Aplicación de técnicas de Deep Learning a la clasificación de nubes

Javier Prieto Cepeda
Programación Automática
Máster en Ciencia y Tecnología Informática
Universidad Carlos III de Madrid
Leganés, Madrid, España
Email: 100307011@alumnos.uc3m.es

***Resumen*—Las técnicas de *Deep Learning* permiten modelos computacionales que están compuestos de múltiples capas de procesamiento para aprender representaciones de datos con diferentes niveles de abstracción. Estos métodos han permitido mejorar drásticamente técnicas en reconocimiento de voz u objetos visuales, en sistemas de recomentación y en muchos otros dominios, como descubrimiento de fármacos y genómica. El aprendizaje profundo o *Deep Learning* divide una estructura compleja en grandes conjuntos de datos mediante el uso del algoritmo de retropropagación para indicar cómo una máquina debe cambiar sus parámetros internos que se utilizan para calcular la representación en cada capa a partir de la representación en la capa anterior. Las redes convolucionales profundas han producido adelantos en el procesamiento de imágenes, video, voz y audio, mientras que las redes recurrentes han iluminado los datos secuenciales, como el texto y el habla. En este trabajo, nos centramos en la utilización de técnicas de Deep Learning para la clasificación de imágenes de nubes tomadas por un equipo de meteorólogos de Jaén (España).**

Palabras clave: **Deep Learning, Classification Algorithms, Artificial Intelligence**

1. Introducción

El aprendizaje automático potencia muchos aspectos de la sociedad moderna: desde búsquedas en la web hasta el filtrado de contenido en redes sociales, pasando por recomendaciones sobre sitios web de comercio electrónico, estando cada vez más presente en productos de consumo como cámaras y smartphones. Los sistemas de aprendizaje automático se utilizan para identificar objetos en imágenes, transcribir un discurso en texto, unir noticias, publicaciones o productos con los intereses de los usuarios y seleccionar resultados relevantes de la búsqueda. Estas aplicaciones utilizán cada vez más una clase de técnicas llamadas *Deep Learning*.

Las técnicas convencionales de aprendizaje automático tenían una capacidad limitada para el procesamiento de datos naturales en su forma original. Durante décadas, la construcción de un sistema de reconocimiento de patrones o aprendizaje automático requería una ingeniería cuidadosa y una experiencia importante en el dominio para diseñar una herramienta que extraiga características que posteriormente transformará los datos brutos (como los valores de píxel de una imagen) en una representación interna o vector de características adecuado a partir del cual el subsistema de aprendizaje, a menudo un clasificador, podría detectar o clasificar patrones en la entrada.

El aprendizaje de representación es un conjunto de métodos que permite alimentar una máquina con datos sin procesar y descubrir automáticamente las representaciones necesarias para la detección o clasificación. Los métodos de aprendizaje profundo son métodos de representación-aprendizaje con múltiples niveles de representación, obtenidos mediante la composición de módulos simples pero no lineales que transforman la representación en un nivel (comenzando con la entrada en bruto) en una representación a un nivel más alto, con mayor abstracción . Con la composición de tales transformaciones, se pueden aprender funciones muy complejas. Para las tareas de clasificación, las capas más altas de representación amplifican aspectos de la entrada que son importantes para la discriminación y suprimen las variaciones irrelevantes. Una imagen, por ejemplo, viene en forma de una matriz de valores de píxeles, y las características aprendidas en la primera capa de representación representan típicamente la presencia o ausencia de bordes en orientaciones y ubicaciones particulares en la imagen. La segunda capa típicamente detecta patrones al detectar modificaciones particulares de bordes, independientemente de pequeñas variaciones en las posiciones de los bordes. La tercera capa puede ensamblar patrones en combinaciones más grandes que corresponden a partes de objetos familiares, y las capas posteriores detectarían objetos como combinaciones de estas partes. El aspecto clave del aprendizaje profundo es que estas capas de características no están diseñadas por ingenieros humanos: se aprenden de los datos utilizando un procedimiento de aprendizaje de propósito general.

El aprendizaje profundo está logrando avances importantes en la solución de problemas que han resistido los mejores intentos de la comunidad durante muchos años. Resultó ser muy bueno para descubrir estructuras complejas en datos de alta dimensión y, por lo tanto, es aplicable a muchos dominios de la ciencia, las empresas y el gobierno.

Además de batir récords en reconocimiento de imágenes [1] y reconocimiento de voz, ha superado a otras técnicas de aprendizaje automático para predecir la actividad de moléculas de potenciales fármacos, analizando datos de un acelerador de partículas [2], reconstruyendo circuitos cerebrales y prediciendo los efectos de mutaciones en ADN no codificante en la expresión genética y de enfermedad. Tal vez lo más sorprendente es que el aprendizaje profundo ha producido resultados extremadamente prometedores para diversas tareas de comprensión del lenguaje natural, en particular la clasificación de temas, el análisis de sentimientos, la respuesta a preguntas y la traducción de idiomas.

