UNIVERSIDAD DEL VALLE DE GUATEMALA

CC3092 - Deep Learning y Sistemas Inteligentes

Sección 21

Ing. Luis Alberto Suriano

Proyecto Final

Clasificación de sonidos Urbanos

Maria Marta Ramírez	#21342
Diego Leiva	#21752
Jose Pablo Orellana	#21970

GUATEMALA, 15 de noviembre del 2024

Descripción del Problema

El entorno urbano está repleto de una variedad de sonidos que van desde ruidos de tráfico y obras de construcción hasta sonidos de la naturaleza y actividades humanas. La clasificación automática de estos sonidos es un campo de investigación emergente con aplicaciones en la recuperación de multimedia y en la mejora de la calidad de vida en las ciudades. Identificar automáticamente los sonidos urbanos puede ser de gran utilidad para la gestión de la contaminación acústica, el análisis del ambiente urbano y la mejora de la seguridad pública. Sin embargo, el progreso en esta área se ha visto obstaculizado por dos problemas principales: la falta de un sistema de categorización o taxonomía común y la escasez de datos reales, etiquetados y de alta calidad.

Nuestro proyecto busca abordar estos desafíos mediante el uso de redes neuronales para la clasificación de sonidos urbanos utilizando el conjunto de datos *UrbanSound*. Este dataset incluye 27 horas de audio de entornos urbanos reales con etiquetas en 10 clases de sonido, tales como sirenas, música, bocinas y ladridos de perro, que representan fuentes de ruido comunes en ciudades.

Análisis

La clasificación de sonidos ambientales implica múltiples retos técnicos. Los sonidos urbanos no solo son diversos sino también complejos: pueden estar presentes de manera simultánea o cambiar en intensidad y duración, lo cual dificulta su identificación precisa. Además, la acústica urbana es caótica y los sonidos de diferentes fuentes pueden solaparse, creando mezclas difíciles de diferenciar. Por ejemplo, el sonido de una sirena puede quedar enmascarado por otros ruidos de tráfico o el sonido de maquinaria.

El conjunto de datos que utilizaremos *UrbanSound* presenta una solución parcial a este problema, ya que contiene una taxonomía inicial de sonidos urbanos clasificados en categorías relevantes para el análisis acústico de las ciudades. Este dataset incluye grabaciones con etiquetas en 10 clases específicas, lo cual permite entrenar modelos de clasificación supervisada, optimizando la capacidad del modelo para identificar y etiquetar estos sonidos con mayor precisión.

Propuestas de la Solución

Red CNN

Para abordar el problema de clasificación de sonidos urbanos, proponemos implementar un modelo de redes neuronales convolucionales (CNN) que tome como entrada representaciones espectrales de los sonidos del dataset *UrbanSound*. Las CNNs son adecuadas para este tipo de tarea debido a su capacidad para captar características espaciales y temporales en los espectrogramas de audio, facilitando la diferenciación entre distintos tipos de sonidos.

De manera que la solución será aplicada mediante el siguiente procedimiento:

1. **Preprocesamiento de los datos**: Los audios serán convertidos en espectrogramas, que son representaciones visuales de las frecuencias a lo largo del tiempo. Esta transformación convierte la señal de audio en una imagen que puede ser procesada por la CNN.
2. **Diseño e Implementación del Modelo**: Desarrollaremos una arquitectura de red CNN optimizada para el análisis de espectrogramas. Esto implica varias capas de convolución y pooling que extraen características de alta y baja frecuencia, permitiendo una clasificación más precisa de los sonidos.
3. **Entrenamiento y Evaluación**: Entrenaremos el modelo en el dataset *UrbanSound*, ajustando hiperparámetros clave y aplicando técnicas como la regularización y el aumento de datos para mejorar la generalización del modelo. El rendimiento del modelo será evaluado utilizando métricas como la precisión, sensibilidad y especificidad en la clasificación de cada clase de sonido.
4. **Optimización y Mejora**: A medida que avanzamos en el entrenamiento, consideraremos otras técnicas avanzadas de Deep Learning, como la transferencia de aprendizaje y el ajuste fino, para mejorar el rendimiento del modelo en clases de sonido menos representadas en el conjunto de datos.

Red LSTM

Otra de las soluciones que se consideraron implementar es el uso de modelos basado en redes neuronales recurrentes de tipo LSTM, estar redes son ideales para procesar secuencias temporales, como lo podría ser una señal de audio. Todo gracias a su capacidad para capturar patrones temporales complejos y relaciones a largo plazo.

En este caso, tomaremos en cuenta el explorar el uso de espectrogramas de Mel como el de coeficientes MFCC como vectores de características. Ambas representaciones ofrecen ventajas específicas, los espectrogramas de Mel capturas la energía en diferentes bandas de frecuencia a lo largo del tiempo, mientras que los coeficientes MFCC abstraen características relevantes del espectro del audio. Para el caso de las LSTM el procedimiento planteado incluiría:

Preprocesamiento de Datos:

- Las señales de audio serán transformadas en espectrogramas de Mel y coeficientes MFCC, aplicando técnicas de normalización y enmascaramiento (de tiempo y frecuencia) para robustecer el modelo ante variaciones en los datos.
- Ambas representaciones serán evaluadas durante el desarrollo para determinar su impacto en el desempeño del modelo.

Diseño del Modelo:

- Se desarrollará una arquitectura de LSTM bidireccional con varias capas ocultas para procesar secuencias temporales derivadas de los espectrogramas de Mel o MFCC.
- Se integrará un mecanismo de atención, lo que permitirá al modelo centrarse en las partes más relevantes de las secuencias para mejorar la precisión de las predicciones.
- Adicionalmente, se emplearán capas de Batch Normalization y Dropout para mejorar la estabilidad y evitar el sobreajuste.

Entrenamiento del Modelo:

- El modelo será entrenado en el conjunto de datos UrbanSound, ajustando parámetros como el número de capas, tamaño de lote y tasa de aprendizaje.
- Se buscará mitigar posibles desbalances de clases en el conjunto de datos mediante técnicas como el aumento de datos y ajustes ponderados en la función de pérdida.

Evaluación y Optimización:

- Se evaluará el desempeño del modelo utilizando métricas como precisión, sensibilidad y F1-score, con un análisis específico de su efectividad en clases minoritarias.
- Los resultados obtenidos con espectrogramas de Mel y coeficientes MFCC serán comparados para seleccionar la representación más adecuada.
- En caso de resultados insuficientes, se considerará el uso de transferencia de aprendizaje con modelos pre entrenados.

Descripción de la Solución

La solución implementada para abordar el problema de clasificación de sonidos urbanos se basó en el uso de una red neuronal recurrente LSTM (Long Short-Term Memory) en combinación con espectrogramas de Mel como representación de las señales de audio. Este enfoque fue seleccionado debido a la capacidad de las redes LSTM para procesar secuencias temporales y la efectividad de los espectrogramas de Mel para capturar información frecuencial de los sonidos.

Preprocesamiento de Datos

El primer paso en la implementación fue el procesamiento de las señales de audio para convertirlas en representaciones adecuadas para el modelo. Las señales de audio fueron tratadas de la siguiente manera:

- **Conversión a espectrogramas de Mel:** Cada archivo de audio fue transformado en un espectrograma de Mel utilizando parámetros ajustados para optimizar la extracción de características, como una tasa de muestreo de 22,050 Hz, 128 bandas de Mel, y un rango de frecuencias de 20 Hz a 11,025 Hz.
- **Máscaras de tiempo y frecuencia:** Durante el entrenamiento, se aplicaron técnicas de enmascaramiento para simular ruidos y variaciones en los datos, mejorando la capacidad del modelo para generalizar.
- **Normalización:** Los espectrogramas fueron normalizados para garantizar que las características estuvieran en un rango uniforme, lo que facilita el entrenamiento del modelo y reduce el sesgo de entrada.

