Tecnológico de Monterrey

Reporte de Modelo de Clasificación de Sonidos

Adrián Matute Beltrán A01703889

19 de noviembre del 2024

Inteligencia artificial avanzada para la ciencia de datos II (Gpo 501)

Tecnológico de Monterrey

Adrián Matute Beltrán A01703889

Desarrollo e Implementación de un Modelo de Deep Learning para la Clasificación de Sonidos Urbanos

Introducción

El propósito de este proyecto es desarrollar un modelo deep learning para clasificar sonidos urbanos en distintas categorías. Para lograrlo, se utilizó el dataset UrbanSound8K, que contiene una variedad de sonidos comunes en entornos urbanos, clasificados en 10 categorías, tales como claxon de autos, ladridos de perros y las voces de niños jugando. La clasificación de sonidos urbanos ha sido un desafío debido a la variabilidad en los patrones sonoros, los diferentes contextos en los que se grabó y el ruido de fondo, lo cual dió la necesidad de técnicas de deep learning.

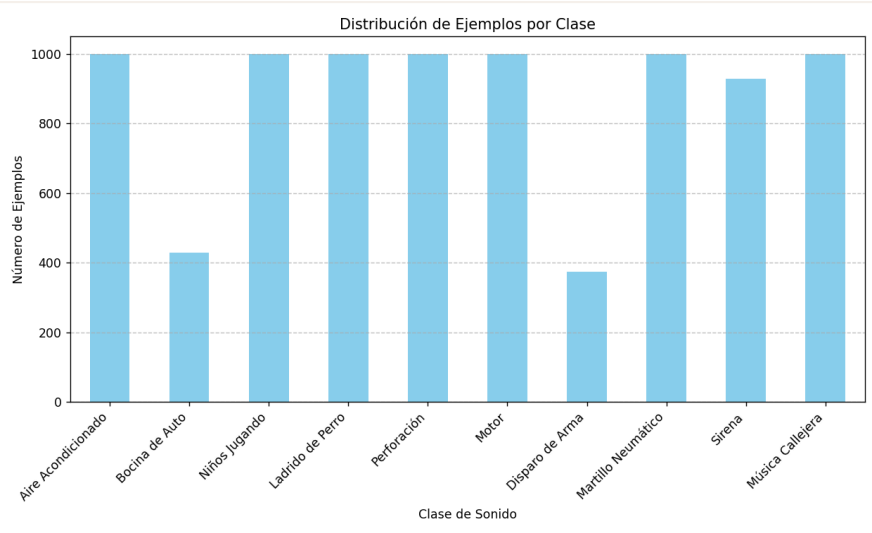
Dataset

El dataset UrbanSound8K fue seleccionado debido a su estructura y relevancia de sus clases en el contexto de aplicaciones de audio. Este dataset contiene 8,732 archivos de audio en formato WAV, distribuidos en 10 clases. Cada archivo tiene una duración de hasta 4 segundos y viene acompañado de un archivo de metadatos en formato CSV que contiene información sobre la clase de sonido, el fold (pliegue) al que pertenece, entre otras cosas.

Clases de sonidos

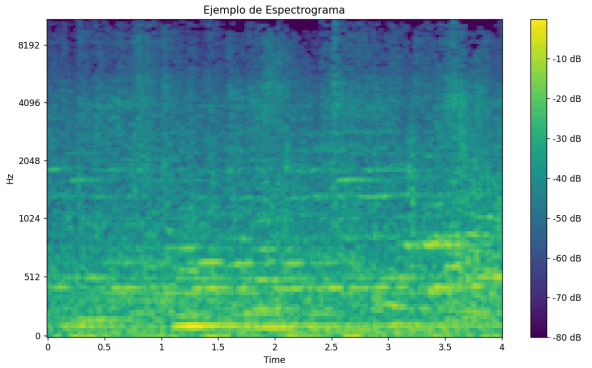
Las 10 categorías de sonido en el dataset son:

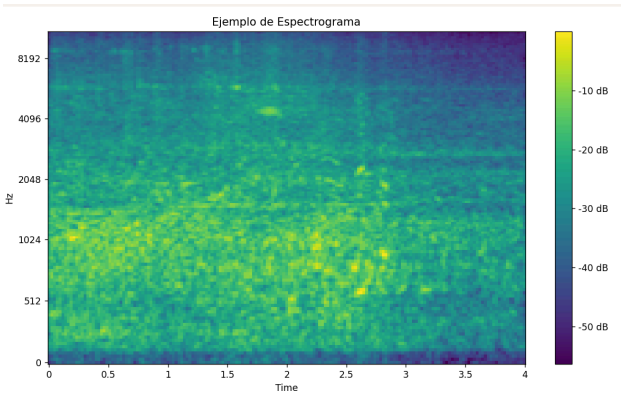
1. Aire acondicionado
2. Bocina de auto
3. Niños jugando
4. Ladrido de perro
5. Perforación
6. Motor
7. Disparo de arma
8. Martillo neumático
9. Sirena
10. Música callejera



Preprocesamiento de audio

Para entrenar el modelo, los archivos de audio fueron transformados en espectrogramas, los cuales son representaciones gráficas de la frecuencia y la amplitud del sonido a lo largo del tiempo. Esta conversión a los espectrogramas permite utilizar redes neuronales convolucionales (CNN), que están diseñadas para trabajar con imágenes, aprovechando su capacidad para detectar patrones visuales en los espectrogramas. La transformación se realizó utilizando la librería 'librosa' para generar imágenes de espectrogramas en escala de grises, que luego fueron almacenadas en una carpeta y se usaron como entrada para el modelo de clasificación.





Arquitectura del Modelo

El modelo desarrollado es una Red Neuronal Convolucional (CNN), diseñada para extraer patrones y características clave en los espectrogramas. La estructura del modelo se compone de:

- Tres capas convolucionales: Cada capa incluye una función de activación 'ReLU' y una capa de max-pooling para reducir la dimensionalidad. Estas capas extraen características espaciales de los espectrogramas, como los patrones de frecuencia, que son relevantes para identificar cada clase de sonido.
- Batch Normalization y Dropout: Se aplicaron después de las capas convolucionales y en las capas densas, para mejorar la estabilidad del modelo y reducir el sobreajuste.
- Capas densas: Al final de las capas convolucionales, el modelo incluye dos capas densas , con la última capa configurada para clasificar en una de las 10 clases.

Esta arquitectura permite que el modelo capte tanto características locales como globales de los espectrogramas, lo cual es esencial para reconocer sonidos complejos y diferenciarlos entre sí.

Entrenamiento y resultados

Configuración de Entrenamiento

Para entrenar el modelo, se utilizó un esquema de validación cruzada de 10 pliegues, donde el dataset se dividió en 10 subconjuntos. Durante el entrenamiento, el modelo se entrena en 9 de estos pliegues y se evalúa en el pliegue restante, repitiendo el proceso para cada uno de los 10 pliegues. Esto ayuda a asegurar que el modelo se evalúe de manera uniforme en todo el dataset y que se reduzca el riesgo de sobreajuste.

Parámetros clave de entrenamiento:

- Tamaño de batch: 32
- Learning rate: 0.0005
- Épocas: 10 por pliegue

Resultados

Al finalizar el entrenamiento en los 10 pliegues, se obtuvo un promedio de:

- Pérdida de validación: 2.6815
- Precisión de validación: 56.70%

Estos resultados indican que el modelo es capaz de aprender características útiles para la clasificación de sonidos urbanos, pero también sugieren que hay margen para mejoras en términos de precisión y estabilidad en la predicción.