La comunidad científica cree que el aprendizaje profundo tendrá muchos más éxitos en un futuro cercano, ya que requiere muy poca ingeniería manual, por lo que puede aprovechar fácilmente los aumentos en la cantidad de cálculos y datos disponibles. Los nuevos algoritmos y arquitecturas de aprendizaje que se están desarrollando actualmente para redes neuronales profundas solo acelerarán este progreso.

La estructura del documento será la siguiente. En primer lugar, en la Sección 2 se realizará una breve presentación de las diferentes técnicas utilizadas para la clasificación de imágenes; en la Sección 3 se explicará el problema, detallando la solución propuesta en la Sección 3.1; en la Sección 4 se explicará la evaluación de la solución propuesta; y por último, en la Sección 5 se expondrán las conclusiones finales.

2. Estado de la cuestión

Esta sección ofrece una breve presentación sobre las diversas técnicas utilizadas para la clasificación de imágenes, realizando una pequeña explicación de su funcionamiento.

2.1. Aprendizaje supervisado

La forma más común a la hora de realizar aprendizaje automático, profundo o no, es el aprendizaje supervisado. Imagine que queremos construir un sistema que pueda clasificar las imágenes como conteniendo, por ejemplo, una casa, un automóvil, una persona o una mascota. Primero recopilamos un conjunto grande de imágenes de casas, automóviles, personas y mascotas, cada una etiquetada con su categoría. Durante el entrenamiento, la máquina muestra una imagen y produce un resultado en forma de un vector de pesos, uno para cada categoría. Queremos que la categoría deseada tenga el peso más alto de todas las categorías, pero es poco probable que esto ocurra antes del entrenamiento. Calculamos una función objetivo que mide el error (o distancia) entre los pesos de salida y el patrón deseado de pesos. La máquina luego modifica sus parámetros internos para reducir este error. Estos parámetros, llamados pesos, son números que definen la función de entrada/salida de la máquina. En un sistema típico de aprendizaje profundo, puede haber cientos de millones de estos pesos y cientos de millones de ejemplos etiquetados con los que entrenar la máquina. Para ajustar correctamente el vector de pesos, el algoritmo de aprendizaje calcula un vector de gradiente que, para cada peso, indica en qué cantidad el error aumentaría o disminuiría si la puntuación aumentara en una pequeña cantidad. El vector de pesos se ajusta después en la dirección opuesta al vector de gradiente.

La función objetivo, promediada sobre todos los ejemplos de entrenamiento, puede verse como una especie de paisaje montañoso en el espacio de valores de puntuación de alta dimensión. El vector de gradiente negativo indica la dirección del descenso más significativo en este paisaje, acercándolo a un mínimo, donde el error de salida es bajo en promedio. En la práctica, la mayoría de los profesionales usan un procedimiento llamado descenso de gradiente estocástico (SGD). Esta técnica consiste en mostrar el vector de entrada para algunos ejemplos, calcular las salidas y los errores, calcular el gradiente promedio para esos ejemplos y ajustar los pesos en consecuencia. El proceso se repite para muchos pequeños conjuntos de ejemplos del conjunto de entrenamiento hasta que el promedio de la función objetivo deja de disminuir. Se denomina estocástico puesto que cada pequeño conjunto de ejemplos proporciona una estimación ruidosa del gradiente promedio en todos los ejemplos. Este procedimiento simple generalmente encuentra un buen conjunto de pesos sorprendentemente rápido cuando se compara con técnicas de optimización [3] mucho más elaboradas. Después del entrenamiento, el rendimiento del sistema se mide en un conjunto diferente de ejemplos llamado conjunto de test. Esta técnica sirve para poner a prueba la capacidad de generalización de la máquina: su capacidad para producir respuestas sensatas en nuevos ejemplos que nunca ha visto durante el entrenamiento.

Muchas de las aplicaciones prácticas actuales del aprendizaje automático usan clasificadores lineales además de las funciones diseñadas a mano. Un clasificador lineal de dos clases calcula una suma ponderada de los componentes del vector de características. Si la suma ponderada está por encima de un umbral, la entrada se clasifica como perteneciente a una categoría particular.

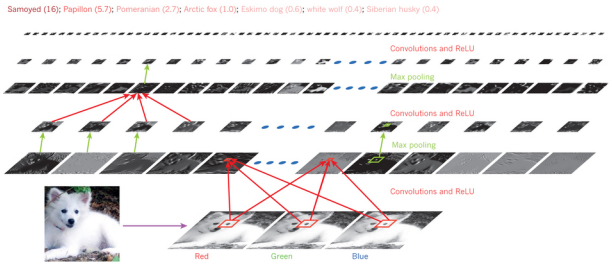

Figura 1: Ejemplo red lobo-samoyedo.

