Reconocimiento de lenguaje mediante un clasificador multiclase SoftMax

Tesis presentada al

Colegio de Física

Como requisito parcial para obtener el grado de

Licenciado en Física Aplicada

Por

Alexis Tomas Cruz

Director de Tesis

Dr. Juan Castillo Mixcoatl

Puebla, Pue.

Junio 2024

Título: Reconocimiento de lenguaje mediante un clasificador multiclase SoftMax

Estudiante: Alexis Tomas Cruz.

Comité

_____
Dr. Muñoz Aguirre Severino
Presidente

_____
Dra. Beltrán Pérez Georgina
Secretaria

_____
Dr. Velázquez Castro Jorge
Vocal

_____
Dr. Robledo Sánchez Carlos Ignacio
Vocal

_____
Dr. Castillo Mixcóatl Juan
Asesor

Agradecimientos

Este trabajo está dedicado a todas las personas que han formado parte de mi vida académica y personal, brindándome su apoyo incondicional y motivación en cada paso del camino.

A mis padres, por su amor inagotable, sacrificio y por enseñarme el valor del esfuerzo y la perseverancia. Sin su guía y confianza, este logro no habría sido posible.

A mis hermanos, por su constante ánimo y por ser mi fuente de inspiración. Gracias por creer en mí y por estar siempre a mi lado.

A mis amigos, por su comprensión y por los momentos compartidos que me ayudaron a mantener el equilibrio y la alegría durante este arduo proceso.

A mis profesores y mentores, quienes con su sabiduría y dedicación han dejado una huella imborrable en mi formación académica y profesional. Sus enseñanzas y consejos han sido fundamentales para alcanzar esta meta.

A todos aquellos que, de una manera u otra, han contribuido a mi crecimiento personal y académico, les expreso mi más sincero agradecimiento.

Contenido

Resumen

En esta tesis se muestran los conceptos básicos e introductorios a la inteligencia artificial, en particular al Machine Learning, y cómo ésta nos puede ayudar a analizar un conjunto de datos (en este caso archivos de audio) para la clasificación e identificación de estos archivos de audio.

En este trabajo se abordarán las bases teóricas del funcionamiento de una neurona natural y una neurona artificial. Se explicará el proceso básico de un clasificador binario, el cual nos permite saber si un dato pertenece o no a cierta clase. Ampliando este concepto veremos un clasificador multiclase (SoftMax) que nos permite saber si un dato pertenece a alguna de las n clases establecidas.

Veremos cómo el error en nuestro modelo es un indicativo del correcto o no funcionamiento del clasificador SoftMax. Así mismo, entenderemos el funcionamiento de PCA (Análisis de Componentes Principales por sus siglas en inglés) y la importancia de éste en el tratamiento de los archivos.

Posteriormente se estudia el uso de distintas herramientas para el procesamiento de los archivos de audio y sus diferentes comportamientos.

Por último, analizaremos cómo al cambiar ciertos parámetros, los resultados obtenidos mediante el clasificador multiclase SoftMax, son malos, aceptables, buenos o tienen sobreajuste.

Objetivos

El objetivo general de este trabajo es el siguiente:

- El objetivo general de este trabajo es el reconocimiento del habla humana mediante un clasificador multiclase SoftMax.

Los objetivos particulares de este trabajo son los siguientes:

- Desarrollo e implementación de un programa para la recopilación de archivos de audio.
- Desarrollo e implementación de un programa para el procesamiento de las señales de audio que permita la creación de la llamada matriz de características.
- Desarrollo e implementación de un programa para la aplicación de la transformada de Fourier y de análisis de espectrograma para cada señal de audio presente en la matriz de características. Después de estas técnicas el programa aplicara el análisis de componentes principales y se determinara el número de componentes principales para formar una nueva matriz de características.
- Desarrollo de un programa para la implementación de un clasificador multiclase SoftMax con 10 clases diferentes (dígitos en español).

Capítulo 1: Fundamentos del Machine Learning

El aprendizaje automático, o machine learning, es una rama de la inteligencia artificial que se basa en el desarrollo de algoritmos y modelos que permiten a las computadoras aprender y tomar decisiones a partir de datos, sin ser programadas explícitamente. Su objetivo es capacitar a las máquinas para que puedan reconocer patrones, hacer predicciones y tomar decisiones basadas en la información disponible.

El proceso de aprendizaje automático consta de varias etapas. En primer lugar, se requiere un conjunto de datos de entrenamiento, que consiste en ejemplos o instancias previamente etiquetadas. Estos datos se dividen en dos partes: una para el entrenamiento del modelo y otra para su evaluación y prueba. Durante el entrenamiento, el modelo utiliza algoritmos para analizar los datos y ajustar sus parámetros internos con el fin de encontrar patrones y relaciones entre las variables. Una vez que el modelo ha sido entrenado, se puede utilizar para hacer predicciones o tomar decisiones sobre nuevos datos.

Existen diferentes tipos de algoritmos de aprendizaje automático, siendo los más comunes los algoritmos supervisados y no supervisados. Los algoritmos supervisados se utilizan cuando se dispone de datos de entrenamiento etiquetados, es decir, se conocen las salidas deseadas para cada instancia. Estos algoritmos buscan encontrar la relación entre las características de entrada y las salidas esperadas, lo que permite realizar predicciones precisas sobre nuevos datos. Por otro lado, los algoritmos no supervisados se utilizan cuando no se tienen etiquetas en los datos de entrenamiento. Estos algoritmos buscan encontrar patrones y estructuras ocultas en los datos, como grupos o “clusters”, sin la necesidad de conocer las salidas esperadas.

El aprendizaje automático tiene numerosas aplicaciones en diversos campos. Por ejemplo, en el campo de la visión por computadora, se utiliza para reconocer objetos y caras en imágenes. En el campo del procesamiento del lenguaje natural, se utiliza para la traducción automática, el análisis de sentimientos y la generación de texto. También se utiliza en áreas como la medicina, las finanzas, la logística y el marketing, entre otros.

Sin embargo, el aprendizaje automático no está exento de desafíos y consideraciones. Uno de los desafíos clave es la calidad y cantidad de los datos de entrenamiento. Un conjunto de datos insuficiente o sesgado puede llevar a un rendimiento deficiente del modelo. Además, es importante considerar la interpretabilidad de los modelos, es decir, la capacidad de comprender y explicar cómo se toman las decisiones. En algunos casos, los modelos de aprendizaje automático pueden ser cajas negras difíciles de entender, lo que puede plantear problemas éticos y de responsabilidad.

En resumen, el aprendizaje automático es una disciplina que permite a las computadoras aprender y tomar decisiones basadas en datos. Utilizando algoritmos y modelos, se busca encontrar patrones y relaciones en los datos de entrenamiento para realizar predicciones y tomar decisiones sobre nuevos datos. Si bien tiene numerosas aplicaciones y beneficios, también presenta desafíos y consideraciones importantes que deben tenerse en cuenta.

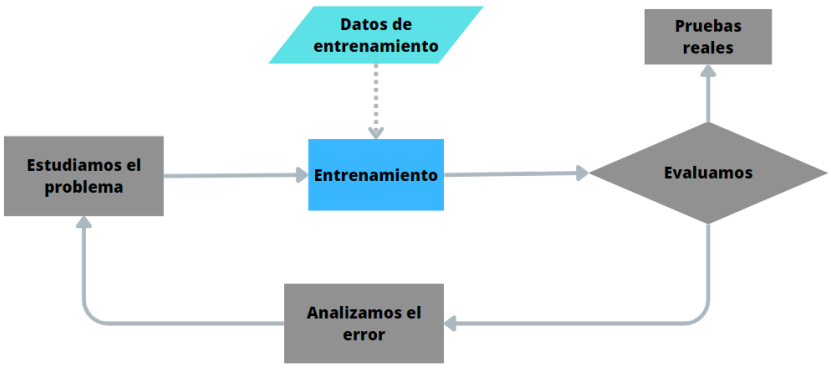

Fig. 1.1 Diagrama de flujo de basico del funcionamiento del algoritomo de Machine Learning. Se entrena el algoritmo, se analiza error y se actualiza hasta que nuestro error sea pequeño sin llegar al sobreajuste.

Una de las tareas de aprendizaje supervisado más comunes son las regresiones (predicciones de valores), y los clasificadores (clasificador de clases).

Veremos un primer algoritmo de aprendizaje automático para clasificación: el perceptrón.

1.1 Neurona y perceptrón

Una neurona es la unidad básica del sistema nervioso, tanto en el sistema nervioso central (cerebro y médula espinal) como en el sistema nervioso periférico. Las neuronas son células especializadas en la transmisión de señales eléctricas y químicas, permitiendo la comunicación en el cuerpo y el procesamiento de información en el sistema nervioso (Aurélien, 2019).

Las partes principales de una neurona son las siguientes y se muestran en la Fig. 1.2:

- Cuerpo celular (soma): El cuerpo celular es la parte central de la neurona. Contiene el núcleo, que alberga el material genético de la célula. Muchos de los procesos metabólicos esenciales de la célula ocurren en el soma.
- Dendritas: Las dendritas son extensiones ramificadas que se extienden desde el cuerpo celular. Su función principal es recibir señales de otras neuronas o células y transmitirlas hacia el cuerpo celular. Las dendritas actúan como antenas que captan información.
- Axón: El axón es una prolongación larga y delgada que se origina en el cuerpo celular. Su función es transmitir señales eléctricas desde el cuerpo celular hacia las estructuras de destino, como otras neuronas o células de tejidos efectoras (como músculos o glándulas).
- Botones sinápticos: Al final del axón, las neuronas tienen estructuras llamadas botones sinápticos o terminales axónicos. Estos botones se comunican con otras neuronas o células mediante sinapsis, que son conexiones especializadas donde se liberan neurotransmisores para transmitir señales a la célula receptora.

- Vaina de mielina: En algunas neuronas, el axón está rodeado por una vaina de mielina, que es una capa de células gliales que aíslan y aceleran la transmisión de las señales eléctricas a lo largo del axón.
- Nódulos de Ranvier: En las neuronas con mielina, los nódulos de Ranvier son pequeñas brechas en la vaina de mielina que permiten que las señales eléctricas salten rápidamente de un nodo a otro, acelerando la conducción de impulsos nerviosos.

En resumen, una neurona es una célula especializada en la comunicación y transmisión de señales en el sistema nervioso. Las dendritas reciben información, el cuerpo celular la integra y, si se alcanza un umbral, genera un impulso eléctrico que se transmite a través del axón hacia otras células a través de las sinapsis. Esta comunicación es fundamental para el funcionamiento del sistema nervioso y el procesamiento de información en el cuerpo.