Análisis de Resultados

La precisión del modelo de 56.70% en el conjunto de validación sugiere que el modelo está identificando patrones en los datos, aunque no con una precisión alta. La complejidad y la variabilidad del audio en entornos urbanos puede contribuir a la dificultad de clasificación, así como posibles limitaciones en la capacidad del modelo para captar todos los matices necesarios en un espectrograma.

Implementación de mejoras y Resultados

Tras identificar áreas de mejora en el modelo, se implementaron los siguientes cambios para optimizar su desempeño:

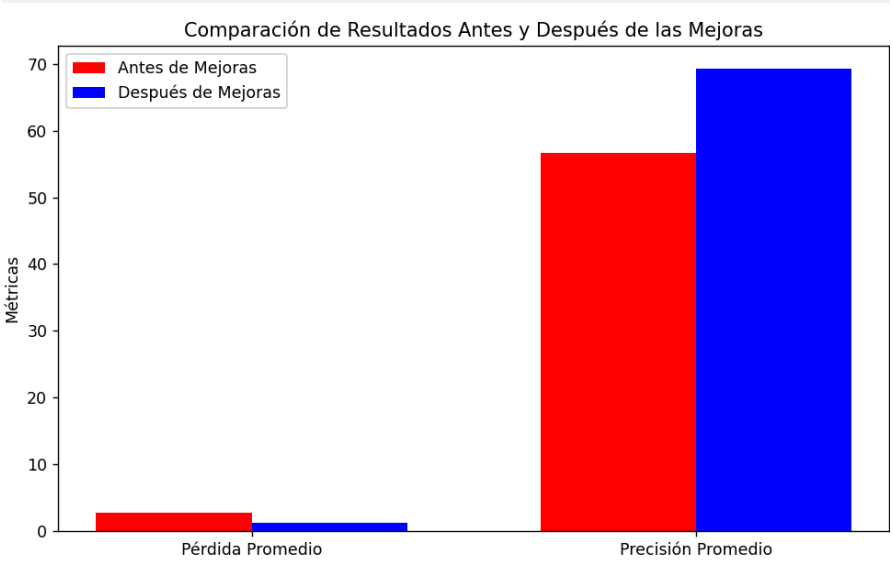
1. Incremento del número de épocas de entrenamiento
    - El modelo aún mostraba aprendizaje activo en las últimas épocas, lo que sugería que no había alcanzado su máximo potencial.
    - Se cambiaron las épocas de entrenamiento de 15 a 20
    - Se observó una mayor estabilización en la pérdida y precisión hacia las últimas épocas. Esto permitió al modelo aprender patrones más complejos, mejorando la precisión promedio de 56.70% a 69.25%.
2. Reducción de la taza de aprendizaje (learning_rate)
    - Una taza de aprendizaje más baja ayuda al modelo a realizar ajustes más finos durante las épocas finales, reduciendo las oscilaciones.
    - Se ajustó el valor de learning_rate de 0.0005 a 0.0001
    - Se logró una disminución significativa en la pérdida promedio, que pasó de 2.6815 a 1.1453, lo que indica un mejor ajuste a los datos de validación.

3. Entrenamiento con un mayor número de filtros convolucionales:
    - Aumentar la capacidad de las capas convolucionales permite al modelo capturar características más complejas de los espectrogramas.
    - Se incrementaron los filtros en las capas convolucionales (1, 32, 64, 128).
    - El modelo mejoró su capacidad para extraer características relevantes, lo que impactó positivamente en su desempeño general.