Desde la década de 1960, hemos sabido que los clasificadores lineales solo pueden dividir su espacio de entrada en regiones muy simples, es decir, medio espacio separado por un hiperplano. Pero problemas como el reconocimiento de imágenes y voz requieren que la función de entrada/salida no sea sensible a variaciones irrelevantes de la entrada, como variaciones en la posición, orientación o iluminación de un objeto, o variaciones en el tono o acento del habla, mientras que son muy sensibles a variaciones mínimas particulares

(por ejemplo, la diferencia entre un lobo blanco y una raza de perro blanco parecido a un lobo llamado samoyedo)(). A nivel de píxel, las imágenes de dos samoyedos en diferentes poses y en diferentes ambientes pueden ser muy diferentes entre sí, mientras que dos imágenes de un samoyedo y un lobo en la misma posición y en fondos similares pueden ser muy similares entre sí. Un clasificador lineal, o cualquier otro clasificador "superficial"que opere en píxeles brutos, no podría distinguir los dos últimos, mientras que los dos primeros los ubica en la misma categoría. Esta es la razón por la cual los clasificadores superficiales requieren de un buen extractor de características que resuelva el dilema de selectividad/invarianza, que produce representaciones que son selectivas a los aspectos de la imagen que son importantes para la discriminación, pero que son invariables a aspectos irrelevantes como la pose del animal. Para hacer que los clasificadores sean más potentes, se pueden usar funciones genéricas no lineales, como con los métodos kernel [4], pero las características genéricas, como las que surgen con el kernel gaussiano, no permiten al alumno generalizar bien lejos de los ejemplos de capacitación. La opción convencional es diseñar a mano buenos extractores de características, lo que requiere una considerable cantidad de habilidades de ingeniería y experiencia en el dominio. Pero todo esto puede evitarse si se pueden aprender buenas características automáticamente mediante un procedimiento de aprendizaje de propósito general. Esta es la ventaja clave del aprendizaje profundo.

Una arquitectura de aprendizaje profundo es una pila multicapa de módulos simples, todos (o la mayoría) de los cuales están sujetos al aprendizaje, y muchos de los cuales computan asignaciones de entrada/salida no lineales. Cada módulo en la pila transforma su entrada para aumentar tanto la selectividad como la invariancia de la representación. Con múltiples capas no lineales, por ejemplo, una profundidad de 5 a 20, un sistema puede implementar funciones extremadamente intrincadas de sus entradas que son simultáneamente sensibles a los detalles minúsculos, distinguiendo a los samoyedos de los lobos blancos e insensibles a las grandes variaciones irrelevantes como el fondo,pose, iluminación y objetos circundantes.

2.2. Retropropagación para el entrenamiento de arquitecturas multicapa

Desde los primeros días del reconocimiento de patrones, el objetivo de los investigadores ha sido reemplazar las funciones de ingeniería manual por redes multicapa entrenables, pero a pesar de su simplicidad, la solución no fue ampliamente comprendida hasta mediados de los años ochenta. Como resultado, las arquitecturas multicapa se pueden entrenar por simple descenso del gradiente estocástico. Siempre que los módulos sean funciones relativamente suaves de sus entradas y de sus ponderaciones internas, se pueden calcular gradientes utilizando el procedimiento de retropropagación. La idea de que esto podría hacerse, y que funcionó, fue descubierta de manera independiente por varios grupos durante los años setenta y ochenta.

El procedimiento de retropropagación para calcular el gradiente de una función objetivo con respecto a los pesos de una pila de módulos multicapa no es más que una aplicación práctica de la regla de la cadena para derivadas. La idea clave es que la derivada (o gradiente) del objetivo con respecto a la entrada de un módulo se puede calcular trabajando hacia atrás desde el gradiente con respecto a la salida de ese módulo (o la entrada del módulo siguiente). La ecuación de retropropagación puede aplicarse repetidamente para propagar gradientes a través de todos los módulos, comenzando desde la salida en la parte superior (donde la red produce su predicción) hasta la parte inferior (donde se alimenta la entrada externa). Una vez que se han calculado estos gradientes, es sencillo calcular los gradientes con respecto a los pesos de cada módulo.

Muchas aplicaciones de aprendizaje profundo utilizan arquitecturas de red neuronal predictivas, que aprenden a mapear una entrada de tamaño fijo (por ejemplo, una imagen) a una salida de tamaño fijo (por ejemplo, una probabilidad para cada una de varias categorías). Para ir de una capa a la siguiente, un conjunto de unidades calcula una suma ponderada de sus entradas de la capa anterior y pasa el resultado a través de una función no lineal. En la actualidad, la función no lineal más popular es la unidad lineal rectificada (ReLU), que es simplemente el rectificador de media onda $f(z) = max(z, 0)$. En décadas pasadas, las redes neuronales usaban funciones no lineales más suaves, como $tanh(z)$ o $1/(1 + exp(-z))$, pero la ReLU generalmente aprende mucho más rápido en redes con muchas capas, lo que permite el entrenamiento no supervisado de una red profunda supervisada [5]. Las unidades que no están en la capa de entrada o salida se denominan convencionalmente unidades ocultas. Se puede ver que las capas ocultas distorsionan la entrada de forma no lineal, de modo que las categorías se vuelven linealmente separables por la última capa.

