Universidad Zaragoza
1542

# Trabajo Fin de Máster

## Estudio y desarrollo de técnicas no supervisadas para la extracción automática de características en datos multimedia

## Study and development of unsupervised learning techniques for automatic feature extraction in multimedia data


Autor

Pablo Alfaro Saracho

Directora

Victoria Mingote Bueno

Ponente

Antonio Miguel Artiaga

Escuela de Ingeniería y Arquitectura


2021

# Agradecimientos

La realización de este trabajo ha sido posible gracias a la ayuda de multitud de personas. En primer lugar, gracias a Victoria y Antonio por el seguimiento y supervisión en el desarrollo y también en la redacción de esta memoria, además de su disponibilidad y disposición a ayudar en todo momento. En segundo lugar, a todos los profesores que he tenido a lo largo de estos 6 años por brindarme los conocimientos y capacidades necesarias para realizar un trabajo de estas características. Por último, a mi familia y amigos por su apoyo incondicional.

# Estudio y desarrollo de técnicas no supervisadas para la extracción automática de características en datos multimedia

## RESUMEN

El objetivo de este Trabajo Fin de Máster es el desarrollo de sistemas automáticos basados en redes neuronales profundas mediante técnicas no supervisadas innovadoras para realizar la extracción de vectores de características representativos sin la utilización de datos etiquetados. Los sistemas resultantes consiguen una mayor capacidad de representación de datos de forma generalizable, permitiendo su aplicación a diferentes tareas con datos de naturaleza muy distinta. Los sistemas se han desarrollado mediante infraestructuras de cálculo masivo distribuido, centrando su aplicación en el tratamiento de datos de vídeo. El trabajo se enmarca en la línea de investigación de aprendizaje automático sobre datos multimedia del *Voice Input Voice Output Laboratory* (ViVoLab) del Instituto de Investigación en Ingeniería de Aragón (I3A) de la Universidad de Zaragoza.

Este trabajo se ha realizado a partir de dos bases de datos de vídeo de diferente tamaño. La primera de ellas se utiliza para el desarrollo de modelos no supervisados, mientras que la segunda se aplica en la evaluación de sus prestaciones en dos tareas secundarias que demuestran su versatilidad y potencia. Para desarrollar estos modelos se han utilizado diferentes arquitecturas, como redes neuronales convolucionales, transformers y redes neuronales recurrentes. Además, al trabajar con señales multimedia, su análisis mediante modelos de secuencia mejora de las capacidades de extracción de información relevante.

La primera de las tareas secundarias mencionadas consiste en un sistema de reconocimiento de acciones mediante la clasificación de un conjunto reducido de vídeos etiquetados. En primer lugar, se construyen varios sistemas supervisados de referencia. Posteriormente, se construye un extractor de características mediante aprendizaje no supervisado. El sistema resultante aprende a representar los datos en un espacio de características potente de forma automática sin etiquetar ningún dato de entrada. Este extractor se aplica a la tarea de clasificación y se comprueba su potencia mediante la comparativa con la referencia supervisada.

En la segunda tarea secundaria se aplican directamente los modelos no supervisados entrenados para la construcción de una herramienta de motor de búsqueda en la que se obtiene, de forma automática a partir de la representación obtenida, un conjunto de vídeos similares a un vídeo de entrada desde el punto de vista de su contenido y temática.

# Study and development of unsupervised learning techniques for automatic feature extraction in multimedia data

## ABSTRACT

The objective of this Master Thesis is the development of automatic systems based on deep neural networks using innovative unsupervised techniques to perform the extraction of representative feature vectors without the use of labeled data. The resulting systems achieve a higher capacity of data representation in a generalizable way, allowing its application to different tasks with data of very different nature. The systems have been developed using massive distributed computing infrastructures, focusing their application on video data processing. The work is part of the research line of machine learning on multimedia data of the Voice Input Voice Output Laboratory (ViVoLab) of the Engineering Research Institute of Aragon (I3A) of the University of Zaragoza.

This work has been carried out using two video databases of different sizes. The first one is used for the development of unsupervised models, while the second one is applied in the evaluation of their performance in two secondary tasks that demonstrate their versatility and power. Different architectures have been used to develop these models, such as convolutional neural networks, transformers and recurrent neural networks. In addition, when working with multimedia signals, their analysis by means of sequence models improves the relevant information extraction capabilities.