La clase AudioDataset fue utilizada para gestionar este preprocesamiento, asegurando la preparación eficiente de los datos y permitiendo su integración con el modelo LSTM.

Arquitectura del Modelo

La arquitectura del modelo implementado se diseñó específicamente para trabajar con datos secuenciales y extraer características temporales complejas de los espectrogramas de Mel:

- **Capas LSTM bidireccionales:** Se emplearon capas LSTM bidireccionales para capturar patrones tanto hacia adelante como hacia atrás en las secuencias temporales, aumentando la capacidad del modelo para comprender el contexto global de los sonidos.
- **Mecanismo de Atención:** Un módulo de atención fue integrado para identificar las partes más relevantes de las secuencias, lo que permitió al modelo enfocarse en segmentos clave de los espectrogramas al realizar la clasificación.
- **Capas totalmente conectadas:** La salida de la LSTM se pasó por dos capas lineales con Batch Normalization y Dropout para mejorar la estabilidad del entrenamiento, reducir el sobreajuste y mapear las características extraídas a las clases de salida.
- **Capa de salida:** Finalmente, una capa completamente conectada generó las probabilidades de pertenencia para cada clase de sonido.

Entrenamiento y Evaluación

El modelo fue entrenado utilizando el conjunto de datos UrbanSound, que incluye diversas clases de sonidos urbanos. Para optimizar el desempeño, se emplearon las siguientes estrategias:

- **Ajuste de hiperparámetros:** Se realizaron ajustes en parámetros como el número de capas LSTM, el tamaño de las capas ocultas, la tasa de aprendizaje y el tamaño del lote.
- **Pérdida y optimización:** Se utilizó la función de pérdida Cross-Entropy y el optimizador Adam para actualizar los pesos del modelo durante el entrenamiento.
- **División de datos:** Los datos se dividieron en conjuntos de entrenamiento, validación y prueba para evaluar el desempeño del modelo en datos no vistos.

Herramientas Aplicadas

Librerías y Frameworks

- Pandas: Para manejar y procesar los metadatos del dataset UrbanSound8K, facilitando la manipulación de información como etiquetas y rutas de los archivos de audio.
- NumPy: Para realizar cálculos y manipular estructuras de datos numéricas como arrays.
- Matplotlib y Seaborn: Para crear visualizaciones, como curvas de precisión y pérdida, matrices de confusión y distribuciones de datos.
- Scikit-learn:
    - Leave-One-Group-Out (LOGO): Usado para implementar la validación cruzada basada en folds.
    - Métricas como classification report y confusion matrix para evaluar el rendimiento del modelo.
- PyTorch:
    - Para construir y entrenar el modelo LSTM.
    - Uso de torch.nn para implementar capas como LSTM, Linear, BatchNormalization y Dropout.
    - Torchaudio: Para el procesamiento de señales de audio, como la conversión a espectrogramas de Mel y la normalización.
    - CUDA: Para acelerar el entrenamiento del modelo utilizando GPU.
- Keras (TensorFlow):
    - Para construir y entrenar los modelos CNN.
    - Data Augmentation con ImageDataGenerator para aumentar la variedad del dataset CNN.
    - Callbacks como EarlyStopping y ReduceLROnPlateau para mejorar la generalización y optimizar el proceso de entrenamiento.
- Librosa: Utilizado en ambos modelos para el procesamiento de audio, como la carga de señales de audio, generación de espectrogramas de Mel, extracción de coeficientes MFCC y augmentación de datos.
- TQDM: Para visualizar barras de progreso durante la ejecución de tareas largas.

Procesamiento de Datos

- Generación de Espectrogramas de Mel: Para el modelo CNN, se generaron espectrogramas de Mel a partir de los archivos de audio usando librosa. Estos se guardaron como imágenes en un directorio y sirvieron como entrada para el modelo CNN.
- Coeficientes MFCC: Para el modelo CNN, se extrajeron los coeficientes MFCC (64 coeficientes) de cada archivo de audio, que se usaron como características numéricas para entrenar el modelo.
- Augmentación de Datos:
    - En audio (Librosa): Se aplicaron transformaciones como time-stretching, preénfasis y adición de ruido para aumentar la diversidad del dataset.
    - En imágenes (Keras): Rotaciones, traslaciones y flips aplicados a los espectrogramas para mejorar la robustez del modelo CNN.

Validación y Métricas

- Validación Cruzada (10-Fold): Aplicada a los modelos LSTM para evaluar la robustez y generalización del modelo.
- Métricas:
    - Accuracy, Precision, Recall, F1-Score: Calculadas por clase y promedio para evaluar la efectividad del modelo.
    - Curvas de Pérdida y Precisión: Visualización del rendimiento en términos de pérdida y precisión a lo largo de las épocas.
    - Matrices de Confusión: Para analizar los errores en predicciones por clase.

Optimización y Regularización

- Regularización con Dropout: Implementado en los modelos LSTM y CNN para evitar el sobreajuste.
- Reducción de Tasa de Aprendizaje: Utilizando ReduceLROnPlateau en los modelos LSTM.
- Early Stopping: Detención temprana basada en la métrica de validación para evitar entrenamientos innecesarios.

Resultados

Modelo CNN con espectrograma lineal
Este primer modelo obtuvo un accuracy general del 54.53% tras 20 épocas de entrenamiento

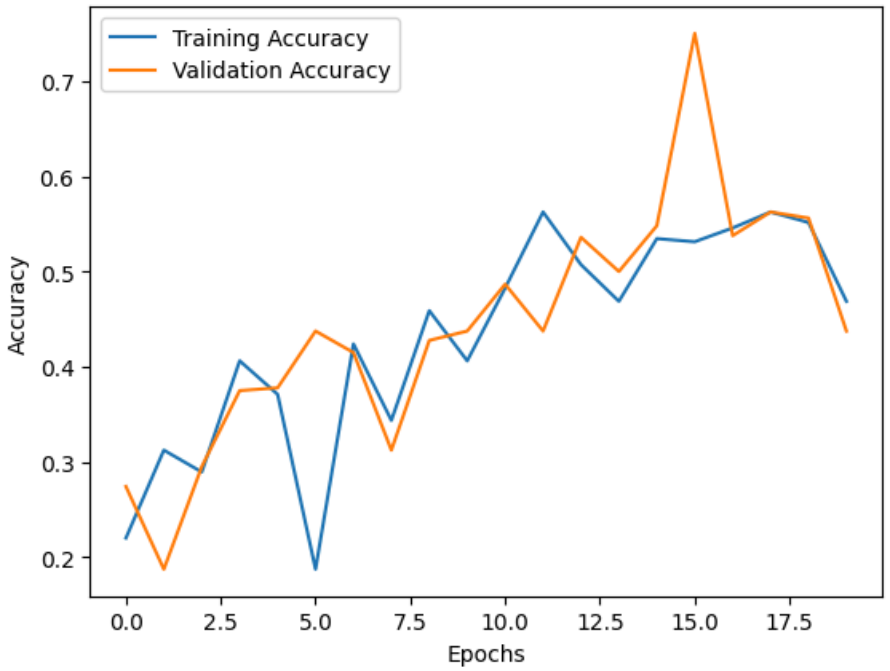

Figura 1: Accuracy de validación y entrenamiento a lo largo de las épocas para modelo CNN

