Índice general

Capítulo 1

Introducción

1.1. Contexto y Justificación del Trabajo

El TFM tratará la mejora de las técnicas de aprendizaje profundo mediante la inclusión en el procesado de transformaciones que incluyan equivarianzas e invarianzas. Hay muchas transformaciones que permiten implementar equivarianzas como por ejemplo a escala, a rotación o a iluminación y a **traslación**. La inclusión de estas equivarianzas en aprendizaje profundo es un tema con un gran auge en la actualidad. En concreto, las neuronas y modelos bioinspirados (es decir, modelos que incluyen estas invarianzas o equivanzas) logran mejorar el rendimiento de las redes convencionales. Estas mejoras se han comprobado en multitud de problemas de procesado de imágenes: codificación [1, 2], medidas de calidad [3, 4], restauración [5, 6], síntesis [7], renderizado [8] y clasificación [9, 10]. Esta mejora no es fortuita, pues tanto el cerebro como el aprendizaje automático se ven sometidos a los mismos problemas estadísticos, lo cual es clave. De hecho, se han desarrollado nuevas técnicas estadísticas en los últimos años debido al estudio exhaustivo en los problemas comunes que atañen al aprendizaje automático y a la neurociencia visual. Estas técnicas estadísticas van desde funciones de densidad de probabilidad (PDF) [11, 12], a la descripción de variedades (manifolds) [13, 14, 15, 16] y medidas basadas en teoría de la información [17, 18, 19] que han demostrado ser útiles en varios campos, como la comprensión de datos en geociencias [20, 21] y en el análisis del flujo de información en neurociencia [22, 23].

1.2. Descripción general

El TFM trata la inclusión de estas equivarianzas desde un punto de vista probabilístico (ya que están en la naturaleza de los datos) y desde un punto de vista biológico (ya que nuestro sistema visual las implementa de una forma natural). En particular, se centra en el estudio de equivarianzas de tipo traslacional que, si bien se conservan en las convoluciones, se pierden en la incorporación de capas adicionales -como las capas densas- y en otro tipo de contextos. El proyecto consiste en estudiar este tipo de invarianzas, los modelos que las incluyen y su aplicación a diferentes problemas.

1.3. Objetivos

1.3.1. Objetivos generales

Mejora de modelos de *Deep Learning* mediante invarianzas y equivarianzas traslacionales en problemas de clasificación y de calidad de imagen (*Image Quality Assessment [IQA]*).

1.3.2. Objetivos específicos

- Familiarización e implementación de modelos que incluyan equivarianzas e invarizandas básicos a la traslación: modelos con capas *Global Average Pooling/Global Max Pooling*.
- Diseño de una arquitectura y entrenamiento estable de modelos que incluyan equivarianzas e invarizandas.
- Aplicación en las problemáticas de clasificación e IQA.

1.4. Impacto en sostenibilidad, ético-social y de diversidad

En esta sección se ahondará sobre cómo el proyecto impacta en las 3 dimensiones de la CCEG. Para cada una de ellas, se reflexionará sobre el impacto y posibles correcciones.

1.4.1. Sostenibilidad

En líneas generales, el entrenamiento de redes neuronales siempre va a impactar de manera negativa en el medio ambiente. El uso -y muchas veces abuso- de corriente eléctrica, servidores, refrigeración y un largo etcétera de recursos, hace que esta práctica sea muy poco sostenible. Este proyecto no es la excepción, para su realización se requieren todos esos recursos. Además, no es un problema totalmente medible, puesto que es complicado inferir qué uso se le va a dar tanto al ordenador como a los servidores, como al resto de recursos. Desde este punto de vista, se intentará minimizar su uso intentando hacer únicamente las pruebas pertinentes, pero es un compromiso complicado de cumplir.

Hay un aspecto positivo a remarcar en cuanto a la sostenibilidad. El hecho de incorporar capas o mecanismos que ayuden a la red a captar equi/invarianzas implica que ya no sería menester utilizar técnicas como el *data-augmentation*. Esta técnica consiste en el uso excesivo de datos de entrada que hagan que la red sea más robusta a datos anómalos -suponiendo que ya los ha visualizado-. Es cierto que esta técnica sí mejora los resultados obtenidos [24] pero requiere de un mayor tiempo computacional y, por ende, de una mayor cantidad de recursos. Si se evita el uso de esta técnica mediante la incorporación de nuevas arquitecturas a las redes, se mejorará el rendimiento a cambio de no necesitar esos recursos.

1.4.2. Ético-Social

El resultado de este proyecto es lo suficientemente técnico como para que no impacte de ninguna manera en la dimensión ético-social. El objetivo final es crear redes que sean más robustas a las traslaciones -ya sean equi o invariantes- por lo que el producto no puede tener beneficios económicos -las redes neuronales se están explotando ya sin este añadido- ni reputacionales ni en privilegios. Tampoco destruye -ni crea- puestos de trabajo porque el uso de estas redes posiblemente más robustas busca acabar con las dificultades asociadas al entrenamiento de las mismas, no con cómo se usan actualmente. Al mitigar estos problemas, no se espera que se

resuelva ni se cambie ningún otro problema tangencial.

Durante el proyecto no se trabajan con datos sensibles ni de la autora ni de ninguna otra persona, por lo que tampoco impacta en esta problemática.

1.4.3. Diversidad

Hay numerosos estudios [25, 26] que profundizan sobre la aparición de sesgos relacionados con el género y la raza en diferentes modelos de Inteligencia Artificial. El cómo se entrena una red y a partir de qué datos se entrena, son el principal causante de la aparición de estos sesgos. Siendo conscientes de por qué aparecen y la importancia de mostrarle a la red datos que reflejen las diferentes diversidades, hay que hacer un esfuerzo por crear bases de datos con estas características.

Debido al carácter más lúdico del que se suele hacer uso en este tipo de proyectos, en general se dispondrá de bases de datos no relacionadas con la realidad, por lo que no se hará referencia a personas -de ningún género ni etnia- que cree sesgos negativos en las redes. No obstante, existe un compromiso sincero con la no aparición de sesgos de género, raza, etnia, origen, condición/orientación sexual, ideología, religión, diversidad funcional o posición social.

1.5. Enfoque y método a seguir

El enfoque y la metodología a seguir es la clásica en este tipo de proyectos. Se puede ver un resumen más visual en la Figura 1.1.

Vamos a explicar detalladamente cada fase.

1. **Fase 1 - Baseline:** Lo primero es generar, con modelos clásicos, una base con la que comparar los modelos futuros en las aplicaciones deseadas. De esa forma podremos saber si los modelos bioinspirados son mejores (o no) que las arquitecturas básicas mejorando sus métricas.

2. **Fase 2 - Modelos bioinspirados:** Se programarán los modelos que incluyan invarianzas y equivarianzas. En primera instancia, modelos básicos y conforme se vayan entendiendo y resulten familiares, modelos más complejos.

3. **Fase 3 - Problemas de clasificación:** Una vez se tengan los modelos, es el momento de aplicarlo a las problemáticas. En esta fase nos centraremos en las tareas de clasificación, en el que trasladaremos los objetos de las imágenes y probaremos si las redes son capaces de clasificar o no los objetos que hay en ellas a pesar de estar desplazadas.

4. **Fase 4 - Problemas IQA:** Otra problemática habitual que está muy relacionada con las invarianzas son la de las medidas de calidad. Este problema consiste en utilizar una red neuronal para transformar las imágenes a un espacio donde la distancia entre ellas se asemejaría más a la percepción humana, comprobando que ambas imágenes (la real y la trasladada) terminan en un lugar próximo en ese espacio.

5. **Fase 5 - Resultados:** Por último, una vez se obtengan las métricas relacionadas de los modelos bioinspirados, se comprarán dichas méticas con las del *baseline* de la primera fase. Se hará un análisis de los resultados obtenidos y se concluirá en qué contextos funcionan

Figura 1.1: Metodología del Proyecto. Dividida en 5 fases: Baseline, modelos bioinspirados, aplicación a problemas de clasificación e IQA y resultados.

mejor qué modelos. Para finalizar, se hará un proyección de futuro sobre el trabajo: en qué otras aplicaciones se puede poner en práctica o qué mejoras se podrían incorporar.

1.6. Planificación

1.6.1. Tareas

El trabajo se compone de 4 tareas principales que se desarrollarán a lo largo de todo el proyecto y cuyo desglose temporal se puede observar en la Sección 1.6.2.

1. **Lecturas:** En este apartado se espera una profunda revisión del estado del arte en la cuestión de las invarianzas y equivarianzas en general y de las de traslación en particular.

2. **Implementación:** En esta etapa del proyecto se propone implementar los modelos básicos que ya experimenten con las invarianzas y equivarianzas. Una vez entendidos estos modelos, se pasará a la implementación de modelos más complejos. Estos modelos serán puestos a prueba por conjuntos de datos sintéticos para probar su efectividad.

3. **Aplicaciones:** Se pondrá en práctica los modelos propuestos en dos problemas diferentes. Por un lado, la clasificación y por otra la calidad de imagen.

4. **Desarrollo de entregables:** A lo largo de todo el proyecto, se deben generar 2 productos

finales: La memoria y la presentación. En particular, la memoria se ha de generar conforme el proyecto avance y la presentación se reserva tiempo específico para ella cuando se hayan acabado todos los apartados técnicos.

1.6.2. Calendario

Teniendo en cuenta todas las tareas y subtareas que componen el proyecto, se ha generado un Diagrama de Gantt, Figura 1.2, con la planificación general. Tanto la tarea de tutorías como la de desarrollo de productos es transversal a todo el trabajo.

1.6.3. Hitos

Como fechas importantes, las hay de dos tipos. Por un lado las fechas correspondientes a la finalización de las fases:

- **Hito 1:** 16/12/2023 → Donde se dispondrá toda la planificación del proyecto.
- **Hito 2:** 10/3/2023 → Se habrá terminado la parte de implementación y ya se dispondrá de todo lo necesario para empezar con las aplicaciones.
- **Hito 3:** 5/5/2023 → Ya se habrá realizado el apartado de aplicaciones, en su gran mayoría, con problemas de clasificaciones y de estimación de calidad de imagen.
- **Hito 4:** 7/6/2023 → Se habrá finalizado ya todo el apartado de aplicaciones y solo quedará elaborar los productos finales.
- **Hito 5:** 12/06/2023 - 22/06/2023 → Defensa

Además de los hitos marcados por las entregas, también se han pensado fechas para hablar con los tutores, de forma que siempre estén informados de los avances del proyecto en momentos importantes:

- **Inicios de Marzo:** Fecha donde se acaba toda la revisión del estado del arte y se habrá trabajado ya los modelos y bancos de datos sintéticos.
- **Finales de Mayo:** Fecha que coincide con el fin de proyecto, se abordarán los resultados obtenidos y las conclusiones.

1.6.4. Análisis de riesgos

Hay dos factores de riesgo que siempre se han de considerar en un proyecto de estas características y que son problemáticas habituales y generales:

- **Alcance del proyecto:** Como se ha comentado en la Sección 1.1, el problema en el que se enmarca el proyecto es una problemática en auge que aún está abierta. Aunque se ha tratado de delimitar las tareas a resolver, es una posibilidad viable el desviarse del camino para probar otras cosas nuevas y probar otros enfoques.
- **Tiempo disponible:** Teniendo en cuenta que simultáneamente al proyecto se está completando el máster y que se ha empezado a trabajar este año en el grupo de investigación (el mismo que el del cotutor), el tiempo podría ser un factor de riesgo importante. Es la razón porque la que se ha intentado espaciar con margen cada tarea por si alguna de las otras ocupaciones requiere más tiempo del previsto no haya problemas con la entrega final.

En cuanto a los riesgos que puede ocasionar este trabajo en particular:

- **Tiempo computacional:** Como ocurre en casi todos los proyectos relacionados con las redes neuronales, el tiempo de computación es un factor crítico a tener en cuenta. Para que los resultados tengan significancia estadística, un mismo modelo ha de ser lanzado varias veces para comprobar su robustez y eso implica, además del tiempo de entrenamiento de un solo modelo, el repetirlo tantas veces como sea preciso. En general, es una limitación técnica que tenemos que asumir.
- **Datos en la aplicación real:** A pesar de que está todavía por definir, adquirir datos de problemáticas reales puede ser un problema. Hay que buscarlos, procesarlos y adecuarlos para el uso propuesto. No siempre se encuentran de manera sencilla, por ejemplo, los datos relacionados con temáticas en salud son complicados de adquirir. Como aún no está pensada sobre qué problemática real se va a trabajar aún es complicado conocer el riesgo al que haremos frente, pero es necesario tenerlo en cuenta.

1.7. Resultados esperados

El resultado final del proyecto será un documento que haga de memoria de todos los pasos e hitos que se han experimentado a lo largo del proyecto, a la vez que reflejará los resultados y conclusiones obtenidas.

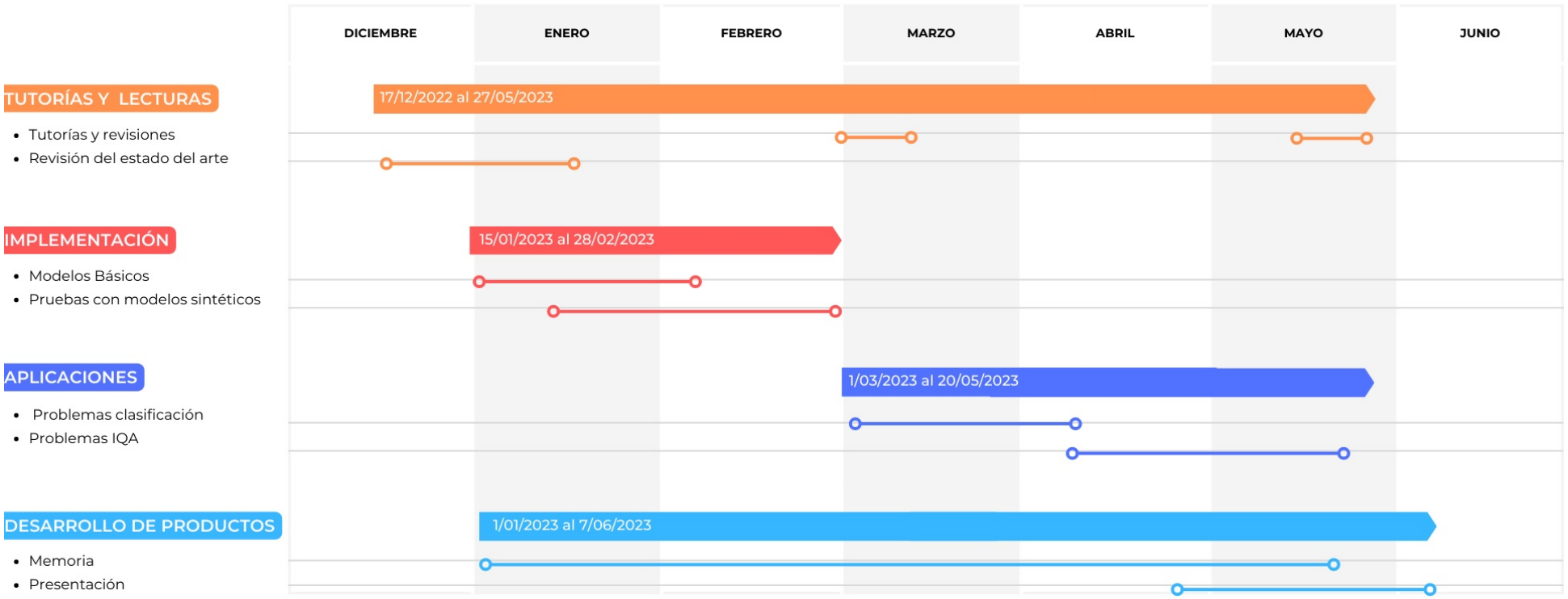


Figura 1.2: Diagrama de Gantt del proyecto. Diferenciando entre las diferentes tareas: Lecturas y desarrollo de productos, que abarcan todo el proyecto, implementación, de Enero a Febrero, y aplicaciones, de Marzo a Mayo.

Capítulo 2

Marco Teórico

2.1. Breve repaso de las invarianzas y equivarianzas

Desde que en 1979 se publicase el *paper* original de las Redes Convolucionales (CNN) [27] y en 1989 se terminasen de asentar en la publicación [28], muchas otras publicaciones han surgido para intentar mejorar su resistencia a diferentes invariancias y datos. Se listan a continuación, algunas de las divulgaciones más relevantes sobre el entrenamiento de las Redes Convoluionales centrándonos en su relación con las equivarianzas e invarianzas.

En 1995 [29] ya se pueden encontrar publicaciones señalando el caracter equivariante de las redes convolucionales. Es una idea recurrente que ha aparecido mucho a lo largo de la literatura [30, 31, 32] donde se expresa que esta característica viene dada de forma innata por la propia capa.

En cuanto a la temática principal del proyecto, las invarianzas traslacionales, empiezan a haber publicaciones en 2014 [33] señalando las limitaciones en este aspecto de las redes convencionales. En relación a esto, también pueden ser interesantes las publicaciones [34] y [35], donde se hace especial hincapié en la importancia de los datos de entrada. En los que se refleja cómo el aumento del conjunto de datos de entrada es el factor el factor más importante para obtener representaciones de imágenes invariantes utilizando redes neuronales convolucionales.

Un inconveniente que ha permeado mucho sobre esta temática es la asidua confusión entre invarianzas entrenadas e invarianzas *online* [36]. Las primeras consisten en enseñar a la red el conjunto que tiene que predecir ya trasladado durante la fase de entrenamiento, lo que se conoce como *data-augmentation*. En 2017 se exploró esta idea en [37, 34]. La segunda, consiste en enseñarle a la red el objeto en una sola posición y que sea capaz de localizar e identificar ese objeto independientemente de dónde esté. Sin embargo, esto tiene que ocurrir sin que lo haya visualizado anteriormente durante el entrenamiento. Esto es una idea especialmente estudiada, con publicaciones desde 1991 y 1992 [38, 39] hasta 2021 [40]. Además, no debe confundirse con *arquitectura online*, donde se busca la robustez a las invarianzas a través de modificaciones en la arquitectura de la red: [40], [41]. Este último es el punto sobre el que versa realmente el trabajo.

En Septiempre de 2021, hace referencia al hecho de preentrenar las redes con datasets más sencillos de formas que al ser usadas con conjuntos de datos más complejos sean capaces de

aplicar las invarianzas aprendidas. Jeffrey S. Bowers tiene dos publicaciones alrededor de esta idea: [42, 43]. Con esto se consigue que la red sea invariante de manera *online*, aunque no lo hace desde un punto de vista estructural.

Por último, en Octubre de 2022, se publica [44], donde se propone una CNN totalmente óptica de espacio libre (denominada Trans-ONN) que clasifica con precisión imágenes trasladadas en sentido horizontal, vertical o diagonal. Esta nueva red aprovecha las ventajas de una capa de agrupación de movimiento óptico que proporciona la propiedad de invarianza traslacional mediante la implementación de diferentes máscaras ópticas en el plano de Fourier para clasificar las imágenes de prueba trasladadas y que obtiene muy buenos resultados.

2.2. Conceptos previos

Para que este documento sea lo más autocontenido posible y facilitar al lector la comprensión, se definen a continuación algunos de los conceptos clave que se usan a lo largo del trabajo.

- **Variedad:** Estructura organizada de los datos en una dimensionalidad inferior a la de su espacio de características a través de un homeomorfismo f(x), que puede ser diferenciable.
- **Isotropía:** Propiedad de transmitir igualmente en todas direcciones cualquier acción recibida en un punto de su masa.
- **Función de activación:** Función que se aplica a la salida de cada capa de la red neuronal y que restringe la salida de la misma. Con las funciones de activación se intenta limitar el rango de la salida o el tipo (discreta o continua). Las que van a tratarse a lo largo del trabajo son:
  - **Sigmoide:** Tiene un rango de salida entre 0 y 1, por lo que se puede interpretar como la probabilidad de pertenencia a una clase. Sigue la fórmula:

    $$f(x) = \frac{1}{1 + e^{-x}}$$

    Se puede observar su representación gráfica en la Figura 2.1a.