A finales de la década de 1990, las redes neuronales y la retropropagación fueron en gran medida abandonadas por la comunidad de aprendizaje automático e ignoradas por las comunidades de visión artificial y reconocimiento de voz. Se pensó ampliamente que el aprendizaje de extractores de características útiles, de etapas múltiples, con poco conocimiento previo, no era factible. En particular, se pensaba comúnmente que el descenso simple del gradiente quedaría atrapado en mínimos locales pobres: configuraciones de peso para las cuales un pequeño cambio reduciría el error promedio.

En la práctica, los mínimos locales pobres rara vez son un problema con redes grandes. Independientemente de las condiciones iniciales, el sistema casi siempre alcanza soluciones de calidad muy similar. Los recientes resultados teóricos y empíricos sugieren con firmeza que los mínimos locales no son un problema serio en general. En cambio, el paisaje está repleto de una gran cantidad combinatoria de puntos críticos donde el gradiente es cero, y la superficie se curva hacia arriba en la mayoría de las dimensiones y se curva hacia abajo en el resto [6]. El análisis parece mostrar que los puntos críticos con solo unas pocas direcciones de

curvatura hacia abajo están presentes en grandes cantidades, pero casi todos tienen valores muy similares de la función objetivo. Por lo tanto, no importa mucho en cuáles de estos puntos críticos el algoritmo se atasque.

El interés en las redes feedforward profundas se revivió alrededor de 2006 por un grupo de investigadores reunidos por el Instituto Canadiense de Investigación Avanzada (CIFAR). Los investigadores introdujeron procedimientos de aprendizaje no supervisados que podrían crear capas de detectores de funciones sin requerir datos etiquetados. El objetivo de aprender cada capa de detectores de funciones era poder reconstruir o modelar las actividades de los detectores de funciones (o entradas sin procesar) en la capa siguiente. Con "preentrenamiento"varias capas de detectores de funciones progresivamente más complejas que utilizan este objetivo de reconstrucción, los pesos de una red profunda podrían inicializarse a valores razonables. Luego, se podría agregar una capa final de unidades de salida a la parte superior de la red y todo el sistema profundo podría ajustarse utilizando la retropropagación estándar [7]. Esta técnica funcionó notablemente bien para reconocer dígitos escritos a mano o para detectar peatones, especialmente cuando la cantidad de datos etiquetados era muy limitada.

La primera aplicación importante de este enfoque de preentrenamiento fue el reconocimiento de voz, y fue posible gracias a la aparición de unidades de procesamiento de gráficos (GPU) que eran convenientes para el programa y permitieron a los investigadores entrenar redes 10 o 20 veces más rápido. En 2009, el enfoque se usó para mapear ventanas temporales cortas de coeficientes extraídos de una onda de sonido a un conjunto de probabilidades para los diversos fragmentos de discurso que podrían representarse mediante el cuadro en el centro de la ventana. Logró resultados récord en un estándar de referencia de reconocimiento de voz que usaba un vocabulario pequeño y se desarrolló rápidamente para dar resultados récord en una tarea de vocabulario grande. En 2012, muchos de los principales grupos de habla estaban desarrollando versiones de la red profunda de 2009 y ya se estaban implementando en teléfonos con el sistema operativo Android. Para conjuntos de datos más pequeños, el preentrenamiento no supervisado ayuda a prevenir el sobreajuste, lo que permite una generalización significativamente mejor cuando el número de ejemplos etiquetados es pequeño o en un entorno de transferencia donde tenemos muchos ejemplos para algunas tareas de fuente pero muy pocos para algunas tareas objetivo. Una vez que se había rehabilitado el aprendizaje profundo, resultó que la etapa previa al entrenamiento solo era necesaria para pequeños conjuntos de datos.

Sin embargo, hubo un tipo particular de red profunda y directa que era mucho más fácil de entrenar y generalizar mucho mejor que las redes con conectividad total entre capas adyacentes. Esta era la red neuronal convolucional (ConvNet) [8]. Logró muchos éxitos prácticos durante el período en que las redes neuronales perdieron popularidad y recientemente ha sido ampliamente adoptado por la comunidad de visión artificial.

