Escola Tècnica Superior d'Enginyeria ETSE-UV

Programa de Doctorat en Tecnologies de la Informació, Comunicacions i Computació.

Vniversitat de València

Doctoral Thesis

Classification of Sound Scenes and Events in Real-World Scenarios with Deep Learning Techniques

Author:

Javier Naranjo Alcázar

Advisors:

Pedro Zuccarello

Máximo Cobos

November, 2020

Escola Tècnica Superior d'Enginyeria ETSE-UV

Programa de Doctorat en Tecnologies de la Informació, Comunicacions i Computació.

Vniversitat de València

Doctoral Thesis

Classification of Sound Scenes and Events in Real-World Scenarios with Deep Learning Techniques

Author: Javier Naranjo Alcázar

Advisors: Pedro Zuccarello and Máximo Cobos

November, 2020

This work has been supported/funded by the "Doctarados Industriales" fellowship DIN2018-009982 from the Spanish Ministry of Science, Innovation and Universities

A mis abuelos

Agradecimientos

Esta tesis es el resultado de un gran esfuerzo realizado por un gran número de personas que componen o han compuesto el equipo de Visualfy. Gracias por el apoyo y ánimo de todos ellos. Todos han contribuido, en mayor o menor medida, al resultado que se presenta en esta memoria. Todo esto empezó hace alrededor de 3 años cuando Manel y Ángel me dieron la oportunidad de trabajar en el proyecto Visualfy. Desde entonces no he parado de crecer como persona y profesional dentro de la empresa, por ello, os estoy agradecido.

No obstante, me gustaría destacar al equipo "del algoritmo" o de "Inteligencia Artificial" (IA). Un equipo pequeño, pero que se enfrenta a las problemáticas de la IA en entornos de aplicación reales con todos los quebraderos de cabeza que eso conlleva. A Jose por su predisposición a aprender IA y en ayudar en todo momento. A Sergi, por trabajar codo con codo en todo momento y ser compañero y amigo dentro y fuera de la empresa. Por último, a Pedro Zuccarello, codirector de esta tesis y líder del equipo. Por enseñarnos tanto: plantear problemas, proponer soluciones, cuestionarnos resultados, y tantas enseñanzas que no pongo porque sería interminable. Tu formación nos ha hecho mejores profesionales, te lo aseguro. En este párrafo, no podía faltar Máximo Cobos, también codirector. Muchas gracias por cumplimentar la formación recibida durante esta tesis. Creo que realizar esta tesis en este contexto con vosotros dos, me ha permitido aprender lo mejor de cada mundo (empresarial y académico) y estar preparado para el día de mañana.

Por último, agradecer a todo mi entorno cercano. Primero a los diseñadores que me han ayudado a ilustrar esta tesis (Álex y Pedro), pero especialmente a Ilham por ayudarme y enseñarme en todo momento. Amigos de La Vila por preguntarme periódicamente como me iba y animándome. A mis amigos y compañeros de piso de València, que conocí durante la carrera por ser vía de escape y de diversión en momentos en los que los resultados no acompañaban. A mis padres y abuelos que siempre me marcaron el camino de la enseñanza y lo poderosa que era. A mi hermana, por ser mi hermana. A mi novia, por entender lo complicado que es hacer una tesis, y aún así apoyarme y animarme en todo momento.

Ellos también son coautores de esta tesis de una manera u otra.

Resumen

La gran cantidad de datos generados por la sociedad en los últimos años ha permitido que las soluciones basadas en Inteligencia Artificial (IA) puedan posicionarse en el estado del arte actual. Muchas de estas soluciones son conocidas como basadas en datos o *data-driven* en inglés, ya que estos sistemas necesitan un gran volumen de datos para ser entrenados. La naturaleza de estas soluciones permite que puedan ser desplegadas en multitud de contextos, como por ejemplo, la automatización de procesos industriales, la conducción autónoma o el diseño de asistentes domésticos. Muchos de estos escenarios contemplan soluciones basadas en el Internet de las Cosas o *Internet of Things* (IoT) en inglés. A día de hoy, cada vez más dispositivos electrónicos están conectados a Internet, lo que permite un abanico de aplicaciones desconocidas hasta hace poco tiempo. Por tanto, pequeños dispositivos, como por ejemplo un reloj, pueden incorporar soluciones de IA para realizar una tarea en concreto, si bien podría o no ser necesaria la conexión a Internet para el funcionamiento de la solución de IA. La conexión a Internet permite la interacción por parte del usuario con el dispositivo y el intercambio de información entre dispositivos para una mejora o ampliación de los posibles servicios que pueden ofrecerse. Otros ejemplos claros de IoT pueden ser los asistentes domésticos por comandos de voz como Alexa© de Amazon o Google Home©. Sin embargo, este tipo de productos también suponen, en muchas ocasiones, una barrera invisible de la que no somos conscientes. En este sentido, la interacción vía voz por parte del usuario requerida por estos dispositivos puede desplazar a personas sordas o con pérdida auditiva. Sin embargo, cada vez más, la comunidad científica está trabajando en el análisis de eventos sonoros mediante técnicas de IA, lo que podría resultar en sistemas inteligentes y asistentes domésticos capaces de facilitar el día a día de este segmento de la poblicación y contribuyendo a conseguir así una accesibilidad real para todos ellos.

La audición es una de las principales formas de interacción con la naturaleza por parte de los seres humanos. El campo que se encarga del estudio de algoritmos que sean capaces de la obtención de información a partir de datos de audio es conocido como Audición por Computador (APC) o *Machine Listening* en inglés. Cabe destacar que un gran número de personas sordas o con pérdida auditiva podrían beneficiarse de soluciones o productos que implementen técnicas de APC. De acuerdo con la Organización Mundial de la Salud (OMS), 466 millones de personas tienen problemas de audición. Además, cien mil millones de personas se encuentran en riesgo de padecer pérdida auditiva. Esto se debe al mal uso de auriculares o la exposición a altos niveles de decibelios en distintos lugares como discotecas o estadios deportivos. Por lo que respecta a la población más mayor, alrededor de un tercio de la gente por encima de 65 años padece de pérdida de audición. Todo esto nos indica que una gran parte de la población se podría beneficiar de soluciones basadas en APC.

Visualfy es una *startup* valenciana cuyo principal objetivo es la creación de soluciones basadas en IA que permitan una mayor accesibilidad a la población sorda o con pérdida auditiva. Actualmente, dispone de dos productos conocidos como Visualfy Home (VH) y

Visualfy Places (VP). El primero de ellos (VH) se puede definir como un asistente doméstico accesible para personas sordas o con pérdida auditiva. El sistema está formado por un *Hub* principal y 3 detectores o micrófonos que se deben colocar en las habitaciones que el usuario quiere tener monitorizadas. El sistema es capaz de registrar el audio, segmentarlo, procesarlo y clasificarlo acorde a un conjunto de categorías/clases/alertas como, por ejemplo, "alarma de incendios" o "bebé llorando". Además, el sistema es capaz de comunicarse con multitud de dispositivos como el teléfono móvil o bombillas inteligentes para que el usuario disponga de más fuentes de información. Cabe destacar que todo el procesado del audio y la clasificación se realizan en el *Hub* sin ningún tipo de conexión a Internet. La única información que transmite el *Hub* vía Internet (IoT) es el tipo de alerta que se ha detectado para poder notificar al usuario conforme a la configuración personal del sistema (bombillas, móvil, ...). El segundo producto (VP) está pensado en hacer más accesibles espacios públicos o de gran concurrencia (teatros, bibliotecas, etc.). De acuerdo con la Agenda 2030, todos los edificios públicos deben ser accesibles para todo el mundo. En este caso, el sistema está compuesto por el *Hub* y una serie de periféricos luminosos como lámparas o bombillas. El funcionamiento de este sistema, por lo que respecta a las técnicas de APC, es el mismo.

En general, estos sistemas se encuentran desplegados en entornos reales no controlados. Esto provoca que el sistema, ya sea VH o VP, deba enfrentarse a clases de sonidos o situaciones para las que no ha sido entrenado. Este fenómeno se conoce como el problema de Reconocimiento de Conjunto Abierto u *Open-Set Recognition (OSR)* en inglés. Además, la finalidad de estos productos es el reconocimiento de patrones de audio muy concretos como puede ser una alarma de incendios o un timbre. Esto ocasiona que cada sistema deba ser entrenado de forma particular para cada usuario, ya sea un cliente final (VH) o un edificio (VP). Como se ha dicho anteriormente, muchas de las soluciones actuales de IA han mostrado resultados muy satisfactorios cuando disponen de una gran cantidad de datos para ser entrenadas. Al ser inviable la adquisición de miles de muestras de audio de un mismo timbre por parte del usuario, el sistema debe ser entrenado con muy pocas muestras (2 o 3). Este fenómeno se conoce como Aprendizaje con Pocos Disparos o *Few-Shot Learning (FSL)* en inglés. Por último, hay que destacar que la respuesta del sistema debe ser la más rápida posible. En este contexto, los tiempos de ejecución son cruciales. Un retraso a la hora de la notificación puede suponer una confusión y una mala experiencia de uso para el usuario. La necesidad de diseñar sistemas lo más simples posible desde el punto de vista computacional se conoce como soluciones de baja complejidad o *low-complexity models* en inglés. Así pues, resulta especialmente interesante en el escenario considerado el diseño de arquitecturas de redes neuronales capaces de mejorar la precisión sin que suponga un incremento considerable en el número de parámetros entrenables de las mismas.

La mayoría de soluciones en el estado del arte suponen una conversión del audio a una representación 2D mediante algún tipo de transformación tiempo-frecuencia. Este pre-procesado supone la elección de hiperparámetros concretos como el tamaño de ventana, solape, *bins* frecuenciales o *frames* temporales si se deseara obtener un espectrograma (representación 2D) a partir del audio, ya sea un espectrograma convencional o con consideraciones perceptuales mediante el uso de bancos de filtros uniformemente espaciados en la escala Mel. Las soluciones que no emplean estas representaciones bidimensionales y que tienen como entrada directamente las muestras de audio son conocidas como extremo a extremo o *end-to-end* en inglés. La peculiaridad de estos sistemas reside en que el sistema está totalmente compuesto por parámetros entrenables. Así, se consigue evitar el sesgo que puede aparecer a la hora elegir ciertos valores concretos de hiperparámetros, tomando decisiones únicamente a partir del audio en su representación unidimensional. Es por esto que también se ha estimado oportuno experimentar con este tipo de soluciones en esta tesis.

Objetivos

Esta tesis se enmarca en el campo concreto del APC donde el objetivo consiste en clasificar/identificar segmentos de audio/eventos sonoros correspondientes a un patrón definido para poder aportar información a un usuario y así ayudar en la toma de decisiones final. De forma más concreta, este proceso de clasificación se produce en entornos reales donde aparecen las problemáticas de OSR y FSL. Los eventos sonoros poseen, además, una serie de particularidades intrínsecas que dificultan el proceso de clasificación incluso aunque no se dieran las problemáticas previamente mencionadas. Estas peculiaridades son: la polifonía de los sonidos en entornos reales, es decir, varios eventos sonoros se pueden superponer en el mismo instante de tiempo. En un día cotidiano, es muy difícil encontrar momentos del día donde solo escuchemos una única fuente aislada. Los sonidos generales y ambientales, además, no poseen una relación temporal. En otros campos como la voz o la música sí que se puede encontrar una relación basada en la estructura gramatical y en la melodía respectivamente. Asimismo, cada fuente de sonido posee una naturaleza distinta. Un audio puede ser transitorio, como, por ejemplo, un timbre, que solo suena una vez durante un instante corto de tiempo o puede ser estacionario como una alarma de incendios que suena durante un periodo largo de tiempo. Por último, cabe destacar los problemas relacionados al proceso de grabación como pueden ser la adición de ruido de fondo o ruido eléctrico que dificultan, en gran medida, el rendimiento del sistema de clasificación. Así pues, un sistema robusto de clasficación de audio (en un entorno controlado) debe tener en cuenta como mínimo las peculiaridades previamente descritas.

Por tanto, el objetivo principal de esta tesis es la proposición y estudio de sistemas de clasificación de eventos sonoros en entornos reales no controlados (abiertos) donde la clasificación debe realizarse en tiempo real y el conjunto de entrenamiento es escaso, teniendo en cuenta también la posibilidad de utilizar soluciones *end-to-end*.

El objetivo anterior engloba las tres problemáticas explicadas en el Resumen. Por tanto, este objetivo general se puede dividir en tres objetivos más concretos. El primero de ellos consiste en la proposición de sistemas que sean desplegables y funcionen en tiempo real, es decir, que cumplan los requisitos temporales de ejecución que demanda la aplicación. Se proporciona, en primer lugar, una visión general de las soluciones de IA de clasificación de eventos sonoros. Dentro del marco de la IA, podemos encontrar el campo del Aprendizaje Máquina o *Machine Learning* (ML). Dentro de los métodos de ML, podemos además encontrar aquellos basados en Aprendizaje Profundo o *Deep Learning* (DL). Los algoritmos clásicos de ML tienen principalmente un fundamento estadístico y requieren normalmente de una interacción mayor por parte de la persona que los implementa, al menos en cuanto a la selección de características se refiere. A medida que los datos disponibles han ido aumentando, el estado del arte ha ido cambiando y las técnicas más prometedoras son aquellas basadas en DL. Además, requieren una menor interacción por parte de la persona que las implementa, lo que ha propiciado que, cada vez más, investigadores e ingenieros se decanten por este tipo de soluciones. Estas técnicas suelen superar a las soluciones clásicas de ML cuando la base de datos es lo suficientemente amplia y los conjuntos de datos están bien etiquetados.

Las técnicas de DL han mostrado resultados muy prometedores en el campo de la Visión por Computador o *Computer Vision* en inglés. Las soluciones más extendidas en el estado del arte están normalmente basadas en Redes Neuronales Convolucionales o *Convolutional Neural Networks (CNNs)* en inglés. Estas redes están formadas por capas convolucionales cuyo objetivo es la creación de mapas de características o *feature maps* en inglés, a partir de una representación bidimensional (2D) del audio o sobre el audio mismo (1D). Estos mapas de características corresponden a representaciones internas que son utilizadas, finalmente, para la clasificación del audio por la misma CNN. Estas redes han proporcionado resultados satisfactorios cuando la entrada se encuentra en dos dimensiones (2D) como es el caso de una imagen en escala de grises. Por tanto, el primer paso de estas soluciones en el contexto de la APC consite en la transformación del audio de una señal unidimensional a una bidimensional. Para ello, se realiza un estudio por ventanas (la señal se divide en fragmentos más pequeños que se consideran estacionarios) y cada una de ellas se representa en el dominio frecuencial, consiguiendo así una representación bidimensional en el dominio tiempo-frecuencia. Este proceso de transformación es conocido como extracción de características y juega un papel crucial en el comportamiento del clasificador. No obstante, la elección de los hiperparámetros que componen un extractor de características pueden propiciar un cierto sesgo para un problema concreto, imposibilitando la generalización del sistema que permita ser aplicado en otro entorno. Así pues, se decide realizar un estudio de distintas redes neuronales unidimensionales que puedan ser más independientes al contexto APC concreto.

Una práctica común para mejorar el rendimiento del clasificador consiste en la creación de muestras artificiales durante el entrenamiento (*data augmentation* en inglés) para que el sistema disponga de un mayor número de eventos durante el entrenamiento. Otra práctica común, pero muy poco aconsejable en aplicaciones a tiempo real consiste en la creación de múltiples clasificadores independientes entrenados con distintas representaciones de audio, sistemas conocidos como agrupaciones o *ensembles* en inglés. A la hora de reconocer un evento sonoro, este debe ser procesado para la extracción de características multitud de veces (una por cada clasificador), clasificado por cada uno de ellos y, por último, combinar la información de cada uno de ellos para obtener una clasificación final. Es lógico pues, que todo este proceso no constituya una práctica recomendable en un escenario de aplicación real. Por tanto, otro de los objetivos secundarios concretos dentro del marco de trabajo establecido en esta tesis consiste en la elaboración de propuestas orientadas a la mejora de la precisión en la clasificación de eventos sin que éstas alarguen los tiempos de ejecución. En este contexto, resultan especialmente interesantes aquellas técnicas que modifican los bloques convolucionales para obtener mayor precisión sin aumentar de forma significativa el número de parámetros. A modo de resumen, podemos definir este objetivo como la búsqueda de una mejora de la precisión del sistema mediante técnicas que no tengan impacto en los tiempos de ejecución, actuando sobre el diseño de la arquitectura. Para la comparación de diversas soluciones, nos basamos en el número de parámetros entrenables que la componen.

Otro objetivo concreto de esta tesis es la proposición de sistemas de IA que sean capaces de trabajar en entornos abiertos no controlados (OSR). Debido a la naturaleza estadística de las soluciones basadas en ML, el problema del OSR se ha conseguido mitigar, en cierta medida, mediante la aplicación de técnicas que complementan a algoritmos clásicos, como las Máquinas de Vectores Soporte o *Support Vector Machines* (SVMs) en inglés. Actualmente, existe muy poca literatura respecto a soluciones de DL que mitiguen la problemática del OSR. Sin embargo, su impresionante desempeño en problemas de clasficación incita al estudio y proposición de soluciones de DL que tengan en cuenta este fenómeno.

Por último, el problema del FSL, no ha captado la atención de la comunidad científica hasta la aparición de su utilidad en los sistemas de reconocimiento facial. Se puede apreciar cierta semejanza entre el problema de reconocer una cara y el de reconocer un patrón concreto de audio en cuanto a la disposición de datos. Por este motivo, con la necesidad de una solución para esta aplicación, las contribuciones basadas en DL para mitigar el problema del FSL se han incrementado considerablemente. Además de la proposición de una solución de FSL para patrones de audio, se realiza un estudio relativamente amplio del estado del arte en FSL. En esta tesis se ha decidido englobar los objetivos de OSR y FSL en uno conjunto, es decir, la solución propuesta debe mitigar las dos problemáticas a la vez. De hecho, las condiciones particulares del escenario de aplicación considerado en esta tesis industrial implican la aparición de ambas problemáticas de forma conjunta.

Respecto a la redacción de esta tesis, la estructura de la misma se define como la modalidad de compendio de artículos. Los anexos corresponden a tres publicaciones realizadas en el marco de esta tesis en revistas de primer cuartil. Los capítulos de la misma explican y detallan de forma amplia las problemáticas que se afrontan en los artículos que componen el compendio (Capítulos 1 y 2). Además, se enumeran las contribuciones y se exponen las conclusines extraídas en los mismos (Capítulos 3 y 4).

Metodología

Para llevar a cabo esta tesis, se ha realizado, en primer lugar, un estudio exhaustivo del estado del arte de todas las problemáticas que se van a afrontar. Por lo que respecta al despliegue en tiempo real, se ha estudiado qué representaciones 2D del audio se proponen en la literatura en soluciones de clasificación de eventos sonoros, aunque estas no tengan en cuenta dicha consideración. Existe un gran número de contribuciones en este campo de estudio. Dependiendo del proceso de grabación del audio (estéreo o mono), las posibles representaciones 2D pueden ser distintas. La mayoría de ellas se basan en un espectrograma lineal y un posterior escalado empleando un banco de filtros en concreto, cuyo objetivo es emular el sistema auditivo humano (*Mel, Gammatone, CQT, ...*). Basándose en las imágenes RGB, se han propuesto representaciones que combinan varias representaciones 2D para generar una representación multicanal del audio. Por otro lado, la disponiblidad de redes neuronales públicas entrenadas ha permitido la creación de representaciones novedosas del audio. El audio es procesado por la red neuronal y esta genera una representación distinta a las basadas en bancos de filtros clásicos. Esta técnica es conocida como transferencia de conocimiento o *transfer learning* en inglés. En ocasiones esta solución no es muy efectiva si se quiere desplegar en tiempo real, ya que estas redes que transfieren su conocimiento suelen ser muy profundas al haber sido previamente entrenadas con bases de datos muy extensas. Como se ha mencionado con anterioridad, no existe mucha literatura que tenga en cuenta el despliegue en tiempo real de la red. Sin embargo, se han estudiado las contribuciones propuestas aunque no tengan en cuenta dicha limitación, es decir, aquellas soluciones propuestas en el estado del arte que intentan mitigar las limitaciones del las CNN.

La técnica de compresión y excitación de canal, o *squeeze-excitation technique* (SE) en inglés, es una de las soluciones propuestas cuyo objetivo es el de mejorar la precisión de las CNNs. Estas técnicas realizan un recalibrado de los mapas de características o *feature maps* en inglés. La finalidad es la de aplacar las limitaciones en la creación de los mapas de características internos que forman parte de la red. Por otro lado, el método de aprendizaje es un factor determinante a la hora del entrenamiento en CNNs. Las primeras CNNs estaban diseñadas como un conjunto de capas convolucionales implementadas de forma secuencial y una capa final de clasificacíon conocida como totalmente-conectada o *fully-connected* en inglés. No obstante, la aparición del aprendizaje residual y su popularidad en los sistemas de visión por computador han extendido su interés también su interés a los sistemas de audio. La idea principal en la que se basan las redes residuales es que es mucho más simple para la red aprender una función residual en un bloque convolucional en lugar de una función de mapeo no referenciada, como en las CNNs secuenciales convencionales. En esta tesis se estudia como la fusión de ambas técnicas (SE y aprendizaje residual) puede mejorar la clasificación de las CNNs en tareas de clasificación de audio. Para ello, se estudian algunas configuraciones propuestas en el estado del arte y se proponen dos más configuraciones para mejorar los resultados de clasificación.

Para estudiar la contribución de cada solución diseñada se deben definir una serie de métricas y de escenarios de experimentación. Cuando se pretende analizar el comportamiento de una solución propuesta donde todas las situaciones son conocidas por el sistema, se ha decidido emplear la métrica llamada exactitud, *accuracy* en inglés. Además, para aportar mayor conocimiento en cuanto a las diferencias existentes entre distintas alternativas propuestas, se ha realizado un test estadístico de *McNemar*. Se trata de un test estadístico de 1 contra 1 que permite discernir si el comportamiento real de dos soluciones es el mismo o no. En consecuencia, si se pretende analizar la diferencia de comportamiento entre dos redes neuronales distintas, en primer lugar, se visualiza la exactitud de cada una de ellas y, posteriormente, se les realiza este test estadístico. Por lo que respecta al tamaño de la red (restricción de baja complejidad), se ha analizado el número de parámetros de cada solución para así discernir un compromiso entre exactitud y complejidad de la red.

El estado del arte del OSR o el FSL no es específico del dominio del audio. Como se ha mencionado previmente, el FSL atrajo un mayor interés de la comunidad científica al proporcionar soluciones para la aplicación del reconocimiento facial. Concretamente, las técnicas FSL estudiadas han mostrado unos resultados prometedores en el dominio de la imagen, *computer vision*. Algunas contribuciones proponen redes neuronales experimentales que son entrenadas por parejas o en conjuntos de tres (*triplets*), modificaciones de funciones de coste de la red neuronal (*Ring Loss, Center Loss, ...*) o *transfer learning*. Por otro lado, el OSR consiste en la modificación del sistema de IA para que sea capaz de enfrentarse a situaciones/clases desconocidas de forma eficaz. Por tanto, el estudio de estos campos no es específico del dominio del audio. Por ello, es necesario estudiar y proponer modificaciones si se quieren obtener resultados prometedores en el dominio del audio también, ya que el contexto no es el mismo y el tamaño de las bases de datos suele ser considerablemente menor.

Para analizar la contribución de las soluciones propuestas en esta tesis en los objetivos de FSL y OSR, se necesita un entorno de experimentación muy específico. Como se ha explicado previamente, el FSL viene determinado por el número de ejemplos disponibles en fase de entrenamiento y el OSR por el número de situaciones desconocidas por el sistema una vez ha sido entrenado. Para simular un contexto donde conviven las dos problemáticas en el dominio del audio, se ha decidio diseñar una base de datos de audio específica. Esta base de datos está compuesta por patrones de audio muy concretos que deben ser reconocidos por el sistema y por eventos de audio genéricos que deben ser rechazados. Es decir, si la red recibe como entrada un patrón conocido, debe reconocer de qué patrón en concreto se trata. Por otra parte, si recibe un audio que no corresponde a ningún patrón, el sistema debe clasificar dicho audio como desconocido, es decir, rechazarlo (consideración OSR). La base de datos está compuesta por 24 patrones distintos (alarmas de incendios, timbres domésticos, ...) y 10 clases de audio genéricos (aplausos, bocina del coche, tecleo, ...). Ésta está diseñada para poder realizar diferentes experimentos que emulan distintos escenarios, dependiendo de las consideraciones de FSL y OSR. Para poder analizar el impacto del FSL, la base de datos está implementada para que el sistema pueda ser entrenado con cuatro, dos o una muestra de cada patrón de audio que quiera ser detectado (tres escenarios distintos). Esto es conocido como número de disparos. Por otro lado, por lo que respecta a la consideracón OSR, se debe tener en cuenta la apertura del problema, u *openness* en inglés. Esta métrica permite saber cual es la relación entre situaciones conocidas-desconocidas a las que se enfrenta el sistema. El valor de *openness* varía entre 0 y 1. Un valor de 0 indica que el sistema no se enfrentaría a ninguna situación desconocida. A medida que el valor se incrementa, el sistema se enfrenta a un mayor número de situaciones desconocidas. La base de datos está configurada de tal forma que hay tres valores distintos de *openness*, generando así tres escenarios distintos. La configuración de distintos escenarios, tanto en la problemática de FSL como en la OSR, permite una búsqueda en cuadrícula o *grid search* en inglés. Este estudio permite discernir que configuración de FSL y OSR es la que presenta un mejor rendimiento. Para analizar aún en más detalle la consideración de OSR, se ha realizado una configuración concreta de la base de datos donde solo tres patrones fijos deben ser reconocidos (y no los 24 disponibles). Esta configuración permite un mayor grado de *openness* y mejor análisis de la consideración OSR. A diferencia del objetivo anterior, la exactitud debe calcularse de forma ponderada entre las situaciones conocidas y desconocidas.

Debido al tamaño de la base de datos generada, la mayoría de técnicas de FSL empleadas en imagen no son válidas en el contexto del audio. Aunque las bases de datos de imágenes están configuradas para que se dispongan de pocas muestras por clase, es cierto que existe un mayor número de clases que en bases de datos de audio. Por consiguiente, el número de total de muestras es mucho mayor. Por esta razón, es necesario investigar una solución novedosa. La solución propuesta reside en los llamados *autoencoders*. Estas arquitecturas permiten la obtención de representaciones internas de la señal de audio de una menor dimensionalidad para que, posteriormente, pueda ser utilizada para la clasificación. Estos han mostrado resultados muy prometedores en problemas como detección anómala o traducción automática. La idea principal consiste en entrenar un autoncoder que sea capaz de encontrar una representación interna para cada patrón, siendo esta tan discriminativa que permita el rechazo de las situaciones no conocidas. Los *autoencoders* fueron originalmente pensados como una arquitectura no supervisada, es decir, no es necesaria la información sobre la clase a la que corresponde la muestra. La finalidad de un *autoencoder* es la recontrucción de la señal original. Para ello, en primer lugar, realiza una codificación (reducción de dimensionalidad) y posteriormente una decodificación volviendo a la dimensionalidad original. El punto de la red entre el codificador y el decodificador es conocido como cuello de botella o *bottleneck* en inglés. No obstante, ya que se dispone de la información sobre la clase de la muestra, se ha decidido analizar el comportamiento de un *autoencoder* con arquitectura semisupervisada. La representación interna y el consiguiente *bottleneck* no se calcula, únicamente, con la muestra a reconstruir, si no que también se debe tener en cuenta la información de la clase. Para la consideración del OSR se ha decidido realizar un clasificador *fully-connected*, pero cuya activación en la capa final corresponde a una sigmoide para que así sea capaz de discernir entre patrones conocidos y muestras de clases desconocidas.

Por último, las soluciones *end-to-end* están poco a poco atrayendo más interés por parte de la comunidad científica en APC. Si bien existen varios trabajos que proponen una red unidimensional, la proposición de ésta se basa en la elección de los investigadores. Debido a las ventajas que aportan las redes residuales, éstas también son una elección muy común en redes convolucionales unidimensionales. Las redes residuales han sido ampliamente estudiadas en el dominio de la imagen analizando la contribución de diferentes redes dependiendo de como se configuren. Dicho estudio no se ha realizado en el dominio del audio y empleando una red unidimensional. Así pues, la contribución en este objetivo es el estudio de distintos bloques residuales para soluciones *end-to-end* para poder así justificar la elección de una red residual u otra. En este caso, el estudio estadístico se lleva a cabo mediante un test de *Friedman* no paramétrico.

Resultados

En esta tesis se han realizado tres estudio distintos (cada uno de ellos dando como resultado final una publicación presente en esta memoria) que componen el compendio de artículos (modalidad en la que se ha redactado esta tesis). Los artículos pueden encontrarse en su versión original al final de esta memoria en forma de anexo. Los tres artículos previamente mencionados hacen referencia a tres problemáticas comunes que aparecen en sistemas de audición por computador. Los resultados obtenidos de cada publicación se resumen a continuación:

- Resultado 1: Respecto a la consideración de complejidad, se ha obtenido como resultado una red novedosa que es capaz de mejorar soluciones actuales del estado del arte con una ligera adición en el número de parámetros. Para ello, se ha combinado el uso de técnicas de *squeeze-excitation* y aprendizaje residual en los bloques convolucionales de la red.

- Resultado 2: Las problemáticas de FSL y OSR se han conseguido mitigar con un sistema basado en la arquitectura conocida como autoencoder. Los resultados muestran una considerable mejora si se compara con otras técnicas como la transferencia de conocimiento.

- Resultado 3: Por último, los resultados obtenidos en sistemas *end-to-end* indican que se debe tener una especial consideración en el diseño de redes de esta naturaleza cuando trabajan en el dominio del audio, ya que pueden diferir de las conclusiones obtenidas previamente en el dominio de la imagen.

Conclusiones

La clasificación de eventos sonoros es un campo que atrae cada vez más el interés de la comunidad científica. Sin embargo, la mayoría de contribuciones solo tienen como objetivo la mejora de la precisión de los sistemas propuestos obviando problemáticas que aparecen en productos que emplean dicha tecnología. En muchas ocasiones, las soluciones en el dominio del audio vienen muy inspiradas por el dominio de la imagen. Como se ha mencionado con anterioridad, el estado del arte actual propone la conversión del audio a una "imagen" para ser procesada, posteriormente, por redes neuronales que han demostrado un gran desempeño en el dominio de la imagen. Esta "imagen" generada a partir del audio no es trivial. Existe un gran número de contribuciones muy heterogéneas que discuten la que debería ser la representación usada. No obstante, muchas soluciones a día de hoy mitigan este fenómeno con soluciones que no son realizables en un contexto de tiempo real como puede ser el *ensemble* de multitud de redes entrenadas sobre distintas representaciones del audio. Estas redes tienen una serie de limitaciones a la hora de su diseño y es conveniente estudiar y proponer mejoras que supongan un incremento de la precisión sin que ello conlleve un aumento de parámetros o profundidad de la misma.

Las soluciones *end-to-end* se encuentran a día de hoy en un estado muy prematuro. Sin embargo, la ventaja que puede suponer la implementación de un sistema que emplea esta tecnología puede ser considerable. Estas redes parecen más propensas a la generalización ya que se evita la elección de hiperparámetros concretos. Todo el sistema es entrenado y por tanto todos los parámetros se pueden ajustar a una base de datos en concreto. A pesar de que existen trabajos en la literatura que proponen redes unidimensionales (y muchas de ellas emplean redes residuales) no se ha realizado un estudio de qué configuración residual aporta una mayor precisión en el contexto del APC. Se ha demostrado como la elección del bloque residual es dependiente de la lectura concreta que se realice del audio.

Por último, las consideraciones de FSL y OSR no han sido estudiadas con profundidad en el dominio del audio. El estudio de estas problemáticas es de vital importancia ya que en muchas aplicaciones reales es imposible recolectar un gran número de muestras por clase (FSL) y, en muchas otras, el sistema se va a encontrar desplegado en un entorno abierto. Para estudiar ambas problemáticas se ha generado, en primer lugar, una base de datos, se ha comprobado el funcionamiento de distintas soluciones del dominio de la imagen sobre esta base de datos y, posteriormente, se ha diseñado un sistema específicamente para el dominio del audio. En esta tesis se propone un sistema que es capaz de aplacar ambas problemáticas a la vez.

En el Capítulo 2, se muestra un amplio repaso de los diferentes aspectos tratados en esta tesis. Se han presentado los conceptos más relevantes relacionados a la inteligencia artificial. Se presenta un diagrama genérico de una solución que emplea tecnologías de audición por computador haciendo un breve resumen de cada una de las partes que lo componen. Por último, los problemas específicos abordados en esta tesis han sido discutidos, es decir, OSR, FSL, modelos de baja complejidad y soluciones *end-to-end*.

Los Capítulos 3 y 4 detallan las contribuciones y conclusiones de esta tesis respectivamente. Las contribuciones se muestran de forma enumerada haciendo referencia a cada uno de las artículos que componen el compendio. En el Capítulo 4 se realiza de la misma manera pero haciendo un breve resumen global y añadiendo el trabajo a futuro junto con un listado de todas las publicaciones que se han realizado en el marco de esta tesis.

En el Anexo A se muestra cómo la combinación de distintas técnicas, en este caso *squeeze-excitation* y aprendizaje residual, consiguen mejorar el rendimiento de una red puramente residual sin necesidad de añadir un número elevado de parámetros. El estudio se realiza desde un punto de vista de precisión global, por clases y mediante un estudio estadístico conocido como test de *McNemar*.

El Anexo B muestra la gran aportación de las soluciones basadas en autoencoder para resolver las problemáticas de FSL y OSR conjuntamente. El autoencoder permite la creación de representaciones robustas de los patrones a detectar siendo posible su discriminación de otros eventos sonoros. Dos autoencoders convolucionales son estudiados: uno no supervisado y otro semisupervisado. Los experimentos se realizan con distintos valores de *openness* (ver Sección 2.2.1) y de número de disparos. La clasificación se realiza mediante una red neuronal que es entrenada a partir de las representaciones generadas por los autoencoders. Los resultados obtenidos muestran que este marco es capaz de clasificar patrones de audio muy concretos aunque sea entrenado con muy pocas muestras y a la vez, es capaz de rechazar muestras que no pertenecen a ningún patrón. Concretamente, el autoencoder semisupervisado muestra un mejor rendimiento es multitud de experimentos.

El Anexo C muestra un estudio comparativo entre distintas redes convolucionales pensadas para un sistema *end-to-end*. Para realizar dicho estudio, se selecciona una red residual del estado del arte y es modificada acorde a los bloques presentados en un estudio similar, pero en el dominio de la imagen. Los resultados (análisis de la precisión y estudio estadístico) muestran que estos difieren entre el dominio del audio y de la imagen. El estudio se realiza sobre dos datasets y con dos preprocesados de audio distintos. Los resultados también muestran como la elección del bloque residual puede ser dependiente del preprocesado.

Palabras clave
Clasificicación de eventos sonoros, reconocimiento del conjunto abierto, aprendizaje con pocas muestras, técnicas de compresión y excitación, autoencoder, soluciones end-to-end, aprendizaje residual.

Abstract

The classification of sound events is a field of machine listening that is becoming increasingly interesting due to the large number of applications that could benefit from this technology. Unlike other fields of machine listening related to music information retrieval or speech recognition, sound event classification has a number of intrinsic problems. These problems are the polyphonic nature of most environmental sound recordings, the difference in the nature of each sound, the lack of temporal structure and the addition of background noise and reverberation in the recording process. These problems are fields of study for the scientific community today. However, it should be noted that when a machine listening solution is deployed in real environments, a number of extra problems may arise. These problems are Open-Set Recognition (OSR), Few-Shot Learning (FSL) and consideration of system runtime (low-complexity). OSR is defined as the problem that appears when an artificial intelligence system has to face an unknown situation where classes unseen during the training stage are present at a usage stage. FSL corresponds to the problem that occurs when there are very few samples available for each considered class. Finally, since these systems are normally deployed in edge devices, the consideration of the execution time must be taken into account, as the less time the system takes to give a response, the better the experience perceived by the users.

Solutions based on Deep Learning techniques for similar problems in the image domain have shown promising results. The most widespread solutions are those that implement Convolutional Neural Networks (CNNs). Therefore, many state-of-the-art audio systems propose to convert audio signals into a two-dimensional representation that can be treated as an image. The generation of internal maps is often done by the convolutional layers of the CNNs. However, these layers have a series of limitations that must be studied in order to be able to propose techniques for improving the resulting feature maps. To this end, novel networks have been proposed that merge two different methods such as residual learning and squeeze-excitation techniques. The results show an improvement in the accuracy of the system with the addition of few number of extra parameters. On the other hand, these solutions based on two-dimensional inputs can show a certain bias since the choice of audio representation can be specific to a particular task. Therefore, a comparative study of different residual networks directly fed by the raw audio signal has been carried out. These solutions are known as end-to-end. While similar studies have been carried out in the literature in the image domain, the results suggest that the best performing residual blocks for computer vision tasks may not be the same as those for audio classification. Regarding the FSL and OSR problems, an autoencoder-based framework capable of mitigating both problems together is proposed. This solution is capable of creating robust representations of these audio patterns from just a few samples while being able to reject unwanted audio classes.

Keywords
Sound event classification, Open-Set Recognition, Few-Shot Learning, squeeeze-excitation, autoencoders, end-to-end frameworks, residual learning

Contents

List of Figures

Chapter 1

Introduction

The data generated by society has been increasing exponentially in the last few years. This is largely due to the number of personal devices that are constantly connected to the Internet, such as laptops, smartphones or tablets. Besides these widespread personal devices, many other physical objects equipped with a variety of sensors and which offer relatively high computing and connectivity capabilities are currently being deployed to create "smart" environments. The convergence of all these technologies at homes, industries, cities and public and private venues has led to the concept of Internet of Things (IoT) [2, 3]. The range of IoT devices has also increased over time. While a few years ago the only personal device connected to the Internet was a computer or a laptop, today we have smartwatches, lamps or home assistants among other products. Thus, the number of solutions that can be provided thanks to these devices has increased to places that were unsuspected until recently. The interest in IoT is such that the European Union created a well-known "Horizon 2020" programme[1] to finance small and medium-sized enterprises whose focus is the creation of technological solutions using IoT devices. The range of possible IoT applications is really wide, covering the processing of information coming from multiple modalities, including images, audio or electric power consumption among others. The large amount of data available in most applications makes data-based or Artificial Intelligence (AI) solutions the choice. These solutions are based on the design of an algorithm, often a neural network, that "learns" from the available data in order to make future decisions. Well-known examples of IoT devices making use of AI algorithms are home assistants like Google Home$^{\mathrm{TM}}$ or Amazon's Alexa$^{\mathrm{TM}}$.

However, we often do not realise that many of the above technological advances can sometimes be a barrier for many people. For example, in home assistants, a voice interaction by the user is required. This is not possible for a segment of the population that is deaf or suffers from hearing loss. According to the World Health Organization (WHO), 466 million people have hearing problems. In addition, 1.1 billion young people (aged between 12 and 35 years) are at risk of suffering from hearing loss.[2] Some of the factors that explain this potential damage are the misuse of headphones or the exposure to high decibel levels in various places such as discotheques or sports stadiums. As far as the older population is concerned, about one third of people over 65 years of age suffer from hearing loss. However, AI can also be used to break down these generated barriers. People who are deaf or suffer from hearing loss could benefit from solutions based on machine listening techniques. Machine listening is the field that aims to extract meaningful information from audio signals by algorithms. These algorithms may be based on the combination of signal processing and AI methods. Some of

[1] https://ec.europa.eu/digital-single-market/en/research-innovation-iot

[2] https://www.who.int/news-room/fact-sheets/detail/deafness-and-hearing-loss

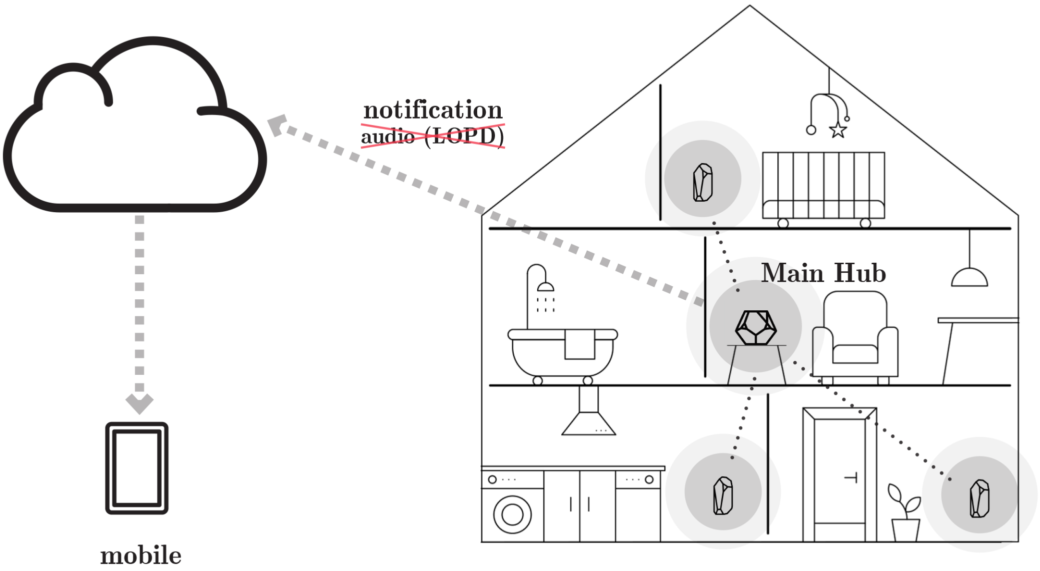


Figure 1.1: Illustration of a real case of using a Visualfy Home. The image shows the communication between detectors and the Hub and the communication of the latter with the API that is in charge of alerting the user through a mobile device.

the problems present in the field of machine listening are the classification of sound scenes, the detection and localization of sound events or the detection of anomalous sound events, among many others. In this context, a large segment of the world's population could improve their daily lives if they had IoT products making use of machine listening technologies.

Visualfy[3] is a Valencian startup whose goal is oriented towards the mission discussed above: the creation of products aimed at improving the daily lives of deaf people or affected by hearing loss. Visualfy was one of the European companies that obtained a grant from the European Union within the Horizon 2020 program[4,5] in the SME Instrument Phase II section. Currently, Visualfy has two IoT products that employ machine listening solutions. The products are known as Visualfy Home[6] (VH) and Visualfy Places[7] (VP). VH can be conceived as a home assistant for deaf people. The product consists of 4 devices, a central Hub and 3 detectors or microphones. The purpose of the product is to visually notify the user in case of an important sound event such as a door bell or fire alarm. Figure 1.1 shows a user case where the 3 detectors plus the Hub are placed in different rooms in order to monitor the desired situations. All devices have a LED that lights up when a sound event is detected (Figure 1.2 shows the variability of colors that can be related to the sound alerts desired to be monitored). The system is designed so that the users have the detectors in the rooms of the house that they want to monitor via audio. The Hub, besides being one more microphone, is in charge of all the audio processing and user notification. It should be noted that European regulations regarding user data have become more restrictive over time (see Figure 1.1 for illustration purposes). Nowadays, an IoT device cannot send private user information. Audio

[3] https://www.visualfy.com/

[4] https://cordis.europa.eu/project/id/662651/reporting/es

[5] https://novobrief.com/deaf-startup-horizon-2020/5962/

[6] https://www.visualfy.com/visualfy-home/

[7] https://www.visualfy.com/visualfy-places/

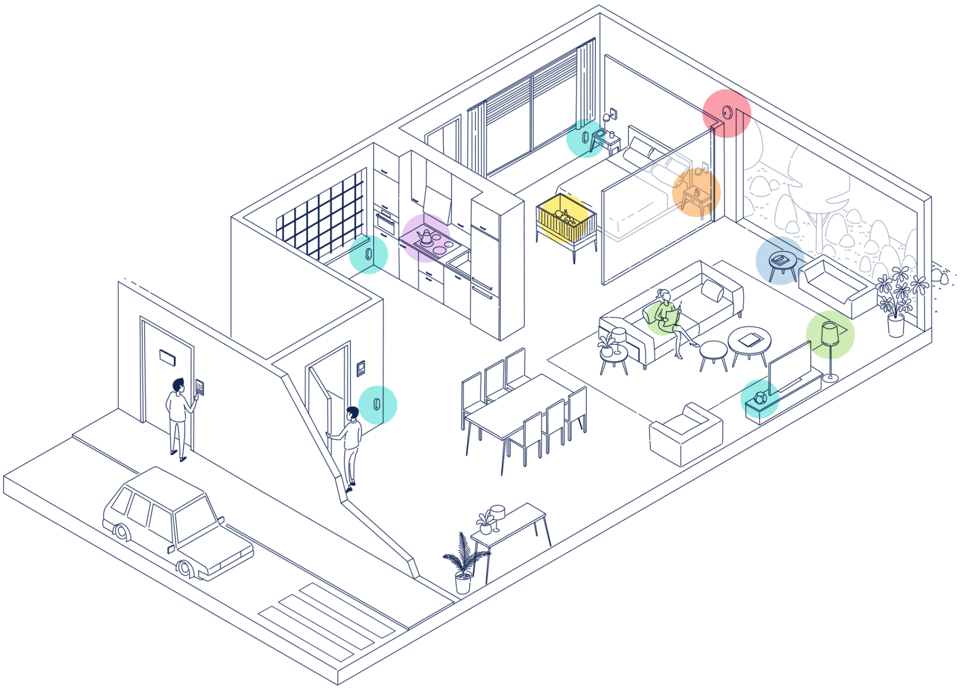


Figure 1.2: Isometric representation of a functional installation of a Visualfy Home. The sky blue color corresponds to the installed devices (detectors and Hub).

is considered private information. Therefore, the entire machine listening system present in the Hub runs locally without any interaction with an external Application Programming Interface (API). The only information that is transmitted over the Internet corresponds to the class that the detected audio event belongs to in order to notify the user. The alert system is fully configurable by the users, who can choose the color to which they want to link each alert and the device on which to receive the alert (mobile phone, smartwatch, smart bulbs, etc.). On the other hand, VP is a product designed to be deployed in public or crowded environments such as theaters, museums or stations. The system consists of the main Hub and lighting devices such as light bulbs or lamps. The machine listening system that is used follows the same considerations as those of the VH.

In addition to this application, solutions based on machine listening techniques can be deployed in a multitude of applications such as the early detection of faults in industrial machines [4], human-machine interaction [5] or sound event localization [6]. The fields of application can be the industrial context itself, ambient assisted living systems, autonomous cars, population monitoring or video games, among others. In addition, it should be noted that a large part of the population is reticent about being monitored visually either at home or on the public highway. In this context, audio is considered to be a less intrusive modality than video. These solutions could displace in the future those that use images or become complementary in order to elicit less intrusiveness. In addition, certain visual monitoring applications, such as those oriented to wildlife monitoring, have a number of limitations when light is poor or when not enough cameras are available to cover the entire space. Machine listening based applications could help to improve such solutions.

This thesis was written using a modality known as a compendium of articles. Under this assumption, the articles made during this research (at least three accepted in journals of the first or second quartile) are introduced in their original version as an annex. In this particular memory, three annexes can be found at the end of the document. The memory is also composed of four chapters that introduce the issues and the motivation to deal with them (Chapter 1), widely detail the state-of-the-art (Chapter 2), list the contributions of the papers (Chapter 3) and conclude the work (Chapter 4).

1.1 Motivation

The industrial nature of the doctoral program implies an orientation of the research carried out throughout this thesis towards the innovative application of its results and conclusions to the context of an industrial product or service. Therefore, a major motivation underlying the research objectives of this work is on the emphasis of developing technological solutions with clear applicability to real-world problems and scenarios.

The emergence of AI-based applications has led to the development and improvement of many products. The area where most effort has been made is computer vision [7, 1, 8]. However, there is an increasing interest from the scientific community and companies in developing products based on the information extracted from acoustic signals [9, 10, 5, 11]. These solutions are considered less intrusive from the user's point of view and can also solve problems that are extremely difficult to tackle in the image domain (e.g. lack of luminosity, impossibility to map the whole space, etc.). When it comes to developing a machine listening solution, a series of considerations must be taken into account. First, general sounds do not have a deterministic structure, that is, the fact that an event occurs does not determine that another one will follow. This is not the case in the voice or music domain, where AI systems can obtain specific patterns thanks to grammatical structures, established rules and pre-defined dictionaries (e.g. phonemes or musical notes). On the other hand, general sounds have a polyphonic nature, i.e. two or more sound events can occur at the same time. Therefore, a masking phenomenon can appear where one event "hides" the other. Another important consideration is that there are several kinds of sounds depending on their nature. If they are examined from a spectral point of view, the acoustic signals can be tonal (e.g. fire alarms) or noise-like (e.g. keyboard tapping). If analysed by their temporal behaviour, the sounds can be transitive (e.g. door slam), continuous which in turn can be divided into stationary (e.g. machine breakdown), non-stationary (human speech), intermittent which in turn can also be divided into sounds with periodic patterns (e.g. foot steps) or irregular intervals (e.g. baby crying). Finally, the recording process must be also taken into account. This process can add noise or filter certain frequency components. The fact that very often the sampling frequency or bit depth is different for different examples within the same training dataset, and those that the system must classify once trained can result in poor system performance. Thus, an audio classification system must at least address these considerations.