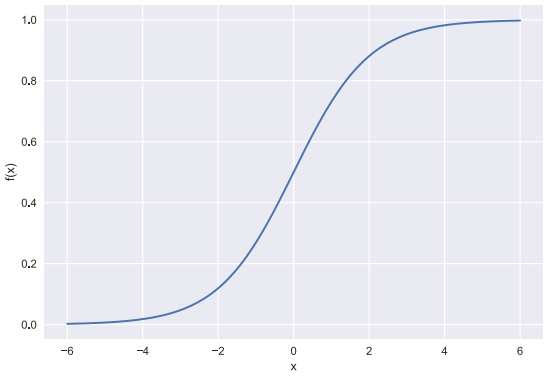
  - **Tangente hiperbólica (tanh):** Tiene un rango de valores en la salida entre -1 y 1, funciona como ampliación de la función sigmoide como se puede apreciar en la Figura 2.1b. Sigue la fórmula:

    $$f(x) = \frac{2}{1 + e^{-2x}} - 1$$

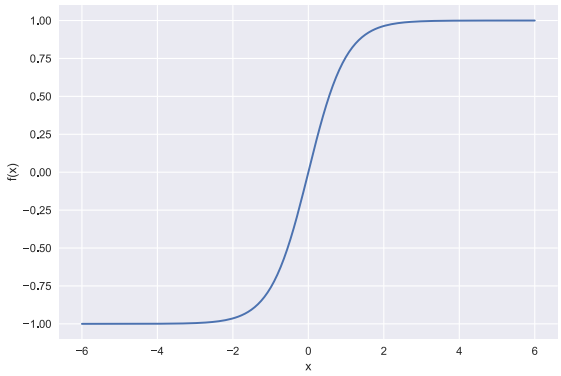
  - **Unidad lineal rectificada (ReLU):** Descrita en el año 2000 [45] [46], propone la anulación de todas las entradas negativas a la función, Figura 2.1c.

    $$f(x) = \begin{cases} \text{x} & \text{si} \quad \text{x} > 0 \\ 0 & \text{si} \quad x \leq 0 \end{cases}$$

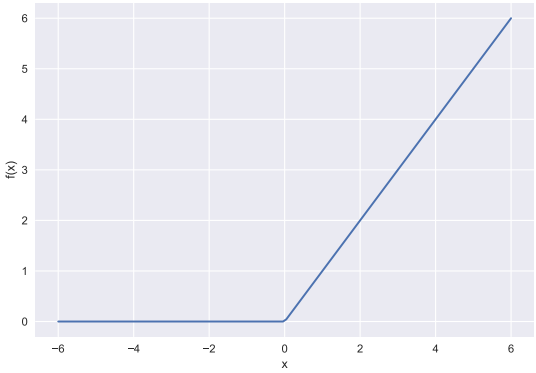
    Es una función de activación sencilla con cálculos lineales que no son complejos de realizar. Como inconveniente, con esta función se da el problema del *desvanecimiento*



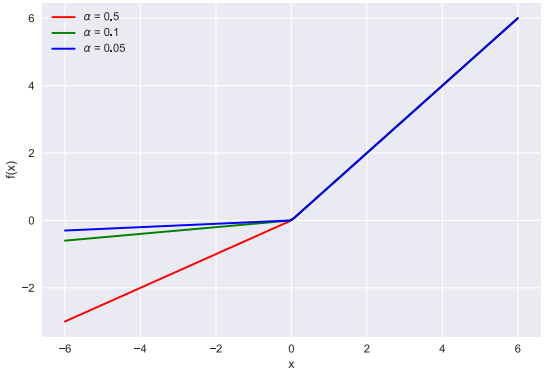
(a) Sigmoide



(b) Tangente hiperbólica



(c) ReLU



(d) Leaky ReLU

Figura 2.1: Diferentes funciones de activación utilizadas en el ámbito de las redes neuronales.

del gradiente en el que un número excesivo de neuronas se apagan por tener peso 0 y el error no puede llegar a las primeras capas. Aunque esto genera redes dispersas y favorece el entrenamiento si no ocurre de manera excesiva. Para solucionarlo, aparece la función Leaky ReLU.

- **Unidad lineal rectificada con fugas (Leaky ReLU):** Se basa en la función ReLU, salvo que cuenta con una pendiente en el dominio negativo. El parámetro α que determina la pendiente no se aprende de manera automática, sino que hay que determinarlo previamente en el entrenamiento.

$$f(x) = \begin{cases} \alpha \text{ x} & \text{si} \quad \text{x} < 0 \\ \text{x} & \text{si} \quad x \geq 0 \end{cases}$$

En la Figura 2.1d se pueden observar diferentes pendientes con sus respectivos parámetros α. Esta función evita el problema del *desvanecimiento del gradiente*, y es la función que se usa en este trabajo para todas las redes profundas.

- **Capas:** A lo largo del trabajo se hacen referencia a diferentes capas que pueden formar parte de la arquitectura de una red neuronal. Las que más se usan, son:

- **Convolucionales:** Son capas especializadas en la extracción de características. Si bien las hay de varias dimensiones -1, 2 o 3-, en este proyecto se usan exclusivamente las de 2 dimensiones por tratarse de un problema cuyos datos de entrada son imágenes bidimensionales. Su objetivo es crear un mapa de características asociado al dato de entrada que resuma sus propiedades [28] [47].
- **Pooling:** Se colocan habitualmente tras las capas de convolución y reducen su dimensionalidad en alto y ancho, pero no en profundidad. La manera en la que reduce la dimensionalidad depende del tipo de pooling que se elija -máximo, mínimo, media, global, etc..- [48].
- **Batch Normalization:** Es una capa de regularización que, para cada lote de datos, lo normaliza para evitar que los rangos de los lotes no sean los mismos y mantener la normalización inicial a la que habremos sometido a los datos. De esta forma, se garantiza que, para toda la red, los datos siempre tengan el mismo rango [49].

2.3. Modelos Equivariantes

Como se comenta en la Sección 1, este trabajo se centra únicamente en el marco de trabajo de las imágenes y en el de las transformaciones producidas por traslaciones, por lo que se define equivarianza en estos términos aunque sea una definición general para todas las trasnforaciones posibles. En este contexto, dada una imagen A y el grupo de transformaciones traslacionales G, se dice que una función o sistema f es **equivariante a la traslación** si al aplicar una transformación g a la imagen A, la salida de la función f mantiene esa transformación g [29, 50, 30, 32]. Es decir:

$$B = f(A)$$
$$g(B) = f(g(A))$$

Lo que se quiere obtener es lo que muestra la Figura 2.2, que una transformación en el dominio de la imagen nos lleve a un punto igualmente transformado en el dominio del problema.

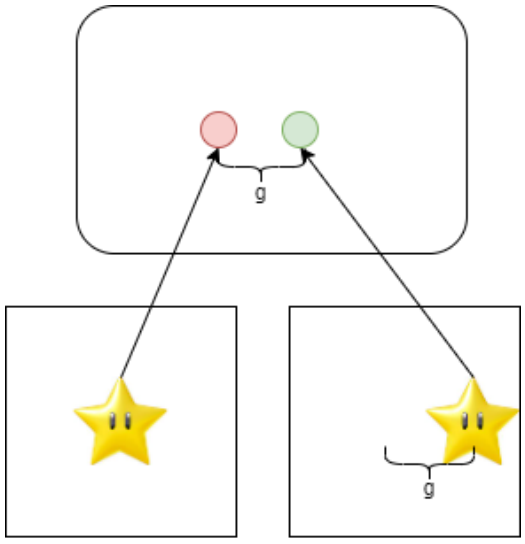
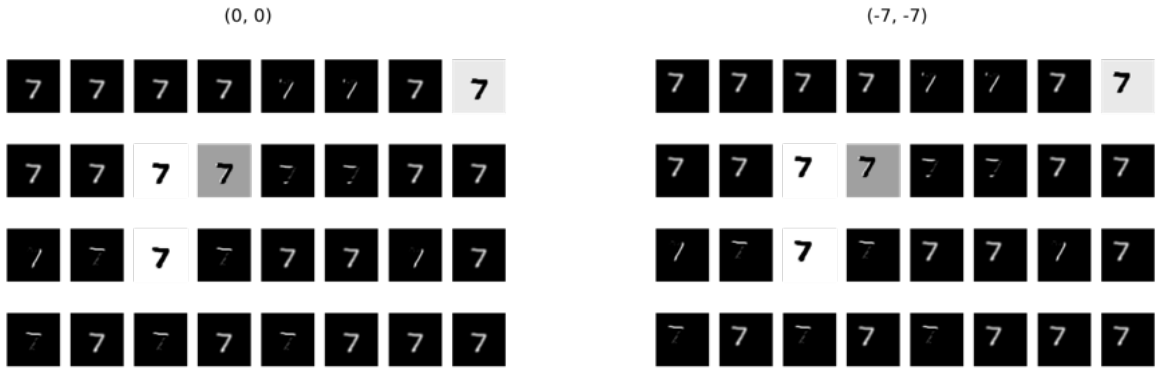


Figura 2.2: Diagrama de modelo equivariante, en el que al trasladar la imagen de entrada se mantiene la misma traslación en el dominio interno.

Esta cualidad es verdaderamente útil para según qué problemáticas, pero no para todas. Por ejemplo, que la transformación se mantenga es útil para problemas de detectores de objetos. En estas problemáticas, es importante saber dónde se ha movido el objeto a reconocer para etiquetarlo de manera correcta. Inicialmente está en una posición cualquiera (x, y) y la tarea consiste en etiquetar de nuevo el mismo objeto que se ha desplazado cuando está en la posición (w, z). Por lo tanto, es una característica deseable en redes cuya finalidad es buscar y señalar al objeto en sus distintas posiciones.

Modelos Equivariantes Básicos

Se ha visto en la Sección 2.1 que la arquitectura básica de las redes convolucionales ya cuenta con esta característica. Una forma sencilla de probar esta propiedad es comprobando los mapas de características resultantes en función de si la imagen de entrada está o no desplazada. En la Figura 2.3 se puede observar más fácilmente la propiedad. Para este ejemplo, se toma una imagen del conjunto MNIST centrado -Sección 4.1- , en este caso del dígito 7. A continuación, se pasa la imagen a través de una única capa convolucional y se pasa a revisar los mapas de características generados. En 2.3a se pueden ver los mapas de características de la imagen sin desplazar, es decir, en el centro de la imagen. En 2.3b se puede ver cómo se extrajeron exactamente las mismas características, pero desplazas de la misma forma en que se desplazó la imagen original, en la esquina izquierda.



(a) Mapas de características sin desplazamiento (b) Mapas de características con desplazamiento

Figura 2.3: Mapas de características con imágenes de entrada desplazadas. En 2.3a, los mapas de características con la imagen de entrada sin desplazar. En 2.3b, los mapas de la imagen de entrada con desplazamiento de 7 píxeles a la izquierda y hacia arriba. Se observa que los mapas de características son exactamente iguales pero desplazados la misma cantidad de píxeles.

Para ponerlo en práctica, se ha utilizado el conjunto de datos de *Buscando a Wally* -explicado detalladamente en la Sección 4.3- y se ha empleado una red relativamente sencilla para comprobar su funcionamiento. El objetivo en este caso es comprobar que la red es equivariante a la traslación si es capaz de reconocer a Wally en cualquier punto del espacio, porque conserva la información espacial y no la elimina. Debido al tamaño de las imágenes -grandes a pesar de reescalar- se han empleado dos conjuntos de convolución más *Max Pooling*. La arquitectura de la red se puede visualizar en la Figura 2.4.

Los resultados que ofrece la red se pueden visualizar en la Figura 2.5, donde podemos ver algunas imágenes del conjunto de test junto con la etiqueta que devuelve, señalizado con un

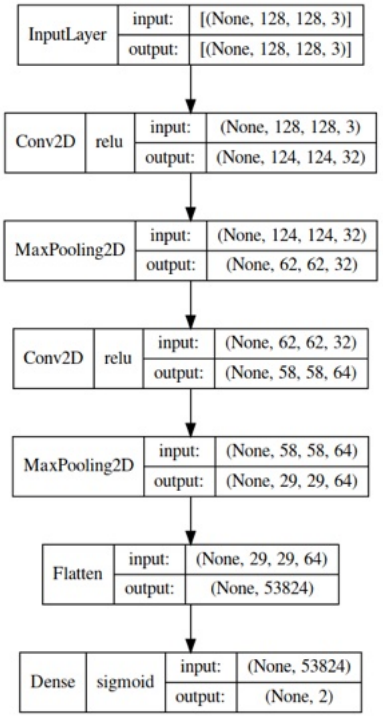


Figura 2.4: Arquitectura del modelo básico utilizado. Se incluyen equivarianzas al utilizar capas convoluciones seguidas de max pooling. Se repite la estructura dos veces para poder procesar imágenes de mayor tamaño y no tener que reducir demasiado su dimensionalidad.

punto rojo. Como se puede observar, la red identifica correctamente la forma de Wally en cualquier punto del fondo.

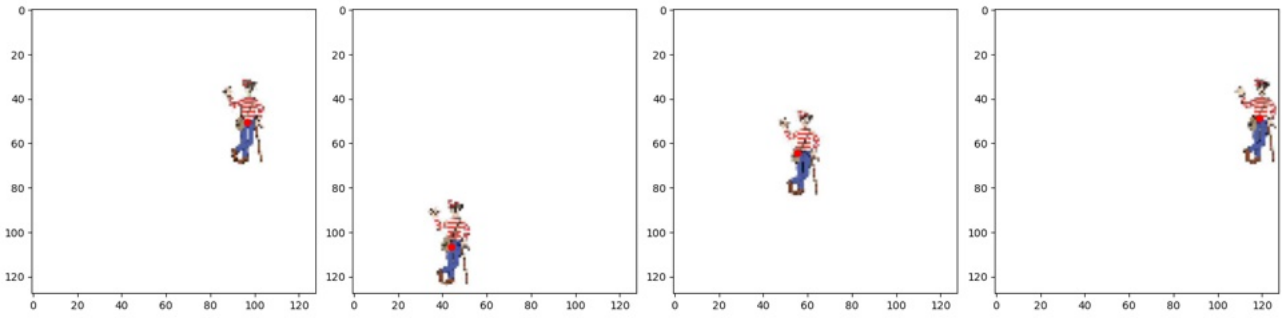


Figura 2.5: Imágenes de muestra del conjunto de datos *Buscando a Wally* explicado en la Sección 4.3. Las imágenes, de 500x500, constan de un fondo blanco sobre el que se superpone un aimagen de Wally en distintas posiciones siendo la etiqueta de cada imagen su ubicación.

Si se quisiera emplear las imágenes originales de 500x500 bastaría con ampliar la arquitectura de la red añadiendo paquetes de capas convolucionales y de *pooling* hasta poder trabajar con imágenes de ese tamaño.

2.4. Modelos invariantes

La definición de invarianzas es general a cualquier transformación pero, teniendo en cuenta el contexto, se define en el marco de las transformaciones en términos de traslación. Dada una imagen A y el grupo de transformaciones traslacionales G, se dice que una función o sistema f

es **invariante a la traslación** si al aplicar una transformación g a la imagen A, la salida de la función f no modifica la salida [[33], [34], [35]]. Es decir:

$$B = f(A)$$
$$B = f(g(A))$$

Lo que se quiere obtener es lo que muestra la Figura 2.6, que una transformación en el dominio de la imagen nos lleve al mismo punto al que nos llevaba la imagen sin transformar.

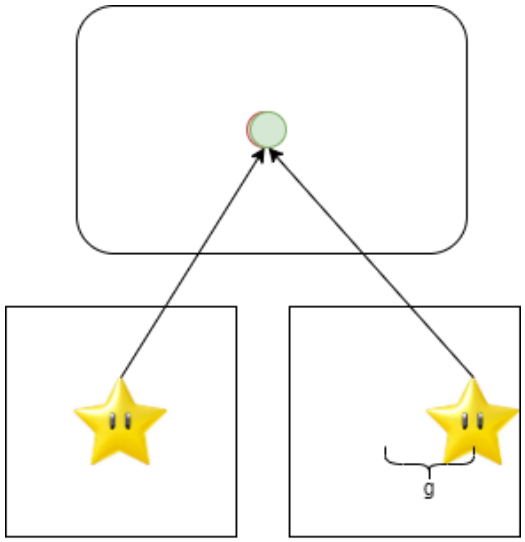


Figura 2.6: Diagrama de modelo invariante, en el que al trasladar la imagen de entrada la representación en el dominio interno es la misma que la de la imagen sin trasladar.

Esta característica es especialmente útil en problemas como el de la segmentación o la clasificación, donde lo importante es reconocer el objeto independientemente de dónde se sitúe en la imagen. Si se observa de nuevo el diagrama de la Figura 2.6, sería deseable que el clasificador etiquetara ambas imágenes de con la etiqueta *estrella*. Para que esto suceda, es necesario que la red sea **invariante**.

Modelos Invariantes Básicos

A lo largo de la literatura, se pueden encontrar muchas posibles soluciones a los problemas de invarianzas. No obstante, hay una solución que se encuentra siempre en primer lugar en cuanto a sencillez e intuición: El uso de *Global Average Pooling* (GAP) [51]. Esta capa, colocada al final de las capas convolucionales, realiza un resumen de un único valor para cada canal del que se disponga, haciendo que el resumen no dependa de la posición de lo que se detectaba en el canal.

En primer lugar, se hace una prueba sobre cómo funciona una red neuronal básica a la que al final no se le ha colocado la capa GAP. Este modelo se ha entrenado con unas imágenes MNIST -Sección 4.2- pero agrandadas, a un tamaño de 56x56 pixels sin desplazar. Una vez entrenado con las imágenes estáticas, se ha testeado en las imágenes desplazadas para probar su resistencia a las traslaciones. La representación gráfica del modelo se puede ver en la Figura 2.7 y cuenta con una sola capa convolucional -de 32 filtros de 3x3- seguida de la capa *flatten* y una última capa densa con 10 nodos para reconocer las 10 posibles etiquetas.

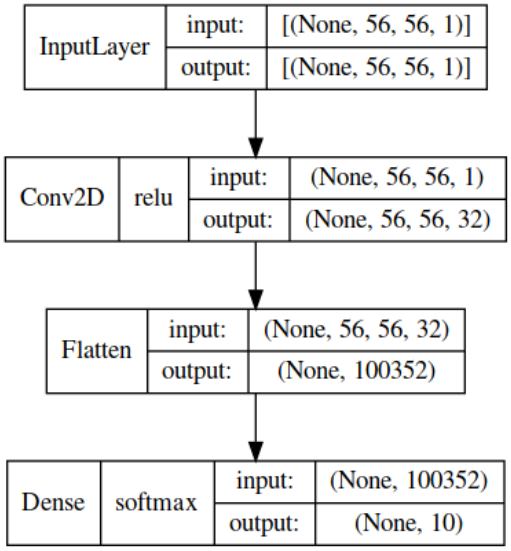


Figura 2.7: Arquitectura básica empleada sin inclusión de invarianzas al incluir una capa final de flatten en lugar de una GAP.

La precisión de esta red se puede visualizar en la Figura 2.8. Vemos como tiene una forma claramente acampanada con el máximo en el punto (0,0), punto donde no hay traslación. Conforme la imagen se va moviendo por el fondo, la precisión va cayendo hasta llegar al mínimo en los puntos donde se ha desplazado la imagen en la orientación diagonal. Teniendo en cuenta que esta red no cuenta con ningún mecanismo que dote de la característica de invariante, era de esperar que la red funcionara peor cuanto mayor sea el movimiento.

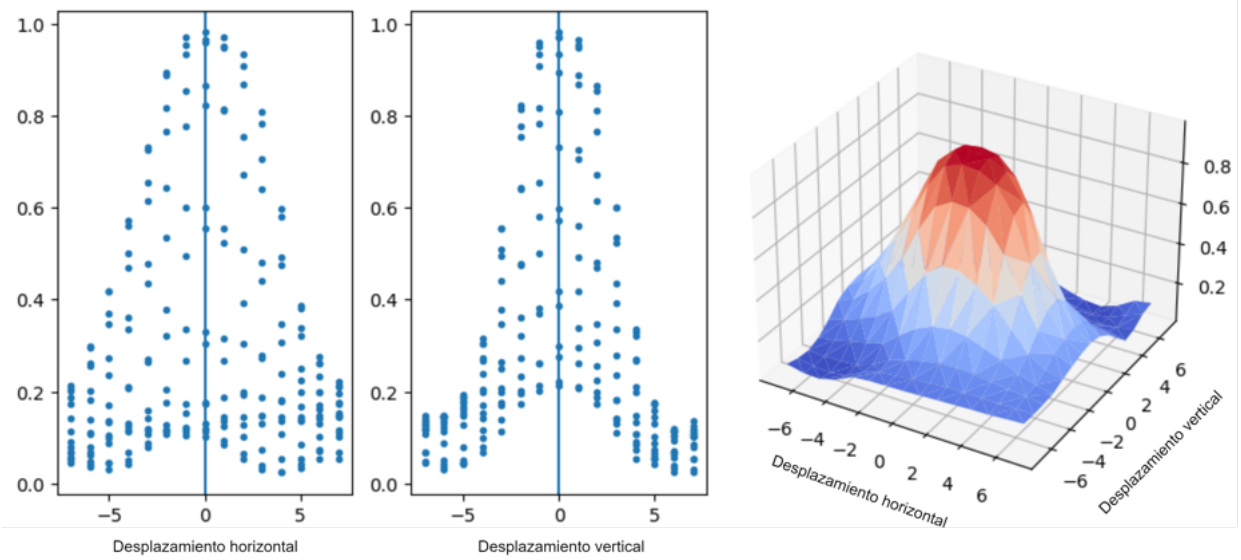


Figura 2.8: Precisión de la red básica no invariante a la traslación descrita en 2.7. La red está entrenada únicamente con imágenes centradas y se evalúa en imágenes desplazadas. En concreto con el conjunto de datos MNIST descrito en 4.3.

Una vez explorada la vía original (no invariante), creamos otra red en la que sí coloquemos la capa *Global Average Pooling*. La arquitectura se puede observar en la Figura 2.9. Esta red, en esencia, es igual a la anterior. Cuenta con las mismas configuraciones en las capas convolucional y densa, pero esta vez entre las dos hay un GAP.

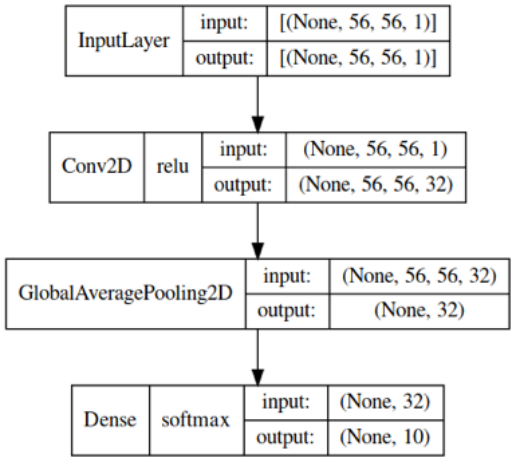


Figura 2.9: Arquitectura básica incluyendo invarianzas al intercambiar la capa *flatten* por una GAP, aumentando el campo receptivo, captando las traslaciones y proyectándolas al mismo punto del espacio de representación.