2.3. Redes neuronales convolucionales

Los ConvNets están diseñados para procesar datos que se presentan en forma de matrices múltiples, por ejemplo, una imagen en color compuesta por tres matrices 2D que contienen intensidades de píxeles en los tres canales de color. Muchas modalidades de datos están en forma de matrices múltiples: 1D para señales y secuencias, incluido el lenguaje; 2D para imágenes o espectrogramas de audio; y 3D para video o imágenes volumétricas. Hay cuatro ideas clave detrás de ConvNets que aprovechan las propiedades de las señales naturales: conexiones locales, pesos compartidos, agrupamiento y el uso de muchas capas.

La arquitectura de una ConvNet típica está estructurada como una serie de etapas. Las primeras etapas se componen de dos tipos de capas: capas convolucionales y capas de agrupamiento. Las unidades en una capa convolucional se organizan en mapas de características, dentro de las cuales cada unidad se conecta a parches locales en los mapas de características de la capa anterior a través de un conjunto de pesos denominado banco de filtros. El resultado de esta suma ponderada local se pasa luego a través de una función no-lineal como una ReLU. Todas las unidades en un mapa de características comparten el mismo banco de filtros. Diferentes mapas de características en una capa usan diferentes bancos de filtros. El motivo de esta arquitectura es doble:

- En primer lugar, en los datos de matriz, como las imágenes, los grupos locales de valores suelen estar altamente correlacionados, formando patrones locales distintivos que se detectan fácilmente.
- Segundo, las estadísticas locales de imágenes y otras señales son invariables a la ubicación. En otras palabras, si un patrón puede aparecer en una parte de la imagen, podría aparecer en cualquier lugar, de ahí la idea de que las unidades en diferentes ubicaciones compartan los mismos pesos y detecten el mismo patrón en diferentes partes de la matriz. Matemáticamente, la operación de filtrado realizada por un mapa de características es una convolución discreta, de ahí el nombre.

Aunque la función de la capa convolucional es detectar conjunciones locales de características de la capa anterior, la función de la capa de agrupación es fusionar características semánticamente similares en una. Debido a que las posiciones relativas de las características que forman un patrón pueden variar un poco, la detección fiable del patrón se puede hacer mediante una granulación gruesa de la posición de cada característica. Una unidad de agrupación típica calcula el máximo de un parche local de unidades en un mapa de características (o en algunos mapas de características). Las unidades de agrupación vecinas reciben información de parches que se desplazan en más de una fila o columna, lo que reduce la dimensión de la representación y crea una invarianza a pequeños cambios y distorsiones. Se apilan dos o tres etapas de convolución, no linealidad y agrupamiento, seguidas por capas más convolucionales y completamente

conectadas. Retropropagar gradientes a través de ConvNet es tan simple como a través de una red profunda regular, lo que permite entrenar a todos los pesos en todos los bancos de filtros.

Las redes neuronales profundas explotan la propiedad de que muchas señales naturales son jerarquías de composición, en las que las características de nivel superior se obtienen componiendo las de nivel inferior. En las imágenes, las combinaciones locales de bordes forman patrones, los patrones se ensamblan en partes y las partes forman objetos. Existen jerarquías similares en el habla y el texto, desde sonidos a teléfonos, fonemas, sílabas, palabras y oraciones. La agrupación permite que las representaciones varíen muy poco cuando los elementos en la capa anterior varían en posición y apariencia.

Las capas convolucionales y de agrupamiento en ConvNets están directamente inspiradas en las nociones clásicas de células simples y células complejas en neurociencia visual, y la arquitectura general recuerda a la jerarquía LGN-V1-V2-V4-IT en la vía ventral de la corteza visual. Cuando los modelos ConvNet y los monos se muestran con la misma imagen, las activaciones de unidades de alto nivel en ConvNet explican la mitad de la varianza de conjuntos aleatorios de 160 neuronas en la corteza inferotemporal del mono [9]. Los ConvNets tienen sus raíces en el neocognitrón, cuya arquitectura era algo similar, pero no tenía un algoritmo de aprendizaje supervisado de extremo a extremo, como la retropropagación. Una primitiva 1D ConvNet llamada red neuronal de retardo se usó para el reconocimiento de fonemas y palabras simples.

Han habido numerosas aplicaciones de redes convolucionales que se remontan a principios de la década de 1990, comenzando con redes neuronales de retardo de tiempo para reconocimiento de voz y lectura de documentos. El sistema de lectura de documentos utilizó una ConvNet entrenada conjuntamente con un modelo probabilístico que implementó restricciones de idioma. A fines de la década de 1990, este sistema estaba leyendo más del 10 % de todos los cheques en los Estados Unidos. A continuación, Microsoft implementó varios sistemas ópticos de reconocimiento de caracteres y reconocimiento de caracteres basados en ConvNet. ConvNets también se experimentó a principios de la década de 1990 para la detección de objetos en imágenes naturales, incluidas las caras y las manos, y para el reconocimiento facial.