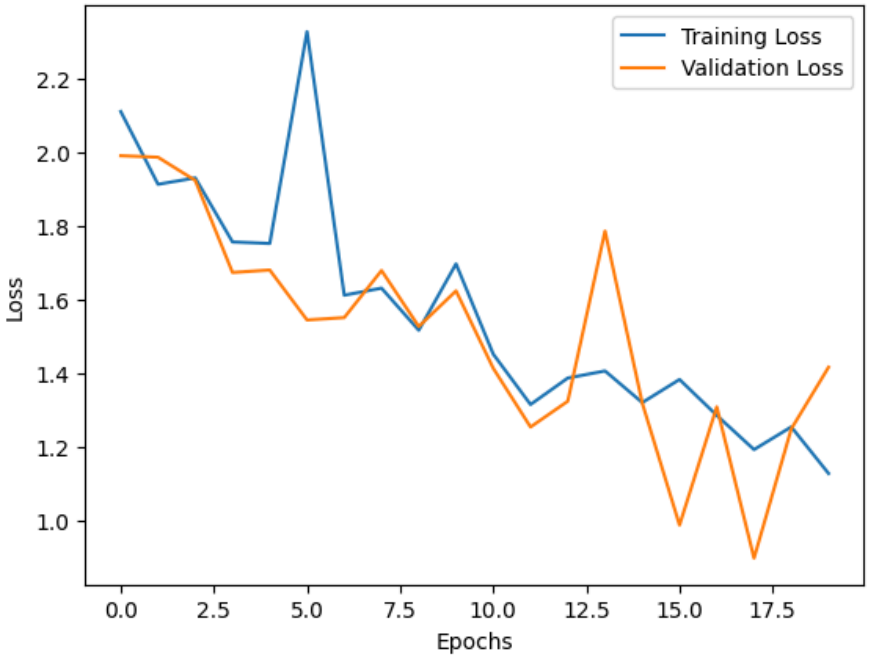

Figura 2: Pérdida de validación y entrenamiento a lo largo de las épocas para CNN

Estos gráficos muestran que pasada la época 17 el modelo comienza a presentar overfitting. Dado el mal rendimiento del modelo se procedió a realizar otro agregando a los features los MFCCs de los audios.

Modelo CNN con espectrograma y MFCCs

Este nuevo modelo que introduce el uso de MFCCs (Coeficientes Cepstrales en las Frecuencias de Mel) obtuvo un incremento significativo en rendimiento llegando a un 94.62% de accuracy tras 50 épocas de entrenamiento.

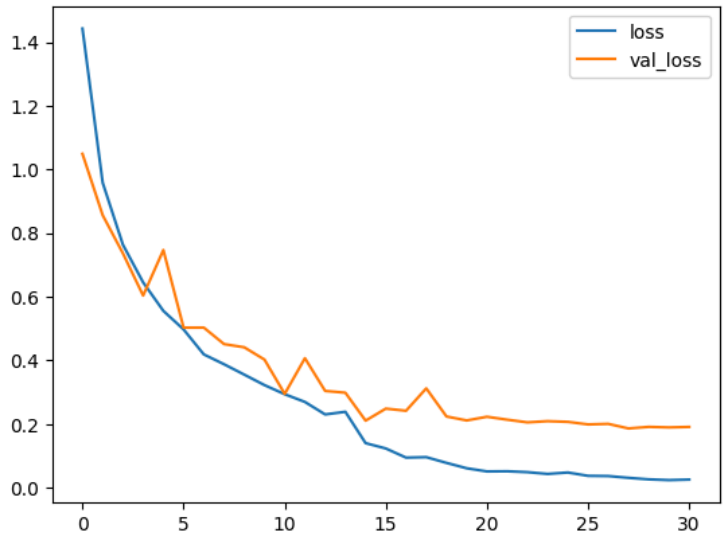

Figura 3: Perdidas de entrenamiento y validación a lo largo de las épocas para CNN con MFCCs

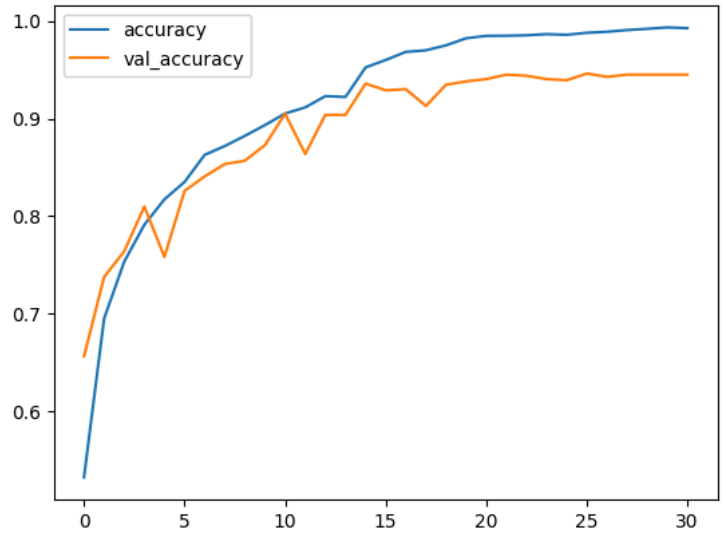

Figura 4: Accuracy de validación y entrenamiento a lo largo de las épocas para CNN con MFCC

Sin embargo, el rendimiento tan bueno del modelo se debe realmente a la forma en que se realizo el split de datos. Salomon (2014), creador del conjunto de datos de UrbanSound8K indica que al aplicar un split entrenamiento/prueba aleatorio sobre todo el conjunto de datos provoca que los resultados esten inflados, volviéndolos inutilizables y poco confiables. Esto debido a que los datos vienen separados en 10 Folds, donde es necesario aplicar una

validacion cruzada de 10 Folds. De lo contrario, al utilizar un split aleatorio se corre el riesgo de que datos asociados queden tanto en el set de entrenamiento como en el de pruebas y por ende el rendimiento general parezca mayor.

Modelo LSTM con espectrograma de mel y MFCC

Tomando en cuenta la consideracion anterior, se decidio realizar un nuevo modelo LSTM, para probar la validacion cruzada de 10 folds. Para cumplir con el objetivo de la validacion cruzada Salomon (2014) indica que es necesario entrenar el modelo con los datos de 9 folds y validarlo con el restante. Este proceso se repite hasta que se haya validado el modelo con cada fold.

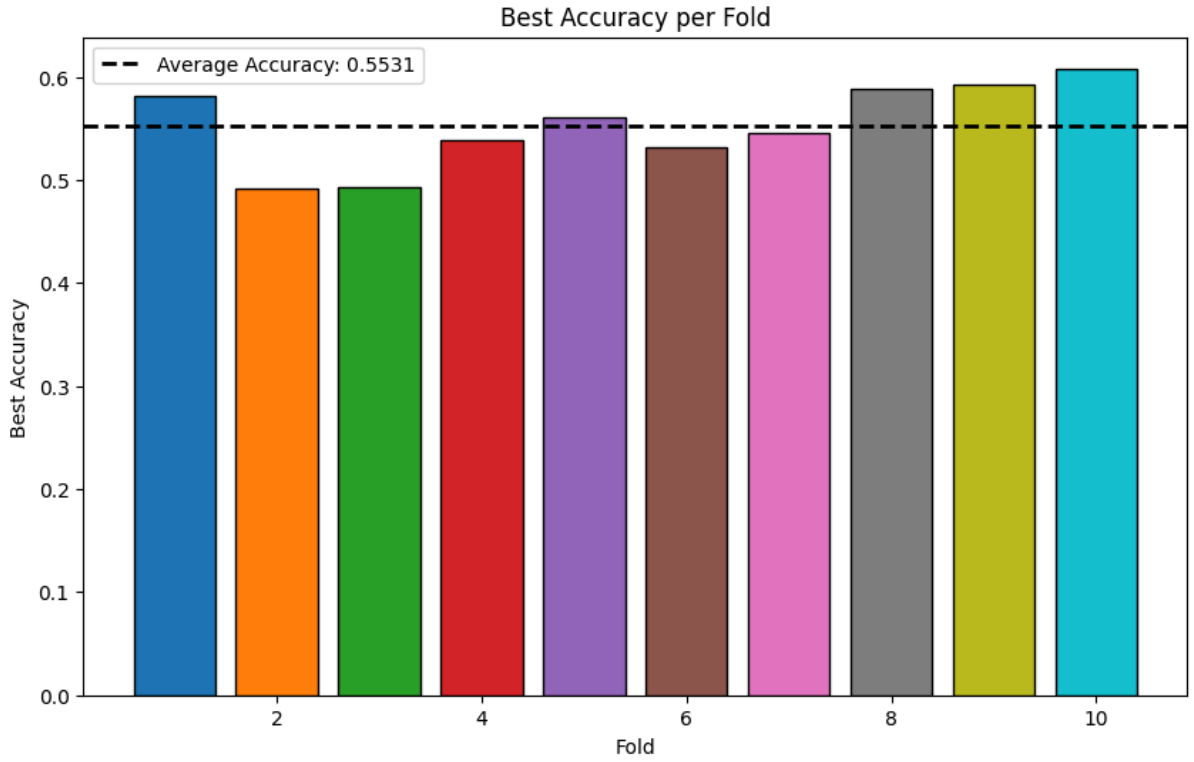

Figura 5: Accuracies de cada Fold para modelo LSTM v1

Por lo tanto se puede decir que el modelo a nivel general tendría un rendimiento con un accuracy del 0.55%