Después de entrenar esta red de la misma manera que la red básica no invariante, obtenemos el mapa de precisiones de la Figura 2.10. En este caso vemos como la precisión obtenida en todas las traslaciones es la misma, por lo que estamos hablando de una red invariante. No obstante, la precisión que se obtiene en este caso es muy inferior a la conseguida con la red básica sin GAP. Esto se debe a la cantidad de parámetros que le llegan a la densa. Si antes de una densa colocamos la capa *flatten*, la cantidad de parámetros que le entran a la densa es mucho mayor -porque estira cada filtro, entrando un dato por componente de filtro y canal- que cuando colocamos la GAP -donde le llegará un parámetro por filtro-.

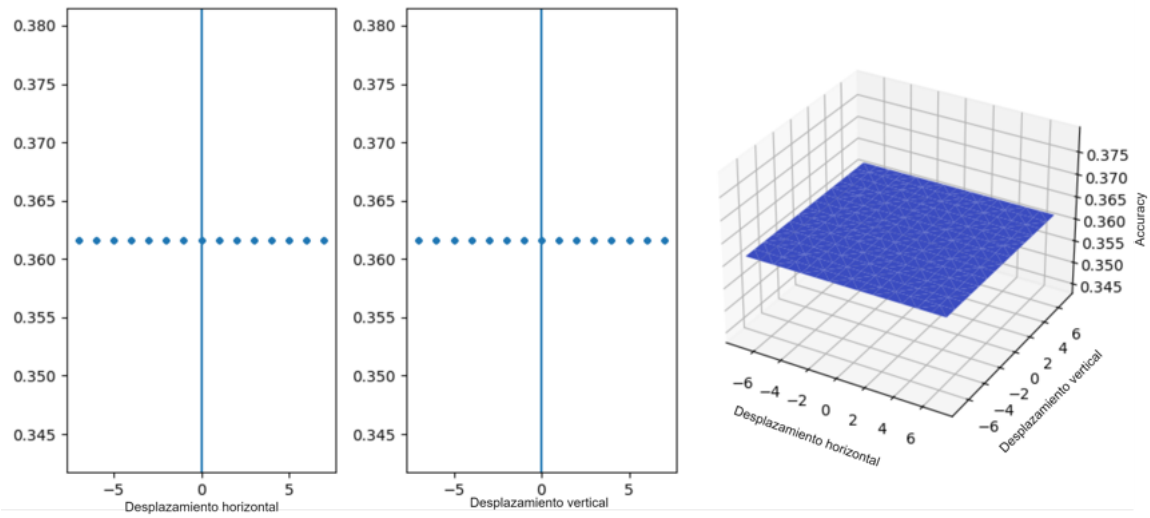


Figura 2.10: Precisión de la red básica invariante a la traslación descrita en 2.9. La red está entrenada únicamente con imágenes centradas y se evalúa en imágenes desplazadas. Esta vez devuelve la misma precisión en todos los puntos. También está entrenada con el conjunto de datos MNIST descrito en 4.3.

Este problema se puede intentar subsanar aumentando la capacidad de la capa convolucional, con más filtros y tamaños de filtro más grandes. Para explorar esta vía, hemos realizado un *sweep* en *wandb* -herramientas explicadas con más detalle en la Sección 3.4- probando diferentes valores para estos parámetros. Además, también se ha probado a añadir capas de *BatchNorma-*

lization, probando a eliminar el parámetro del *bias* o añadiendo a la función de coste términos regularización. Como ninguna de estas pruebas devuelve el resultado esperado y solo empeora el rendimiento, no se tienen en cuenta en el barrido de hiperparámetros. Las diferentes pruebas pueden verse en la Figura 2.11, donde cada línea indica una configuración diferente de tamaño de filtro -*kernel_size*- y número de filtros -*filters*- que termina en una precisión final -*epoch/val_accuracy*-.

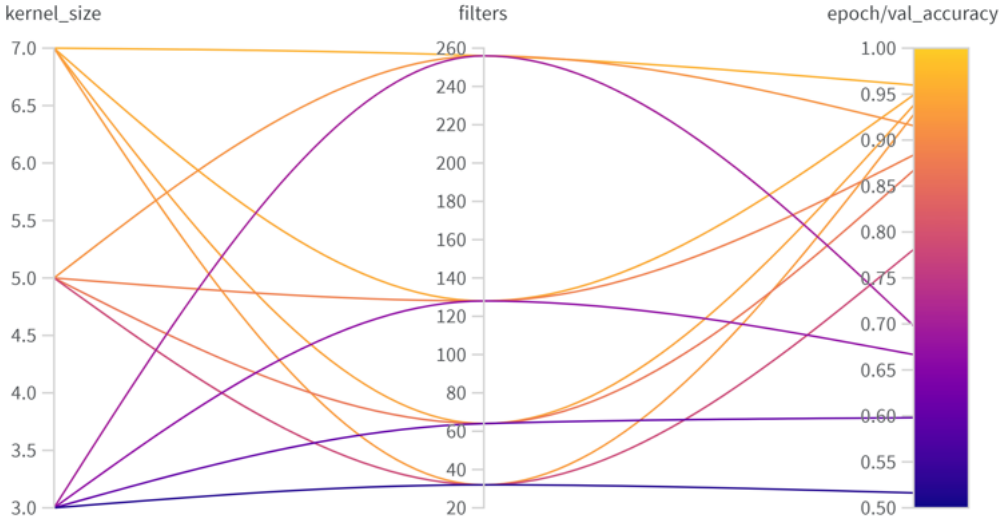


Figura 2.11: Optimización de hiperparámetros en el modelo básico invariante a la traslación. Modificando los siguientes parámetros para ver cómo impacta la configuración en el rendimiento de la red: Tamaño del filtro (*kernel_size*) variando entre los tamaños tres, cinco y siete y la cantidad de filtros (*filters*) variando entre 32, 64, 128 y 256.

Como cabría esperar, un mayor número de filtros combinado con un tamaño mayor de los mismos, devuelve una mayor precisión. Se alcanza, incluso la precisión obtenida en una red no invariante. Con la diferencia de que este modelo requiere de un mayor y más costoso entrenamiento, por la cantidad de parámetros extra que hay que ajustar, pero a cambio es totalmente invariante a la traslación.

Si a este modelo ya optimizado, se le pasa las imágenes de MNIST -Sección 4.2- pero con el dígito saliendo de la imagen. Lo esperado es que tenga un peor rendimiento debido a la incapacidad del modelo de reconocer completamente los números que aparecen en ellas. No obstante, como cada dígito desaparece de la imagen de una forma distinta, el modelo podría devolver extraños resultados que mejoran puntualmente su precisión. En la Figura 2.12 se muestra su superficie de precisión. Se puede ver como en las esquinas, una vez que el dígito ha salido parcialmente de la imagen, el rendimiento del modelo baja.

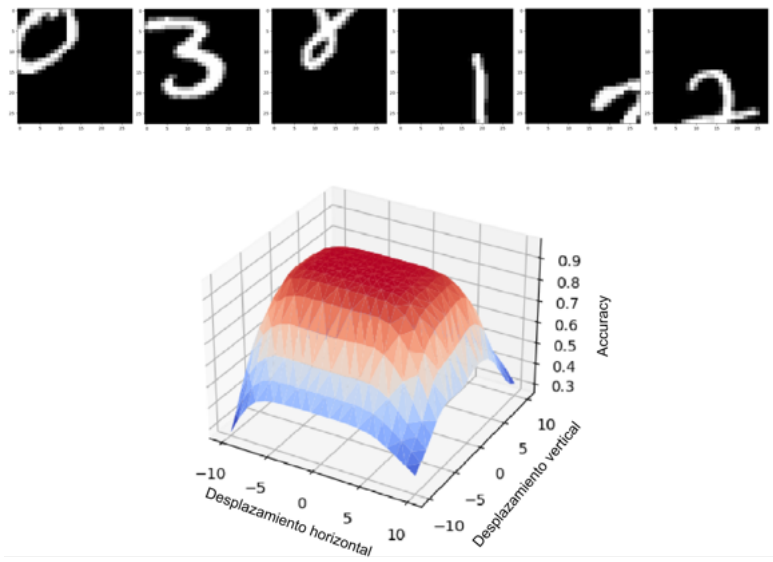


Figura 2.12: Precisión de la red básica invariante a la traslación. La red está entrenada únicamente con imágenes centradas y se evalúa en imágenes desplazadas pero en las que el dígito no aparece de manera completa. El conjunto de datos utilizado es el de la primera parte de la Sección 4.3.

Capítulo 3

Metodología

Como se ha comentado en la Sección 1.1, las posibles aplicaciones que se les pueden dar a las características de invariancia y equivariancia son muchas. Como explorarlas todas saldría de los límites del proyecto, se ha optado por la elección de dos aplicaciones especialmente relevantes en el contexto de la Inteligencia Artificial: Problemas de clasificación y problemas de calidad de imagen (*Image Quality Assessment* - IQA).

En este Capítulo se va a ampliar lo comentado en la Sección 1.5, explicando más detalladamente los pasos seguidos en los experimentos y su motivación.

3.1. Problemas de clasificación

En el contexto del aprendizaje automático y aprendizaje profundo, los problemas de clasificación han estado siempre muy presentes. Desde situaciones médicas, con la clasificación de pacientes [52], pasando por clasificación de audio [53]. Dentro de toda esta variedad de aplicaciones, la clasificación de imagen es una de las que más interés atrae por su reaplicación en multitud de escenarios. En este contexto, se ha hecho un breve resumen en la Sección 2.4, que se extenderá ligeramente en esta.

Dentro del marco de trabajo lo que se pretende es conseguir una situación como la que muestra el diagrama de la Figura 3.1. Una en la que, para un problema determinado -en este caso, MNIST , Sección 4.1- independiente de la imagen trasladada de entrada, se reconozca como la imagen no distorsionada y a todas ellas se las etiquete como lo haría con la original, es decir, que cuente con la característica de ser invariante -Sección 2.4-.

En este sentido, se han hecho algunas pruebas en la Sección 2.4 y se pretende hacer un estudio más en profundidad tanto de arquitecturas básicas -modificando ligeramente la ya vista- y probando arquitecturas más potentes y profundas.

3.1.1. Modificaciones ligeras

En la Sección 2.4 se ha comentado la importancia de la inclusión de la capa GAP para conseguir redes invariantes a la traslación. No obstante, usar una capa GAP implica necesariamente hacer una reducción muy poco gradual de la dimensionalidad, pues se pasa de la dimensión de la imagen a un solo valor. Lo habitual es que la red vaya reduciendo la dimensionalidad de

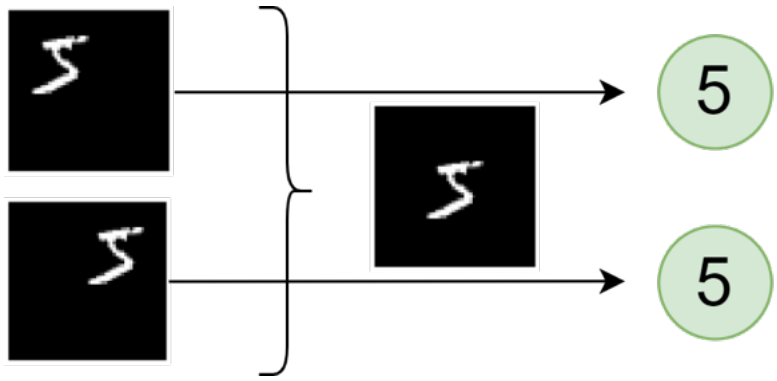


Figura 3.1: Diagrama del problema de clasificación. El objetivo es que independiente de la imagen trasladada de entrada, se reconozca como la imagen no distorsionada y a todas ellas se las etiquete como lo haría con la original

manera suave usando capas como la de *max pooling* [48]. Vamos a ver un pequeño experimento con el que se pretende ilustrar cómo impacta en el resultado el hecho de que la red vaya realizando internamente operaciones de *pooling* pero que al final acabe con una GAP intentando conseguir que la transformación sea invariante a traslaciones.

Un diagrama de lo que se espera que ocurra se puede visualizar en la Figura 3.2. Partiendo de la imagen original centrada -de 8x8 píxeles- se le aplica un *max pooling* de tamaño dos, lo que nos devuelve es un mapa de características de tamaño 4x4, representado justo debajo. Después, ese mapa de características pasará por la capa GAP y hará un resumen de un solo valor, representado con un valor de color morado. Ahora bien, conforme vamos desplazando la imagen a través del fondo, el filtro del *max pooling* cae sobre otras partes de la imagen y obtiene mapas de características y valores del GAP diferentes -por ejemplo, el valor rosa-. A medida la imagen se siga desplazando, el filtro volverá a caer en las posiciones en las que caía en la imagen centrada y recuperará los resultados obtenidos. Esta situación se dará en aquellos desplazamientos que sean divisibles entre el tamaño del *pooling*. Para un filtro de tamaño dos, devolverá los mismos resultados en desplazamientos pares y distintos para los impares. Para uno de tamaño tres, devolverá resultados iguales en desplazamientos divisibles entre tres y así sucesivamente.

Para este apartado, en el Capítulo 5 se hacen pruebas con capas *max pooling* de tamaño dos, tres y cuatro.

3.1.2. Arquitecturas preentrenadas

En este apartado se explican las arquitecturas seleccionadas para probar los problemas de invarianzas traslacionales en el ámbito de la clasificación. Se han seleccionado dos familias de arquitecturas por ser las más famosas y por comprobar si las diferencias entre ambos modelos terminan por impactar de forma diferente en los resultados.

Ambas redes están ya preentrenadas, es decir, cuentan con unos pesos internos ya definidos que no requieren de entrenamiento extra. No obstante, las redes fueron entrenadas en conjuntos de datos que no se asemejan en nada a los que se pretende emplear, por lo que será necesario hacerlas específicas a nuestros datos. A esta técnica se la conoce como *transfer learning*, y consiste en dejar los pesos por defecto en todas las capas internas y solo modificar los pesos de la última. De esta forma, se reentrenan solo esas capas y adquieren las características específicas

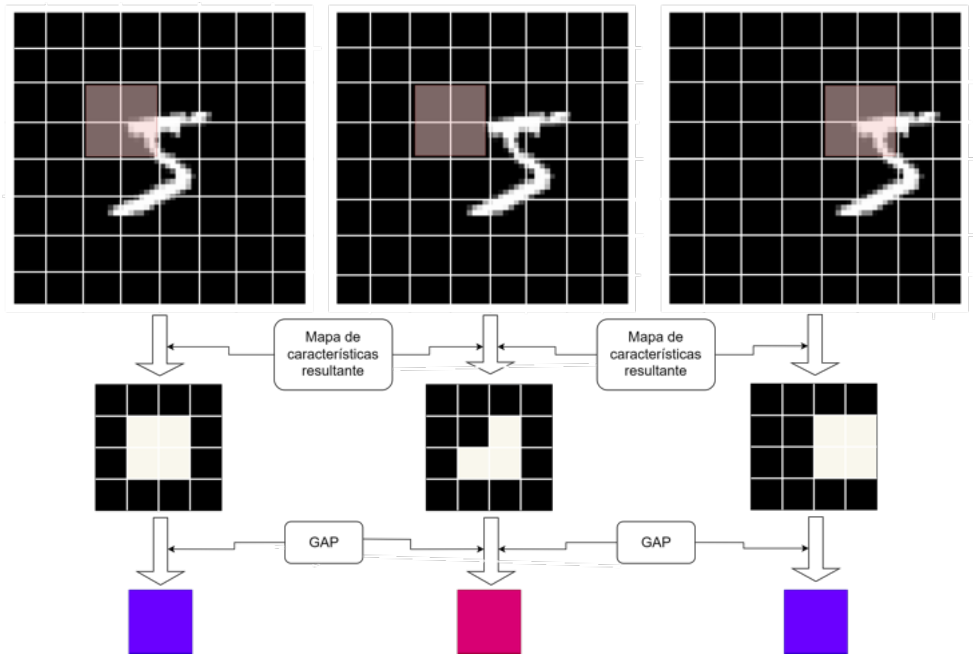


Figura 3.2: Diagrama del comportamiento de la capa *pooling* combinado con la capa GAP. En un inidio, la imagen centrada genera un mapa de caracterísiticas y, posteriormente, un valor resumen tras el GAP. Cuando desplazamos un píxel la imagen original, el *pooling* ya no cae en las zonas en las que caía, genera un mapa de características diferente y el GAP un nuevo valor resumen. Por último, al desplazar dos píxeles, el *pooling* cae en las mismas posiciones que en la imagen centrada y el GAP resumen de la misma forma que al principio.

del problema, como el número de clases.

VGG

En 2014, con el fin de poder participar en el *Desafío de Reconocimiento Visual a Gran Escala de ImageNet*, se presentó el modelo VGG [54]. Desde entonces, y debido a los buenos resultados obtenidos, se posicionó como un modelo de referencia en el campo de la visión por computador. Un diagrama de su funcionamiento interno se puede visualizar en la Figura 3.3.

Toda las redes VGG tienen un comportamiento casi idéntico. Consta de una entrada de preprocesado de la señal, unos bloques de extracción de características -compuesta por capas convolucionales y de *max pooling*- y por último una capa totalmente conexa para poder predecir la señal. En función del número de bloques de los que disponga la red, será una VGG16, VGG17, VGG19, etc.

ResNet

ResNet (Residual Network, 2015) [55] es una familia de redes convolucionales profundas de procesado de imágenes. La caracteristica que diferencia sustancialmente estas redes del resto de redes profundas, es la existencia de *skip connections*. Estas conexiones permiten a la información extraída de las capas saltarse estadios intermedios y conectarse a capas situadas más abajo en la red, lo que permite que la información (y el gradiente) fluya más fácilmente.

Toda la familia tiene un comportamiento casi idéntico, tanto entre ellas, como con otras familias. De hecho, su funcionamiento, dejando a un lado las *skip connections*, es igual que en VGG. Consta de una entrada de preprocesado de la señal, unos bloques de extracción de características y por último una capa totalmente conexa. En función del número de bloques de

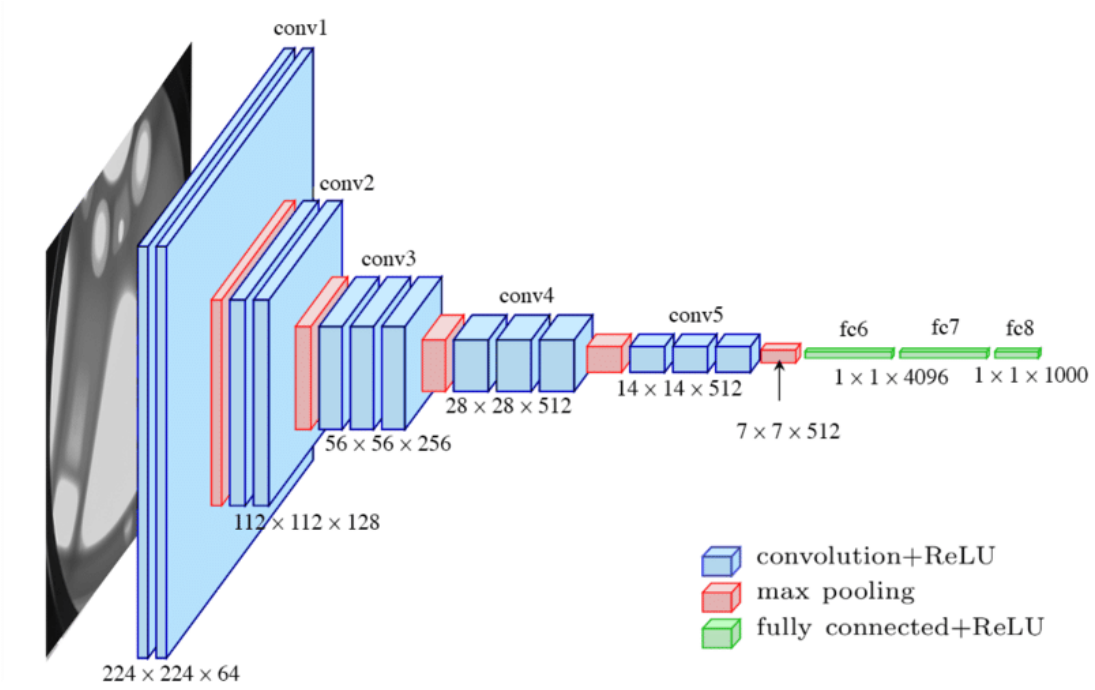


Figura 3.3: Diagrama interno de VGG16. Extraido de [54]

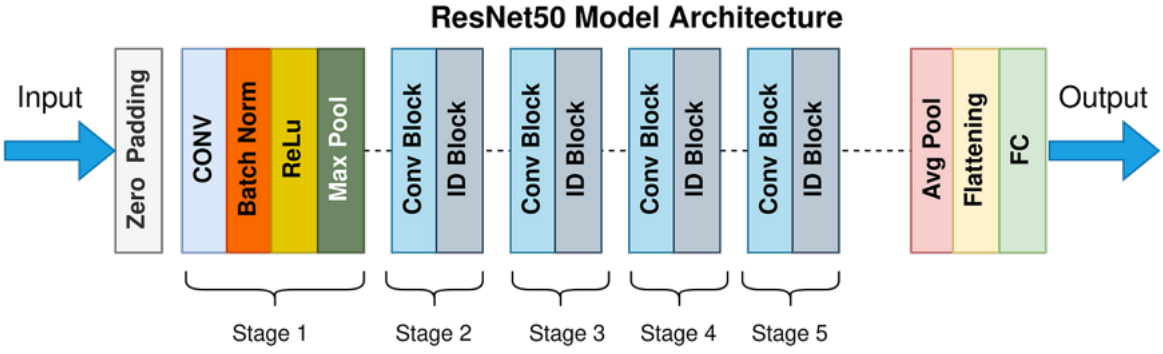


Figura 3.4: Diagrama interno de ResNet50. Extraido de ResNet50

los que disponga la red, será una ResNet50, ResNet100, ResNet150, etc. Lo que sí que se debe remarcar en este aspecto es la inclusión de una capa de *average pooling* al final de la red, que podría hacerla más robusta a invarianzas traslacionales, razón por la que se ha seleccionado esta red. Es importante comprobar cómo influye en los resultados tanto la inclusión del GAP como la de las conexiones entre capas.