2.4. Reconocimiento de imagen con redes convolucionales profundas

Desde principios de la década de 2000, ConvNets se han aplicado con gran éxito a la detección, segmentación y reconocimiento de objetos y regiones en imágenes. Estas fueron todas las tareas en las que los datos etiquetados fueron relativamente abundantes, como el reconocimiento de señales de tráfico, la segmentación de imágenes biológicas particularmente para la conectómica y la detección de caras, texto, peatones y cuerpos humanos en imágenes naturales [10]. Un gran éxito práctico reciente de ConvNets es el reconocimiento facial.

Es importante destacar que las imágenes se pueden etiquetar a nivel de píxel, lo que tendrá aplicaciones en la tecnología, incluidos los robots móviles autónomos y los automóviles autónomos. Compañías como Mobileye y NVIDIA están utilizando dichos métodos basados en ConvNet en sus próximos sistemas de visión para automóviles. Otras aplicaciones que adquieren importancia implican la comprensión del lenguaje natural y el reconocimiento del habla.

A pesar de estos éxitos, ConvNets fueron en gran parte abandonadas por las comunidades de visión computacional y aprendizaje automático hasta la competencia de ImageNet en 2012. Cuando se aplicaron redes convolucionales profundas a un conjunto de datos de alrededor de un millón de imágenes de la web que contenían 1,000 clases diferentes, lograron resultados espectaculares, casi reduciendo a la mitad las tasas de error de los mejores enfoques competitivos. Este éxito provino del uso eficiente de GPU, ReLUs, dropout y técnicas para generar más ejemplos de entrenamiento al deformar los existentes. Este éxito ha provocado una revolución en la visión artificial; ConvNets son ahora el enfoque dominante para casi todas las tareas de reconocimiento y detección [11], y abordan el rendimiento humano en algunas tareas. Una impresionante demostración reciente combina ConvNets y módulos netos recurrentes para la generación de subtítulos de imagen.

Las arquitecturas recientes de ConvNet tienen de 10 a 20 capas de ReLU, cientos de millones de pesos y miles de millones de conexiones entre unidades. Mientras que entrenar redes tan grandes podría haber tomado semanas hace solo dos años, el progreso en la paralelización de hardware, software y algoritmos ha reducido los tiempos de capacitación a unas pocas horas.

El rendimiento de los sistemas de visión basados en ConvNet ha causado que la mayoría de las principales compañías tecnológicas, incluidas Google, Facebook, Microsoft, IBM, Yahoo, Twitter y Adobe, así como un número cada vez mayor de nuevas empresas para iniciar proyectos de investigación y desarrollo e implementar productos y servicios de comprensión de imágenes basados en ConvNet.

ConvNets son fácilmente susceptibles de implementaciones de hardware eficientes en chips o matrices de compuertas programables en el campo. Varias compañías como NVIDIA, Mobileye, Intel, Qualcomm y Samsung están desarrollando chips ConvNet para permitir aplicaciones de visión en tiempo real en teléfonos inteligentes, cámaras, robots y automóviles sin conductor.

3. Clasificación de nubes

En el año 1896 se publicó por primera vez el Atlas Internacional de Nubes [12], una publicación que constaba únicamente de 18 imágenes en color de diferentes tipos de nubes. En el año 2017 se actualizó la versión de este Atlas, que llevaba sin ser actualizado desde el año 1987. Esta actualización, incorporaba cientos de fotografías con

12 nuevos tipos de nubes. Al igual que los animales, las nubes son clasificadas en géneros, especies y otras categorías inferiores, con nombres en latín. Estos géneros, describen la apariencia aproximada de las nubes y dónde se forman, como ocurre con los altocumulus. En la edición de 2017, se describe una nueva especie bautizada como *Volutus*. Esta especie es una masa nubosa de escasa altura, que parece enrollarse en un eje horizontal.

Uno de los principales beneficios que se obtienen de la comprensión de las nubes, es la predicción del tiempo. A la hora de modelizar el sistema climático, es necesario entender correctamente las nubes. Una de las principales causas por la cual se desea modelar este sistema, es la escasez de agua que se está viviendo en los últimos años y que cada vez se hace más notable debido al cambio climático. Para evitar esta escasez, se hace indispensable la creación de un sistema que permita predecir la disponibilidad de recursos hídricos.

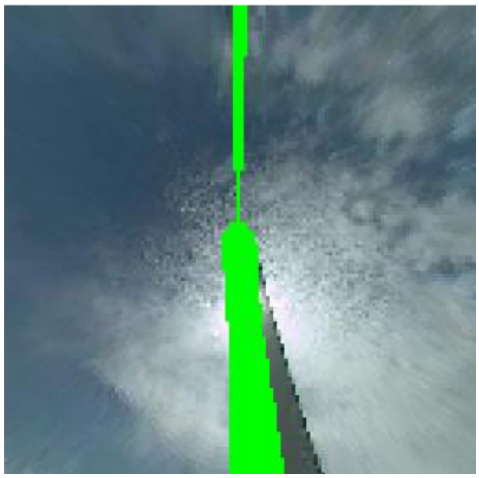

Figura 2: Imagen de nube Cirros.