However, these are not the only issues that arise when deploying a machine listening solution in a real, uncontrolled scenario. The first problem that appears is the one known as *Open-Set Recognition* (OSR) [12]. This problem refers to all the circumstances, in our case sounds or audio events, that the system will have to face without having been trained on them. Let's imagine that a classifier is designed to detect 3 specific audio patterns: doorbell, fire alarm and a specific telephone melody. When this system is deployed in a domestic context, it will have to reject a multitude of sounds present at home. It is evident that it is impossible to account for all the casuistry of a home: conversations, television, pet sounds, street noise, etc. Therefore, a system must be designed with a special emphasis on the rejection of samples that do not belong to the patterns or classes for which it has been trained. Thus, a machine listening system cannot be conceived as a closed set system where all the audio samples it is going to face will belong to a class the system has seen during the training stage. The second problem that arises is the *Few-Shot Learning* (FSL)[13]. When a specific pattern detection system is desired, for example, for a particular doorbell, a high number of samples will not be available because it is unfeasible to request hundreds or thousands of recordings from a user. Systems based on AI methods, based whether on machine learning or deep learning techniques, have shown very promising results when trained with a large number of samples. In the field of audio, these techniques have also shown very satisfactory results when used as generic classifiers, that is, when the classes to be detected show a high degree of intra-class variability. Let's imagine a classifier that classifies between dog barks and baby cries. In general, each bark can be different depending on the breed of dog, intensity, etc. Just like a cry, which can vary due to the sex of the baby, the claiming (hunger, sleepiness, etc). However, what is desired in this type of system is the capability to detect every event of a bark or babycry nature, not a specific bark or cry. In the case of alarms, since each user can have his or her own, some specific training is needed. Since it is unfeasible from a user experience point of view being required to record a large number of samples, the system must be trained with very few. The last issue to be considered is the execution time. As it can be expected, the less time the system takes to recognize, the better user experience the customer will have. Classifiers should be as simple as possible, which leads to the development of *low-complexity models* [14]. Currently, many state of the art solutions propose systems formed with a multitude of independent classifiers where each one of them provides a result and the final prediction comes from the output of some fusion mechanism. These ensemble-based methods may not be practical in a real environment and results must be improved in another way without a decrease in execution time. Also, as previously discussed, due to data protection legislation (LOPD) no user data is allowed to leave an IoT device. Therefore, the whole data processing and classification (the machine listening pipeline) must be performed locally at the egde device.

In summary, given the scenario discussed above, it is necessary to propose machine listening systems that mitigate these problems. Some existent solutions in audio come from studies that have addressed similar problems in other domains. For example, the problem of FSL is widespread in the audio domain thanks to the advantages of facial recognition or signature recognition [15, 16]. FSL methods have also been studied in music applications related to genre classification [17]. On the other hand, the OSR problem has been analysed in detail in image recognition by experimenting with large image databases [12, 18]. Finally, with regard to the consideration of low complexity models, it has only recently begun to receive considerable attention in audio [14, 19]. Thus, this thesis is motivated by the need to analyze and propose novel solutions in the audio domain that can be deployed in real environments and showing robustness to the problems discussed above. In addition, it has been decided to experiment with end-to-end frameworks (where the whole system is made up of trainable parameters) in the audio domain [20, 21]. These frameworks are increasingly attracting the attention of the scientific community for their ability to generalize to different problems in the context of machine listening.

1.2 Objectives

Taking into consideration all of the above, the objective of this thesis can be defined as follows:

To design, implement and evaluate sound event and sound scene classification systems that must be deployed in real-world, uncontrolled environments, where the classification must be performed locally in an edge device and only a very limited training dataset is available.

This global objective can be divided into the following sub-objectives:

- To provide an overview of state-of-the-art solutions to the problems addressed in the thesis. This includes from the first techniques to the current ones. In addition, it is intended to provide detailed explanations on how these solutions have been implemented in different contexts, be it images, speech or music, since sometimes there is not much literature on how such problems affect the sound event classification performance.
- To analyze the behavior of different end-to-end residual networks in the context of machine listening. Due to the advantages that this family of solutions can provide, such as the elimination of non-trainable parameters chosen manually for each problem, it has been deemed necessary to experiment with different residual architectures to study their behavior depending on the audio reading and training dataset. These solutions are thought to be easier to be generalized to other problems, since the whole system is made up of trainable parameters.
- To propose novel residual squeeze-excitation modules in order to improve the accuracy of audio classification systems. Most deep Learning-based frameworks rely on the ability of convolutional neural networks to learn features from audio signals that lead to good discriminative properties. These networks are composed of several stacked convolutional layers. However, they also have a number of limitations. Newer techniques such as squeeze-excitation or residual learning have shown promising results. The proposal of a module that combines both techniques can improve the classification without increasing significantly the depth of the network.
- Finally, to design an FSL/OSR framework capable of detecting specific audio patterns while rejecting all unknown audio classes. The FSL consideration must be taken into account when very few samples of each pattern are available to train the system. Moreover, considering jointly FSL and OSR within the same system requires both network structures coping simultaneously with both problems and developing meaningful datasets for this task.

1.3 Structure of the thesis

This thesis is divided into four chapters and three annexes, with an additional section for bibliographical references. The contents of each chapter are as follows:

Chapter 1 introduces the scenario in which this thesis is developed, emphasizing the aspects related to its industrial character and the solutions that are intended to be integrated into real market products. In addition, it presents the motivation and objectives guiding this research work.

Chapter 2 provides some background on the main concepts used throughout this work. The fundamental problems addressed in the thesis are presented, discussing some of the initial solutions found in the literature and current state-of-the-art techniques. In addition, the main public audio databases that have been used in this thesis to evaluate the proposed contributions are presented.

In **Chapter 3** the most significant contributions of each of the papers that make up the compendium of this thesis are listed and summarized.

Finally, **Chapter 4** concludes the work carried out in this thesis by discussing the main outcomes of the above research work. It also mentions some future research lines that may be considered to gain further insight into the results obtained in this thesis.

Annex A corresponds to the first publication that makes up the compendium of articles. It corresponds to a paper published in *IEEE Access* (doi: *10.1109/ACCESS.2020.3002761*), which presents some novel configurations for convolutional neural networks that combine residual learning and squeeze-excitation techniques. These are shown to improve considerably audio classification performance without increasing significantly the number of parameters in the network.

Annex B corresponds to the second publication of the compendium. The publication was accepted in a special issue of the *Sensors* journal called *Intelligent Sound Measurement Sensor and Systems* (doi:*10.3390/s20133741*). This paper presents a novel architecture based on autoencoders aimed at addressing jointly the problems of Open-Set Recognition (OSR) and Few-Shot Learning (FSL).

Annex C presents the latest publication of the compendium, corresponding to a paper published in *IEEE Access* (doi: *10.1109/ACCESS.2020.3031685*). This publication explores and compares different residual configurations but in the context of end-to-end solutions accepting as input the raw audio signal. Such kind of solutions have been shown to provide promising results without the need to apply signal transformations to the input.

Chapter 2

Background

2.1 Brief overview

This section explains the context in which this thesis is framed. Firstly, it explains the rise of artificial intelligence (AI) in recent years and how the appearance of these technologies has allowed a great improvement in machine listening algorithms. Nowadays, classical algorithms are being replaced by algorithms more data-driven, that is, a considerable amount of data is needed for this solutions to perform well. It also explains the modules that make up a machine listening system today. Finally, the event known as DCASE is presented. This event was held for the first time in 2013 and can be understood as a consequence of the great interest in machine listening solutions from the scientific community. As it will be explained later, the tasks presented at the event can be considered as the most topical in the state of the art.

2.1.1 Artificial Intelligence history review

Today, the multitude of technological solutions employ methods or techniques of artificial intelligence (AI). Although this technology seems to be very new, its theoretical basis were first developed back in the 1940's. The first idea was to imitate the human brain by shaping the behaviour of neurons. The first attempt to understand how neurons work can be found in [22], published in 1943. This is recognized as the first AI-related work. Furthermore, this assumptions were even modelled by means of electrical circuits. In 1949, [23] introduced the idea that neural pathways are strengthened every time they are used, a concept fundamentally essential to the manners in which humans learn. If two nerves fire at the same time, the connection between them is strengthened. The field of artificial intelligence was born in a workshop at Dartmouth College in 1956 [24]. John McCarthy was responsible for the conference and is considered the father of artificial intelligence. From this conference, the first AI solutions began to be proposed. In 1956, the Logic Theorist was introduced. It is a program created to perform autonomous reasoning and is considered the first artificial intelligence program. In 1958, the Perceptron was created [25]. Perceptron consists of a supervised binary classification algorithm. The classification must be carried out by means of a feature vector. Perceptron is a linear function with weights that are adjusted to correctly predict the available data. The Perceptron equation can be defined as follows:

$$f(\mathbf{x}) = \begin{cases} 1, & \text{if } w \cdot \mathbf{x} + b > 0 \\ 0, & \text{otherwise} \end{cases} \tag{2.1}$$

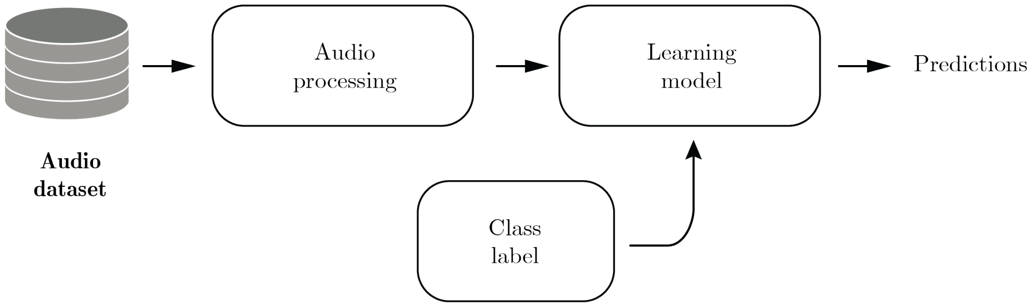


Figure 2.1: Generic illustration of an artificial intelligence solution (either Machine Learning or Deep Learning) for an audio database. In this case a supervised model is represented as the information from the tags/labels is used.

where w represents the trainable weights, $\mathbf{x}$ is the data feature vector and b is a constant bias. A neural network consists of a set of layers: input layer, hidden layers and output layer in which each one is composed of several neurons based on the Perceptron idea. The first multi-layered neural network was designed in 1965, thus causing the birth of Deep Learning [26].

Nevertheless, the lack of computer power and the scarce databases available at that time made the advances in the AI field very difficult [27]. This is how, in 1974, the AI field got into the so called AI winter.

Despite this, during that time, some relevant contributions were made, as in [28], where a backpropagation algorithm was proposed for multi-layer neural networks. From the 1980s onwards, more powerful computers were designed, allowing the development and implementation of modern artificial intelligence methods.

2.1.2 Machine Learning and Deep Learning

The rise of artificial intelligence has led to the proposal of a multitude of algorithms that can be classified into different categories according to their design. On the one hand, Machine Learning (ML) algorithms are mathematical algorithms that learn from the data provided to them. ML algorithms have adjustable parameters whose value is changed and calculated base on the available training data (see Figure 2.1). ML algorithms are not necessarily based on neural networks; Support Vector Machines [29] or Decision Trees [30] are clear examples of these. On the other hand, the so called, and very widely spreaded, Deep Learning (DL) algorithms are based on neural networks that have at least one intermediate layer. Deep Learning (DL) solutions are based on different types of neural networks, be they Deep Neural Networks (DNNs) [26] or Convolutional Neural Networks (CNNs) [31]. As databases have become larger, the number of layers required to generalise all possible cases must be greater. The explosion of DL-based solutions can be considered relatively recent and is due to the availability of an unprecedented large amount of data and sophisticated computing hardware such as GPUs.

From an implementation point of view, ML algorithms require a prior process of data cleansing and selection, known as feature engineering. However, in the DL scenario, feature engineering might not be necessary in many applications. A common state-of-the-art example of feature engineering for ML computer vision classification is to use Local Binary Patterns [32] to create an histogram that can be used as a feature vector. This is, the image is transformed in a feature vector prior to the classification/training process. On the other hand, DL algorithms can be fed by the image itself omitting the feature extraction step. CNNs are deep neural networks that extract characteristics from the image itself to perform the task for which it has been designed (classification, detection, etc.). Thus, the implementation of DL algorithms can be much faster from the implementation point of view. However, in certain application domains, such as audio or machine listening, a feature engineering process is still required.

Both, ML and DL algorithms, can be classified as follows according to the information available in the dataset:

- **Supervised**: every data in the database is mapped to a class label. The objective of the algorithm is to create a function that allows to map new input data to its corresponding output label. The available data is associated to a series of labels. That is, an entry corresponds to a specific output. The objective of the algorithm is to create a function that allows to map all the inputs to their corresponding outputs. Types of problems supervised may be classification problems (assigning an input to a specific class), detection (determining the existence of relevant information) or regression (predicting an actual value from an input).

- **Unsupervised**: data in the database is not mapped to any kind of class label, therefore the goal of the algorithm is to find similarity between the data itself. Some unsupervised examples may be clustering (grouping of data because of their similarity) or data coding (the algorithm creates richer internal representations of an entry through a learning process).

- **Semi-Supervised**: is a trade-off between the two previously explained approaches. For a dataset to be considered semi-supervised, some samples of it must be labelled and others not. Therefore, both supervised and unsupervised techniques must be used. We will also consider a semi-supervised problem when all the labels in a dataset are available but unsupervised and supervised techniques are used to solve the task in question.

In this thesis the 3 approaches have been worked on. The classification of audio patterns is done in a supervised way since the DL algorithm must map an audio clip to a specific labelling class or category. Unsupervised learning is used to mitigate the Few-Shot Learning problem (FSL), deeply analyzed in this thesis (see Section 2.3.1). In addition, a semi-supervised approach is analyzed to study the same problem comparing both approaches (unsupervised and semi-supervised).

Likewise, supervised algorithms can be further divided in different subcategories depending on the number of labelling-classes in the dataset or how many classes can be associated to a single sample:

- **Binary classification**: is the simplest classification problem since only two classes appear. Normally, these classes are understood as positive class and negative class $[0, 1]$ [33].

- **Multi-class classification**: the classification problem presents more than two classes. It should be noted that a sample belongs to only one of the classes. To be considered as a class present in the dataset, it must appear at least once. This approach appears in audio events classification (AEC) task [34, 35, 36].

- **Multi-label classification**: can be considered the most complex problem of classification. In this scenario, apart from having several classes, a sample may or may not belong to more than one class. This scenario appears in the audio domain due to its polyphonic nature [37, 38].

In this thesis, the classification problems studied correspond to multi-class problems since an audio can only belong to a specific pattern within an amalgam of possible patterns. Because an audio can only belong to one pattern, it does not seem to make sense to analyze from a multi-label point of view.

2.1.3 Basics of Deep Learning

The Deep Learning architectures used in this thesis are those known as DNNs and CNNs (see Figure 2.2). A DNN can be defined as a NN where intermediate hidden layers have been added. The layers made up of nodes are called *fully-connected* (see Figure 2.2 a)).

Node equation and activation functions

The node is simply a point of computation where the inputs to the node are weighted by a weighting vector whose values are adjusted at training stage. To achieve non-linear network behavior, a non-linear trigger function is usually applied. Thus, the mathematics of a single node responds to the following equation:

$$z = f(a) = f(w \cdot \mathbf{x} + b) = f(\sum_{i=1}^{n} w_i \mathbf{x}_i + b) \tag{2.2}$$

where $\mathbf{x}$ represents the input to the neuron, w the trainable weights, b a trainable bias and f denotes an activation function. The activation functions used in this thesis are Rectified Linear Units (ReLU) [39] and Exponential Linear Units (ELU) [40]. ReLU activation avoids gradient vanishing, this being a known phenomenon that appears when gradient diminishes dramatically as it is propagated backward through the network [41]. In addition, they are computationally more efficient, allowing the network to learn the parameters much faster than other activation [42]. As shown in [40], ELU activation can be understood as a modification of the ReLU activation that allows faster training and higher accuracies while maintaining all the benefits reported by ReLU. Both activation equations are:

$$\text{ReLU }(a) = \max(0, a) \tag{2.3}$$

$$\text{ELU}(a) = \begin{cases} a, & \text{if } a > 0 \\ \alpha(e^a - 1), & \text{otherwise} \end{cases} \tag{2.4}$$

where α is a fixed constant and a is an independent variable that can have values in the $[-\infty, +\infty]$ interval. As it can be observed, when $a > 0$, both activations behave in a linear way.

Other very important activation functions are the so called sigmoid and softmax. The softmax function receives an input vector with K positions and normalizes it into a distribution consisting of K probabilities proportional to the exponentials of the input numbers:

$$\text{softmax}(a)_i = \frac{e^{a_i}}{\sum_{k=1}^{K} e^{a_k}} \tag{2.5}$$

$$\sum_{i=1}^{K} \text{softmax}(a)_i = 1 \tag{2.6}$$

where the input is the K-dimensional vector $[a_1 \dots a_K]$ and $\mathrm{softmax}(a)_i$ is the output value for i-th vector coefficient. Softmax is used in multi-class classification problems.

On the other hand, the sigmoid function maps the $(-\infty, \infty)$ range to the $[0, 1]$ interval, thus transforming any real value into another real value that can be interpreted as a probability:

$$\mathrm{sigmoid}(a) = \frac{1}{1 + e^{-a}} \tag{2.7}$$

where $\mathrm{sigmoid}(a)$ is the output value corresponding to any real value a in the $(-\infty, \infty)$ range.

Cost functions

The values of the parameters of the network, that is, the values of the linear coefficients that compose the nodes of the network, are found by means of iterative optimization procedures that take into account the set of training samples and the expected network output for those input samples. The so called *cost function* is evaluated on each iteration of the procedure, this function being a complex mathematical representation of the difference between the expected network output and the output observed on each iteration of the procedure. The standard coefficient optimization is based on an iterative gradient descent intended to find a local or global minimum of the cost function [43].

Over the years, many different cost function have been implemented. The particular problem to be treated greatly influences on the choice of the cost function. Some of the most usual choices are the *mean squared error* (MSE), the *categorical cross entropy* (CCE) and the *binary cross entropy* (BCE). MSE [44] is the most suitable choice when calculating the error in the prediction of real values. Its equation is:

$$\mathcal{L}_{mse} = \frac{1}{N} \sum_{i=1}^{N} \left(X_i - \hat{X}_i \right)^2 \tag{2.8}$$

where $\mathcal{L}_{mse}$ is the MSE cost function, N is the number of training samples, and X_i and $\hat{X}_i$ represent the target and predicted real valued outputs for the i-*th* training sample respectively.

The categorical cross entropy, CCE, should be used in multi-class classification problems where each sample belongs to only one amongst K different categories. This problem is usually tackled with a softmax activation in the last layer of the network. Its mathematical formulation is:

$$\mathcal{L}_{cce} = -\sum_{i=1}^{N} y_i \log \hat{y}_i \tag{2.9}$$

where $\mathcal{L}_{cce}$ is the CCE cost function, N is the number of training samples and y_i and $\hat{y}_i$ represent the original and predicted class probability respectively. As each sample belongs to one single class, y is an one-hot vector meaning it is equal to 1 in the position of the class that the sample belongs to, and 0's in the other positions. $\hat{y}$ is a probability vector output by the network. As it can be seen, this cost function takes into account only the probability assigned by the network to the positive class. The probabilities assigned to the other classes are multiplied by 0 by the the one-hot vector.

The binary cross entropy, BCE, is the proper choice for multi-label classification problems. The difference between this type of problems and the multi-class classification described above is that in multi-label classification labels are not exclusive, meaning that one sample can be labelled with more than one label. The vector y is not one-hot, instead, it can be 1-valued in more than one position. Hence, assuming a multi-label problem with K different labels, the output layer of the network is typically formed by K independent neurons with sigmoid

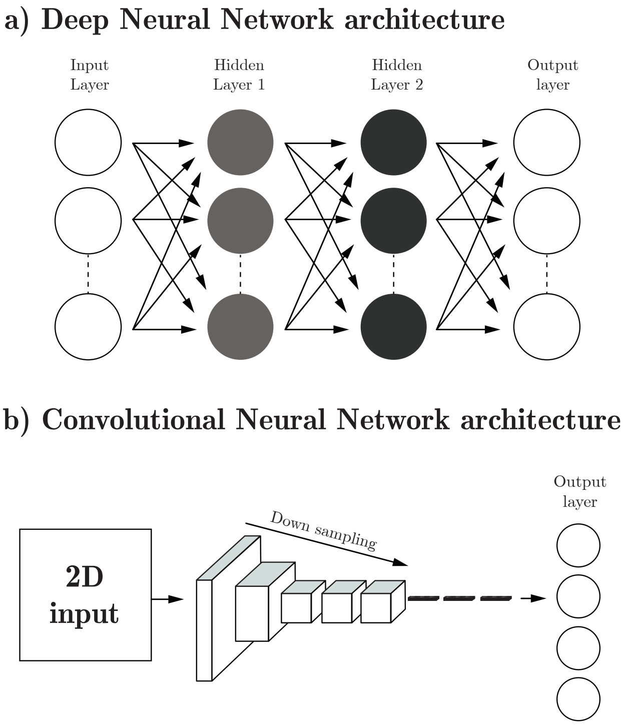


Figure 2.2: Illustration of two different Deep Learning architectures.

activation on each one. The BCE cost function, $\mathcal{L}_{bce}$, would be mathematically formulated as follows:

$$\mathcal{L}_{bce} = -\frac{1}{N}\sum_{i=1}^{N}\sum_{k=1}^{K}\left[y_{i,k}\log\hat{y}_{i,k} + (1-y_{i,k})\log(1-\hat{y}_{i,k})\right] \tag{2.10}$$

where N is the number of samples, $y_{i,k}$ is a binary variable that indicates if sample i should be labeled with label k and $\hat{y}_{i,k}$ is the probability that the network assigns to sample i to be labelled with label k. Although $\hat{y}_{i,k}$ are probability values, it must be noticed that, since each sample can be labelled with more than one label, $\sum_{k=1}^{K}\hat{y}_{i,k}$ is not necessarily equal to one for any sample i.

Different types of layers in a CNN

Deep neural networks DNNs based on the linear node equation described in Equation 2.2 were the first to be developed and implemented. An important improvement to this was the development of convolutional neural networks (CNNs). These type of networks base there node equation in the linear correlation of the input with a numerical element called kernel. While these types of networks can be used with 1D inputs, their true benefits are seen when dealing with 2D input signals [7]. The first contribution on which these networks are currently based was presented in [45]. This contribution is known as Neocognitron. The fundamentals presented in this paper are used today. However, at that time there was no training with backpropagation. The first paper that presented CNNs as they are understood today was [46]. The network was trained with backpropagation and used the fundamentals of Neocognitron to extract two-dimensional signal patterns. Its design became gradually formalized in the following works by LeCun *et al.* [31, 47]. These networks were inspired by the work done in [48] where 2D data input was contemplated and allowed (see Figure 2.2 b)). The most relevant contribution of these works is the creation of the convolutional layer. In the convolutional layers, the linear coefficients of the node Equation 2.2 are replaced by a convolutional kernel specially designed to pinpoint particular patterns in the input signal. Although general equations for the convolutional layers can be derived, in this thesis the focus is put on 2D and 1D input data. For the particular case of 2D inputs and one single kernel function, $\mathcal{K}$, the node equation would be:

$$z(x,y) = f(\sum_{i=1}^{k_H}\sum_{j=1}^{k_W}\mathcal{K}_{i,j}a_{x+i-1,y+j-1} + b_{x,y}) \tag{2.11}$$

where $z(x,y)$ is the node output, $\mathcal{K}$ is a 2D kernel of size $k_H \times k_W$, a is the 2D input, b is a fixed bias and $f(.)$ is an activation function (see 2.1.3). The particular values of the kernel coefficients $\mathcal{K}_{i,j}$ are found during the training process.

A convolutional layer is defined by the following parameters:

- **Kernel size**: most usual sizes for CNNs with 2D inputs are 3×3 or 5×5. However, in some CNNs 7×7 or 11×11 are used [49].
- **Stride**: represents the displacement, or hop, of the filter as it moves across the input image. The most common stride is $(1,1)$.
- **Number of channels**: number of filters that form a convolutional layer.
- **Padding**: determines how the edges of the image are processed. If no padding is added, the output size is reduced compared to the input. In order for input and output to have the same size, the edges of the input must be padded, with zeros or with another technique such as replicating the values of the edge of the input, as many times as necessary according to the kernel size.

Therefore, given an input of size $n_H \times n_W$ and a kernel, $\mathcal{K}$, of size $k_H \times k_W$, it can be formulated that the output, also known as feature map, has size:

$$\text{feature map size} = \left\lfloor \frac{n_H + 2p - k_H}{s} + 1 \right\rfloor \times \left\lfloor \frac{n_W + 2p - k_W}{s} + 1 \right\rfloor \tag{2.12}$$

where s denotes the stride and p the padding, both supposed to be equal in both directions. Therefore, a convolutional layer composed of one single kernel takes an input that belongs to the $\mathbb{R}^{n_H \times n_W}$ space and outputs a matrix in the $\mathbb{R}^{n'_H \times n'_W}$ space. In a more general case the input can have another dimension of size C. In, for example, an RGB image composed of three color channels, $C = 3$. Another degree of generalization can be added by assuming that a convolutional layer can be composed of several kernel filters. So, assuming these two generalizations Eq. 2.11 can be rewritten as:

$$z^{c'}(x,y) = f(\sum_{i=1}^{k_H}\sum_{j=1}^{k_W}\sum_{k=1}^{C}\mathcal{K}^{c'}_{i,j,k}a_{x+i-1,y+j-1,k} + b^{c'}_{x,y}) \tag{2.13}$$

where C' would be the number of kernels and $c' \in [1,2,...C']$, $\mathcal{K}^{c'}$ is the c'-th kernel of size $k_H \times k_W \times C$, a is the input signal, $b^{c'}_{x,y}$ is a fixed bias associated to the c'-th kernel and $f(.)$ is an activation function (see 2.1.3). As in can be observed, the output signal size would be:

$$\left\lfloor \frac{n_H + 2p - k_H}{s} + 1 \right\rfloor \times \left\lfloor \frac{n_W + 2p - k_W}{s} + 1 \right\rfloor \times C' \tag{2.14}$$

In addition to convolutional layers, CNNs are made up of other types of layers. These are:

- **Batch normalization layers**: this layer corresponds to a standardization layer whose objective is to achieve a better generalization during training. When training many parameters present on the network, special care must be taken in the learning rate and initialization of the weights. This means that training deep networks can be complex when there are a large number of non-linearities (internal covariate shift). The normalization of each batch allows to be a little less careful with the previously exposed [50, 51, 52]. Thus, the normalization process is within the model. This layer is usually placed after the convolutional layer, before activation. In this context, the activation function can be defined as an independent layer of the model.

- **Pooling layers**: the number of filters of the convolutional layers (C') usually increases as going deeper in the CNN. In the event that no reduction in dimensionality (n_H or n_W) is made by this layer, it may be the case that the last layers have to process 3D matrices of the same width and height as the input to the first layer and with the substantial increase in the third dimension (C' in that specific layer). Since this data processing is not feasible, a susampling layer is required. These layers are known as pooling. Like convolutional layers they have a receptive field of a certain size known as pool size. The reduction of dimensionality is done by taking a single value of the receptive. The most common choices are to the highest value or to the average. In addition, there are global poolings where the pool size is the width and height of the three-dimensional matrix, thus achieving a single value per filter.

- **Dropout layers**: in dropout layers [53, 54] a certain ratio of randomly chosen neurons is deactivated. The objective of this is to avoid the over-adjustment, also known as overfitting (see Section 2.7). In many cases, dropout is necessary to achieve better generalization of the model in validation and test stages.

- **Flatten**: on many occasions, CNNs have to carry out classification tasks. In many occasions, this task is carried out by fully-connected layers. Thus, to achieve a 1D representation of a three-dimensional signal, a resample known as flatten is performed where all values are placed in a one-dimensional way [55, 56].

- **Fully-connected layers**: the first CNNs, implemented fully connected layers as they were in charge of establishing the relationships between the feature maps generated by the previous convolutional layers and the final output of the classification [31, 42, 7]. This trend is still in use [57, 58]. However, there is a parallel tendency that implements fully-convolutional networks for classification [59, 60, 61].

Transfer learning and fine-tuning

DNN/CNN training procedure can be, in many cases, very costly in terms of time, computing power and sample availability resources. Techniques have been developed to take advantage of large pre-trained networks by only adapting the value of the trainable coefficients to the particular problem being treated. The most widely used of these techniques are transfer learning [62, 63, 64, 65] and fine tuning [66, 67]:

- **Transfer Learning**: under this approach, the pre-trained network is used as an intelligent feature extractor. The main idea is to use the output of a pre-trained network to feed a simpler one with much less coefficients to be trained. This allows much less samples and computational power resources to achieve the necessary level of accuracy. Well-known pre-trained networks for audio classification are VGGish[1] [7], L3net[2]

[1] https://github.com/tensorflow/models/tree/master/research/audioset/vggish

[2] https://github.com/marl/openl3

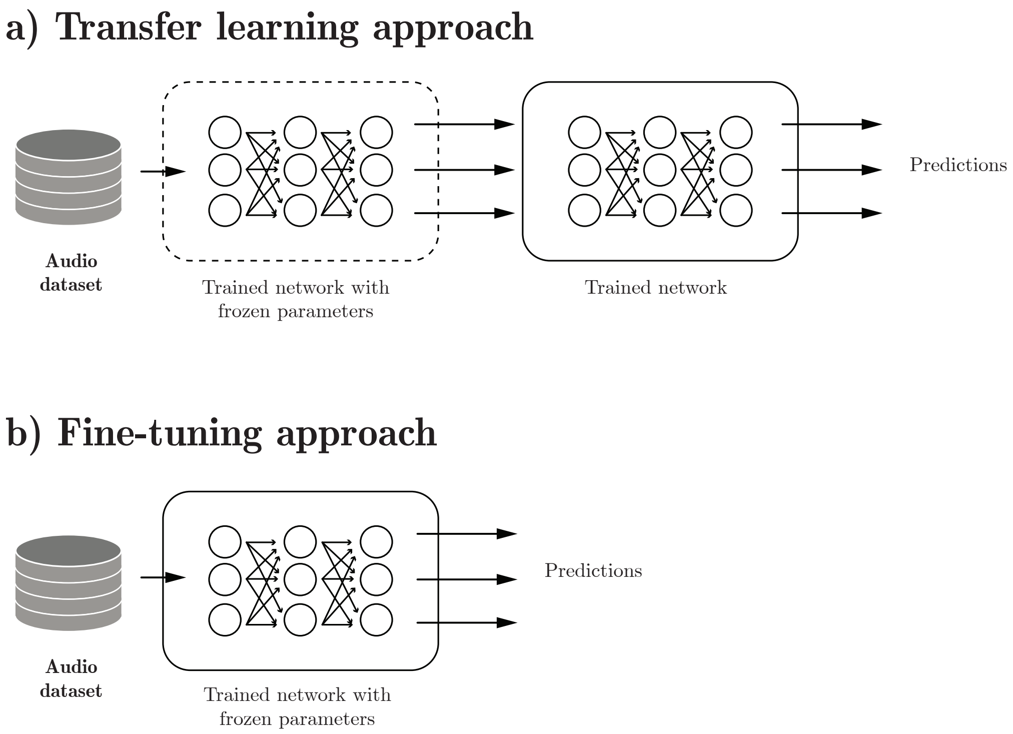

a) Transfer learning approach
Audio dataset
Trained network with frozen parameters
Trained network
Predictions
b) Fine-tuning approach
Audio dataset
Trained network with frozen parameters
Predictions

Figure 2.3: Generic illustration showing the difference between a system based on transfer learning and one based on fine-tuning. As can be seen, the difference is found in the freezing of the weights of the pre-trained network with another database.

[68, 69] or Soundnet[3] [70, 71].

- **Fine Tuning**: the fine-tuning process is the one in which a net is retrained but having some weights pre-learned in another previous training process [66, 67]. The retraining process is usually shorter for the following reasons: the database is usually smaller and the weights are closer to the global minimum. Therefore, this must be taken into account when designing the second training process. The learning rate is usually lower and the schedule is usually somewhat different.

2.1.4 Machine Listening pipeline

Computer audition or Machine Listening can be defined as the field of study of algorithms whose objective is the extraction of relevant information from audio data by a machine. Broadly speaking, Machine Listening can be defined as the set of algorithms that attempt to mimic the behavior of the human ear. It includes methods from different fields such as: signal processing, pattern recognition and artificial intelligence, among others. Within the field of machine listening several subgroups can be found depending on the nature of the audio signal; music or environmental are examples of this. A specific nature and a problem to be solved determine a machine listening task. Some tasks in the machine listening field are Acoustic Event Classification (AEC) [72, 73], Acoustic Scene Classification (ASC) [74, 57] or Sound Event Detection (SED) [38, 11, 75]. In this context, the problem of classification and detection is different. Classification is understood as the assignment of a particular class

[3] https://github.com/cvondrick/soundnet

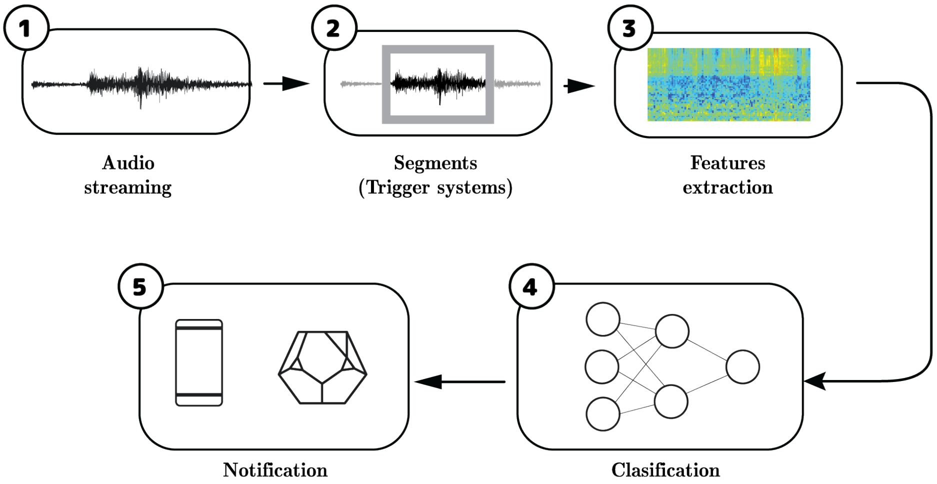


Figure 2.4: Illustration of the pipeline of a machine listening solution. This illustration is based on the commercial product Visualfy Home.

to an audio clip. When the classification is multi-label it is usually defined as tagging. On the other hand, detection consists of assigning a specific class and determining the start and end time of the recognized sound event. Recognizing and classifying can be understood as synonyms. The detection consists, therefore, of classification and determination of the temporal boundaries of a specific event in an audio clip of greater duration.

Any solution based on machine listening technologies can be divided into a number of steps. These steps range from audio recording to user notification. In this thesis any machine listening framework is divided into 5 steps.

1. Audio streaming

The first stage of a machine listening system is to capture audio from its source. A microphone, or in some cases arrays of them, together with the digitizing system are crucial since they can determine parameters of the signal such as noise floor, dynamic range or number of samples to be processed. The sampling rate and the bit depth will determine the resolution level of the audio and therefore its 2D representation (see Sect. 2.1.4).

2. Segmentation

The captured audio streaming must be segmented in order to be processed. A first approach could be to process all the captured audio in segments of a certain length L with $L/2$ overlap. Other approaches make a pre-selection of the audio before segmentation process/overlapping by implementing some triggering algorithms [76, 77, 78]. Many of them are based on the well-known *cusum* statistics [79, 80] that pinpoints changes in the mean of a time series. Thanks to the emergence of new edge devices with great computing power as Raspberry Pi[4]

[4] https://www.raspberrypi.org/

or Google Edge TPU[5] the trigger process is losing relevance since it is feasible to implement fast processing systems directly from the streaming.

The particular value of L is highly dependent on the particular task. Common values are $L = 10s$ [81] for Acoustic Scene Classification, $4s$ [82] for Acoustic Event Classification or $40ms$ [83] for Sound Event Detection.

3. Feature extraction

CNNs have shown excellent performance in classification and detection in the image domain [7, 1, 84]. The first machine listening solutions based on CNNs imitate frameworks of the image domain [85, 86] by converting the 1D audio signal to a spectro-temporal representation where one axis corresponds to the frequency and another to the time [87, 88, 89, 90, 91]. The choice of representation is a field of study nowadays as there are many options. The window size and overlap must be chosen for the time axe, whereas the scale and number of bins must be selected for the frequency axe (see Section 2.4.2 for further details about the frequency scale issues). Decisions about the frequency representations are very relevant since it has been proven to affect the behaviour of the system in a relevant manner [92].

In a more recent approach the raw 1D audio signal is input to the network without implementing any previous feature extraction. In this case the CNN acts not only as a classifier but as feature extractor as well [60, 21, 93, 20]. So far, 2D representations have shown better results than raw audio inputs [94]. However, this is an open research line since it could lead to computer power savings.

4. AI system

The remarkable results achieved during the last decade by AI systems based on CNNs make a very wide range of machine listening problems to be solved by these techniques. Examples of this are classification or tagging [9], detection [95] and localization [96].

Most solutions implement CNN networks due to the pre-processing (convert the audio into a time-frequency 2D representation) explained above (step 3). The AI module is responsible for the output of relevant information from the audio input.

5. User notification

In the particular case of the system worked during this thesis a final notification stage exists. Users receive a notification, a visual representation of the sound event processed by the AI stage. Visualfy Home or Places systems are able to send push notifications to the mobile devices of the users and to illuminate smart bulbs in the room.

2.1.5 Detection and Classification of Acoustic Scenes and Events (DCASE)

The interest of the scientific community both from academia and from companies in solutions based on machine listening has been increasing in recent years. The Detection and Classification of Acoustic Scenes and Events (DCASE) is an annual event that is used as a backbone by the community working on machine listening methods. This event is divided into a Challenge and a Workshop. The first edition took place in 2013 and was organized by the Centre for Digital Music, from the Queen Mary University of London, and IRCAM (*Institut de Recherche et Coordination Acoustique/Musique*) from Paris. The Workshop was

[5] https://cloud.google.com/edge-tpu

not a stand-alone conference but a special session in WASPAA2013 (e.g. Workshop on Applications of Signal Processing to Audio and Acoustics). The next edition was not until 2016. From that year on, there has been an annual event until 2020.

The reputation of the conference can be seen in how the number of papers submitted has increased over the years. The following table shows the number of papers, the acceptance rate and the attendance rate by the academic sector and by companies.

Edition	Papers	Acceptance rate (%)	Attendance (academic (%)/companies(%))
2016	23	100	68/32
2017	27	90	61/39
2018	43	73	62/38
2019	54	66	50/50
2020	49	58	56/44

Table 2.1: Evolution of the DCASE Workshop.

Two conclusions can be drawn from the table above (see Table 2.1). The first is that the Workshop is gaining a reputation over the years. Each year the number of papers accepted has increased while the acceptance rate has decreased, which indicates that more and more papers are sent to the Workshop. The second conclusion is that there is interest from private companies in solutions based on machine listening technologies. The attendance ratio has been increasing on the part of the companies until there is a 50-50 balance (2019 edition). In addition, companies such as Apple, Google, Mitsubishi or Hitachi sponsor or have sponsored the Workshop.

On the other hand, the Challenge has a much more competitive approach where novel solutions to a particular problem or task may or may not appear. Over the years, the problems proposed have increased. The proposal of a task can have several motivations. The most basic would be to encourage research related to that task by the scientific community. Each task is defined by a specific dataset and a baseline proposed by the task organizers. The simple fact of having specific data for a task and a starting point may be enough for the scientific community to focus on that problem. The second motivation is that the solutions proposed by the teams are very easy to compare since all the proposed systems must be trained as specified in the task presentation. The number of tasks has tripled from the first edition to the last one, from 2 to 6. The period in which the challenge is held is usually 3 months (normally between March and June). However, during the rest of the year, researchers continue to use the datasets and systems proposed in the Challenge as a comparison for their contributions. Therefore, the celebration of this challenge is very useful and has clearly accelerated the proposal of machine listening solutions for a multitude of tasks and problems present. Below are the tasks present in each of the editions held:

- **2013 (2)**: Acoustic scene classification [97, 98] and Sound event detection [97, 98]
- **2016 (4)**: Acoustic scene classification [99, 100], Sound event detection in synthetic audio [100, 101], Sound event detection in real life audio [99, 100] and Domestic audio tagging [100].
- **2017 (4)**: Acoustic scene classification [102, 103], Detection of rare sound events [102, 104], Sound event detection in real life audio [102, 104] and Large-scale weakly supervised sound event detection for smart cars [102, 104].

- **2018 (5)**: Acoustic Scene Classification [81], General-purpose audio tagging of Freesound content with AudioSet labels [105], Bird audio detection [106], Large-scale weakly labeled semi-supervised sound event detection in domestic environments [107] and Monitoring of domestic activities based on multi-channel acoustics [108, 109].
- **2019 (5)**: Acoustic Scene Classification [81], Audio tagging with noisy labels and minimal supervision [110], Sound Event Localization and Detection [111, 112], Sound event detection in domestic environments [113] and Urban Sound Tagging [114].
- **2020 (6)**: Acoustic Scene Classification [81, 14], Unsupervised Detection of Anomalous Sounds for Machine Condition Monitoring [115, 116, 117], Sound Event Localization and Detection [111, 83], Sound Event Detection and Separation in Domestic Environments [118, 119, 113], Urban Sound Tagging with Spatiotemporal Context [120] and Automated Audio Captioning [121, 122, 123].

The tasks presented at the DCASE have increased over the years. The tasks of Acoustic Scene Classification (ASC) and Sound Event Detection (SED) have been maintained over the years. On the one hand, the ASC task has been updated according to the problems that have arisen. The first editions presented an ASC task without any added restrictions. However, the 2018 edition already presented the ASC problem with mistmatch devices, the 2019 edition added the problem of Open-Set Recognition (OSR) (see Section 2.2.1) and the 2020 edition added the restriction of low complexity models (see Section 2.4.1). The SED task has been analyzed in several scenarios as the editions have progressed. In the last two editions, the spatial location of the source has been added, so the task has been redefined as Sound Event Localization and Detection (SELD). The rest of the tasks have been changing, but the change in complexity of these tasks is worth mentioning. In the first editions, the tasks were tagging or detection. The tasks proposed now imply a design of more complex systems capable of captioning an audio or detecting fault patterns in an unsupervised way.

The contribution of the DCASE to this thesis has been remarkable. Datasets presented in the Challenge have been used, as well as comparing the system's performance with those proposed in the Challenge. It is a good reference point to check the state of the art with respect to some technology. Finally, during the thesis, there has been participation in 1 task in the 2019 edition and 4 in the 2020 edition (see Section 4.2 for the Technical Reports related to the tasks).

2.2 Open-Set Recognition

2.2.1 Definition

In real scenarios, it is impossible to collect data on all possible situations the system will face once trained and deployed [12]. In the specific case of a machine listening system installed in a real world environment performing classification tasks 24 hours a day, it is nearly impossible to collect audios from all the classes that the system will be forced to face. In public or crowded spaces, a multitude of new situations appear even every day, such as construction sites, crowds, different types of background noises, traffic, etc. The AI system must, therefore, be able to cope with events from unseen classes by rejecting them in the classification stage. This is called an OSR context. On the contrary, in a closed-set recognition context the input to the system is restricted to events that belong to one of the pre-trained classes; think, for example, in a medical image system where the usual practice is to input only images of certain parts of the human body to recognize a determined type of pathology. See Figure 2.5 for a visual representation of these situations.

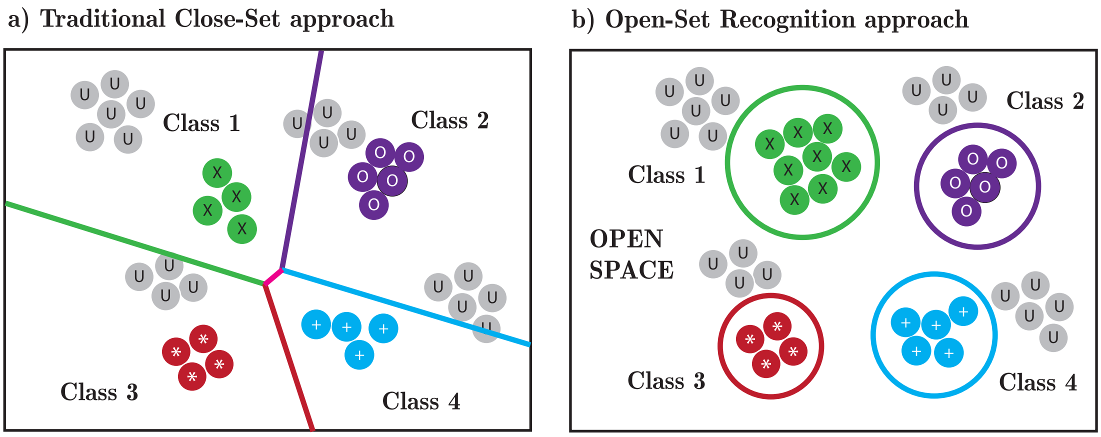

a) Traditional Close-Set approach
b) Open-Set Recognition approach
Class 1
Class 2
Class 3
Class 4
OPEN SPACE
U
X
O
*
+

Figure 2.5: OSR problem illustration. Classes defined with U denote unknown classes (either KU or UU) and color classes represent known ones.

When dealing with OSR problems, special consideration should be given to defining which classes should be recognized and which ones should be rejected. With this idea in mind, the following classification of the classes that appear in an OSR problem can be made [18]:

- **Known Known (KK) classes:** classes that are used in the training and validation stage and that must be classified as positive events by the system.
- **Known Unknown (KU) classes:** classes that are available during the training stage and should be classified as negative events (rejected) by the recognition system. These classes are very useful since they allow the system to make representations and generate boundaries that can help to discern samples from the unwanted category.
- **Unknown Known (UK) classes:** classes for which no samples are available during training but side-information such as semantic/attribute information is available during training. This category is not considered in this work.
- **Unknown Unknown (UU) classes:** classes that are not used nor in the training nor in the validation stage and must obviously be rejected by the classifier. The system only sees these classes in the test stage.

The OSR describes a situation in which the UU classes appear in the test stage without appearing in training. An OSR system must therefore classify the KK classes at the same time as rejecting the samples from the UU classes. If the system has been trained with a KU class, these must also be rejected (see Figure 2.5b)).

The OSR consideration is determined by the values known as open space risk and openness. As described in [12], the space in the feature map away from the KK and KU classes is known as *open space* $\mathcal{O}$. Close-set systems are forced to label these samples in one KK class. This carries a risk known as *open space risk* $R_{\mathcal{O}}$. The lack of knowledge of the UU classes makes the calculation of this risk complex to obtain. However, in [12], it is formalized as the quotient between the open space and the overall measure space S_o:

$$R_{\mathcal{O}}(f) = \frac{\int_{\mathcal{O}} f(x)dx}{\int_{S_o} f(x)dx} \tag{2.15}$$

where f denotes the measurable identification function. $f(x) = 1$ means that some kind of KK is recognised, otherwise $f(x) = 0$. Under such formality, the higher the samples in the open space are labeled as KKs, the highest $R_{\mathcal{O}}$ is.

The metric known as openness provides a relationship between the classes used in the training phase and the classes to be recognized in the test phase. In a close-set problem, this value would be the same. However, in an OSR context the number of classes in test phase is greater than the number of classes you have been trained with. The first definition of openness, proposed in [12], is:

$$O = 1 - \sqrt{\frac{2 \times |C_{TR}|}{|C_{TA}| + |C_{TE}|}} \tag{2.16}$$

where C_{TA}, C_{TR} and C_{TE} denote the set of classes to be targeted, the set of classes used during training and the set of classes that the system is facing in testing stage. $|\cdot|$ indicates the number of classes of each set. The higher the value of O, the more OSR the context is. In [12], no relationship is specified between the variables that define the number of classes. In more recent works [18, 124, 125, 126] the relationship $C_{TA} = C_{TR} \subseteq C_{TE}$ is held. However, this case does not contemplate the use of KU classes since all classes employed during training must be recognized. Thus, in [127], a relationship is established that does take this consideration into account, $C_{TA} \subseteq C_{TR} \subseteq C_{TE}$. However, this relationship may have some inconsistency with the definition of openness presented in Equation 2.16. Let's suppose this case: $|C_{TA}| = 3$, $|C_{TR}| = 20$ and $|C_{TE}| = 25$:

$$O = 1 - \sqrt{\frac{2 \times |C_{TR}|}{|C_{TA}| + |C_{TE}|}} = 1 - \sqrt{\frac{2 \times 20}{3 + 25}} = -0.195 \tag{2.17}$$

In this scenario, a value of $O < 0$ is obtained, which is unreasonable. Hence, a new reformulation of the value of openness is needed that is consistent with this scenario. As can be appreciated, C_{TA} is a set within the C_{TR} set, that is, any class that wants to be recognized must appear in C_{TR}. So, in [128], the value of openness (O^*) is reformulated with this consideration as:

$$O^* = 1 - \sqrt{\frac{2 \times |C_{TR}|}{|C_{TR}| + |C_{TE}|}} \tag{2.18}$$

With this new formulation the range of the openness value is bounded to the range $0 \leq O^* < 1$. When $O^* = 0$, $C_{TR} = C_{TE}$, indicating that no UU classes exist. On the other hand, as C_{TE} becomes larger, $C_{TE} > C_{TR}$, $O^* \rightarrow 1$ indicating a greater complexity in the OSR consideration.