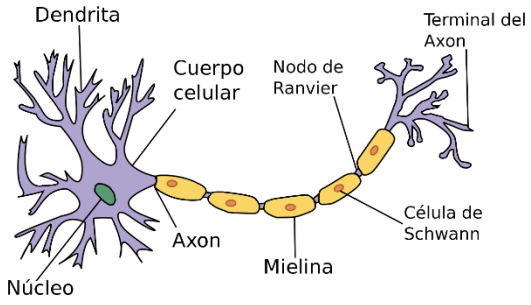

Fig. 1.2. Ejemplo de una neurona biológica típica y su composición general.

Un perceptrón o neurona artificial es una unidad básica de procesamiento en las redes neuronales artificiales, un concepto fundamental en el campo de la inteligencia artificial y el aprendizaje automático. Fue propuesto por Frank Rosenblatt en 1957 y se inspira en el funcionamiento de las neuronas biológicas en el cerebro.

Un perceptrón toma una serie de entradas (números) ponderadas por ciertos pesos y las suma. Luego, aplica una función de activación a esta suma ponderada para producir una salida. La función de activación típicamente utilizada es una función escalón, que produce un valor de 1 si la suma ponderada supera un cierto umbral y

0 en caso contrario. Matemáticamente, el proceso del perceptrón se puede expresar de la siguiente manera:

Salida = 1 si (w_1 * $entrada_1$ + w_2 * $entrada_2$ + ... + w_n * $entrada_n$) >= umbral

Salida = 0 si (w_1 *$entrada_1$ + w_2 * $entrada_2$ + ... + w_n *$entrada_n$) < umbral

Donde:

- $w_1, w_2, \ldots, w_n$ son los pesos asociados a cada entrada.
- $entrada_1, entrada_2, \ldots, entrada_n$son los valores de entrada.
- "umbral" es un valor umbral que determina si la neurona se activa o no.

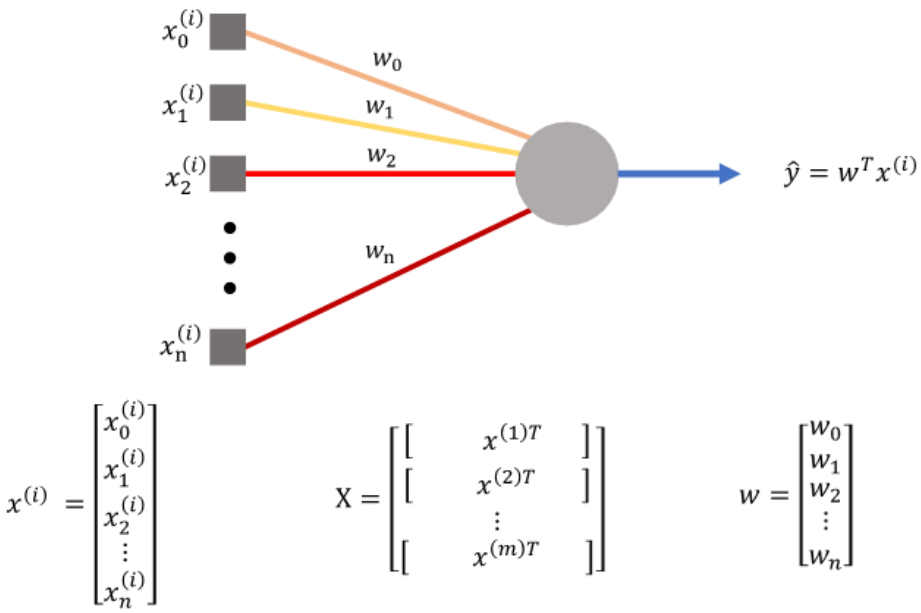

Fig. 1.3. Perceptrón. El perceptrón consta de distintos nodos de entrada con diferentes señales: $x_0^{(i)} - x_n^{(i)}$, las cuales son conocidas como características. Estos se agrupan en un vector columna conocido como vector de características. Se tiene un conjunto de estas medidas que se agrupan en los renglones de una matriz conocida como la matriz de características. La salida del perceptrón es una combinación lineal del i-ésimo vector de características. Los factores de peso de cada característica están determinados por las componentes de un vector columna conocido como vector de parámetros.

El perceptrón, que vemos de manera gráfica en la Fig. 1.3, es una unidad de procesamiento muy simple y solo puede resolver problemas linealmente separables, es decir, problemas en los que se puede trazar una línea recta o un hiperplano para separar las clases de entrada. Sin embargo, los perceptrones se utilizan como componentes básicos en redes neuronales más complejas, como las

redes neuronales multicapa, que pueden resolver problemas más complejos mediante la combinación de múltiples perceptrones en capas interconectadas y la utilización de funciones de activación no lineales.

En particular veremos softwares de clasificación de datos, como el clasificador logístico y el clasificador SoftMax.

1.2 Clasificador logístico o Binario.

Un clasificador logístico binario (Kim, 2017), también conocido como regresión logística, es un algoritmo de aprendizaje supervisado utilizado para predecir la probabilidad de que una instancia pertenezca a una de dos clases. A pesar de su nombre, la regresión logística se utiliza comúnmente en problemas de clasificación binaria en lugar de problemas de regresión.

- Entrada de Características (Features): Se recopilan las características relevantes de las instancias que se quieren clasificar. Cada instancia se representa como un vector de características.
- Pesos y Bias: Cada característica tiene asociados pesos que se ajustan durante el proceso de entrenamiento del modelo. Además, hay un término de sesgo (bias) que se utiliza para ajustar la salida del modelo.
- Función Logística: La salida del modelo se calcula aplicando una función logística (también llamada función sigmoide) a la combinación lineal de las características ponderadas por los pesos y sumando el sesgo. La función logística transforma la salida a un rango entre 0 y 1, que se interpreta como la probabilidad de pertenecer a la clase positiva.
- Umbral de Decisión: Se establece un umbral de decisión (generalmente 0.5) para convertir las probabilidades en predicciones binarias. Si la probabilidad calculada es mayor que el umbral, la instancia se clasifica como perteneciente a la clase positiva; de lo contrario, se clasifica como perteneciente a la clase negativa.
- Entrenamiento: Durante la fase de entrenamiento, los pesos se ajustan utilizando un algoritmo de optimización para minimizar la diferencia entre las predicciones del modelo y las etiquetas reales de los datos de entrenamiento.

La regresión logística es ampliamente utilizada en problemas de clasificación binaria, como la detección de spam, diagnósticos médicos binarios y muchas otras aplicaciones. A pesar de su nombre, no se utiliza para problemas de regresión continua, sino para problemas de clasificación.

En el caso de los clasificadores para un clasificador binario la salida del perceptrón debe darnos ahora la probabilidad (o confianza) que se tiene de que $X^{(i)}$ pertenece a una clase particular. Para esto necesitaremos las funciones de activación:

$$\sigma(t) = \frac{1}{1 + e^{-t}} \; función\ sigmoide \qquad (1)$$

Que tiene una gráfica como se muestra en la Fig. 1.4:

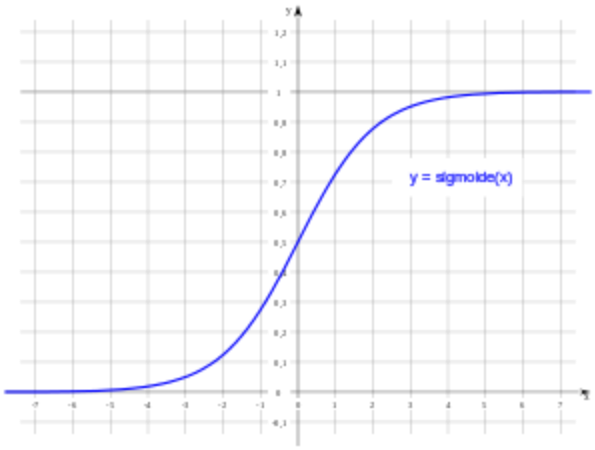

Fig. 1.4 Gráfica de la función sigmoide.

Y representa la probabilidad de que $X^{(i)}$ pertenezca a una clase.

Representación del clasificador logístico:

- 1 si $\hat{p} > 0.5$ Pertenece
- 0 si $\hat{p} < 0.5$ No pertenece
- ¿? si $\hat{p} = 0.5$ No sabemos

Para este clasificar usaremos de ejemplo un paquete de datos realizado por Fisher el cual tiene los datos de 3 flores (iris setosa, virginica y versicolor) (Donado, 2021), el conjunto contiene las medidas de 150 flores iris, cada muestra de flor representa una fila de nuestro conjunto de datos y las medidas de la flor en centímetros se

almacenan en columnas, que también denominaremos características del conjunto de datos. En la Fig. 1.5, se observa cómo se logran clasificar los datos en dos clases diferentes.

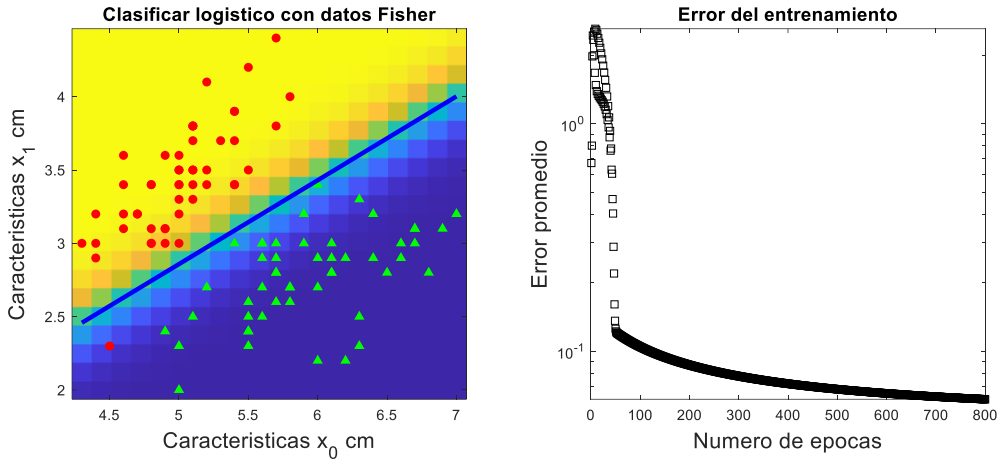

Fig. 1.5 Clasificador binario o logístico, la gráfica de la izquierda muestra cómo se separan los datos con el criterio, pertenece o no pertenece. La gráfica de la derecha muestra el error en cada iteración y como va disminuyendo.