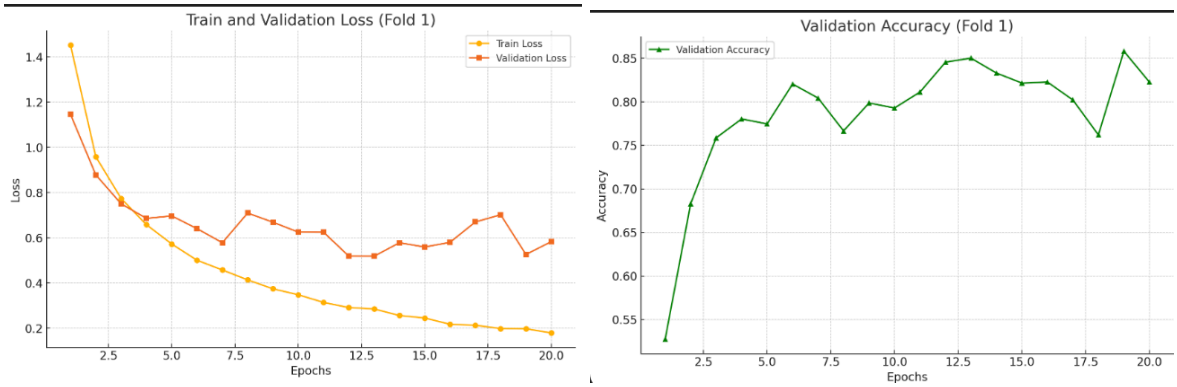
Resultados Finales

Pérdida promedio (Val Loss): Se redujo de 2.6815 a 1.1453, representando un ajuste más preciso del modelo a los datos de validación.

Precisión promedio (Val Accuracy): Aumentó de 56.70% a 69.25%, reflejando una mejora significativa en la capacidad del modelo para clasificar correctamente los espectrogramas.



Esta tabla comparativa representa los resultados antes y después de las mejoras realizadas dentro del modelo, podemos observar que gracias a estos cambios la eficiencia del modelo mejoró significativamente, lo que nos ayudó a realizar mejores predicciones.



Las gráficas presentadas de pérdida y precisión provienen de datos utilizados del Fold 1. Estas representan dos temas diferentes:

1. Gráfica de pérdidas (Train y Validation): Muestra cómo la pérdida disminuye durante las épocas tanto para los datos de entrenamiento como de validación, reflejando el aprendizaje del modelo.
2. Gráfica de precisión de validación: Indica cómo mejora la precisión en el conjunto de validación a medida que avanza el entrenamiento.

Implementación del Modelo en una Aplicación Web

Como parte de este proyecto, se desarrolló una aplicación web que permite cargar archivos de audio, con lo que en conjunto al modelo de clasificación desarrollado, determina el tipo de sonido urbano presente en el archivo.

La aplicación fue implementada utilizando diversas tecnologías:

- Se desarrolló una interfaz intuitiva donde el usuario puede subir un archivo de audio en formato WAV.
- El modelo de clasificación, entrenado y optimizado, fue integrado utilizando Flask. Cuando un usuario carga un archivo de audio, este se convierte automáticamente en un espectrograma, el cual es procesado por el modelo.
- El resultado devuelto es una predicción para el usuario, mostrando la categoría de sonido detectada (“ladrido de perro”, “bocina de auto, etc…).

Esta implementación demuestra cómo el modelo de deep learning puede integrarse en una aplicación real, proporcionando herramientas útiles para la clasificación automática de sonidos en escenarios urbanos.

Flujo de la Aplicación:
El usuario carga un archivo de audio desde su computadora.
El servidor convierte el audio en un espectrograma.
El modelo clasifica el espectrograma en una de las 10 categorías predefinidas.
La categoría de sonido detectada es mostrada al usuario en la interfaz.

Ejemplo de Predicción:

Audio de YouTube grabado desde la computadora convertido a formato .WAV para su uso en la aplicación web y en el modelo: https://www.youtube.com/watch?v=9KScQy6sQUQ

Conclusión

La implementación de estas mejoras permitió optimizar el modelo, abordando problemas de sobreajuste y aumentando su capacidad para aprender características útiles. Estas modificaciones demostraron ser efectivas al reducir la pérdida promedio y aumentar la precisión promedio en los datos de validación cruzada.