2.2.2 Background

The design of systems that can be deployed in OSR contexts currently has several lines of research. A first division can be made between discriminative and generative models [129, 130]. In the case of this thesis the work is done with discriminative models. The following is a review of the state of the art of solutions with descriptive models based on both traditional machine learning methods and Deep Learning solutions.

Classic ML methods start from the premise that the distribution of the training set and the test set is the same, i.e. they do not expect to have to deal with an unknown situation. The pipeline is fixed: an input is taken, mapped in the desired feature space and classified in a predefined class (see Figure 2.5a)). As it can be observed, this clearly closed-set functionality cannot be assumed in OSR contexts. As an example, let's imagine that we perform a 1-9 digit classifier. Once the system is trained, it expects the entries to be predicted to be digits. However, if images of animals are entered, it can be observed how the system classifies them as digits by, for example, saying that a duck is a number two.

If one ML technique has attracted a lot of interest when implemented in OSR scenarios it has been the Support Vector Machine (SVM) [29]. The main idea of SVM is the creation of a hyperplane or set of hyperplanes in a feature space, which allows the separation of the different classes present in that space. The so-called *support vectors* are used to calculate this hyperplane. These vectors correspond to the closest samples between different classes. The method is designed so that the hyperplane is as far away as possible from each support vector, positioning itself right in the middle of the support vectors of different classes.

In [12], a modification of the SVM, known as 1-vs-Set, is presented and incorporates an open-space risk term to take into account space beyond the KK classes. The classification/recognition function is optimized as:

$$\arg \min_{f \in \mathcal{H}} \{R_{\mathcal{O}}(f) + \lambda_r R_{\epsilon}(f(V))\} \tag{2.19}$$

where V denotes the training data, $R_{\mathcal{O}}$ the open space risk, R_{ϵ} the empirical risk, f a recognition function and λ_r a regularization constant. In particular, for the OSR consideration, a parallel hyperplane is added to the one obtained by the SVM. This allows a slab to be left in the feature space to consider samples as unknown. Another similar approach is presented in [131, 132] where a new method for calculating the hyperplane known as Best Fitting Hyperplanes Classifier (BFHC) is proposed. It should be pointed out that these methods reduce the space occupied by the KK classes by generating gaps in the feature space. However, the feature space occupied by these classes is not confined.

To achieve the confinement of each KK class in the feature space, the hyperplanes separating them need to be non-linear. In [18], a modification of the SVM is proposed that achieves this non-linear separation. This method is called Weilbull-calibrated SVM (W-SVM). The method combines the extreme value theory (EVT) [133] for the calibration of the score of two independent SVMs. The first corresponds to one-class SVM compact abating probability (CAP). This SVM is the first step. If the probability of belonging to a class does not exceed a certain threshold, the sample is considered as unknown. If the sample exceeds the threshold, the sample is passed to the second SVM. This second step corresponds to a binary CAP SVM trained with the Weilbull cumulative distribution function (CDF). This CDF returns both $P_\eta(y|x)$ based on the Weibull CDF derived from the match data and $P_\psi(y|x)$ based on the reverse Weibull CDF derived from the nonmatch data, which is equivalent to rejecting the Weibull fitting on the non-match data given an input x. The W-SVM recognition for a multi-class KK ($\mathcal{Y}$) scenario is defined as:

$$\begin{aligned} y^* &= \arg\max_{y \in \mathcal{Y}} P_\eta(y|x) \times P_\psi(y|x) \times \iota_y \\ &\text{subject to } P_\eta(y^*|x) \times P_\psi(y^*|x) \geq \delta_R \end{aligned} \tag{2.20}$$

where ι_y denotes a boolean factor to determinate if the given input has been predicted as KK by the first one-class SVM classifier. Therefore:

$$\iota_y = \begin{cases} 1 & \text{if } P_O(y|x) > \delta_\tau \\ 0 & \text{otherwise} \end{cases} \tag{2.21}$$

where $P_O(y|x)$ is the posterior estimate of the first SVM classifier given an input x and δ_τ indicates a defined threshold. Both threshold δ_τ and δ_R are set empirically according to the authors [18]. δ_τ is set to 0.001 and δ_R should be specified depending on the openness value such as:

$$\delta_R = 0.5 \times \text{openness} \tag{2.22}$$

Another algorithm based on SVM and EVT is known as P$_I$-SVM [134]. This algorithm follows the idea of thresholds with the same strategy as the W-SVM. In this the probability of inclusion P$_I$ for an input x conditioned on the parameters θ_y is defined as:

$$P_I(y|x, \theta_y) = \xi \rho(y) P_I(x|y, \theta_y) \tag{2.23}$$

where $\rho(y)$ is the prior probability of the class y and ξ some constant. The recognition using a set of Weibull models is defined as:

$$\begin{aligned} y^* &= \arg\max_{y\in\mathcal{Y}} P_I(y|x,\theta_y) \\ \text{subject to } & P_I(y^*|x,\theta_{y^*}) \geq \delta \end{aligned} \tag{2.24}$$

where δ denotes a fixed threshold. As it can be appreciated, both algorithms base their capacity to reject the unknown classes on the establishment of fixed thresholds. Both W-SVM and P_I-SVM set a single threshold for all KK classes to be recognized, which does not make much sense since certain classes may be closer to KU or UU classes than others. In addition, the information that provides the value of openness that could provide some insight to the problem, is also often unknown, that is, it is complicated to know all the classes that make up the C_{TE} set.

To mitigate these limitations, a new modification of the SVM algorithm for the OSR context known as probabilistic open-set SVM (POS-SVM) is proposed in [135]. The contribution of this new classifier is that it is able to empirically obtain a single threshold for each KK class. The main difference is that $R_{\mathcal{O}}$ and R_ϵ is defined as a probabilistic representation.

Another family of machine learning algorithms for OSR are those known as Sparse Representation-based. These algorithms have shown very promising results in the image domain, in particular for facial recognition [136, 137]. The Sparse Representation-based Classifier (SRC) [138] aims at the poorer identification of the test sample in terms of training. This classifier (like the SVM) has a close-set nature. The SRC classifier adapted to an open-set environment (SROSR) is presented in [124]. The state of the art does not show sparse classifiers when classifying sound events, so this family of algorithms was discarded to make an audio classifier with open-set consideration.

The distance-based algorithms have also been adapted to the OSR context. In [139] the Nearest Non-Outlier (NNO) algorithm is presented as an extension of the so-called Nearest Class Mean (NCM). The classification is made on the basis of the distance between the test sample and the average of the KKCs. The interest of this algorithm is that it is able to add new classes dynamically. The well-known Nearest Neighbor algorithm has also been modified for the OSNN open-set classification [140]. Unlike many OSR algorithms, the classification is not made from a threshold but from a ratio of the similarity of scores to the two most similar classes.

The latest family of machine learning algorithms that have been modified to take into account OSR consideration are those known as margin-distribution based. In [141], a sound classifier is introduced, the Extreme Value Machine (EVM) which is derived from the concept of margin distributions. The EVM is based on the assumption that there is a positive sample $\mathbf{x}_i$ and sufficient negative samples $\mathbf{x}_j$, resulting in pairwase margin estimates m_{ij}. Also, use me that there is a non-degenerate and continuous margin distribution. Then the distribution for the minimal values of the margin distance for $\mathbf{x_i}$ is given by a Weibull distribution. Once the EVM is trained, the classification is done with the following decision function:

$$y^* = \begin{cases} \text{arg max}_{l\in\{1,\ldots,C\}} \hat{P}(\mathcal{C}_l|\mathbf{x}') & \text{if } \hat{P}(\mathcal{C}_l|\mathbf{x}') > \delta \\ \text{unkown} & \text{otherwise} \end{cases} \tag{2.25}$$

where $\mathbf{x}'$ represents a test sample, C the number of KKCs following the definition in this context, δ a fixed threshold, $\mathcal{C}_l$ the lth known class and $\hat{P}(\mathcal{C}_l|\mathbf{x}')$ the probability that the test sample belongs to the lth class.

As shown in Equation 2.6, the softmax function shows a close-set nature. This function is usually implemented in the last layer (classification layer) of the deep neural network (either DNN or CNN). The first approach to the adaptation of deep networks is the creation of the model known as OpenMax [142]. The network is first trained in a close-set context.

Following the idea of the NCM, each class is represented by a mean activation vector (MAV) with the average of the classifying examples correctly in the training stage in the penultimate layer of the network. Subsequently, the training sample distances from their corresponding MAVs are calculated to fit a Weibull distribution to calculate the psuedo-activation of the unknown classes. In [143] a slightly different approach is taken and the replacement of the last layer by a 1-vs-rest final layer of sigmoids, known as a Deep Open classifier (DOC), is proposed. In [144] the competitive overcomplete output layer (COOL) is presented to circumvent the overgeneralization of neural networks over regions far from the training data. The solution presented as tWiSARD is based on the idea of distance-based algorithms [127]. Other algorithms such as [145], use the information available in the KU classes to combine the Softmax loss and the novel Entropic Open-Set and Objectosphere losses.

Within the algorithms that implement deep models, it is intended to highlight those that are based on reconstruction as they have been the choice for this thesis. In [146], an algorithm based on latent representation reconstruction is presented that allows robust detection of UU classes. This algorithm extends the idea of Openmax. The equations on which the open-set with latent representations is based are the following:

$$(\boldsymbol{y}, \mathbf{z}) = \mathcal{E}(\mathbf{x})$$

$$\hat{P}(\mathcal{C}_l|\mathbf{x}, \mathbf{x} \in C) = \text{Softmax}_i(y) \tag{2.26}$$

$$\hat{\mathbf{x}} = \mathcal{D}(\mathbf{z})$$

where $\mathcal{E}$ and $\mathcal{D}$ denote the encoder and decoder networks (more detail is provided in Section 2.3.2). $\mathbf{x}$ indicate the input sample, $\boldsymbol{y}$ the the representation of the final hidden layer whose dimensionality is the same as the number of KK classes and $\mathbf{z}$ the latent representation that can also be indicated as bottleneck. $\hat{\mathbf{x}}$ is the reconstructed input, ideally, it should be the same as $\mathbf{x}$. The main advantage of this design is that it calculates the prediction $\boldsymbol{y}$ together with the latent representation $\mathbf{z}$. Unlike the original distance-based works where the distance from $\boldsymbol{y}$ to the mean of each μ_i class was calculated, this approach jointly takes into account the $\boldsymbol{y}$ and $\mathbf{x}$ distributions. Therefore, the distance measurement is calculated by the following equation:

$$d(\mathbf{x}|Ci) = |[\boldsymbol{y}, \mathbf{z}] - \mu_i|_2 \tag{2.27}$$

where $[\boldsymbol{y}, \mathbf{z}]$ denotes the concatenation of both vectors. However, this approach can be considered as one that has used autoencoders since the $\mathcal{E}$ encoder is in charge of generating the dimensionality vector equal to the number of KK classes. As it will be explained later, the $\mathcal{E}$ encoder is only in charge of calculating the latent or bottleneck representations.

In [147] the use of autoencoders is proposed for the detection of UU classes. A system based on 3 steps is proposed: two of training and one of inference (the system once trained). The first step is to train the $\mathcal{E}$ encoder and a $\mathcal{C}$ classifier for the recognition of the KK classes. The classifier takes as input the bottlenecks generated by the encoder. This process is defined as Closed-set training. The second step is known as Open-Set training, with the frozen weights of the encoder ($\mathcal{E}$), a Decoder ($\mathcal{D}$) is trained to perfectly construct those samples of KK classes (remember that the encoder has been trained with these samples) and to poorly reconstruct those samples that correspond to unknown classes (conditional decoder training). In order for the decoder to know which samples it must reconstruct perfectly and which ones it must not, a vector called label condition vector is used. Reconstruction errors are modeled using an extreme value distribution to calculate the threshold. Once the system is trained (inference phase) the system produces a prediction by the classifier and C reconstruction errors (one per known class). For the sample to be classified among one of the KK, the minimum reconstruction error must be less than a certain threshold. If this is the case, the class will be predicted as the class indicated by the classifier. If not, it is considered unknown.

As can be seen, almost all algorithms are based on a final decision based on an empirical threshold. So, this choice is crucial. For the choice of the particular algorithm, the arrangement of samples of KU classes must also be taken into account. However, it should be noted that this thesis is based on the resolution of the OSR problem but also of the Few-Shot Learning (FSL) problem presented in the following section. Therefore, choices based on deep networks do not make much sense as will be seen below. However, as will be seen below, autoencoders are a great choice to mitigate the FSL problem and since they have been introduced into the OSR problem they will solve both problems together. For the detection of an unknown class the idea presented in Equation 2.25 is used. Moreover, the choice of a last sigmoid layer seems very interesting for the problem that we are trying to face in this work.

Therefore, as it will be seen below, a solution with the capability of recognizing KK classes and rejecting KU or UU classes at the same time that only few samples of each KK class are available is presented. The framework is based on the autoencoder architecture, this contribution can be seen in Annex B.

2.3 Few-Shot Learning

2.3.1 Definition

Few-Shot Learning (FSL) is the problem that appears when systems based on artificial intelligence have to be trained with very few samples per class. Deep Learning based solutions have become state of the art due to the large amount of data available for training. A standard configuration for this type of problem is one in which the system sees more samples of the same class in the training stage than in the test stage. In FSL problems, the approach is completely the opposite, the system is trained with very few samples (shots in this context) and must predict a large number of examples of that class in test phase.

One characteristic of this problem is the emergence of intra-class classification. This phenomenon can be perfectly understood with the example of facial recognition. The objective of an FSL system is not to detect faces in an image or to classify an image as a human face, but to detect the identity of the person. It is nearly impossible to collect, let say, thousands of photos of a single person just to train, for example, an access control system. A simple and topical example would be a commercial system for unlocking the mobile phone where a small set of photos is used to recognize the identity of the user. As it can be noticed, the feature space that composes the class of a person is very small and is within a feature space that includes all the faces. The face of a particular user must by identified within the more generic class *human face*; this is an intra-class classification problem. This scenario also appears in the context of this thesis where very particular, specific and individualized pattern alarms need to be detected.

2.3.2 Background

To summarise, FSL learning can be tackled in three different ways:

- **Modifying the available data**: these solutions propose the increase of the training set so that conventional DL techniques and algorithms can be applied. Most data augmentation techniques follow a set of hand-crafted rules. In the image domain, this is, the application domain where FSL has been most widely developed, set-augmentation using translations [148, 149], flipping [150, 151], scaling [149], cropping [150] or rotation [152] can be found. In the audio domain, although used for other purposes, data augmentation techniques such as mixup [153], temporal cropping [154], pitch sifting, time stretching or loudness modification have appeared [155]. Other novel techniques are based on the manipulation of the spectrogram [156]. The main problem with these techniques is that the method that may be applicable to a particular dataset does not necessarily have to be applicable to another. Furthermore, by means of these techniques the FSL problem is not being solved as such, but rather an attempt is being made to change the philosophy of the problem to a standard artificial intelligence problem by artificially enlarging the dataset.

- **Selecting a particular model with FSL considerations**: There are a number of models that take into account the consideration of FSL. In this thesis, due to the nature of the problem, the solution based on embedding learning [13] is studied. There are three other learning methodologies such as multi-task learning [157], learning with external memory [158] or generative modeling [13]. The main objective of embedding learning is to make smaller dimensional representations in order to group together, in the feature space, samples of the same class while moving them away from those of other classes. Normally, the classification step is performed with some similarity function as, for example, the euclidean distance. In this thesis, the results of embedding learning implemented with autoencoders are presented.

- **Use prior knowledge solutions**: These techniques solve the problem of FSL thanks to the previous knowledge that the neural network possesses having been trained with other data. Thus, it can be understood as a refinement of the network parameters to solve a specific FSL problem, for example, fine-tuning.

Before going deeper into the details of embedding learning based methods, it is worth highlighting another family of algorithms that can be used for FSL. These have in common the modification of the cost function to emphasize the distance between classes during the training stage [63]. It can be understood as a variant of a classic Deep Learning classification problem that can be implemented in concrete FSL scenarios. Ring Loss [159] is presented as a modification of the softmax loss function. The objective of this cost function is to perform a soft normalization, where it gradually learns to constrain the norm to the scaled unit circle while preserving convexity leading to a more robust features. The ring loss L_R is defined as:

$$\mathcal{L}_R = \frac{\lambda}{2m}\sum_{i=1}^{m}(||\mathcal{F}(\mathbf{x}_i)||_2 - R)^2 \tag{2.28}$$

where $\mathcal{F}(\mathbf{x}_i)$ is the final feature map created by the deep network for the sample $(\mathbf{x}_I)$. R represents the norm value which is also learned and λ is the loss weight trade-off. Finally m is the batch size. $\mathcal{L}_R$ can be used with other loss functions such as softmax. Ring loss stabilizes the feature norm of all classes, that is, rectifies the classification imbalance that softmax may lead to perform better overall [159]. The cost function known as Center Loss is intended to enhance the discriminative power of the deep feature maps created by CNNs. This cost function was thought for the problem of facial recognition. The discrimination is done through the creation of clusters. The center loss simultaneously learns the center of the cluster of each class (class center) to classify and penalizes the distance between the feature maps and therefore, different class centers. In this case, the system must use both the softmax loss and the center loss ($\mathcal{L}_C$). The center loss is defined as:

$$\mathcal{L}_C = \frac{1}{2}\sum_{i=1}^{m}||\mathcal{F}(\mathbf{x}_i) - \mathbf{c}_{y_i}||_2^2 \tag{2.29}$$

where $\mathbf{c}_{y_i}$ denotes the y_ith class center. Ideally, the centers of each class should be updated every time the feature maps change, that is, once the whole training set has been seen by the network (performing an epoch). This makes the center loss an inefficient and impractical function. To solve this fact, two modifications are made: the centers are updated each batch (the center corresponds to the average of the feature maps of each of the samples of a class) and a scalar value α is used to avoid high disturbances between batches. This value is restricted in $[0, 1]$. The final system loss function is defined as:

$$\mathcal{L} = \mathcal{L}_S + \lambda \mathcal{L}_C = -\sum_{i=1}^{m} \log(\text{softmax}) + \lambda \mathcal{L}_C \tag{2.30}$$

where softmax corresponds to the function explained in 2.1.3. As it can be appreciated, both $\mathcal{L}_R$ and $\mathcal{L}_C$ can be understood as modifications of the softmax function whose objective is the creation of more discriminative feature maps. These cost functions have shown promising results in the image domain (facial recognition). The databases of this problem, although they have few samples per class, are considerably large since they contain many classes. Thus, in total number of samples, a system can be obtained that converts correctly to the desired solution. Unfortunately, in the audio domain, there is no FSL dataset with as many samples as is in the public domain.

Techniques based on embedding learning have the objective of reducing the dimensionality of the entrance to the network. One of the first contributions to embedding learning was in the task of signature verification. The proposed architecture is known as Siamese network [160]. The main characteristic of a system based on a Siamese network is that it is based on the instantiation of two identical networks (same architecture) and with shared weights. Thus, the entry to the system corresponds to a pair of inputs (let us suppose a pair of signatures) where the system is trained to obtain similar internal representations when the pair corresponds to the same signature and to obtain very scattered representations when the signature is different. You can see how the reduction of dimensionality is done in a smart way. Once the weights of the network are trained, the prediction is made by means of a distance metric between the signature to be predicted and those in the database. The cost function used for these networks is known as contrastive loss [161] and is defined as:

$$\mathcal{L}_{\text{contrastive}} = \sum_{i=1}^{m} [(1-Y)\frac{1}{2}(D_{W_i})^2 + (Y)\frac{1}{2}\{max(0, m - D_{W_i})\}^2] \tag{2.31}$$

where Y represents a binary that is set to 0 if both inputs correspond to the same class and set to 1 if both inputs are from different classes. m is a fixed margin that must be greater than 0. D_{W_i} denotes the distance between both input feature maps (ith pair) and can be formulated as:

$$D_{W_i}(\mathbf{x}_i^1, \mathbf{x}_i^2) = ||\mathcal{F}(\mathbf{x}_i^1) - \mathcal{F}(\mathbf{x}_i^2)||_2 \tag{2.32}$$

where $\mathbf{x}_i^1$ denotes one example of the ith pair and $\mathbf{x}_i^2$ the second example. In this case, the distance equation corresponds to the euclidean distance. m factor becomes a crucial factor since it defines a radius around $\mathcal{F}(\mathbf{x})$. Dissimilar pairs contribute to the loss function if their corresponding distance is within the defined radius [161]. Only selecting similar pairs will lead to a collapsed function. Therefore, the selection of pairs is crucial.

Triplet based networks [15] can be understood as a modification of Siamese networks. In this case, instead of instantiating 2 equal nets with tied weights, 3 are instantiated. In each triplet, the system sees an anchor sample, a positive sample (corresponding to the same class as anchor) and a negative sample (corresponding to a different class than anchor). In this case, the cost function is defined by:

$$\mathcal{L}_{triplets} = \sum_{i=1}^{m} [\ ||\mathcal{F}(\mathbf{x}_i^a) - \mathcal{F}(\mathbf{x}_i^p)||_2^2 - ||\mathcal{F}(\mathbf{x}_i^a) - \mathcal{F}(\mathbf{x}_i^n)|| + \alpha\ ] \tag{2.33}$$

where $(\mathbf{x}_i^a)$, $(\mathbf{x}_i^p)$ and $(\mathbf{x}_i^n)$ correspond to the anchor, positive and negative example respectively. α is a margin that forces the between negative pairs. As explained in the original work [15], the choice of triplets plays a crucial role. Two strategies are proposed, an offline choice every n steps and an online choice (recommended) that selects the so-called hard triplets to form the convergence of the system to a desired solution. In the online choice, one should look for which negative samples are closer to the anchor and which are more positive. Thus, the system is forced to learn relevant triplets.

As it can be observed, the main limitation in these systems is the choice of training groups. This process involves a slow and difficult to converge training. In addition, in the test phase, a function must be decided upon to determine the distance and establish a fixed threshold to decide whether or not there is a match. In the audio domain, this phenomenon can lead to false negatives. Let us imagine the problem of signature verification. When the system must discern whether a signature corresponds to a class with which it has been trained, the input has the same structure, i.e. the size of the signature is similar and the writing is the same. However, when an audio pattern is to be detected, it may have occurred at the beginning, middle, or end of the audio clip. Determining a fixed structure with a non-causal phenomenon such as enviromental sounds is extremely complicated. On the other hand, solutions based on modifying the softmax function aim to eliminate the source of error of the group selection by training the system with a single sample at a time. However, they require a large database to be able to converge. With all this background on the state of the art of FSL, it was decided to use autoencoders to perform embedding learning on an FSL problem in the audio domain such as fixed pattern recognition.

The choice of autoencoders (based on convolutional layers) for the detection of fixed patterns in this thesis is highly motivated by the contributions that it can provide for embedding and reduce the open space simultaneously. First of all, these systems (formed by convolutional layers) have shown to be able to provide relevant information about the audio spectrogram. Secondly, they allow discrimination on unknown or unwanted samples thanks to their nature of reconstructing known inputs. Therefore, both OSR and FSL consideration can be jointly mitigated with this kind of solution. This specific scenario appears in real-world environments for which the system to be deployed is intended. The contributions of this thesis regarding FSL/OSR issues and their mitigations using autoencoders can be seen in Annex B.

2.4 Complexity Considerations

2.4.1 Definition

Over the last years, Deep Learning techniques have shown promising results in the task of audio classification, either AEC or ASC. Today, CNN-based systems conform the state-of-the-art. There are many details to consider when implementing a machine listening system (without real time streaming implementation as shown in Figure 2.4) for audio clip classification. However, three fundamental design aspects may be considered: the choice of the audio representation (step 3 in Figure 2.4), the design of the classifier and the design of a post-processing module, if necessary (step 4 according to Figure 2.4).

When machine listening systems must be deployed in edge devices, analyzing their performance considering some underlying low-complexity restrictions can be of major importance. In real application environments, a trade-off must be found between accuracy and execution time. Many times, the scientific community ignores this trade-off, as the goal is usually to obtain the best possible performance without further considerations on computational cost.

In many occasions, the proposed frameworks implement ensembles of several models and this may result in the use of multiple feature extractors (since each model can be trained with a different representation), multiple classifiers and an additional information fusion scheme.

Complexity considerations have been taken into account within this thesis by studying several alternatives in the design of the neural network architecture, which are capable of improving the global performance of the system without involving a large number of extra trainable parameters.

2.4.2 Background

Complexity restrictions have not been intensively discussed within the machine listening community in recent years. Many of the proposed methods leading to improved accuracy are based on the combination of multiple independent models, also known as ensembles [162, 163, 164, 165, 166, 55, 167]. For example, a common practice is to train the same model with different representations of the audio signal [168, 169, 170]. However, approaches of this kind can be too complex to be deployed in real-world products. Note that the use of ensembles may imply, first, to extract all the relevant input representations, then, obtain the result with multiple classifiers, and lastly, combine intelligently those results to end up with a final output. All those stages can significantly increase the computation time.

In this thesis, we take into account complexity considerations by studying the impact of low-complexity techniques applied over conventional CNNs. These are shown to lead to better performance without the need to add a high number of extra parameters. The inputs to the CNNs are based on Mel spectrograms [171], and the separation of harmonic/percussive components using median filtering [172, 173], which has previously been successfully applied in audio classification tasks [163, 164]. These techniques are based on the use of Squeeze-Excitation (SE) techniques applied within the convolutional blocks of residual networks. Both techniques are briefly described in what follows.

Residual Learning

Residual learning is understood as an architecture of CNNs that was first presented in [1]. The convolutional layers that are stacked on classic CNNs are replaced by residual blocks [174, 154]. These blocks are designed to approximate the residual function $\mathcal{F}(\mathbf{X}) := \mathcal{H}(\mathbf{X}) - \mathbf{X}$, where $\mathcal{H}(\cdot)$ represents the mapping to be fit by a set of stacked layers and $\mathbf{X}$ denotes the input of the first of such stacked layers. The original function $\mathcal{H}$ can be expressed as $\mathcal{H}(\mathbf{X}) = \mathcal{F}(\mathbf{X}) + \mathbf{X}$. The original residual block can be found in Figure 2.6.

The main motivation of selecting this kind of network lies on the intuition that optimizing a residual mapping may be easier than optimizing an unreferenced one. A way to implement this kind of learning is by adding a shortcut connection that performs as identity mapping as shown in Figure 2.6. Note that this connection does not add any extra parameters to the network or computational cost. Therefore, very deep neuronal networks can be designed and trained without additional effort while reducing the vanishing gradient problem [175].

Squeeze-Excitation

As explained in Section 2.1.3, CNNs are designed by staking out several convolutional layers. The trainable coefficients of these layers are obtained by capturing local spatial relationshps (kernel size neighborhood information). The feature maps are obtained by encoding the

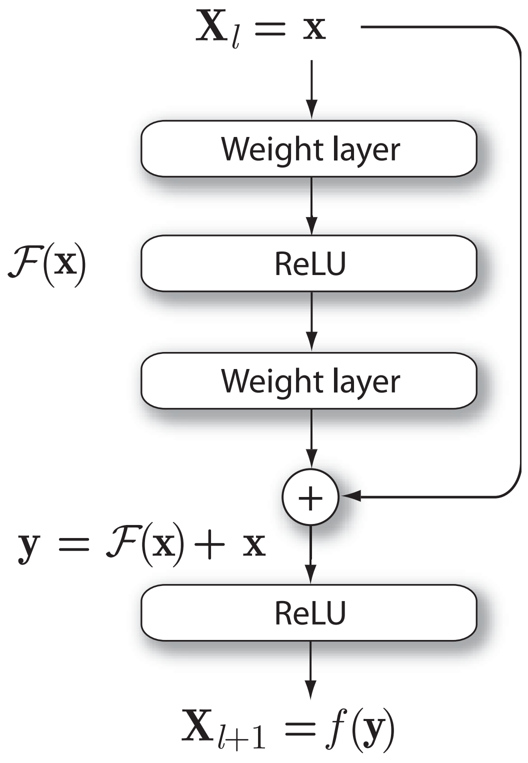
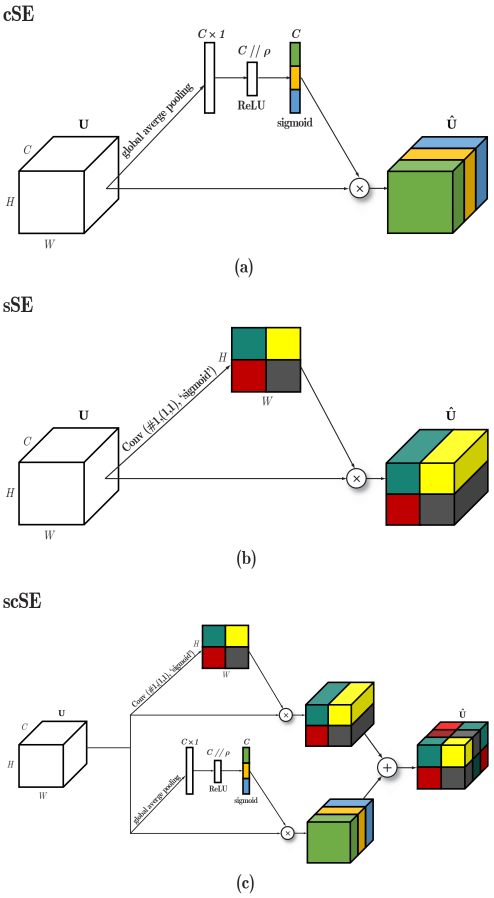

$\mathbf{X}_l = \mathbf{x}$
Weight layer
$\mathcal{F}(\mathbf{x})$
ReLU
Weight layer
+
$\mathbf{y} = \mathcal{F}(\mathbf{x}) + \mathbf{x}$
ReLU
$\mathbf{X}_{l+1} = f(\mathbf{y})$

Figure 2.6: Residual block presented in [1]. Weight layer refers to a Convolutional layer.

spatial and channel information together. With this limitation in mind, the SE techniques were first presented in [176]. The main idea is to encode the spatial and channel information independently; this process can be understood as a rescaling of the feature maps generated by the convolutional layers.

The nomenclature of this section slightly differs from that presented in Section 2.1.3. It has been decided to use this notation for consistency in the works present in the state of the art [176, 177] and the published article present in Annex A. Here, we define $\mathbf{X} \in \mathbb{R}^{H\times W\times C'}$ as an input of a convolutional layer. The feature map generated by the convolutional layer is denoted as $\mathbf{U} \in \mathbb{R}^{H\times W\times C}$. Another way to express $\mathbf{U}$ can be $\mathbf{U} = [\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_C]$ where $\mathbf{u}_i \in \mathbb{R}^{H\times W}$. Each $\mathbf{u}_i$ represents a channel output. Considering this notation, H and W represents the height and the width, while C' and C defines the number of input and output channels, respectively. The convolutional process is expressed using $\mathbf{F}(\cdot)$. Therefore, the obtaining of $\mathbf{U}$ is expressed: $\mathbf{F}(\mathbf{X}) = \mathbf{U}$. The squeeze-excitation (SE) blocks transform this output into $\hat{\mathbf{U}}$ using a function denoted by $\mathbf{F}_{SE}(\cdot)$. This rescheduling process is exemplified by $\mathbf{F}_{SE}(\cdot) : \mathbf{U} \rightarrow \hat{\mathbf{U}}$. Thus, these new feature maps are used for the forthcoming pooling and convolutional layers. The SE blocks can be used in each convolutional block or not. The function in charge of the scaling can be implemented in different ways depending on what is going to be excited and what is going to be squeezed from the feature maps.

The first SE module to be presented is the one known as cSE [176]. This module obtains a single value per channel to later obtain relationships between them (see Figure 2.7a)). So, it could be defined as a module that squeezes spatially and excites channel-wise. The obtaining of a single value per channel is done through a global average pooling, that is, from each channel an average value is obtained. This vector is denoted by the variable $\mathbf{z}$. The relationship between the channels is done by two fully-connected layers. Thanks to these two layers, the vector $\hat{\mathbf{z}}$ is obtained, which can be defined as $\hat{\mathbf{z}} = \mathbf{W}_1(\delta(\mathbf{W}_2\mathbf{z}))$ where δ represents



(a)

(b)

(c)

Figure 2.7: Diagram of different SE blocks: (a) describes cSE block procedure, (b) ilustrates sSE block framework and (c) shows scSE block by combining (a) and (b).

ReLU activation. $\mathbf{W}_1 \in \mathbb{R}^{C\times\frac{C}{\rho}}$ and $\mathbf{W}_2 \in \mathbb{R}^{\frac{C}{\rho}\times C}$ are the weights of the fully-connected layers, and ρ is a ratio parameter. Once the relationships between the channels are obtained, they are scaled in the range [0.1], sigma activation σ is used for this purpose. This last step determines the importance of each channel since this vector is multiplied with the feature maps $\mathbf{U}$. The obtainment of $\hat{\mathbf{U}}$ can be finally expressed as:

$$\hat{\mathbf{U}}_{cSE} = \mathbf{F}_{cSE}(\mathbf{U}) = [\sigma(\hat{z}_1)\mathbf{u}_1, \ldots, \sigma(\hat{z}_C)\mathbf{u}_C], \tag{2.34}$$

where $\hat{z}_k$ are the elements of the transformed vector $\hat{\mathbf{z}}$.

The second SE module is known as the sSE and was first introduced in [177]. In this case, the rescaling function is performed by a single filter and a $(1,1)$ kernel size convolutional layer (see Figure 2.7b)). Thus, in this case, the squeeze is performed channel-wise and the excitation is performed spatially. To better illustrate this module, let's represent the input feature map as $\mathbf{U} = [\mathbf{u}^{1,1}, \mathbf{u}^{1,2}, \ldots, \mathbf{u}^{i,j}, \ldots, \mathbf{u}^{H,W}]$ where $\mathbf{u}^{i,j} \in \mathbb{R}^{1\times1\times C}$. The convolution process of this module is expressed as $\mathbf{q} = \mathbf{W} \star \mathbf{U}$, being $\mathbf{W} \in \mathbb{R}^{1\times1\times C\times1}$ and $\mathbf{q} \in \mathbb{R}^{H\times W}$. Each $q_{i,j}$ represents the combination of all channels in location (i,j). Like the cSE block, the ratio must be scaled up to a range of $[0,1]$. Passing each (i,j) location through the σ function gives the relevance of that location acrosss the feature map. Thus, more relevance is given to the meaningful pixels. The output of this SE module is defined as:

$$\hat{\mathbf{U}}_{sSE} = \mathbf{F}_{sSE}(\mathbf{U}) = [\sigma(q_{1,1})\mathbf{u}^{1,1}, \dots, \sigma(q_{H,W})\mathbf{u}^{H,W}]. \tag{2.35}$$

The last block is known as scSE. It consists of the sum of the two blocks explained previously (see Figure 2.7c)). Both are done in parallel and then the outputs are added together. In this case, relevance is given to a location (i, j, c) in case it is spatially relevant as well as channel-wise. This rescaling can be defined as:

$$\hat{\mathbf{U}}_{scSE} = \hat{\mathbf{U}}_{cSE} + \hat{\mathbf{U}}_{sSE}. \tag{2.36}$$

SE modules can be found in several state-of-the-art works. In [178], a two-step neural network is presented. The first step consists of the merging of several feature maps using the network known as Xception [84]. Once merged, the last step implements a cSE module to recalibrate the feature maps obtained. In [179], a CNN 1D is presented (the input of the network is the audio itself). The CNN is formed by residual blocks that incorporate cSE modules. Finally, the SE modules have been proposed in DCASE [180]. However, few details are provided, which shows the need for further investigation on this topic.

2.5 End-to-end Frameworks

2.5.1 Definition

Although most of the solutions proposed for AEC involve the conversion of audio into a time-frequency representation, there is a less explored line of research known as end-to-end solutions. The main idea on which these frameworks are based is that all its parameters are trainable, therefore, in this context, there would not be an audio transfomation and this would be the one that would feed the classifier. The only choice the researcher has to make is to decide on the type of normalisation of the audio: to scale to the maximum value or to carry out a normalisation of the mean and standard deviation. The main advantage of these solutions is that they avoid biases in the representation for a certain dataset. As already discussed, a multitude of representations of the audio have been proposed. Such representations may show good results in a certain task and with a specific database but may not show the same results in another machine listening task. In this field a multitude of contributions can be made. However, following the research line of residual learning it has been decided to carry out a study of different residual neural networks by changing the residual module of them. The aim is to analyse which residual block fits better in an end-to-end system for audio classification.

2.5.2 Background

Most end-to-end solutions present in the state of the art are aimed at environmental sound event classification [21, 181, 182, 60] or music tagging [183, 184]. However, research is also being done on the location of sound sources from one-dimensional audio [96]. Most of these frameworks propose a 1D CNN that intercalates convolutional layers and pooling layers to reduce the dimensionality of the input vector as the network gets deeper. One of the first contributions in this field was presented in [184] where an experiment is carried out comparing the results obtained using an end-to-end system and another where the input to the system is a spectrogram in a context of music tagging. The experiment was carried out using a simple CNN and the results showed that the end-to-end framework was able to learn interesting characteristics of the input audio even though it was not able to achieve the same results as the spectrogram-based system. This same line of research continued with other works such as [185, 94]. In [183] the proposed end-to-end framework reached the performance of the spectrogram-based one in the same task as [184] dealing with music tagging. In this case, the end-to-end framework implemented residual learning in some parts of the network. In [21] a study of the same end-to-end system is performed, suggesting modifications due to the length of the audio. That is, it is intended to analyze the trade-off between the number of parameters that can be trained and the audio analysis window, that is, a network whose input is an audio of 4 seconds can be different from when it enters 1 second. This analysis is done because in many situations, the lengths of the audio clips in the same database are different and one must decide the input lengths for such audios. Other different approach is the one presented in [181], where recurrent layers are also implemented. The main objective in this case is to analyze which layer performs best at the end of the framework: recurrent or convolutional. The experimentation carried out in the work showed that convolutional layers are the most reasonable choice. In [186] a completely different motivation is followed considering a network that is a mix of a 1D CNN and a 2D CNN. The network is first fed with the audio signal and after some convolutional layers, the feature maps are resampled in order to obtain 2Ds feature maps. Then, the network changes to a 2D CNN. This approach also shows promising results.

In this thesis we analyze the performance of end-to-end networks considering different residual block alternatives. The analysis is performed by using as a baseline the work of Dai *et al.* [60], which studied the performance of different end-to-end audio classification systems, including one based on residual learning. However, while previous works in the image domain have analyzed and compared the performance of different residual block designs, such analysis has not been done for end-to-end audio networks.

2.6 Metrics and Performance Analysis

In order to evaluate the behavior of a system based on artificial intelligence techniques, it is necessary that the metrics decided upon reliably represent its behavior. Once it has been decided which set corresponds to training and testing splits (see Section 2.7 for more detail on the division of datasets), these metrics must be applied to the training and testing set. The results obtained during the training stage allow us to discern whether the system is learning the desired objective. The result obtained on the test set once the system has finished the training process allows to analyze the generalization properties of the system. The exaggerated discrepancy between training and test metrics is known as overfitting. The system learns such strict parameters for the training set that it is not able to generalize for samples that are not exactly the same. On the other hand, underfitting is the phenomenon that appears when the system is not able to capture with sufficient detail the underlying distribution of the training set. Figure 2.8 illustrates these two phenomena.

In a supervised learning environment, metrics are extracted taking the ground-truth labels into account. In the case of binary classification these 4 scenarios can be observed:

- **True positive (TP):** The system correctly predicts that the sample corresponds to the appropriate class.
- **True negative (TN):**The system correctly predicts that the sample does not correspond to the relevant class.
- **False positive (FP):** The system incorrectly predicts that the sample corresponds to the relevant class.

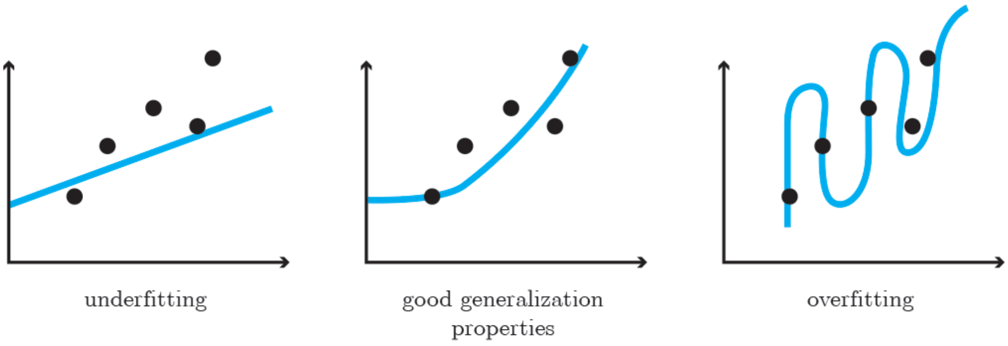


Figure 2.8: Representations of the different scenarios that can appear during the training of a system based on artificial intelligence. A two-dimensional representation was made to facilitate the visualization of these scenarios. Each axis represents a generic feature.

- **False negative (FN):** The system incorrectly predicts that the sample does not correspond to the relevant class.

The percentage of samples corresponding to each of the 4 previous definitions, allows the calculation of performance metrics.

2.6.1 Accuracy

Accuracy (ACC) is one of the most widely used metrics when analyzing artificial intelligence systems. This metric corresponds to the quotient between the number of correctly classified samples and the total number of samples. According to the previous definitions, the ACC can be formulated as:

$$ACC = \frac{TP + TN}{TP + TN + FP + FN}, \tag{2.37}$$

where TP, TN, FP, FN, refer to the previous definitions. The limitation of this metric is that it can show misleading results if the classes are not balanced.

Accuracy in Open-Set Recognition

The above definition is intended for close-set environments. When analyzing OSR systems, the accuracy metric must be modified slightly to correctly represent this phenomenon. This accuracy is known as weighted accuracy or Acc_w [187, 188]. Depending on the value of openness and how the unwanted category is configured, a different accuracy must be used for that category:

$$\begin{aligned} &O^* = 0 \text{ (without UU)}: \\ &\qquad ACC_w = wACC_{KK} + (1-w)ACC_{KU}, \end{aligned} \tag{2.38a}$$

$$\begin{aligned} &O^* \neq 0 \text{ (with KU and UU)}: \\ &\qquad ACC_w = wACC_{KK} + (1-w)ACC_{KUU}, \end{aligned} \tag{2.38b}$$

$$\begin{aligned} &O^* \neq 0 \text{ (with only UU)} : \\ &\qquad ACC_w = wACC_{KK} + (1-w)ACC_{UU}, \end{aligned} \tag{2.38c}$$

where w represents a factor that weights the accuracy between the accuracy obtained over KK classes and other unwanted classes (KU or UU). ACC_{KK} corresponds to the accuracy with which the system correctly classifies the KK samples into their respective classes.

The performance metrics involving the unwanted category (ACC_{KU}, ACC_{KUU} or ACC_{UU}), indicate the ability of the system to reject samples that do not belong to any KK class. The reason why there are three different metrics relates to the different openness conditions O^*. When the system sees all possible unwanted classes during training, the unwanted category only consists of KU classes. On the other extreme, if the system does not see any unwanted samples during training, the unwanted category is only made up of UU classes. When the system sees some of the classes pertaining to the unwanted category, the ACC_{KUU} metric is used, which is defined as the average of ACC_{KU} and ACC_{UU}. Therefore, depending on the context in which the system is to be deployed, one metric or another must be used.

2.7 Datasets

The data provided to the system are of great importance. The factors that determine a good dataset are the quality of the data provided and the proposed configuration for the analysis of the system. The creation of an audio dataset is a time-consuming and complex task. Firstly, the taxonomy of the dataset must be defined, that is to say, which categories or classes will be present and how they will be structured. Then, audio clips must be recorded in which the sound events decided in the taxonomy appear. Depending on the number of audio clips in each class the dataset can be defined as balanced or unbalanced. If the number of samples of each class is similar, the dataset can be defined as balanced. However, if there are many more samples in some specific classes than in others, the dataset can be considered as unbalanced. The recording process is a crucial phase when creating a dataset, for example, the decision of the microphone can be important. Currently, audio datasets are being generated using different microphones to address the problem of mismatched devices [188, 14]. This phenomenon appears in some real applications where datasets recorded with high quality microphones are available but the system is deployed in a device where the microphone does not have the same characteristics, such as a mobile phone. To avoid duplicate datasets with different microphones, it is being studied how to solve this issue by modifying the system [189, 190, 191, 192]. Finally, the last phase corresponds to the cleaning and labelling of the samples generated in the recording phase. This phase is the most costly in terms of time since it must be done manually by human taggers. The effort employed in these three phases: taxonomy, recording and labeling determines the quality of the audios provided by a dataset. A desirable virtue of any dataset to be used for training/validating an artificial intelligence system is its size. The more extensive it is, the better, since the system will be able to better generalize the categories with which it is being trained.

Regarding the configuration of the dataset, the two most common configurations are the one that corresponds to the creation of a k-fold (cross-validation) or a single training/validation (hold-out) division. Each configuration can be defined in the following way:

- **cross-validation configuration (Figure 2.9a)):** the dataset is divided into k folders. This division is made in order to analyze the system's generalization capability since in each iteration the system is trained with different folders. It is important to

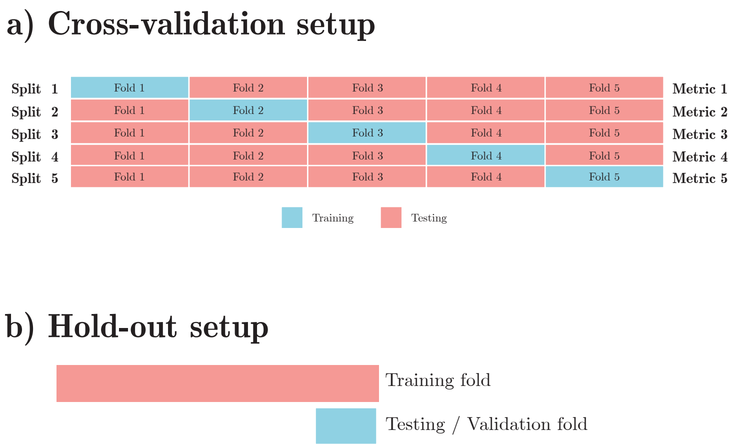


Figure 2.9: Different dataset configuration.

emphasize that each audio must appear in only one folder and that these must be balanced according to the total dataset. That is, if the dataset has the same number of samples for each class, each folder must follow this rule proportionally. The training process must be performed at least as many times as there are folders in the dataset. One folder is left to test the system and the rest are used for training. Once the k results have been obtained in the test folders, an average is made and it is detected if the system has shown a similar result for each interaction.

- **hold-out configuration (Figure 2.9b)):** the dataset is divided into two parts: training and testing. An extended ratio is 80%-20% for training-testing respectively.

If both configurations are compared, advantages can be found in each of them. The choice of configuration is determined in part by what is to be analyzed. In general, a cross-validation configuration provides more information when it comes to showing the behaviour of the system when faced with unseen data. That is, at least, information is obtained from k unseen data scenarios. So, if it is intended to analyze the generalization capacity of a system, the k-fold configuration is very useful. The hold-out configuration has the limitation that only one partition has been performed and the results are determined by the way the dataset was partitioned. However, the hold-out configuration is very useful when a particular part of the system is to be analysed. Let's suppose a framework that is formed by a feature extractor, a classifier and a post-processing module. If a study of different feature extractors is to be examined, a hold-out configuration is very useful because in all cases the system has used the same data in training and test stages and the comparison among them is trustworthy. Furthermore, this configuration is highly advisable when a very large dataset is available or when it is used for competitive purposes.

The public datasets used to study the different problems presented in this thesis are described below.

2.7.1 ESC-10

The dataset called environmental sound classification (ESC) was released in 2015 [85]. It should be noted that this dataset has several versions depending on the number of classes that compose it. The one known as ESC-10 (used in this thesis) is composed of 10 classes. The classes that compose it are: *sneezing, dog barking, clock ticking, crying baby, crowing rooster, rain, sea waves, fire crackling, helicopter* and *chainsaw*. Each class is composed of 40 audios, with a total of 400 audio clips available. The configuration of this dataset corresponds to a cross-validation one, being divided in 5 folders. The duration of each audio is 5 seconds, padding it with zeros if necessary. It should be noted that this dataset has not been recorded by the designer, but it is composed of audios present in the Freesound[6] audio platform. Freesound is a collaborative repository where anyone can upload or download audio clips. The dataset ESC-10 can be downloaded at the following link.[7]

2.7.2 UrbanSound8k

Urbansound8k was released in 2014 [82]. As its name indicates, it is composed of urban sounds, they are grouped into the following 10 classes: *air conditioner, car horn, children playing, dog bark, drilling, idling enginge, gun shot, jackhammer, siren* and *street music.* The audio is composed of 8732 audio clips. Like ESC-10, the audios have been extracted from the Freesound repository. In this dataset, the duration of the audio chunks is 4 seconds maximum (filling in with zeros in case the duration is lower). The configuration is cross-validation, being the division of the dataset in 10 folders. As explained in the download web,[8] the k-fold must be done correctly so that the results are comparable with the rest of the literature. The samples are thoroughly divided in the different folders to show a different behaviour in each of them.