1.3 Clasificador multiclase: SoftMax

La clasificación SoftMax (Aurélien, 2019) es una técnica utilizada en el aprendizaje automático y la estadística para asignar probabilidades a varias categorías mutuamente excluyentes. Se utiliza comúnmente en problemas de clasificación multiclase, como reconocimiento de imágenes, procesamiento de lenguaje natural y otros tipos de análisis de datos. En este clasificador tenemos n neuronas artificiales, las cuales están conectadas en capas ocultas y se obtienen tantas salidas como clasificaciones se tienen, como se aprecia en la Fig. 1.6.

El proceso de clasificación SoftMax funciona de la siguiente manera:

1. Entrada de datos: Se tienen datos de entrada, que pueden ser características de un objeto que se quiere clasificar. Por ejemplo, si estás clasificando imágenes de animales, las características podrían ser el tamaño, el color, la forma de las patas, etc.
2. Puntuaciones: Cada categoría posible recibe una puntuación. Estas puntuaciones se calculan mediante una función lineal que combina las

características de entrada. Cuanto mayor sea la puntuación de una categoría, más probable es que el objeto pertenezca a esa categoría.

3. Función SoftMax: A continuación, se aplica la función SoftMax a estas puntuaciones. La función SoftMax toma las puntuaciones y las convierte en probabilidades. Básicamente, hace que las puntuaciones sean más interpretables como probabilidades, donde una puntuación alta significa una alta probabilidad de pertenecer a una categoría y una puntuación baja significa una baja probabilidad.

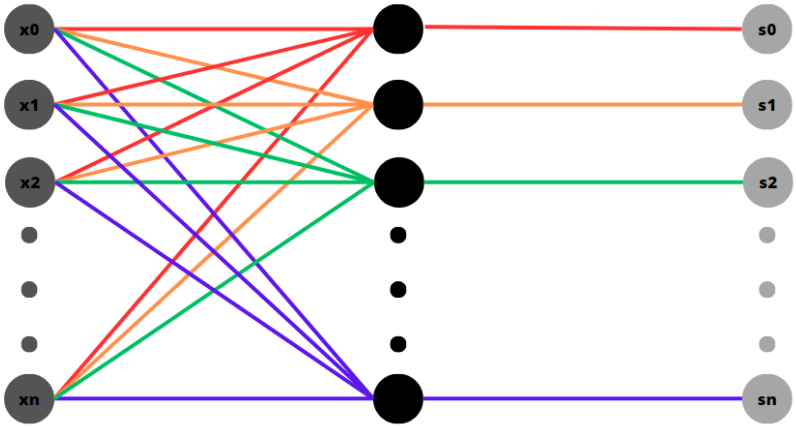

Fig. 1.6. Siguiendo el mismo principio podemos extender a n neuronas artificiales, en donde pasamos de tener un vector de parámetros a una matriz de parámetros W, que consta de todas las combinaciones lineales de cada característica con sus respectivos pesos. Y tenemos n salidas.

La fórmula de la función SoftMax (Mirjalili, Segunda edición en español 2019) para una categoría "k" en particular es:

$$P(k) = \frac{e^{s_k}}{\sum_{j=1}^{K} e^{s_j}} \quad (2)$$

Donde:

$P(k)$ es la probabilidad de que la entrada pertenezca a la categoría "k".

s_k es la puntuación asociada a la categoría "k".

K es el número total de categorías.

Elección de la categoría: Finalmente, la categoría con la probabilidad más alta después de aplicar SoftMax se elige como la predicción. En otras palabras, la categoría con la puntuación más alta se considera la categoría más probable para la entrada dada.

En resumen, el clasificador SoftMax toma puntuaciones de entrada y las convierte en probabilidades asignadas a diferentes categorías. Esta técnica es ampliamente utilizada en problemas de clasificación multiclase y es una parte esencial de muchas redes neuronales y modelos de aprendizaje automático para la clasificación.

1.3.1 El error (Función de costo)

El error en el clasificador SoftMax se refiere a la discrepancia entre las probabilidades predichas por el clasificador y las probabilidades reales o correctas (Francesco Camastra, 2008). Este error se utiliza comúnmente para evaluar el rendimiento de un modelo de clasificación SoftMax en problemas de clasificación multiclase.

El error en el clasificador SoftMax puede calcularse de varias formas, pero una de las métricas más comunes para medir el error es la "entropía cruzada" o "pérdida logarítmica" (cross-entropy loss en inglés). La fórmula de la entropía cruzada para un solo ejemplo de entrenamiento es: (Shannon, 1948)

$$E(y, \hat{y}) = -\sum_{i} y_i \log(\widehat{y_i}) \qquad (3)$$

Donde:

y_i es la probabilidad real o correcta de que el ejemplo pertenezca a la clase "i".

$\widehat{y_i}$ es la probabilidad predicha por el clasificador SoftMax para la clase "i".

El objetivo es minimizar esta pérdida o error durante el proceso de entrenamiento del modelo. Cuanto más cerca estén las probabilidades predichas de las probabilidades reales, menor será la pérdida y mejor será el rendimiento del clasificador.

En resumen, el error en el clasificador SoftMax mide cuán bien el modelo está haciendo sus predicciones en comparación con las respuestas reales. Minimizar

este error es esencial para entrenar un modelo de clasificación SoftMax preciso y efectivo.

1.4 Gradiente descendente

El gradiente descendente es un algoritmo de optimización utilizado en el aprendizaje automático y en la optimización de funciones en general. Su objetivo principal es encontrar el mínimo de una función (o máximo en el caso del gradiente ascendente) ajustando iterativamente sus parámetros. Esto es especialmente útil en el entrenamiento de modelos de aprendizaje automático, donde queremos minimizar una función de pérdida para que el modelo haga predicciones más precisas.

El funcionamiento del gradiente descendente se puede entender de la siguiente manera:

1. **Inicialización**: Comienza con un conjunto de valores iniciales para los parámetros del modelo. Estos valores iniciales pueden ser aleatorios o definidos de alguna otra manera.
2. **Cálculo del gradiente**: Calcula el gradiente (la derivada) de la función que deseas optimizar con respecto a los parámetros. El gradiente indica la dirección en la que la función aumenta más rápidamente. El objetivo es moverse en la dirección opuesta al gradiente para encontrar el mínimo.
3. **Actualización de los parámetros**: Ajusta los parámetros del modelo moviéndolos en la dirección opuesta al gradiente. Esto se hace multiplicando el gradiente por una tasa de aprendizaje, que controla el tamaño de los pasos que das en cada iteración. Una tasa de aprendizaje más pequeña puede dar resultados más precisos, pero requerir más tiempo de entrenamiento, mientras que una tasa de aprendizaje más grande puede converger a mayor velocidad, pero puede ser menos estable.
4. **Repetición**: Repite los pasos 2 y 3 hasta que se cumpla alguna condición de parada, como un número fijo de iteraciones o cuando el cambio en la función de pérdida se encuentra en el intervalo deseado.

El proceso se repite hasta que se encuentra un mínimo (o máximo) de la función, lo que significa que los parámetros del modelo están ajustados de manera óptima para minimizar la función de pérdida. Esto permite entrenar modelos de aprendizaje automático y ajustar sus parámetros para hacer predicciones precisas en función de los datos de entrenamiento. El gradiente descendente es esencial en el entrenamiento de redes neuronales y otros algoritmos de aprendizaje automático.

Usualmente los datos que se desean analizar están mezclados con otros que son redundantes o ajenos a la información de interés. En estos casos es necesario buscar técnicas que permitan resaltar la información relevante. Una de las técnicas más comunes y poderosas es el llamado Análisis de Componentes Principales. Esta técnica permite no solo resaltar la información relevante sino además permite incluso reducción en la dimensionalidad del conjunto de datos. En la Fig. 1.7, se ilustra esta técnica.

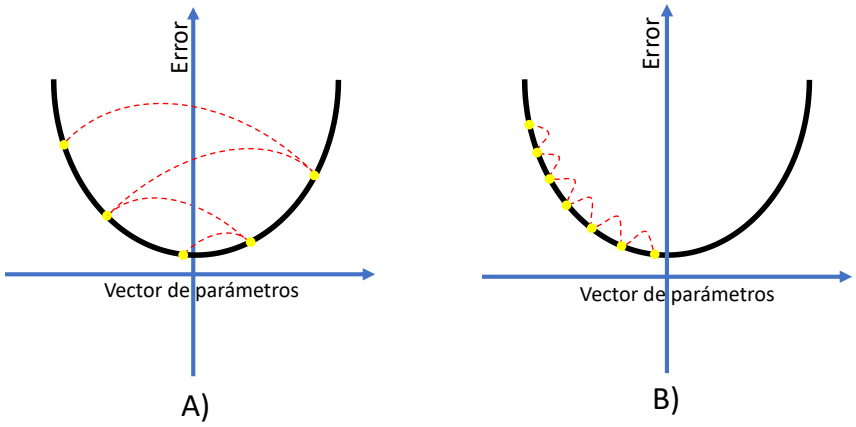

Fig. 1.7. Gradiente descendente. La tasa de aprendizaje determina la rapidez de convergencia, sin embargo, valores muy grandes pueden resultar en una divergencia. A) una tasa de aprendizaje grande puede generar fluctuaciones alrededor del mínimo he incluso divergir, B) valores pequeños de η aseguran la convergencia a consta del número de iteraciones.

1.5 Análisis de Componentes Principales

El método matemático del Análisis de Componentes Principales (PCA, por sus siglas en inglés) (Steven L. Brunton, 2019) se basa en la diagonalización de la matriz de covarianza de los datos originales. A continuación, se describen los pasos matemáticos fundamentales involucrados en el PCA:

Obtención de los datos: Comenzamos con un conjunto de datos en forma de una matriz, donde las filas representan observaciones y las columnas representan variables. Denotemos esta matriz como X, donde tiene dimensiones nxp, con n observaciones y p variables.

Normalización de datos: En ocasiones la escala de las características es relevante para un adecuado tratamiento de los datos, en esos casos es útil que todas las variables tengan la misma escala para realizar un PCA adecuado. Esto implica restar la media de cada variable y, opcionalmente, dividir por la desviación estándar (normalización estándar) o por el rango (mínimo – máximo).