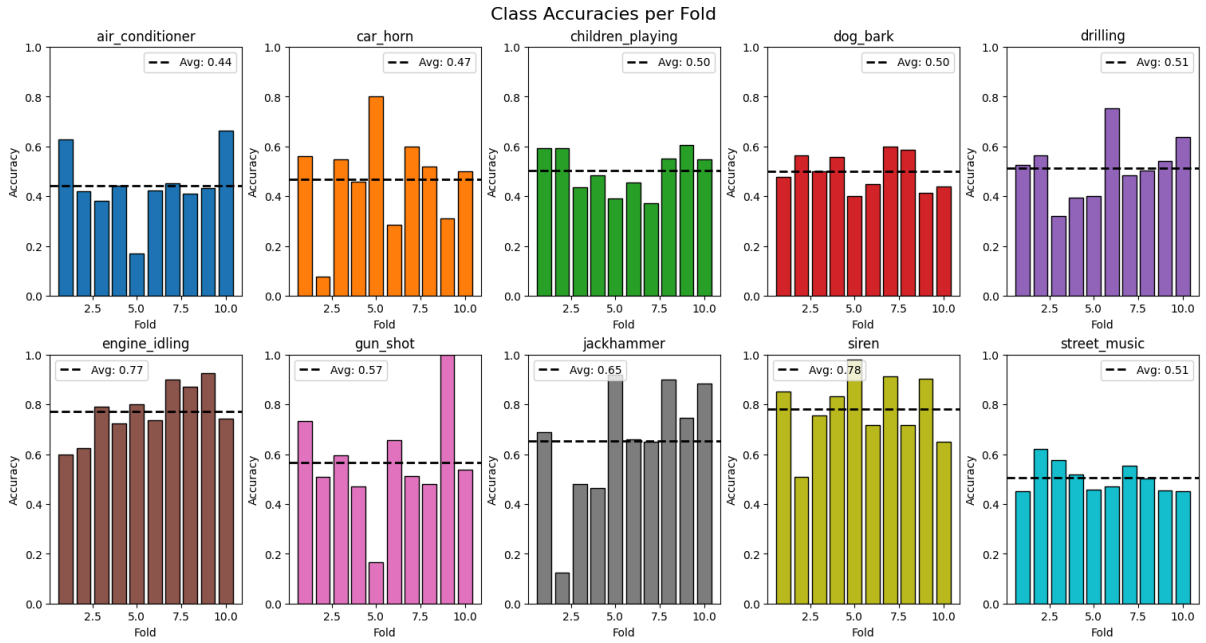

Figura 6: Accuracies por clase para LSTM v1

En este gráfico se puede apreciar como esta la distribución de accuracies para cada una de las clases del conjunto de datos

Por fines prácticos se mostraran los rendimientos promedio a través de todos los Folds

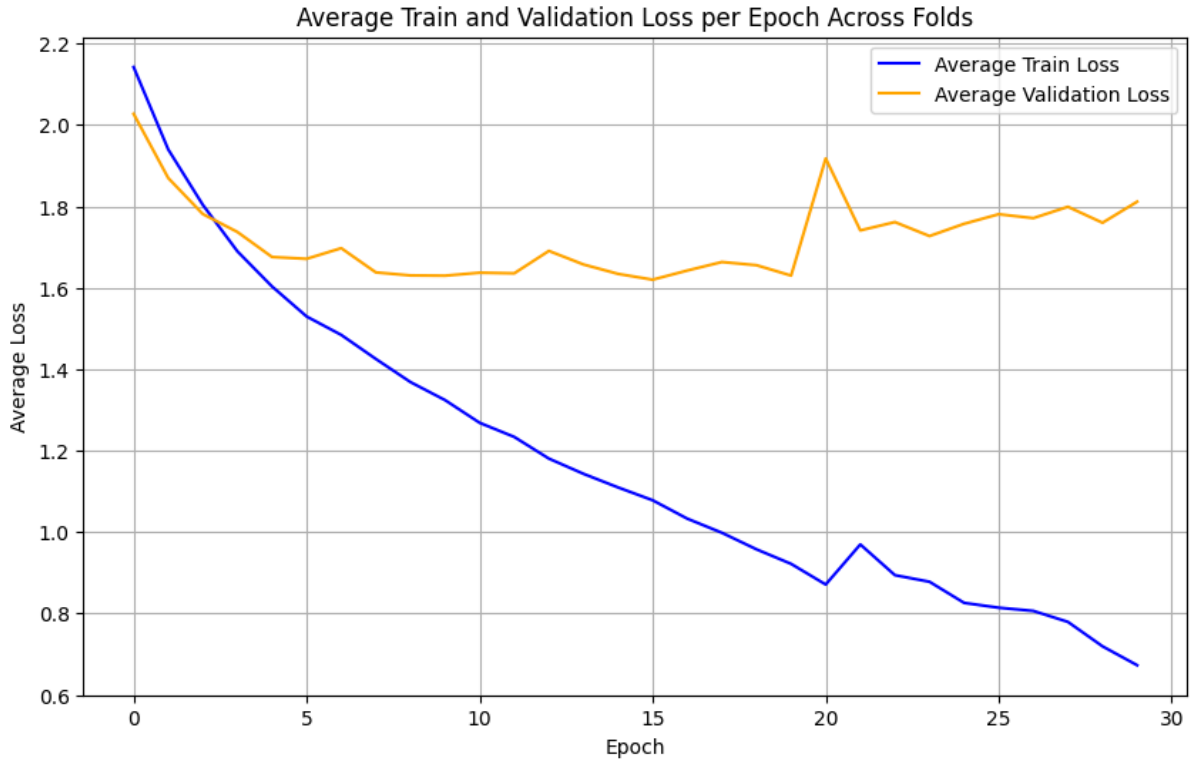

Figura 7: Pérdidas promedio de entrenamiento y validación a través de las épocas para LSTM v1