La comunidad autónoma de Andalucía (España), es una de las zonas más afectadas en los últimos años por el cambio climático. La sequía y la desertificación avanzan año tras año, y por ello, un conjunto de meteorólogos de la provincia de Jaén trabaja actualmente en el estudio de los diferentes tipos de nubes. Para ello han tomado como muestra 737 fotografías del cielo en color (formato RGB) con una resolución de 256x256 píxeles. Uno de los problemas que existen, es la complejidad que existe a la hora de obtener las características de cada tipo de nube para poder realizar su clasificación, y por ello se desea realizar el diseño e implementación de un prototipo que permita diferenciar las nubes de tipo *Cirrus* y *Cirrostratus* con respecto al resto de tipos de nubes.

3.1. Solución propuesta

Para lograr la configuración final, se ha realizado un proceso empírico hasta lograr unos resultados aceptables teniendo en cuenta los recursos computacionales de los cuales se disponía. El modelo diseñado, cuenta con 11 capas (sin contar las capas de normalización batch, que sirve para normalizar los valores de salida de la capa de convolución).

- Capa 1: Capa convolucional con 64 filtros y dimensiones de 11*11.
- Capa 2: Capa de pooling para la reducción de dimensiones (downsampling).
- Capa 3: Capa convolucional con 32 filtros y dimensiones de 7*7.
- Capa 4: Capa de pooling para la reducción de dimensiones (downsampling).
- Capa 5: Capa convolucional con 32 filtros y dimensiones de 7*7.
- Capa 6: Capa de pooling para la reducción de dimensiones (downsampling).
- Capa 7: Capa convolucional con 16 filtros y dimensiones de 5*5.
- Capa 8: Capa de pooling para la reducción de dimensiones (downsampling).
- Capa 9: Capa densa.
- Capa 10: Capa densa.
- Capa 11: Capa densa de salida con función de activación *Sigmoid*.

Como se puede ver, las 8 primeras capas son 4 conjuntos de capa convolucional más capa de pooling para la realización del *downsampling*. El motivo por el cual el número de filtros se ve reducido se debe a que cuantos más filtros se aplican, más características de alto nivel se obtienen de la imagen; como por ejemplo bordes. Conforme se reduce el tamaño de la porción de la imagen analizada, el número de filtros aplicados se va reduciendo, es decir, se cogen detalles de bajo nivel hasta formar objetos más complejos. A la salida de cada uno de estos conjuntos hay una capa de normalización batch, que se encarga de normalizar los valores de salida de la capa de convolución. Las últimas 3 capas (incluyendo la capa de salida) reciben la salida de las capas convolucionales, es decir, las características de la imagen extraídas, y se encargan de realizar el último paso del aprendizaje. La capa de salida tiene una función de activación *Sigmoid* debido al tipo de problema que estamos resolviendo, un problema de clasificación binaria de una única neurona (0 si es el tipo de nube, 1 si no es el tipo de nube).

4. Evaluación

Para observar el comportamiento de las solución propuesta, se han realizado dos implementaciones distintas. Por un lado, se han realizado ejecuciones con el conjunto de datos íntegro, mientras que para buscar optimizar el rendimiento del modelo se han realizado ejecuciones mediante la validación cruzada. A la hora de realizar la validación cruzada, se ha ordenado el conjunto de datos por fecha de realización de la foto. Esto es debido a que el intervalo de tiempo transcurrido entre una imagen y otra es muy pequeño, por lo que la diferencia entre una imagen y la anterior es prácticamente ínfima, teniendo como resultado la misma imagen. Con esta mejora, se pretende garantizar

que los resultados son independientes entre los datos de entrenamiento y los datos de test. Para ello, se ha dividido en conjunto de datos en 5 partes, realizando entrenamiento con 4 y el test con la parte restante. El calculo de la tasa de acierto es el resultado de la media entre todas las combinaciones posibles de entrenamiento/test.

Tabla 1: Comparación resultados.

Tipo de modelo	Epochs	Número de Conjuntos	Acierto %
Normal	2	*	**68.02**
Normal	4	*	44.97
Normal	8	*	42.60
Normal	10	*	65.09
Validación cruzada	2	5	**71.15**
Validación cruzada	4	5	64.05
Validación cruzada	8	5	49.94
Validación cruzada	10	5	63.39

Como se puede observar en la Tabla 1, en cuanto a la implementación Normal (sin separación de entrenamiento y test) los mejores resultados se obtienes con un valor de epochs igual a 10. En cuanto la implementación con validación cruzada, los resultados obtenidos son más fiables e incluso se obtiene una tasa de acierto media superior, siendo la mejor configuración con un valor de epochs igual a 2. La mejor tasa de acierto obtenida es del 71.15 %, un resultado aceptable teniendo en cuenta el tamaño del conjunto de datos y el poco número de imágenes por tipo de nube.