2.7.3 TAU Urban Acoustic Scenes 2019

This dataset corresponds to the one released in Task 1 of the DCASE 2019 edition[9] [81]. The aim of this task is to classify an audio clip in one of the possible predefined scenes (ASC). The scenes have been recorded in 12 different European cities such as: Amsterdam, Barcelona, Helsinki, Lisbon, London, Lyon, Madrid, Milan, Prague, Paris, Stockholm and Vienna. The locations of the scenes are: *airport, shopping mall, metro station, street pedestrian, public square, street traffic, tram, bus, metro* and *park.* The recording device is the Soundman OKM II Klassik/studio A3, electret binaural microphone and a Zoom F8 audio recorder using 48kHz sampling rate and 24 bit resolution. Unlike the rest, the audio clips were recorded by the team that designed the dataset. The dataset was collected by Tempere University of Technology (Finland).The duration of the audios is 10 seconds each, having a total of 40 hours of recording. The classes are balanced. The configuration of this dataset is hold-out. There is a 70%-30% partition providing 9185 audio clips to train the system and 4185 to test it. There are a total of 14400 audio clips which corresponds to 144 per city per acoustic scene. The training set only has audio clips from 9 cities to be able to test the generalization properties of the system. Audios from the city of Milan only appear in the test phase. As you can see, with this configuration there are only 10 cities. The other 2 remaining cities appear in another release called "evaluation" which is used to rank the systems in the DCASE challenge. This release is not used in this thesis since the audio notes are not public and this release has a competitive nature.

[6] https://freesound.org/

[7] https://github.com/karolpiczak/ESC-50

[8] https://urbansounddataset.weebly.com/urbansound8k.html

[9] https://zenodo.org/record/2589280

2.7.4 An Open-Set Recognition and Few-Shot Learning Dataset for Audio Event Classification in Domestic Environments

For the study of the OSR and FSL phenomena, the design of a specific dataset has been required. This dataset can be considered as a contribution to this thesis [187]. The dataset is composed of two categories. The first one is composed of 24 KK patterns and the second one of 10 unwanted classes. The difference between pattern and class should be highlighted. In this case, pattern refers to the set of samples of the same class but with the peculiarity that these samples have a very similar spectrogram because in this case, they correspond to domestic alarms. The Log-Mel spectrogram of the 24 patterns provided in the dataset can be seen in Figure 2.10. The log-Mel spectrogram has been calculated with a window size of 40 ms, an overlap of 50% and 64 Mel filters. All frequency bins have been normalized with zero mean and standard deviation equal to one.

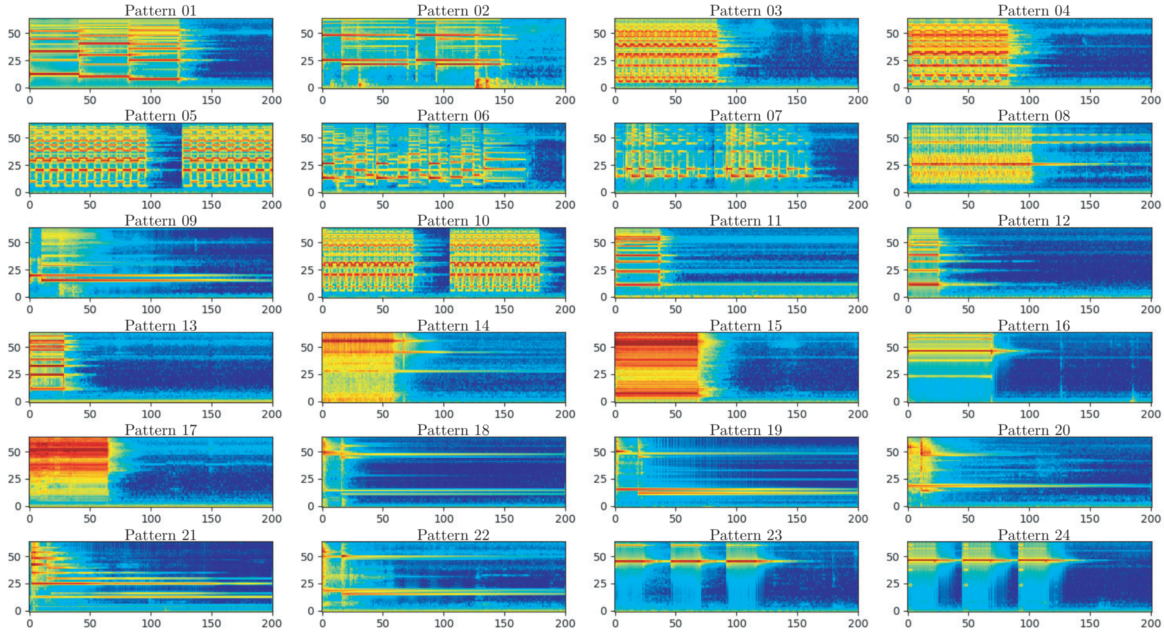


Figure 2.10: Log-Mel spectrogram of the sounds in *pattern sounds* category. One sample per specific pattern is shown. The horizontal axis correspond to the time frame and the vertical axis to the Mel frequency band.

Figure 2.11 below shows the difference between pattern and class as conceived in this dataset. For this, 3 examples of a pattern and 3 examples of two unwanted classes are shown.

Each of the 34 classes (24 patterns and 10 unwanted) is composed of 40 audio clips. The classes that compose the unwanted category are: car horn, clapping, cough, door slam, engine, keyboard tapping, music, pots and pans, steps and water falling. All the audios have been recorded with a sample rate of 16 kHz, 16 bits per sample, PCM and mono encoding. The dataset is therefore designed so that a machine listening system is able to classify each pattern in its corresponding class and to reject all unwanted classes, i.e. to determine the sample as unwanted regardless of the class it belongs to.

To evaluate the OSR phenomenon, 3 different scenarios have been carried out according to the manipulation of the classes belonging to the unwanted category. In the first scenario, all unwanted classes are seen by the system in training stage. In the second scenario, only

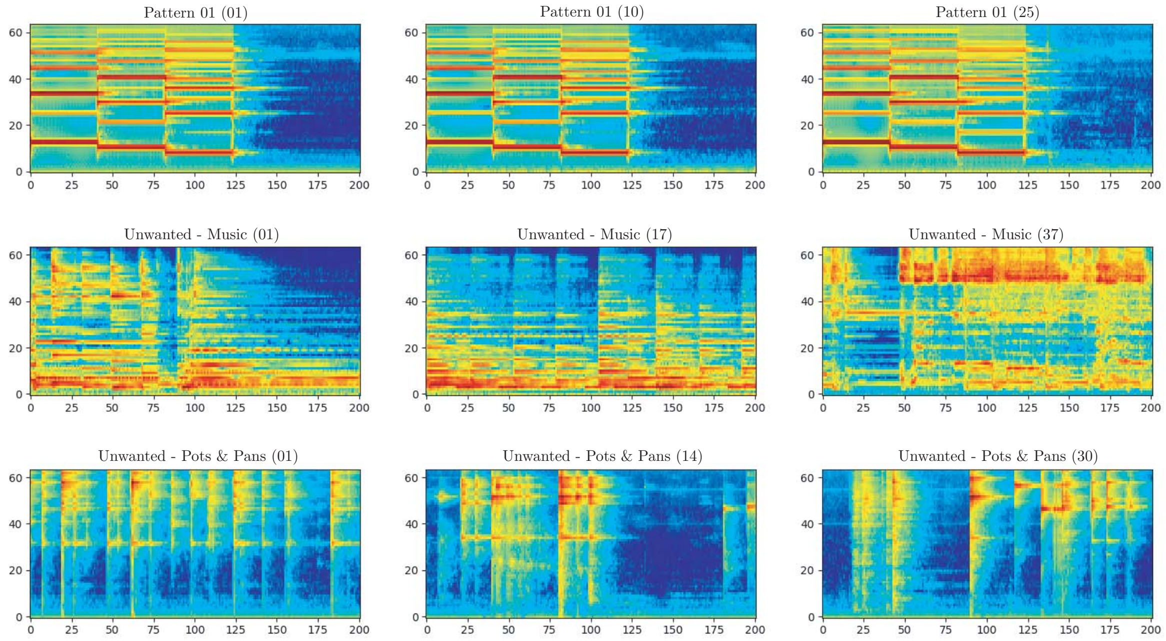


Figure 2.11: Comparison of log-Mel spectrograms form within-class examples corresponding to a pattern sound (first row) and two classes from the unwanted category (second and third rows). Note that inter-class variability of the examples in the first row is considerably smaller than in the rest, that is why it is considered as a sound pattern. The number in parenthesis denotes the example index within the class.

half of the unwanted classes are used in training, i.e. there are 5 KU classes and 5 UU classes. The last scenario corresponds to a maximum openness value since no unwanted classes are used in training. These 3 unwanted scenarios have been studied in two different contexts according to the number of KK classes. In the first context, all classes (all 24 available) have to be classified. As it can be appreciated, having a higher number of KK classes than unwanted (either KU or UU), the value of openness is not very high. Therefore, a second context is created where the 24 patterns are divided into 8 groups of 3. With this more reliable configuration to a real environment, higher openness values are achieved. The following table shows the different openness values for each configuration described above:

Pattern Sounds	**KK**	**KU**	**UU**	C_{TR}	C_{TE}	O^*
Full set	24	10	0	34	34	0
		5	5	29	34	0.04
		0	10	24	34	0.09
Trios	3	10	0	13	13	0
		5	5	8	13	0.13
		0	10	3	13	0.39

Table 2.2: Number of classes of each configuration and the corresponding openness value.

With regard to FSL consideration, the dataset is prepared so that the previously described scenarios can be trained with 4, 2 or 1 shots. The classes of the patterns sounds category and the unwanted category are trained with the same number of shots. The configuration of this dataset corresponds to cross-validation. However, this dataset has the peculiarity that the number of k folders, depends on the number of shots with which the system is going to be trained. This means that when training with 4 samples per class, $k = 10$. For 2 shots, $k = 20$. Finally, when training with one shot, $k = 40$, each sample is an independent folder.

The dataset is presented with a baseline based on transfer learning. The L3net [68] network is used for the extraction of features prior to training. The final classifier consists of a DNN with sigmoid activation in the last layer to mitigate the OSR test. A feature map using a t-SNE of the L3 features is shown below where the audio clips of the KK classes are represented.

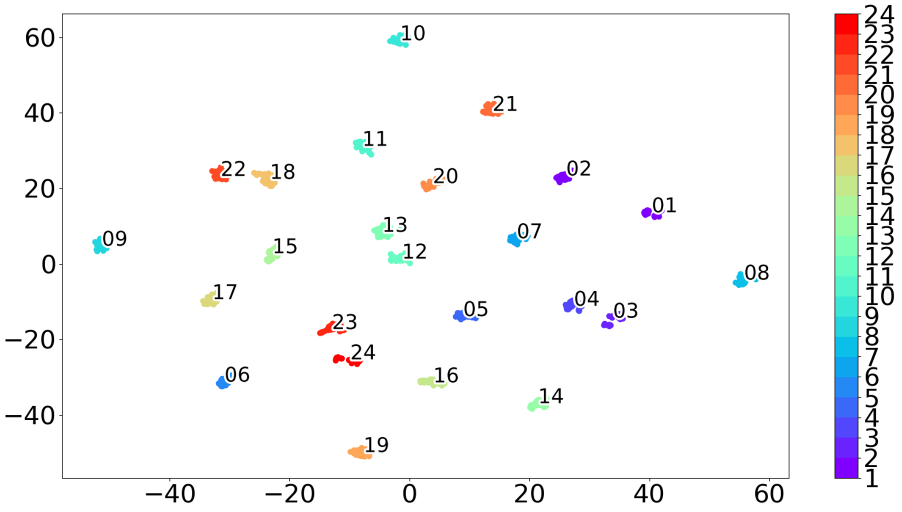


Figure 2.12: t-SNE mapping from L3 representation of 24 KK classes.

As it can be noticed (see Figure 2.12), the different classes create very concentrated clusters. There is no sample of one class that appears far from the others. However, there are clusters that are relatively close together in the feature space. Classes 3/4 or 23/24 are an example. As can be seen in Figure 2.10, although they are different patterns, they have a very similar spectro-temporal representation.

The dataset proposed in this thesis can be downloaded at the following link.[10]

[10] https://zenodo.org/record/3689288

Chapter 3

Contributions

The contributions of this thesis can be summarized into three main results, each of them corresponding to one of the scientific publications making up the compendium (see Annexes). Each particular contribution is somehow related to the problems presented in Chapter 2. In summary, such contributions can be summarized as follows:

- **Contribution 1**:
The first publication attached to this thesis (see Annex A) shows an analysis of different residual networks implemented by residual blocks including squeeze-excitation (SE) methods. In order to study with better detail the contribution of these methods in residual blocks, a purely residual block (without SE) was also added to the study. Some of the mixed blocks (residual and SE) analyzed had already been proposed in different works of the state of the art. The main contribution in this topic is the proposal of a novel residual block with SE techniques that adds the shortcut twice, before and after the SE block. The results show that this configuration was able to improve significantly the performance in an Acoustic Scene Classification (ASC) task. An important aspect from a practical perspective is that all the mixed configurations have the same number of parameters. In addition to an overall performance analysis, a class-wise and a statistical analysis of significance using McNemar's test was carried out. The results confirm that slight modifications in residual convolutional neural networks can lead to different feature maps and performance even if they have the same number of parameters. With the novel configurations proposed in this study, we managed to improve the performance of residual networks with SE without adding additional parameters. Thus, the results show that the addition of a controlled number of parameters under a proper design can lead to systems of improved performance.

- **Contribution 2**:
The results corresponding to our proposal to mitigate the FSL/OSR problems can be found in Annex B. Regarding these two problems, the first contribution made in the framework of this thesis has been the creation of an audio dataset that takes into account the mentioned problems jointly. For this purpose, in addition to the recording of 24 audio patterns related to different types of alarms, several generic audio classes were recorded, as keyboard tapping or foot steps, among others. This dataset can be effectively used to analyze if the proposed systems are capable of rejecting these unwanted classes while correctly classifying the target audio patterns. Each class (24 pattern classes and 10 unknown generic classes) has 40 samples. Both considerations are studied from different approaches. For example, the final dataset counts several configuration files where the patterns are grouped in groups of 4, 2 or 1 samples to take into

account different FSL degrees. Regarding the OSR problem, the dataset also has different grouping files where the unknown classes, (known-unknown or unknown-unknown) are grouped differently in order to study different openness values. To increase this value and make the OSR consideration even more complex, the dataset also has a configuration where the patterns are grouped in groups of 3, thus creating a higher openness situation. This dataset was released along with a baseline system based on transfer learning. The journal paper describing this dataset and baseline is currently under review. In this context, another contribution in this thesis is related to the improvement of the dataset baseline. For this purpose, a framework based on autoencoders was proposed. As shown in Annex B, this framework is capable of generating more robust representations of the target audio patterns, thus achieving a better description leading to a better rejection of the unwanted classes. In addition, such representations can be obtained without a high number of samples. However, the performance of the system deteriorates as less samples are used (4 to 1). Since the dataset has the information of the class to which each audio clip corresponds, a semi-supervised architecture of the autoencoder was also proposed. The results show that this framework shows the best results in most of the configurable scenarios in the dataset, being a solution that performs robustly under both FSL and OSR situations in the context of audio pattern detection.

- **Contribution 3**:
  End-to-end networks are increasingly attracting the attention of the scientific community. These types of architectures can lead to more generic solutions, as several choices related to the selection of meaningful two-dimensional audio representations would be avoided. Although most end-to-end solutions in the audio domain employ residual blocks, to the best of the author's knowledge there are no studies that analyze which kind of residual block works better in this domain, in contrast to other similar studies already conducted in computer vision. The contribution of this thesis with this regard corresponds to a comparative analysis of different residual blocks implemented in end-to-end solutions for the classification of sound events. The results obtained using two different datasets show that the conclusions drawn in the image domain cannot be easily extrapolated to the audio domain. Apart from a global analysis, a Friedman non-parametric statistical test is performed, concluding that further considerations are required when deploying these solutions to audio-related problems.

Chapter 4

Conclusions

Interest from the scientific community in the search for machine listening systems has increased considerably over the years. Unlike other fields of artificial intelligence (AI) such as speech recognition or computer vision (much more established areas with commercial solutions nowadays) the problems dealing with the understanding of general environmental sounds have only attracted widespread research interest over the last decade. This increase can be easily verified by looking into the number works published in this field in the last years, as well as the interest that the well-known *DCASE* (Detection and Classification of Acoustic Scenes and Events) Challenge and Workshop have aroused. The demand for machine listening solutions is due to the number of existing applications that can benefit from this technology: home assistants, early detection of breakdowns, smart hearing aids, ambient assisted living systems, sound retrieval, security applications, autonomous driving, or gaming, among many others. However, most of the works proposed in the field of environmental sounds involve a very controlled and ideal context that does not always match that of commercial applications. In fact, the great success of AI applications in other domains, such as imaging, is based on two necessary assumptions that should hold in practice for assuring a proper performance: a controlled environment allowing for a close-set assumption and the provision of a sufficiently large and correctly labeled dataset. While it is true that some applications are already being deployed on edge devices such as mobile phones, a number of them are designed to run on powerful computers (higher computing capacity) or in the cloud. However, certain applications in the audio domain, such as the ones considered in the industrial context of this thesis, are subjected to privacy preservation constraints that prevent the use of audio data exchange out of the user's premises.

In this thesis, the problem of classification of acoustic scenes and events under a series of restrictions commonly present in real-world audio applications has been addressed. Such restrictions are related to the deployment of audio classification systems in uncontrolled environments, with little computational capacity and that must be trained with only a few samples. This is the scenario that real products like Visualfy Home and Visualfy Places have to face. As previously explained, these systems are aimed at monitoring certain sound patterns such as fire alarms or doorbells. The first one is designed to be deployed in a domestic environment and the second one in a free attendance environment such as a museum. The first problem that has been faced has been the one known as Open-Set Recognition (OSR). This phenomenon appears when the system must reject examples at test time that do not belong to any of the classes on which the algorithm has been trained. In our scenario, this translates to a system that must detect a fire alarm among all the sound events that can occur in a home or in a public space. Therefore it is necessary to train machine listening systems taking into account the OSR context, so that once they are deployed, false positives are minimized (samples incorrectly detected as alarms).

Collecting large number of samples for each specific sound to be detected can be difficult. Consider for example the case of the Visualfy Home. If users want to monitor the doorbell of the building main entrance and the doorbell of their apartments, they may record several samples of both sounds, but not too many. In fact, the number of times the user must record each sound would be inversely proportional to the user experience. Moreover, in the case of the Visualfy products, users suffer from hearing impairments, which is an additional barrier for setting properly the system. This situation leads to a Few-Shot Learning (FSL) scenario. AI algorithms have shown outstanding results when trained with a large number of samples per class, however, their performance is severely degraded when they are trained with just a few samples, e.g. 4. This second problem is also addressed in this thesis, proposing a solution aimed at tackling jointly the FSL and OSR problems.

Finally, another restriction appearing in the considered products are those arising from the privacy restrictions of the system and the deployment over resource-constrained devices. In this context, the data collected in a user's home cannot leave the system and all the processing must be done in the device itself. Moreover, since the system has to perform other simultaneous tasks (communication with detectors, communication with APIs, etc.), it is necessary to optimize the execution time of the algorithms without affecting their classification performance. Thus, the design of models that incorporate mechanisms that can considerably improve the performance without adding many additional parameters is a very desired feature.

In **Chapter 2**, a broad overview of the different aspects covered in this thesis has been presented. First, several AI concepts were introduced. A generic machine listening pipeline oriented towards the development of commercial products has been presented, as well as the well-established DCASE framework (a challenge/workshop used as a backbone to analyze the interest of the community on machine listening technologies). Finally, the specific problems addressed in this thesis have been discussed, namely, OSR, FSL, low-complexity models and end-to-end solutions.

The contributions made in this thesis, enumerated for each article that makes up the compendium, have been summarized in **Chapter 3**, whereas the conclusions of each of these articles are summarized and explained below.

Annex A analyzed the performance of residual block designs employing squeeze-excitation techniques, proposing novel alternatives in this context. It was shown that these blocks are able to improve the performance of the system with respect to the use of purely residual blocks. The performance was analyzed in terms of global and class-wise accuracy, using McNemar's test for assessing the statistical significance of the results. It is concluded that slight modifications to the configuration of the network, in this case by creating residual blocks including squeeze-excitation, can improve the global performance without the addition of a large number of extra parameters.

The problems of FSL and OSR were addressed in **Annex B**. To mitigate both problems at the same time, a two-step learning framework was proposed that makes use of a convolutional autoencoder and a multi-layer perceptron. The goal of the autoencoder is the creation of robust embeddings corresponding to the known classes and the goal of the classifier is to recognize if an embedding belongs to a class to be classified or not. Two different configurations of autoencoders were analyzed: unsupervised and semi-supervised. Several scenarios considering different degrees of openness and different number of shots are compared with a baseline based on transfer learning. The results show that the proposed framework is capable of classifying and rejecting known and unknown classes trained with few samples. In most cases, our proposal improves the performance of the baseline and, in turn, the semi-supervised configuration shows better metrics than the unsupervised one.

Annex C considered the use of end-to-end frameworks for audio event classification. Since end-to-end residual networks are known to be successful in the image domain, we analyzed the performance of residual block alternatives in the context of end-to-end audio classification. The study considered two different datasets and waveform input normalizations. The results showed that special considerations should be taken when working with raw audio data in the design of end-to-end residual networks, as the best performing blocks in the image domain are not generally the best for audio data. In addition, it was shown how the pre-processing that can be applied to the audio data may be relevant in some cases, as significantly different results are obtained even when the same network architecture is used.

In summary, the contributions proposed in this thesis show solutions that mitigate the problems present in machine listening products deployed in real scenarios. First, it has been shown how incorporating slight modifications to deep neural network architectures (e.g. squeeze-excitation) can lead to improve classification performance. Then, the power of autoencoder-based architectures for generating robust embeddings to address the FSL and OSR problems has been discussed. Finally, the potential of end-to-end audio solutions has been analyzed, showing that suitably designed residual blocks can lead to improved performance over well-established configurations widely used in the image domain.

4.1 Further work

This thesis has addressed some problems that are very present in machine listening solutions in commercial products such as home assistants. However, despite the research results were satisfactory, they indicate that there is work to be done in these areas. Future lines of research would include:

- **One-shot learning**: our results confirmed that the improvement in accuracy for systems trained on very few samples are very dependent on the actual number of instances used in the learning stage. Despite satisfactory results were obtained with only 4 samples, the ideal case would be to achieve good results with only one sample required from the user. Surely, some prior knowledge would be necessary in such case, and further research should be done on how to incorporate priors effectively into the learning process when the system must learn from just a single sample.

- **OSR robustness**: although the rejection rate of the system is acceptable for unknown samples, there is a certain predisposition of the system to classify known samples as unknown as the system has fewer samples per known class. In parallel with the previous point, it should be investigated how to make systems robust to false negatives when the number of shots decreases.

- **Robustness to incorrect segmentation**: the proposed OSR-FSL system has been analyzed with correctly segmented samples, that is, the start of the audio clip corresponds to the start of the sound event (e.g. the fire alarm). However, it may happen that the segmentation process is not perfect, leading to uncorrect or weakly segmented events [193, 194], where audio events or where the clips have a considerable temporal uncertainty. Therefore, this possible phenomenon should be characterized and corrected, if necessary.

- **Modified convolutions**: there are a variety of new convolutional blocks [84, 8, 195, 196] that present an improvement on detection or classification problems. However, they are analyzed considering very deep neural networks. Therefore, it should be studied whether these new blocks show these satisfactory results in the audio domain and in low-complexity contexts.

4.2 Publications

The publications related to this thesis are presented in this section. Apart from the articles that make up the compendium, 3 more conference papers and 5 technical reports corresponding to the participation in DCASE have also been published.

Compendium Journal Articles

Publication 1:

J. Naranjo-Alcazar, S. Perez-Castanos, P. Zuccarello, and M. Cobos, "**Acoustic scene classification with squeeze-excitation residual networks**", in *IEEE Access*, vol. 8, pp. 112287-112296, June 2020. (doi: 10.1109/ACCESS.2020.3002761)

Publication 2:

J. Naranjo-Alcazar, S. Perez-Castanos, P. Zuccarello, F. Antonacci, and M. Cobos, "**Open set audio classification using autoencoders trained on few data**", in *Sensors*, vol. 20, no. 13, pp. 3741, July 2020. (doi: https://doi.org/10.3390/s20133741)

Publication 3:

J. Naranjo-Alcazar, S. Perez-Castanos, I. Martín-Morató, P. Zuccarello, F. J. Ferri, and M. Cobos, "**A Comparative Analysis of Residual Block Alternatives for End-to-End Audio Classification**", in *IEEE Access*, vol. 8, pp. 188875-188882, October 2020. (doi: 10.1109/ACCESS.2020.3031685)

Journal Articles Under Review

Publication Under Review:

J. Naranjo-Alcazar, S. Perez-Castanos, P. Zuccarello and M. Cobos, "**An Open-Set Recognition and Few-Shot Learning Dataset for Audio Event Classification in Domestic Environments**", submitted to *Expert System with Applications*

Related Publications in International Conferences

Publication 4:

S. Perez-Castanos, J. Naranjo-Alcazar, P. Zuccarello and M. Cobos,"**Listen Carefully and Tell: An Audio Captioning System Based on Residual Learning and Gammatone Audio Representation**," in *DCASE2020 Workshop*, Tokyo, Japan, November 2020, pp. 145-149.
(doi: https://doi.org/10.5281/zenodo.4061782)

Publication 5:

S. Perez-Castanos, J. Naranjo-Alcazar, P. Zuccarello and M. Cobos, "**Anomalous Sound Detection using Unsupervised and Semi-Supervised Autoencoders and Gammatone Audio Representation**," in *DCASE2020 Workshop*, Tokyo, Japan, November 2020, pp. 150-154.
(doi: https://doi.org/10.5281/zenodo.4061782)

Publication 6:

S. Perez-Castanos, J. Naranjo-Alcazar, P. Zuccarello, M. Cobos and F. J. Ferri, "**CNN depth analysis with different channel inputs for Acoustic Scene Classification**," in *URSI 2020*, Málaga, Spain, September 2020.

Technical Reports related to DCASE tasks

Technical Report 1:

J. Naranjo-Alcazar, S. Perez-Castanos, P. Zuccarello and M. Cobos, "**DCASE 2019: CNN Depth Analysis with Different Channel Inputs for Acoustic Scene Classification**," in *DCASE2019 Challenge*, June 2019.
`http://dcase.community/documents/challenge2019/technical_reports/DCASE2019_Naranjo-Alcazar_13.pdf`

Technical Report 2:

J. Naranjo-Alcazar, S. Perez-Castanos, P. Zuccarello and M. Cobos,"**Task 1 DCASE 2020: ASC with Mismatch Devices and Reduced Size Model Using Residual Squeeze-Excitation CNNs**," in *DCASE2020 Challenge*, June 2020.
`http://dcase.community/documents/challenge2020/technical_reports/DCASE2020_Naranjo-Alcazar_34_t1.pdf`

Technical Report 3:

J. Naranjo-Alcazar, S. Perez-Castanos, P. Zuccarello and M. Cobos, "**TASK 2 DCASE 2020: Anomalous Sound Detection Using Unsupervised and Semi-Supervised Autoencoders and Gammatone Audio Representation**," in *DCASE2020 Challenge*, June 2020.
`http://dcase.community/documents/challenge2020/technical_reports/DCASE2020_Naranjo-Alcazar_34_t2.pdf`

Technical Report 4:

J. Naranjo-Alcazar, S. Perez-Castanos, J.Ferrandis, P. Zuccarello and M. Cobos, "**TASK 3 DCASE 2020: Sound Event Localization and Detection Using Residual Squeeze-Excitation CNNs**," in *DCASE2020 Challenge*, June 2020.

`http://dcase.community/documents/challenge2020/technical_reports/DCASE2020_Naranjo-Alcazar_34.pdf`

Technical Report 5:

J. Naranjo-Alcazar, S. Perez-Castanos, P. Zuccarello and M. Cobos, "**Task 6 DCASE 2020: Listen Carefully and Tell: An Audio Captioning System Based on Residual Learning and Gammatone Audio Representation**," in *DCASE2020 Challenge*, June 2020.

`http://dcase.community/documents/challenge2020/technical_reports/DCASE2020_Naranjo_Alcazar_34_t6.pdf`

Bibliography

[1] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[2] Lu Tan and Neng Wang. Future internet: The internet of things. In *2010 3rd international conference on advanced computer theory and engineering (ICACTE)*, volume 5, pages V5–376. IEEE, 2010.

[3] Luigi Atzori, Antonio Iera, and Giacomo Morabito. The internet of things: A survey. *Computer networks*, 54(15):2787–2805, 2010.

[4] Yuma Koizumi, Shoichiro Saito, Hisashi Uematsu, Yuta Kawachi, and Noboru Harada. Unsupervised detection of anomalous sound based on deep learning and the neyman–pearson lemma. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 27(1):212–224, 2018.

[5] Konstantinos Drossos, Sharath Adavanne, and Tuomas Virtanen. Automated audio captioning with recurrent neural networks. In *2017 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, pages 374–378. IEEE, 2017.

[6] Sharath Adavanne, Archontis Politis, Joonas Nikunen, and Tuomas Virtanen. Sound event localization and detection of overlapping sources using convolutional recurrent neural networks. *IEEE Journal of Selected Topics in Signal Processing*, 13(1):34–48, 2018.

[7] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[8] Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alex Alemi. Inception-v4, inception-resnet and the impact of residual connections on learning. *arXiv preprint arXiv:1602.07261*, 2016.

[9] Emre Cakır, Toni Heittola, and Tuomas Virtanen. Domestic audio tagging with convolutional neural networks. *IEEE AASP Challenge on Detection and Classification of Acoustic Scenes and Events (DCASE 2016)*, 2016.

[10] Annamaria Mesaros, Toni Heittola, and Tuomas Virtanen. Acoustic scene classification: an overview of dcase 2017 challenge entries. In *2018 16th International Workshop on Acoustic Signal Enhancement (IWAENC)*, pages 411–415. IEEE, 2018.

[11] Sharath Adavanne, Giambattista Parascandolo, Pasi Pertilä, Toni Heittola, and Tuomas Virtanen. Sound event detection in multichannel audio using spatial and harmonic features. *arXiv preprint arXiv:1706.02293*, 2017.

[12] Walter J Scheirer, Anderson de Rezende Rocha, Archana Sapkota, and Terrance E Boult. Toward open set recognition. *IEEE transactions on pattern analysis and machine intelligence*, 35(7):1757–1772, 2012.

[13] Yaqing Wang, Quanming Yao, James T Kwok, and Lionel M Ni. Generalizing from a few examples: A survey on few-shot learning. *ACM Computing Surveys (CSUR)*, 53(3):1–34, 2020.

[14] Toni Heittola, Annamaria Mesaros, and Tuomas Virtanen. Acoustic scene classification in dcase 2020 challenge: generalization across devices and low complexity solutions. In *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2020 Workshop (DCASE2020)*, 2020. Submitted.

[15] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 815–823, 2015.

[16] Yandong Wen, Kaipeng Zhang, Zhifeng Li, and Yu Qiao. A discriminative feature learning approach for deep face recognition. In *European conference on computer vision*, pages 499–515. Springer, 2016.

[17] Shubham Dokania and Vasudev Singh. Graph representation learning for audio & music genre classification. *arXiv preprint arXiv:1910.11117*, 2019.

[18] Walter J Scheirer, Lalit P Jain, and Terrance E Boult. Probability models for open set recognition. *IEEE transactions on pattern analysis and machine intelligence*, 36(11):2317–2324, 2014.

[19] Yuzhong Wu and Tan Lee. Reducing model complexity for dnn based large-scale audio classification. In *2018 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 331–335. IEEE, 2018.

[20] Jongpil Lee, Jiyoung Park, Keunhyoung Luke Kim, and Juhan Nam. Samplecnn: End-to-end deep convolutional neural networks using very small filters for music classification. *Applied Sciences*, 8(1):150, 2018.

[21] Sajjad Abdoli, Patrick Cardinal, and Alessandro Lameiras Koerich. End-to-end environmental sound classification using a 1d convolutional neural network. *Expert Systems with Applications*, 136:252–263, 2019.

[22] Warren S McCulloch and Walter Pitts. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, 5(4):115–133, 1943.

[23] Donald Olding Hebb. *The organization of behavior: a neuropsychological theory.* J. Wiley; Chapman & Hall, 1949.

[24] Daniel Crevier. *AI: the tumultuous history of the search for artificial intelligence.* Basic Books, Inc., 1993.

[25] Frank Rosenblatt. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, 65(6):386, 1958.

[26] Sourav Dutta. An overview on the evolution and adoption of deep learning applications used in the industry. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 8(4):e1257, 2018.

[27] James Lighthill. Artificial intelligence: A general survey. In *Artificial Intelligence: a paper symposium*, pages 1–21. Science Research Council London, 1973.

[28] Paul J Werbos. Backpropagation through time: what it does and how to do it. *Proceedings of the IEEE*, 78(10):1550–1560, 1990.

[29] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

[30] S Rasoul Safavian and David Landgrebe. A survey of decision tree classifier methodology. *IEEE transactions on systems, man, and cybernetics*, 21(3):660–674, 1991.

[31] Yann LeCun, Patrick Haffner, Léon Bottou, and Yoshua Bengio. Object recognition with gradient-based learning. In *Shape, contour and grouping in computer vision*, pages 319–345. Springer, 1999.

[32] Timo Ojala, Matti Pietikainen, and David Harwood. Performance evaluation of texture measures with classification based on kullback discrimination of distributions. In *Proceedings of 12th International Conference on Pattern Recognition*, volume 1, pages 582–585. IEEE, 1994.

[33] Khaled El-Maleh, Mark Klein, Grace Petrucci, and Peter Kabal. Speech/music discrimination for multimedia applications. In *2000 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings (Cat. No. 00CH37100)*, volume 4, pages 2445–2448. IEEE, 2000.

[34] Zvi Kons, Orith Toledo-Ronen, and M Carmel. Audio event classification using deep neural networks. In *Interspeech*, pages 1482–1486, 2013.

[35] Minkyu Lim, Donghyun Lee, Hosung Park, Yoseb Kang, Junseok Oh, Jeong-Sik Park, Gil-Jin Jang, and Ji-Hwan Kim. Convolutional neural network based audio event classification. *KSII Transactions on Internet & Information Systems*, 12(6), 2018.

[36] Minkyu Lim, J Kim, K Kim, and J Kim. Audio event classification using deep neural networks. *Phonetics and Speech Sciences*, 7(4):27–33, 2015.

[37] Emre Cakir, Toni Heittola, Heikki Huttunen, and Tuomas Virtanen. Polyphonic sound event detection using multi label deep neural networks. In *2015 international joint conference on neural networks (IJCNN)*, pages 1–7. IEEE, 2015.

[38] Emre Cakır, Giambattista Parascandolo, Toni Heittola, Heikki Huttunen, and Tuomas Virtanen. Convolutional recurrent neural networks for polyphonic sound event detection. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 25(6):1291–1303, 2017.

[39] Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *ICML*, 2010.

[40] Djork-Arné Clevert, Thomas Unterthiner, and Sepp Hochreiter. Fast and accurate deep network learning by exponential linear units (elus). *arXiv preprint arXiv:1511.07289*, 2015.

[41] Bing Xu, Naiyan Wang, Tianqi Chen, and Mu Li. Empirical evaluation of rectified activations in convolutional network. *arXiv preprint arXiv:1505.00853*, 2015.

[42] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[43] Kevin P Murphy. *Machine learning: a probabilistic perspective*. MIT press, 2012.

[44] Claude Sammut and Geoffrey I. Webb, editors. *Encyclopedia of Machine Learning*. Springer US, Boston, MA, 2010.

[45] Kunihiko Fukushima and Sei Miyake. Neocognitron: A self-organizing neural network model for a mechanism of visual pattern recognition. In *Competition and cooperation in neural nets*, pages 267–285. Springer, 1982.

[46] Yann LeCun, Bernhard Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne Hubbard, and Lawrence D Jackel. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4):541–551, 1989.

[47] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[48] David H Hubel and Torsten N Wiesel. Receptive fields, binocular interaction and functional architecture in the cat's visual cortex. *The Journal of physiology*, 160(1):106, 1962.

[49] Md Zahangir Alom, Tarek M Taha, Christopher Yakopcic, Stefan Westberg, Paheding Sidike, Mst Shamima Nasrin, Brian C Van Esesn, Abdul A S Awwal, and Vijayan K Asari. The history began from alexnet: A comprehensive survey on deep learning approaches. *arXiv preprint arXiv:1803.01164*, 2018.

[50] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.

[51] Shibani Santurkar, Dimitris Tsipras, Andrew Ilyas, and Aleksander Madry. How does batch normalization help optimization? In *Advances in Neural Information Processing Systems*, pages 2483–2493, 2018.

[52] Nils Bjorck, Carla P Gomes, Bart Selman, and Kilian Q Weinberger. Understanding batch normalization. In *Advances in Neural Information Processing Systems*, pages 7694–7705, 2018.

[53] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958, 2014.

[54] Pierre Baldi and Peter J Sadowski. Understanding dropout. In *Advances in neural information processing systems*, pages 2814–2822, 2013.

[55] Zhicun Xu, Peter Smit, Mikko Kurimo, et al. The aalto system based on fine-tuned audioset features for dcase 2018 task2——general purpose audio tagging. In *Scenes and Events 2018 Workshop (DCASE2018)*, page 24, 2018.

[56] Kevin Wilkinghoff. General-purpose audio tagging by ensembling convolutional neural networks based on multiple features. In *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2018 Workshop (DCASE2018)*, pages 44–48, 2018.

[57] Zhao Ren, Kun Qian, Yebin Wang, Zixing Zhang, Vedhas Pandit, Alice Baird, and Bjorn Schuller. Deep scalogram representations for acoustic scene classification. *IEEE/CAA Journal of Automatica Sinica*, 5(3):662–669, 2018.

[58] Truc Nguyen and Franz Pernkopf. Acoustic scene classification using a convolutional neural network ensemble and nearest neighbor filters. In *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2018 Workshop (DCASE2018)*, pages 34–38, 2018.

[59] Qiuqiang Kong, Yin Cao, Turab Iqbal, Yong Xu, Wenwu Wang, and Mark D Plumbley. Cross-task learning for audio tagging, sound event detection and spatial localization: Dcase 2019 baseline systems. *arXiv preprint arXiv:1904.03476*, 2019.

[60] Wei Dai, Chia Dai, Shuhui Qu, Juncheng Li, and Samarjit Das. Very deep convolutional neural networks for raw waveforms. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 421–425. IEEE, 2017.

[61] Matthias Dorfer, Bernhard Lehner, Hamid Eghbal-zadeh, Heindl Christop, Paischer Fabian, and Widmer Gerhard. Acoustic scene classification with fully convolutional neural networks and i-vectors. *Proceedings of the Detection and Classification of Acoustic Scenes and Events*, 2018.

[62] Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10):1345–1359, 2009.

[63] Iacopo Masi, Yue Wu, Tal Hassner, and Prem Natarajan. Deep face recognition: A survey. In *2018 31st SIBGRAPI conference on graphics, patterns and images (SIBGRAPI)*, pages 471–478. IEEE, 2018.

[64] Kasthurirangan Gopalakrishnan, Siddhartha K Khaitan, Alok Choudhary, and Ankit Agrawal. Deep convolutional neural networks with transfer learning for computer vision-based data-driven pavement distress detection. *Construction and Building Materials*, 157:322–330, 2017.

[65] Zhongling Huang, Zongxu Pan, and Bin Lei. Transfer learning with deep convolutional neural network for sar target classification with limited labeled data. *Remote Sensing*, 9(9):907, 2017.

[66] Ali Sharif Razavian, Hossein Azizpour, Josephine Sullivan, and Stefan Carlsson. Cnn features off-the-shelf: an astounding baseline for recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 806–813, 2014.

[67] Hoo-Chang Shin, Holger R Roth, Mingchen Gao, Le Lu, Ziyue Xu, Isabella Nogues, Jianhua Yao, Daniel Mollura, and Ronald M Summers. Deep convolutional neural networks for computer-aided detection: Cnn architectures, dataset characteristics and transfer learning. *IEEE transactions on medical imaging*, 35(5):1285–1298, 2016.

[68] Jason Cramer, Ho-Hsiang Wu, Justin Salamon, and Juan Pablo Bello. Look, listen, and learn more: Design choices for deep audio embeddings. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3852–3856. IEEE, 2019.

[69] Relja Arandjelovic and Andrew Zisserman. Look, listen and learn. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 609–617, 2017.

[70] Yusuf Aytar, Carl Vondrick, and Antonio Torralba. Soundnet: Learning sound representations from unlabeled video. In *Advances in neural information processing systems*, pages 892–900, 2016.

[71] Arshdeep Singh, Padmanabhan Rajan, and Arnav Bhavsar. Deep multi-view features from raw audio for acoustic scene classification. 2019.

[72] Karol J Piczak. Environmental sound classification with convolutional neural networks. In *2015 IEEE 25th International Workshop on Machine Learning for Signal Processing (MLSP)*, pages 1–6. IEEE, 2015.

[73] Roneel V Sharan and Tom J Moir. Robust acoustic event classification using deep neural networks. *Information Sciences*, 396:24–32, 2017.

[74] Daniele Barchiesi, Dimitrios Giannoulis, Dan Stowell, and Mark D Plumbley. Acoustic scene classification: Classifying environments from the sounds they produce. *IEEE Signal Processing Magazine*, 32(3):16–34, 2015.

[75] Hyungui Lim, Jeongsoo Park, and Yoonchang Han. Rare sound event detection using 1d convolutional recurrent neural networks. In *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2017 Workshop*, pages 80–84, 2017.

[76] Yibin Zhang and Jie Zhou. Audio segmentation based on multi-scale audio classification. In *2004 IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 4, pages iv–iv. IEEE, 2004.

[77] Lie Lu, Hong-Jiang Zhang, and Stan Z Li. Content-based audio classification and segmentation by using support vector machines. *Multimedia systems*, 8(6):482–492, 2003.

[78] Lie Lu, Hao Jiang, and HongJiang Zhang. A robust audio classification and segmentation method. In *Proceedings of the ninth ACM international conference on Multimedia*, pages 203–211, 2001.

[79] SR Mohanty, AK Pradhan, and A Routray. A cumulative sum-based fault detector for power system relaying application. *IEEE transactions on power delivery*, 23(1):79–86, 2007.

[80] Maximo Cobos, Juan J Perez-Solano, Santiago Felici-Castell, Jaume Segura, and Juan M Navarro. Cumulative-sum-based localization of sound events in low-cost wireless acoustic sensor networks. *IEEE/ACM transactions on audio, speech, and language processing*, 22(12):1792–1802, 2014.

[81] Annamaria Mesaros, Toni Heittola, and Tuomas Virtanen. A multi-device dataset for urban acoustic scene classification. In *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2018 Workshop (DCASE2018)*, pages 9–13, November 2018.

[82] Justin Salamon, Christopher Jacoby, and Juan Pablo Bello. A dataset and taxonomy for urban sound research. In *Proceedings of the 22nd ACM international conference on Multimedia*, pages 1041–1044, 2014.

[83] Archontis Politis, Sharath Adavanne, and Tuomas Virtanen. A dataset of reverberant spatial sound scenes with moving sources for sound event localization and detection. *arXiv e-prints: 2006.01919*, 2020.

[84] François Chollet. Xception: Deep learning with depthwise separable convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1251–1258, 2017.

[85] Karol J Piczak. Esc: Dataset for environmental sound classification. In *Proceedings of the 23rd ACM international conference on Multimedia*, pages 1015–1018, 2015.

[86] Shawn Hershey, Sourish Chaudhuri, Daniel PW Ellis, Jort F Gemmeke, Aren Jansen, R Channing Moore, Manoj Plakal, Devin Platt, Rif A Saurous, Bryan Seybold, et al. Cnn architectures for large-scale audio classification. In *2017 ieee international conference on acoustics, speech and signal processing (icassp)*, pages 131–135. IEEE, 2017.

[87] Muhammad Huzaifah. Comparison of time-frequency representations for environmental sound classification using convolutional neural networks. *arXiv preprint arXiv:1706.07156*, 2017.

[88] Zhouyu Fu, Guojun Lu, Kai Ming Ting, and Dengsheng Zhang. A survey of audio-based music classification and annotation. *IEEE transactions on multimedia*, 13(2):303–319, 2010.

[89] Sunit Sivasankaran and KMM Prabhu. Robust features for environmental sound classification. In *2013 IEEE International Conference on Electronics, Computing and Communication Technologies*, pages 1–6. IEEE, 2013.

[90] Justin Salamon and Juan Pablo Bello. Unsupervised feature learning for urban sound classification. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 171–175. IEEE, 2015.

[91] Linus Lexfors and Malte Johansson. Audio representation for environmental sound classification using convolutional neural networks. *Master's Theses in Mathematical Sciences*, 2018.

[92] Karol J Piczak. The details that matter: Frequency resolution of spectrograms in acoustic scene classification. *Detection and Classification of Acoustic Scenes and Events*, pages 103–107, 2017.

[93] Jongpil Lee, Jiyoung Park, Keunhyoung Luke Kim, and Juhan Nam. Sample-level deep convolutional neural networks for music auto-tagging using raw waveforms. *arXiv preprint arXiv:1703.01789*, 2017.

[94] Shuhui Qu, Juncheng Li, Wei Dai, and Samarjit Das. Understanding audio pattern using convolutional neural network from raw waveforms. *arXiv preprint arXiv:1611.09524*, 2016.

[95] Thomas Grill and Jan Schlüter. Two convolutional neural networks for bird detection in audio signals. In *2017 25th European Signal Processing Conference (EUSIPCO)*, pages 1764–1768. IEEE, 2017.

[96] Juan Manuel Vera-Diaz, Daniel Pizarro, and Javier Macias-Guarasa. Towards end-to-end acoustic localization using deep learning: From audio signals to source position coordinates. *Sensors*, 18(10):3418, 2018.

[97] D. Giannoulis, D. Stowell, E. Benetos, M. Rossignol, M. Lagrange, and M. D. Plumbley. A database and challenge for acoustic scene classification and event detection. In *21st European Signal Processing Conference (EUSIPCO 2013)*, pages 1–5, Sep. 2013.

[98] D. Stowell, D. Giannoulis, E. Benetos, M. Lagrange, and M. D. Plumbley. Detection and classification of acoustic scenes and events. *IEEE Transactions on Multimedia*, 17(10):1733–1746, Oct 2015.

[99] Annamaria Mesaros, Toni Heittola, and Tuomas Virtanen. TUT database for acoustic scene classification and sound event detection. In *24th European Signal Processing Conference 2016 (EUSIPCO 2016)*, Budapest, Hungary, 2016.

[100] A. Mesaros, T. Heittola, E. Benetos, P. Foster, M. Lagrange, T. Virtanen, and M. D. Plumbley. Detection and classification of acoustic scenes and events: Outcome of the DCASE 2016 challenge. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 26(2):379–393, Feb 2018.

[101] G. Lafay, E. Benetos, and M. Lagrange. Sound event detection in synthetic audio: Analysis of the DCASE 2016 task results. In *2017 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, pages 11–15, Oct 2017.

[102] A. Mesaros, T. Heittola, A. Diment, B. Elizalde, A. Shah, E. Vincent, B. Raj, and T. Virtanen. DCASE 2017 challenge setup: Tasks, datasets and baseline system. In *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2017 Workshop (DCASE2017)*, pages 85–92, November 2017.

[103] Annamaria Mesaros, Toni Heittola, and Tuomas Virtanen. Acoustic scene classification: An overview of DCASE 2017 challenge entries. In *2018 16th International Workshop on Acoustic Signal Enhancement (IWAENC)*, pages 411–415, September 2018.

[104] A. Mesaros, A. Diment, B. Elizalde, T. Heittola, E. Vincent, B. Raj, and T. Virtanen. Sound event detection in the DCASE 2017 challenge. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 2019. In press.

[105] Eduardo Fonseca, Manoj Plakal, Frederic Font, Daniel P. W. Ellis, Xavier Favory, Jordi Pons, and Xavier Serra. General-purpose tagging of freesound audio with audioset labels: Task description, dataset, and baseline. In *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2018 Workshop (DCASE2018)*, pages 69–73, November 2018.

[106] D. Stowell, Y. Stylianou, M. Wood, H. Pamuła, and H. Glotin. Automatic acoustic detection of birds through deep learning: the first bird audio detection challenge. *Methods in Ecology and Evolution*, 2018.

[107] Romain Serizel, Nicolas Turpault, Hamid Eghbal-Zadeh, and Ankit Parag Shah. Large-scale weakly labeled semi-supervised sound event detection in domestic environments. In *Proceedings of the Detection*

and Classification of Acoustic Scenes and Events 2018 Workshop (DCASE2018), pages 19–23, November 2018.

[108] Gert Dekkers, Steven Lauwereins, Bart Thoen, Mulu Weldegebreal Adhana, Henk Brouckxon, Toon van Waterschoot, Bart Vanrumste, Marian Verhelst, and Peter Karsmakers. The SINS database for detection of daily activities in a home environment using an acoustic sensor network. In *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2017 Workshop (DCASE2017)*, pages 32–36, November 2017.

[109] Gert Dekkers, Lode Vuegen, Toon van Waterschoot, Bart Vanrumste, and Peter Karsmakers. DCASE 2018 Challenge - Task 5: Monitoring of domestic activities based on multi-channel acoustics. Technical report, KU Leuven, 2018.