3.2. Problemas de calidad de imagen

Las imágenes sufren muchas modificaciones en multitud de pasos de su procesado: en la adquisición, transmisión, almacenaje o debido a condiciones externas, como el movimiento del dispositivo de captura. El *Image Quality Assessment* (IQA) es una técnica que predice la calidad perceptual de las imágenes, es decir, intenta asemejar sus evaluaciones a las que realizaría un humano. Es una téctica que se aplica a diferentes trabajos, como son restauración de imágenes [56], la superresolución de imágenes [57] y la recuperación de imágenes [58]. Es importante señalar que la percepción humana depende también de muchos factores, como la estrucutra de

la imagen o el contexto, lo que hace de esta problemática una de especial dificultad.

Dependiendo de la información de la que se disponga, se pueden clasificar los modelos de IQA en las siguientes categorías:

- **Sin referencia:** Los modelos deben estimar la calidad de una imagen distorsionada sin conocer la imagen original. Debido a la complejidad de estos modelos, muchos métodos se centran en la estimacion de ciertas distorsiones concretas [59], con lo que se reduce mucho su uso. En la aplicación en la que suelen despuntar este tipo de modelos es en el de creación de *rankings* de imágenes [60]. En este tipo de problemas, lo que se le pasa al modelo es una serie de imágenes -no necesariamente relacionadas entre ellas- y en función de la aplicación, se etiquetarán de una forma u otra.

  Un ejemplo de las aplicaciones relacionadas con la calidad de imagen sin referencia se muestra en la Figura 3.5. En la parte superior, al modelos se le pasaría esa seríe de imágenes y la tarea consistiría en asignarles una valoración, en este caso de 0 a 100. Una vez valoradas, se haría el *ranking*. En este ejemplo, todas las imagenes están relacionadas y son la misma pero con diferente nivel de distorsión. No obstante, las imágenes pueden ser todo lo dispares que se desee. En la parte inferior, el objetivo consiste en agrupar las imágenes por calidad perceptual, lo que se conoce como agrupamiento libre. En el ejemplo, etiqueta con C1 las imágenes naturales y con C2 las que no lo son.

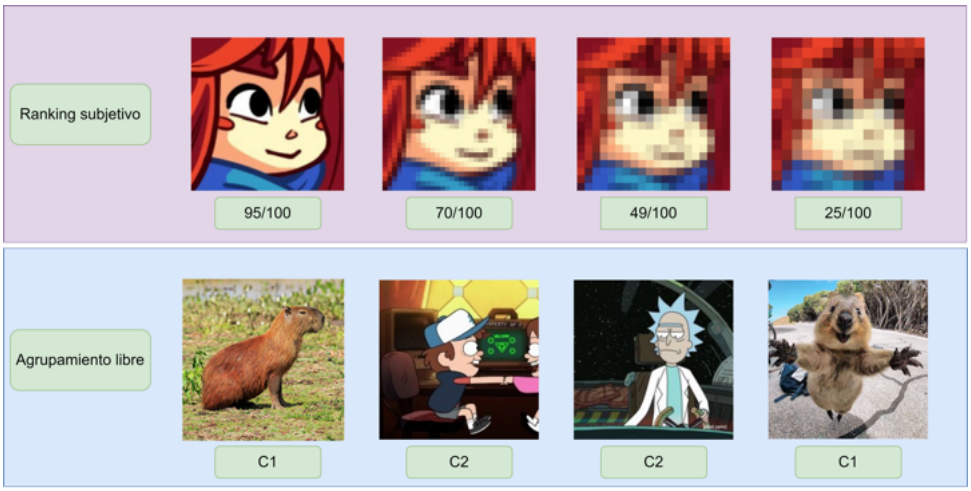


Figura 3.5: Ejemplo de problema IQA sin referencia. En la parte superior, una problemática de *ranking* en la que se debe ordenar una serie de imágenes en función de su calidad. En la parte inferior, una de *clustering* en la que se debe agrupar las imágenes en función de características comunes.

- **Con referencia completa:** En estos modelos, además de la imagen modificada, se muestra también la imagen original con la que debe compararse. De esa manera se evalúa cuánto desplaza la modificación la representación interna de la imagen. Un ejemplo se puede ver en la Figura 3.6, donde la problemática reside en cuál de las dos imágenes dista menos de la original, qué distorsión, desde el punto del observador, es menos molesta. Como el modelo cuenta siempre con la imagen original desde la que se ha realizado la modificación, se puede utilizar para multitud de distorsiones.

  Un ejemplo sencillo de cómo trabajan estos modelos se puede apreciar en la Figura 3.6. Partiendo desde la imagen original, se crean dos imágenes modificadas -imagen 1 e imagen 2- con el mismo tipo de distorsión -superposición de artefactos- pero con niveles diferentes

de impacto -coinciden o no con el color base-. Lo que se pretende ahora es saber qué imagen -de las dos modificadas- se asemeja más a la original imitando lo que evaluaría un observador humano. Una forma de resolver este problema es utilizar un modelo que transforme la imágen y la lleve a un espacio de características en el que sí tenga sentido medir distancias. En el caso de la Figura 3.6, la distancia que aparece es la devuelta por PerceptNet [61] -Subsección 3.2.1- que sí parece representar lo que diría un humano. No obstante, si por ejemplo usamos SSIM [62] -Subsección 3.2.1-, nos devuelve que la imagen con cuadrados rojos está más próxima a la original. La elección de la métrica es de vital importancia para según qué aplicaciones y necesidades. En general, todavía se sigue trabajando, desde diferentes grupos de investigación, en diseñar una métrica capaz de trabajar correctamente en todos los contextos.

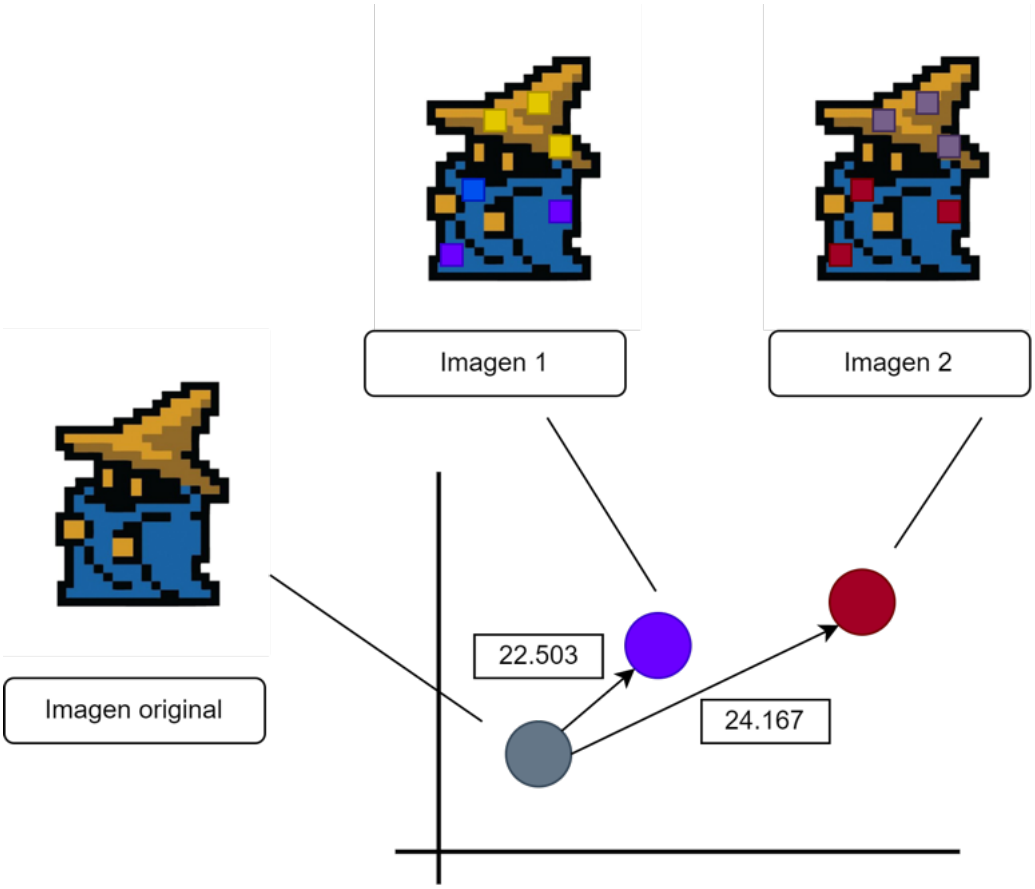


Figura 3.6: Ejemplo de problema IQA con referencia completa. Dada una imagen original, se crean dos modificaciones a las que se mide la distancia en el dominio interno utilizando la métrica PerceptNet.

En este proyecto, como es indispensable contar con la imagen sin modificar para hacer constar su modificación y comprobar que dicha modificación es invariante o equivariante, siempre se utilizarán técnicas de IQA con referencia completa. En este apartado del problema, lo que se busca en concreto es la característica de invarianza. Se pretende que la calidad referida a la imagen original y a la trasladada sea prácticamente la misma, pues la distorsión no cambia la percepción de la imagen.

Para la evaluación, se dispone de múltiples métricas y modelos -que se explicarán más adelante, Subsección 3.2.1- con los que compararemos resultados dependiendo de la traslación y la métrica.

3.2.1. Métricas y modelos

A continuación se listan y se detalla el funcionamiento de las métricas y modelos seleccionados para la problemática de la calidad de imagen en términos de desplazamiento. Para cada una de ellas, se probarán diferentes desplazamientos y se guardarán los valores de las métricas, para los que luego compararemos su desempeño.

El objetivo de absolutamente todos los modelos que se van a explicar a continuación es tener una respuesta lo más similar posible a la que daría un humano, es decir, que ambas respuestas estén lo más correladas posibles. A la respuesta del humano se le denomina MOS (Mean Opinion Score), que funciona como una media sobre todos los humanos a los que se les presentan las imágenes. Depende del experimento que se esté realizando, a veces se les pide hacer un *ranking* entre todas las imágenes modificadas para saber cuáles se parecen más o menos; y otras que decidan entre las categorías *buena, regular, mala*, etc...

A modo de introducción a la problemática, se muestra en la Figura 3.7 las correlaciones de todas las métricas que se van a estudiar en los siguientes apartados con el MOS para el conjunto de datos de TID 2013[63], un conjunto habitual en las pruebas de calidad de imagen. Ya solo con ella, se intuyen qué metricas tienen un mejor desempeño. No obstante, en las subsecuentes secciones se detalla el funcionamiento y el rendimiento para cada una de ellas.

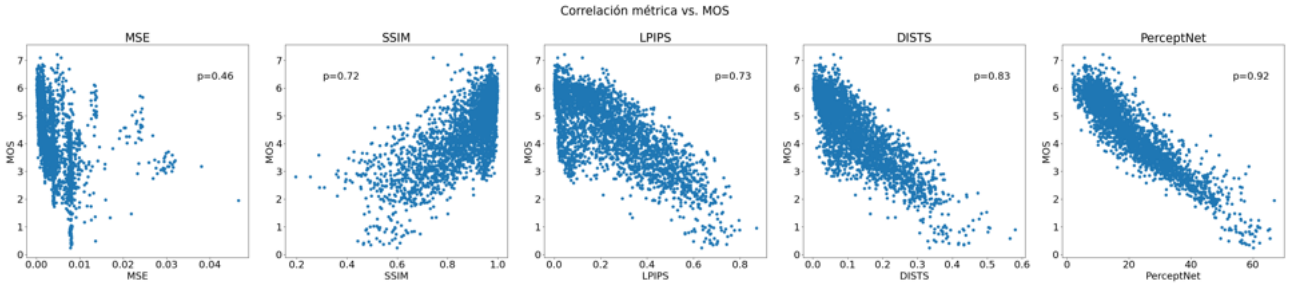


Figura 3.7: Correlación de cada una de las métricas seleccionadas con el MOS junto con su correlación de Pearson. La métrica que mejor resultados obtiene es PerceptNet.

Mean Square Error - MSE

El Error Cuadrático Medio (MSE por las siglas en inglés) es un estimador que mide la media de los errores al cuadrado producidos entre lo estimado y el estimador. Siendo $\hat{Y}$ un vector de predicciones e Y el vector que se intenta estimar, se define el MSE del predictor como se indica en la Ecuación 3.1.

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}\left(Y_i - \hat{Y}_i\right)^2 \tag{3.1}$$

No obstante, cuando se pretende calcular el MSE en el contexto de las imágenes, la definición cambia ligeramente y la operación se realiza para cada píxel de ambas imágenes. Dadas dos imágenes A, B se toma el cuadrado de la diferencia entre cada píxel de A y el píxel correspondiente de B, sumados y divididos por el total de píxeles -de una sola imagen-.

Es la métrica de calidad con referencia completa más sencilla y utilizada porque es fácil de calcular, tiene un significado físico claro y es matemáticamente conveniente en el contexto de

la optimización -como en el ejemplo de los Autoencoders [64]-. Pero no se ajustan muy bien a la calidad visual percibida, teniendo en cuenta que únicamente considera el valor de cada píxel [65]. Otras métricas perceptuales tienen en cuenta otras características de la imagen como la luminancia, el contraste, frecuencias altas y bajas, etc.

Con todo esto, se puede concluir que utilizar el MSE como métrica perceptual es una mala elección. Si bien es cierto que se ha estado empleando durante décadas, queda claro que no es lo más adecuado si queremos medir calidad de imagen [66, 67]. Dos imágenes que presenten el mismo MSE podrían reflejar distorsiones muy diferentes y eso es lo que se pretende medir con las métricas perceptuales [68]. Un ejemplo de ello se puede observar en la Figura 3.8, donde se muestra una fotografía original de Einstein y cinco distorsiones de la misma indicando su MSE. Todos los valores de la métrica son similares, pero resulta evidente para cualquier observador que las dos últimas tienen una peor calidad perceptual.

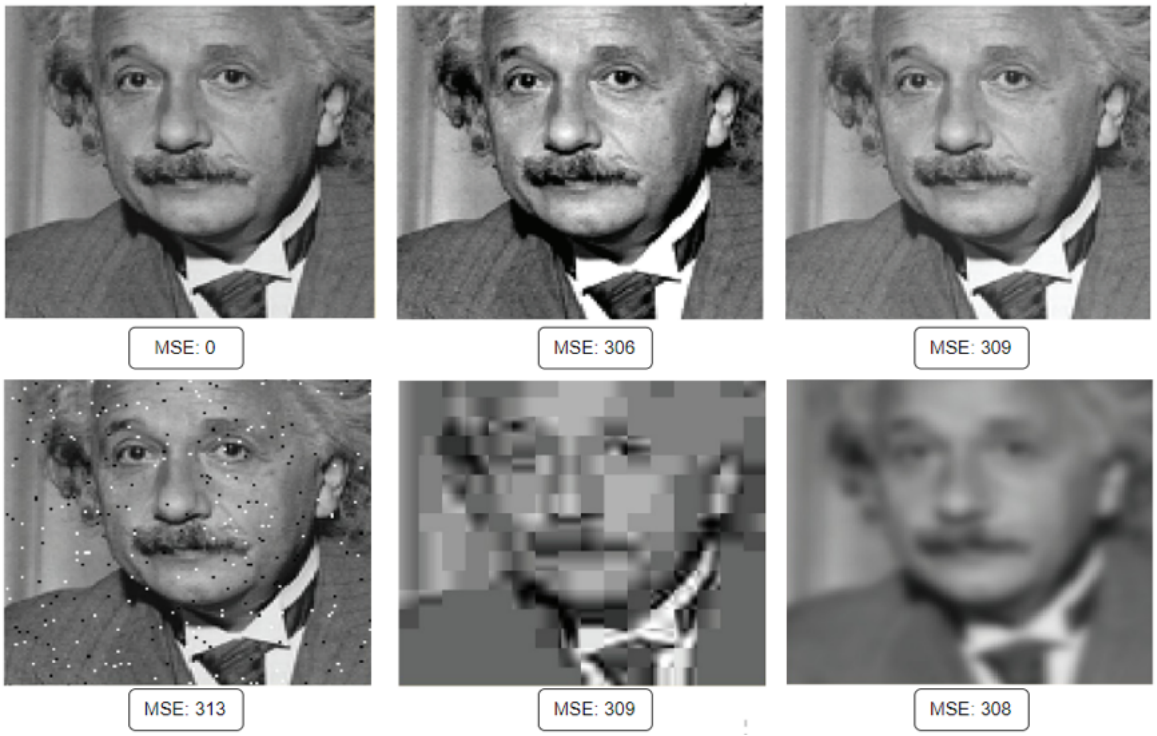


Figura 3.8: Fotografía original de Einstein (arriba izquierda) y cinco distorsiones de la misma indicando su MSE. Todos los valores de la métrica son similares, pero resulta evidente para cualquier observador que las dos últimas tienen una peor calidad perceptual. Extraido de [69]

Structural Similarity Index Measure - SSIM

Teniendo en cuenta que la métrica MSE no es la más adecuada para evaluar la percepción humana, y en vista de la necesidad de usar una que sí la represente realmente; en 2004 se presenta la métrica SSIM [62]. Nació como una alternativa viable al uso del MSE y al resto de métricas perceptuales que fueron saliendo esos años [69], y su uso se ha mantenido hasta prácticamente hoy. Su funcionamiento se muestra en el diagrama de la Figura 3.9, donde para dos señales -la imagen original y distorsionada en nuestro caso- se calculan tres comparaciones diferentes: Luminancia, contraste y estructura. Con ellas, se intenta establecer un índice de similaridad, que indica cuán parecidas son ambas señales.

Para ilustrar esto último, se puede observar la Figura 3.10, donde ahora se comparan el MSE con el SSIM. Para imágenes con mismo MSE, SSIM devuelve diferentes valores que esta

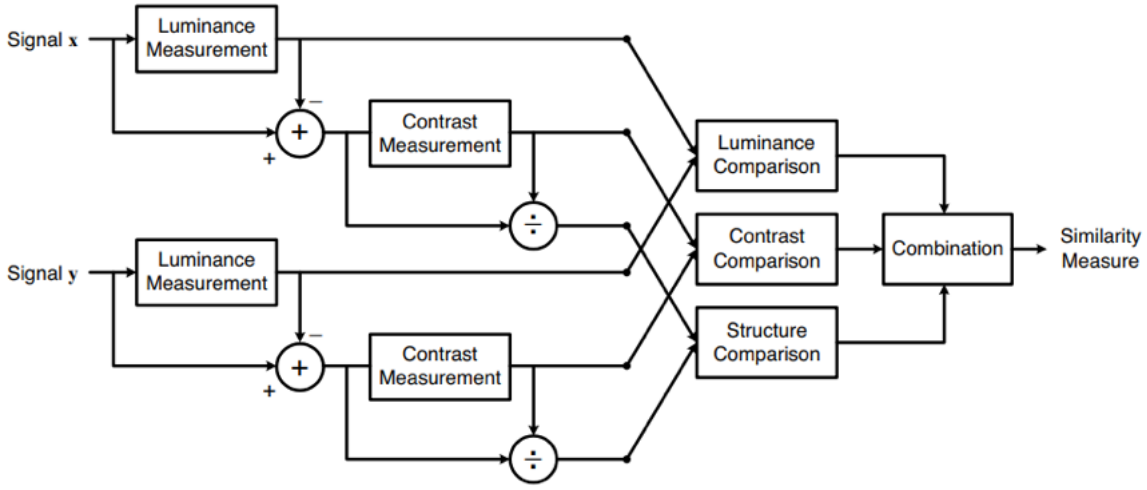


Figura 3.9: Diagrama del funcionamineto interno de SSIM. Extraido de [62]

vez sí parecen reflejar las valoraciones humanas, otorgándole a las dos últimas un valor más bajo que al resto.