Cálculo de la matriz de covarianza: Después de la normalización, calculamos la matriz de covarianza de los datos. La matriz de covarianza, denotada como Σ, es una matriz pxp donde cada elemento Σ_{ij} representa la covarianza entre las variables *i* y *j*.

$$\sum = \frac{1}{n}\mathrm{X}^{T}\mathrm{X} \qquad (4)$$

Donde X^T es la matriz traspuesta de *X*.

Cálculo de los eigenvectores y eigenvalores: El paso crucial en PCA es encontrar los eigenvectores y eigenvalores de la matriz de covarianza Σ. Estos eigenvectores representan las direcciones en las cuales los datos tienen la mayor variabilidad, y los autovalores indican la cantidad de varianza explicada en cada dirección.

$$\sum_{v} = \lambda_{v} \qquad (5)$$

Donde:

v es un eigenvector de Σ.

λ es el autovalor correspondiente a ese eigenvector.

Es común ordenar los eigenvectores en orden descendente de acuerdo con los autovalores, de manera que el primer eigenvector representa la dirección con la

mayor varianza en los datos, el segundo con la segunda mayor varianza, y así sucesivamente.

1.5.1 Selección de componentes principales

Para reducir la dimensionalidad, puedes seleccionar un subconjunto de los componentes principales (eigenvectores) que explican la mayoría de la varianza en los datos. La elección depende de tus objetivos y del nivel de retención de información que desees.

Transformación de datos: Finalmente, para obtener las componentes principales en términos de los eigenvectores seleccionados, multiplicamos la matriz de datos original X por los eigenvectores seleccionados. Esto crea una nueva matriz donde las columnas representan las componentes principales.

$$Y = X \cdot V \tag{6}$$

Donde:

Y es la matriz transformada con las componentes principales.

X es la matriz original de datos normalizados.

V es una matriz cuyas columnas son los eigenvectores seleccionados.

El resultado es que obtienes una representación de los datos en un nuevo espacio de menor dimensión, donde las componentes principales capturan la mayoría de la variabilidad de los datos originales. Esto facilita la visualización, el análisis y la reducción de dimensionalidad en diversas aplicaciones. Podemos verlo gráficamente en dos dimensiones tomando 2 características, como se aprecia en Fig. 1.8:

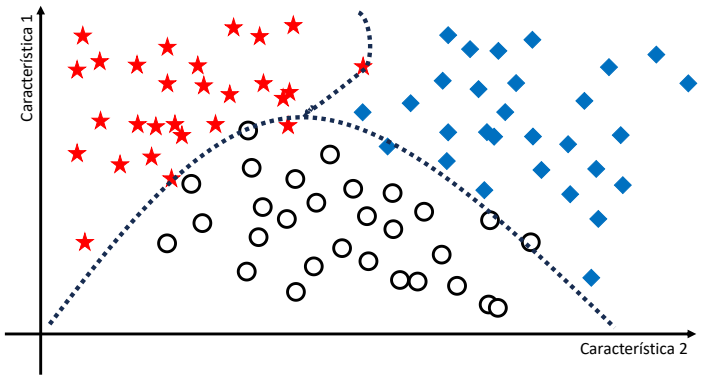

Fig. 1.8. Clasificador multiclase. El clasificador multiclase separa los datos por características en común y se observar cómo se agrupan en regiones determinadas y visiblemente separadas.

Capítulo 2: Análisis de señales de audio

Para los archivos de audio que utilizamos de base los creamos utilizando Matlab, por cada persona grabamos los números del 0 al 9, 50 veces cada uno, teniendo así 50 archivos con 10 números por archivo, haciendo en total 500 audios por persona.

Los audios se grabaron mediante la realización de un programa en Matlab que graba, recopila y recorta los audios, en este mismo definimos la frecuencia que tendrá el audio (8 KHz), los Bits de muestreo (16) y el número de canales (1), el cual utiliza el micrófono de la computadora. El programa en un lapso de 10 segundos graba a la persona repitiendo el mismo dígito 10 veces y muestra la gráfica de los audios, cuando se tiene la gráfica se recortan los audios.

Estos audios delimitan el ancho de las grabaciones para que todas estas fueran del mismo tamaño y sea un poco más fácil su manera de analizar, cada grabación de digito tiene un ancho de 2500 datos.

Al intentar analizar las grabaciones con los datos crudos (raw data), nos dimos cuenta de que había demasiados errores, por lo que se tuvieron que aplicar la transformada de Fourier para poder hacer un mejor análisis de los datos.

Para esto hicimos un script en Matlab en donde cargamos todos los archivos de audio, calculamos el número de características, también creamos la matriz de características, y se comienza el análisis.

2.1.Transformada de Fourier

La transformada de Fourier es una herramienta matemática fundamental que se utiliza para analizar las frecuencias presentes en una señal o función. Fue desarrollada por el matemático francés Jean-Baptiste Joseph Fourier a principios del siglo XIX.

La idea principal detrás de la transformada de Fourier es descomponer una función (o una señal en el caso de las aplicaciones prácticas) en una combinación de

funciones sinusoidales (senos y cosenos) de diferentes frecuencias (Butz, 2015). Esto permite representar la función original en el dominio de la frecuencia, lo que facilita el análisis de sus componentes armónicos.

La transformada de Fourier se puede dividir en dos formas principales: la transformada de Fourier continua y la transformada de Fourier discreta. La primera se aplica a funciones continuas en el tiempo o el espacio, mientras que la segunda se utiliza para señales discretas, como las señales muestreadas en el procesamiento digital de señales.

La transformada de Fourier tiene una amplia gama de aplicaciones en áreas como la ingeniería, la física, las telecomunicaciones, el procesamiento de imágenes y el procesamiento de señales. Se utiliza para analizar y sintetizar señales, filtrar ruido, comprimir datos, resolver ecuaciones diferenciales, entre muchas otras aplicaciones. En Fig. 2.1, se muestra el diagrama de flujo que sigue el programa para realizar la Transformada de Fourier.

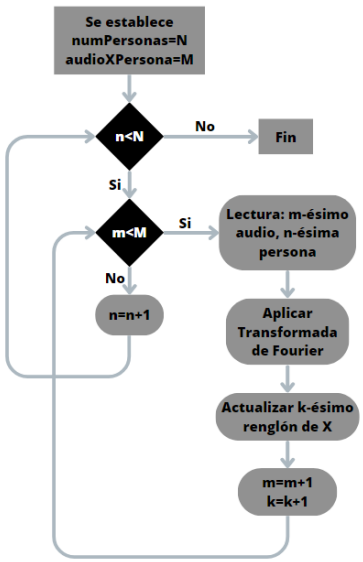

Fig. 2.1.Diagrama de flujo. Teniendo el número de personas y el número de grabaciones por persona se inicia el programa, se lee un archivo, se le aplica Transformada de Fourier y se actualiza el k-ésimo renglón de la matriz de características, el total de iteraciones es la matriz de características con Transformada de Fourier para analizar los datos.

Lo que hacemos crear un ciclo *for* para cada persona, con otro anidado ya que cada persona tiene 10 dígitos diferentes, con un tercer ciclo *for* anidado porque cada

dígito tiene 5 archivos con 10 grabaciones. En el último ciclo *for* lo que se hace es extraer el s-ésimo clip, a este clip le aplicamos la transformada de Fourier y guardamos los datos obtenidos.

2.2. Espectrograma

Un espectrograma es una representación gráfica del espectro de frecuencias de una señal en función del tiempo (Fulop, 2006). Es una herramienta ampliamente utilizada en diferentes campos como el procesamiento de señales, el análisis de audio, la ingeniería acústica y otras disciplinas relacionadas.

El espectrograma toma una señal de entrada, generalmente una señal de audio, y muestra cómo varían las diferentes frecuencias a lo largo del tiempo. La señal de entrada se divide en pequeñas ventanas de tiempo y, para cada ventana, se calcula la transformada de Fourier, que descompone la señal en sus diferentes componentes de frecuencia. Luego, los valores resultantes se representan visualmente mediante colores o sombreado en un gráfico 2D, donde el eje horizontal representa el tiempo y el eje vertical muestra la frecuencia.

En un espectrograma, las áreas más brillantes o intensas indican que hay mayor energía en esa frecuencia particular en ese momento específico. Por el contrario, las áreas más oscuras o menos intensas indican menor energía en esas frecuencias y momentos.

El espectrograma se utiliza para visualizar y analizar la evolución de las frecuencias en una señal a lo largo del tiempo, lo que es útil para identificar patrones, eventos importantes y características específicas. Algunos ejemplos de aplicaciones prácticas incluyen la identificación de tonos y notas musicales, la detección de sonidos específicos en el audio, el análisis de señales de voz y la monitorización de señales en diversas industrias.

En resumen, un espectrograma es una representación visual del contenido de frecuencia de una señal a lo largo del tiempo, lo que permite una comprensión más detallada y visualmente informativa del comportamiento de la señal en el dominio de la frecuencia y el tiempo.

En Matlab para poder trabajar con espectrogramas se tiene la función:

$$spectrogram(x, window, noverlap, nfft, fs)$$

Donde:

- x es la señal de entrada que deseas analizar.
- window es la ventana utilizada para dividir la señal en segmentos solapados. Puedes especificar una ventana predefinida como 'hamming' o 'hann', o puedes definir tu propia ventana.
- noverlap es la cantidad de muestras en las que se superponen los segmentos de la señal. Un valor común es window/2.
- nfft es el tamaño de la transformada de Fourier de cada segmento de la señal. Si no se especifica, MATLAB utiliza un valor predeterminado.
- fs es la frecuencia de muestreo de la señal.

La función spectrogram devuelve tres valores (que podemos renombrar): S, F y T. S es la matriz del espectrograma, F es el vector de frecuencias y T es el vector de tiempo.

El script que utilizamos es de la siguiente forma:

Definiendo todas las variables que utilizaremos.

```
%% Parametros del spectrograma
% Numero de datos por segmento
numSamples=2^8;
% Muestras por traslape
numTraslape=250;
% Frecuencias a tomar del segmento
numFreg=numSamples;
% Frecuencia de muestreo
Fs=8e3;
%% Determinamos las dimensiones del espectrograma (numero de caracteristicas)
clip=Xdat(:,1);
[CLIP,fs,ts]=spectrogram(clip,numSamples,numTraslape,numFreg,Fs);
[nfreg,nt]=size(CLIP);
```

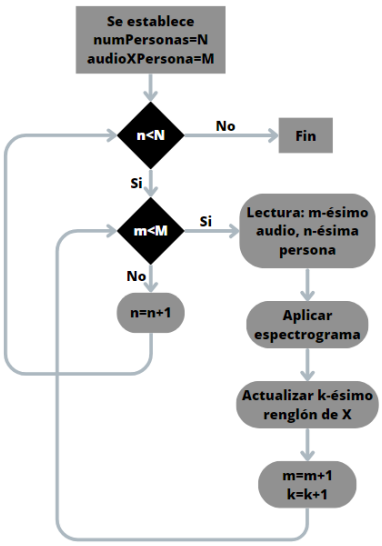

Fig. 2.2. Diagrama de flujo aplicando espectrograma. Después de realizar análisis con Transformada de Fourier, se realiza esencialmente lo mismo, pero ahora se analizan los audios con espectrograma, se lee cada audio y de cada audio se obtiene un espectrograma.