[110] Eduardo Fonseca, Manoj Plakal, Frederic Font, Daniel P. W. Ellis, and Xavier Serra. Audio tagging with noisy labels and minimal supervision. In *Submitted to DCASE2019 Workshop*, NY, USA, 2019.

[111] Sharath Adavanne, Archontis Politis, Joonas Nikunen, and Tuomas Virtanen. Sound event localization and detection of overlapping sources using convolutional recurrent neural networks. *IEEE Journal of Selected Topics in Signal Processing*, pages 1–1, 2018.

[112] Sharath Adavanne, Archontis Politis, and Tuomas Virtanen. A multi-room reverberant dataset for sound event localization and uetection. In *Submitted to Detection and Classification of Acoustic Scenes and Events 2019 Workshop (DCASE2019)*, 2019.

[113] Nicolas Turpault, Romain Serizel, Ankit Parag Shah, and Justin Salamon. Sound event detection in domestic environments with weakly labeled data and soundscape synthesis. In *Workshop on Detection and Classification of Acoustic Scenes and Events*, New York City, United States, October 2019.

[114] Juan P. Bello, Claudio Silva, Oded Nov, R. Luke Dubois, Anish Arora, Justin Salamon, Charles Mydlarz, and Harish Doraiswamy. Sonyc: A system for monitoring, analyzing, and mitigating urban noise pollution. *Communications of the ACM*, 62(2):68–77, Feb 2019.

[115] Yuma Koizumi, Yohei Kawaguchi, Keisuke Imoto, Toshiki Nakamura, Yuki Nikaido, Ryo Tanabe, Harsh Purohit, Kaori Suefusa, Takashi Endo, Masahiro Yasuda, and Noboru Harada. Description and discussion on DCASE2020 challenge task2: Unsupervised anomalous sound detection for machine condition monitoring. In *arXiv e-prints: 2006.05822*, pages 1–4, June 2020.

[116] Yuma Koizumi, Shoichiro Saito, Hisashi Uematsu, Noboru Harada, and Keisuke Imoto. ToyADMOS: A dataset of miniature-machine operating sounds for anomalous sound detection. In *Proceedings of IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, pages 308–312, November 2019.

[117] Harsh Purohit, Ryo Tanabe, Takeshi Ichige, Takashi Endo, Yuki Nikaido, Kaori Suefusa, and Yohei Kawaguchi. MIMII Dataset: Sound dataset for malfunctioning industrial machine investigation and inspection. In *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2019 Workshop (DCASE2019)*, pages 209–213, November 2019.

[118] Ilya Kavalerov, Scott Wisdom, Hakan Erdogan, Brian Patton, Kevin Wilson, Jonathan Le Roux, and John R. Hershey. Universal sound separation. In *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, pages 175–179, October 2019.

[119] Scott Wisdom, John R Hershey, Kevin Wilson, Jeremy Thorpe, Michael Chinen, Brian Patton, and Rif A Saurous. Differentiable consistency constraints for improved deep speech enhancement. In *ICASSP*

2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pages 900–904. IEEE, 2019.

[120] Mark Cartwright, Ana Elisa Mendez Mendez, Jason Cramer, Vincent Lostanlen, Graham Dove, Ho-Hsiang Wu, Justin Salamon, Oded Nov, and Juan Bello. SONYC urban sound tagging (SONYC-UST): A multilabel dataset from an urban acoustic sensor network. In *Proceedings of the Workshop on Detection and Classification of Acoustic Scenes and Events (DCASE)*, pages 35–39, October 2019.

[121] Konstantinos Drossos, Sharath Adavanne, and Tuomas Virtanen. Automated audio captioning with recurrent neural networks. In *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, New Paltz, New York, U.S.A., Oct. 2017.

[122] Samuel Lipping, Konstantinos Drossos, and Tuoams Virtanen. Crowdsourcing a dataset of audio captions. In *Proceedings of the Detection and Classification of Acoustic Scenes and Events Workshop (DCASE)*, Nov. 2019.

[123] Konstantinos Drossos, Samuel Lipping, and Tuomas Virtanen. Clotho: An audio captioning dataset. In *45th IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Barcelona, Spain, May 2020.

[124] He Zhang and Vishal M Patel. Sparse representation-based open set recognition. *IEEE transactions on pattern analysis and machine intelligence*, 39(8):1690–1696, 2016.

[125] Yang Yang, Chunping Hou, Yue Lang, Dai Guan, Danyang Huang, and Jinchen Xu. Open-set human activity recognition based on micro-doppler signatures. *Pattern Recognition*, 85:60–69, 2019.

[126] Chuanxing Geng and Songcan Chen. Collective decision for open set recognition. *IEEE Transactions on Knowledge and Data Engineering*, 2020.

[127] Douglas O Cardoso, João Gama, and Felipe MG França. Weightless neural networks for open set recognition. *Machine Learning*, 106(9-10):1547–1567, 2017.

[128] Chuanxing Geng, Sheng-jun Huang, and Songcan Chen. Recent advances in open set recognition: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020.

[129] Guillaume Bouchard and Bill Triggs. The tradeoff between generative and discriminative classifiers. In *16th IASC International Symposium on Computational Statistics (COMPSTAT '04))*, pages 721–728, 2004.

[130] Julia A Lasserre, Christopher M Bishop, and Thomas P Minka. Principled hybrids of generative and discriminative models. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 1, pages 87–94. IEEE, 2006.

[131] Hakan Cevikalp, Bill Triggs, and Vojtech Franc. Face and landmark detection by using cascade of classifiers. In *2013 10th IEEE International Conference and Workshops on Automatic Face and Gesture Recognition (FG)*, pages 1–7. IEEE, 2013.

[132] Hakan Cevikalp. Best fitting hyperplanes for classification. *IEEE transactions on pattern analysis and machine intelligence*, 39(6):1076–1088, 2016.

[133] Samuel Kotz and Saralees Nadarajah. *Extreme value distributions: theory and applications*. World Scientific, 2000.

[134] Lalit P Jain, Walter J Scheirer, and Terrance E Boult. Multi-class open set recognition using probability of inclusion. In *European Conference on Computer Vision*, pages 393–409. Springer, 2014.

[135] Matthew D Scherreik and Brian D Rigling. Open set recognition for automatic target classification with rejection. *IEEE Transactions on Aerospace and Electronic Systems*, 52(2):632–642, 2016.

[136] John Wright, Yi Ma, Julien Mairal, Guillermo Sapiro, Thomas S Huang, and Shuicheng Yan. Sparse representation for computer vision and pattern recognition. *Proceedings of the IEEE*, 98(6):1031–1044, 2010.

[137] Ron Rubinstein, Alfred M Bruckstein, and Michael Elad. Dictionaries for sparse representation modeling. *Proceedings of the IEEE*, 98(6):1045–1057, 2010.

[138] John Wright, Allen Y Yang, Arvind Ganesh, S Shankar Sastry, and Yi Ma. Robust face recognition via sparse representation. *IEEE transactions on pattern analysis and machine intelligence*, 31(2):210–227, 2008.

[139] Abhijit Bendale and Terrance Boult. Towards open world recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1893–1902, 2015.

[140] Pedro R Mendes Júnior, Roberto M De Souza, Rafael de O Werneck, Bernardo V Stein, Daniel V Pazinato, Waldir R de Almeida, Otávio AB Penatti, Ricardo da S Torres, and Anderson Rocha. Nearest neighbors distance ratio open-set classifier. *Machine Learning*, 106(3):359–386, 2017.

[141] Ethan M Rudd, Lalit P Jain, Walter J Scheirer, and Terrance E Boult. The extreme value machine. *IEEE transactions on pattern analysis and machine intelligence*, 40(3):762–768, 2017.

[142] Abhijit Bendale and Terrance E Boult. Towards open set deep networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1563–1572, 2016.

[143] Lei Shu, Hu Xu, and Bing Liu. Doc: Deep open classification of text documents. *arXiv preprint arXiv:1709.08716*, 2017.

[144] Navid Kardan and Kenneth O Stanley. Mitigating fooling with competitive overcomplete output layer neural networks. In *2017 International Joint Conference on Neural Networks (IJCNN)*, pages 518–525. IEEE, 2017.

[145] Akshay Raj Dhamija, Manuel Günther, and Terrance Boult. Reducing network agnostophobia. In *Advances in Neural Information Processing Systems*, pages 9157–9168, 2018.

[146] Ryota Yoshihashi, Wen Shao, Rei Kawakami, Shaodi You, Makoto Iida, and Takeshi Naemura. Classification-reconstruction learning for open-set recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4016–4025, 2019.

[147] Poojan Oza and Vishal M Patel. C2ae: Class conditioned auto-encoder for open-set recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2307–2316, 2019.

[148] Sagie Benaim and Lior Wolf. One-shot unsupervised cross domain translation. In *Advances in Neural Information Processing Systems*, pages 2104–2114, 2018.

[149] Brenden M Lake, Ruslan Salakhutdinov, and Joshua B Tenenbaum. Human-level concept learning through probabilistic program induction. *Science*, 350(6266):1332–1338, 2015.

[150] Hang Qi, Matthew Brown, and David G Lowe. Low-shot learning with imprinted weights. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5822–5830, 2018.

[151] Pranav Shyam, Shubham Gupta, and Ambedkar Dukkipati. Attentive recurrent comparators. *arXiv preprint arXiv:1703.00767*, 2017.

[152] Adam Santoro, Sergey Bartunov, Matthew Botvinick, Daan Wierstra, and Timothy Lillicrap. Meta-learning with memory-augmented neural networks. In *International conference on machine learning*, pages 1842–1850, 2016.

[153] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*, 2017.

[154] Mark D McDonnell and Wei Gao. Acoustic scene classification using deep residual networks with late fusion of separated high and low frequency paths. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 141–145. IEEE, 2020.

[155] L Rafael Aguiar, MG Yandre Costa, and N Carlos Silla. Exploring data augmentation to improve music genre classification with convnets. In *2018 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2018.

[156] Daniel S Park, William Chan, Yu Zhang, Chung-Cheng Chiu, Barret Zoph, Ekin D Cubuk, and Quoc V Le. Specaugment: A simple data augmentation method for automatic speech recognition. *arXiv preprint arXiv:1904.08779*, 2019.

[157] Yu Zhang and Qiang Yang. A survey on multi-task learning. *arXiv preprint arXiv:1707.08114*, 2017.

[158] Jason Weston, Sumit Chopra, and Antoine Bordes. Memory networks. *arXiv preprint arXiv:1410.3916*, 2014.

[159] Yutong Zheng, Dipan K Pal, and Marios Savvides. Ring loss: Convex feature normalization for face recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5089–5097, 2018.

[160] Jane Bromley, Isabelle Guyon, Yann LeCun, Eduard Säckinger, and Roopak Shah. Signature verification using a" siamese" time delay neural network. In *Advances in neural information processing systems*, pages 737–744, 1994.

[161] Raia Hadsell, Sumit Chopra, and Yann LeCun. Dimensionality reduction by learning an invariant mapping. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 2, pages 1735–1742. IEEE, 2006.

[162] Carlos N Silla Jr, Celso AA Kaestner, and Alessandro L Koerich. Automatic music genre classification using ensemble of classifiers. In *2007 IEEE International Conference on Systems, Man and Cybernetics*, pages 1687–1692. IEEE, 2007.

[163] Yuma Sakashita and Masaki Aono. Acoustic scene classification by ensemble of spectrograms based on adaptive temporal divisions. *IEEE AASP Challenge on DCASE 2018 technical reports*, 2018.

[164] Rohith Mars, Pranay Pratik, Srikanth Nagisetty, and Chongsoon Lim. Acoustic scene classification from binaural signals using convolutional neural networks. In *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2019 Workshop (DCASE2019)*, pages 149–153, New York University, NY, USA, October 2019.

[165] Jaehun Kim and Kyogu Lee. Empirical study on ensemble method of deep neural networks for acoustic scene classification. *Proc. of IEEE AASP Challenge on Detection and Classification of Acoustic Scenes and Events (DCASE)*, 2016.

[166] Jonathan Huang, Hong Lu, Paulo Lopez Meyer, Hector Cordourier, and Juan Del Hoyo Ontiveros. Acoustic scene classification using deep learning-based ensemble averaging. In *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2019 Workshop (DCASE2019)*, pages 94–98, New York University, NY, USA, October 2019.

[167] Wootaek Lim, Sangwon Suh, and Youngho Jeong. Weakly labeled semi-supervised sound event detection using crnn with inception module. In *Proc. Detection and Classification of Acoustic Scenes and Events Workshop (DCASE)*, pages 74–77, 2018.

[168] Sergi Perez-Castanos, Javier Naranjo-Alcazar, Pedro Zuccarello, Maximo Cobos, and Frances J Ferri. Cnn depth analysis with different channel inputs for acoustic scene classification. *arXiv preprint arXiv:1906.04591*, 2019.

[169] Hyeji Seo, Jihwan Park, and Yongjin Park. Acoustic scene classification using various pre-processed features and convolutional neural networks. In *Proceedings of the Detection and Classification of Acoustic Scenes and Events Workshop (DCASE), New York, NY, USA*, pages 25–26, 2019.

[170] Octave Mariotti, Matthieu Cord, and Olivier Schwander. Exploring deep vision models for acoustic scene classification. *Proc. DCASE*, pages 103–107, 2018.

[171] Stanley Smith Stevens, John Volkmann, and Edwin B Newman. A scale for the measurement of the psychological magnitude pitch. *The Journal of the Acoustical Society of America*, 8(3):185–190, 1937.

[172] Derry Fitzgerald. Harmonic/percussive separation using median filtering. In *Proc. of DAFX*, volume 10, 2010.

[173] Jonathan Driedger, Meinard Müller, and Sascha Disch. Extending harmonic-percussive separation of audio signals. In *ISMIR*, pages 611–616, 2014.

[174] Logan Ford, Hao Tang, François Grondin, and James R Glass. A deep residual network for large-scale acoustic scene analysis. In *INTERSPEECH*, pages 2568–2572, 2019.

[175] Andreas Veit, Michael J Wilber, and Serge Belongie. Residual networks behave like ensembles of relatively shallow networks. In *Advances in neural information processing systems*, pages 550–558, 2016.

[176] Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7132–7141, 2018.

[177] Abhijit Guha Roy, Nassir Navab, and Christian Wachinger. Concurrent spatial and channel 'squeeze & excitation'in fully convolutional networks. In *International conference on medical image computing and computer-assisted intervention*, pages 421–429. Springer, 2018.

[178] Liping Yang, Xinxing Chen, Lianjie Tao, and Xiaohua Gu. Multi-scale fusion and channel weighted cnn for acoustic scene classification. In *Proceedings of the 2019 2nd International Conference on Signal Processing and Machine Learning*, pages 41–45, 2019.

[179] Jongpil Lee, Taejun Kim, Jiyoung Park, and Juhan Nam. Raw waveform-based audio classification using sample-level cnn architectures. *arXiv preprint arXiv:1712.00866*, 2017.

[180] Osamu Akiyama and Junya Sato. Multitask learning and semisupervised learning with noisy data for audio tagging. *DCASE2019 Challenge*, 2019.

[181] Maximilian Schmitt and Björn Schuller. End-to-end audio classification with small datasets–making it work. In *2019 27th European Signal Processing Conference (EUSIPCO)*, pages 1–5. IEEE, 2019.

[182] Jonathan J Huang and Juan Jose Alvarado Leanos. Aclnet: efficient end-to-end audio classification cnn. *arXiv preprint arXiv:1811.06669*, 2018.

[183] Jordi Pons, Oriol Nieto, Matthew Prockup, Erik Schmidt, Andreas Ehmann, and Xavier Serra. End-to-end learning for music audio tagging at scale. *arXiv preprint arXiv:1711.02520*, 2017.

[184] Sander Dieleman and Benjamin Schrauwen. End-to-end learning for music audio. In *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6964–6968. IEEE, 2014.

[185] Yuan Gong and Christian Poellabauer. How do deep convolutional neural networks learn from raw audio waveforms? 2018.

[186] Jiaxu Chen, Jing Hao, Kai Chen, Di Xie, Shicai Yang, and Shiliang Pu. An end-to-end audio classification system based on raw waveforms and mix-training strategy. *arXiv preprint arXiv:1911.09349*, 2019.

[187] Javier Naranjo-Alcazar, Sergi Perez-Castanos, Pedro Zuccarrello, and Maximo Cobos. An open-set recognition and few-shot learning dataset for audio event classification in domestic environments. *arXiv preprint arXiv:2002.11561*, 2020.

[188] Annamaria Mesaros, Toni Heittola, and Tuomas Virtanen. Acoustic scene classification in dcase 2019 challenge: Closed and open set classification and data mismatch setups. In *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2019 Workshop (DCASE2019)*, pages 164–168, New York University, NY, USA, October 2019.

[189] Paul Primus, Hamid Eghbal-zadeh, David Eitelsebner, Khaled Koutini, Andreas Arzt, and Gerhard Widmer. Exploiting parallel audio recordings to enforce device invariance in cnn-based acoustic scene classification. In *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2019 Workshop (DCASE2019)*, pages 204–208, New York University, NY, USA, October 2019.

[190] Truc Nguyen and Franz Pernkopf. Acoustic scene classification with mismatched recording devices using mixture of experts layer. In *2019 IEEE International Conference on Multimedia and Expo (ICME)*, pages 1666–1671. IEEE, 2019.

[191] Truc Nguyen and Franz Pernkopf. Acoustic scene classification with mismatched devices using cliquenets and mixup data augmentation. In *Interspeech*, pages 2330–2334, 2019.

[192] Seongkyu Mun and Suwon Shon. Domain mismatch robust acoustic scene classification using channel information conversion. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 845–849. IEEE, 2019.

[193] Irene Martin-Morato, Maximo Cobos, and Francesc J Ferri. Adaptive mid-term representations for robust audio event classification. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 26(12):2381–2392, 2018.

[194] Irene Martín-Morató, Maximo Cobos, and Francesc J Ferri. Adaptive distance-based pooling in convolutional neural networks for audio event classification. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 28:1925–1935, 2020.

[195] Irwan Bello, Barret Zoph, Ashish Vaswani, Jonathon Shlens, and Quoc V Le. Attention augmented convolutional networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3286–3295, 2019.

[196] Max Jaderberg, Karen Simonyan, Andrew Zisserman, et al. Spatial transformer networks. In *Advances in neural information processing systems*, pages 2017–2025, 2015.

[197] Nicolas Turpault, Romain Serizel, Ankit Parag Shah, and Justin Salamon. Sound event detection in domestic environments with weakly labeled data and soundscape synthesis. working paper or preprint, June 2019.

[198] Siddharth Sigtia, Adam M Stark, Sacha Krstulović, and Mark D Plumbley. Automatic environmental sound recognition: Performance versus computational cost. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 24(11):2096–2107, 2016.

[199] Allen Newell and Herbert Simon. The logic theory machine–a complex information processing system. *IRE Transactions on information theory*, 2(3):61–79, 1956.

[200] Xavier Glorot, Antoine Bordes, and Yoshua Bengio. Deep sparse rectifier neural networks. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 315–323, 2011.

[201] Christopher M Bishop. *Pattern recognition and machine learning*. springer, 2006.

[202] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[203] Michele Valenti, Aleksandr Diment, Giambattista Parascandolo, Stefano Squartini, and Tuomas Virtanen. Dcase 2016 acoustic scene classification using convolutional neural networks. In *Proc. Workshop Detection Classif. Acoust. Scenes Events*, pages 95–99, 2016.

[204] Ronan Collobert and Jason Weston. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th international conference on Machine learning*, pages 160–167, 2008.

[205] Aaron Van den Oord, Sander Dieleman, and Benjamin Schrauwen. Deep content-based music recommendation. In *Advances in neural information processing systems*, pages 2643–2651, 2013.

[206] Avraam Tsantekidis, Nikolaos Passalis, Anastasios Tefas, Juho Kanniainen, Moncef Gabbouj, and Alexandros Iosifidis. Forecasting stock prices from the limit order book using convolutional neural networks. In *2017 IEEE 19th Conference on Business Informatics (CBI)*, volume 1, pages 7–12. IEEE, 2017.

[207] Bernhard Lehner, Hamid Eghbal-Zadeh, Matthias Dorfer, Filip Korzeniowski, Khaled Koutini, and Gerhard Widmer. Classifying short acoustic scenes with i-vectors and cnns: Challenges and optimisations for the 2017 dcase asc task. *DCASE2017 Challenge*, 2017.

[208] Thomas Lidy and Alexander Schindler. Cqt-based convolutional neural networks for audio scene classification. In *Proceedings of the detection and classification of acoustic scenes and events 2016 workshop (DCASE2016)*, volume 90, pages 1032–1048, 2016.

[209] Hangting Chen, Pengyuan Zhang, and Yonghong Yan. An audio scene classification framework with embedded filters and a dct-based temporal module. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 835–839. IEEE, 2019.

[210] Zhitong Li, Yuanbo Hou, Xiang Xie, Shengchen Li, Liqiang Zhang, Shixuan Du, and Wei Liu. Multilevel attention model with deep scattering spectrum for acoustic scene classification. In *2019 IEEE International Conference on Multimedia & Expo Workshops (ICMEW)*, pages 396–401. IEEE, 2019.

[211] Sangwook Park, Seongkyu Mun, Younglo Lee, and Hanseok Ko. Acoustic scene classification based on convolutional neural network using double image features. In *Proc. of the Detection and Classification of Acoustic Scenes and Events 2017 Workshop (DCASE2017)*, pages 98–102, 2017.

[212] Zhejian Chi, Ying Li, and Cheng Chen. Deep convolutional neural network combined with concatenated spectrogram for environmental sound classification. In *2019 IEEE 7th International Conference on Computer Science and Network Technology (ICCSNT)*, pages 251–254. IEEE, 2019.

[213] Zheng Weiping, Yi Jiantao, Xing Xiaotao, Liu Xiangtao, and Peng Shaohu. Acoustic scene classification using deep convolutional neural network and multiple spectrograms fusion. In *Detection and Classification of Acoustic Scenes and Events 2017 Workshop (DCASE2017)*, 2017.

[214] Jivitesh Sharma, Ole-Christoffer Granmo, and Morten Goodwin. Environment sound classification using multiple feature channels and attention based deep convolutional neural network. *arXiv*, pages arXiv–1908, 2019.

[215] PLM Johannesma. The pre-response stimulus ensemble of neurons in the cochlear nucleus. In *Symposium on Hearing Theory, 1972*. IPO, 1972.

[216] Venkata Neelima Parinam, Chandra Sekhar Vootkuri, and Stephen A Zahorian. Comparison of spectral analysis methods for automatic speech recognition. In *INTERSPEECH*, pages 3356–3360, 2013.

[217] Yong Xu, Qiuqiang Kong, Wenwu Wang, and Mark D Plumbley. Large-scale weakly supervised audio classification using gated convolutional neural network. In *2018 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 121–125. IEEE, 2018.

[218] Turab Iqbal, Qiuqiang Kong, Mark D Plumbley, and Wenwu Wang. General-purpose audio tagging from noisy labels using convolutional neural networks. In *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2018 Workshop (DCASE2018*, pages 212–216. Tampere University of Technology, 2018.

[219] Yuanbo Hou, Qiuqiang Kong, Jun Wang, and Shengchen Li. Polyphonic audio tagging with sequentially labelled data using crnn with learnable gated linear units. In *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2018 Workshop (DCASE2018)*, pages 78–82, November 2018.

[220] Qingkai Wei, Yanfang Liu, and Xiaohui Ruan. A report on audio tagging with deeper CNN, 1D-ConvNet and 2D-ConvNet. In *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2018 Workshop (DCASE2018)*, pages 49–53, November 2018.

[221] Javier Naranjo-Alcazar, Sergi Perez-Castanos, Jose Ferrandis, Pedro Zuccarello, and Maximo Cobos. Sound event localization and detection using squeeze-excitation residual cnns. *arXiv preprint arXiv:2006.14436*, 2020.

[222] Zhichao Zhang, Shugong Xu, Shan Cao, and Shunqing Zhang. Deep convolutional neural network with mixup for environmental sound classification. In *Chinese Conference on Pattern Recognition and Computer Vision (PRCV)*, pages 356–367. Springer, 2018.

[223] Brian CJ Moore, Robert W Peters, and Brian R Glasberg. Auditory filter shapes at low center frequencies. *The Journal of the Acoustical Society of America*, 88(1):132–140, 1990.

[224] Hangting Chen, Zuozhen Liu, Zongming Liu, Pengyuan Zhang, and Yonghong Yan. Integrating the data augmentation scheme with various classifiers for acoustic scene modeling. *arXiv preprint arXiv:1907.06639*, 2019.

[225] Aggelina Chatziagapi, Georgios Paraskevopoulos, Dimitris Sgouropoulos, Georgios Pantazopoulos, Malvina Nikandrou, Theodoros Giannakopoulos, Athanasios Katsamanis, Alexandros Potamianos, and Shrikanth Narayanan. Data augmentation using gans for speech emotion recognition. In *INTERSPEECH*, pages 171–175, 2019.

[226] Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell. Caffe: Convolutional architecture for fast feature embedding. In *Proceedings of the 22nd ACM international conference on Multimedia*, pages 675–678, 2014.

[227] Jake Snell, Kevin Swersky, and Richard Zemel. Prototypical networks for few-shot learning. In *Advances in neural information processing systems*, pages 4077–4087, 2017.

[228] Yu-An Chung, Chao-Chung Wu, Chia-Hao Shen, Hung-Yi Lee, and Lin-Shan Lee. Audio word2vec: Unsupervised learning of audio segment representations using sequence-to-sequence autoencoder. *arXiv preprint arXiv:1603.00982*, 2016.

[229] Sergi Perez-Castanos, Javier Naranjo-Alcazar, Pedro Zuccarello, and Maximo Cobos. Listen carefully and tell: an audio captioning system based on residual learning and gammatone audio representation. *arXiv preprint arXiv:2006.15406*, 2020.

[230] Konstantinos Drossos, Samuel Lipping, and Tuomas Virtanen. Clotho: an audio captioning dataset. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 736–740. IEEE, 2020.

[231] Chris Dongjoo Kim, Byeongchang Kim, Hyunmin Lee, and Gunhee Kim. Audiocaps: Generating captions for audios in the wild. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 119–132, 2019.

[232] Marco Tagliasacchi, Beat Gfeller, Félix de Chaumont Quitry, and Dominik Roblek. Self-supervised audio representation learning for mobile devices. *arXiv preprint arXiv:1905.11796*, 2019.

[233] Félix de Chaumont Quitry, Marco Tagliasacchi, and Dominik Roblek. Learning audio representations via phase prediction. *arXiv*, pages arXiv–1910, 2019.

[234] Jort F Gemmeke, Daniel PW Ellis, Dylan Freedman, Aren Jansen, Wade Lawrence, R Channing Moore, Manoj Plakal, and Marvin Ritter. Audio set: An ontology and human-labeled dataset for audio events. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 776–780. IEEE, 2017.

[235] Yuma Koizumi, Shoichiro Saito, Hisashi Uematsu, Noboru Harada, and Keisuke Imoto. Toyadmos: A dataset of miniature-machine operating sounds for anomalous sound detection. In *2019 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, pages 313–317. IEEE, 2019.

[236] Harsh Purohit, Ryo Tanabe, Kenji Ichige, Takashi Endo, Yuki Nikaido, Kaori Suefusa, and Yohei Kawaguchi. Mimii dataset: Sound dataset for malfunctioning industrial machine investigation and inspection. *arXiv preprint arXiv:1909.09347*, 2019.

[237] Yong Xu, Qiang Huang, Wenwu Wang, Peter Foster, Siddharth Sigtia, Philip JB Jackson, and Mark D Plumbley. Unsupervised feature learning based on deep models for environmental audio tagging. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 25(6):1230–1241, 2017.

[238] Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th international conference on Machine learning*, pages 1096–1103, 2008.

[239] Taejun Kim, Jongpil Lee, and Juhan Nam. Comparison and analysis of samplecnn architectures for audio classification. *IEEE Journal of Selected Topics in Signal Processing*, 13(2):285–297, 2019.

[240] Taejun Kim, Jongpil Lee, and Juhan Nam. Sample-level cnn architectures for music auto-tagging using raw waveforms. In *2018 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 366–370. IEEE, 2018.

[241] Kaize Ding, Jianling Wang, Jundong Li, Kai Shu, Chenghao Liu, and Huan Liu. Graph prototypical networks for few-shot learning on attributed networks. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, pages 295–304, 2020.

[242] Dawei Zhou, Jingrui He, Hongxia Yang, and Wei Fan. Sparc: Self-paced network representation for few-shot rare category characterization. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 2807–2816, 2018.

Annexes

A. Acoustic Scene Classification With Squeeze-Excitation Residual Networks

Received May 22, 2020, accepted June 11, 2020, date of publication June 16, 2020, date of current version June 29, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3002761

Acoustic Scene Classification With Squeeze-Excitation Residual Networks

JAVIER NARANJO-ALCAZAR[1,2], (Graduate Student Member, IEEE), SERGI PEREZ-CASTANOS[1], PEDRO ZUCCARELLO[1], AND MAXIMO COBOS[2], (Senior Member, IEEE)

[1]Visualfy, 46181 Benisanó, Spain
[2]Computer Science Department, Universitat de Valencia, 46100 Burjassot, Spain

Corresponding author: Javier Naranjo-Alcazar (javier.naranjo@visualfy.com)

This work was supported by the European Union's Horizon 2020 Research and Innovation Program under Grant 779158. The work of Javier Naranjo-Alcazar was supported by the Spanish Ministry of Science, Innovation, and Universities, through the Torres Quevedo Fellowship, under Grant DIN2018-009982. The work of Pedro Zuccarello was supported by the Spanish Ministry of Science, Innovation, and Universities, through the Torres Quevedo Fellowship, under Grant PTQ-17-09106. The work of Maximo Cobos was supported in part by the European Regional Development Fund (ERDF), and in part by the Spanish Ministry of Science, Innovation, and Universities under Grant RTI2018-097045-B-C21.

ABSTRACT Acoustic scene classification (ASC) is a problem related to the field of machine listening whose objective is to classify/tag an audio clip in a predefined label describing a scene location (e. g. park, airport, etc.). Many state-of-the-art solutions to ASC incorporate data augmentation techniques and model ensembles. However, considerable improvements can also be achieved only by modifying the architecture of convolutional neural networks (CNNs). In this work we propose two novel squeeze-excitation blocks to improve the accuracy of a CNN-based ASC framework based on residual learning. The main idea of squeeze-excitation blocks is to learn spatial and channel-wise feature maps independently instead of jointly as standard CNNs do. This is usually achieved by combining some global grouping operators, linear operators and a final calibration between the input of the block and its learned relationships. The behavior of the block that implements such operators and, therefore, the entire neural network, can be modified depending on the input to the block, the established residual configurations and the selected non-linear activations. The analysis has been carried out using the TAU Urban Acoustic Scenes 2019 dataset presented in the 2019 edition of the Detection and Classification of Acoustic Scenes and Events (DCASE) challenge. All configurations discussed in this document exceed the performance of the baseline proposed by the DCASE organization by 13% percentage points. In turn, the novel configurations proposed in this paper outperform the residual configurations proposed in previous works.

INDEX TERMS Acoustic scene classification, deep learning, machine listening, pattern recognition, squeeze-excitation.

I. INTRODUCTION

The analysis of everyday ambient sounds can be very useful when developing intelligent systems in applications such as domestic assistants, surveillance systems or autonomous driving. Acoustic scene classification (ASC) is one of the most typical problems related to machine listening [1]–[4]. Machine listening is understood as the field of artificial intelligence that attempts to create intelligent algorithms capable of extracting meaningful information from audio data. Therefore, ASC can be defined as the area of machine listening that attempts to tag an audio clip in one of the predefined tags related to the description of a scene (for example, airport, park, subway, etc.).

The first approaches to the ASC problem were centered on the design of proper inputs to the classifier, this is, feature engineering [5]. Most research efforts tried to create meaningful representations of the audio data to later feed gaussian mixture models (GMMs), hidden Markov models (HMM) or support vector machines (SVMs) [6]. In this context, a wide range of input representations were proposed such as Mel-frequency cepstral coefficients (MFCCs) [7], [8],

The associate editor coordinating the review of this manuscript and approving it for publication was Shiqing Zhang.

Wavelets [8], constant-Q transform (CQT) or histograms of oriented gradientes (HOG) [9], among others.

With the years and the emergence of convolutional networks in the field of image and computer vision, CNNs have become a preferred option for the design of machine listening systems, usually fed with a 2D audio representation such as log-Mel spectrograms [1], [10]. These networks have shown very satisfactory results, especially when they are trained on large datasets. This is why data augmentation techniques are commonly applied, such as mixup strategies [11] or temporal cropping [12]. In addition, to improve the final accuracy, many studies use ensembles, combining the output from different classifiers to obtain a single more robust prediction. Unfortunately, the use of ensembles makes it more difficult to analyze the contribution to the classification performance of a new CNN architecture integrated within the proposed ensemble. To avoid such issue, this work considers isolated contributions of several CNN architectures implemented with different residual blocks based on squeeze-excitation, without any extra modifications during the training or inference phases.

CNNs are built with stacked convolutional layers. These layers learn its filter coefficients by capturing local spatial relationships (neighbourhood information) along the input channels and generate features maps (filtered inputs) by jointly encoding the spatial and channel information. In all application domains (image classification/segmentation, audio classification/tagging, etc.), the idea of encoding the spatial and the channel information independently has been less studied, despite having shown promising results [13], [14].

In order to provide insight about the behaviour of CNNs when analyzing spatial and channel information independently, several squeeze-excitation (SE) blocks have been presented in the image classification literature [13], [14]. In [14], a block that ''squeezes'' spatially and ''excites'' channel-wise with linear relationships was presented. The idea behind this block, denoted as *cSE* in this work, is to model the interdependencies between the channels of feature maps by exciting in a channel-wise manner. This type of block showed its effectiveness in image classification tasks, outperforming other state-of-the-art networks only by inserting it at a specific point of the network. Following this idea, two more blocks were presented in [13]. The first one, denoted as *sSE*, ''squeezes'' along the channels and ''excites'' spatially, whereas the last block, *scSE*, combines both strategies. The scSE block recalibrates the feature maps along spatial and channel dimensions independently (*cSE* and *sSE*) and then combines the information of both paths by adding their outputs. This last block showed the most promising results in image-related tasks. According to [13], this block forces the feature maps to be more informative, both spatially and channel-wise.

This work analyzes the performance of conventional SE blocks for addressing the ASC problem and proposes two novel block configurations in this context. The new configurations are intended to enhance the benefits of residual learning and feature map recalibration in a jointly fashion. This is achieved by a double short-cut connection that enforces residual learning both with and without recalibrated outputs. The use of SE techniques allows the network to extract more meaningful information during training, while residual learning facilitates the training procedure by mitigating vanishing gradient problems. The results show that, by using the proposed block configurations, results are considerably improved. Moreover, it is shown that all the residual SE configurations perform better than a classical convolutional residual block in the considered task.

The following of the paper is organized as follows. Section II presents the the background for the techniques used in this work in the context of ASC, namely Squeeze-Excitation and residual learning. Section III introduces the different SE blocks analyzed in this work and the baseline CNN architecture. Section IV describes the dataset used in the experiments, the audio pre-processing and the training procedure of the CNN. Section V discusses the experimental results, while Section VI concludes our work.

II. BACKGROUND

This section summarizes the technical background for this work and describes the ideas underlying SE blocks and residual networks.

A. RELATED WORK

Some previous works have shown that the use of SE modules can be a simple and effective approach to tackle audio classification problems. In [15], a multi-scale fusion and channel weighted CNN was proposed within an ASC context. The framework consists of two stages: a multi-scale feature fusion scheme that integrates a hierarchy of semantic-features extracted from a simplified Xception architecture, and a final SE-based channel weighting stage. However, such work considers only channel recalibration by using a *cSE*-like block at a final stage, without further integration of other SE-based calibration modules. In contrast, the configurations proposed in this work consider both spatial and channel-wise weighting within a residual learning framework jointly and at multiple depths within the network architecture.

Another work using SE techniques in the audio domain is [16], which presented a VGG-style CNN and compared its performance with an enhanced version including residual connections and SE modules. In contrast to the work presented in this paper, an end-to-end 1D architecture accepting raw audio inputs was proposed, with *cSE* channel-wise recalibration. The results over three different tasks (music auto-tagging, speech command recognition and acoustic event detection) confirmed the superiority of the enhanced network.

Finally, although some technical reports could not corroborate the improvements offered by SE modules over plain residual networks in audio-oriented tasks [17], few details were given, which motivates further the analysis carried out in this work.

B. SQUEEZE-EXCITATION BLOCKS

Squeeze-excitation (SE) blocks can be understood as modules for channel recalibration of feature maps [13]. Let' s assume an input feature map, $\mathbf{X} \in \mathbb{R}^{H\times W\times C'}$, that feeds any convolutional block, usually implemented by convolutional layers and non-linearities, and generates an output feature map $\mathbf{U} \in \mathbb{R}^{H\times W\times C}$. Here, $\mathbf{U}$ could also be expressed as $\mathbf{U} = [\mathbf{u}_1, \mathbf{u}_2, \dots, \mathbf{u}_C]$, being $\mathbf{u}_i \in \mathbb{R}^{H\times W}$ a channel output. Considering this notation, H and W represents the height and the width, while C' and C defines the number of input and output channels, respectively. The convolutional process function can be defined as $\mathbf{F}(\cdot)$, so that $\mathbf{F}(\mathbf{X}) = \mathbf{U}$. The output $\mathbf{U}$ is generated by combining the spatial and channel information of $\mathbf{X}$. The objective of SE blocks is to recalibrate $\mathbf{U}$ with $\mathbf{F}_{SE}(\cdot)$ to generate $\hat{\mathbf{U}}$, i.e. $\mathbf{F}_{SE}(\cdot) : \mathbf{U} \rightarrow \hat{U}$. This recalibrated feature map, $\hat{\mathbf{U}}$, can be stacked after every convolutional block and then used as input to the forthcoming pooling layers. This recalibration can be carried out with different types of block functions $\mathbf{F}_{SE}(\cdot)$, as it is next explained.

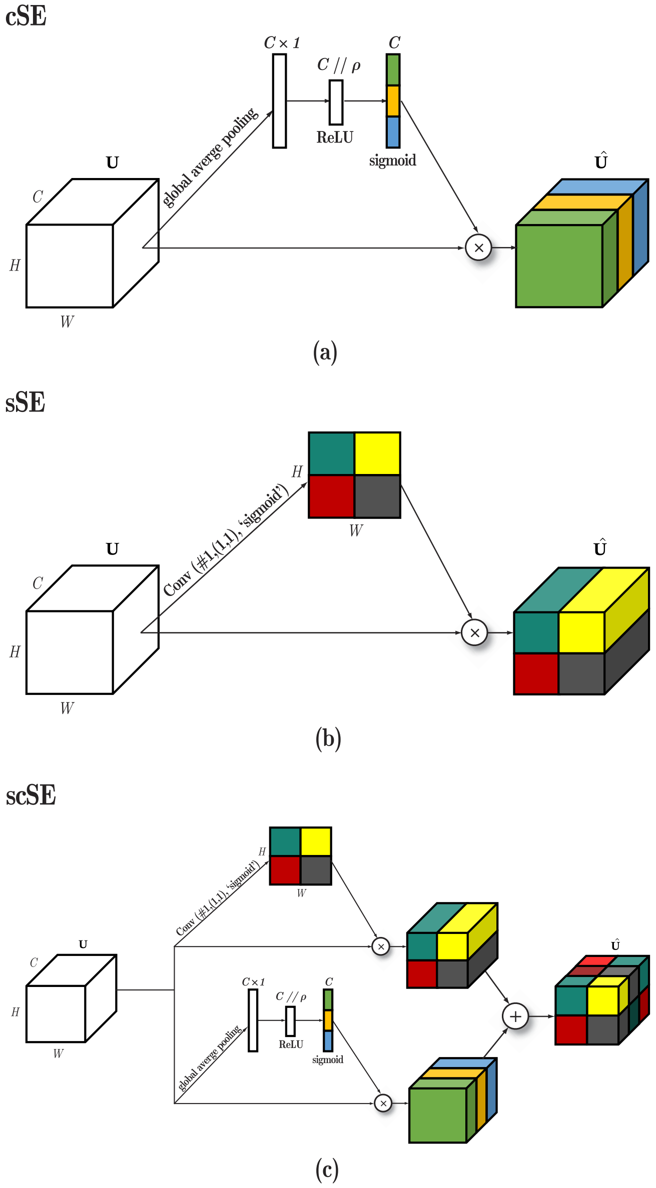


FIGURE 1. Diagram of different SE blocks: (a) describes cSE block procedure, (b) ilustrates sSE block framework and (c) shows scSE block by combining (a) and (b).

1) SPATIAL SQUEEZE AND CHANNEL EXCITATION BLOCK (cSE)

In a *cSE* module (depicted in Fig. 1(a)) for spatial squeeze and channel excitation, a unique feature map of each channel from $\mathbf{U}$ is first obtained by means of global average pooling. This operator produces a vector $\mathbf{z} \in \mathbb{R}^{1\times 1\times C}$. The kth element of such vector can be expressed as:

$$z_k = \frac{1}{H \times W} \sum_{i}^{H} \sum_{j}^{W} \mathbf{u}_k(i,j), \quad k = 1, \dots, C, \tag{1}$$

where $\mathbf{u}_k(i,j)$ denotes the (i,j) element of the kth channel feature map.

As suggested by Eq. (1), global spatial information is embedded in vector $\mathbf{z}$. This representation is then used to extract channel-wise dependencies using two fully-connected layers, obtaining the transformed vector $\hat{\mathbf{z}}$. Therefore, $\hat{\mathbf{z}}$ can be expressed as $\hat{\mathbf{z}} = \mathbf{W}_1(\delta(\mathbf{W}_2\mathbf{z}))$, where δ represents ReLU activation. $\mathbf{W}_1 \in \mathbb{R}^{C\times\frac{C}{\rho}}$ and $\mathbf{W}_2 \in \mathbb{R}^{\frac{C}{\rho}\times C}$ are the weights of the fully-connected layers, and ρ is a ratio parameter. As last step, the activation range is compressed to the interval [0, 1] using a sigmoid activation function, σ. This final step indicates the importance of each channel and how they should be rescaled. The purpose of this recalibration is to let the network ignore channels with less information and emphasize the ones that provide more meaningful information. Then, the rescaled feature maps, $\hat{\mathbf{U}}$, can be expressed as [13], [14]:

$$\hat{\mathbf{U}}_{cSE} = \mathbf{F}_{cSE}(\mathbf{U}) = [\sigma(\hat{z}_1)\mathbf{u}_1, \dots, \sigma(\hat{z}_C)\mathbf{u}_C], \tag{2}$$

where $\hat{z}_k$ are the elements of the transformed vector $\hat{\mathbf{z}}$.

2) CHANNEL SQUEEZE AND SPATIAL EXCITATION BLOCK (sSE)

In the case of an *sSE* block [13], as shown in Fig. 1(b), a unique convolutional layer with one filter and (1, 1) kernel size is implemented to obtain a channel squeeze and spatial excitation effect. Here, it is assumed an alternative representation of the input tensor as $\mathbf{U} = [\mathbf{u}^{1,1}, \mathbf{u}^{1,2}, \dots, \mathbf{u}^{i,j}, \dots, \mathbf{u}^{H,W}]$ where $\mathbf{u}^{i,j} \in \mathbb{R}^{1\times 1\times C}$. The convolution can be expressed as $\mathbf{q} = \mathbf{W} \star \mathbf{U}$, being $W \in \mathbb{R}^{1\times 1\times C\times 1}$ and $q \in \mathbb{R}^{H\times W}$. Each $q_{i,j}$ represents the combination of all channels in location (i,j). As done with *cSE*, the output of this convolution is passed through a sigmoid function. Each $\sigma(q_{i,j})$ determines the importance of the specific location (i,j) across the feature map. Like the previous block, this recalibration process indicates which locations are more meaningful during the training procedure. As a result, the output of the SE block can be expressed as [13]:

$$\hat{\mathbf{U}}_{sSE} = \mathbf{F}_{sSE}(U) = [\sigma(q_{1,1})u^{1,1}, \dots, \sigma(q_{H,W})u^{H,W}]. \tag{3}$$

3) SPATIAL AND CHANNEL SQUEEZE & EXCITATION BLOCK (scSE)

The *scSE* block [13] is implemented by declaring *cSE* and *sSE* blocks in parallel and adding both outputs (see Fig. 1(c)). It has been reported that the *scSE* block shows better performance than *cSE* and *sSE* used independently. In this case, a location (i, j, c) gets a higher sigmoid or activation value when both channel and spatial recalibration get it at the same time [13]:

$$\hat{\mathbf{U}}_{scSE} = \hat{\mathbf{U}}_{cSE} + \hat{\mathbf{U}}_{sSE}. \tag{4}$$

In this case, the network focuses on feature maps that are meaningful from both a spatial and channel-wise point of view.

C. RESIDUAL NETWORKS

Residual networks were first proposed in [18]. A network of this kind replaces the standard stacked convolutional layers [19] by residual blocks. Residual layers are designed to approximate a residual function: $\mathcal{F}(\mathbf{X}) := \mathcal{H}(\mathbf{X}) - \mathbf{X}$, where $\mathcal{H}(\cdot)$ represents the mapping to be fit by a set of stacked layers and $\mathbf{X}$ represents the input to the first of such stacked layers. The original function $\mathcal{H}$ can therefore be defined as $\mathcal{H}(\mathbf{X}) = \mathcal{F}(\mathbf{X}) + \mathbf{X}$. The main motivation of choosing this kind of network corresponds to the intuition that optimizing a residual mapping may be easier than optimizng the original unreferenced one, as in a classical convolutional network. A simple way of implementing residual learning in CNNs is by adding a shortcut connection that performs as an identity mapping, adding back the input $\mathbf{X}$ to the output of the residual block $\mathcal{F}(\mathbf{X})$. In the first proposition of the residual block, Rectified ReLU activation is applied after the addition and the result of such activation becomes the input for the next residual block. Note, that in the first configuration, shortcut connections do not add more parameters nor extra computational cost. Therefore, deeper networks can be trained with little additional effort, reducing vanishing-gradient problems. As it will be later explained, in this work, the identity mapping is replaced with a 1 × 1 convolutional layer as it is explained in Section III. Therefore, this work function can be expressed as $\mathcal{H}(\mathbf{X}) = \mathcal{F}(\mathbf{X}) + g(\mathbf{X})$, where $g(\cdot)$ represents the convolutional process with the learnt filter coefficients.

III. CONFIGURATIONS FOR SQUEEZE-AND-EXCITATION RESIDUAL NETWORKS

According to [14], SE blocks exhibit better performance when deployed on networks with residual configuration than on VGG-style networks. Therefore, two novel residual blocks implementing *scSE* modules are presented in this paper. The performance of these two newly proposed blocks is compared against other state-of-the-art residual configurations that incorporate SE modules.

A. SE BLOCK DESCRIPTION

All the configurations analyzed in this work are depicted in Fig. 2. In the following, we describe in details these blocks.

1) Conv-RESIDUAL

Shown in Fig. 2(a), is inspired by [18]. It is used as a baseline in order to validate the network performance without any SE and how much it can be improved when incorporating these blocks. In the present work some slight modifications for a more convenient implementation were introduced: the shortcut connection was implemented with a 1 × 1 convolutional layer and the activation after the addition was set to an exponential linear unit (ELU) function [20], [21].

2) Conv-POST

Shown in Fig. 2(b), is inspired by the block referred to as *se-POST* in [14]. The *scSE* block is included at the end and is equivalent to a recalibration of the *Conv-residual* block.

3) Conv-POST-ELU

Shown in Fig. 2(c), is very similar to the above *Conv-POST* block, but the recalibration is performed over the ELU-activated output of the residual block.

4) Conv-STANDARD

Shown in Fig. 2(d), is inspired by [14], where the *scSE* block is stacked after the convolutional block for recalibrating prior to adding the shortcut branch.

5) Conv-StandardPOST

Shown in Fig. 2(e) is proposed in this work to create a double shortcut connection, one before SE calibration and one after. The idea is to let the network learn residual mappings simultaneously with and without SE recalibration, thus, affecting the way in which the block optimizes the residual by considering jointly standard and post SE-calibrated outputs.

6) Conv-StandardPOST-ELU

Shown in Fig. 2(f) is the other proposed block, corresponding to the above explained *Conv-StandardPOST* block, but followed by ELU activation.