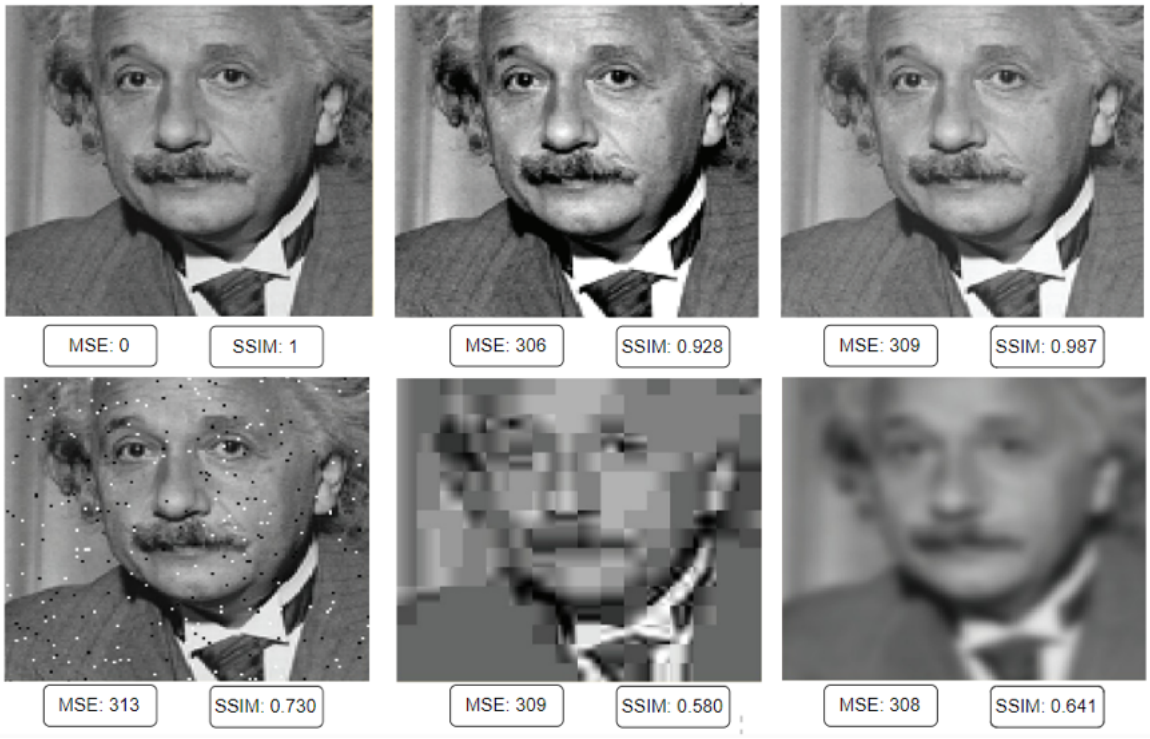


Figura 3.10: Fotografía original de Einstein (izquierda) junto con dos distorsiones de traslación, hacia la derecha (centro) y hacia la izquierda (derecha) indicando tanto su MSE como su SSIM. Extraido de [69]

Siguiendo con la temática que ocupa este proyecto, las invarianzas traslacionales, las dos métricas anteriores ya han sido empleadas para evaluar desplazamientos y comprobar la métrica resultante. En la Figura 3.11 se puede ver un ejemplo. A la izquierda, la imagen original, en el centro, la misma imagen pero desplazada hacia la derecha y a la derecha, la imagen desplazada hacia la izquierda. Desplazamiento que queda casi imperceptible al ojo humano, pero que ya es suficiente para que el MSE se dispare y para que el SSIM de un valor muy por debajo de lo que se obtenía con los ejemplos de 3.10.

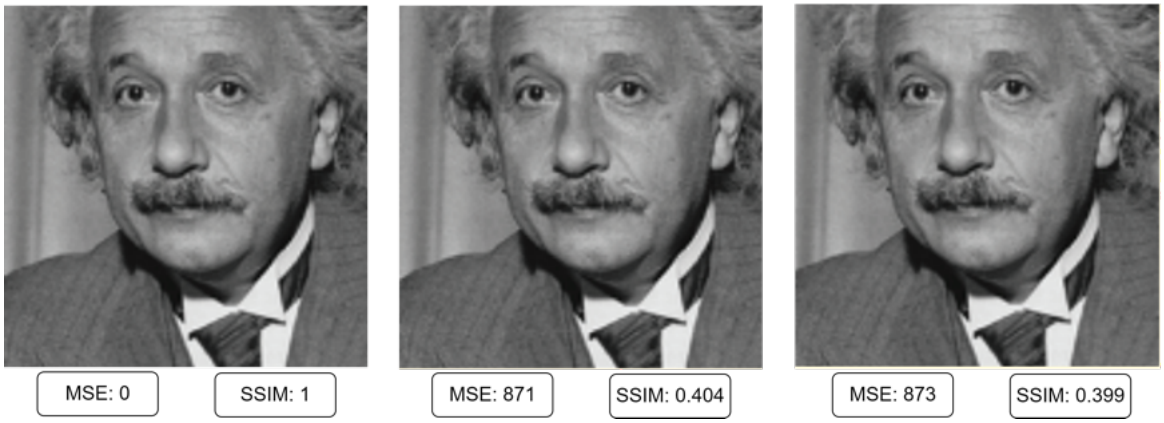


Figura 3.11: Fotografía original de Einstein (izquierda) junto con dos distorsiones de traslación, hacia la derecha (centro) y hacia la izquierda (derecha) indicando tanto su MSE como su SSIM. Extraido de [69]

De hecho, no solo se obtienen resultados deficientes con distorsiones traslacionales. Existen otras muchas distorsiones para las que SSIM no devuelve el valor deseado. Tanto es asi, que desde que salió en 2004, se ha tratado de modificar el índice para hacerlo más robusto -no solo a invarianzas traslacionales o rotacionales, sino en general-. Algunas de las variaciones más famosas son: FSIM [70], MS-SSIM [71] y CW-SSIM[72].

Por lo tanto, parece evidente que para las dos métricas más longevas -MSE y SSIM- las distorsiones en términos de desplazamientos, son distorsiones para las que son muy poco robustas. En este trabajo, se probará con ambas y se comprarán resultados con la literatura.

Learned Perceptual Image Patch Similarity - LPIPS

La métrica LPIPS [73] aparece en 2018 definida como una métrica perceptual con referencia completa. Lo que se propone es que la percepción humana no es una única función única con sentido propio, sino más bien la consecuencia de las representaciones visuales que predicen la importancia de las estructuras del mundo. Por ello, lo que se plantea es el uso de una gran base de datos con multitud de distorsiones aplicados a diferentes ámbitos: tareas supervisadas -clasificación-, no supervisadas -kMeans- o autosupervisadas -resolución de puzzles-. El diagrama de su funcionamiento se puede encontrar en la Figura 3.12 y cómo se obtiene la distancia entre la imagen original y la distorsionada.

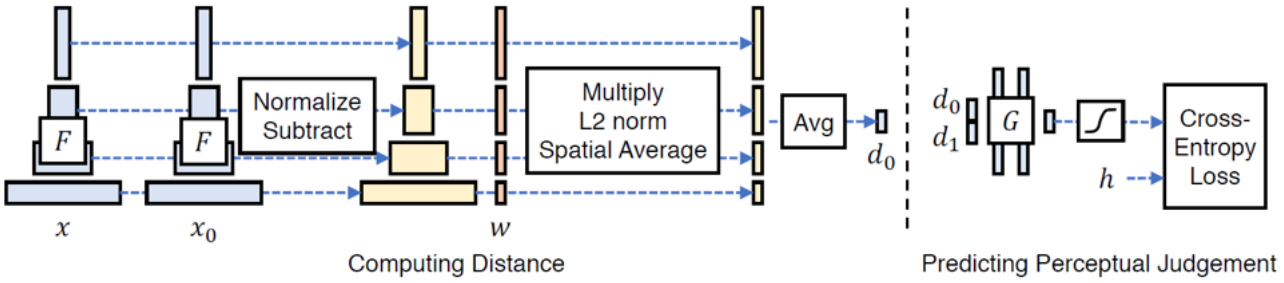


Figura 3.12: Diagrama interno del funcionamiento de LPIPS. Extraido de [73]

Para dos pares de imágenes -x y x_0- se extraen las caracterísiticas internas usando la red VGG -Subsección 3.1.2- y se normalizan por unidades en la dimensión de canal para cada

capa. Se escalan las activaciones por canal y se calcula la distancia l_2. Finalmente, se promedia espacialmente y se suman las activaciones por canal. Para simplificar, lo que hace es usar la red VGG para llevar las imágenes de entrada a distintos espacios internos (uno por cada capa) donde medir la distancia euclídea en ese nuevo espacio de características.

Deep Image Structure and Texture Similarity - DISTS

En 2020, se publica el paper sobre DISTS [?], una nueva métrica perceptual con referencia completa que procura subsanar los problemas de métricas anteriores. Este nuevo índice combina la sensibilidad de las distorsiones estructurales con la tolerancia a las texturas muestreadas en otros lugares de la imagen. En este caso lo que se hace es emplear la red VGG 3.1.2 y medir en las capas intermedias el índice SSIM 3.2.1 y cómo correlaciona la métrica a lo largo de la red.

PerceptNet

El último modelo que se va a estudiar es PerceptNet [61]. Publicado en 2020, la intención del modelo es tomar prestado el conocimiento que se tiene del sistema visual para plantear una arquitectura acorde, elegida para reflejar la estructura y las distintas etapas del sistema visual humano. Lo que propone es un sucesión de bloques de filtros lineales canónicos junto con la normalización divisiva que realizan una serie de operaciones perceptivas simulando a su vez la vía retina - LGN - córtex V1 [74]. Un diagrama de la arquitectura se puede observar en 3.13.

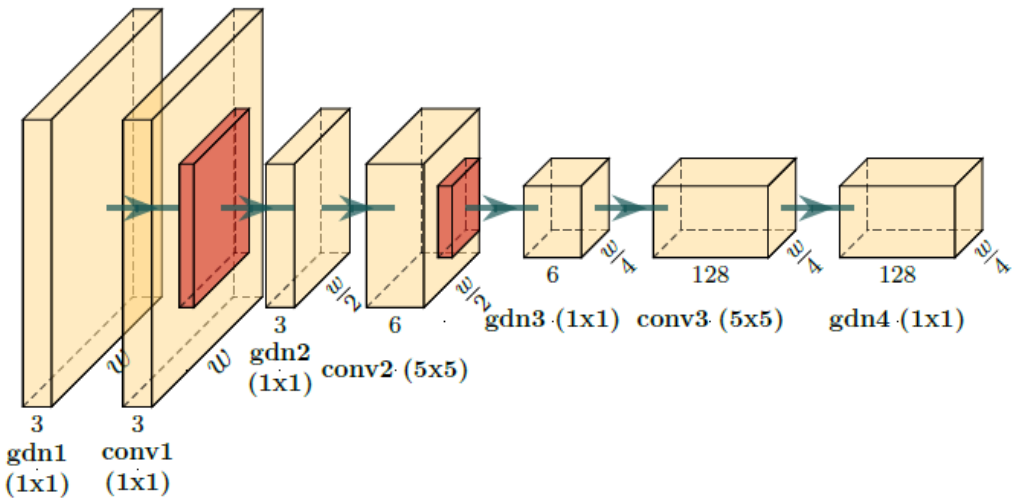


Figura 3.13: Diagrama interno de PerceptNet. Extraido de [61]

La red está entrenada para maximizar la correlación de *pearson* entre el MOS y la distancia l_2 entre las dos imágenes de entrada en el dominio transformado, como indica la Ecuación 3.2.

$$\max_f \rho\left(\|f(x) - f(d(x))\|_2, y\right) \tag{3.2}$$

3.3. Propuesta de mejora

Teniendo en cuenta todo el estudio realizado en ambas problemáticas y sobre la característica de invarianzas y equivarianzas, se propone una mejora sobre los modelos de clasificación. La mejora consiste en extraer características invariantes a diferentes escalas del modelo a través

de la incorporación de GAPs. En lugar de emplear las características que se extraen al final del modelo, se concatenan las salidas de los GAPs en estas escalas y se utilizan como entrada al clasificador final. Como la convolución es una operación equivariante, con la capa GAP se obtiene la invarianza, como se comenta en la Sección 2.4. Repitiendo esto en distintas etapas, se consigue un buen conjunto de características invariantes y de esta forma, se mantiene la invarianza hasta el final del modelo, como se explica en la Figura 3.14. Con esto se pretende añadir varios caminos invariantes a la red de forma que generalice mejor frente a las traslaciones. Las dos redes modificadas reciben el nombre de VGG16GAP y ResNet50GAP.

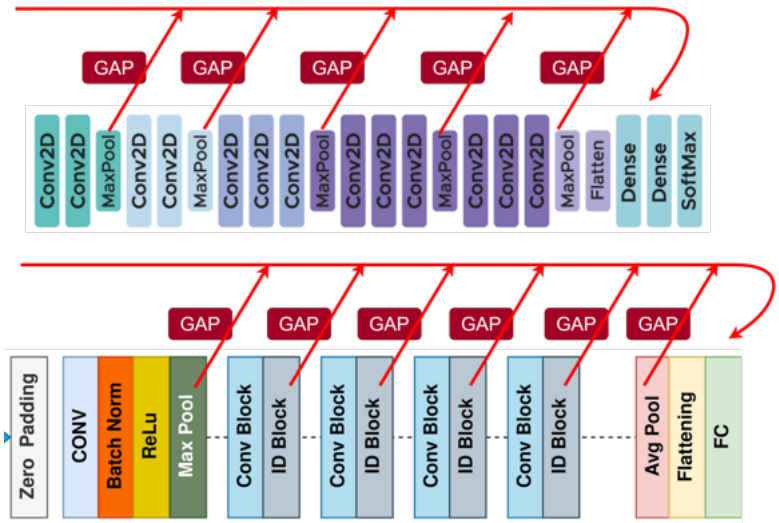


Figura 3.14: Diagrama interno de modificado de las redes VGG16 y ResNet50, añadiendo los diferentes caminos invariantes a través de capas GAP hasta el final de la red, donde se concatenan todos en la capa de clasificación.

En el ejemplo de la red VGG16GAP el esquema de la red queda como el que señala la Figura 3.15, con los caminos alternativos concatenando GAPs en diferentes estadios de la red -coincidiendo con las salidas de max pooling- y que son la entrada a la última capa.

Se espera un mejor rendimiento de los modelos frente a datos de entrada trasladados, es decir, que las nuevas redes adquieran cierta consistencia a la traslación mostrando la característica de invarianza.

3.4. Experimentación

En cuanto a las especificaciones técnicas, todos los experimentos se han realizado con un servidor utilizado por el grupo de investigación IFIC de la Universitat de València (Artemisa)[1] Se ha hecho el seguimiento de todos los experimentos con la herramienta de WanDB (Weights & Biases). Se puede visitar en el siguiente enlace:

https://wandb.ai/rietta/TFM_PruebasInv?workspace=user-rietta
https://wandb.ai/rietta/Sweep_VGG_ResNet_256/table?workspace=user-rietta

Con ella, se pueden estudiar las curvas de aprendizaje de diferentes modelos entrenados, guardar resultados y realizar *sweeps* -barridos de hiperparámetros-. Estas dos últimas, empleadas en las Secciones 2.3 y 2.4, son especialmente útiles para optimizar de manera correcta los

[1]Some computer 342 resources were provided by Artemisa, funded by the European Union ERDF and Comunitat Valenciana as 343 well as the technical support provided by the Instituto de Física Corpuscular, IFIC (CSIC-UV). El listado de sus componentes se puede ver en el siguiente enlace: Artemisa hardware composition.

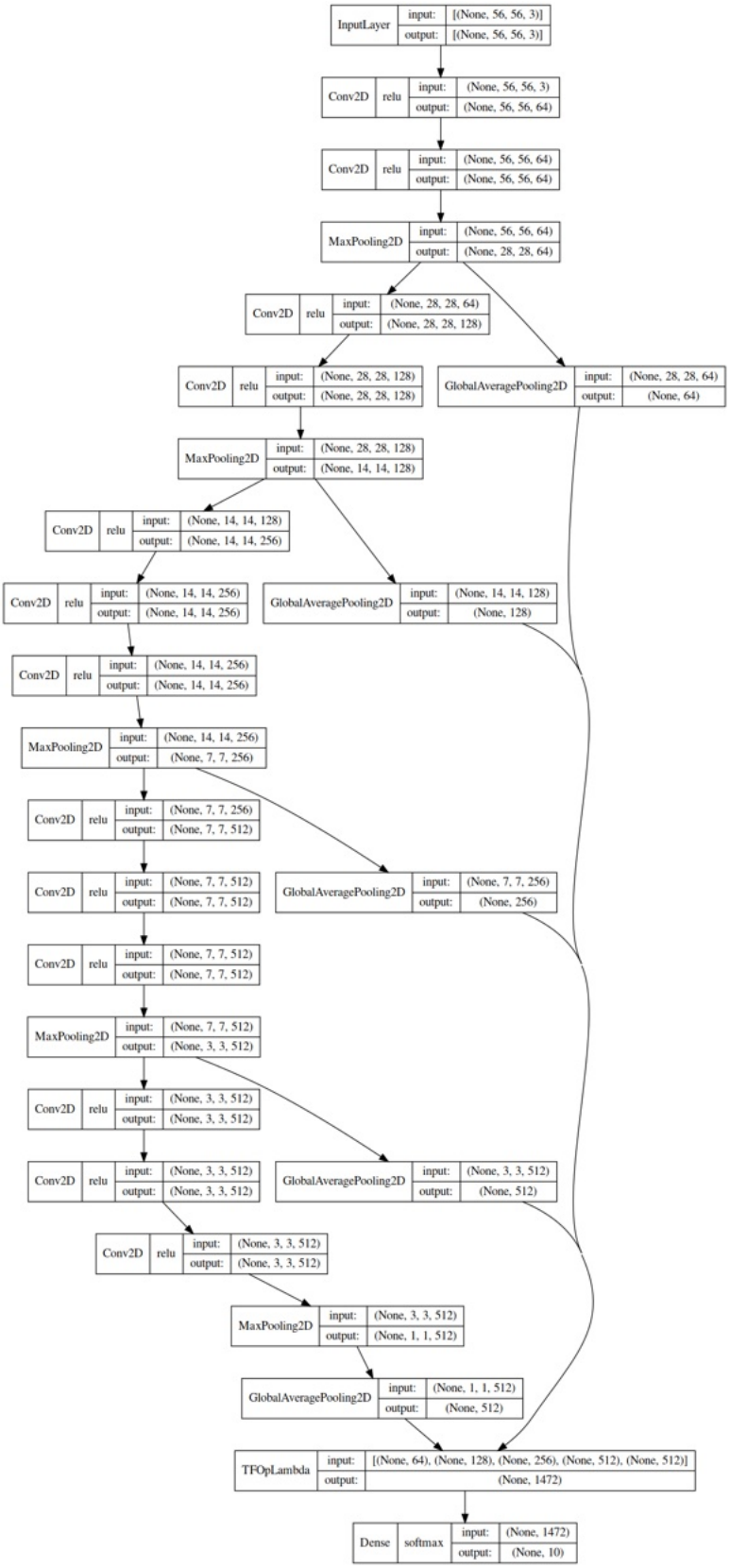


Figura 3.15: Arquitectura modificada de la red VGG16, añadiendo los diferentes caminos invariantes a través de capas GAP.

hiperparámetros de los modelos. En el ejemplo de la Figura 2.11, se hace un barrido de los hiperparámetros *tamaño del filtro* y la *cantidad de filtros* buscando los valores que devolvían mejor rendimiento para la red. Este procedimiento se utiliza habitualmente en el marco de trabajo

de la Inteligencia Artificial y *Deep Learning* y se utiliza también en los experimentos propuestos.

Para cada punto de la metodología se realizan un diferente número de experimentos motivados por diferentes aspectos del problema. Dentro de cada experimento, lo que se estudia es:

1. Problemas de clasificación: En estos problemas es importante conocer la precisión de los modelos. No obstante, como en este caso lo importante no es cómo de bien predice cada clase por separado, sino el conjunto general, no se harán diferenciaciones por etiqueta. Lo que se estudiará será la superficie de precisión para cada desplazamiento dentro de la imagen. En esta sección, el conjunto de datos es el de MNIST modificado 4.2 sin que los dígitos se salgan del fondo. En el apartado de modificaciones ligeras se hacen pruebas con desplazamientos de 10 píxeles en imágenes de 56x56. En el apartado de redes preentrenadas, se realizan pruebas con diferentes tamaños de imágenes y desplazamientos. En las pruebas para imágenes de 56x56 y 128x128 se desplaza el dígito hasta 10 píxeles y para las imágenes de 256x256 se aumenta el rango de desplazamiento hasta 50 píxeles.

2. Problemas de calidad: En la calidad de imagen se estudiará algo muy similar a la de clasificación. En lugar de profundizar en la precisión del modelo, se hará sobre la propia métrica, observando su comportamiento en los desplazamientos de la imagen. Esto se estudiará con un gráfico similar a los mapas de calor clásicos. Se ensalzan aquellas posiciones que devuelven valores extremos, ya sea por excesiva similitud o por completa diferenciación. En un principio se testeó con que todas las gráficas tuvieran el mismo rango de valores para facilitar las comparaciones entre diferentes métodos, pero entonces se pierde detalle para la propia gráfica. Se considera, entonces, que cada gráfica tenga su propia escala y se destaquen los patrones de forma natural. Siguen siendo comparables entre sí, pero hay que prestar especial atención a la escala de cada gráfica. En todos los experimentos de esta sección las imágenes empleadas son las de MNIST modificado 4.2 sin que los dígitos se salgan del fondo, tienen un tamaño de 56x56 y los desplazamientos son de 10 píxeles. Los estadísticos que se estudian en esta sección son la media y la desviación estándar.

Todos los experimentos se analizan en función de dos principales características deseables en estas problemáticas: La isotropía y anchura. En ambas aplicaciones, se espera un comportamiento que no diferencie entre direcciones, es decir, que no haya sesgo respecto a si el movimiento es hacia la derecha o a la izquierda, arriba o abajo. El otro punto es la anchura. Cuanto más robusto sea un modelo, mayor cantidad de desplazamiento soportará, retrasando la caída de *accuracy*. Por lo tanto, se espera que en todos los experimentos realizados se observe un patrón circular que, cuanto mejor sea su rendimiento, más ancho sea.