Capítulo 3: Resultados experimentales

En este capítulo se presentan los resultados experimentales de la adquisición de las señales de audio, su procesamiento y el desempeño del clasificador multiclase desarrollado en este trabajo.

En la Fig. 3.1 se muestran los archivos de audio obtenidos al grabar a la persona 1 pronunciando los dígitos del 1 al 9. Como se observa en términos generales puede decirse que existen diferencias visuales en los perfiles de cada audio.

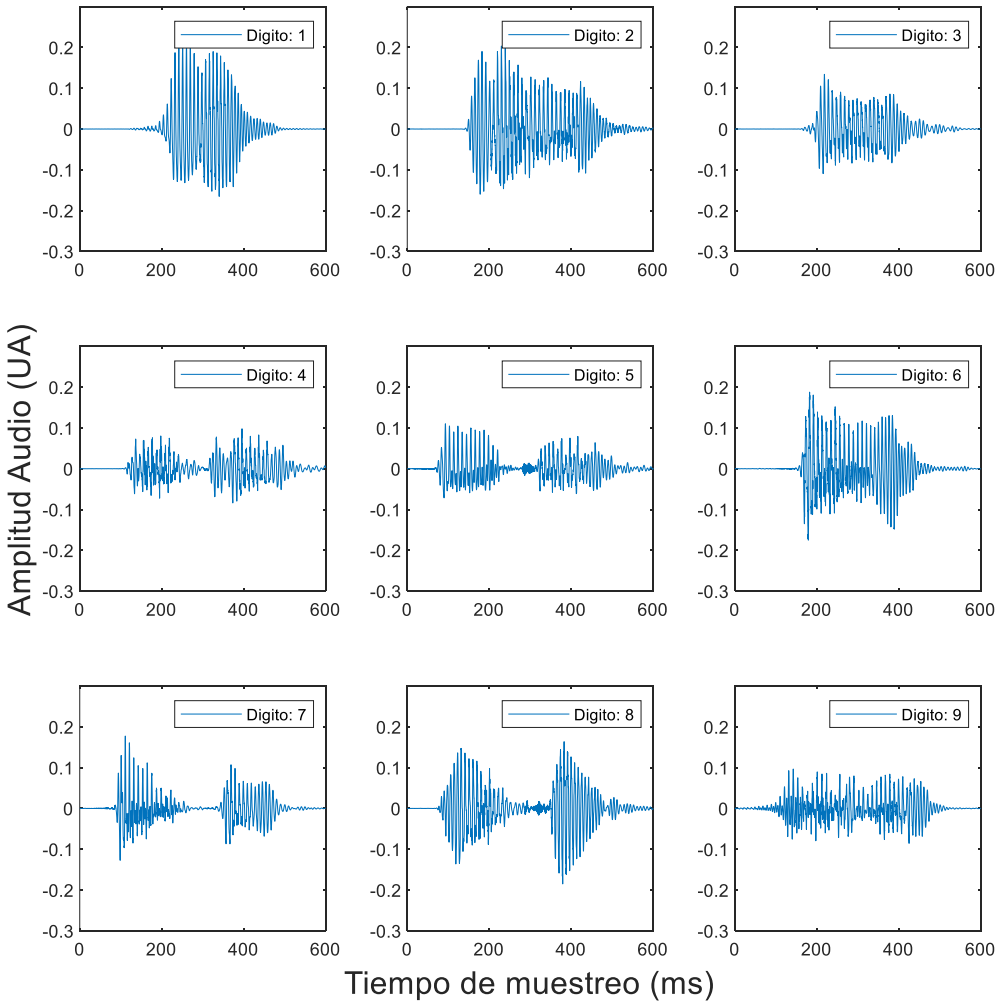

Fig. 3.1. Ejemplos de las señales de audio grabadas. Se muestran los audios de los dígitos del 1 al 9 para la persona 1.

Cada archivo de audio fue muestreado con una frecuencia de 8 KHz con un solo canal y una duración de 625 ms. Esto significa que cada archivo de audio consta de 5000 datos. Este número tan elevado de características no permite visualizar de manera simple la existencia o no de patrones. Por otra parte, existe mucha información que es redundante. Para resaltar la información relevante y disminuir el número de características se aplicó la transformada de Fourier a cada señal de audio y posteriormente se aplicó el PCA al nuevo conjunto de datos. Los resultados de esto se muestran en la Fig. 3.2. Esta figura muestra que con solo dos componentes principales es ya posible distinguir claramente entre dos dígitos diferentes (0 y 9 en este ejemplo). A pesar de que en su mayoría los datos de cada dígito se agrupan en áreas distintas, aún existe un número de datos que se superponen, lo que implica una alta posibilidad de generar errores en su clasificación.

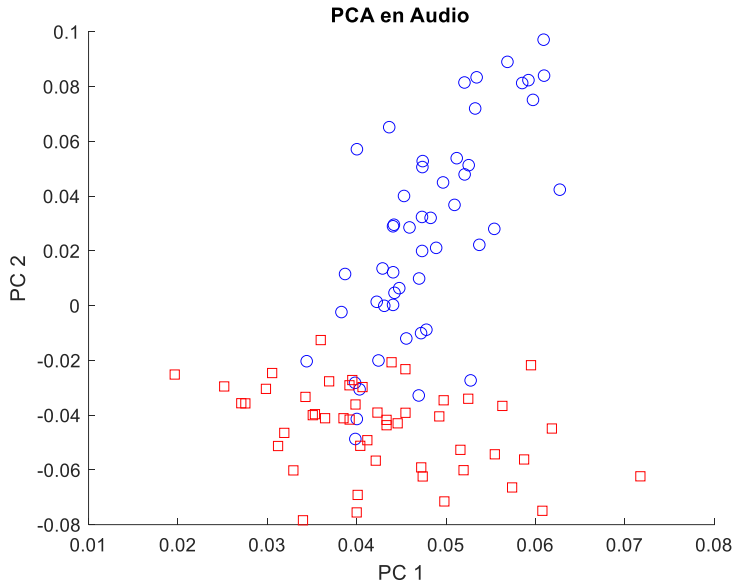

Fig. 3.2. Análisis de componentes principales de los audios de los distintos dígitos. Como se observa la separación de los dígitos 0 y 9 es posible, sin embargo, existen datos que se traslapan lo que minimiza la eficiencia de reconocimiento.

Esto puede arreglarse agregando más componentes principales al proceso de clasificación, sin embargo, en realidad esta es una de las mejores curvas de datos, pero existen otras en donde los dígitos se traslapan de manera más severa. Por esta razón se decidió substituir la transformada de Fourier por el análisis de

espectrograma. Un ejemplo de los espectrogramas obtenidos se presenta en la Fig. 3. 3. En estos espectrogramas se presenta la información de las componentes espectrales y su evolución en el tiempo por ejemplo para el digito 0 se nota la presencia de señales de algunos cientos de Hz, en los 150 ms la cual desaparece súbitamente a los 300 ms, aparece nuevamente y se mantiene alrededor de los 400 ms. Esto significa esencialmente que se tiene ahora una mayor cantidad de información con este análisis.

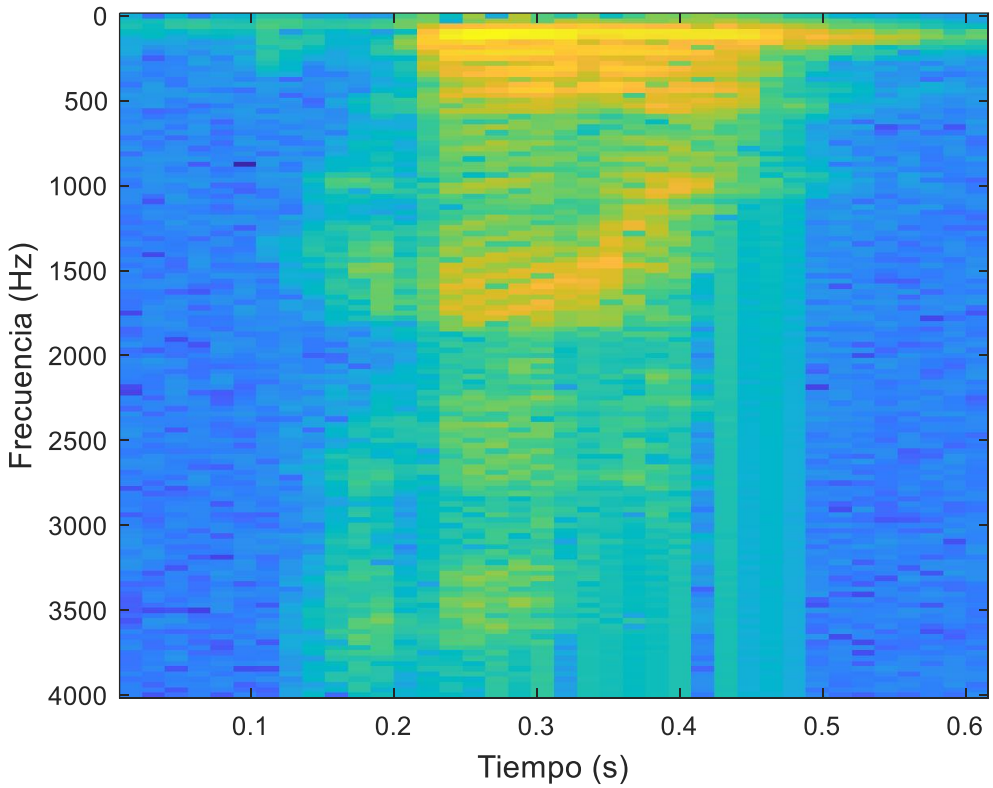

Fig. 3. 3. Espectrograma de una señal de audio. El espectrograma de una señal permite visualizar la evolución espectral (frecuencias) en el tiempo. En este caso el tiempo se observa en el eje horizontal y en el eje vertical se grafican las frecuencias que aparecen en este tiempo. El espectrograma pertenece a la señal de audio del digito 0.