To summarize, the output $\mathbf{X}_{l+1}$ of each block for an input $\mathbf{X}$ is given by:

$$\text{a)}\quad \mathbf{X}_{l+1} = \mathcal{R}\left(\mathcal{F}(\mathbf{X}) + g(\mathbf{X})\right), \tag{5}$$

$$\text{b)}\quad \mathbf{X}_{l+1} = \mathbf{F}_{SE}\left(\mathcal{F}(\mathbf{X}) + g(\mathbf{X})\right), \tag{6}$$

$$\text{c)}\quad \mathbf{X}_{l+1} = \mathbf{F}_{SE}\left(\mathcal{R}\left(\mathcal{F}(\mathbf{X}) + g(\mathbf{X})\right)\right), \tag{7}$$

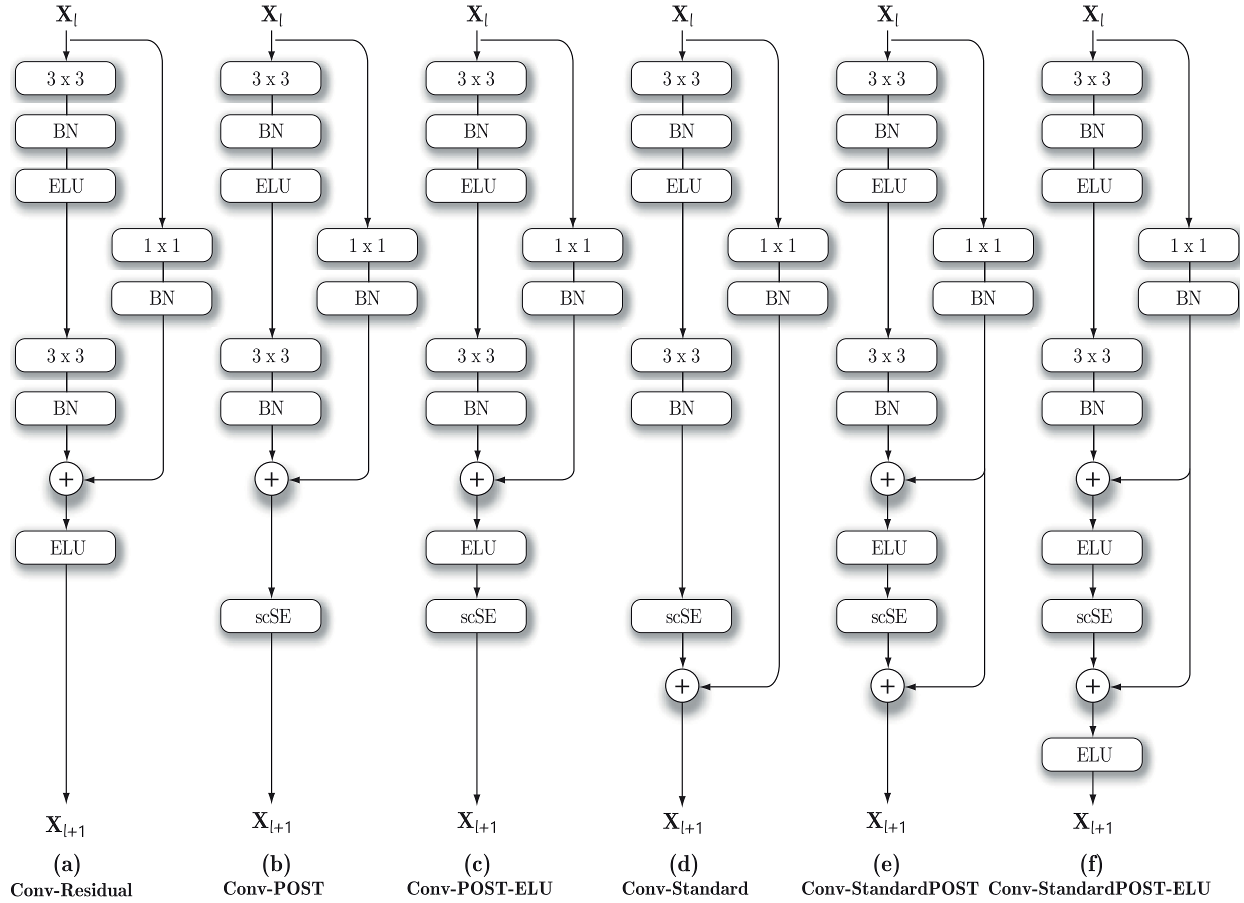
$$\text{d)}\quad \mathbf{X}_{l+1} = \mathcal{F}_{(SE)}(\mathbf{X}) + g(\mathbf{X}), \tag{8}$$

$$\text{e)}\quad \mathbf{X}_{l+1} = \mathbf{F}_{SE}\left(\mathcal{R}\left(\mathcal{F}(\mathbf{X}) + g(\mathbf{X})\right)\right) + g(\mathbf{X}), \tag{9}$$

$$\text{f)}\quad \mathbf{X}_{l+1} = \mathcal{R}\left(\mathbf{F}_{SE}\left(\mathcal{R}\left(\mathcal{F}(\mathbf{X}) + g(\mathbf{X})\right)\right) + g(\mathbf{X})\right), \tag{10}$$

where $\mathcal{R}(\cdot)$ refers to ELU activation function with α parameter set to 1 and $\mathcal{F}_{(SE)}$ denotes a residual function that includes SE calibration. As it will be discussed in Section V, the two proposed configurations have been shown to outperform the rest in the considered acoustic scene analysis task.

In order to avoid possible duplications or expansion processes in the channel dimension, the identity branch is replaced by a convolutional layer with a (1, 1) kernel size



(a) Conv-Residual (b) Conv-POST (c) Conv-POST-ELU (d) Conv-Standard (e) Conv-StandardPOST (f) Conv-StandardPOST-ELU

FIGURE 2. Different residual squeeze-excitation blocks analyzed in this work: (a) is inspired by the first residual block proposed in [18]; (b), (c) and (d) are inspired by the work done in [14]; (e) and (f) are the two novel configurations proposed in this work.

and with the same number of filters as the residual branch. Including such convolutional layer in the shortcut branch creates a projection that avoids dimensionality conflicts in the residual block addition.

By looking at Fig. 2, it can be clearly observed that the most representative feature of the two proposed blocks, (e) and (f), resides in the use of two skip connections: one before SE re-calibration and one after. This double short-cut connection leads the network towards the learning of a global residual function embedding an inner and SE-calibrated partial residual. The objective is to facilitate the learning of calibration weightings by using the same residual rationale.

In general, the presence or absence of relevant acoustic events within an input audio clip can be very important when addressing the ASC problem. The use of spatial and channel-wise recalibration at different depths of the network adds a mechanism to allow the network weight properly, according to their importance, the different dimensions of the information flowing throughout the network. Therefore, SE modules are expected to add flexibility for identifying relevant acoustic textures or events, making easier to infer the type of underlying acoustic scene.

B. NETWORK ARCHITECTURE

The CNN implemented in order to validate the behaviour of the different SE configurations has been inspired on [22] where a VGG-style [19] network with 3 convolutional blocks followed by different max-pooling and dropout [23] operators is implemented. In the present work, the original convolutional blocks have been replaced with the different residual squeeze-excitation blocks proposed in this study. The max-pooling, dropouts and linear layers are configured with the same parameters as in [22]. The network architecture can be found in Table 1.

As the database used in the current work is much smaller than the one in [14], some of the hyperparameters that define the components of the scSE block had to be modified. The number of elements in the Dense layer with ReLU activation in Fig. 1(a) has been set to 16 in the first Residual-scSE block, the same as in [14] in its cSE block, but the number of filters at the input, C, has been set to $C = 32$. Therefore, the ratio between these parameters throughout the network is two, as observed Table 1. The number of network parameters that implement SE residual blocks, i.e. those represented in Fig. 2(b)-(f), is 528,334. On the other hand, the network that does not integrate SE modules has 506,606. Note, therefore, that there is only a slight increase of approximately 4% in the

TABLE 1. Proposed network for validating the scSE configurations of Fig. 2. Values preceeded by # correspond to the number of filters. Kernel sizes are set as indicated in Fig. 2. This architecture is inspired by the work in [22].

Residual-scSE block (#32, $\rho = 2$)
MaxPooling(2,10)
Dropout(0.3)
Residual-scSE block (#64, $\rho = 2$)
MaxPooling(2,5)
Dropout(0.3)
Residual-scSE block (#128, $\rho = 2$)
MaxPooling(2,5)
Dropout(0.3)
Flatten
[Dense(100), batch normalization, ELU]
Dropout(0.4)
[Dense(10), batch normalization, softmax]

SE networks. Table 2 shows as well the number of floating point operations (FLOPs) involved in each network.

IV. EXPERIMENTAL DETAILS

This section describes in detail the experimental implementation carried out to conduct the analysis of the presented SE residual blocks, including the datasets, the audio representation selected to feed the network and the training configuration.

A. DATASET

To check the behavior of these implementations in an ASC problem, the TAU Urban Acoustic Scenes 2019, Development dataset presented in Task 1A of the 2019 edition of DCASE has been used [10]. The database consists of 40 hours of stereo audio-recording in different urban environments and landscapes such as parks, metro stations, airports, etc. making a total of 10 different scenes. These have been recorded in different cities such as Barcelona, Paris or Helsinki, among others. All audio clips are 10-second long. They are divided into two subsets of 9185 and 4185 clips for training and validation, respectively. Although there are a slightly different number of samples available for each class, the data set is not severely unbalanced.

B. AUDIO PROCESSING

The input to the network is a 2D log-Mel spectrogram representation with 3 audio channels. The three channels are composed of the harmonic and percussive component [24], [25] of the signal converted to mono and the difference between left (L) and right (R) channels. That is, the first channel corresponds to the log-Mel spectrogram of the harmonic source, the second channel corresponds to the same representation but over the percussive source and the last one to the log-Mel spectrogram of the difference between channels calculated by subtracting left and right channels ($L - R$). This representation, known as HPD, was presented in [22]. The log-Mel spectrogram is calculated using 64 Mel filters with a window size of 40 ms and 50% overlap. Therefore, an audio clip becomes a $64 \times T \times 3$ array with T being the number of time frames. In this specific dataset, the input audio representation corresponds to an array of dimension $64 \times 500 \times 3$.

TABLE 2. Parameters and FLOPs analysis from the studied network configurations.

Residual block	Network parameters	Network FLOPs
Conv-Residual	506606	1009115
Conv-POST Conv-POST-ELU Conv-Standard Conv-StandardPOST Conv-StandardPOST-ELU	528334	1052571

C. TRAINING PROCEDURE

The training process was optimized using the Adam optimizer [26]. The cost function used was the categorical crossentropy. Training was limited to a maximum of 500 epochs but early stopping is applied if the validation accuracy does not improve by 50 epochs. If this same metric does not improve in 20 epochs, the learning rate is decreased by a factor of 0.5. The batch size used was 32 samples.

V. RESULTS

In order to analyze the contributions of this work with respect to other state-of-the-art approaches, the results obtained with the different configurations presented in this work (see Fig. 2) are compared to the ones obtained by different authors in Task 1A of DCASE 2019 using the same dataset. For a fair comparison, only submissions not making use of data augmentation techniques are considered. In the case of submissions that presented an ensemble of several models, only the results of the best performing model making up the ensemble are taken into account. For example, in [27] a global development accuracy of 78.3% is reported, but that value was obtained by averaging 5 models. The best individual model obtained 72.4%, so this is the value presented in Table 3. This said, please be aware that the accuracy of the final submission[1] may differ from that presented in Table 3. Next, we summarize some important features of the competing approaches.

- **Wang_NWPU_task1a** [27]: the audio representation considers two channels using a log-Mel Spectrogram from harmonic and percussive sources similar to our representation. The number of Mel filters is set to 256. The window size is set to 64 ms and the hop size to 15 ms. Mel filters are calculated with cutoff frequencies from 50 Hz to 14 kHz. A VGG-style CNN [19] is used as a classifier.

[1] http://dcase.community/challenge2019/task-acoustic-scene-classification-results-a

TABLE 3. Accuracy results from the validation partition in development phase.

System	Development accuracy (%)
Baseline [10]	62.5
Wang_NWPU_task1a [27]	72.4
Fmta91_KNToosi_task1a [28]	70.49
MaLiu_BIT_task1a [29]	76.1 (evaluation)
DSPLAB_TJU_task1a [30]	64.3
Kong_SURREY_task1a [31]	69.2
Liang_HUST_task1a [32]	70.70
Salvati_DMIF_task1a [33]	69.7
Conv-Residual	*74.51 ± 0.65*
Conv-Standard	*75.16 ± 0.33*
Conv-POST	*75.84±0.65*
Conv-POST-ELU	*75.81±0.47*
Conv-StandardPOST	***76.72±0.59***
Conv-StandardPOST-ELU	*76.00±0.55*

- **Fmta91_KNToosi_task1a** [28]: wavelet scattering spectral features are extracted from the mono audio signal. A random subspace method is used as classifier.
- **MaLiu_BIT_task1a** [29]: Deep Scattering Spectra features (DSS) are extracted from each stereo channel. Classification is performed with a Convolutional Recurrent Neural Network (CRNN). For this network, Table 3 does not report the accuracy on the development set (only on the evaluation set). This is because of some mismatch reported by the authors in the validation procedure with the configuration of the dataset.
- **DSPLAB_TJU_task1a** [30]: this submission approaches the problem in a more classical way extracting audio statistical features such as ZRC, RMSE, spectrogram centroid, etc. A GMM is used as a classifier.
- **Kong_SURREY_task1a** [31]: this submssion can be defined as the state-of-the-art framework in ASC problem. The audio representation considers also the log-Mel spectrogram. The classifier is a VGG-based [19] CNN. This network is a fully convolutional network with no linear layers implemented. The feature maps are reshaped into a one dimensional vector using a global average pooling before the decision layer.
- **Liang_HUST_task1a** [32]: in this method, the log-Mel spectrogram is first extracted after converting the audio signal to mono. Interestingly, the log-Mel spectrogram is divided into two-seconds spectrograms, that means that spectrogram shapes change from $[F \times T \times 1]$ to $[F \times (T/5) \times 1]$. This configuration allows training with audio samples consisting of 5 different spectrograms instead of one. A CNN with frequency attention mechanism is implemented as classifier. For more detail of the attention implementation, see [32].
- **Salvati_DMIF_task1a** [33]: unlike the other submissions, this one works directly on the audio vector. To this end, a 1D convolutional network is implemented. Although some recent efforts have been made in this direction [34], the state-of-the-art literature shows that 2D audio representations, such as spectrograms, still obtain the better classification results [35].
- **DCASE baseline** [10]: the audio is first converted to mono and a log-Mel spectrogram is extracted. In this case, only 40 Mel bins are calculated instead of 64, which is the typical state-of-the-art choice. A CNN is used as a classifier with 2 convolutional layers. The 1D conversion before classification layers is performed by a flatten layer. A dense layer is stacked before the decision layer.

A. GLOBAL PERFORMANCE

Although the results of the DCASE challenge only report the mean accuracy value, we consider 10 runs to provide not only the mean accuracy value, but also the standard deviation. As it can be seen in Table 3, all the configurations detailed in Fig. 2 obtain better accuracy than the DCASE baseline. The contribution of the scSE block is easily justified as *Conv-Residual* gets the lowest performance among the studied configurations. In general, *POST* configurations show a slight improvement compared to the *Standard* configuration. This behaviour differs from what was reported in the original paper, [14], in which these blocks were analyzed in the image domain, where the Standard block outperforms the POST block. There is no remarkable difference between *Conv-POST* and *Conv-POST-ELU*. It is also shown that the networks that incorporate the two novel blocks presented in this work, the ones depicted in Figs. 2(e) and (f), exhibit the best accuracy values. The shortcut addition at two differente points of the residual block, this is, before and after the scSE block, allowed the network to obtain a more precise classification in this ASC task.

B. CLASS-WISE PERFORMANCE

Fig. 3 shows confusion matrices for each of the analyzed residual blocks in this work. In general, the performance across the different classes is considerably balanced. The "Public square" class is the one showing the worst performance, tending to be misclassified as "Street, Pedestrian". Other similar classes such as "Airport" and "Shopping mall" or "Tram" and "Bus" or "Metro" tend also to produce common errors in the analyzed networks.

By analyzing the class-wise performance of the two proposed blocks with respect to the conventional *Conv-Residual* block, substantial improvements are observed. Considering the proposed *Conv-StandardPOST* block, a significant improvement is observed for the classes "Metro station" and "Street, Pedestrian". Other classes showing slight improvements are "Shopping mall", "Park" or "Public square". The class showing the worst relative result was "Airport". On the other hand, the second proposed block *Conv-StandardPOST-ELU* provides substantial improvements in "Street traffic"

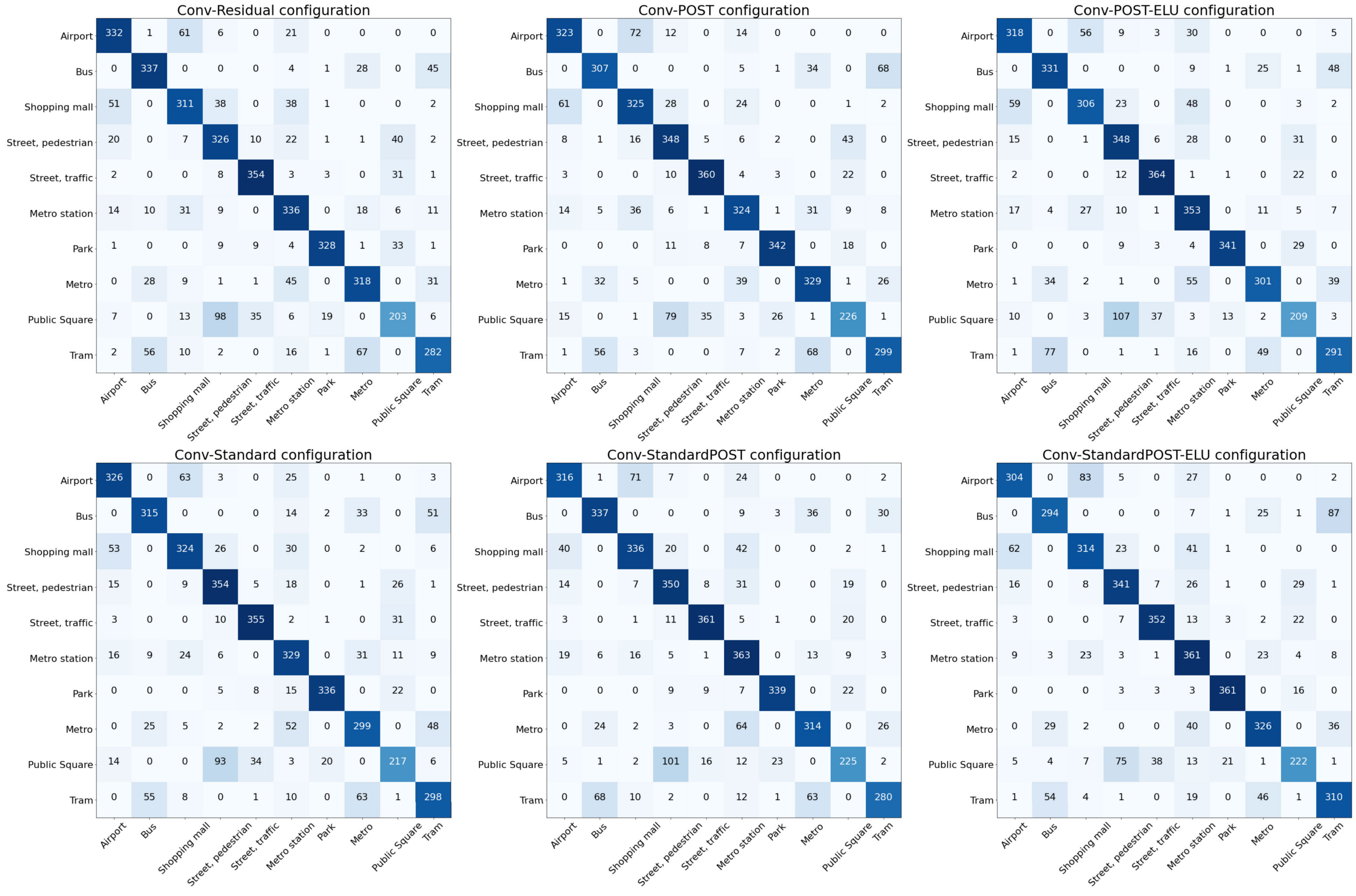
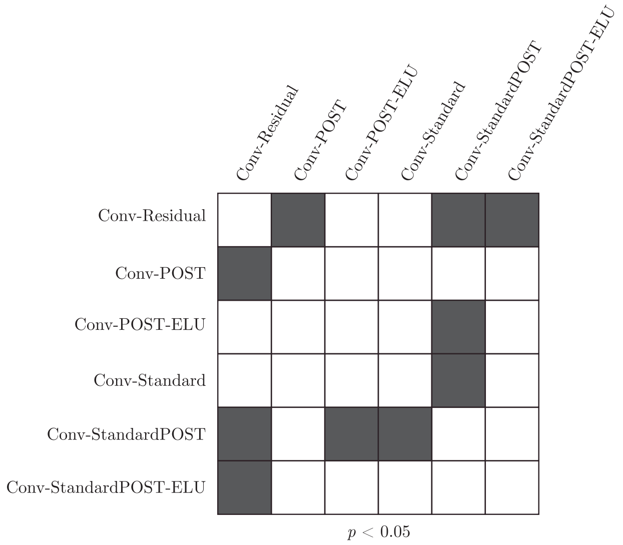


FIGURE 3. **Confusion matrices for the generated models over the evaluation dataset.**

and "Park", but other classes like "Airport" or "Bus" were degraded.

Finally, when considering the performance of networks implementing SE blocks together, from a general perspective, it is noticed that classes like "Street, Pedestrian", "Park" or "Public square" are improved with respect to the conventional residual network. Only the class "Airport" shows the best performance in the conventional network, followed by "Bus". The remaining classes are improved or worsened across all configurations in a degree not as significant as the aforementioned ones.



	Conv-Residual	Conv-POST	Conv-POST-ELU	Conv-Standard	Conv-StandardPOST	Conv-StandardPOST-ELU
Conv-Residual		■			■	■
Conv-POST	■					
Conv-POST-ELU					■	
Conv-Standard					■	
Conv-StandardPOST	■		■	■		
Conv-StandardPOST-ELU	■					

$p < 0.05$

FIGURE 4. **Pairwise analysis of the studied residual networks using McNemar's test. Gray cells indicate *p*-values below a 0.05 significance level.**

C. SIGNIFICANCE TEST

To determine if there are statistically significant differences in the performance of the different blocks analyzed in this work, a McNemar's test has been carried out [36]. This test, which is a paired non-parametric hypothesis test, has been widely recommended for evaluating deep learning models, which are often trained on very large datasets. The test is based on a contingency table created from the results obtained for two methods trained on exactly the same training test and evaluated on the same test set. The null hypothesis of the test is that the performance of the two analyzed systems disagree to the same amount. If the null hypothesis is rejected, there is evidence to suggest that the two systems have different performance when trained on a particular training set. Given a significance level α, if $p < \alpha$, there may be sufficient evidence to claim that the two classifiers show different proportions of errors. The result of applying the McNemar's test to all the available system pairs is shown in Fig. 4. Gray cells indicate p-values below a significance level of

0.05. It is confirmed that the two proposed blocks, *Conv-StandardPOST* and *Conv-StandardPOST-ELU*, show significant differences in performance with respect to all the other blocks but *Conv-POST*, which was the third best performing block. However, no significant differences can be observed between these new blocks, which only differ in the final ELU activation.

VI. CONCLUSION

The use of squeeze-excitation blocks in convolutional neural networks allows to perform a spatial and channel-wise recalibration of its inner feature maps. This work presented the use of squeeze-excitation residual networks for addressing the acoustic scene classification problem, and presented two novel block configurations that consider residual learning of standard and recalibrated outputs jointly. Results over the well-known DCASE dataset confirm that the proposed blocks provide meaningful improvements by adding a slight architecture modification, outperforming other competing approaches when no data augmentation or model ensembles are considered.

REFERENCES

[1] M. Valenti, S. Squartini, A. Diment, G. Parascandolo, and T. Virtanen, "A convolutional neural network approach for acoustic scene classification," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, May 2017, pp. 1547–1554.

[2] S. H. Bae, I. Choi, and N. S. Kim, "Acoustic scene classification using parallel combination of LSTM and CNN," in *Proc. Detection Classification Acoustic Scenes Events Workshop (DCASE)*, 2016, pp. 11–15.

[3] L. Pham, H. Phan, T. Nguyen, R. Palaniappan, A. Mertins, and I. McLoughlin, "Robust acoustic scene classification using a multi-spectrogram encoder-decoder framework," 2020, *arXiv:2002.04502*. [Online]. Available: http://arxiv.org/abs/2002.04502

[4] Y. Han and K. Lee, "Acoustic scene classification using convolutional neural network and multiple-width frequency-delta data augmentation," 2016, *arXiv:1607.02383*. [Online]. Available: http://arxiv.org/abs/1607.02383

[5] I. Martín-Morató, M. Cobos, and F. J. Ferri, "A case study on feature sensitivity for audio event classification using support vector machines," in *Proc. IEEE 26th Int. Workshop Mach. Learn. Signal Process. (MLSP)*, Sep. 2016, pp. 1–6.

[6] I. Martín-Morató, M. Cobos, and F. J. Ferri, "Adaptive mid-term representations for robust audio event classification," *IEEE/ACM Trans. Audio, Speech, Language Process.*, vol. 26, no. 12, pp. 2381–2392, Dec. 2018.

[7] G. Roma, W. Nogueira, and P. Herrera, "Recurrence quantification analysis features for auditory scene classification," Tech. Rep., 2013. [Online]. Available: http://c4dm.eecs.qmul.ac.uk/sceneseventschallenge/abstracts/SC/RNH.pdf

[8] D. Li, J. Tam, and D. Toub, "Auditory scene classification using machine learning techniques," Tech. Rep., 2013. [Online]. Available: http://c4dm.eecs.qmul.ac.uk/sceneseventschallenge/abstracts/SC/LTT.pdf

[9] A. Rakotomamonjy and G. Gasso, "Histogram of gradients of time-frequency representations for audio scene classification," Tech. Rep., 2013. [Online]. Available: http://c4dm.eecs.qmul.ac.uk/sceneseventschallenge/abstracts/SC/RG.pdf

[10] A. Mesaros, T. Heittola, and T. Virtanen, "Acoustic scene classification in DCASE 2019 challenge: Closed and open set classification and data mismatch setups," in *Proc. Detection Classification Acoustic Scenes Events Workshop (DCASE)*. New York, NY, USA: New York Univ., Oct. 2019, pp. 164–168. [Online]. Available: http://dcase.community/documents/workshop2019/proceedings/DCASE2019Workshop_Mesaros_14.pdf

[11] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, "Mixup: Beyond empirical risk minimization," 2017, *arXiv:1710.09412*. [Online]. Available: http://arxiv.org/abs/1710.09412

[12] M. D. McDonnell and W. Gao, "Acoustic scene classification using deep residual networks with late fusion of separated high and low frequency paths," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, 2020, pp. 141–145.

[13] A. G. Roy, N. Navab, and C. Wachinger, "Concurrent spatial and channel 'squeeze & excitation'in fully convolutional networks," in *Proc. Int. Conf. Med. Image Comput. Comput.-Assist. Intervent.* Cham, Switzerland: Springer, 2018, pp. 421–429.

[14] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 7132–7141, doi: 10.1109/CVPR.2018.00745.

[15] L. Yang, X. Chen, L. Tao, and X. Gu, "Multi-scale fusion and channel weighted CNN for acoustic scene classification," in *Proc. 2nd Int. Conf. Signal Process. Mach. Learn.*, Nov. 2019, pp. 41–45.

[16] J. Lee, T. Kim, J. Park, and J. Nam, "Raw waveform-based audio classification using sample-level CNN architectures," 2017, *arXiv:1712.00866*. [Online]. Available: http://arxiv.org/abs/1712.00866

[17] O. Akiyama and J. Sato, "Multitask learning and semisupervised learning with noisy data for audio tagging," DCASE2019 Challenge, Tech. Rep., 2019. [Online]. Available: https://pdfs.semanticscholar.org/95d3/1eb466b591161c7fd6fd8e14c146a3ccab71.pdf

[18] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.

[19] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014, *arXiv:1409.1556*. [Online]. Available: http://arxiv.org/abs/1409.1556

[20] A. Shah, E. Kadam, H. Shah, S. Shinde, and S. Shingade, "Deep residual networks with exponential linear unit," in *Proc. 3rd Int. Symp. Comput. Vis. Internet*, 2016, pp. 59–65.

[21] D.-A. Clevert, T. Unterthiner, and S. Hochreiter, "Fast and accurate deep network learning by exponential linear units (ELUs)," 2015, *arXiv:1511.07289*. [Online]. Available: http://arxiv.org/abs/1511.07289

[22] S. Perez-Castanos, J. Naranjo-Alcazar, P. Zuccarello, M. Cobos, and F. J. Ferri, "CNN depth analysis with different channel inputs for acoustic scene classification," 2019, *arXiv:1906.04591*. [Online]. Available: http://arxiv.org/abs/1906.04591

[23] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 1929–1958, 2014.

[24] D. Fitzgerald, "Harmonic/percussive separation using median filtering," in *Proc. DAFX*, 2010, vol. 10, no. 4, pp. 1–4.

[25] J. Driedger, M. Müller, and S. Disch, "Extending harmonic-percussive separation of audio signals," in *Proc. ISMIR*, 2014, pp. 611–616.

[26] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*. [Online]. Available: http://arxiv.org/abs/1412.6980

[27] M. Wan, R. Wang, B. Wang, J. Bai, C. Chen, Z. Fu, J. Chen, X. Zhang, and S. Rahardja, "Ciaic-ASC system for DCASE 2019 challenge task1," DCASE2019 Challenge, Tech. Rep., 2019. [Online]. Available: http://dcase.community/documents/challenge2019/technical_reports/DCASE2019_Mou_41_t1.pdf

[28] F. Arabnezhad and B. Nasersharif, "Urban acoustic scene classification using binaural wavelet scattering and random subspace discrimination method," DCASE2019 Challenge, Tech. Rep., Jun. 2019. [Online]. Available: http://dcase.community/documents/challenge2019/technical_reports/DCASE2019_Fmta91_67.pdf

[29] S. Ma and W. Liu, "Acoustic scene classification based on binaural deep scattering spectra with neural network," DCASE2019 Challenge, Tech. Rep., Jun. 2019. [Online]. Available: http://dcase.community/documents/challenge2019/technical_reports/DCASE2019_MaLiu_112.pdf

[30] B. Ding, G. Liu, and J. Liang, "Acoustic scene classification based on ensemble system," DCASE2019 Challenge, Tech. Rep., Jun. 2019. [Online]. Available: http://dcase.community/documents/challenge2019/technical_reports/DCASE2019_DSPLAB_44.pdf

[31] Q. Kong, Y. Cao, T. Iqbal, Y. Xu, W. Wang, and M. D. Plumbley, "Cross-task learning for audio tagging, sound event detection and spatial localization: DCASE 2019 baseline systems," 2019, *arXiv:1904.03476*. [Online]. Available: https://arxiv.org/abs/1904.03476

[32] H. Liang and Y. Ma, "Acoustic scene classification using attention-based convolutional neural network," DCASE2019 Challenge, Tech. Rep., Jun. 2019. [Online]. Available: http://dcase.community/documents/challenge2019/technical_reports/DCASE2019_Liang_3.pdf

[33] D. Salvati, C. Drioli, and G. L. Foresti, "Urban acoustic scene classification using raw waveform convolutional neural networks," DCASE2019 Challenge, Tech. Rep., Jun. 2019. [Online]. Available: http://dcase.community/documents/challenge2019/technical_reports/DCASE2019_Salvati_35.pdf

[34] J. Naranjo-Alcazar, S. Perez-Castanos, I. Martin-Morato, P. Zuccarello, and M. Cobos, ''On the performance of residual block design alternatives in convolutional neural networks for end-to-end audio classification,'' 2019, *arXiv:1906.10891*. [Online]. Available: http://arxiv.org/abs/1906.10891

[35] S. Dieleman and B. Schrauwen, ''End-to-end learning for music audio,'' in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, May 2014, pp. 6964–6968.

[36] T. G. Dietterich, ''Approximate statistical tests for comparing supervised classification learning algorithms,'' *Neural Comput.*, vol. 10, no. 7, pp. 1895–1923, Oct. 1998.

JAVIER NARANJO-ALCAZAR (Graduate Student Member, IEEE) received the Telecommunications degree and the master's degree in telecommunications engineering from the Universitat Politècnica de València, Valencia, Spain, in 2016 and 2018, respectively. He is currently pursuing the Ph.D. degree with the Department of Computer Science, Universitat de València, funded by the Torres Quevedo Program and the Valencian Start-Up Visualfy. His research interests include machine listening, few-shot learning, and open-set recognition. He was a recipient of the Best M.Sc. Thesis Award from the Regional Telecommunications Engineering Association, in 2019.

SERGI PEREZ-CASTANOS received the Telecommunications degree and the master's degree in telecommunications engineering from the Universitat Politècnica de València, Valencia, Spain, in 2016 and 2018, respectively. He is currently working as a Machine Learning Engineer with Visualfy. His research interests include machine listening, anomaly detection, and audio captioning.

PEDRO ZUCCARELLO received the Electronics Engineering degree from the University of Buenos Aires, Argentina, the M.Sc. degree in telecommunications from the Universitat Politècnica de Valencia, Valencia, Spain, and the Ph.D. degree from the Universitat de València, Valencia. He developed most of his career as a Researcher in several public Research and Development institutions, such as the Institute of Microelectronics of Barcelona, Barcelona, Spain, or the Institute of Corpuscular Physics, Valencia. Since 2017, he has been the Head of the Artificial Intelligence Group, Visualfy, private startup company. He received several Postdoctoral Fellowships such as Val-I+D, from the Valencian Government, or Torres Quevedo, from the Spanish Ministry of Science and Education. His research interests include artificial intelligence, machine learning, signal processing, electronics, and microelectronic design. He has coauthored nearly 30 articles in international peer-review journals and conferences in these areas.

MAXIMO COBOS (Senior Member, IEEE) received the master's degree in telecommunications and the Ph.D. degree in telecommunications engineering from the Universitat Politècnica de València, Valencia, Spain, in 2007 and 2009, respectively. In 2010, he received the Campus de Excelencia Postdoctoral Fellowship to work at the Institute of Telecommunications and Multimedia Applications. In 2011, he joined the Universitat de València, where he is currently an Associate Professor. His research interests include digital signal processing and machine learning for audio and multimedia applications. He has authored/coauthored more than 100 technical articles in international journals and conferences in these areas. He is a member of the Audio Signal Processing Technical Committee of the European Acoustics Association. He completed his studies with honors under the University Faculty Training Program (FPU) and was a recipient of the Ericsson Best Ph.D. Thesis Award from the Spanish National Telecommunications Engineering Association. He serves as an Associate Editor for the IEEE SIGNAL PROCESSING LETTERS.

B. Open set audio classification using autoencoders trained on few data

sensors

MDPI

Article

Open Set Audio Classification Using Autoencoders Trained on Few Data

Javier Naranjo-Alcazar [1,2,*], Sergi Perez-Castanos [1], Pedro Zuccarello [1], Fabio Antonacci [3] and Maximo Cobos [2]

[1] Visualfy, 46181 Benisanó, Spain; sergi.perez@visualfy.com (S.P.-C.); pedro.zuccarello@visualfy.com (P.Z.)
[2] Computer Science Department, Universitat de València, 46100 Burjassot, Spain; maximo.cobos@uv.es
[3] Dipartimento di Elettronica, Informazione e Bioingegneria (DEIB), Politecnico di Milano, 20133 Milan, Italy; fabio.antonacci@polimi.it
[*] Correspondence: janal2@alumni.uv.es; Tel.: +34-669-287-584

Received: 22 May 2020; Accepted: 1 July 2020; Published: 3 July 2020

Abstract: Open-set recognition (OSR) is a challenging machine learning problem that appears when classifiers are faced with test instances from classes not seen during training. It can be summarized as the problem of correctly identifying instances from a known class (seen during training) while rejecting any unknown or unwanted samples (those belonging to unseen classes). Another problem arising in practical scenarios is few-shot learning (FSL), which appears when there is no availability of a large number of positive samples for training a recognition system. Taking these two limitations into account, a new dataset for OSR and FSL for audio data was recently released to promote research on solutions aimed at addressing both limitations. This paper proposes an audio OSR/FSL system divided into three steps: a high-level audio representation, feature embedding using two different autoencoder architectures and a multi-layer perceptron (MLP) trained on latent space representations to detect known classes and reject unwanted ones. An extensive set of experiments is carried out considering multiple combinations of openness factors (OSR condition) and number of shots (FSL condition), showing the validity of the proposed approach and confirming superior performance with respect to a baseline system based on transfer learning.

Keywords: open set recognition; open set classification; audio classification; autoencoders; few-shot learning

1. Introduction

Machine listening is the branch of artificial intelligence that aims to create intelligent systems that are capable of extracting relevant information from audio data. Acoustic event classification (AEC) and acoustic scene classification (ASC) are two areas that have grown significantly in the last years [1–4], often included within the machine listening field. The increase in research proposals related to these areas is motivated by the number of applications that can benefit from automation systems incorporating audio-based solutions, such as home assistants or autonomous driving. This interest is also evidenced by the multiple editions of the successful international DCASE challenge (Detection and Classification of Acoustic Scenes and Events). From its very first edition in 2013 [5], different ASC and AEC tasks have been presented during the past years (2013, 2016, 2017 and 2018). In fact, the 2019 edition incorporated an open-set recognition (OSR) task within the scope of ASC, where the idea was to classify an audio clip to a known scene type or to reject it when it belonged to an unknown scene.

In general terms, OSR is a problem that appears when an intelligent system has to classify (in inference stage) a sample from an unknown class, i.e., a class that has not been seen during training. The complexity of the OSR problem can be quantified by using the openness factor (O^*) presented in [6], which measures the relationship between the number of classes seen during training and the number of classes seen during the inference stage only. The objective of a system that is deployed to face an OSR environment is to classify correctly the samples that belong to classes that have been seen during training, while properly rejecting samples from unknown classes. The most popular solutions aimed at solving OSR problems make use of classic machine learning algorithms such as support vector machines [7] or nearest neighbors [8]. In this context, deep learning solutions are not so common in this problem, showing the need for further investigation in this direction [9,10].

Few-shot learning (FSL) is another phenomenon related to real-world applications that aims to detect a specific pattern or class with little amount of data for training the classification system, i.e., using few examples per class. FSL has been widely investigated in face recognition tasks. However, contributions in the audio domain are not so common and are mostly related to music fraud detection [11] or speaker identification [12,13]. A main feature of FSL has to do with the "intra-class" behavior of coarse categories. As an example, assume that a general class "bell" groups samples from different types of bells. The goal of FSL would be to discern among the different bell types, even if all of them can be categorized into a general "bell" class. Two different approaches can be followed to tackle FSL. On the one hand, the transfer learning (TL) approach [14] tries to solve the problem of having only few samples by using prior knowledge. This prior-knowledge is usually represented by the use of a neural network pre-trained on external data that is employed as a feature extractor [15]. The other approach lies on novel neural network architectures such as Siamese [16,17], facenet (trained with triplets) [18,19] or on classical networks trained with novel loss functions such as ring loss [20] or center loss [21]. The main problem with these networks is that a relatively large amount of data is required to properly generalize FSL tasks, i.e., the need to consider many different classes even if only few samples are available per each class.

Recently, a dataset that takes into account both limitations (OSR and FSL) has been made public by the authors [22]. This dataset is composed by two coarse classes: pattern and unwanted sounds. The pattern sounds class is made up of 24 subclasses. These subclasses correspond to specific patterns of different domestic alarms such as bells or fire alarms. Therefore, all these 24 subclasses can be considered as a coarse, more general, "domestic alarm" class, but providing intra-class differentiation within it. On the other hand, the unwanted sounds are grouped into 10 different subclasses with more general and likely to appear domestic sounds, such as keyboard tapping, cough or music among others. These samples must be rejected by an OSR classification algorithm. All these subclasses, either pattern and unwanted, contain 40 samples. The dataset is provided with different configurations depending on the openness factor or the number of shots. In turn, these configurations are divided into different k-fold configurations depending on the number of training examples to facilitate the analysis of the generalization of the proposed solutions.

This paper proposes a novel deep learning approach to tackle OSR and FSL problems within an AEC context, based on a combined two-stage method. As a first step, an embedded or bottleneck representation from the audio log-Mel spectrogram is obtained by means of an autoencoder architecture. Once the autoencoder is trained, the bottleneck representation is used to train a simple multi-layer perceptron (MLP) classifier with sigmoid activation for OSR classification. The autoencoder part aims at solving the FSL limitation, while the MLP classifier mitigates the OSR problem. Moreover, two autoencoder alternatives are suggested within the considered framework, considering both semi-supervised and unsupervised training. Thus, the contributions of this paper reside on the proposal of a full framework for AEC OSR/FSL tasks, the analysis of this framework in different OSR/FSL conditions (different openness values and number of training samples) and the comparison with the baseline method presented in the dataset release [22], showing significant improvement without the need to use prior knowledge from external data.

The rest of the paper is organized as follows. The required background describing OSR openness and autoencoders can be found in Section 2. The proposed system and its different parts are presented in Section 3. The experimental details, including datasets and parameter configuration are described in Section 4, while the results are discussed in Section 5. Finally, conclusions and future work are summarized in Section 6.

2. Background

This section reviews the background and previous works related to the proposed framework, including FSL, OSR and the use of autoencoders in audio-related tasks.

2.1. Few-Shot Learning

Few-shot learning (FSL) is the problem that appears in machine learning applications when a small amount of data is provided per class. In fact, machine learning techniques have become state-of-the-art solutions in many domains due to the huge data sets available.

The FSL limitation can be addressed in three different strategies according to [23]. The three possible approaches are: modifying the available data (increasing the training data), choosing a particular model with FSL considerations during the training and testing stages or using prior knowledge solutions. In this work, FSL was approached by the creation of a specific model making use of autoencoders. Within the strategy of creating specific models for FSL there are, in turn, different approximations. In our particular case, the use of autoencoders represents a solution based on embedding learning, that is, a model that is capable of discovering important structure within the input data by forcing a reduction of dimensionality. For a complete review of FSL approaches, the reader is referred to [23].

One of the first appearances of an architecture to solve an FSL task with an embedding learning approach was in signature recognition. The proposed architecture is known as the Siamese network [16]. The main feature of a framework based on Siamese networks is that it instances two networks having the same architecture and tied weights, forcing the network to learn the similarities between the two inputs. The purpose of this framework is to train a network that is able to embed the inputs into a domain having lower dimensionality in a smart way. That means that if the two entries are very similar, the embeddings must be similar. Once the network is trained, two entries are passed through the network and a measurement metric is calculated to determine if both entries are in the same class.

Triplet networks appeared as a modification of Siamese networks [18]. In a similar way, the framework is created by instantiating three networks with tied weights. In each step, the network is fed with one example called anchor, one positive and one negative. The positive sample has to belong to the same class as the anchor and the negative one to a different class. In both cases (Siamese or triplet networks), the selection of pairs or triplets is crucial for an efficient training process.

A different approach to address the FSL issue is to modify the network loss function to emphasize the distance between classes in the feature map space during training [24]. Some examples are ring loss [20], center loss [21] or prototypical networks [25]. Ring loss and center loss can be understood as a modified softmax that tries to obtain more discriminative features with a modification during loss calculation. The objective of prototypical networks is to obtain a cluster center for each class. During the inference stage, the classification is carried out by using the distances to each center.

The above solutions have shown promising results in the field of image and computer vision. Note, however, that although there are few samples per class, the datasets are considerably large. For example, in [26], there are about 13,500 examples in total. This number of examples might be enough to train the above kind of solutions. However, as far as this group is concerned, in the audio domain, there are not FSL datasets with such amount of data. As a result, the proposed autoencoder-based approach accommodates better the scenario considered in this work.

2.2. Open-Set Recognition

In realistic scenarios there is usually an incomplete knowledge of all the possible surrounding classes at the time of training, and a trained classifier may face unknown classes during testing. As a result, algorithms need to accurately classify the known classes, but also to deal effectively with the unknown ones. OSR approaches are designed to do both things properly. When dealing with OSR problems, certain considerations should be kept in mind when defining which classes are to be recognized and which should be rejected. The evaluation of OSR systems is based on the concept of openness factor O^* [6], which introduces a categorization on the classes involved in the training and testing stages:

- *Known Known* (KK) classes: classes that are used in the training and validation stage and that must be correctly classified by the system.
- *Known Unknown* (KU) classes: classes that are available during the training stage but must not be categorized into the specific class they belong to. In other words, they must be rejected by the classifier. These classes are very useful since they allow the system to make representations and generate boundaries that can help to discern samples from the unwanted category.
- *Unknown Known* (UK) classes: classes for which no samples are available during training but side-information such as semantic/attribute information is available during training. This category is not considered in this work.
- *Unknown Unknown* (UU) classes: classes that are not used nor in the training nor in the validation stage and must obviously be rejected by the classifier. The system only sees these classes in the test stage.

According to [27], the openness factor is defined as

$$O^* = 1 - \sqrt{\frac{2 \times T_{TR}}{T_{TR} + T_{TE}}}, \tag{1}$$

where T_{TR} corresponds to the total number of classes used in the training stage (either KK or KU) and T_{TE} corresponds to the number of classes used in inference stage. When $O^* = 0$, $T_{TR} = T_{TE}$, meaning that there is no UU class. On the other hand, when T_{TE} becomes larger and $T_{TE} > T_{TR}$, $O^* \rightarrow 1$, leading to a more complex OSR task. Note that, by definition, the openness factor is bounded to the range $0 \leq O^* < 1$.

Different approaches have been taken to address the issue of OSR, either with discriminative models or generative models. Traditional machine learning frameworks have been used as enhanced discriminatory models, such as those based on SVM solutions, including the Weibull-calibrated SVM (W-SVM) [28] or $\mathrm{P_I}$-SVM [29]. Other approaches based on classic techniques are those based on sparse representation (SROSR) [30]. As reported in the original paper, the training set must be large enough to cover the conditions that may be present in the test stage. Distance-based methods with modifications have also been proposed [8,27,31,32].

With regard to deep-learning-based solutions, there is the problem of their original close-set nature. The first approach to create deep neural networks of open-set nature was to replace the commonly used final Softmax layer with an OpenMax layer [9]. Other approaches are the deep open classifier (DOC) [33] or the competitive overcomplete output layer (COOL) [34]. More solutions provided in the context of DNN are discussed in [27].

While all the these approaches have shown to improve classification systems in OSR conditions, they also have their limitations [27]. One of the main problems is that the classifier is not able to understand the whole context when dealing with unknown classes. The framework presented in this paper relies on the latent space distribution learned by autoencoders, which is assumed to compact the information from the training classes into a space that can be more easily handled by a subsequent decision stage. As it will be explained in Section 3, a DNN with sigmoid activation will be used for this task.

2.3. Autoencoders in Audio Processing Tasks

The autoencoder is a machine learning solution made up of two blocks, encoder and decoder, whose purpose is to obtain internal representations usually with smaller dimensionality than the input. This process is known as encoding. For this representation to be obtained, the decoding phase is also necessary so that the system can encode efficiently the input data. The purpose of this block of the autoencoder is to reconstruct the input signal from the intermediate representation obtained by the encoder. The difference between the reconstructed signal by the autoencoder and the original input signal is known as the reconstruction error. In essence, the autoencoder tries to learn an identity function $h(\mathbf{x}) \approx \mathbf{x}$, which makes the output $\hat{\mathbf{x}}$ be similar to the input $\mathbf{x}$. By placing constraints on the network, such a limitation in the number of hidden units, interesting structure about the data can be discovered. Although there are different types of autoencoders (e.g., vanilla multi-layer autoencoders, denoising autoencoders, convolutional autoencoders or variational autoencoders) the underlying fundamental principle is the same. For example, convolutional autoencoders are designed to encode the input into a set of simpler signals and reconstruct the input from them. The encoder layers are in this case convolutional layers and the decoder layers are called deconvolution or upsampling layers.

In the audio domain, autoencoders have become the state-of-the-art solution for speech translation applications [35]. Besides, other tasks such as learning more sophisticated or universal audio representations or anomalous sound detection currently tend to solve their limitations using autoencoders. The following paragraphs describe some previous work in this direction, where autoencoders are used to solve the aforementioned problems.

In [36,37], different autoencoder architectures and approaches were presented to obtain robust audio representations that can be used in a variety of audio tasks. In [36], audio representations are learned by addressing a phase prediction task. The autoencoder in [37] was trained in an unsupervised way using Audioset [38], one of the largest audio datasets. In this case, the autoencoder was implemented with convolutional layers. The experimental work is performed considering small encoder architectures that can be potentially deployed on mobile devices.

Another interesting audio application of autoencoders is anomalous sound detection, which is the task of identifying whether a sound corresponds to a normal (known) or abnormal (unwanted) class [39,40]. The main challenge of this problem is to detect the anomaly having only training samples of normal behavior. The objective can be to detect machine faults only by monitoring the sound produced by these machines. The mean squared error obtained when reconstructing the signal can provide information on whether the sample is normal or abnormal.

The approach presented in this work uses an autoencoder in order to obtain discriminative intra-class audio representations. The use of autoencoders to discriminate unwanted classes has already been suggested in the literature. For example, in [41], a solution to detect known or unwanted scenes is presented. In this method, an autoencoder is trained for each known class and the reconstruction error is used to decide if the class is known or not. In contrast, our proposal considers a single autoencoder trained on all the known classes (KK) and the intermediate layer or bottleneck is used to train a MLP to distinguish unwanted samples.

3. Proposed Approach

This section presents the proposed solution to address the problems of FSL and OSR jointly, which consists of three blocks: a high-level 2D time-frequency audio representation, a smaller dimensional encoding of such representation using an autoencoder and a final MLP classifier aimed at discerning whether the input corresponds to a known class or to an unknown class. The full framework is depicted in Figure 1.