Capítulo 4

Conjuntos de datos

Como se ha comentado durante el Cápitulo 1, hay infinidad de aplicaciones posibles para las equi e invarianzas. Para reflejar el proceso descrito en el trabajo, explicado en el Capítulo 3, se han elegido diferentes conjuntos de datos para hacer las pruebas pertinentes sobre ellos y testar su funcionamiento. En primer lugar, el conjunto de datos MNIST. La elección de este conjunto está motivado por el conocimiento general que se tiene de los datos, es conocido y habitual en todas las aplicaciones iniciales de procesado de imágenes, por lo que es fácil de comparar. En segundo lugar, el conjunto de datos *Buscando a Wally*, un juego en formato libro muy en conocido y famoso, que cuenta con unas particularidades especiales. Se explicará detalladamente en la Sección 4.3.

4.1. MNIST

MNIST es un conjunto de datos que contiene imágenes de números manuscritos. Es muy habitual dentro del ámbito de los modelos de procesado de imágenes por ser el conjunto inicial con el que se suele trabajar a modo de bienvenida. Es el conjunto que se ha estado usando hasta el momento para ejemplificar los modelos teóricos vistos la Sección 2.4. A continuación, se ampliará la información sobre las imágenes.

Esta base de datos está formada por imágenes de tamaño 28x28, en blanco y negro -un solo canal- etiquetado con el número manuscrito que aparece en la imagen. Cuenta con 60.000 imágenes de entrenamiento y 10.000 imágenes de prueba. En la Figura 4.1 se muestran algunos ejemplos de imágenes que compone el conjunto.

Figura 4.1: Ejemplos de imágenes de MNIST. Imágenes de 28x28, en blanco y negro -un solo canal- etiquetado con el número manuscrito que aparece en la imagen.

4.2. Dataset propio: MNIST desplazado

Además de la versión clásica, se ha creado para el proyecto una versión del dataset trasladado, que se ajusta más a nuestra problemática. En esta versión las imágenes mantienen sus características de tamaño y canales, pero la posición del número manuscrito cambia. Se ha ido moviendo el número desde la posición inial, hasta un máximo de [+10,-10] píxeles en cada orientación: horizontal, vertical y en la diagonal cuando se unen ambos moviemientos -Ver Figura 4.2-. Se aclará cuándo se esté usando el MNIST clásico del nuevo conjunto MNIST desplazado.

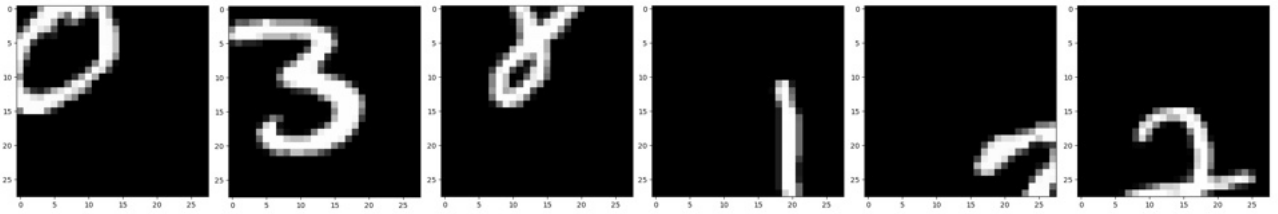


Figura 4.2: Ejemplos de imágenes de MNIST desplazado. Imágenes de 28x28, en blanco y negro -un solo canal- etiquetado con el número manuscrito que aparece en la imagen, pero en el que no aparece el dígito en su totalidad.

Por último, se ha creado un conjunto que cumple con la misma función de desplazamiento, pero esta vez asegurando que el número manuscrito no salga nunca del marco de la imagen. Para esto, se ha creado un canvas que duplica el tamaño de la imagen real -56x56- y sobre él se ha ido desplazando la imagen hasta un rango de [+10,-10]. De esta forma es imposible que el número se quede incompleto al salir del fondo. En la Figura 4.3 se pueden ver algunos ejemplos.

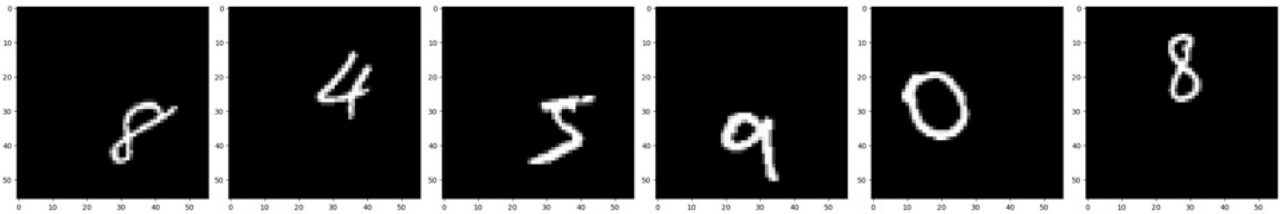


Figura 4.3: Ejemplos de imágenes de MNIST agrandado. Imágenes de 56x56, en blanco y negro -un solo canal- etiquetado con el número manuscrito que aparece en la imagen

Para el diferente entrenamiento de las redes, será interesante ver el funcionamiento con estas diferentes imágenes de entradas y cómo lidian con esos números incompletos. En cada entrenamiento se aclará qué dataset de los dos -el solo desplazao o el también agrandado- cuando se use.

4.3. Dataset propio: Buscando a Wally

Buscando a Wally es una serie de libros creada por el dibujante británico Martin Handford en 1987. No obstante, la finalidad de esos libros no es leer, sino jugar a encontrar a Wally, el personaje perdido que se esconde en las diferentes escenas que componen el libro. Estas escenas están repletas de personajes, objetos y detalles que pretenden despistar al jugador sobre el verdadero paradero de Wally. Para facilitar la tarea, Wally siempre viste igual: con una camiseta de rayas rojas y blancas, un sombrero a juego, unos vaqueros y un bastón. Este último objeto,

además, puede ser perdido por el propio Wally y debe ser encontrado también por el jugador, dificultando todavía más la tarea de búsqueda. Se puede ver el modelo de Wally y un ejemplo de las páginas del libro en la Figura 4.4.

Figura 4.4: Ejemplo del modelo de Wally y una página del libro.

Estos libros se hicieron tremendamente populares en la década de los 90 y son el ejemplo perfecto de la habilidad humana para la equivarianza traslacional. Como se ha comentado en la Sección 2.3, si nuestro cerebro no tuviera esta capacidad equivariante, no podríamos ejecutar tareas de búsqueda porque perderíamos la referencia espacial. Este es el motivo porque el que se ha creado un dataset sencillo para probar esta propiedad, realizada en la Sección 2.3.

Para la creación del dataset, lo que se ha hecho es coger la imagen de referencia del modelo de Wally con el fondo transparente y escalarlo a un tamaño de (80,156), crear un fondo blanco de dimensiones (500, 500) e ir moviendo a Wally a través del fondo. Empezando desde la esquina superior izquierda, con pasos de 5 píxeles, se ha ido desplanzando hasta llegar a la esquina inferior derecha. En total, se han creado 5880 imágenes en las que en cada una Wally está en una posición diferente del fondo. Además de guardarnos las imágenes generadas, se ha guardado también la posición en la que se ha colocado a Wally para usarlo de etiqueta en la red.

Una vez se ha generado todo el material, se ha normalizado tanto las imágenes como las etiquetas para que estén en el rango [0,1] y facilitar así el trabajo computacional de la red. En la Figura 4.5 se pueden visualizar algunos ejemplos de imágenes de este conjunto de datos.

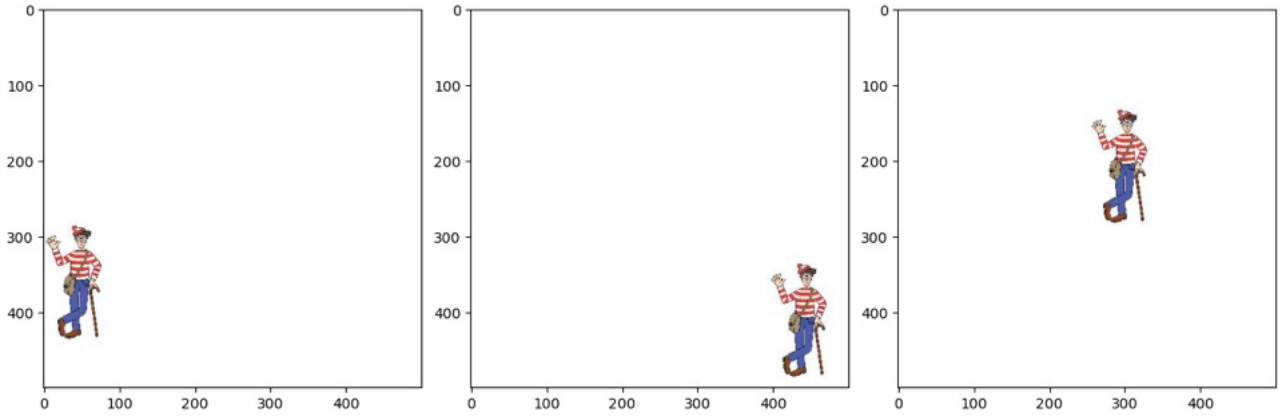


Figura 4.5: Ejemplo del dataset creado. Imágenes de 500x500 de un fondo blanco en el que se superponen la figura de Wally en diferentes localizaciones.

Para agilizar el trabajo de la red, se han reescalado las imágenes a 128x128. Por lo que Wally aparece un poco pixelado pero es suficiente para comprobar que cumple su cometido de detector de objetos. Los nuevos ejemplos con las imágenes reescaladas se pueden ver en la Figura 4.6.

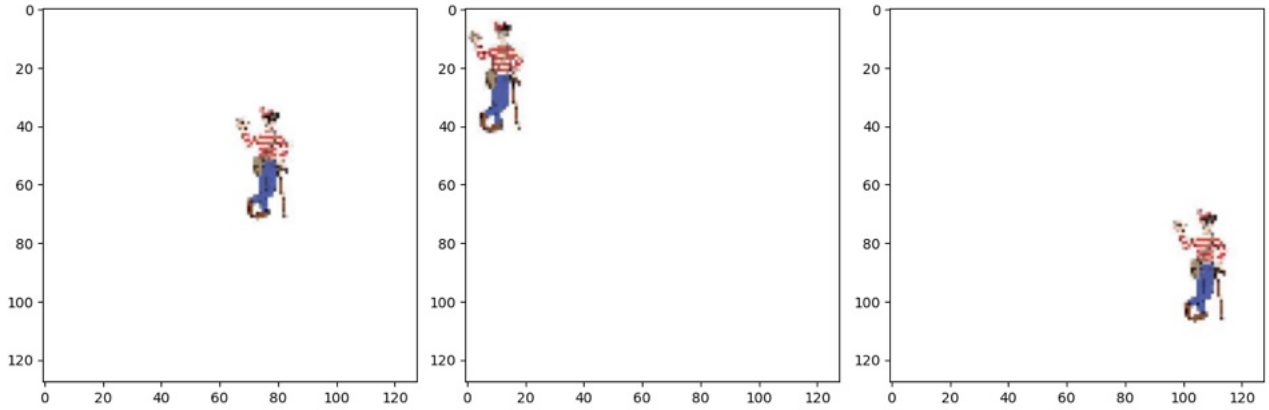


Figura 4.6: Ejemplo del dataset creado con las imágenes reescaladas a 128x128 para mejorar su procesado en redes de tamaño reducido.

Por lo tanto, y a modo de resumen, este conjunto de datos cuenta con 5880 -3939 imágenes para el conjunto de entrenamiento y 1941 para el conjunto de test- imágenes de 128x128 a color -es decir, con tres canales- y etiquetado con la posición en la que está Wally.

Capítulo 5

Resultados

En esta sección se presentan los resultados de todos los experimentos comentados en la Sección 3 .Todo el código empleado se puede encontrar en el GitHub:

https://github.com/Rietta5/TFM

5.1. Problemas de clasificación

Todos los experimentos de esta sección se han realizado con los datos de MNIST ampliado, Sección 4.2. El conjunto empleado ha sido el de train, con un total de 60.000 imágenes, desplazadas a través de todo el *canvas* y evaluando posteriormente en los diferentes modelos de clasificación.

5.1.1. Modificaciones ligeras

Teniendo en cuenta el modelo que se mostraba en la Sección 2.4, y lo comentado en la Subsección 3.1.1, la primera prueba será sobre el modelo básico de invarianzas a la que se le añade una capa de *pooling* antes de la capa GAP. La Figura 5.1 se muestra el impacto de los hiperparámetros en la precisión final del modelo.

Se observa que, a pesar de que se esperaba que el *pooling* impactara negativamente sobre la precisión general del modelo, no se observa como tal. Los modelos con la capa pooling -independientemente del tamaño del filtro- consiguen un rendimiento muy similar al que tenía el mejor modelo sin esta capa.

Como se explica en el Capítulo 3, se espera en la superficie de predicción un comportamiento como el de las olas del mar, que alcance su pico en posiciones divisibles entre el desplazamiento, se vaya reduciendo la precisión en otras posiciones no divisibles. En las siguientes figuras se pueden observar los resultados obtenidos con diferentes valores en el *max pooling*.

- *Max pooling* = 2: En la Figura 5.2, se pueden ver los resultados de un modelo que contiene un *max pooling* de tamaño dos. En la gráfica central se pueden distinguir claramente dos

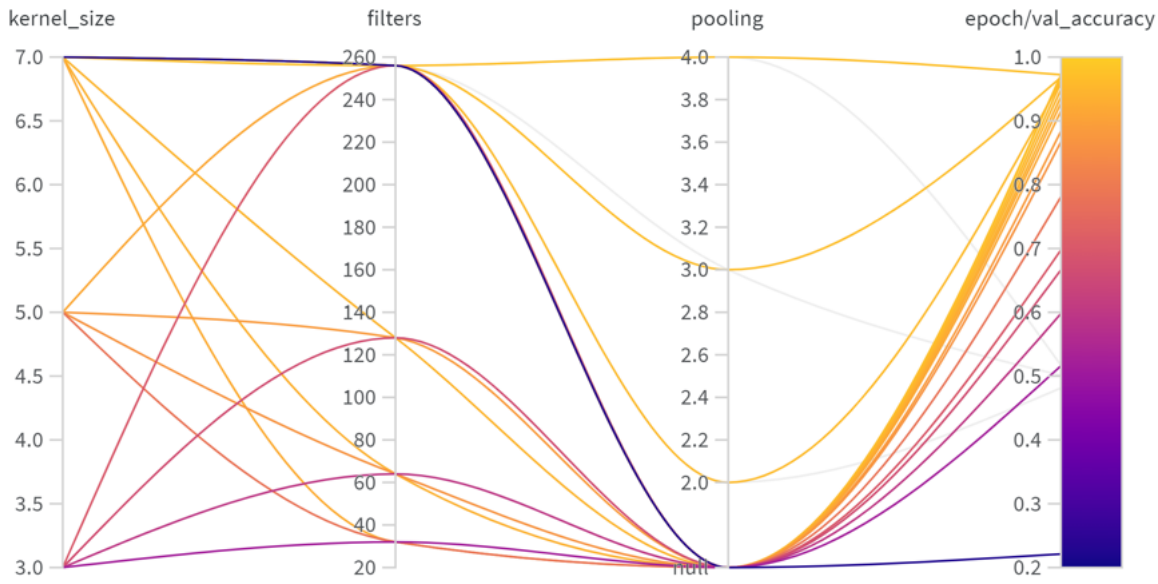


Figura 5.1: Optimización de hiperparámetros con la capa *pooling*. Modificando los siguientes parámetros para ver cómo impacta la configuración en el rendimiento de la red: Tamaño del filtro (*kernel_ size*) variando entre los tamaños tres, cinco y siete; la cantidad de filtros (*filters*) variando entre 32, 64, 128 y 256 y el tamaño del *pooling* variando entre los tamaños dos, tres y cuatro.

valores: uno alcanzando casi el 0.97 y otro más bajo en el 0.96. Demostrando que alcanza dos valores en función de si el desplazamiento es divisible o no.

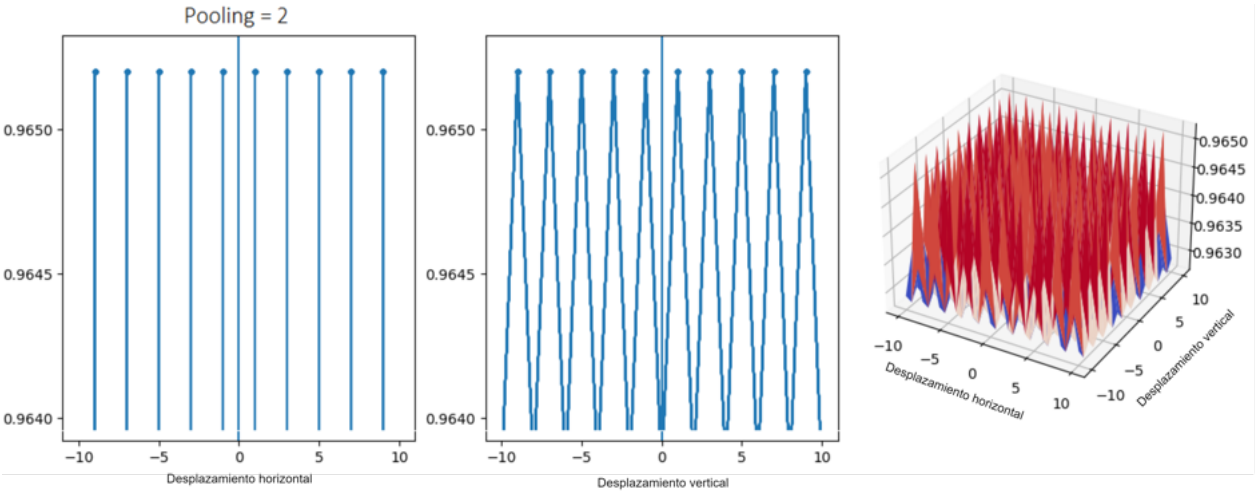


Figura 5.2: Superficie de precisión de una red con *max pooling* de tamaño dos. Alcanza dos valores en función de si el desplazamiento es divisible o no entre el tamaño del *pooling*.

- *Max pooling* = 3: Los resultados se puede visualizar en la Figura 5.3. En la gráfica de la izquierda se pueden distinguir dos valores diferentes, pero el más alto se alcanza en ciclos de tres -el tamaño del *pooling*-. Esto indica que para desplazamientos 1 y 2, se baja en precisión pero no hay diferencia entre ambos desplazamientos. Notar también que la diferencia de rendimiento para las diferentes traslaciones no es especialmente relevante.
- *Max pooling* = 4: Los resultados obtenidos se muestran en la Figura 5.4. Las gráficas

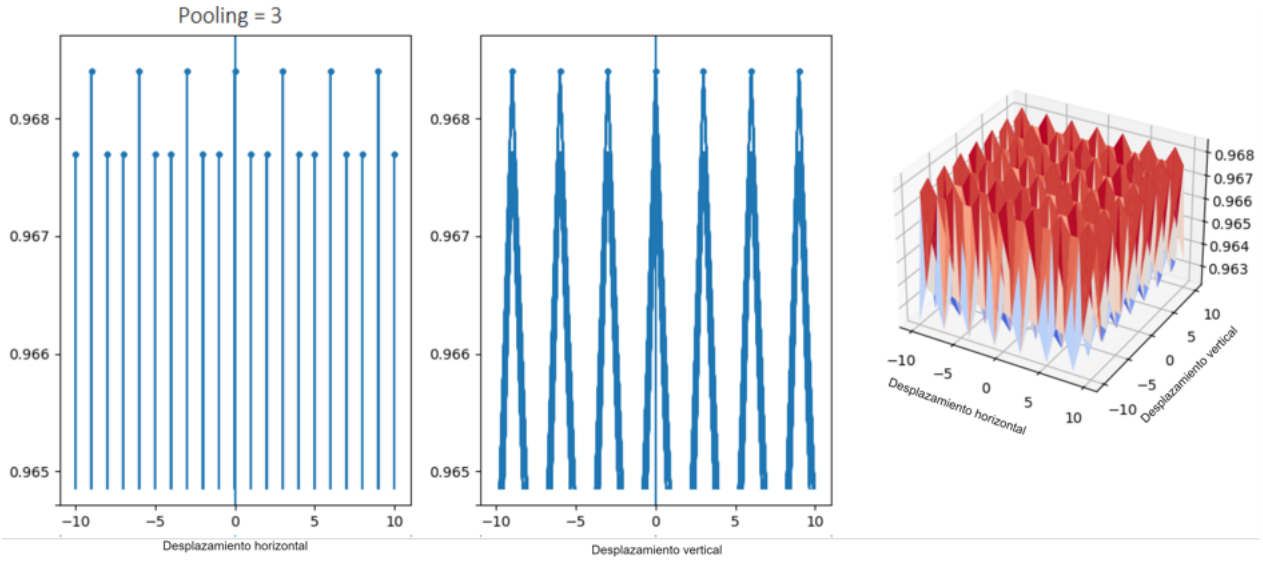


Figura 5.3: Superficie de precisión de la red con *max pooling* de tamaño tres. Alcanza tres valores en función de si el desplazamiento es divisible o no entre el tamaño del *pooling*.

muestran lo comentado en apartados anteriores: el máximo aparece en ciclos del tamaño del *pooling*. Además en este caso la diferencia de precisiones, si bien es mayor que en otros casos, tampoco es excesiva la diferencia.