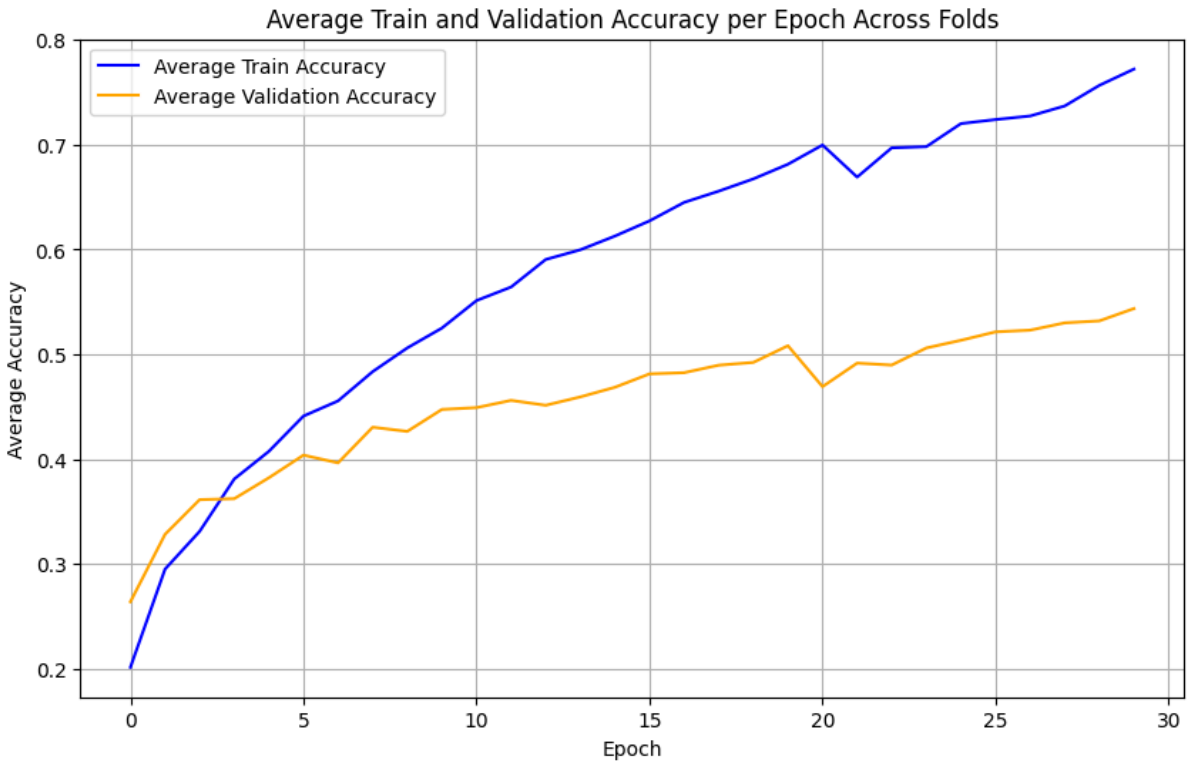

Figura 8: Accuracy promedio de entrenamiento y validación a través de las épocas para LSTM v1

Estos gráficos representan una aproximación del rendimiento del modelo a nivel general si se entrena con todos los datos disponibles.

Modelo LSTM con espectrograma de mel y mecanismo de atención

Finalmente el modelo utilizado usó mecanismos de regularización como dropout, schedulers y mecanismo de atención para mejorar su rendimiento.

Nuevamente se aplicó una validación cruzada de 10 Folds para obtener el rendimiento general del modelo.

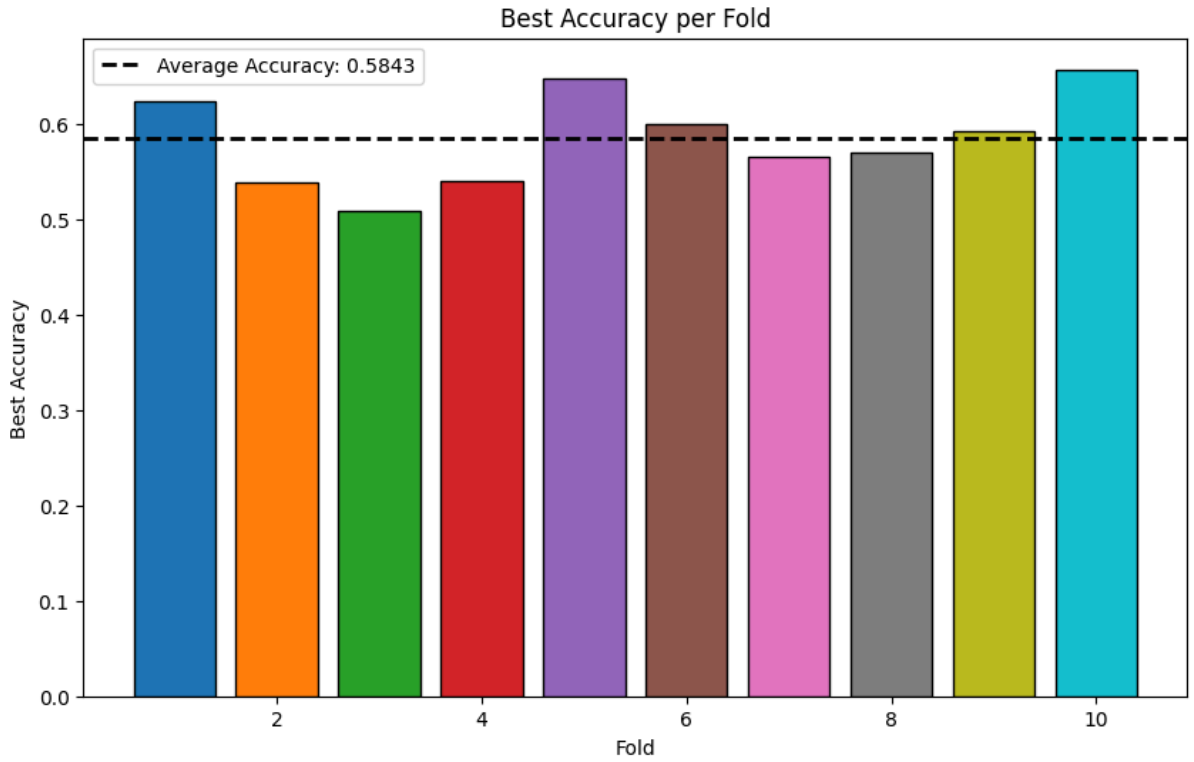

Figura 9: Accuracy por Fold para el modelo LSTM v2

En este caso vemos un incremento en el promedio de accuracy para el modelo, sin embargo sigue siendo menor a un rendimiento aceptable.

Para entender el por que de estos porcentajes es necesario revisar la distribución de precisiones a través de las clases

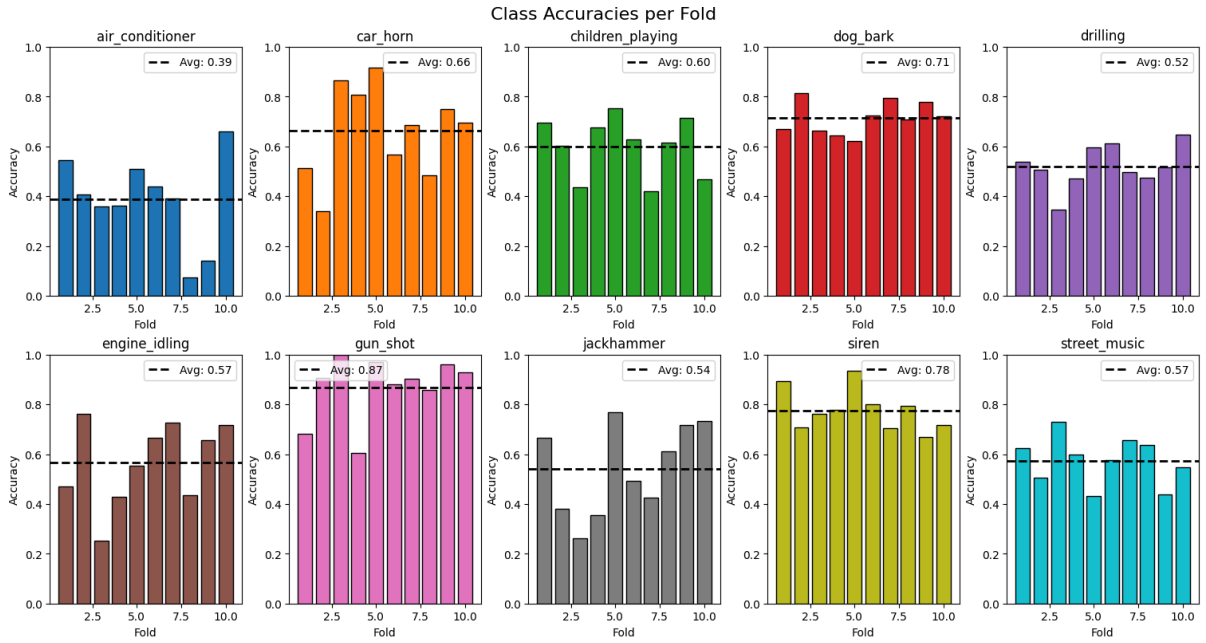

Figura 10: Accuracies por clase para LSTM v2

En este caso se observa que hay 3 clases que sobresalen al resto “gun_shot”, “dog_bark” y “siren”

Esto se debe a que estos sonidos son los que tienen patrones más distintivos respecto al resto de clases en cuanto a espectrogramas de mel se refiere.