Como ya se mencionó anteriormente la creación de espectrogramas depende de dos parámetros cruciales: el ancho de la ventana y el número de traslape. Estos dos valores determinan la resolución temporal y espectral de cada espectrograma. En la Fig. 3.4 se presentan los espectrogramas de 9 dígitos con ventanas/traslape de 256/250 y 512/500, respectivamente. Como se observa de las figuras a simple vista

no parece haber grandes diferencias, sin embargo, con un análisis más detallado pueden percibirse cambios sutiles que pueden afectar el desempeño de la clasificación. Mas adelante se presentan resultados del clasificador para distintos valores de estos parámetros.

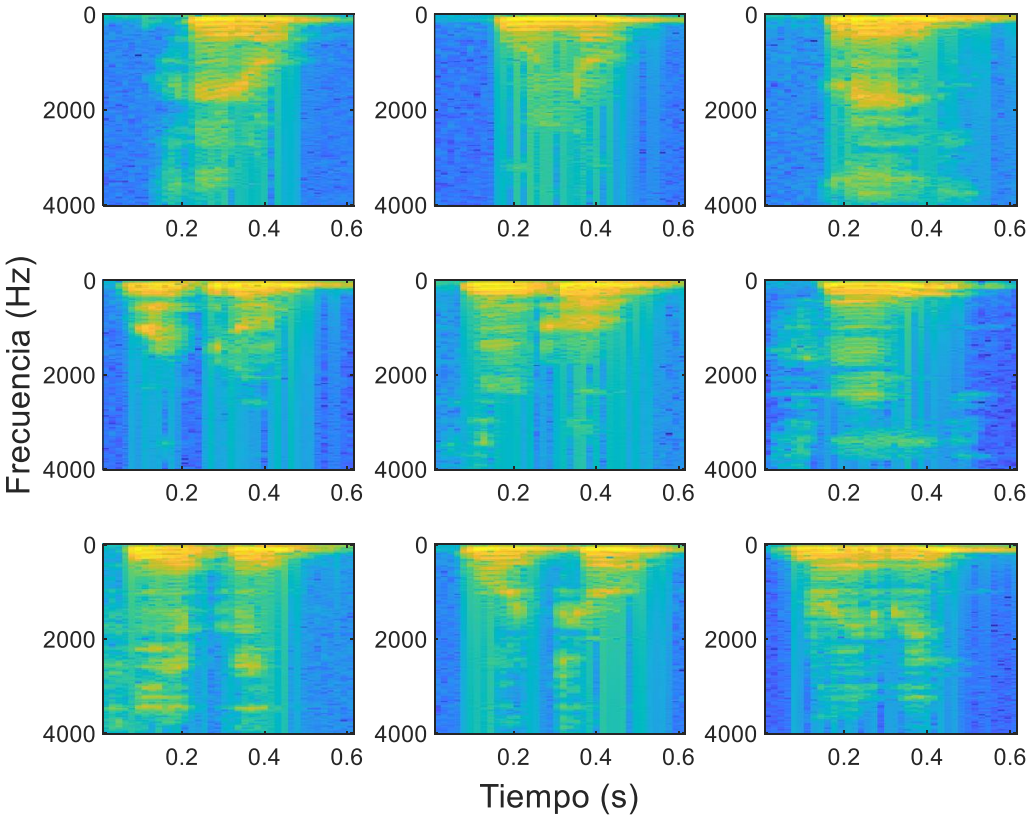

Fig. 3.4. Espectrogramas de las señales de audio. Se muestran los espectrogramas de los dígitos del 1-9 (de izquierda a derecha y de arriba a abajo) con ventanas y traslapes de A) 256, 250 y B) 512, 500, respectivamente.

Una vez realizados los espectrogramas se aplica el PCA al valor absoluto de estos () y se obtiene un resultado como el que se muestra en la Fig. 3.5. Como se observa, el agrupamiento es mucho mejor que el logrado en la Fig. 3.2, ahora no existe ningún tipo de traslape entre las muestras de audio analizadas lo que implica una alta razón de clasificación. Después de analizar múltiples graficas se llegó a la conclusión de que una ventana de 256 puntos junto con un traslape de 250 puntos presentaba un resultado aceptable.

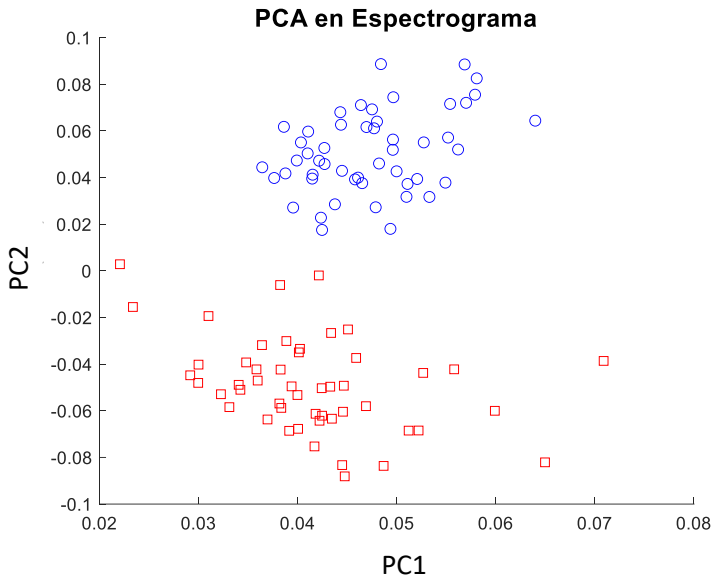

Fig. 3.5. Análisis de componentes principales sobre datos de espectrograma. Al aplicar el PCA a los datos de los espectrogramas se logra una mejor separación entre las distintas clases. En este caso los dígitos 0 y 9 se separan casi por completo a diferencia del caso en la Fig. 3.2, de esta manera se aumenta la eficiencia de clasificación.

El siguiente paso fue utilizar las componentes principales obtenidas mediante PCA sobre el valor absoluto de los espectrogramas en el clasificador multiclase SoftMax. En la Fig. 3.6 se muestra las curvas de entrenamiento en función del número de épocas del clasificador SoftMax con un distinto número de componentes principales.

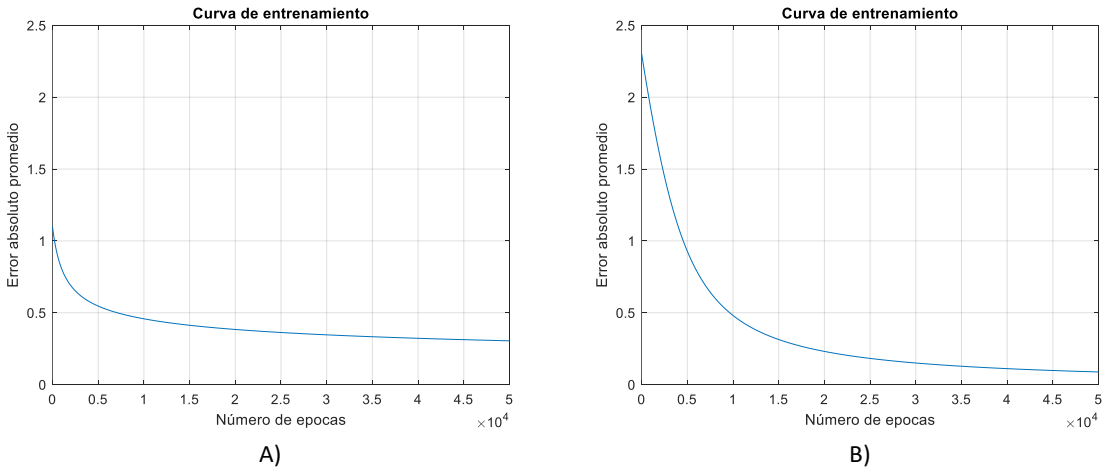

Fig. 3.6. Curvas de entrenamiento del clasificar SoftMax. Las curvas de entrenamiento muestran la evolución del error en el clasificador en función del número de épocas. En esta figura se presentan las curvas de entrenamiento con A) 10 y B) 400 componentes principales. Como era de esperarse con un mayor número de componentes se tiene un menor error en el proceso de clasificación, en este caso el error final es 3.75 veces más pequeño cuando se usan 400 PC's.

En este caso cuando se usan las primeras 10 y 400 componentes se obtiene como resultado errores finales de 0.30 y 0.08 respectivamente, lo que significa una disminución del error de alrededor de 3.75 veces. De esta manera uno puede pensar que en términos generales se debería tomar el mayor número de componentes principales. Sin embargo, esto no necesariamente es cierto, primeramente, porque al usar un numero grande de componentes principales se pierde la idea principal de reducción de dimensiones, por otra parte, al tener un alto número de componentes también puede generarse el llamado sobreajuste.

El sobreajuste (overfitting) es un fenómeno común en el aprendizaje automático donde un modelo se adapta excesivamente a los datos de entrenamiento. Esto significa que el modelo captura tanto el patrón real en los datos como el ruido aleatorio, lo que resulta en un rendimiento deficiente cuando se aplican los datos del mundo real o de prueba.

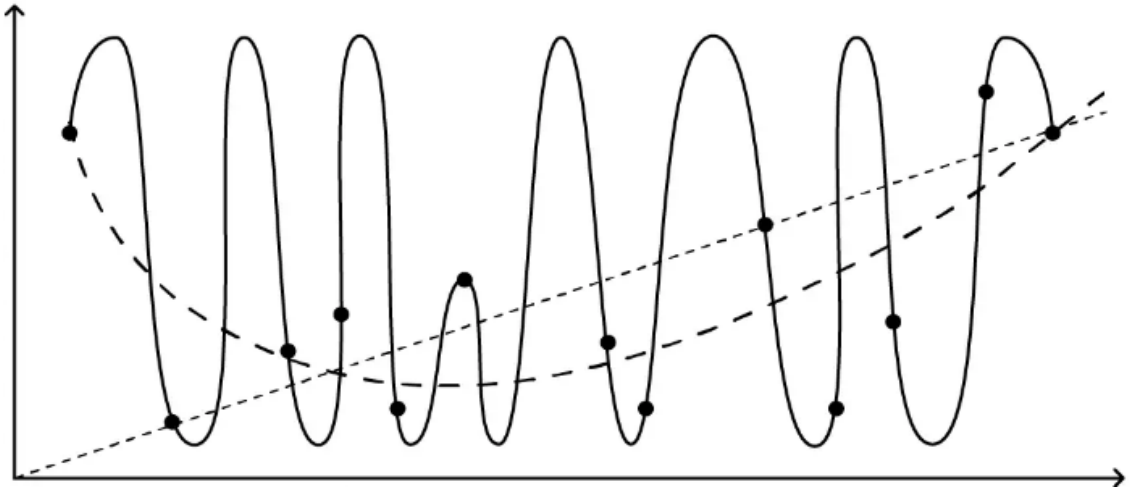

Fig. 3.7. Ejemplo de sobreajuste. En la figura se presenta un conjunto de datos que fueron generados a partir de una función cuadrática a la cual se le ha anexado ruido. Un modelo subajustado puede ser una regresión a una linea recta (linea roja), mientras que un sobre ajuste se presenta cuando se usa un polinomio de grado 300. El mejor modelo debe de ser un ajuste cuadrático.