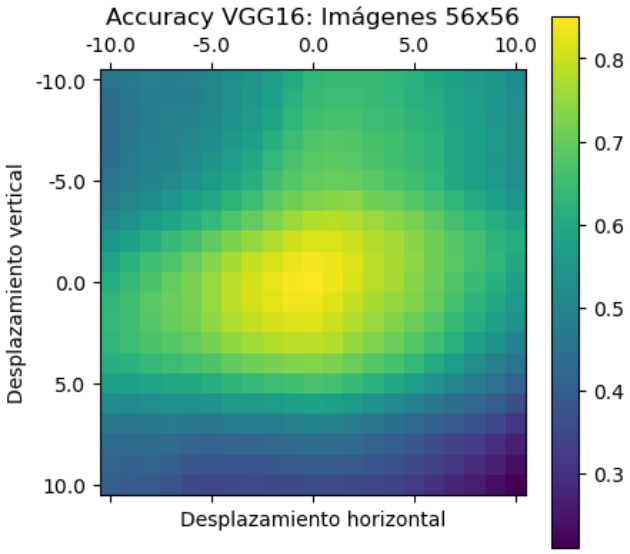
Figura 5.4: Superficie de precisión de la red con *max pooling* de tamaño cuatro. Alcanza cuatro valores en función de si el desplazamiento es divisible o no entre el tamaño del *pooling*.

5.1.2. Arquitecturas preentrenadas

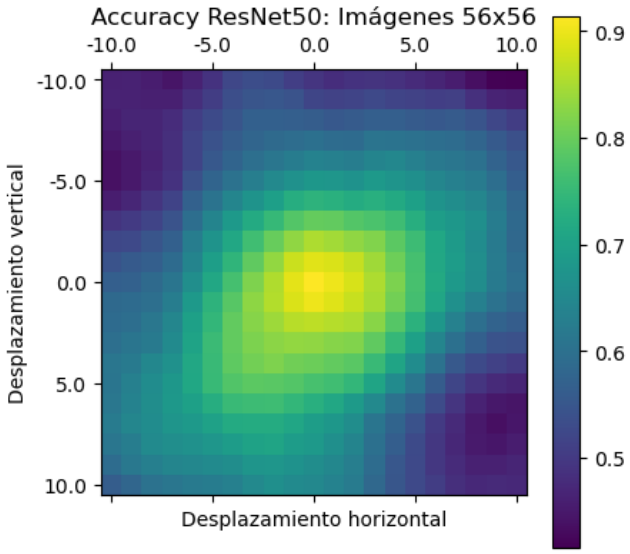
En la Figura 5.5 se pueden observar los resultados obtenidos con las redes preentrenadas. Las de la izquierda corresponden a la red VGG16 y las de la derecha a ResNet50, entrenada con los tamaños que se indican.

Como cabría esperar, los resultados de la ResNet50 son más radiales y robustos que para la red VGG. Esto se debe a que la VGG no cuenta con la capa GAP al final de su arquitectura, devolviendo una superficie de *accuracy* muy poco plana que, dependiendo del tamaño de la imagen de entrada denota preferencia por ciertos desplazamientos. Por ejemplo, para imágenes de 56x56, los desplazamientos diagonales hacia la derecha son más sensibles que en cualquier otra dirección, mientras que para imágenes de 128x128 son los desplazamientos diagonales hacia la izquierda los más sensibles. Por último, para imágenes de 256x256, el comportamiento es muy similar al del resto de tamaño, pero al haber mayor número de desplazamientos, la zona de de mayor *accuracy* es más pequeña. Se sigue percibiendo una preferencia por los desplazamientos horizontales y verticales.

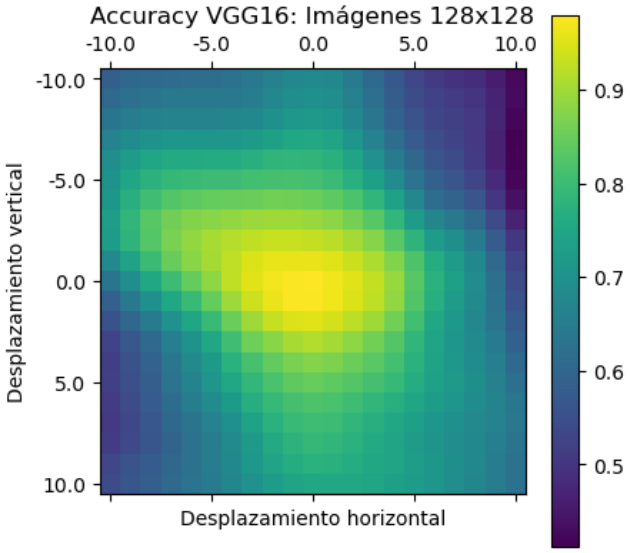
Por su parte, ResNet50 que sí cuenta con una GAP al final de su arquitectura, muestra mejores resultados. Para imágenes de 56x56 el comportamiento es bastante isótropo con una clara preferencia por los desplazamientos horizontal y vertical pero no horizontal. Sin embargo,



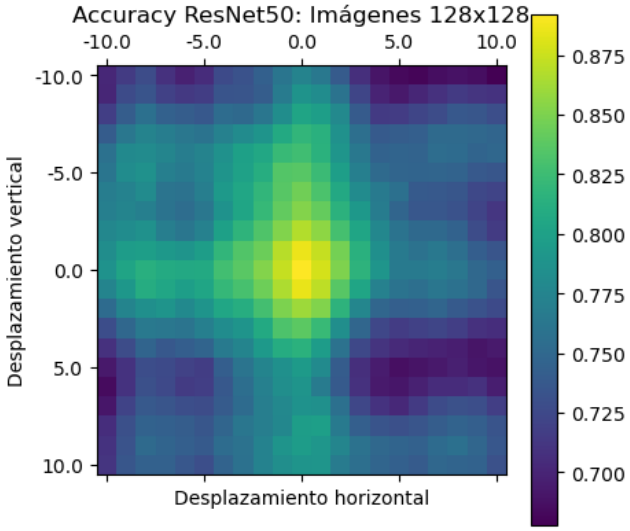
(a) VGG16: 56x56



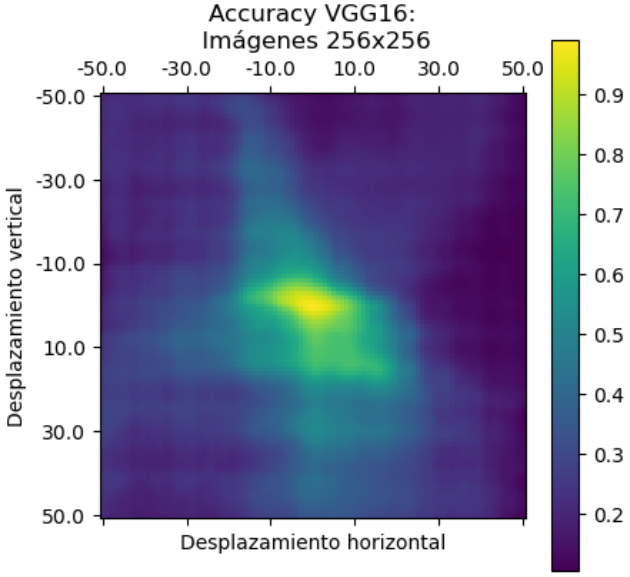
(b) ResNet50: 56x56



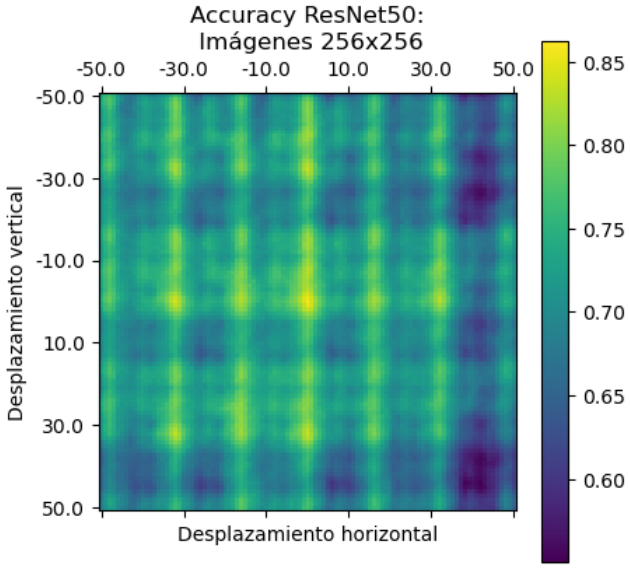
(c) VGG16: 128x128



(d) ResNet50: 128x128



(e) VGG16: 256x256



(f) ResNet50: 256x256

Figura 5.5: Resultados obtenidos con las redes preentrenadas. Las de la izquierda corresponden a la red VGG16 y las de la derecha a ResNet50, entrenada con los tamaños que se indican.

el comportamiento para imágenes de 128x128, aunque no muestra un patrón circular, es claramente una superficie más plana, devolviendo valores más altos en todos los puntos del fondo, aunque mantiene la preferencia por los movimientos no diagonales. En cuanto a las imágenes de 256x256, se puede observar que vuelven los patrones con forma de olas que se formaban en los experimentos de la Subsección 5.1.1

Por lo tanto, en general, parece que la red con mayor robustez hacia la traslación sea la de ResNet50, inducido por su última capa.

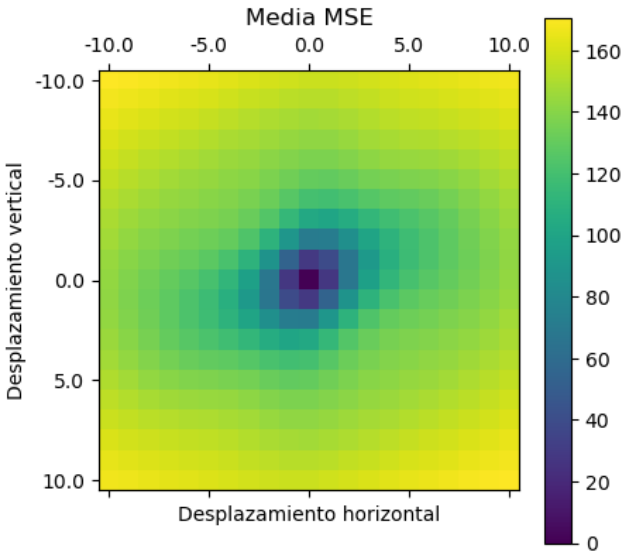
5.2. Problemas de calidad de imagen

Como se explica en la Sección 3.4, se va a profundizar en dos estadísticos: La media y la desviación estándar. Los resultados correspondientes a las medias se pueden visualizar en la Figura 5.6, mientras que la desviación estándar se puede visualizar en la Figura 5.7.

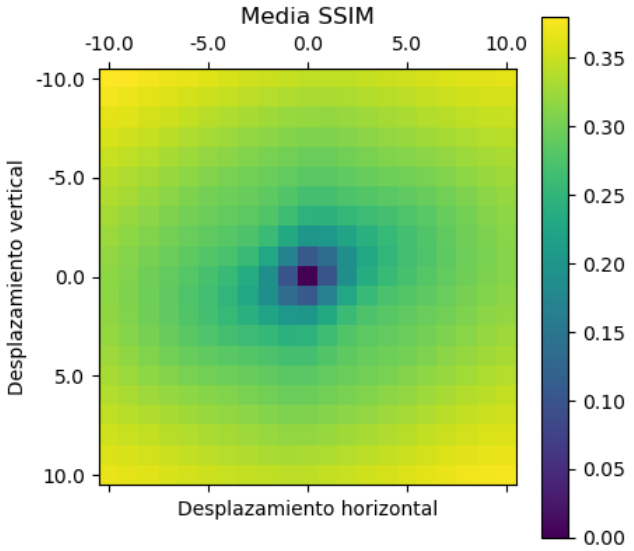
El MSE es una métrica que comprende el intervalo $[0, \infty)$. La superficie mostrada en la media es la que esperaríamos para todos los modelos, cuanto más al centro del *canvas* se sitúe el objeto, menos distancia hay con la imagen inicial. Conforme el número se va desplazando, aumenta la distancia entre la original y la modificada. Es decir, un patrón circular, aunque ligeramente achatado y por ende no es totalmente isótropo. Si bien es cierto que el patrón es el esperado, lo que no lo es tanto son los valores resultantes. En cuanto se la imagen sale del centro de la figura, el MSE toma un valor muy alto, que además es muy poco consistente -se muestra también una desviación estándar muy alta-. Esto concuerda con lo que se comentaba en la Sección 3.2, cuando se comentaba los fallos frente a traslaciones.

En cuanto al SSIM, el rango de valores de esta métrica va del 0 al 1. Donde el 1 nos indica que la imagen de referencia y la modificada son perceptualmente idénticas y el 0 que son radicalmente opuestas. Como era de esperar, los valores en el punto (0,0) nos devuelven los mejores valores. Un 1 en media y 0 en desviación estándar. Una vez se sale del punto central, los valores cambian de forma muy poco gradual. La media tiene una forma similar a un patrón radial y más isótropa que la del MSE. Si bien es el comportamiento esperado, tiene un cambio de valor muy fuerte, devolviendo peores valores en las esquinas. Vemos una conducta muy peculiar en la desviación estándar. En lugar de encontrar un patrón circular, se encuentra un patrón con forma de reloj de arena, pero que ni siquiera está centrado sobre el eje de abscisas. Es un patrón extraño que parece favorecer las imágenes desplazadas en esa diagonal y para las que la desviación estándar muestra un mayor valor. Comentar también que se diferencia un registro de mejores valores sobre el eje de coordenadas.

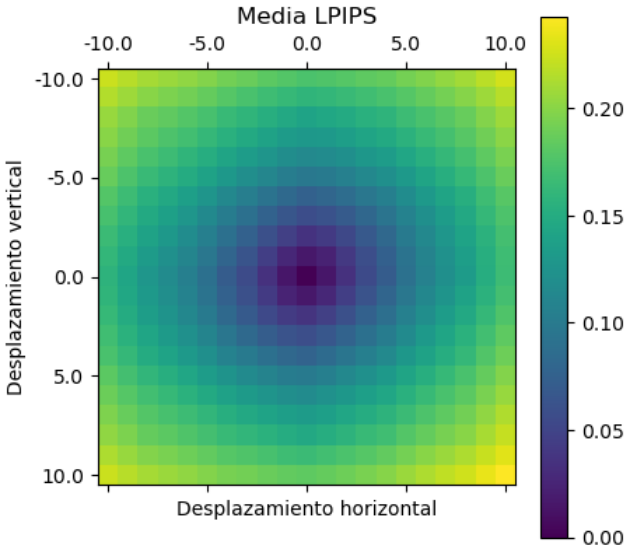
Por su parte, la métrica LPIPS comparte rango de valores con el MSE $[0, \infty]$, pero no por ello son comparables. Un valor de 0.2 en LPIPS no implica un valor 0.2 en MSE. Se puede notar un patrón radial en el valor de la media. De hecho, en el estadístico media, notamos que es la métrica con mayor anchura de todas. No obstante, este comportamiento no es el mismo en la desviación estándar. En este último, el patrón tiene una forma de bandas verticales que, además, no es simétrico. Es un comportamiento poco esperado, pues también se esperaba un patrón isótropo, similar al de la media. Todo esto parece remarcar que LPIPS tiene cierta robustez en los desplazamientos verticales, pero no así a los horizontales.



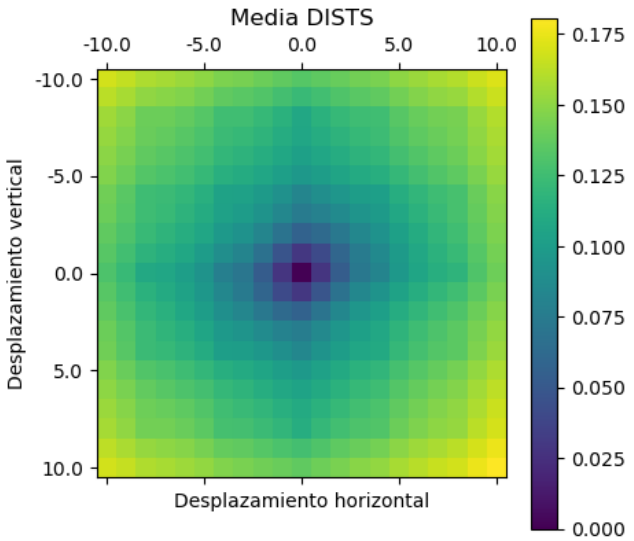
(a) MSE



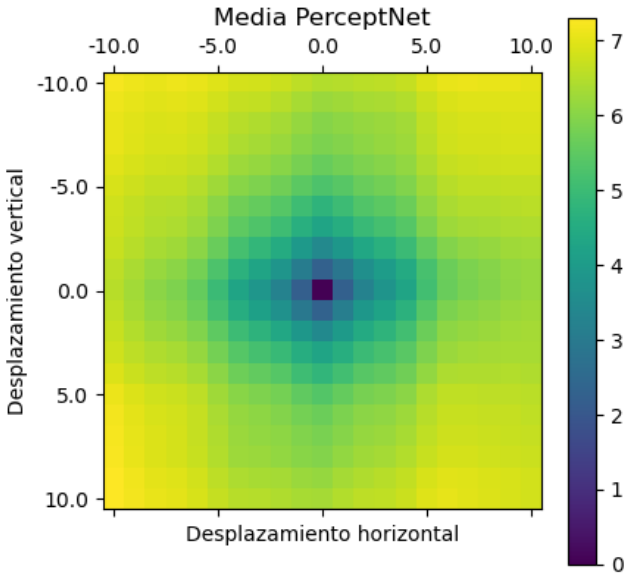
(b) SSIM



(c) LPIPS

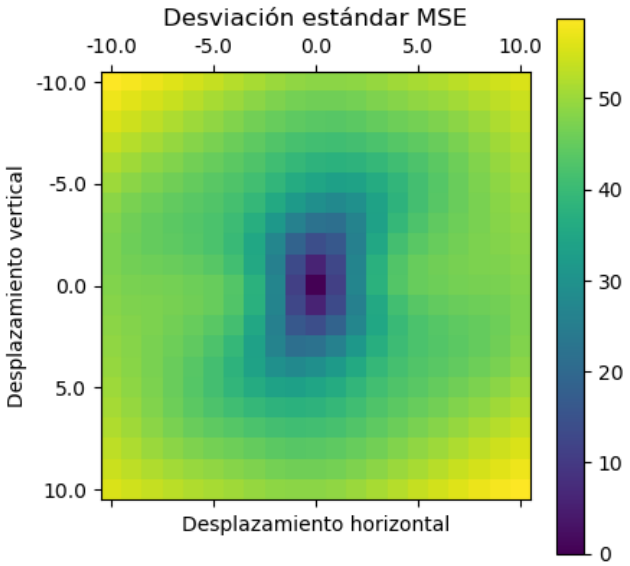


(d) DISTS

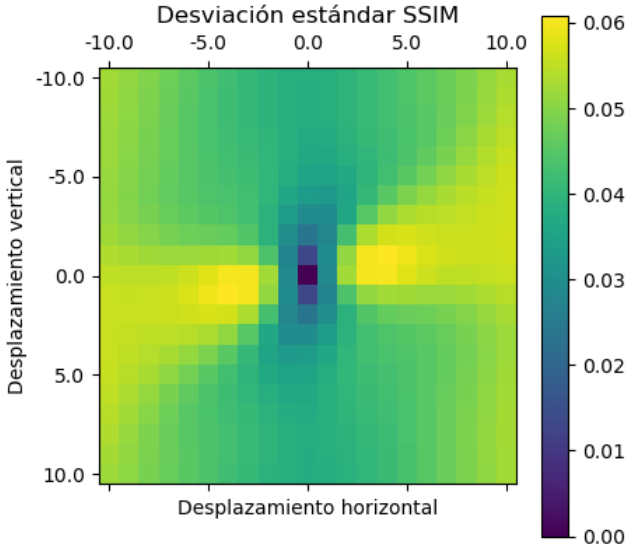


(e) PerceptNet

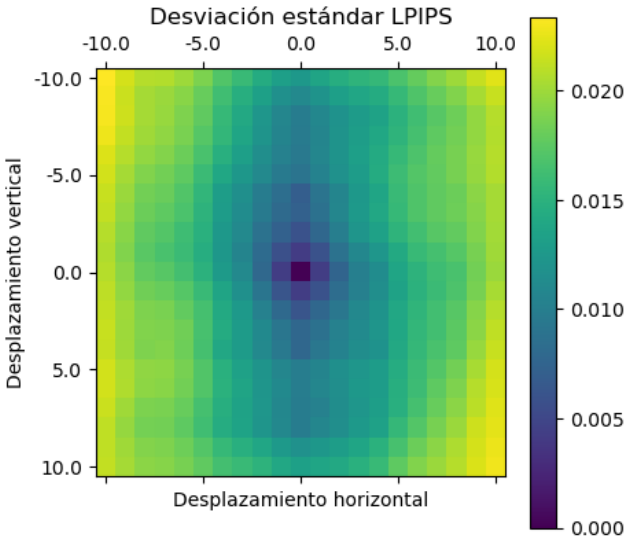
Figura 5.6: Superficie de la percepción media para cada desplazamiento en las diferentes métricas seleccionadas. En general se aprecia un comportamiento isótropo en la mayoría de ellas. Aunque los desplazamientos diagonales afectan más que los horizontales o verticales.