Si vemos un ejemplo de cómo se ven los espectrogramas para las diferentes clases de sonido

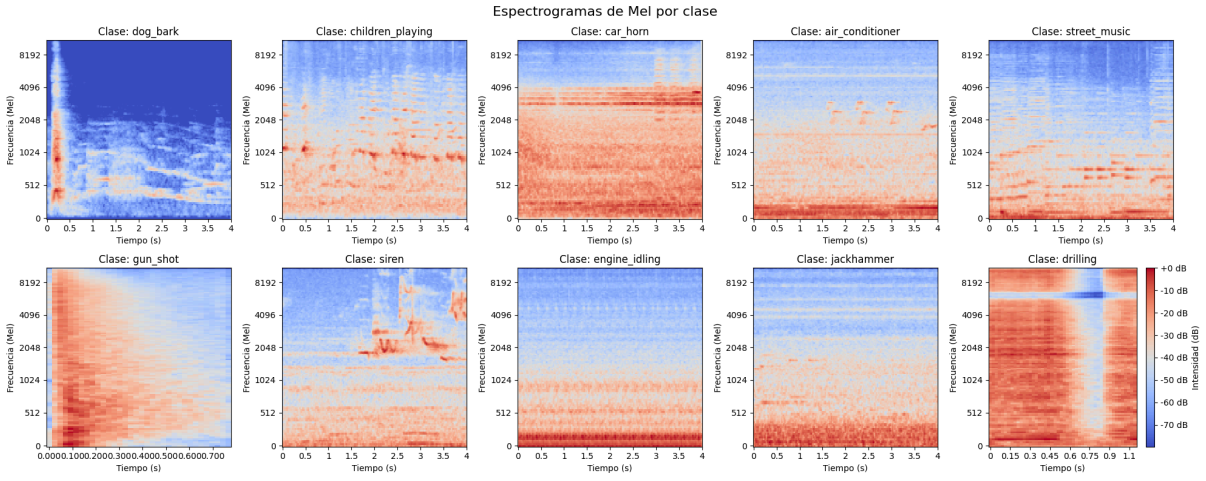

Figura 11: Espectrogramas de mel por clase

Es evidente que hay una diferencia visual apreciable entre esas clases sobresalientes respecto al resto de clases.

Por ende en promedio el modelo tiene mayor capacidad de distinguir disparos, sirenas y perros ladrando

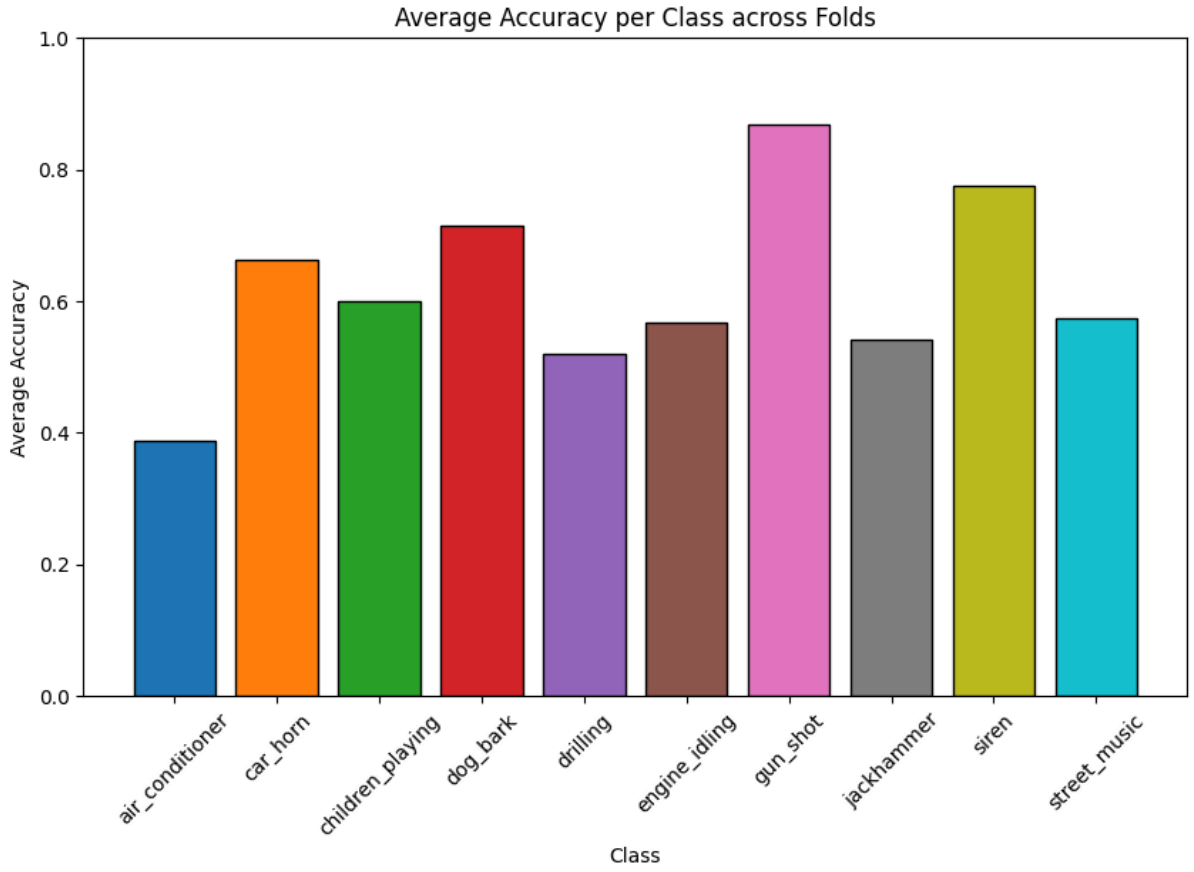

Figura 12: Promedios de accuracy por clase para el modelo LSTM v2

Etrando a mas detalles de como fue el rendimiento del modelo en todos los folds, podemos evaluar como fue evolucionando la perdida y el accuracy