Este último representa una complejidad innecesaria en el modelo, de tal forma que el error disminuye en el entrenamiento, pero aumenta cuando se utilizan datos distintos al conjunto de entrenamiento. De esta manera es necesario encontrar un balance entre el número de componentes a utilizar y el error obtenido en el

entrenamiento del clasificador. Para determinar el número adecuado de componentes, se decidió elegir distintas PC´s y observar las curvas de probabilidad de pertenencia obtenidas del clasificador SoftMax. En la Fig. 3.8 se muestran los resultados de dichas graficas para el clasificador SoftMax cuando se usan 10 y 60 PC´s.

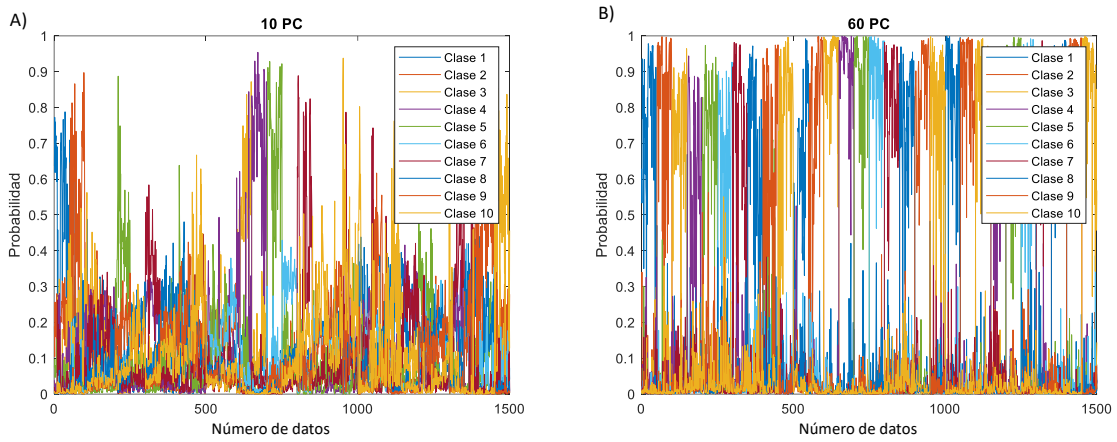

Fig. 3.8. Grafica de probabilidad de pertenencia del clasificar SoftMax. La clasificación fue realizada con A) 10 y B) 60 componentes principales. Como se observa existe una mejoría en el desempeño del clasificador al aumentar el número de componentes principales, sin embargo, aún existe un subajuste.

Como se observa, para el caso de 10 PC´s las probabilidades de pertenencia no superan el 90%, además la Fig. 3.8A difícilmente muestra el patrón que uno esperaría, un perfil cuadrado de valor 1 cuando se detecta la k-ésima clase y 0 en caso contrario. Por otra parte, cuando se emplean 60 PC´s las probabilidades ya superan el 90% y se percibe de manera clara la presencia de los perfiles cuadrados cuando se detecta la k-ésima clase o no.

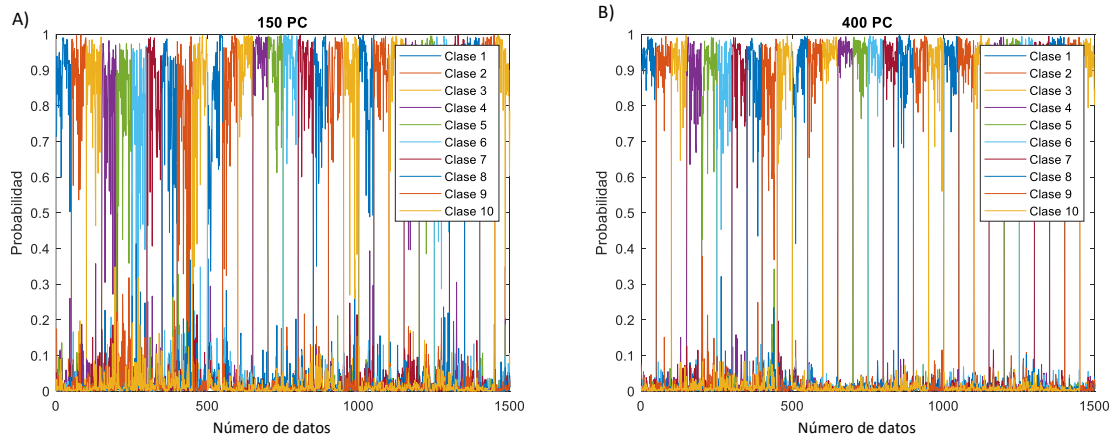

Fig. 3.9. A medida que se aumenta el número de componentes principales la eficiencia del clasificador mejora. En estas gráficas se usan A) 150 y B) 400 componentes principales respectivamente. La mejoría es notable respecto de los resultados mostrados en la Fig. 3.8.

A medida que uno aumenta el número de componentes este patrón cuadrado se hace aún más evidente. En la Fig. 3.9 se muestran las curvas de probabilidad para

150 y 400 PC´s. En ambas gráficas es ya evidente la presencia de los perfiles cuadrados de las curvas que se esperaban, aunque aún es visible que la detección de la k-ésima clase fluctúa entre valores del 50 y 100%. Incluso en estas condiciones la eficiencia de clasificación debe ser alta pues cualquier probabilidad mayor a 50% en la práctica se considera como una pertenencia a la k-ésima clase, lo que significa un error bajo en el proceso de clasificación.

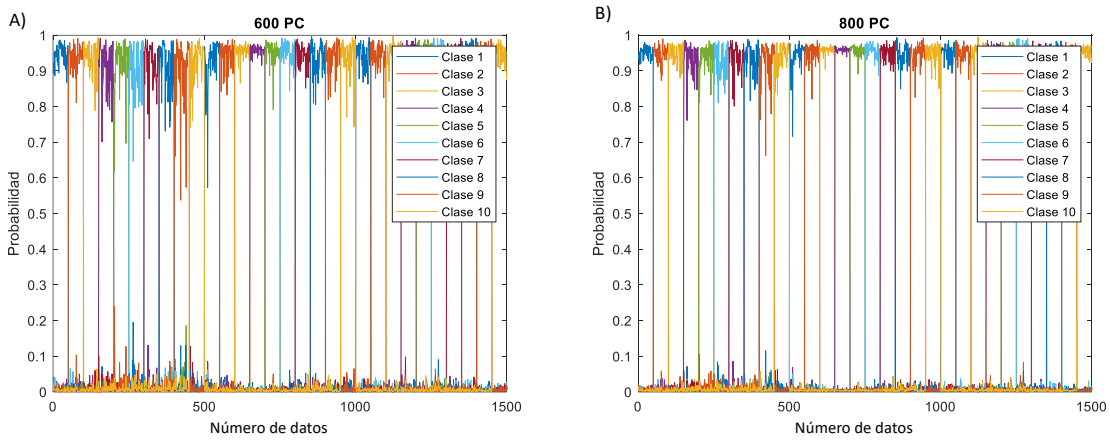

Fig. 3.10. La eficiencia del clasificador puede llegar a ser muy alta con un alto número de componentes principales. En este caso se usan A) 600 y B) 800 Componentes principales. A pesar de los altos niveles de eficiencia estos resultados pueden no ser adecuados, puesto que puede aparecer el llamado sobreajuste a los datos de entrenamiento.

Es posible elegir un mayor número de PC´s y con esto lograr una mejor definición de las funciones de probabilidad. En la Fig. 3.10 se muestran los resultados del clasificador cuando se utilizan 600 y 800 componentes. Como lo muestran las gráficas, las funciones de probabilidad se encuentran por encima del 70%, lo que significa una alta certeza en la clasificación de la k-ésima clase. Sin embargo, también debe aparecer el llamado sobreajuste, que no es otra cosa más que una muy acertada clasificación de los datos de entrenamiento mientras que si se alimentara el clasificador con nuevos datos esta clasificación será errónea. Este proceso debe evitarse de manera que exista un balance entre la clasificación con los datos de entrenamiento y los nuevos datos con los cuales trabajará el clasificador. En nuestro caso consideramos que un número de componentes que permita un buen nivel de clasificación tanto en los datos de entrenamiento como en datos fuera de este conjunto puede ser obtenido con el uso de entre 300 y 400 PC´s

lo cual significaría una reducción significativa del número de características originales que era de 5 mil. Esta reducción está basada en la observación de las gráficas de probabilidades (Fig. 3.9-9), sin embargo, existen otras formas típicas de análisis de desempeño de clasificadores. Una de estas formas es la llamada matriz de confusión. En esta grafica se contabilizan los aciertos junto con los errores en la predicción de las distintas clases. Estas estadísticas se colocan en una matriz de manera que uno de los ejes establece la clase correcta mientras que en el otro se coloca la predicción. De esta manera al colocar los números de errores y aciertos en las predicciones se obtiene una imagen en la que en el mejor de los casos la diagonal presenta números distintos de ceros, mientras que en las demás posiciones solo existen ceros. En cualquier otro caso existen valores en todas las casillas de la matriz determinando las confusiones en las clasificaciones de cada clase.