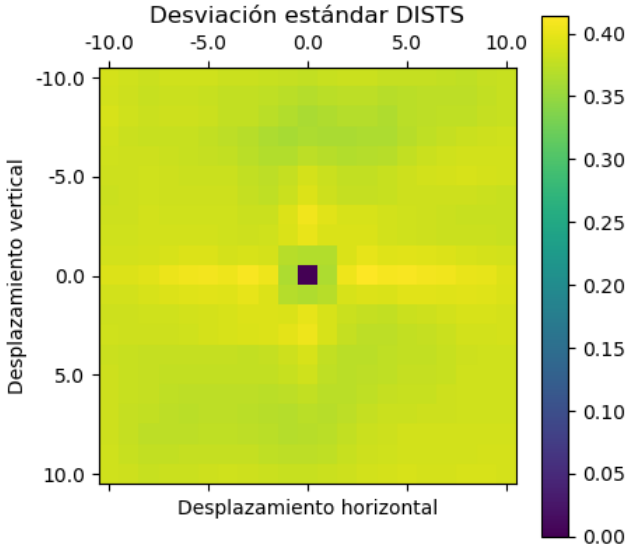
(a) MSE



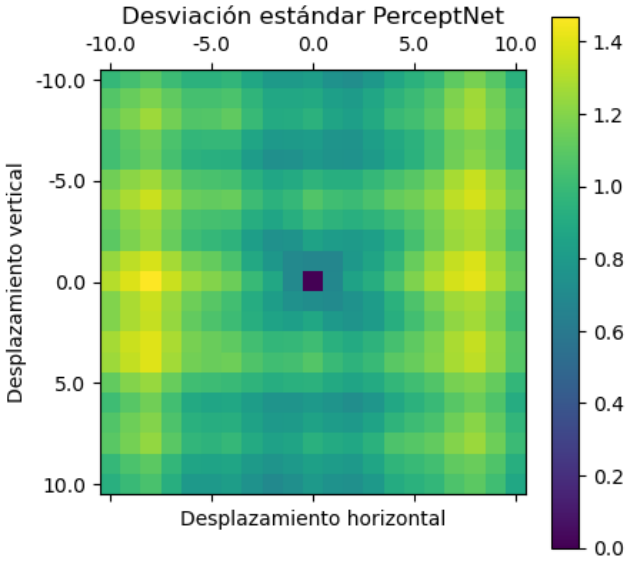
(b) SSIM



(c) LPIPS



(d) DISTS



(e) PerceptNet

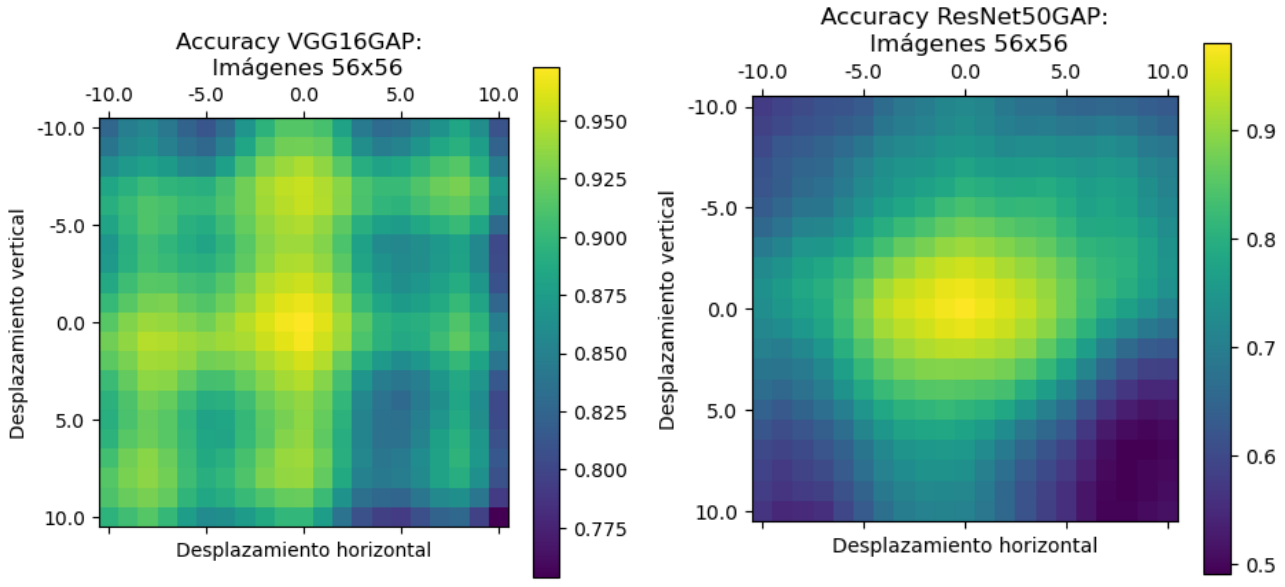
Figura 5.7: Superficie de la desviación estándar de la percepción para cada desplazamiento en las diferentes métricas seleccionadas. Aunque se esperaba también un comportamiento similar al de la media, cada una refleja una sensibilidad diferente a distintos desplazamientos.

En DISTS, el rango de valores es el mismo que para MSE y SSIM, aunque siguen sin ser comparables. Se repite de nuevo el patrón del resto de gráficas, donde los valores centrales muestran los mejores resultados. En comparación a las gráficas obtenidas con LPIPS, estas gráficas son algo peores porque reflejan zonas más planas en general junto con cambios muy poco graduales. Se intuye una mayor sensibilidad hacia desplazamientos diagonales, que no ocurre con LPIPS, que es muy isótropo. Desde el punto de vista de la desviación estándar, llama la atención que en la zona de menor media es la zona con mayor desviación estándar. Se repite el patrón de la media, pero a la inversa.

Por último, en PerceptNet se vuelve a tener un patrón relativamente isótropo, porque la dirección diagonal le afecta más las direcciones horizontal y vertical. Teniendo en cuenta las diferencias de escala, es muy similar a la gráfica de DISTS. El patrón verdaderamente peculiar se encuentra en la desviación estándar, donde se pueden ver unas bandas verticales y horizontales, creando una rejilla que no es el comportamiento esperado en absoluto. Parece que los movimientos verticales más alejados del centro devuelven resultados poco robustos.

5.3. Propuesta de mejora

Para finalizar el apartado de resultados, se muestran a continuación lo obtenido en los experimentos relacionados con la propuesta de mejora -Sección 3.3-, en la Figura 5.8 se pueden ver los resultados.



(a) VGG16GAP (b) ResNet50GAP

Figura 5.8: Superficie de*accuracy* de las redes modifcadas VGG16GAP y ResNet50GAP para cada desplazamiento. La superficie para ambos modelos es mucho más plana que con los anteriores expremientos devolviendo mejores resultados para todos los desplazamientos.

Se puede observar en las gráficas que ambos modelos mejoran su rendimiento con la incorporación del camino alternativo de las GAP para imágenes del mismo tamaño y número

de desplazamientos en la Subsección 5.1.2. VGG16GAP tiene un rango de valores más pequeño indicando una mayor robustez, y, a pesar de que aparecen unas bandas verticales, se ve que la *accuracy* máxima aparece en muchas partes del mapa. De hecho, el patrón de máximo rendimiento se repite, cada 5 píxeles de desplazamiento, coincidiendo con el número de capas GAP que hemos introducido en la red y repitiendo el comportamiento de olas de la Sección 5.1.1.

Por su parte, ResNet50GAP, devuelve una superficie muy similar a la que se obtenía en la Sección 5.1.2, salvo que más ancha y el mínimo del rendimiento es mayor, por lo que se puede decir que su rendimiento también ha mejorado respecto a su modelo sin modificación. Se aprecia que las direcciones en las que es sensible también son las mismas.

Capítulo 6

Conclusiones y trabajo futuro

Después de lo visto a lo largo del TFM, se puede concluir que se han diseñado e implementado con éxito modelos que incluyen tanto equivarianzas como invarianzas. Se ha propuesto incluir caminos invariantes que mejoran el rendimiento de los modelos de clasificación preentrenados, demostrando su efectividad a través de diferentes desplazamiento. Por todo esto, se consideran cumplidos los objetivos planteados al inicio del trabajo.

Se ha explorado una modificación a los modelos de clasificación bases a partir de los resultados obtenidos. Expandiendo la arquitectura de los modelos preentrenados, se ha planteado la adición de nuevas capas GAP que generen más robustez en dichas redes, que mejoran considerablemente los resultados obtenidos para los experimentos hom. Se podría explorar más profundamente en futuros trabajos. Por otro lado, se ha estudiado profundamente la capacidad de los modelos de calidad de imagen de ser invariantes a la traslación, comprobando que no todos se comportan de la misma manera.

En cuanto a posibles ampliaciones futuras se podría, teniendo en cuenta la falta de isotropía de algunos resultados, hacer un estudio de simetría de los datos. De esa forma, se podría discernir si la preferencia de los modelos por ciertas direcciones viene dada por la morfología de los propios datos de entrenamiento. Por ejemplo, el dígito *8* es simétrico en ambas direcciones, pero no así el *5* que tiene mucha más carga de información en la parte derecha. Además, las redes mejoradas solo se han aplicado sobre imágenes de 56x56, sería interesante comprobar su funcionamiento para datos de entrada más grandes. Como última propuesta, se podría averiguar el rendimiento de observadores humanos frente a la problemática traslacional. Pese a que se esperaba un comportamiento isótropo ancho, tal vez el desempeño de los humanos no sea realmente así.

Bibliografía

[1] I. E. J. Malo and E. P. Simoncelli, "Nonlinear image representation for efficient perceptual coding," *IEEE Trans Image Processing*, 2006.

[2] V. L. J. Ballé and E. Simoncelli, "End-to-end optimized image compression," 2017.

[3] J. M. V. Laparra and J. Malo, "Divisive norm. image quality metric revisited," *Journal of Optical Society of America A*, 2010.

[4] A. B. V Laparra, J Ballé and E. Simoncelli, "Perceptual image quality assessment using a normalized laplacian pyramid," *Journal of Optical Society of America A*, 2016.

[5] J. M. R. M. A. Hepburn, V. Laparra and R. Santos-Rodriguez, "Perceptnet: A human visual system inspired neural network for estimating perceptual distance," *IEEE ICIP*, 2020.

[6] M. J. W. J Portilla, V Strela and E. Simoncelli, "Image denoising using scale mixtures of gaussians in thewavelet domain," *IEEE Trans Image Processing*, 2003.

[7] G. C.-V. V. Laparra, J. Gutiérrez and J. Malo, "Image denoising with kernels based on natural image relations," *The Journal of Machine Learning Research*, 2010.

[8] J. Portilla and E. Simoncelli, "A parametric texture model based on joint statistics of complex wavelet coefficients," *J. Comp. Vis.*, 2000.

[9] J. B. V. Laparra, A. Berardino and E. Simoncelli, "Perceptually optimized image rendering," *Journal of the Optical Society of America A*, 2017.

[10] E.-T. J. al, "Rotation-invariant optical recognition of three-dimensional objects," *Appl Opt.*, 2000.

[11] G. C.-V. V. Laparra and J. Malo, "Iterative gaussianization: From ica to random rotations," *IEEE Transactions on Neural Networks*, 2012.

[12] E. Laparra V. Pérez-Suay A. Mahecha M. Johnson and G. Camps-Valls, "Kernel methods and their derivatives: Concept and perspectives for the earth system sciences," *PLOS ONE*, 2020.

[13] J. Malo and J. Gutiérrez, "V1 non-linear properties emerge from local-to-global non-linear ica," *Network: Computation in Neural Systems*, 2006.

[14] G. C.-V. V. Laparra, S. Jiménez and J. Malo., "Nonlinearities and adaptation of color vision from sequential principal curves analysis," *Neural Computation*, 2012.

[15] D. T.-G. C.-V. V. Laparra, S. Jimenez and J. Malo., "Principal polynomial analysis," *Int. J. Neural syst,*, 2014.

[16] J. M. V. Laparra and G. Camps-Valls, "Dimensionality reduction via regression in hyperspectral imaging," *Sel. Topics Signal Processing*, 2015.

[17] G. C.-V. R. S.-R. E. Johnson, V. Laparra and J. Malo, "Information theory in density destructors," 2020.

[18] G. C.-V. R. S.-R. V. Laparra, E. Johnson and J. Malo, "Information theory measures via multidimensional gaussianization," 2020.

[19] V. N. R. L.-R. R. A. Tauste, M. Martinez-Garcia and G. Deco, "Task-driven intra-and interarea communications in primate cerebral cortex," *PNAS*, 2015.

[20] M. P. E. Johnson, V. Laparra and G. Camps-Valls, "Gaussianizing the earth: Multidimensional information measures for earth data analysis," 2020.

[21] L. G.-C. G. Camps-Valls, D. Tuia and J. Malo, "Remote sensing image processing," *Morgan & Claypool*, 2011.

[22] M. B. A. Gomez-Villa and J. Malo, "Visual information flow in wilson-cowan networks," *J. Neurophysiol*, 2020.

[23] J. Malo, "Spatio-chromatic information available from different neural layers via gaussianization," *J. Math. Neurosci.*, 2020.

[24] L. Perez and J. Wang, "The effectiveness of data augmentation in image classification using deep learning," 2017.

[25] A. P. Tal Feldman, "End-to-end bias mitigation: Removing gender bias in deep learning," 6 2021.

[26] L. C. B.-S. N. H. X. Y. S. K. P. A. F. F. L.-J. Peter A Noseworthy, Zachi I Attia, "Assessing and mitigating bias in medical artificial intelligence: The effects of race and ethnicity on a deep learning model for ecg analysis," 2 2016.

[27] S. M. Kunihiko Fukushima, "Neocognitron: A self-organizing neural network model for a mechanism of visual pattern recognition," *Biological Cybernetics*, 1979.

[28] Y. LeCun, B. Boser, J. Denker, D. Henderson, R. Howard, W. Hubbard, and L. Jackel, "Handwritten digit recognition with a back-propagation network," in *Advances in Neural Information Processing Systems* (D. Touretzky, ed.), vol. 2, Morgan-Kaufmann, 1989.

[29] Y. LeCun, Y. Bengio, *et al.*, "Convolutional networks for images, speech, and time series," *The handbook of brain theory and neural networks*, vol. 3361, no. 10, p. 1995, 1995.

[30] R. Gens and P. M. Domingos, "Deep symmetry networks," in *Advances in Neural Information Processing Systems* (Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Weinberger, eds.), vol. 27, Curran Associates, Inc., 2014.

[31] D. Marcos, M. Volpi, and D. Tuia, "Learning rotation invariant convolutional filters for texture classification," *CoRR*, vol. abs/1604.06720, 2016.

[32] K. Fukushima, "Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position," *Biological Cybernetics*.

[33] Y. Gong, L. Wang, R. Guo, and S. Lazebnik, "Multi-scale orderless pooling of deep convolutional activation features," *CoRR*, vol. abs/1403.1840, 2014.

[34] E. Kauderer-Abrams, "Quantifying translation-invariance in convolutional neural networks," *CoRR*, vol. abs/1801.01450, 2018.

[35] A. Azulay and Y. Weiss, "Why do deep convolutional networks generalize so poorly to small image transformations?," *CoRR*, vol. abs/1805.12177, 2018.

[36] C. J. H. L. Jeffrey S. Bowers, Ivan I. Vankov, "The visual system supports online translation invariance for object identification," *Psychonomic Bulletin Review*, 2016.

[37] H. Furukawa, "Deep learning for target classification from SAR imagery: Data augmentation and translation invariance," *CoRR*, vol. abs/1708.07920, 2017.

[38] I. Biederman and E. E. Cooper, "Evidence for complete translational and reflectional invariance in visual object priming," *SAGE*, 1991.

[39] J. E. H. Eric E Cooper, Irving Biederman, "Metric invariance in object recognition: a review and further evidence," 1992.

[40] I. I. V. C. J. H. L. J. S. B. Ryan Blything, Valerio Biscione, "The human visual system and cnns can both support robust online translation tolerance following extreme displacements," *Journal of Vision*, 1992.

[41] O. S. Kayhan and J. C. van Gemert, "On translation invariance in cnns: Convolutional layers can exploit absolute spatial location," *CoRR*, vol. abs/2003.07064, 2020.

[42] V. Biscione and J. S. Bowers, "Convolutional neural networks are not invariant to translation, but they can learn to be," *CoRR*, vol. abs/2110.05861, 2021.

[43] V. Biscione and J. Bowers, "Learning translation invariance in cnns," 2020.

[44] S. K. Hoda Sadeghzadeh, "Translation-invariant optical neural network for image classification," *Scientific Reports*, 2022.

[45] R. Hahnloser and H. S. Seung, "Permitted and forbidden sets in symmetric threshold-linear networks," in *Advances in Neural Information Processing Systems* (T. Leen, T. Dietterich, and V. Tresp, eds.), vol. 13, MIT Press, 2000.

[46] R. H. R. Hahnloser, R. Sarpeshkar, M. A. Mahowald, R. J. Douglas, and H. S. Seung, "Digital selection and analogue amplification coexist in a cortex-inspired silicon circuit," *Nature*, vol. 405, no. 6789, pp. 947–951, 2000.

[47] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[48] M. Ranzato, Y.-L. Boureau, and Y. LeCun, "Sparse feature learning for deep belief networks," *Advances in Neural Information Processing Systems*, vol. 20, pp. 1185–1192, 01 2008.

[49] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *CoRR*, vol. abs/1502.03167, 2015.

[50] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel, "Backpropagation applied to handwritten zip code recognition," *Neural Computation*, vol. 1, no. 4, pp. 541–551, 1989.

[51] M. Lin, Q. Chen, and S. Yan, "Network in network," 2014.

[52] R. An, G.-m. Chang, Y.-y. Fan, L.-l. Ji, X.-h. Wang, and S. Hong, "Machine learning-based patient classification system for adult patients in intensive care units: A cross-sectional study," *Journal of Nursing Management*, vol. 29, no. 6, pp. 1752–1762, 2021.

[53] H. Lu, H. Zhang, and A. Nayak, "A deep neural network for audio classification with a classifier attention mechanism," 2020.

[54] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2015.

[55] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," 2015.

[56] M. Banham and A. Katsaggelos, "Digital image restoration," *IEEE Signal Processing Magazine*, vol. 14, no. 2, pp. 24–41, 1997.

[57] Z. Wang, J. Chen, and S. C. H. Hoi, "Deep learning for image super-resolution: A survey," 2020.

[58] J. Yan, S. Lin, S. Bing Kang, and X. Tang, "A learning-to-rank approach for image color enhancement," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2014.

[59] Q. Yan, Y. Xu, and X. Yang, "No-reference image blur assessment based on gradient profile sharpness," in *2013 IEEE International Symposium on Broadband Multimedia Systems and Broadcasting (BMSB)*, pp. 1–4, 2013.

[60] K. Ma, W. Liu, T. Liu, Z. Wang, and D. Tao, "dipIQ: Blind image quality assessment by learning-to-rank discriminable image pairs," *IEEE Transactions on Image Processing*, vol. 26, pp. 3951–3964, aug 2017.

[61] A. Hepburn, V. Laparra, J. Malo, R. McConville, and R. Santos-Rodriguez, "Perceptnet: A human visual system inspired neural network for estimating perceptual distance," in *2020 IEEE International Conference on Image Processing (ICIP)*, IEEE, oct 2020.

[62] Z. Wang, A. Bovik, H. Sheikh, and E. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE Transactions on Image Processing*, vol. 13, no. 4, pp. 600–612, 2004.

[63] N. Ponomarenko, L. Jin, O. Ieremeiev, V. Lukin, K. Egiazarian, J. Astola, B. Vozel, K. Chehdi, M. Carli, F. Battisti, and C.-C. Jay Kuo, "Image database tid2013: Peculiarities, results and perspectives," *Signal Processing: Image Communication*, vol. 30, pp. 57–77, 2015.

[64] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, 2006.

[65] B. Girod, "Psychovisual aspects of image processing: What's wrong with mean squared error?," in *Proceedings of the Seventh Workshop on Multidimensional Signal Processing*, pp. P.2–P.2, 1991.

[66] P. Teo and D. Heeger, "Perceptual image distortion," in *Proceedings of 1st International Conference on Image Processing*, vol. 2, pp. 982–986 vol.2, 1994.

[67] A. Eskicioglu and P. Fisher, "Image quality measures and their performance," *IEEE Transactions on Communications*, vol. 43, no. 12, pp. 2959–2965, 1995.

[68] Z. Wang and A. Bovik, "A universal image quality index," *IEEE Signal Processing Letters*, vol. 9, no. 3, pp. 81–84, 2002.

[69] Z. Wang and A. C. Bovik, "Mean squared error: Love it or leave it? a new look at signal fidelity measures," *IEEE Signal Processing Magazine*, vol. 26, no. 1, pp. 98–117, 2009.

[70] L. Zhang, L. Zhang, X. Mou, and D. Zhang, "Fsim: A feature similarity index for image quality assessment," *IEEE Transactions on Image Processing*, vol. 20, no. 8, pp. 2378–2386, 2011.

[71] Z. Wang, E. Simoncelli, and A. Bovik, "Multiscale structural similarity for image quality assessment," in *The Thrity-Seventh Asilomar Conference on Signals, Systems Computers, 2003*, vol. 2, pp. 1398–1402 Vol.2, 2003.

[72] Y. Gao, A. Rehman, and Z. Wang, "Cw-ssim based image classification," in *2011 18th IEEE International Conference on Image Processing*, pp. 1249–1252, 2011.

[73] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang, "The unreasonable effectiveness of deep features as a perceptual metric," 2018.

[74] M. Martinez-Garcia, P. Cyriac, T. Batard, M. Bertalmí o, and J. Malo, "Derivatives and inverse of cascaded linearnonlinear neural models," *PLOS ONE*, vol. 13, p. e0201326, oct 2018.