5. Conclusiones

El Deep Learning representa un acercamiento más íntimo al modo de funcionamiento del sistema nervioso humano. Nuestro encéfalo tiene una microarquitectura de gran complejidad, en la que se han descubierto núcleos y áreas diferenciados cuyas redes de neuronas están especializadas para realizar tareas específicas. Los modelos computacionales de Deep Learning imitan las características arquitecturales del sistema nervioso, permitiendo que dentro del sistema global haya redes de unidades de proceso que se especialicen en la detección de determinadas características ocultas en los datos. Este enfoque ha permitido mejores resultados en tareas de percepción computacional, si las comparamos con las redes monolíticas de neuronas artificiales.

Como hemos podido observar en los apartados anteriores, gracias a ésta técnica se pueden resolver problemas que previamente eran muy costosos tanto en tiempo de cómputo como en tiempo de diseño del sistema resolutivo. Gracias a herramientas de alto nivel como Keras y TensorFlow, la abstracción a la hora de realizar el diseño de la red reduce la complejidad del problema permitiendo lograr resultados satisfactorios con un esfuerzo significativamente menor a otras técnicas. Por otro lado, resulta de vital importancia la creación de sistemas que permitan la predicción de sucesos de forma que sea posible la creación de protocolos de prevención adecuados para evitar desastres naturales. En el caso expuesto en este trabajo, la importancia de la creación de un sistema de predicción de recursos hídricos permitiría adecuar los niveles de consumo de agua tanto para la población como para la industria (que supone un gran porcentaje de la economía española); o el trasavase de aguas fluviales, que cada vez se hace más necesario para cubrir las necesidades de ciertas regiones de la población en España.

En cuanto a las conclusiones personales, la resolución de este problema como práctica final de la asignatura **Programación Automática** me ha parecido muy interesante a la vez que me ha resultado de gran ayuda para la comprensión de los conceptos vistos en la asignatura y el aprendizaje de uno de los paradigmas más importantes de la actualidad. Pese a la dificultad que suponía no haberme enfrentado nunca a la utilización de técnicas de *Deep Learning*, el trabajo de documentación realizado para comprender el funcionamiento de estos sistemas y los trabajos previos relacionados con la clasificación de imagen ha supuesto una enorme ayuda a la hora de poder diseñar una solución que logre unos resultados aceptables. En definitiva, considero que gracias a esta práctica he logrado adentrarme en un paradigma que desconocía, y he sembrado una base muy importante para poder continuar profundizando en éste área.

Referencias

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[2] C. Adam-Bourdarios, G. Cowan, C. Germain, I. Guyon, B. Kégl, and D. Rousseau, "The higgs boson machine learning challenge," in *NIPS 2014 Workshop on High-energy Physics and Machine Learning*, 2015, pp. 19–55.

[3] L. Bottou and O. Bousquet, "The tradeoffs of large scale learning," in *Advances in neural information processing systems*, 2008, pp. 161–168.

[4] A. J. Smola and B. Schölkopf, *Learning with kernels*. Citeseer, 1998, vol. 4.

[5] X. Glorot, A. Bordes, and Y. Bengio, "Deep sparse rectifier neural networks," in *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, 2011, pp. 315–323.

[6] Y. N. Dauphin, R. Pascanu, C. Gulcehre, K. Cho, S. Ganguli, and Y. Bengio, "Identifying and attacking the saddle point problem in high-dimensional non-convex optimization," in *Advances in neural information processing systems*, 2014, pp. 2933–2941.

[7] Y. Bengio, P. Lamblin, D. Popovici, and H. Larochelle, "Greedy layer-wise training of deep networks," in *Advances in neural information processing systems*, 2007, pp. 153–160.

[8] Y. LeCun, B. E. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. E. Hubbard, and L. D. Jackel, "Handwritten digit recognition with a back-propagation network," in *Advances in neural information processing systems*, 1990, pp. 396–404.

[9] D. J. Felleman and D. E. Van, "Distributed hierarchical processing in the primate cerebral cortex." *Cerebral cortex (New York, NY: 1991)*, vol. 1, no. 1, pp. 1–47, 1991.

[10] Y. Taigman, M. Yang, M. Ranzato, and L. Wolf, "Deepface: Closing the gap to human-level performance in face verification," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 1701–1708.

[11] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun, "Overfeat: Integrated recognition, localization and detection using convolutional networks," *arXiv preprint arXiv:1312.6229*, 2013.

[12] "Cloud ATLAS international cloud atlas," https://cloudatlas.wmo.int/home.html, accessed: 2018-05-31.