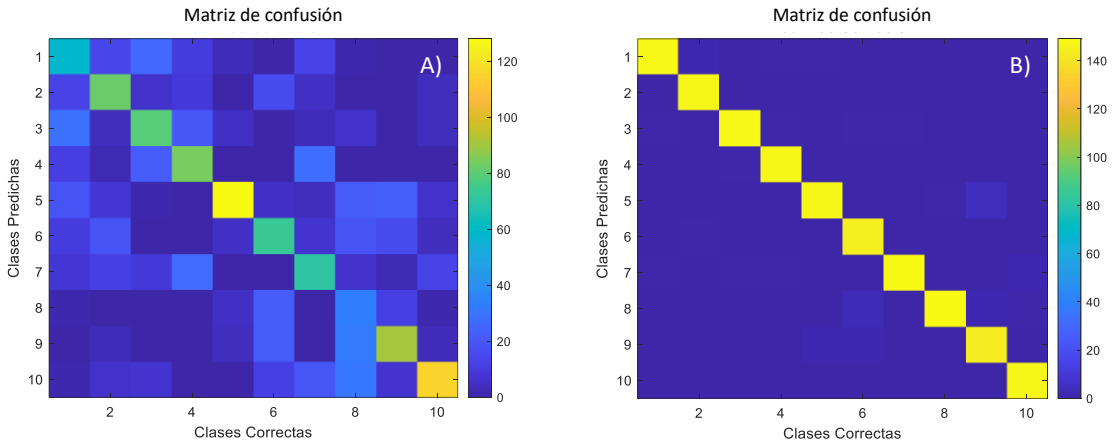

Fig. 3.11. Matriz de confusión para el clasificador SoftMax desarrollado. Se han utilizado: A) 10 y B) 100 componentes principales.

En la Fig. 3.11 se muestra la matriz de confusión del clasificador SoftMax desarrollado en este trabajo. En la Fig. 3.11A se observa que con el uso de solo 10 componentes principales existe un numero apreciable de confusiones en la determinación de las distintas clases. Sin embargo, aun así, los aciertos en términos generales son mayores (la diagonal contiene los valores más altos). Cuando se utilizan 100 componentes principales (Fig. 3.11B) el cambio es dramático, pues ahora se observa que el desempeño del clasificador es muy superior (el valor más alto en las confusiones es de apenas 5 contra 143 que es el valor más pequeño en

los aciertos). Cuando se aumenta el número de componentes principales los resultados son aún más espectaculares como lo muestran las gráficas de la Fig. 3.12. Aquí se observa que con más de 400 componentes principales los errores en las clasificaciones son prácticamente nulos, sin embargo, esto también es un indicativo de un posible sobreajuste, lo cual tampoco es deseable.

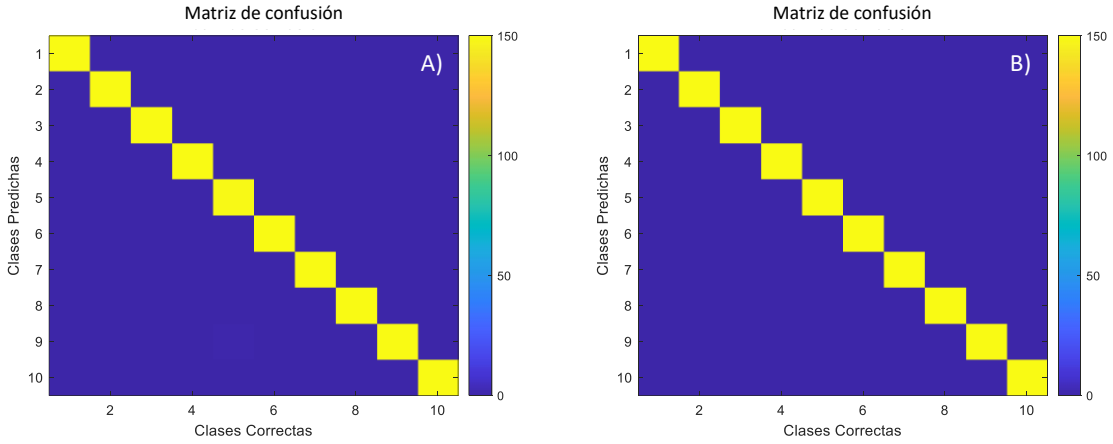

Fig. 3.12. Matriz de confusión para el clasificador SoftMax desarrollado. Se han utilizado: A) 400 y B) 600 componentes principales.

De esta manera se considera que posiblemente la mejor elección en el número de componentes principales puede estar alrededor de 100, en donde el desempeño del clasificador es del orden de más de 90% de acierto en cualquiera de las clases.

En la Fig. 3.13 se muestra la matriz de parámetros como una pseudo imagen. La “brillantez” de la imagen representa el valor particular de cada elemento de los distintos vectores de parámetros. Cada renglón es un vector de parámetros que corresponde a cada una de las clases discriminadas. Las columnas representan el número de características presentes en los vectores de características de entrada. Esta matriz representa los datos relevantes del modelo entrenado y el conocimiento de esta establece por completo el funcionamiento del clasificador. Esta imagen es interesante porque a simple vista pareciera que la información contenida en ella es más bien aleatoria y sin ningún sentido, sin embargo, estos valores permiten la correcta clasificación en un sistema de reconocimiento de voz.

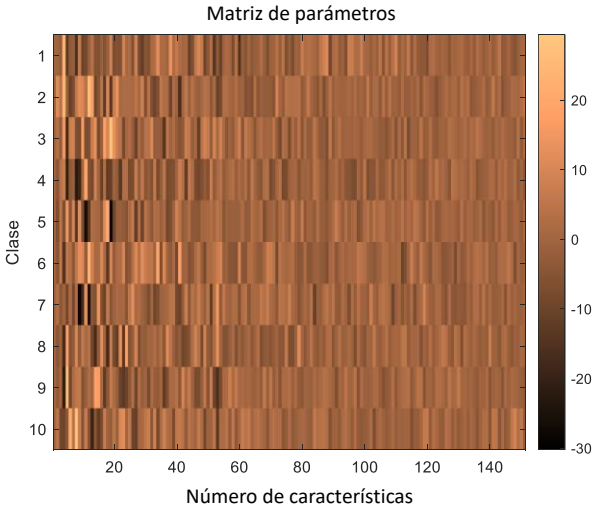

Fig. 3.13. Matriz de parámetros. Aquí se muestra la matriz de parámetros como una pseudo imagen, los renglones representan las distintas clases del clasificador mientras que el número de columnas es el número de características a la entrada del clasificador.

Capítulo 4: Conclusiones

En este trabajo se desarrollaron distintos programas que permiten lo siguiente:

1) La captura de señales de audio.
2) La selección de regiones particulares en cada señal.
3) Procesamiento de las señales seleccionadas mediante la transformada de Fourier o los Espectrogramas para enseguida aplicar un análisis de componentes principales. En cualquier caso, se obtiene la llamada matriz de características con el número de componentes principales deseado.
4) La clasificación fue desarrollada mediante un clasificador multiclase SoftMax el cual fue desarrollado por completo.

A partir de los resultados obtenidos con los programas mencionados anteriormente se llegaron a las siguientes conclusiones.

- Las señales de audio “crudas” no permiten una clasificación adecuada, por lo que siempre es necesario realizar algún preprocesamiento previo al proceso de clasificación.
- El uso de PCA disminuye dramáticamente el número de características necesarias para observar patrones que permitan la clasificación de las señales de audio. Sin embargo, su desempeño aun no es máximo.
- El análisis de PCA posterior a la aplicación de la Transformada de Fourier en las señales de audio mejora substancialmente el desempeño del clasificador, lo que indica que la transformada de Fourier es una buena herramienta extractora de información.
- El uso de los espectrogramas junto con el PCA da aún mejores resultados que el uso de la Transformada. La precisión del reconocimiento de las señales de audio obtenida a través de las matrices de confusión depende fuertemente del número de componentes principales elegidas. Estas precisiones pueden ir desde el 26 hasta más del 95%, para 10 y hasta 600 componentes principales respectivamente.

- En un inicio puede pensarse que un número mayor de componentes principales es lo mejor para incrementar la precisión de clasificación, sin embargo, esto no es del todo cierto, pues puede aparecer el llamado sobreajuste. Creemos que los resultados que se han obtenido son mejores y no cuentan con un sobreajuste elevado cuando se utilizan entre 100 y 200 componentes principales con lo que se logran precisiones del orden del 90% lo cual se considera un resultado satisfactorio.
- Debe resaltarse que los archivos de audio originales fueron grabados con una frecuencia de muestreo de 8 kHz, con lo que se cubre el ancho de banda presente en la voz humana. Esto significa que los archivos de audio presentaban en promedio 2500 datos (características) con duraciones de aproximadamente 300 ms. Con el uso de 150 PC's se tiene una reducción del 94% en el número de características a procesar, lo cual implica un menor tiempo de procesamiento computacional.
- El desempeño del clasificador permite reconocer el digito hablado con una precisión del orden de más de 90%, se cree que también sería posible reconocer a la persona e incluso el género de esta, pero es necesario incrementar el número de grabaciones para corroborar estas ideas.

Finalmente se debe mencionar que como trabajo a futuro se espera realizar un estudio más detallado acerca de la dependencia del clasificador respecto del uso o no del espectrograma o alguna otra técnica de análisis de audio respecto de la señal cruda de estos datos. Esto, para determinar de manera cuantitativa la mejoría con el uso de estas técnicas.

Bibliografía

Aurélien, G. (2019). *Hands- on Machine Learning with Scikit-Learn, Keras & TensorFlow.* United States of America: O'Reilly Media, Inc.

Butz, T. (2015). *Fourier Transformation for Pedestrians.* Editorial Springer.

Donado. (2021). *UC Irvine Machine Learning Repository*. Obtenido de https://archive.ics.uci.edu/dataset/53/iris

Francesco Camastra, A. V. (2008). *Machine Learning for Audio, Image and Video Analysis: Theory and Applications.* Editorial Springer.

Fulop, S. A. (2006). *Algorithms for computing the time-corrected instantaneous frequency (reassigned) spectrogram, with applications.* Journal of the Acoustical Society of America.

Kim, P. (2017). *MATLAB Deep Learning With Machine Learning, Neural Netwrok and ARtificial Intelligence.* Editorial Apress.

Mirjalili, R. S. (Segunda edición en español 2019). *Python Machine Learning Aprendizaje automático y aprendizaje profundo con Python, scikit-learn y Tensor-Flow.* Marcombo.

Shannon, C. E. (1948). A mathematical theory of communication. *The Bell System Technical Journal*, 349-423.

Shlens, J. (10 de diciembre, 2005). A Tutorial on Principal Component Analysis. *Systems Neurobiology Laboratory, Salk Insitute for Biological Studies La Jolla, CA 92037 and Institute for Nonlinear Science, University of California, San Diego La Jolla, CA 92093-0402*, 13.

Steven L. Brunton, J. N. (2019). *Data Driven Science & Engineering Machine Learning, Dynamical System and Control.* Editorial Cambridge University Press.