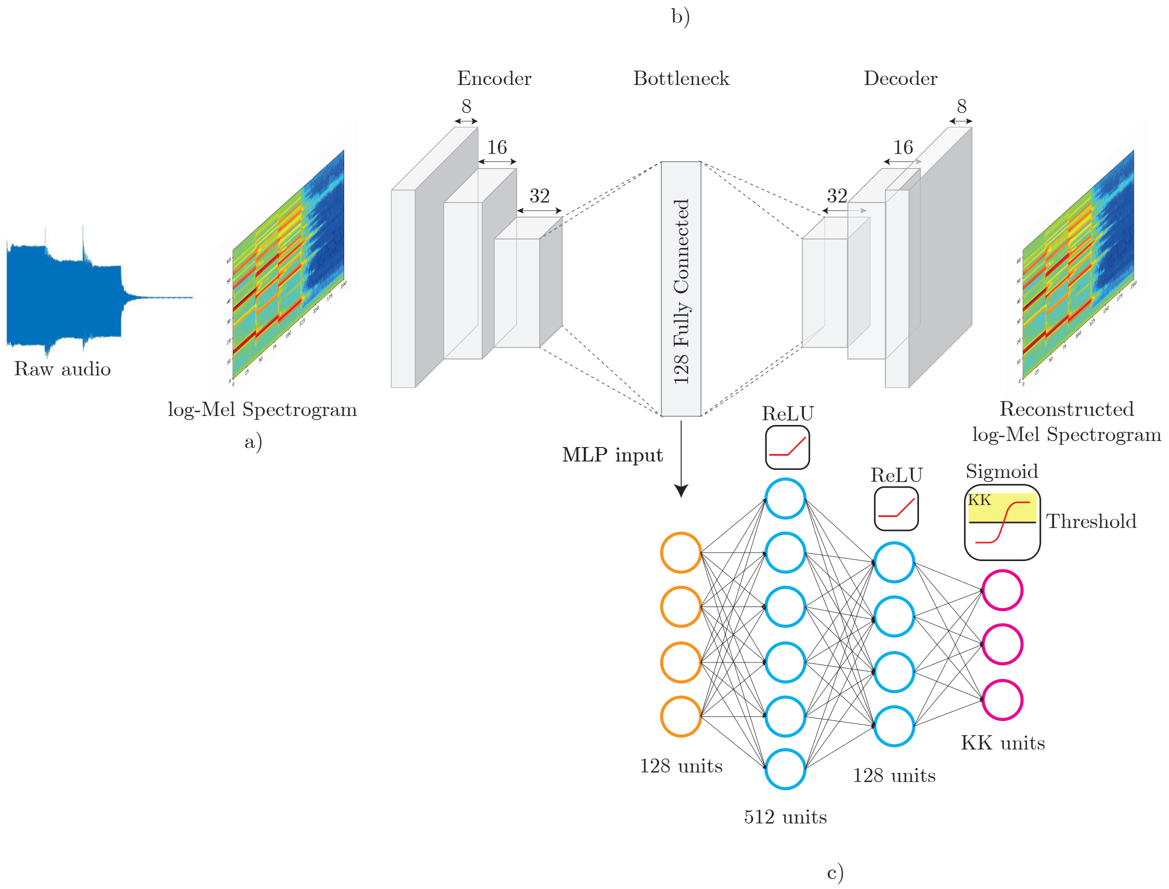


Figure 1. Proposed open-set recognition (OSR)/few-shot learning (FSL) framework for audio classification. In this scheme, an unsupervised autoencoder is considered. (**a**) Log-Mel spectrogram representation. (**b**) Autoencoder. (**c**) multi-layer perceptron (MLP) classifier.

3.1. Input Audio Representation

To facilitate learning from few data, the raw audio input is first transformed into a meaningful time-frequency audio representation. A state-of-the-art choice for many audio processing tasks is the use of log-Mel spectrograms [3,42]. This representation is calculated with a window size of 40 ms and an 50% overlap. The number of Mel filters is set to 64. Each frequency bin is normalized to zero mean and unit standard deviation using all the available training data.

3.2. Convolutional Autoencoder

The proposed system considers the use of a convolutional autoencoder made up of convolutional layers. In this work, a convolutional block is understood to consist of a convolutional layer, a batch normalization layer (BN) and a non-linear activation, in our case rectified linear units (ReLU). This same configuration is stacked again with an increasing number of filters and ends with an average pooling layer $(2,2)$ [43]. Therefore, each convolutional block (ConvBlock) is made up of seven layers (see Figure 2). The decoder follows a symmetric structure with respect to the encoder, that is, the number of filters in each decoding ConvBlock decreases until reaching the last convolutional layer, which has only a single filter and is in charge of obtaining the reconstructed input. Another consideration is that average pooling layers are replaced by upsampling layers. The ConvBlock architecture can be seen in Figure 2. The last convolutional layer is not accompanied by a normalization or activation layer. The only layers with linear activation are the last convolutional layer of the decoder and the bottleneck layer that corresponds to a dense layer. This dense layer acts as a representation of the encoded audio and is made up of 128 neurons. To prevent the autoencoder from learning the identity function, a dropout layer is introduced at the start of the encoder [44,45]. The autoencoder architecture is detailed in Table 1.

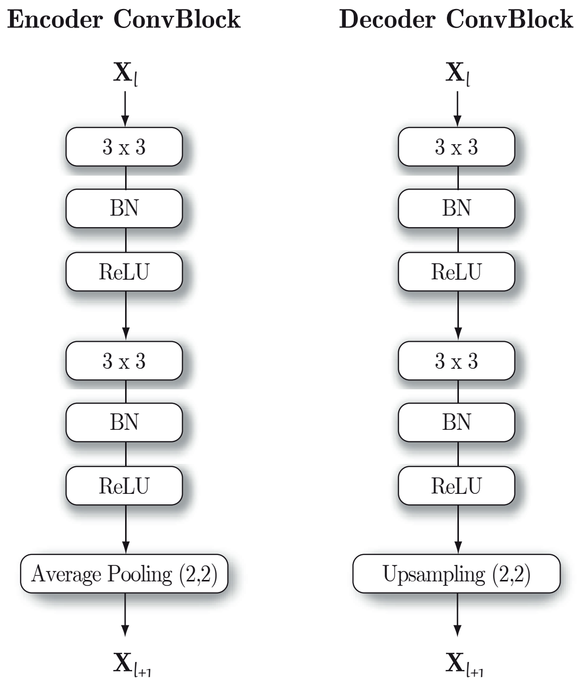


Figure 2. Architecture of the ConvBlocks for the two parts of the autoencoder. $\mathbf{X}_l$ denotes the input to the block, while $\mathbf{X}_{l+1}$ denotes the output.

Table 1. Autoencoder architecture. Values preceded by # correspond to the number of filters and values in parenthesis correspond to kernel size.

Autoencoder Architecture
Dropout(0.1) Enc. ConvBlock(#8, (3, 3)) Enc. ConvBlock(#16,(3, 3)) Enc. ConvBlock(#32,(3, 3)) Flatten
Bottleneck/Dense(128, 'linear')
Upsampling Reshape Dec. ConvBlock(#32,(3, 3)) Dec. ConvBlock(#16,(3, 3)) Dec. ConvBlock(#8, (3, 3)) Conv2D (#1, (3, 3), 'linear')

3.2.1. Unsupervised Autoencoder

Autoencoders originally appeared as a solution to unsupervised problems [46], where no information about the class to which each sample belonged was available. Autoencoders can not be used for supervised classification problems and audio labels are not used in this stage of the training. The objective of the autoencoder is only to extract meaningful internal representations of each audio independently, leading to similar audio representations for samples belonging to the same class. In this case, the loss function during training corresponds to the mean squared error (MSE):

$$\mathcal{L}_{mse} = \frac{1}{N}\sum_{i=1}^{N}(X_i - \hat{X}_i)^2 \quad (2)$$

where X_i corresponds to an original log-Mel spectrogram and $\hat{X}_i$ to the one reconstructed by the autoencoder. Finally, N represents the number of samples in the batch.

3.2.2. Semi-Supervised Autoencoder

In order to mitigate the assumption that samples of the same class have a similar representation and try to achieve more similar representations within the same intra-class, the autoencoder has been modified so that it not only takes into account the reconstruction error but also the classification error. The goal is to force the encoder to approximate representations of the same class in the feature space. Therefore, the bottleneck layer is stacked with a classification layer, a dense layer with the number of neurons equal to the number of KK classes. A block diagram of this architecture can be found in Figure 3. In this case, the total loss is a weighted sum of the reconstruction error (MSE) and the classification error, which in our case is the binary cross-entropy (BCE):

$$\mathcal{L}_{bce} = -\frac{1}{N}\sum_{i=1}^{N}[y_i \log \hat{y}_i + (1 - y_i)\log(1 - \hat{y}_i)] \tag{3}$$

$$\mathcal{L}_{ss} = \frac{1}{2}\mathcal{L}_{mse} + \frac{1}{2}\mathcal{L}_{bce} \tag{4}$$

where y_i corresponds to the label of the original class instance and $\hat{y}_i$ to the predicted label. $\mathcal{L}_{ss}$ is the loss used in the semi-supervised configuration. Each partial loss is multiplied by 1/2 to use a uniform weighting between reconstruction and classification. The choice of these weights is designed so that the framework is purely semi-supervised. If more weight is given to the reconstruction error, the framework will be more likely to appear to be the unsupervised system. On the other hand, if more weight is given to the classification error, the autoencoder will be more likely to address a closed-set classification problem and the open-set consideration will not be properly handled.

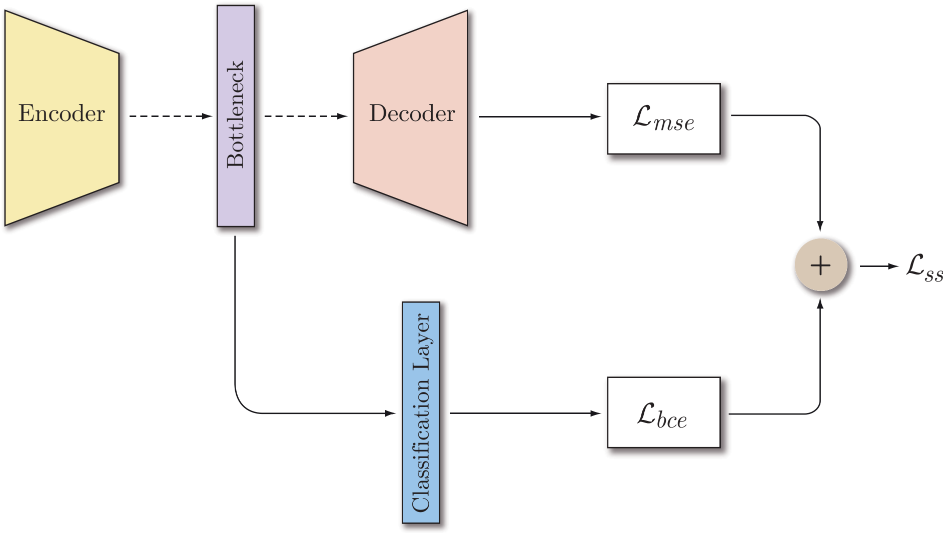


Figure 3. Semi-supervised autoencoder architecture.

3.3. Multi-Layer Perceptron

Once the autoencoder has been trained, a MLP is trained on the learned latent space representations. This block will be in charge of classifying a sample if it belongs to a KK class (pattern category) or rejecting it either if it belongs to KU or UU classes (unwanted category). The MLP is trained with the representations that are obtained in the bottleneck layer of the autoencoder. Each audio sample is represented by 128 features. The MLP consists of three layers. The first two layers have 512 and 128 units respectively with ReLU activation. The last layer has as many units as

KK classes and its activation is a sigmoid function. The output of the sigmoid is used as a likelihood score, so that a threshold is established for deciding if a given audio sample belongs to a KK class. This threshold is set to 0.5. If none of the outputs corresponding to the KK classes is above the threshold, the sample will be rejected. This MLP architecture is inspired in previous works, such as [15] or the baseline method used in [22]. This also allows for emphasizing the contribution of the proposed autoencoders, verifying their validity in a clearer way. In this context, the baseline method proposed in [22] uses audio embeddings obtained from the L3net network [15]. As a result, while the baseline employs transfer learning (the method relies on prior knowledge from a pre-trained network), we only make use of the samples available in the training dataset. Training details will be presented in Section 4.2.

4. Materials and Methods

This section describes the experimental framework considered in this paper, including the dataset used, the FSL/OSR conditions, the performance metrics considered or the network training details. Note that the proposed method is intended to address both the problem of OSR and the FSL problem. State-of-the-art solutions for such learning problems may not be suitable when both problems appear simultaneously. Therefore, special care must be taken when comparing systems not specifically designed to tackle both aspects.

4.1. Dataset

The dataset used to validate the framework proposed in this paper was recently presented in [22]. The dataset contains audio samples of a domestic nature to address FSL audio event recognition in an OSR context. The data set consists of two general classes or coarse categories defined as:

- *Pattern sounds category*: includes all the classes that must be recognized. Samples belonging to one of these classes must be classified as such. In our scenario, this category is made up of KK classes.
- *Unwanted category*: includes all the classes that should not be classified. Samples from these classes must be rejected by the system without labeling them. In this context, the unwanted category consists of the KU and UU classes, depending on the openness configuration.

The pattern category contains 24 classes that correspond to different domestic alarms and the unwanted category contains 10 classes of different nature such as cough, keyboard tapping or door slam among others. The dataset comes prepared for different openness values and different shots during training. As can be seen from Equation (1), the openness condition is affected by the number of KK classes to be classified or the number of unwanted classes used for training. Therefore, two approaches are presented within the dataset. In the first approach, the system is trained to recognize the full set of pattern classes (24 KK classes). As far as the unwanted classes are concerned, the dataset is designed so that the system is trained using all, half or no unwanted classes, leading to the set of openness values $O^* \in \{0, 0.04, 0.09\}$, respectively. Another scenario is the creation of known trios, i.e., only 3 classes of KK are used for training/testing. With this configuration 8 different training trios are created. By repeating the same process with respect to the unwanted ones, the resulting openness values are $0^* \in \{0, 0.13, 0.39\}$ with this configurations. The experiments that make up the configuration $O^* = 0.39$ were not carried out because it was not possible to get a feasible solution with such an openness factor. The number of classes used during the training and inference stages that correspond to the openness values previously explained are specified in Table 2.

On the other hand, all these settings can be trained with different shots. The dataset was pre-configured for 4, 2 or 1-shot training. The number of shots modifies the k-fold cross-validation. For example, when training with 4 shots a 10-fold configuration was used, while when training with 1 shot, a 40-fold configuration was employed.

Table 2. Number of classes of each configuration and the corresponding openness value.

Pattern Sounds	**KK**	**KU**	**UU**	T_{TR}	T_{TE}	O^*
Full set	24	10	0	34	34	0
		5	5	29	34	0.04
		0	10	24	34	0.09
Trios	3	10	0	13	13	0
		5	5	8	13	0.13
		0	10	3	13	0.39

4.2. Training Procedure

The training setup was very similar for the autoencoder and the MLP. The optimizer used was Adam and the batch size was set to 32 samples. The learning rate starts with an initial value of 0.001 and decreases when the validation metric had not improved for 20 epochs by a factor of 0.75. The training was terminated if the validation metric did not improve for 50 epochs. The final selection corresponds to the model that had obtained a better metric in validation. The difference was the maximum number of epochs yet for the autoencoder is 500 epochs and for the MLP is 200 epochs. It must be here considered that when training from the scratch with few samples the system can easily converge to different local minima depending on its initialization. Therefore, each k-fold configuration was also repeated 5 times in order to provide insight about its statistical behavior and robustness.

4.3. Performance Metrics

The metrics used to analyze the performance of the proposed systems are presented in [22] and summarized here for the convenience of the reader. They are based on the weighted global accuracy, ACC_w, which is computed differently depending on the value of openness.

$$\begin{aligned} &O^* = 0 \text{ (without UU)}: \\ &\qquad ACC_w = wACC_{KK} + (1-w)ACC_{KU}, \end{aligned} \tag{5a}$$

$$\begin{aligned} &O^* \neq 0 \text{ (with KU and UU)}: \\ &\qquad ACC_w = wACC_{KK} + (1-w)ACC_{KUU}, \end{aligned} \tag{5b}$$

$$\begin{aligned} &O^* \neq 0 \text{ (with only UU)}: \\ &\qquad ACC_w = wACC_{KK} + (1-w)ACC_{UU}, \end{aligned} \tag{5c}$$

where w represents a factor that weights the accuracy between the accuracy obtained over KK classes and other unwanted classes (KU or UU). ACC_{KK} corresponds to the accuracy with which the system correctly classifies the KK samples into their respective classes. In this case, to accept a sample as a valid KK class, the OSR threshold (0.5) must be exceeded.

The performance metrics involving the unwanted category (ACC_{KU}, ACC_{KUU} or ACC_{UU}), indicate the ability of the system to reject samples that do not belong to any KK class. For a sample to be considered unwanted, none of the KK classes must exceed the OSR threshold. The reason why there are three different metrics relates to the different openness conditions O^*. When the system sees all possible unwanted classes during training, the unwanted category only consists of KU classes. On the other extreme, if the system does not see any unwanted samples during training, the unwanted category is only made up of UU classes. When the system sees some of the classes pertaining to the unwanted category, the ACC_{KUU} metric is used, which is defined as the average of ACC_{KU} and ACC_{UU}.

5. Results and Discussion

This section discusses the results obtained for the different FSL/OSR configurations considered in the described dataset. The performance of the two proposed autoencoder-based systems is compared to the one obtained by the dataset baseline system. More detailed information about the number of classes used in each experiment can be seen in Table 2.

5.1. Full Set (24 KK) Performance

All the results of this configuration can be seen in Table 3. In the one-shot case, a substantial improvement over the baseline system for all the openness cases can be observed, with our two proposed approaches achieving a considerably higher ACC_w value. The lowest improvement is of 7 percentage points ($O^* = 0.04$), while the highest is almost of 30 percentage points ($O^* = 0$). With this number of shots, both the ACC_{KK} (in all cases) or the ACC_{UU} (in $O^* = 0.09$) are greatly improved. As observed, the baseline is very likely to classify the pattern sounds (KK classes) into the unwanted category, except when $O^* \neq 0.09$. Using autoencoders, more reliable representations are obtained for the KK classes, resulting in improved accuracy (see ACC_{KK} and $O^* = 0$). Something similar occurs for ACC_{UU}, where autoencoders can get more distant representations within the latent space for unwanted classes. All the metrics remain very similar for the unwanted category when $O^* \in \{0, 0.04\}$.

The behavior is very similar when the number of training samples is 2 or 4. When $O^* \in \{0, 0.04\}$ most metrics are improved to a greater or lesser extent. Only the unsupervised autoencoder shows worse behavior on the ACC_{KUU} metric although it improves ACC_{UU}. However, the most significant contribution of the proposed frameworks can be seen when $O^* = 0.09$. Just like for the one-shot case, ACC_{UU} is clearly enhanced with this framework. In this case, the ACC_{UU} is considerably improved with respect to the baseline, going from 33.3% to 72.5% (semi-supervised) and 26.1% to 69.9% (unsupervised).

Another factor to be analyzed is the standard deviation. That is to say, how the generalization of the solution is affected by the fact that few training samples are available. Depending on the initialization of the network, the results may differ. When a large data set is available, this fact is usually not very decisive. This is not the case in an FSL context. If the standard deviation is analyzed, it must be done taking into account the number of shots and the corresponding openness factor. Analyzing the KK classes, when $O^* = 0$ it can be seen how the framework with the semi-supervised autoencoder has the lowest deviation in all possible cases depending on the number of shots. The unsupervised one has a higher standard deviation than the baseline when the number of shots is 2 or 4. This may be due to the fact that even though it is trained with more samples of the same class, the framework is not aware of it since it does not have such an information. Also, when $O^* \neq 0$ the standard deviation of the unsupervised is higher than the baseline if the number of shots is greater than 1. Probably, the system is becoming more prone to false negatives as the value of openness increases. When the framework does have information about the sample class (semi-supervised architecture), and the number of shots is bigger than one, the standard deviation is reduced. When $O^* > 0$ and the number of shots is higher than one, the semi-supervised architecture has lower standard deviation than the baseline. This does not happen when the number of shots is equal to one since in this case the unsupervised has the lowest deviation. Regarding the unwanted category, it can be observed how deviations increase for all the methods as the value of openness increases. In this case, the framework with the semi-supervised autoencoder shows better results than the unsupervised one except for a single case with ACC_{UU} and $O^* = 0.04$. The reduction in standard deviation is much greater as the number of shots is increased, as seem for ACC_{UU} when $O^* \in \{0.04, 0.09\}$.

Table 3. Final classification results (%). Baseline results correspond to the L3 approach framework presented in [22]. The bold numbers indicate winning configurations according to the number of shots.

Shots	Framework	Openness Coefficient									
		$O^* = 0$			$O^* = 0.04$				$O^* = 0.09$		
		ACC_{KK}	ACC_{KU}	ACC_w	ACC_{KK}	ACC_{KUU}	ACC_{UU}	ACC_w	ACC_{KK}	ACC_{UU}	ACC_w
1	Baseline	13.8 ± 12.9	99.8 ± 1.0	56.8	57.7 ± 8.4	90.4 ± 5.4	84.8 ± 9.8	74.1	60.1 ± 7.8	39.6 ± 13.4	49.9
	Unsupervised	68.8 ± 10.3	95.4 ± 3.4	82.1	**76.0 ± 7.6**	90.1 ± 7.6	87.6 ± 10.9	**83.1**	**78.5 ± 7.5**	70.6 ± 15.5	**74.5**
	Semi-supervised	**73.5 ± 8.8**	97.4 ± 2.8	**85.4**	73.1 ± 10.3	90.2 ± 7.4	86.9 ± 11.1	81.7	77.2 ± 9.7	69.7 ± 10.4	73.5
2	Baseline	81.1 ± 5.5	99.4 ± 0.8	90.3	83.2 ± 4.8	90.2 ± 5.1	82.5 ± 9.6	86.7	83.3 ± 5.6	33.3 ± 11.6	58.3
	Unsupervised	82.4 ± 7.2	94.6 ± 3.2	88.5	86.0 ± 5.9	88.7 ± 6.3	84.4 ± 10.0	87.3	86.3 ± 6.6	59.3 ± 14.7	72.8
	Semi-supervised	**90.2 ± 4.9**	98.6 ± 1.8	**94.4**	**89.9 ± 4.4**	93.4 ± 6.0	90.1 ± 9.5	**91.6**	**90.7 ± 4.5**	72.5 ± 8.6	**81.6**
4	Baseline	94.8 ± 2.2	99.6 ± 0.4	97.2	94.3 ± 2.2	88.3 ± 5.7	79.4 ± 9.5	91.3	94.8 ± 2.4	26.1 ± 10.1	60.5
	Unsupervised	91.8 ± 3.8	94.6 ± 2.3	93.2	92.4 ± 4.4	85.8 ± 7.8	80.2 ± 12.0	89.1	91.1 ± 4.9	51.4 ± 17.1	71.2
	Semi-supervised	**97.7 ± 1.6**	99.7 ± 0.5	**98.7**	**97.7 ± 1.5**	97.0 ± 3.1	95.0 ± 5.7	**97.3**	**97.93 ± 1.3**	69.9 ± 8.7	**83.9**

5.2. Trio Performance

The results obtained with this configuration are presented in Tables 4 and 5. By looking first at the results in Table 4 ($O^* = 0$), it is observed that the baseline is very prone to false negatives, i.e., it tends to reject examples from KK classes, as derived from its low ACC_{KK} value. In contrast, the proposed autoencoder-based approaches improve considerably the performance in all cases, discerning more easily KK classes from the unwanted ones.

Table 4. Results with trios configuration and $O^* = 0$. The number list under the trio number corresponds to the number of patterns that make up that trio. The bold numbers indicate winning configurations according to the number of shots.

Trio	Shots	Framework								
		Baseline			Unsupervised			Semi-Supervised		
		ACC_{KK}	ACC_{KU}	ACC_w	ACC_{KK}	ACC_{KU}	ACC_w	ACC_{KK}	ACC_{KU}	ACC_w
	1	65.1 ± 16.1	99.4 ± 1.1	82.3	97.4 ± 5.4	97.4 ± 2.5	**97.4**	91.5 ± 10.1	97.9 ± 2.2	94.7
0	2	80.2 ± 15.0	99.6 ± 0.5	89.9	98.4 ± 3.3	98.3 ± 1.5	**98.4**	96.2 ± 6.8	98.2 ± 2	97.2
(1, 9, 17)	4	90.1 ± 14.5	99.7 ± 0.4	94.9	99.0 ± 2.2	98.9 ± 1.0	98.9	98.8 ± 3.7	99.4 ± 0.7	**99.1**
	1	68.9 ± 12.9	99.9 ± 0.2	84.4	96.9 ± 6.4	96.2 ± 3.8	**96.6**	94.8 ± 9.3	96.8 ± 3.7	95.8
1	2	84.7 ± 16.5	99.9 ± 0.4	92.3	99.0 ± 1.3	98.5 ± 1.5	96.8	98.4 ± 3.4	98.3 ± 2.1	**98.4**
(10, 12, 19)	4	88.0 ± 15.6	99.9 ± 0.4	93.9	99.5 ± 0.9	98.9 ± 1.2	99.2	99.2 ± 1.8	99.4 ± 1.1	**99.3**
	1	55.5 ± 18.6	99.9 ± 1.0	77.7	92.1 ± 8.5	97.4 ± 3.1	**94.8**	86.4 ± 12.2	98.4 ± 2.9	92.4
2	2	76.1 ± 14.7	99.9 ± 0.1	88.0	95.9 ± 6.7	98.6 ± 1.6	**97.3**	94.8 ± 6.2	99.1 ± 1.3	97.0
(2, 14, 22)	4	83.1 ± 20.7	99.9 ± 0.1	91.5	97.8 ± 2.3	98.8 ± 1	98.3	98.6 ± 2.2	99.8 ± 0.8	**99.2**
	1	53 ± 12.1	99.9 ± 0.4	76.5	90.0 ± 11.4	94.7 ± 3.7	**92.3**	83.5 ± 12.2	95.2 ± 3.6	89.3
3	2	64.6 ± 16.1	99.9 ± 0.3	82.2	93.8 ± 7.2	96.8 ± 2.2	**95.3**	89.8 ± 10.7	96.9 ± 2.4	93.4
(3, 6, 13)	4	77.4 ± 19.0	99.8 ± 0.9	88.6	96.9 ± 5.5	97.9 ± 1.4	97.4	97.1 ± 3.9	99.0 ± 1.1	**98.0**
	1	71.7 ± 15.2	100 ± 0	85.8	91.2 ± 9.3	96.1 ± 3.1	**93.7**	87.3 ± 12.2	96.3 ± 3.2	91.8
4	2	86.8 ± 14.5	100 ± 0	93.4	94.7 ± 6.3	97.8 ± 1.7	**95.3**	93.2 ± 8.4	97.3 ± 2.1	**95.3**
(4, 5, 16)	4	88.1 ± 18.6	99.9 ± 0.6	94.0	96.3 ± 4.8	98.4 ± 1.5	97.4	99.1 ± 1.9	99.2 ± 1.0	**99.1**
	1	76.5 ± 15.2	99.9 ± 0.2	88.2	92.3 ± 9.7	98.0 ± 2.3	95.2	92.4 ± 8.5	98.5 ± 2.6	**95.4**
5	2	85.1 ± 15.4	99.9 ± 0.1	92.5	95.4 ± 6.8	98.8 ± 1.6	97.1	96.0 ± 5.2	99.2 ± 1.2	**97.6**
(18, 21, 23)	4	89.3 ± 16.4	100 ± 0.1	94.6	98.0 ± 2.6	99.3 ± 0.9	98.6	99.0 ± 1.4	99.8 ± 0.3	**99.4**
	1	87.0 ± 13.5	99.7 ± 0.5	93.4	95.3 ± 7.2	97.0 ± 3.1	96.1	94.4 ± 7.7	97.9 ± 2.7	**96.2**
6	2	87.6 ± 16.0	99.6 ± 0.6	93.6	96.2 ± 4.9	97.9 ± 1.9	97.1	97.1 ± 4.3	98.8 ± 1.5	**98.0**
(8, 11, 24)	4	89.9 ± 14.5	99.7 ± 0.5	94.8	99.1 ± 1.9	98.8 ± 1.2	98.9	99.3 ± 1.4	99.6 ± 0.6	**99.4**
	1	66.4 ± 15.7	99.6 ± 0.6	83.0	89.0 ± 9.7	96.3 ± 3.0	**92.7**	77.4 ± 14.2	97.2 ± 3.1	87.3
7	2	82.1 ± 13.7	99.5 ± 0.7	90.8	92.8 ± 6.4	97.8 ± 1.8	**95.3**	86.4 ± 9.2	98.2 ± 1.9	92.3
(7, 15, 20)	4	83.7 ± 15.3	99.5 ± 0.9	91.6	95.5 ± 3.9	98.8 ± 1.1	**97.2**	91.0 ± 7.3	99.1 ± 1.1	95.1

A similar behavior is observed when looking at the results from the first five trios (from trio 0 to 4). The unsupervised autoencoder shows better performance than the semi-supervised autoencoder with few samples in training. When the number of training samples is four, the semi-supervised always shows the best result. This may reflect that using classification error in the autoencoder training may only have a relevant effect for a sufficient number of shots. Trios 5 and 6 show better results with the semi-supervised autoencoder, especially for two and four shots. Finally, trio 7 shows a quite different behavior, since the unsupervised autoencoder provides the best results for any number of shots. This may be due to the closeness in the feature space of classes 7 and 20.

Regarding the analysis of trios with $O^* = 0.13$ in Table 5, we can see that the semi-supervised autoencoder obtains better performance in most cases. In this case, where not all unwanted classes are seen in training, semi-supervision helps to obtain more discriminative representations even with few samples. Note, however, that in this set of experiments, the baseline obtains the best result in some cases, like in trios 4, 6 or in all the shots of the trio 7.

Comparing the results obtained in this section with those for the full set with the 24 KK classes, a similar behavior is observed. When $O^* = 0$ the system is more prone to false negatives, showing lower ACC_{KK} with $O^* = 0$ than with $O^* = 0.13$. On the other hand, ACC_{KU} and ACC_{KUU} show worse performance with $O^* = 0.13$ than with $O^* = 0$.

With respect to the performance concerning unwanted classes, Table 6 presents the analysis of ACC_{KUU} and ACC_{UU} for the different trio configurations. Note that the former takes into account both unwanted classes seen in training and those that are not. The second only corresponds to the accuracy of the unwanted classes that are only seen in the test stage. The OSR system is expected to have good generalization properties if both are similar. Thus, a more realistic behavior would usually result in a slightly lower metric in ACC_{UU}. It is observed that the accuracies are a little lower for autoencoders than for the baseline. Note, however, that this lower performance is significantly compensated by the tradeoff involving better accuracy in KK classes. This means that, although the baseline may have better ability to reject unwanted classes, it is at the expense of rejecting as well pattern sounds. Both the unsupervised and semi-supervised autoencoders show good rejection generalization. Thus, these solutions can be competitive in OSR problems as long as some of the classes to be rejected take part in the training stage.

Table 5. Results with trios configuration and $O^* = 0.13$. The number list under the trio number corresponds to the number of patterns that make up that trio. The bold numbers indicate winning configurations according to the number of shots.

Trio	Shots	**Framework**								
		Baseline			**Unsupervised**			**Semi-Supervised**		
		ACC_{KK}	ACC_{KU}	ACC_w	ACC_{KK}	ACC_{KU}	ACC_w	ACC_{KK}	ACC_{KUU}	ACC_w
0 (1, 9, 17)	1	85.88 ± 13.4	97.7 ± 4.6	91.8	97.7 ± 5.2	93.9 ± 5.7	**95.8**	95.4 ± 7.4	93.9 ± 6.6	94.6
	2	89.2 ± 12.5	99.6 ± 0.5	94.4	99.1 ± 1.8	96.5 ± 3.7	**97.8**	97.7 ± 4.7	96.9 ± 3.7	97.3
	4	97.5 ± 8.1	99.7 ± 0.4	98.6	99.4 ± 1.8	98.1 ± 2.5	98.7	99.2 ± 2.1	98.4 ± 1.9	**98.8**
1 (10, 12, 19)	1	88.8 ± 13.1	98.3 ± 2.8	93.5	98.5 ± 3.7	88.6 ± 9.4	93.6	97.6 ± 6.2	89.7 ± 9.8	**93.7**
	2	89.0 ± 14.5	98.7 ± 2.4	93.8	99.4 ± 1.2	94.7 ± 5.8	97.0	98.6 ± 2.7	95.6 ± 4.7	**97.2**
	4	96.2 ± 9.6	96.7 ± 3.1	96.5	99.5 ± 1.0	97.7 ± 3.0	**99.6**	99.4 ± 1.2	97.7 ± 4.4	98.6
2 (2, 14, 22)	1	78.4 ± 13.4	99.8 ± 0.9	89.1	95.2 ± 7.3	89.6 ± 9.0	92.4	94.1 ± 8.3	94.6 ± 6.4	**94.3**
	2	82.6 ± 13.9	99.8 ± 0.5	91.2	97.6 ± 4.2	93.2 ± 7.1	95.4	97.7 ± 3.8	97.1 ± 5.1	**97.4**
	4	91.9 ± 12.3	99.4 ± 0.9	95.6	98.9 ± 2.0	94.1 ± 6.9	96.5	98.6 ± 3.0	99.1 ± 1.8	**98.9**
3 (3, 6, 13)	1	72.3 ± 13.4	96.2 ± 4.2	84.3	94.2 ± 8.2	86.0 ± 8.7	**90.1**	90.4 ± 10.9	88.7 ± 8.3	89.6
	2	78.37 ± 13.7	95.7 ± 4.6	87.2	96.9 ± 4.3	90.0 ± 7.9	93.5	94.8 ± 7.4	94.0 ± 5.9	**94.4**
	4	90.3 ± 11.4	92.0 ± 3.2	91.1	97.1 ± 3.9	95.2 ± 4.6	96.1	97.0 ± 4.9	96.6 ± 3.6	**96.8**
4 (4, 5, 16)	1	88.5 ± 10.1	99.3 ± 1.3	**93.9**	94.6 ± 7.4	91.3 ± 6.7	92.9	94.2 ± 8.6	89.1 ± 9.9	91.6
	2	93.2 ± 9.2	99.4 ± 1.1	**96.3**	96.2 ± 5.5	95.1 ± 4.8	95.6	97.1 ± 5.5	94.2 ± 5.8	95.7
	4	97.0 ± 9.1	99.0 ± 1.2	98.0	98.2 ± 3.0	96.2 ± 3.8	97.2	99.1 ± 2.4	97.6 ± 2.9	**98.4**
5 (18, 21, 23)	1	87.9 ± 11.8	99.1 ± 1.2	93.5	96.2 ± 5.2	93.2 ± 6.1	94.7	95.9 ± 7.0	95.5 ± 6.2	**95.7**
	2	93.4 ± 7.7	98.8 ± 1.2	96.1	98.0 ± 2.6	95.9 ± 4.0	97.0	98.8 ± 2.0	98.6 ± 2.9	**98.7**
	4	97.2 ± 8.1	98.3 ± 1.2	97.7	98.8 ± 1.9	94.0 ± 5.9	96.4	99.6 ± 1.0	99.7 ± 0.6	**99.6**
6 (8, 11, 24)	1	96.0 ± 7.8	99.3 ± 0.8	**97.6**	96.4 ± 6.6	93.9 ± 5.07	95.1	97.00 ± 5.73	94.23 ± 6.35	95.6
	2	95.8 ± 9.1	99.4 ± 0.7	97.6	98.6 ± 3.1	96.0 ± 3.9	97.3	99.0 ± 2.7	97.2 ± 3.9	**98.1**
	4	96.8 ± 9.2	99.2 ± 0.8	98.0	99.5 ± 1.0	95.9 ± 4.4	97.7	99.5 ± 1.2	98.7 ± 2.1	**99.1**
7 (7, 15, 20)	1	87.0 ± 11.4	97.6 ± 2.9	**92.3**	91.1 ± 9.2	87.3 ± 7.8	89.2	84.9 ± 12.4	87.3 ± 8.9	86.1
	2	90.0 ± 9.8	98.6 ± 1.7	**94.3**	94.8 ± 6.0	92.4 ± 5.6	93.6	89.2 ± 9.4	93.0 ± 6.2	91.1
	4	94.4 ± 10.1	98.5 ± 1.5	**96.5**	96.0 ± 4.2	95.2 ± 4.2	95.6	93.8 ± 5.8	95.8 ± 4.2	94.8

Table 6. Results of unwanted category with trios configuration and $O^* = 0.13$. The number list under the trio number corresponds to the number of patterns that make up that trio.

Trio	Shots	Framework					
		Baseline		Unsupervised		Semi-Supervised	
		ACC_{KUU}	ACC_{UU}	ACC_{KUU}	ACC_{UU}	ACC_{KUU}	ACC_{UU}
0 (1, 9, 17)	1	97.7 ± 4.6	98.4 ± 4.1	93.9 ± 5.7	93.7 ± 8.8	93.9 ± 6.6	93.0 ± 9.2
	2	99.6 ± 0.5	99.8 ± 0.6	96.5 ± 3.7	96.2 ± 5.6	96.9 ± 3.7	96.2 ± 6.5
	4	99.7 ± 0.4	99.9 ± 0.4	98.1 ± 2.5	98.2 ± 3.7	98.4 ± 1.9	98.5 ± 2.8
1 (10, 12, 19)	1	98.3 ± 2.8	96.8 ± 5.6	88.6 ± 9.4	87.0 ± 12.5	89.7 ± 9.8	87.0 ± 13.5
	2	98.7 ± 2.4	97.6 ± 4.7	94.7 ± 5.9	93.4 ± 8.9	95.6 ± 4.7	94.4 ± 7.1
	4	96.7 ± 3.1	93.8 ± 5.8	97.7 ± 3.0	96.8 ± 4.6	97.7 ± 4.4	96.6 ± 6.8
2 (2, 14, 22)	1	99.8 ± 0.9	99.7 ± 1.7	89.6 ± 8.9	85.3 ± 13.9	94.6 ± 6.4	92.2 ± 9.7
	2	99.8 ± 0.5	99.7 ± 0.6	93.2 ± 7.1	89.2 ± 11.7	97.1 ± 5.1	95.9 ± 7.7
	4	99.4 ± 0.9	99.0 ± 1.5	94.1 ± 6.9	90.2 ± 12.3	99.1 ± 1.8	98.7 ± 3.3
3 (3, 6, 13)	1	96.2 ± 4.2	92.7 ± 8.2	86.0 ± 8.7	84.7 ± 13.4	88.7 ± 8.3	87.5 ± 12.6
	2	95.7 ± 4.6	91.6 ± 8.7	90.0 ± 7.9	87.5 ± 13.3	94.0 ± 5.9	93.3 ± 9.5
	4	92.0 ± 3.2	84.8 ± 6.0	95.2 ± 4.6	93.5 ± 7.3	96.6 ± 3.6	96.0 ± 5.7
4 (4, 5, 16)	1	99.3 ± 1.3	98.6 ± 2.5	91.3 ± 6.7	90.4 ± 9.5	89.1 ± 9.9	87.0 ± 13.6
	2	99.4 ± 1.1	98.8 ± 2.2	95.1 ± 4.8	94.4 ± 7.1	94.2 ± 5.8	92.5 ± 9.8
	4	99.0 ± 1.2	98.1 ± 2.2	96.2 ± 3.8	94.6 ± 6.6	97.7 ± 2.9	97.2 ± 4.7
5 (18, 21, 23)	1	99.1 ± 1.2	98.5 ± 2.2	93.2 ± 6.1	90.0 ± 9.7	95.5 ± 6.3	94.0 ± 9.0
	2	98.8 ± 1.2	97.8 ± 2.3	95.9 ± 4.0	93.5 ± 6.9	98.6 ± 2.9	97.8 ± 5.1
	4	98.3 ± 1.2	96.8 ± 2.1	94.0 ± 5.9	98.4 ± 10.9	99.7 ± 0.6	99.4 ± 1.2
6 (8, 11, 24)	1	99.3 ± 0.8	99.4 ± 0.6	93.9 ± 5.1	92.5 ± 7.9	94.2 ± 6.4	93.4 ± 8.9
	2	99.4 ± 0.7	99.2 ± 1.0	96.0 ± 3.9	94.0 ± 7.0	97.2 ± 3.9	96.4 ± 6.8
	4	99.2 ± 0.8	98.9 ± 1.0	95.9 ± 4.4	93.1 ± 8.0	98.7 ± 2.1	98.2 ± 3.0
7 (7, 15, 20)	1	97.6 ± 2.9	96.8 ± 5.4	87.3 ± 7.8	83.7 ± 12.1	87.3 ± 8.9	82.6 ± 13.6
	2	98.6 ± 1.7	98.4 ± 3.0	92.4 ± 5.6	89.7 ± 9.3	93.0 ± 6.2	89.6 ± 9.9
	4	98.5 ± 1.5	98.1 ± 2.7	95.2 ± 4.2	93.1 ± 7.6	95.8 ± 4.2	93.4 ± 7.4

5.3. Performance on ASC Task

Finally, to study the generalization capability of the proposed framework to other tasks not related to the detection of specific sound patterns, we consider Task 1C of the DCASE 2019 [47] edition. This is related to ASC in OSR conditions. The aim of the task is to classify a scene among one of the ten known classes or to consider it as unknown (reject the sample) if it does not belong to any of them. The dataset is designed so that unknown samples are available during training. Therefore, in this task, only the Acc_{KK} and Acc_{KU} metrics are provided. The results are shown in Table 7.

Table 7. Results(%) of the proposed frameworks using DCASE (Detection and Classification of Acoustic Scenes and Events) 2019 Task 1C dataset. The baseline results correspond to the one presented by the task organization as a starting point.

Framework	Acc_{KK}	Acc_{KU}	Acc_w
Baseline	54.2	43.1	48.7
Unsupervised	39.3	69.0	54.1
Semi-supervised	53.5	25.8	39.6

As it can be observed, the results for this task are considerably worse than those for the FSL/OSR dataset. This is because even if there are many samples of a certain class, they are not necessarily very similar or follow a certain spectro–temporal pattern. However, the unsupervised system improves the trade-off of the system proposed as a baseline. It improves considerably the detection of unwanted sounds but worsens the classification of known classes. The semi-supervised system obtains practically the same result of the baseline for the known classes but the detection capability of unwanted sounds is lower. Such result is in line with our expectations. When the framework does not have information about the class it is reconstructing, it tends to create independent internal representations that lead to an improvement in the classification of unwanted ones. On the other hand, when it is forced to obtain representations that do take into account the information of the class, the capability of classifying known classes is improved to the detriment of the detection of unwanted classes.

6. Conclusions

This work presented a novel framework capable of classifying audio pattern samples with few data within an open-set recognition context. The proposed system is based on the use of autoencoders to learn latent space representations with few data and a multi-layer perceptron classifier to classify target sound classes and reject unwanted ones. Both unsupervised and semi-supervised autoencoder architectures were considered.

It has been confirmed that, by increasing the number of training samples, a smaller standard deviation and a higher classification accuracy for target classes is obtained, reducing the number of false negatives with respect to the baseline method. In this context, if the number of known known classes is high, the semi-supervised autoencoder seems to perform best. On the other hand, with a small number of known known classes, the autoencoder type has a bigger influence. In this case, the semi-supervised approach usually outperforms the unsupervised one for most openness conditions. Only for zero openness and very few training shots, the unsupervised approach showed increased performance.

Author Contributions: The contribution of each author can be summarized as: "Conceptualization, J.N.-A., S.P.-C. and P.Z.; methodology, J.N.-A.; software, J.N.-A. and S.P.-C.; validation, P.Z., F.A. and M.C.; formal analysis, J.N.-A. and M.C.; investigation, J.N.-A.; resources, J.N.-A.; data curation, S.P.-C.; writing—original draft preparation, J.N.-A. and M.C.; writing—review and editing, F.A. and M.C.; visualization, M.C.; supervision, F.A. and M.C.; project administration, J.N.-A. and M.C.; funding acquisition, J.N.-A., P.Z. and M.C. All authors have read and agreed to the published version of the manuscript.

Funding: This project has received funding from the European Union's Horizon 2020 research and innovation program under grant agreement No 779158. The participation of Javier Naranjo-Alcazar and Pedro Zuccarello in this work is partially supported by Torres Quevedo fellowships DIN2018-009982 and PTQ-17-09106 respectively from the Spanish Ministry of Science, Innovation and Universities. The participation of Maximo Cobos is supported by FEDER and the Spanish Ministry of Science, Innovation and Universities under Grant RTI2018-097045-B-C21.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Piczak, K.J. Environmental sound classification with convolutional neural networks. In Proceedings of the 2015 IEEE 25th International Workshop on Machine Learning for Signal Processing (MLSP), Boston, MA, USA, 17–20 September 2020; IEEE: Piscataway, NJ, USA, 2015; pp. 1–6.
2. Cakır, E.; Heittola, T.; Virtanen, T. Domestic audio tagging with convolutional neural networks. In Proceedings of the IEEE AASP Challenge on Detection and Classification of Acoustic Scenes and Events (DCASE 2016), Budapest, Hungary, 3 September 2016.
3. Valenti, M.; Diment, A.; Parascandolo, G.; Squartini, S.; Virtanen, T. DCASE 2016 acoustic scene classification using convolutional neural networks. In Proceedings of the Workshop on Detection and Classification of Acoustic Scenes and Events, Budapest, Hungary, 3 September 2016; pp. 95–99.
4. Bae, S.H.; Choi, I.; Kim, N.S. Acoustic scene classification using parallel combination of LSTM and CNN. In Proceedings of the Detection and Classification of Acoustic Scenes and Events 2016 Workshop (DCASE2016), Budapest, Hungary, 3 September 2016; pp. 11–15.
5. Stowell, D.; Giannoulis, D.; Benetos, E.; Lagrange, M.; Plumbley, M.D. Detection and classification of acoustic scenes and events. *IEEE Trans. Multimed.* **2015**, *17*, 1733–1746. [CrossRef]

6. Scheirer, W.J.; Jain, L.P.; Boult, T.E. Probability models for open set recognition. *IEEE Trans. Pattern Anal. Mach. Intell.* **2014**, *36*, 2317–2324. [CrossRef] [PubMed]
7. Battaglino, D.; Lepauloux, L.; Evans, N. The open-set problem in acoustic scene classification. In Proceedings of the 2016 IEEE International Workshop on Acoustic Signal Enhancement (IWAENC), Xi'an, China, 14–16 September 2016; IEEE: Piscataway, NJ, USA, 2016; pp. 1–5.
8. Júnior, P.R.M.; de Souza, R.M.; Werneck, R.d.O.; Stein, B.V.; Pazinato, D.V.; de Almeida, W.R.; Penatti, O.A.; Torres, R.d.S.; Rocha, A. Nearest neighbors distance ratio open-set classifier. *Mach. Learn.* **2017**, *106*, 359–386. [CrossRef]
9. Bendale, A.; Boult, T.E. Towards open set deep networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 1563–1572.
10. Rakowski, A.; Kosmider, M. Frequency-Aware CNN for Open Set Acoustic Scene Classification; In Proceedings of the Detection and Classification of Acoustic Scenes and Events 2019 Workshop (DCASE2019), New York, NY, USA, 25–26 October 2019.
11. Lu, R.; Wu, K.; Duan, Z.; Zhang, C. Deep ranking: Triplet MatchNet for music metric learning. In Proceedings of the 2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), New Orleans, LA, USA, 5–9 March 2017; IEEE: Piscataway, NJ, USA, 2017; pp. 121–125.
12. Chen, K.; Salman, A. Extracting speaker-specific information with a regularized siamese deep network. In Proceedings of the Advances in Neural Information Processing Systems (NIPS 2011), Granada, Spain, 12–17 December 2011; pp. 298–306.
13. Bredin, H. Tristounet: Triplet loss for speaker turn embedding. In Proceedings of the 2017 IEEE International Conference on Acoustics, Speech and Signal processing (ICASSP), New Orleans, LA, USA, 5–9 March 2017; IEEE: Piscataway, NJ, USA, 2017; pp. 5430–5434.
14. Pan, S.J.; Yang, Q. A survey on transfer learning. *IEEE Trans. Knowl. Data Eng.* **2009**, *22*, 1345–1359. [CrossRef]
15. Cramer, J.; Wu, H.H.; Salamon, J.; Bello, J.P. Look, Listen, and Learn More: Design Choices for Deep Audio Embeddings. In Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Minneapolis, MN, USA, 27–30 April 2019; IEEE: Piscataway, NJ, USA, 2019; pp. 3852–3856.
16. Bromley, J.; Guyon, I.; LeCun, Y.; Säckinger, E.; Shah, R. Signature verification using a "siamese" time delay neural network. In Proceedings of the Advances in Neural Information Processing Systems, Denver, CO, USA, 28 November–1 December 1994; pp. 737–744.
17. Melekhov, I.; Kannala, J.; Rahtu, E. Siamese network features for image matching. In Proceedings of the 2016 23rd International Conference on Pattern Recognition (ICPR), Cancun, Mexico, 4–8 December 2016; IEEE: Piscataway, NJ, USA, 2016; pp. 378–383.
18. Schroff, F.; Kalenichenko, D.; Philbin, J. Facenet: A unified embedding for face recognition and clustering. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 815–823.
19. Hoffer, E.; Ailon, N. Deep metric learning using triplet network. In *International Workshop on Similarity-Based Pattern Recognition*; Springer: Berlin/Heidelberg, Germany, 2015; pp. 84–92.
20. Zheng, Y.; Pal, D.K.; Savvides, M. Ring loss: Convex feature normalization for face recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 5089–5097.
21. Wen, Y.; Zhang, K.; Li, Z.; Qiao, Y. A discriminative feature learning approach for deep face recognition. In *European Conference On Computer Vision*; Springer: Berlin/Heidelberg, Germany, 2016; pp. 499–515.
22. Naranjo-Alcazar, J.; Perez-Castanos, S.; Zuccarrello, P.; Cobos, M. An Open-set Recognition and Few-Shot Learning Dataset for Audio Event Classification in Domestic Environments. *arXiv* **2020**, arXiv:2002.11561.
23. Wang, Y.; Yao, Q.; Kwok, J.; Ni, L.M. Generalizing from a few examples: A survey on few-shot learning. *arXiv* **2019**, arXiv: 1904.05046.
24. Masi, I.; Wu, Y.; Hassner, T.; Natarajan, P. Deep face recognition: A survey. In Proceedings of the 2018 31st SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI), Paraná, Brazil, 29 October–1 November 2018; IEEE: Piscataway, NJ, USA, 2018; pp. 471–478.
25. Snell, J.; Swersky, K.; Zemel, R. Prototypical networks for few-shot learning. In Proceedings of the Advances in Neural Information Processing Systems, Long Beach, CA, USA, 4–9 December 2017; pp. 4077–4087.