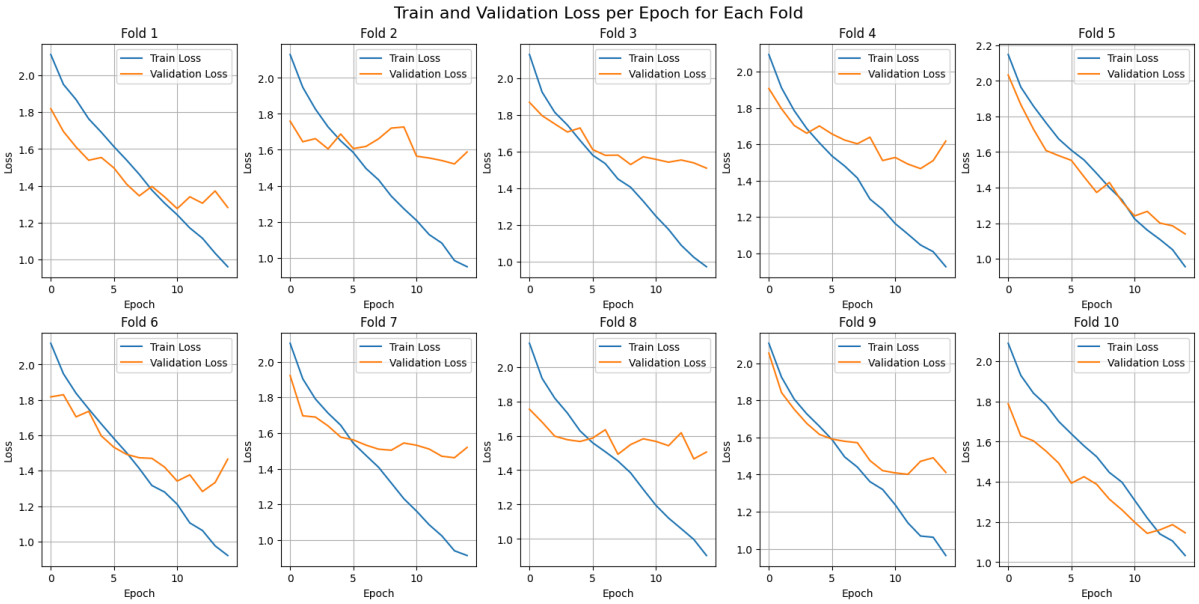

Figura 13: Perdida de entrenamiento y validacion para cada Fold de LSTM v2

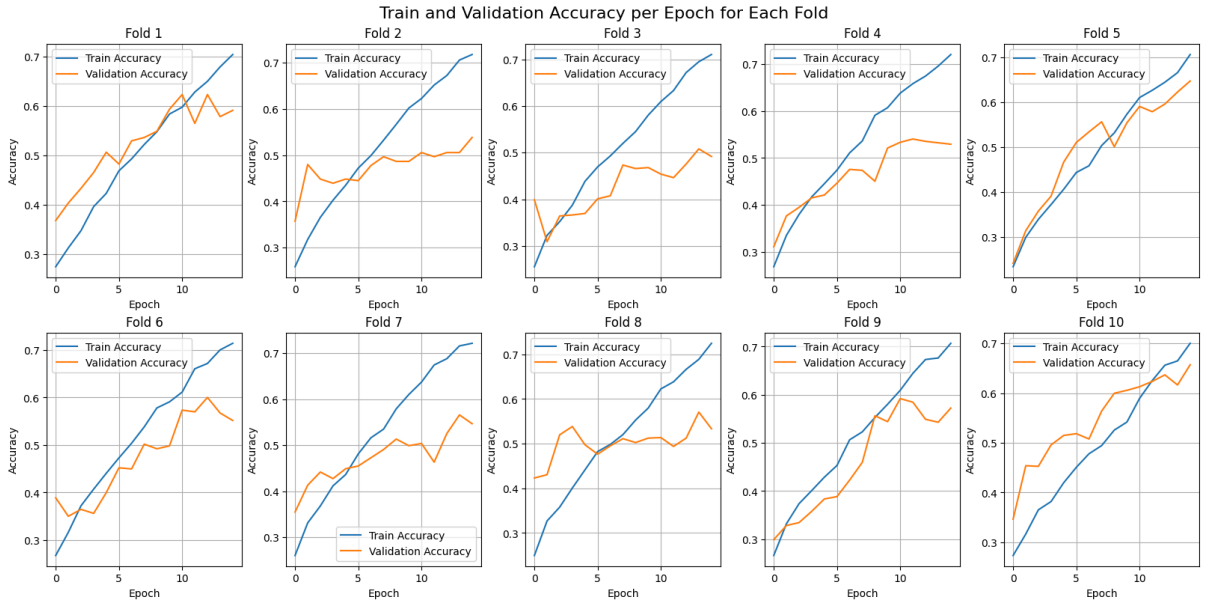

Figura 14: Accuracy de entrenamiento y validación para cada Fold de LSTM v2

Como se observa en estos gráficos, el modelo comienza aprendiendo bien los datos, sin embargo a partir de la época 5 el modelo comienza a alcanzar su límite.