26. Huang, G.B.; Mattar, M.; Berg, T.; Learned-Miller, E. Labeled Faces in the Wild: A Database forStudying Face Recognition in Unconstrained Environments. In Proceedings of the Workshop on Faces in 'Real-Life' Images: Detection, Alignment, and Recognition, Erik Learned-Miller and Andras Ferencz and Frédéric Jurie, Marseille, France, 12–18 October 2008.
27. Geng, C.; Huang, S.j.; Chen, S. Recent advances in open set recognition: A survey. *IEEE Trans. Pattern Anal. Mach. Intell.* **2020**. [CrossRef] [PubMed]
28. Kotz, S.; Nadarajah, S. *Extreme Value Distributions: Theory and Applications*; World Scientific: Singapore, 2000.
29. Jain, L.P.; Scheirer, W.J.; Boult, T.E. Multi-class open set recognition using probability of inclusion. In *European Conference on Computer Vision*; Springer: Berlin/Heidelberg, Germany, 2014; pp. 393–409.
30. Zhang, H.; Patel, V.M. Sparse representation-based open set recognition. *IEEE Trans. Pattern Anal. Mach. Intell.* **2016**, *39*, 1690–1696. [CrossRef] [PubMed]
31. Bendale, A.; Boult, T. Towards open world recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 1893–1902.
32. Mensink, T.; Verbeek, J.; Perronnin, F.; Csurka, G. Distance-based image classification: Generalizing to new classes at near-zero cost. *IEEE Trans. Pattern Anal. Mach. Intell.* **2013**, *35*, 2624–2637. [CrossRef] [PubMed]
33. Shu, L.; Xu, H.; Liu, B. Doc: Deep open classification of text documents. *arXiv* **2017**, arXiv:1709.08716.
34. Kardan, N.; Stanley, K.O. Mitigating fooling with competitive overcomplete output layer neural networks. In Proceedings of the 2017 International Joint Conference on Neural Networks (IJCNN), Anchorage, AK, USA, 14–19 May 2017; pp. 518–525.
35. Chung, Y.A.; Wu, C.C.; Shen, C.H.; Lee, H.Y.; Lee, L.S. Audio word2vec: Unsupervised learning of audio segment representations using sequence-to-sequence autoencoder. *arXiv* **2016**, arXiv:1603.00982.
36. Tagliasacchi, M.; Gfeller, B.; Quitry, F.d.C.; Roblek, D. Self-supervised audio representation learning for mobile devices. *arXiv* **2019**, arXiv:1905.11796.
37. Quitry, F.d.C.; Tagliasacchi, M.; Roblek, D. Learning audio representations via phase prediction. *arXiv* **2019**, arXiv:1910.11910.
38. Gemmeke, J.F.; Ellis, D.P.W.; Freedman, D.; Jansen, A.; Lawrence, W.; Moore, R.C.; Plakal, M.; Ritter, M. Audio Set: An ontology and human-labeled dataset for audio events. In Proceedings of the IEEE ICASSP 2017, New Orleans, LA, USA, 5–9 March 2017.
39. Koizumi, Y.; Saito, S.; Uematsu, H.; Harada, N.; Imoto, K. ToyADMOS: A Dataset of Miniature-machine Operating Sounds for Anomalous Sound Detection. In Proceedings of the IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA), New Paltz, NY, USA, 20–23 October 2019; pp. 308–312.
40. Purohit, H.; Tanabe, R.; Ichige, T.; Endo, T.; Nikaido, Y.; Suefusa, K.; Kawaguchi, Y. MIMII Dataset: Sound Dataset for Malfunctioning Industrial Machine Investigation and Inspection. In Proceedings of the Detection and Classification of Acoustic Scenes and Events 2019 Workshop (DCASE2019), New York, NY, USA, 25–26 October 2019; pp. 209–213.
41. Wilkinghoff, K.; Kurth, F. Open-Set Acoustic Scene Classification with Deep Convolutional Autoencoders. In Proceedings of the Detection and Classification of Acoustic Scenes and Events 2019, New York, NY, USA, 25–26 October 2019.
42. Cakir, E.; Heittola, T.; Huttunen, H.; Virtanen, T. Polyphonic sound event detection using multi label deep neural networks. In Proceedings of the 2015 International Joint Conference on Neural Networks (IJCNN), Killarney, Ireland, 12–17 July 2015; pp. 1–7.
43. Simonyan, K.; Zisserman, A. Very deep convolutional networks for large-scale image recognition. *arXiv* **2014**, arXiv:1409.1556.
44. Xu, Y.; Huang, Q.; Wang, W.; Foster, P.; Sigtia, S.; Jackson, P.J.; Plumbley, M.D.; Xu, Y.; Huang, Q.; Wang, W.; et al. Unsupervised feature learning based on deep models for environmental audio tagging. *IEEE/ACM Trans. Audio, Speech Lang. Process.* **2017**, *25*, 1230–1241. [CrossRef]
45. Vincent, P.; Larochelle, H.; Bengio, Y.; Manzagol, P.A. Extracting and composing robust features with denoising autoencoders. In Proceedings of the 25th International Conference on Machine Learning, Helsinki, Finland, 5–9 July 2008; pp. 1096–1103.

46. Baldi, P. Autoencoders, unsupervised learning, and deep architectures. In Proceedings of the ICML Workshop on Unsupervised and Transfer Learning, Bellevue, Washington, DC, USA, 2 July 2011; pp. 37–49.
47. Mesaros, A.; Heittola, T.; Virtanen, T. A multi-device dataset for urban acoustic scene classification. In Proceedings of the Detection and Classification of Acoustic Scenes and Events 2018 Workshop (DCASE2018), Surrey, UK, 19–20 November 2018; pp. 9–13.

C. A Comparative Analysis of Residual Block Alternatives for End-to-End Audio Classification

IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received October 2, 2020, accepted October 13, 2020, date of publication October 16, 2020, date of current version October 28, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3031685

A Comparative Analysis of Residual Block Alternatives for End-to-End Audio Classification

JAVIER NARANJO-ALCAZAR[1,2], **(Graduate Student Member, IEEE),**
SERGI PEREZ-CASTANOS[1], **IRENE MARTÍN-MORATÓ**[3], **(Graduate Student Member, IEEE),**
PEDRO ZUCCARELLO[1], **FRANCESC J. FERRI**[2], **(Senior Member, IEEE),**
AND MAXIMO COBOS[2], **(Senior Member, IEEE)**
[1]Visualfy, 46181 Benisano, Spain
[2]Computer Science Department, Universitat de Valencia, 46100 Burjassot, Spain
[3]Computing Sciences, Tampere University, 33720 Tampere, Finland

Corresponding author: Javier Naranjo-Alcazar (janal2@alumni.uv.es)

This work was supported by the European Union's Horizon 2020 Research and Innovation Program under Grant 779158. The work of Javier Naranjo-Alcazar was supported in part by the Spanish Ministry of Science, Innovation, and Universities through the Torres Quevedo Fellowships under Grant DIN2018-009982. The work of Irene Martín-Morató was supported by the Spanish Ministry of Science, Innovation, and Universities under Grant FPU14/06329. The work of Pedro Zuccarello was supported in part by the Spanish Ministry of Science, Innovation, and Universities through the Torres Quevedo Fellowships under Grant PTQ-17-09106. The work of Francesc J. Ferri and Maximo Cobos was supported in part by the ERDF, in part by the Spanish Ministry of Science, Innovation, and Universities under Grant RTI2018-097045-B-C21, and in part by the Generalitat Valenciana under Grant AICO/2020/154 and Grant AEST/2020/012.

ABSTRACT Residual learning is known for being a learning framework that facilitates the training of very deep neural networks. Residual blocks or units are made up of a set of stacked layers, where the inputs are added back to their outputs with the aim of creating identity mappings. In practice, such identity mappings are accomplished by means of the so-called skip or shortcut connections. However, multiple implementation alternatives arise with respect to where such skip connections are applied within the set of stacked layers making up a residual block. While residual networks for image classification using convolutional neural networks (CNNs) have been widely discussed in the literature, their adoption for 1D end-to-end architectures is still scarce in the audio domain. Thus, the suitability of different residual block designs for raw audio classification is partly unknown. The purpose of this article is to compare, analyze and discuss the performance of several residual block implementations, the most commonly used in image classification problems, within a state-of-the-art CNN-based architecture for end-to-end audio classification using raw audio waveforms. Deep and careful statistical analyses over six different residual block alternatives are conducted, considering two well-known datasets and common input normalization choices. The results show that, while some significant differences in performance are observed among architectures using different residual block designs, the selection of the most suitable residual block can be highly dependent on the input data.

INDEX TERMS Audio classification, convolutional neural networks, residual learning, urbansound8k, ESC.

I. INTRODUCTION

Audio event classification (AEC) is the problem of categorizing an audio sequence into exclusive classes [1]–[3]. Basically, AEC is aimed at recognizing and understanding the acoustic environment based on sound information. This is usually treated as a supervised learning problem where a set of labels (such as siren, dog barking, etc.) describe the content of the different sound clips. In contrast to classical schemes based on feature extraction followed by classification, Deep Neural Networks (DNNs) [4] reduce these steps by working both as feature extractors and classifiers. Among the many different deep learning techniques, the ones based on Convolutional Neural Networks (CNNs) have shown very successful results in areas such as image classification and object detection [5]–[8]. CNNs are able to learn spatial or time invariant features from pixels (i.e. images) or from time-domain waveforms (i.e. audio signals). Several convolutional layers can be stacked to get different levels of representation of the input signal. As a result, CNNs have

The associate editor coordinating the review of this manuscript and approving it for publication was Tallha Akram.

been proposed to tackle audio related problems such as sound event detection or audio tagging [9]–[11].

Although audio signals are natively one-dimensional sequences, most state-of-the-art approaches to audio classification based on CNNs use a two-dimensional (2D) input [12], [13]. Usually, these 2D inputs computed from the audio signal are well-known time-frequency representations such as Mel-spectrograms [14]–[17] or the output of constant-Q transform [18] (CQT) filterbanks, among others. Time-frequency 2D audio representations are able to accurately extract acoustically meaningful patterns but require a set of parameters to be specified, such as the window type and length, hop size or the number of frequency bins. The choice of these hyperparameters can lead to different optimal settings depending on the particular problem being treated or the particular type of input signals [19]. In order to overcome these problems and providing an end-to-end solution, other approaches have proposed the use of 1D convolutions using the raw audio signals as input. Recent works have shown satisfactory results in this direction [20]–[28].

This article is focused on the analysis of the performance of a particular CNN architecture, called Residual Network (ResNet), fed with 1D audio data. The ResNet architecture was first introduced in [29] with the purpose of dealing with the vanishing gradient issue. The core idea of ResNet is to introduce the so-called *identity weight shortcut connection* that skips one or more layers and adds the input of such layers to their stacked output. After the first residual unit was presented in [29], an exhaustive analysis of different variations of such a configuration was done for CNNs with 2D input signals to tackle the image classification problem [30]. Nevertheless, although other works have studied the contribution of residual blocks in the context of 1D raw audio input waveforms [28], [31], a comprehensive analysis of how different residual block designs may affect the overall performance of audio recognition systems has not been provided so far.

The main objective of this work is to analyze the influence on the performance of different residual block alternatives, the ones more commonly used in the image domain, within the context of 1D raw audio classification. To this end, a baseline architecture is slightly modified considering six different residual block implementations that have been shown to lead to satisfactory results in image classification problems. These blocks provide a varying scheme with regard to where identity mappings are created within the set of stacked layers that conform the block. The common baseline architecture is the one presented in [20], which proposed a 1D CNN for raw audio waveform classification using the public dataset UrbanSound8k.[1] For the sake of consistency, the same dataset will be considered in this work. Additionally, the public dataset ESC-50[2] (concretely, the ESC-10 subset) is also used in the experimentation to evaluate the potential differences arising over different datasets. The results suggest that the best performing blocks in the image domain are not the ones showing significant advantages in performance for raw audio classification [30], nor the one originally suggested in [20] for audio data using the baseline architecture.

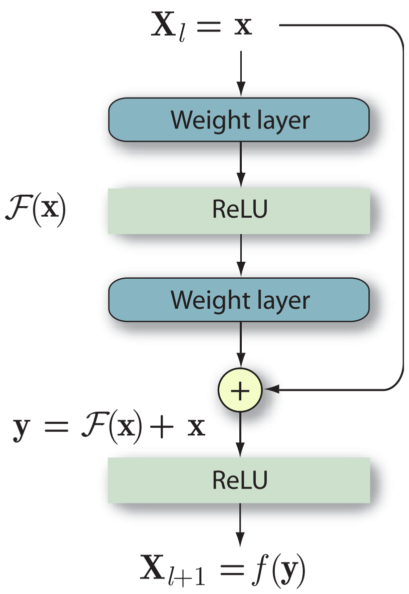


FIGURE 1. Originally proposed residual block or unit [29].

II. BACKGROUND

Residual neural networks -or ResNets- can be understood as modular networks whose building blocks are the so-called residual units or blocks. These residual blocks (RB) are usually characterized by two or three convolutional layers and a shortcut connection that guarantees residual learning during the network training process. The original residual block proposed in [29] is shown in Fig. 1. Consider $\mathcal{H}(\mathbf{x})$ an underlying mapping to be fit by a set of stacked layers in a particular network module, where $\mathbf{x}$ is the input to the first of such layers. Residual blocks are designed to let such layers approximate a residual function, $\mathcal{F}(\mathbf{x}) := \mathcal{H}(\mathbf{x}) - \mathbf{x}$, which means that the original function can be expressed as $\mathcal{H}(\mathbf{x}) = \mathcal{F}(\mathbf{x})+\mathbf{x}$. Similar to predictive coding, the motivation of using residual blocks comes from the intuition that it may be easier to optimize the above residual mapping than to optimize the original, unreferenced mapping. A straightforward way of implementing residual learning is by adding shortcut connections performing identity mappings. In such connections, the input to the set of layers $\mathbf{x}$ is added back to their output, so that $\mathbf{y} = \mathbf{x} + \mathcal{F}(\mathbf{x})$. The function $\mathcal{F}(\mathbf{x})$ represents the residual to be learned by a set of stacked layers of the CNN, where the weight layers are convolutional. In the original residual block, Rectified Linear Unit (ReLU) activation is applied to the result after each identity mapping, resulting in a final output $f(\mathbf{y})$ that acts as input to the next residual block, where $f(\cdot)$ denotes the ReLU function. Thus, in general, the input to the l-th block, $\mathbf{X}_l$, is the output from the previous block and its output becomes the input to the next one, $\mathbf{X}_{l+1}$. Note that shortcut connections do not add extra

[1] https://urbansounddataset.weebly.com/urbansound8k.html

[2] https://github.com/karoldvl/ESC-50

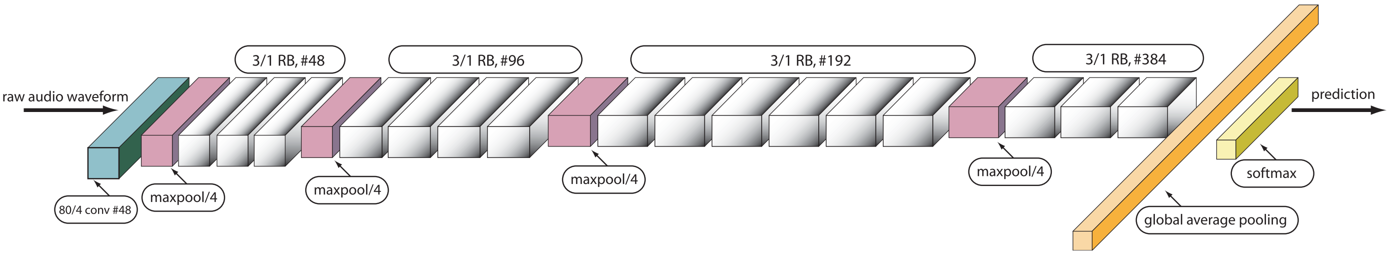


FIGURE 2. Network analyzed [20]. The architecture is explained as follows: [80/4, #48] denotes a layer with 48 filters, 80 of kernel size and stride equal to 4. RB blocks are indicated with kernel size, stride and number of filters.

parameters nor additional computational cost. Thus, deeper networks can be trained with little additional effort, substantially reducing vanishing-gradient problems. However, CNNs often include Batch Normalization (BN) layers and vary slightly with regard to where the activation function is applied. Therefore, the performance of residual learning may also depend both on the order followed by these layers and on the selected point at which shortcut connections are established. In [30], a careful discussion on identity mappings is provided, proposing the use of pre-activated residual units where f is an identity mapping, i.e. $\mathbf{X}_{l+1} = \mathbf{y}_l$. Such slight modification is shown to benefit the training process and to achieve better results in image recognition tasks. However, such analysis has only been performed for 2D architectures and, to the best of the authors' knowledge, a similar study analyzing residual blocks in 1D CNNs has not been addressed so far.

A. RELATED WORK

The use of residual networks for audio-related tasks has already been explored in the literature, usually taking as input frame-level features such as the outputs from mel-scale or logarithmic filterbanks [32]–[34]. As in the present work, several variants of a CNN-based audio classification system accepting raw audio waveforms as input was proposed in [20], including a particular residual architecture. Similarly, end-to-end audio classification systems using residual networks were covered by Kim *et al.* in [28], [31], proposing as well the use of squeeze-and-excitation strategies [35] for increased accuracy. Such strategies are aimed at rescaling the convolutional feature maps by learning proper weightings using temporal aggregation (squeeze) and channel-wise recalibration (excitation). The residual blocks presented in these works, named Res$-n$ (purely residual) and ReSE$-n$ (combinining squeeze-and-excitation), considered the original residual design of [29] depicted in Fig. 1. Both works showed that CNN architectures making use of such blocks provided promising results for learning from raw data and analyzed in detail the effect of including squeeze-and-excitation recalibration. However, the influence of the specific residual block design, as considered in [30] for the image domain, has not been covered so far and its effect in 1D raw audio learning is still unclear.

III. NETWORK ARCHITECTURE

The experimentation conducted in this work considers as a baseline the architecture originally proposed in [20] for raw audio waveforms, consisting in a fully-convolutional network intercalating convolutional and pooling layers. Fully-convolutional networks can usually obtain better generalization properties, whereas, fully-connected layers at the end of the network are more prone to suffer from overfitting. In [20], the convolutional layers are configured with small receptive fields, with the exception of the first layer, whose receptive field is bigger in order to emulate a band-pass filter. Therefore, temporal resolution is reduced in the first two layers with large convolution and max pooling strides. After these layers, resolution reduction is complemented by doubling the number of filters in specific layers. Finally, after the last residual unit, global average pooling is applied to reduce each feature into a single value by averaging the activation across the input. To study the behavior of a given residual block (RB), this article focuses on the residual variant proposed in [20] (originally labeled as M34-res), which follows the general architecture shown in Fig. 2.

Six different RB implementation alternatives are analyzed: the original block proposed by He *et al.* [29] plus the other four blocks proposed by the same authors in [30] and the one introduced by Dai *et al.* in [20] (see Fig. 3). In ResNets, convolutional layers are replaced by different RBs. To isolate the effect of these blocks from the rest of parameters of the network, the number of filters, the receptive field size and the number of convolutional layers remain the same as in [20]. The analyzed residual blocks are the following:

- **RB1 [29]**: the input is first convolved and the output of the second convolution is the input of a batch normalization layer. After the addition, ReLU activation is applied.
- **RB2 [30]**: the input is first convolved and no post-processing is done after the second convolution. The only difference with respect to RB1 is that normalization is applied after adding the input and consequently f corresponds to the composition of BN and ReLU.
- **RB3 [30]**: the input is first convolved as in [20] and the activation is performed before the addition.
- **RB4 [30]**: the input is first passed through a ReLU activation layer and then normalized after the second convolution.
- **RB5 [30]**: the input is first normalized and there are no layers after the second convolution as well as after the addition. RB3-5 constitute a family in which there are no layers after the addition and consequently f is exactly the identity. The differences are in the order in the layers

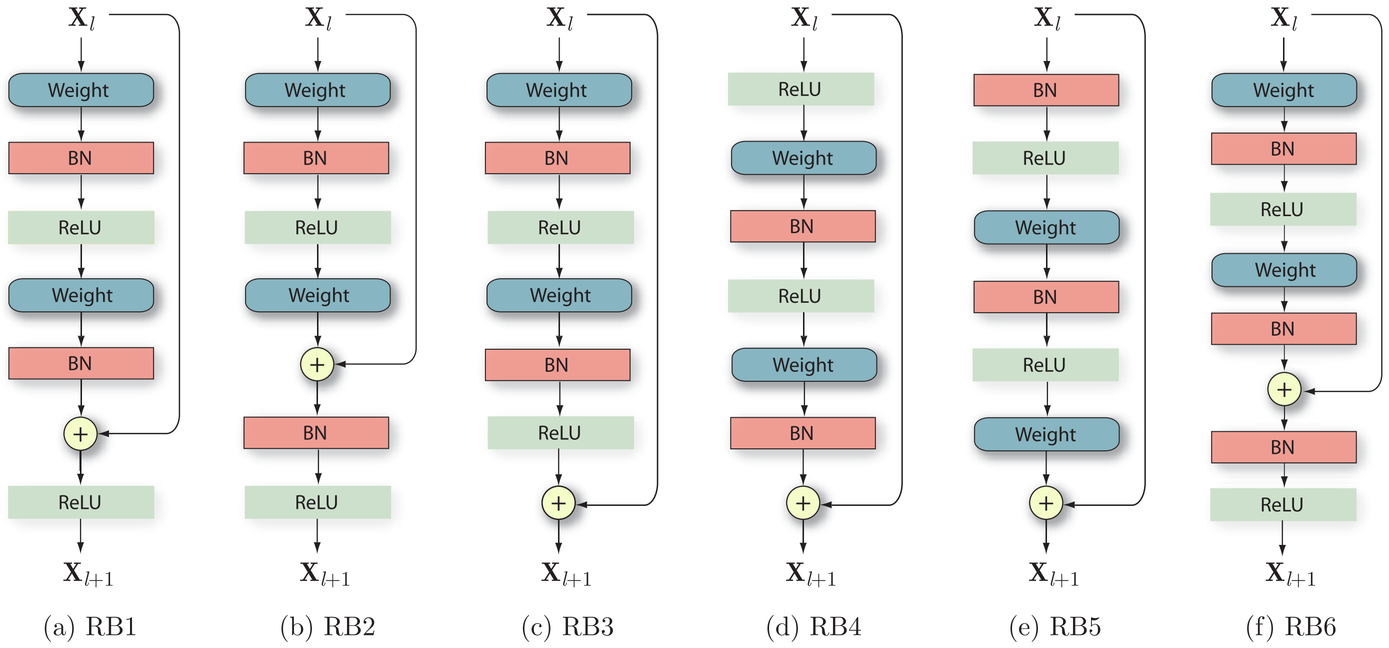


FIGURE 3. Residual units implemented in this work. RB1 to RB5 (a-e) were first introduced in [30], whereas RB6 (f) was presented in [20].

ranging from post-activation (RB3) to pre-activation (RB5).

- **RB6 [20]**: the input is first convolved and the output of the second convolution is the input of a batch normalization layer. After the addition, a new normalization is applied followed by ReLU activation which constitutes a very slight variation of RB2.

The M34-res presented in [20] has 4,001,242 parameters because it uses RB6. When using RB5 the network has 3,988,570 parameters while using RB1-4 the network is composed by 3,989,914 parameters. Dropout layers [36] have not been implemented neither after the pooling layers nor in the residual block, as set out in [20].

IV. EXPERIMENTAL DETAILS

A. DATASETS AND AUDIO PRE-PROCESSING

As in [20], the experimental setup of the present work is based on UrbanSound8k (UBS8k) [37], a public sound-database that contains 8732 sound clips of duration of up to 4 seconds with 10 different classes such as dog barking, car horn, drilling, etc. The dataset is partitioned into 10 different folds and the last one is commonly used as a test while the previous ones are left for training and validation. Additionally, the ESC-10 dataset [38], a public sound-database that contains 400 clips of 5 seconds of duration with 10 different categories (40 samples each category), is also considered. This dataset contains the same number of categories than UBS8k, making the comparison more precise. This dataset is also officially partitioned into different folds (5 in this case).

Clips from both datasets were resampled to 8 kHz and padded with zeros to reach 4 s or 5 s length if necessary after being pre-processed. Once an audio sequence has been read, two different pre-processings have been carried out to check how these can affect the behavior of the final system. The first processing is the scaling of the audio to the maximum absolute value (Scalemax). The second processing consists in normalizing to a signal with zero mean and unit standard deviation (Mean 0 Std 1) as in [20]. As mentioned earlier, padding is done once the signal has been accordingly pre-processed.

B. EXPERIMENTAL SETUP

Instead of using only the last fold of each dataset as a test, a full k-fold cross validation analysis will be carried out in order to obtain more accurate averaged measurements related to the generalization capabilities of the systems under study. The value of k is 10 and 5 for UBS8k and ESC-10, respectively.

Due to the stochastic nature of the experiments and to account for variability, the k-fold cross validation run is repeated a number of times for each dataset (5 and 10 for UBS8k and ESC-10, respectively) so that a total of 50 models are fully trained for each dataset. The final performance measures correspond to the classification accuracy over the whole dataset and averaged over all repetitions along with the corresponding standard deviation.

C. IMPLEMENTATION DETAILS

The optimizer used was Adam [39]. The models were trained with a maximum of 400 epochs. Batch size was set to 128. The learning rate started with a value of 0.001 decreasing with a factor of 0.2 in case of no improvement in the validation accuracy after 15 epochs. The training is early stopped if the validation accuracy does not improve during 50 epochs. The initialization method was glorot-uniform [40] and all weight parameters were subject to L2 regularization with a 0.0001 coefficient as in [20]. Keras with Tensorflow backend was used to implement the models in the experiments. The audio manipulation module used in this work was LibROSA [41].

TABLE 1. Averaged accuracies of the different blocks presented in this article depending on the pre-processing of the audio and dataset used for the experimentation.

Pre-processing	Dataset	RB1	RB2	RB3	RB4	RB5	RB6
Scalemax	UBS8k	**0.68**±0.01	**0.68**±0.00	0.64±0.02	**0.68**±0.00	**0.68**±0.01	**0.68**±0.01
	ESC-10	**0.77**±0.02	0.75±0.01	0.68±0.06	**0.79**±0.02	0.56±0.05	0.74±0.02
Mean 0 Std 1	UBS8k	0.68±0.01	0.68±0.01	0.59±0.03	**0.69**±0.01	0.68±0.01	0.68±0.01
	ESC-10	0.75±0.05	**0.79**±0.01	0.59±0.05	0.75±0.05	0.58±0.04	**0.79**±0.01

V. RESULTS

Given the number of folds and repetitions in the two datasets considered, a total number of 50 independent results are available in each case. With these we have carried out a careful analysis, first comparing averaged accuracies, and second performing a rank-based analysis. Note that the results can not be fairly compared to other previous published results (e.g. [20]) that are more challenge-oriented, but instead the followed procedure allows to compare the different alternatives more accurately.

A. AVERAGED PERFORMANCE ANALYSIS

Averaged rates of accuracy for all the experiments carried out are shown in Table 1 along with standard deviations across repetitions. Best results for each dataset and pre-processing method are marked in bold. We naively assume Gaussianity and perform a parametric multiple comparison test [42] that only discovers significant differences between RB3 and RB5 (shaded in the table) and the remaining options depending on dataset but regardless of pre-processing.

From this first analysis we can hardly observe differences among RBs but it is worth mentioning several surprising facts. First, the RB3 is significantly worse in all cases. Even though this was also the worst in the exact identity family (RB3-5) according to [30], its behavior in the image context was clearly better than that of RB2 which is now among the bests along with its slight variation RB6. Second, the full-preactivation option, RB5, which was the best in the image context is now significantly the worst for ESC-10.

It can be also observed that systems trained on the ESC-10 dataset seem to be more sensitive to the selected input pre-processing. Blocks RB1, RB3 and RB4 show better performance when the audios have been processed with Scalemax. On the other hand, blocks RB2, RB5 and RB6 show better performance when the audios have been normalized to zero mean and unit standard deviation.

Apart form putting forward normalization sensibility and the surprising dependence on data of RB5, the clearest conclusion that we can draw from comparing averaged rates is the very poor behavior of RB3. This could be somewhat expected as RB3 is the only block having a ReLU activation just before the addition leading to a non-negative output which is an unnatural option for a residual function. Note that having a non-negative residual function can have an undesirable impact on learned internal representations, which in turn may substantially affect the robustness and generalization capabilitites of the network.

B. NON-PARAMETRIC RANK-BASED COMPARISON

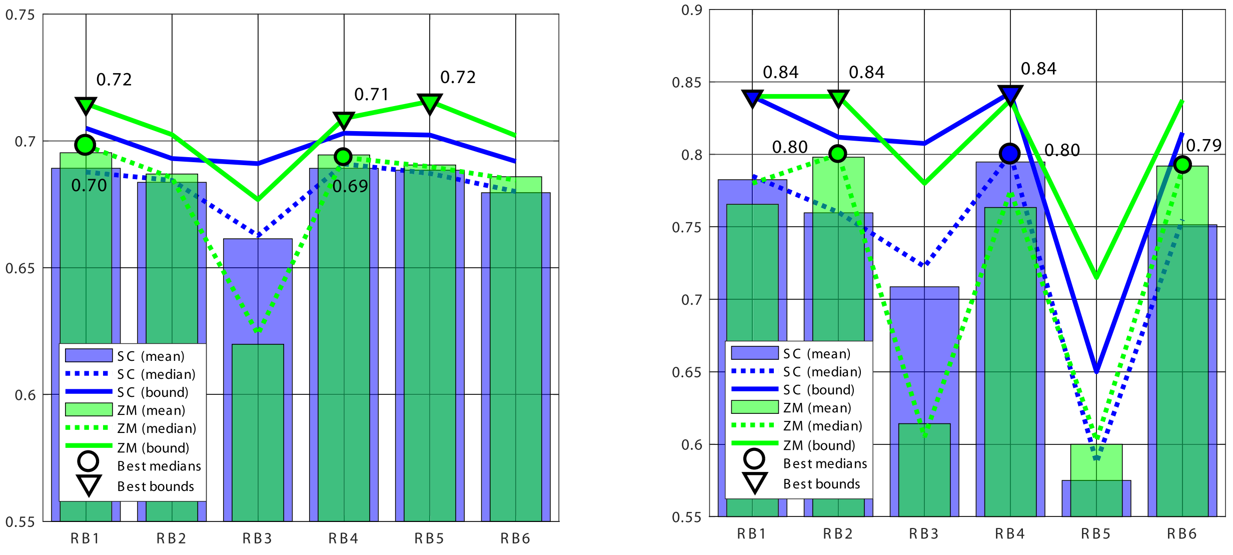
In order to provide more insight about the RB choice, a non-parametric Friedman test with Holm post-hoc has been carried out [43]. Moreover, medians of all repetitions and an optimistic bound obtained by selecting the best model for each fold have been computed and are shown in Fig. 4 along with averaged rates. Table 2 shows the test results including average ranks and corrected p-values. Significance level has been set to $\alpha = 0.05$. The value 0.00 means $p < 0.005$. Apart from the results for each dataset, we also show the ones corresponding to both datasets. Results that are significantly worse than the best according to the selected level appear as shaded in the table, with the corresponding p-values in bold.

The results of the non-parametric analysis confirm the findings from the parametric one and uncovers further differences among the best performing options. Unfortunately, and as previously observed, different datasets imply slightly different conclusions.

According to UBS8k results, the best performing blocks are RB1, RB4 and RB5, partially confirming the inappropriateness of RB2 as in [30]. Even though RB1 ranks the first and all means are indistinguishable we can still find some interesting differences. On the one hand, RB4 using both pre-processing options has almost the best median (0.69) which may suggest that the RB4 option is more robust. On the other hand, we obtain an optimistic bound of 0.72 both for RB5 and RB1 with Mean 0 Std 1 pre-processing. The value of this bound for the next best options is 0.71 for RB4 using the same preprocessing.

When considering the ESC-10 results, the previous surprising behavior of RB5 is confirmed in all cases. Moreover, the more specific differences among methods also confirm that pre-processing affects the behavior of RB options for this dataset. In particular, RB1 and RB4 on one hand, and RB2 and RB6 on the other, are the best performing blocks depending on pre-processing, all with indistinguishable means. If we compute the medians as with the previous dataset we find slight differences between RB2 and RB4 (0.80) and RB1 and RB6 (0.79). Finally, the best options according to the optimistic bound when the best models are selected are RB4, RB2 and RB1 (0.84) for different pre-processing options. These bounds, together with the fact that RB1 exhibits significantly worse medians, suggest that both RB4 and RB2 constitute a more robust alternative.

Given these overall results, drawing a general conclusion looks difficult. The more remarkable fact is that the best block considered for the image domain RB5 is not, in general,



(a) UBS8k

(b) ESC-10

FIGURE 4. Means, medians and optimistic bounds on accuracies of the considered residual blocks on two datasets for different pre-processings: Scalemax (SC) in blue and Mean 0 Std 1 (ZM) in green. The best medians and optimistic bounds are marked as cercles and triangles, respectively.

TABLE 2. Ranking results of the different RB configurations.

Pre-processing	Dataset								
	UBS8k			ESC-10			Both		
	R	AvRnk	pval	R	AvRnk	pval	R	AvRnk	pval
Scalemax	RB1	2.75	-	RB4	1.53	-	RB4	2.14	-
	RB4	3.00	0.98	RB1	2.02	0.09	RB1	2.37	0.46
	RB5	3.00	0.98	RB2	3.13	**0.00**	RB2	3.52	**0.00**
	RB2	3.80	**0.04**	RB6	3.71	**0.00**	RB6	3.80	**0.00**
	RB6	4.08	**0.01**	RB3	4.60	**0.00**	RB3	4.43	**0.00**
	RB3	4.38	**0.00**	RB5	6.00	**0.00**	RB5	4.74	**0.00**
Mean 0 Std 1	RB1	2.38	-	RB2	1.73	-	RB1	2.64	-
	RB4	2.77	0.42	RB6	2.24	0.61	RB2	2.72	1.00
	RB5	2.92	0.42	RB1	2.89	**0.01**	RB6	2.85	1.00
	RB2	3.67	**0.01**	RB4	3.16	**0.01**	RB4	2.96	0.93
	RB6	3.73	**0.01**	RB3	5.38	**0.00**	RB5	4.41	**0.00**
	RB3	5.60	**0.00**	RB5	5.60	**0.00**	RB3	5.43	**0.00**

among the bests. Also interesting is the fact that the block RB6 proposed in [20] and specially its close variant RB2 with normalized inputs are among the bests but only for one of the datasets. Finally, the original block RB1 is among the bests for all datasets even though it exhibits a dependence on pre-processing. Also among the bests for all datasets is the RB4 option that can be considered as a small variation of the RB5 recommended in [30]. If one had to put one of the options ahead of the others for 1D end-to-end audio classification from the experimentation carried out in the present work it would be the RB4 design. This ReLU-only pre-activation, as named in [30] had also a very good behavior in image classification. Moreover, it consistently produces median accuracies among the best in all datasets and pre-processing options (except Mean 0 Std 1 in the case of ESC-10).

VI. CONCLUSION

End-to-end 1D architectures are very convenient for addressing audio classification tasks, as they avoid making certain decisions related to the adoption of suitable input representations for the input audio data. As a result, raw audio waveforms can be fed directly into convolutional networks without the need for a prior feature extraction process. While residual learning has been widely demonstrated to be a successful approach for training deep neural networks, different residual block designs may affect the final performance of the classification system. In this context, while the study of the appropriateness of different residual block designs has been previously addressed in the image domain, similar analyses have not been previously reported when considering 1D audio data. In this work, it has been shown that previous results obtained for image classification can not be easily extrapolated to the audio domain. Moreover, significant differences in the performance provided by different residual blocks have been observed when considering different audio datasets and pre-processings. With the considered baseline architecture, some of the recommended residual blocks in the literature did not achieve the best performance, nor even the the most successful block recommended for image classification tasks.

REFERENCES

[1] S. Adavanne and T. Virtanen, "A report on sound event detection with different binaural features," 2017, *arXiv:1710.02997*. [Online]. Available: http://arxiv.org/abs/1710.02997

[2] H. Zhang, I. McLoughlin, and Y. Song, "Robust sound event recognition using convolutional neural networks," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Apr. 2015, pp. 559–563.

[3] P. Foggia, N. Petkov, A. Saggese, N. Strisciuglio, and M. Vento, "Reliable detection of audio events in highly noisy environments," *Pattern Recognit. Lett.*, vol. 65, pp. 22–28, Nov. 2015. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0167865515001981

[4] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, p. 436, 2015.

[5] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014, *arXiv:1409.1556*. [Online]. Available: http://arxiv.org/abs/1409.1556

[6] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2012, pp. 1097–1105.

[7] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 2818–2826.

[8] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 1251–1258.

[9] L. Zhang and J. Han, "Acoustic scene classification using multi-layer temporal pooling based on convolutional neural network," 2019, *arXiv:1902.10063*. [Online]. Available: http://arxiv.org/abs/1902.10063

[10] S. Hershey, S. Chaudhuri, D. P. W. Ellis, J. F. Gemmeke, A. Jansen, R. C. Moore, M. Plakal, D. Platt, R. A. Saurous, B. Seybold, M. Slaney, R. J. Weiss, and K. Wilson, "CNN architectures for large-scale audio classification," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Mar. 2017, pp. 131–135.

[11] Y. Xu, Q. Kong, W. Wang, and M. D. Plumbley, "Large-scale weakly supervised audio classification using gated convolutional neural network," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Apr. 2018, pp. 121–125.

[12] E. Cakır, T. Heittola, and T. Virtanen, "Domestic audio tagging with convolutional neural networks," in *Proc. IEEE AASP Challenge Detection Classification Acoustic Scenes Events (DCASE)*, 2016. [Online]. Available: http://dcase.community/documents/challenge2016/technical_reports/DCASE2016_Cakir_4003.pdf

[13] E. Cakir, G. Parascandolo, T. Heittola, H. Huttunen, and T. Virtanen, "Convolutional recurrent neural networks for polyphonic sound event detection," *IEEE/ACM Trans. Audio, Speech, Language Process.*, vol. 25, no. 6, pp. 1291–1303, Jun. 2017.

[14] J. Naranjo-Alcazar, S. Perez-Castanos, P. Zuccarello, and M. Cobos, "Acoustic scene classification with squeeze-excitation residual networks," *IEEE Access*, vol. 8, pp. 112287–112296, 2020.

[15] J. Naranjo-Alcazar, S. Perez-Castanos, P. Zuccarello, F. Antonacci, and M. Cobos, "Open set audio classification using autoencoders trained on few data," *Sensors*, vol. 20, no. 13, p. 3741, Jul. 2020.

[16] M. Valenti, A. Diment, G. Parascandolo, S. Squartini, and T. Virtanen, "DCASE 2016 acoustic scene classification using convolutional neural networks," in *Proc. Workshop Detection Classification Acoust. Scenes Events*, 2016, pp. 95–99.

[17] A. Mesaros, T. Heittola, and T. Virtanen, "Acoustic scene classification in DCASE 2019 challenge: Closed and open set classification and data mismatch setups," in *Proc. Detection Classification Acoustic Scenes Events Workshop (DCASE)*, New York, NY, USA, Oct. 2019, pp. 164–168.

[18] T. Lidy and A. Schindler, "CQT-based convolutional neural networks for audio scene classification," in *Proc. Detection Classification Acoustic Scenes Events Workshop (DCASE)*, vol. 90, 2016, pp. 1032–1048.

[19] K. J. Piczak, "The details that matter: Frequency resolution of spectrograms in acoustic scene classification," in *Proc. Detection Classification Acoustic Scenes Events Workshop (DCASE)*, Nov. 2017, pp. 103–107.

[20] W. Dai, C. Dai, S. Qu, J. Li, and S. Das, "Very deep convolutional neural networks for raw waveforms," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Mar. 2017, pp. 421–425.

[21] S. Qu, J. Li, W. Dai, and S. Das, "Understanding audio pattern using convolutional neural network from raw waveforms," 2016, *arXiv:1611.09524*. [Online]. Available: http://arxiv.org/abs/1611.09524

[22] J. Lee, J. Park, K. Kim, and J. Nam, "SampleCNN: End-to-end deep convolutional neural networks using very small filters for music classification," *Appl. Sci.*, vol. 8, no. 1, p. 150, Jan. 2018.

[23] Y. Gong and C. Poellabauer, "How do deep convolutional neural networks learn from raw audio waveforms?" Tech. Rep., 2018. [Online]. Available: https://openreview.net/pdf?id=S1Ow_e-Rb

[24] J. Lee, J. Park, K. L. Kim, and J. Nam, "Sample-level deep convolutional neural networks for music auto-tagging using raw waveforms," 2017, *arXiv:1703.01789*. [Online]. Available: http://arxiv.org/abs/1703.01789

[25] J. J. Huang and J. J. A. Leanos, "AclNet: Efficient end-to-end audio classification CNN," 2018, *arXiv:1811.06669*. [Online]. Available: http://arxiv.org/abs/1811.06669

[26] J. Vera-Diaz, D. Pizarro, and J. Macias-Guarasa, "Towards end-to-end acoustic localization using deep learning: From audio signals to source position coordinates," *Sensors*, vol. 18, no. 10, p. 3418, Oct. 2018.

[27] S. Abdoli, P. Cardinal, and A. L. Koerich, "End-to-end environmental sound classification using a 1D convolutional neural network," *Expert Syst. Appl.*, vol. 136, pp. 252–263, Dec. 2019.

[28] T. Kim, J. Lee, and J. Nam, "Comparison and analysis of SampleCNN architectures for audio classification," *IEEE J. Sel. Topics Signal Process.*, vol. 13, no. 2, pp. 285–297, May 2019.

[29] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.

[30] K. He, X. Zhang, S. Ren, and J. Sun, "Identity mappings in deep residual networks," in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, 2016, pp. 630–645.

[31] T. Kim, J. Lee, and J. Nam, "Sample-level CNN architectures for music auto-tagging using raw waveforms," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Apr. 2018, pp. 366–370.

[32] J.-W. Jung, H.-S. Heo, I. Yang, S.-H. Yoon, H.-J. Shim, and H.-J. Yu. (2017). *DNN-Based Audio Scene Classification for DCASE 2017: Dual Input Features, Balancing Cost, and Stochastic Data Duplication Detection and Classification of Acoustic Scenes and Events*. [Online]. Available: http://dcase.community/documents/workshop2017/proceedings/DCASE2017Work shop_Jung_187.pdf

[33] J. H. Yang, N. K. Kim, and H. K. Kim. (2018). *Se-Resnet With Gan-Based Data Augmentation Applied to Acoustic Scene Classification Detection and Classification of Acoustic Scenes and Events*. [Online]. Available: https://pdfs.semanticscholar.org/e95f/b1ac75c42943c4a74e5c082bfdcc07d90c1f.pdf

[34] M. Liu, W. Wang, and Y. Li. (2019). *The System for Acoustic Scene Classification Using Resnet Detection and Classification of Acoustic Scenes and Events*. [Online]. Available: http://dcase.community/documents/challenge2019/technical_reports/DCASE 2019_SCUT_19.pdf

[35] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 7132–7141.

[36] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 1929–1958, 2014. [Online]. Available: http://jmlr.org/papers/v15/srivastava14a.html

[37] J. Salamon, C. Jacoby, and J. P. Bello, "A dataset and taxonomy for urban sound research," in *Proc. 22nd ACM Int. Conf. Multimedia*, 2014, pp. 1041–1044.

[38] K. J. Piczak, "ESC: Dataset for environmental sound classification," in *Proc. 23rd ACM Int. Conf. Multimedia (MM)*, 2015, pp. 1015–1018.

[39] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*. [Online]. Available: http://arxiv.org/abs/1412.6980

[40] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proc. Int. Conf. Artif. Intell. Statist. (AISTATS)*, 2010, pp. 249–256.

[41] B. McFee, C. Raffel, D. Liang, D. Ellis, M. McVicar, E. Battenberg, and O. Nieto, "Librosa: Audio and music signal analysis in Python," in *Proc. 14th Python Sci. Conf.*, 2015, pp. 18–25.

[42] J. W. Tukey, "Comparing individual means in the analysis of variance," *Biometrics*, vol. 5, no. 2, pp. 99–114, 1949. [Online]. Available: http://www.jstor.org/stable/3001913

[43] S. Garcia and F. Herrera, "An extension on 'statistical comparisons of classifiers over multiple data sets' for all pairwise comparisons," *J. Mach. Learn. Res.*, vol. 9, pp. 2677–2694, Dec. 2008.

JAVIER NARANJO-ALCAZAR (Graduate Student Member, IEEE) received the Telecommunications degree and the master's degree in telecommunications engineering from the Universitat Politècnica de València, Valencia, Spain, in 2016 and 2018, respectively. He is currently pursuing the Ph.D. degree with the Department of Computer Science, Universitat de València, funded by the Torres Quevedo Program and the valencian start-up Visualfy. His research interests include machine listening, few-shot learning, and open-set recognition. He was a recipient of the Best M.Sc. Thesis Award from the Regional Telecommunications Engineering Association in 2019.

SERGI PEREZ-CASTANOS received the Telecommunications degree and the master's degree in telecommunications engineering from the Universitat Politècnica de València, Valencia, Spain, in 2016 and 2018, respectively. He is currently working as a Machine Learning Engineer with Visualfy. His research interests include machine listening, anomaly detection, and audio captioning.

IRENE MARTÍN-MORATÓ (Graduate Student Member, IEEE) received the bachelor's (Hons.) and M.Sc. degrees in telecommunications and the Ph.D. degree in information technology, communications, and computing under the University Faculty Training Program (FPU) from the Universitat de València, in 2014, 2016, and 2019, respectively. She is currently a Postdoctoral Researcher with Tampere University, Finland. Her research interests include acoustic signal processing, machine learning, and audio event detection and classification.

PEDRO ZUCCARELLO received the Electronics Engineering degree from the University of Buenos Aires, Argentina, the M.Sc. degree in telecommunications from the Universitat Politècnica de València, Valencia, Spain, and the Ph.D. degree from the Universitat de València, Valencia. He developed most of his career as a Researcher in several public research and development institutions such as the Institute of Microelectronics of Barcelona, Barcelona, Spain, or the Institute of Corpuscular Physics, Valencia. From 2017 to June 2020, he has developed as the Head of the Artificial Intelligence Group, Visualfy, a private startup company. He currently works as a Senior Artificial Intelligence Researcher with Tyris IA private company. He has coauthored nearly 30 papers in international peer-review journals and conferences in topics that include artificial intelligence, machine learning, signal processing, electronics, and microelectronic design. He received several postdoctoral fellowships such as the Val-I+D, from the Valencian Government, or the Torres Quevedo, from the Spanish Ministry of Science and Education.

FRANCESC J. FERRI (Senior Member, IEEE) received the Licenciado degree in physics (electricity, electronics, and computer science) and the Ph.D. degree in pattern recognition from the Universitat de València, in 1987 and 1993, respectively. He has been with the Computer Science Department, Universitat de València, since 1986. His current research interests include feature selection, nonparametric classification methods, machine learning, computer vision, and image retrieval. He has authored or coauthored more than 100 technical papers in international conferences and well established journals in his fields of interest. He is a member of ACM and IAPR.

MAXIMO COBOS (Senior Member, IEEE) received the master's degree in telecommunications and the Ph.D. degree in telecommunications engineering from the Universitat Politècnica de València, Valencia, Spain, in 2007 and 2009, respectively. In 2011, he joined the Universitat de València, where he is currently an Associate Professor. His research interests include digital signal processing and machine learning for audio and multimedia applications. He has authored/coauthored more than 100 technical papers in international journals and conferences in his areas of interest. He is a member of the Audio Signal Processing Technical Committee of the European Acoustics Association. He completed with honors his studies under the University Faculty Training program (FPU) and was a recipient of the Ericsson Best Ph.D. Thesis Award from the Spanish National Telecommunications Engineering Association. In 2010, he received the Campus de Excelencia Postdoctoral Fellowship to work at the Institute of Telecommunications and Multimedia Applications. He serves as an Associate Editor for IEEE SIGNAL PROCESSING LETTERS and the *EURASIP Journal on Audio, Speech, and Music Processing*.