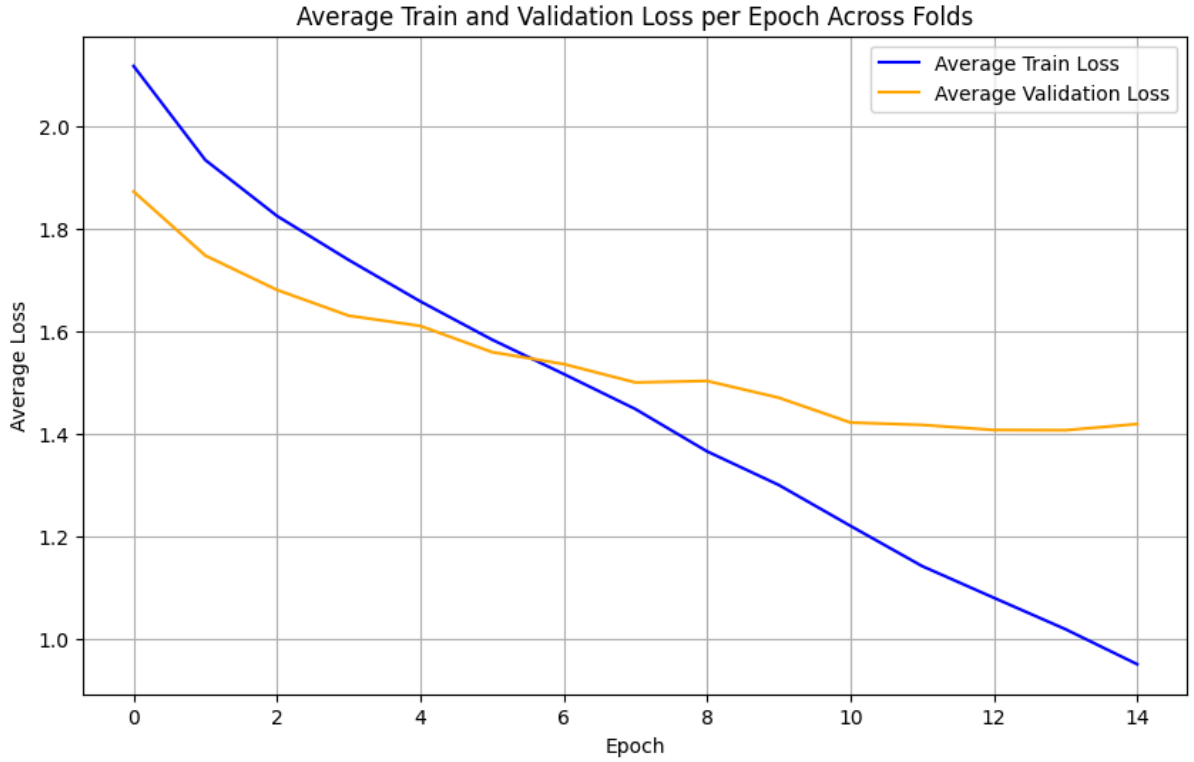

Figura 15: Perdida y validacion promedio a traves de las epocas para LSTM v2

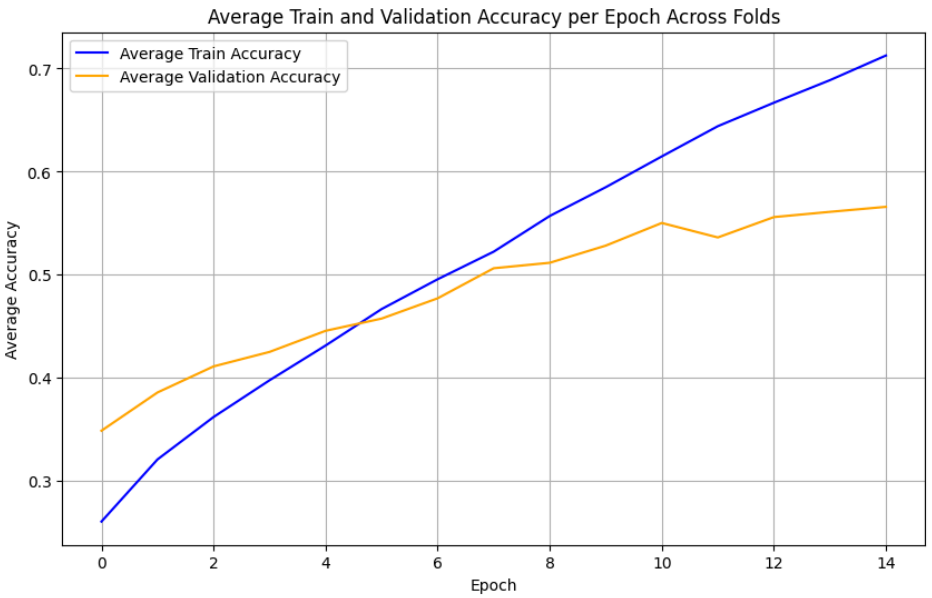

Figura 16: Accuracy promedio de entrenamiento y validacion para LSTM v2

Finalmente podemos observar que a comparacion del LSTM v1 este nuevo modelo tiene una mejor significativa

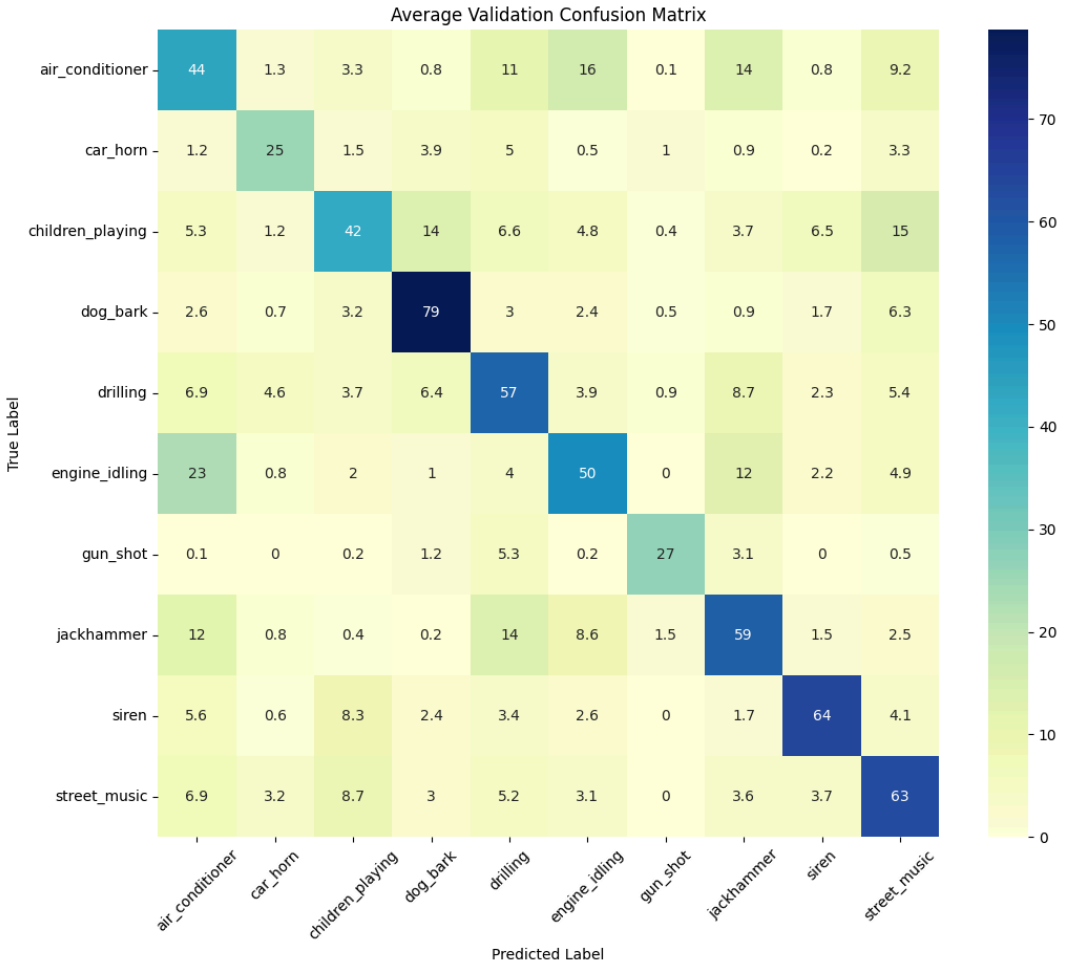

Figura 17: Matriz de confusión general del modelo.

Conclusión

El modelo desarrollado demuestra un desempeño sólido, considerando la arquitectura y los recursos utilizados. Los resultados son particularmente destacados al clasificar sonidos que poseen características distintivas, como sirena, car_horn, dog_bark y gun_shot. Esto se debe en parte al uso de características basadas en la transformación MEL, que resalta las diferencias en las frecuencias de estos sonidos.

Sin embargo, el modelo enfrenta mayores dificultades para diferenciar sonidos con características similares, como aire acondicionado y sonido de motor. Estas categorías presentan patrones acústicos más cercanos, lo que complica su clasificación.

A pesar de estas limitaciones, las métricas generales del modelo reflejan un buen desempeño. La matriz de confusión evidencia la capacidad del modelo para distinguir adecuadamente entre las clases más diferenciadas, lo que valida su efectividad para tareas de clasificación de sonidos urbanos en escenarios específicos.

Bibliografía

Deruty, E. (2022, 15 diciembre). Intuitive understanding of MFCCs - Emmanuel Deruty - Medium. *Medium*. https://medium.com/@derutycsl/intuitive-understanding-of-mfccs-836d36a1f779

Gartzman, D. (2021, 11 diciembre). Getting to Know the Mel Spectrogram - Towards Data Science. *Medium*. https://towardsdatascience.com/getting-to-know-the-mel-spectrogram-31bca3e2d9d0

GeeksforGeeks. (2024, 26 junio). *Melfrequency Cepstral Coefficients (MFCC) for Speech Recognition*. GeeksforGeeks. https://www.geeksforgeeks.org/mel-frequency-cepstral-coefficients-mfcc-for-speech-recognition/

Introducción a la memoria a corto-largo plazo (LSTM). (s. f.). MATLAB & Simulink. https://la.mathworks.com/discovery/lstm.html#:~:text=Las%20LSTM%20son%20una%20excelente,las%20palabras%20de%20una%20oraci%C3%B3n.

librosa.feature.melspectrogram — librosa 0.10.2.post1 documentation. (s. f.). https://librosa.org/doc/main/generated/librosa.feature.melspectrogram.html

¿Qué es el espectrograma? (s. f.). https://emastered.com/es/blog/what-is-spectrogram

Roberts, L. (2024, 17 enero). Understanding the Mel Spectrogram - Analytics Vidhya - Medium. *Medium*. https://medium.com/analytics-vidhya/understanding-the-mel-spectrogram-fca2afa2ce53

Ruiz, I. V. M. (2013, 28 marzo). *MFCCs*. Ivan Vladimir Meza Ruiz. https://turing.iimas.unam.mx/~ivanvladimir/posts/mfcc/

Salamon, J., Jacoby, C., & Bello, J. (2014). A Dataset and Taxonomy for Urban Sound Research. *22nd ACM International Conference On Multimedia*. http://www.justinsalamon.com/uploads/4/3/9/4/4394963/salamon_urbansound_acmmm14.pdf

Anexos

Video

https://youtu.be/aiLlmZv3xEY

Repositorio

https://github.com/LeivaDiego/Urban-Sound-Classification.git