The first of the aforementioned secondary tasks consists of an action recognition system by classifying a reduced set of labeled videos. First, several supervised reference systems are built. Subsequently, a feature extractor is built using unsupervised learning. The resulting system learns to represent the data in a powerful feature space automatically without labeling any input data. This extractor is applied to the classification task and tested for its power by benchmarking against the supervised benchmark.

In the second secondary task, the trained unsupervised models are directly applied to the construction of a search engine tool in which a set of videos similar to an input video in terms of content and subject matter are automatically obtained from the obtained representation.

# Índice de contenido

# 1 Introducción

El presente trabajo aborda el estudio y desarrollo de técnicas de aprendizaje automático no supervisado para la construcción de extractores de características aplicados a datos multimedia. Este proyecto se ha desarrollado en el *Voice Input Voice Output Laboratory* (ViVoLab) del Instituto de Investigación en Ingeniería de Aragón (I3A) de la Universidad de Zaragoza. El grupo centra sus trabajos en diferentes áreas, como las tecnologías del habla, del lenguaje y el aprendizaje automático a partir de grandes conjuntos de datos (texto, voz, audio y vídeo), con objetivos tanto de investigación como de aplicación en colaboración con empresas e instituciones públicas. Este Trabajo Fin de Máster queda enmarcado en la línea de investigación de aprendizaje automático sobre datos multimedia, en particular, vídeo.

En este primer apartado se presenta el contexto en el que se ha desarrollado así como los antecedentes en la materia que han servido de referencia. Posteriormente, se describen determinados aspectos en cierto detalle relacionados con los fundamentos y principios técnicos aplicados. Además, se explicarán los objetivos y el alcance que delimitan el proyecto. Finalmente, se introduce la estructura completa seguida en el documento.

## 1.1 Contexto y Antecedentes

Las interacciones entre personas y máquinas han evolucionado enormemente en los últimos años, consiguiendo tecnologías cada vez más intuitivas. Una gran parte de los avances actuales son resultado de la aplicación de técnicas de aprendizaje automático sobre grandes volúmenes de datos. La revolución digital lleva consigo la generación y el almacenamiento de información en todos los ámbitos, con una potencial utilidad en este tipo de sistemas. La cantidad de datos multimedia generados y accesibles de forma pública ha crecido exponencialmente. Este enorme volumen de datos introduce ciertas necesidades, como el análisis y catalogación de dicha información para conseguir una interacción sencilla con ella y su aplicación a otras tareas. Sin embargo, la mayor parte de sistemas actuales se basan en técnicas supervisadas, requiriendo una costosa labor de etiquetado de los datos. En este proyecto se desarrollan sistemas automáticos basados en aprendizaje no supervisado, capaces de extraer información relevante a partir de bases de datos multimedia sin necesidad de etiquetas y que sean generalizables para problemas con necesidades diferentes.

El aprendizaje no supervisado está cobrando cada vez más importancia en la actualidad debido al enorme volumen de datos generados y la inviabilidad de realizar un tratamiento manual de los mismos. En concreto, el desarrollo de métodos de aprendizaje autosupervisado o *self-supervised* se ha incrementado fuertemente en los últimos años. Se pueden encontrar trabajos que compiten en prestaciones con potentes modelos supervisados utilizando diferentes técnicas que permiten explotar datos sin etiquetar [1] [2] [3].

El grupo ViVoLab cuenta con desarrollos anteriores en la materia, utilizando datos multimedia, especialmente, audio. En particular, el Trabajo Fin de Máster "Estudio de técnicas de extracción automática de características de señales de voz mediante aprendizaje no supervisado" [4] tiene un enfoque similar a este proyecto pero aplicado a voz. A pesar de trabajar con vídeo en este proyecto, sus resultados han servido de referencia en la elección de las técnicas utilizadas debido a su gran potencial.

## 1.2 Descripción y Principios

El trabajo realizado consiste en el desarrollo de modelos no supervisados para datos multimedia, concretamente vídeo, y una evaluación de prestaciones en tareas secundarias para mostrar las capacidades de la técnica escogida tanto desde el punto de vista de potencia como de versatilidad. A continuación, se presenta una descripción inicial de los fundamentos teóricos en los que se apoya el proyecto y las características del tipo de dato utilizado.

### 1.2.1 Aprendizaje automático

El aprendizaje automático o *machine learning* consiste en el desarrollo de algoritmos que proporcionan a los sistemas informáticos la capacidad de aprender sin necesidad de una programación específica para cada tarea. En concreto, se plantean una serie de arquitecturas que son entrenadas a partir de unos datos de entrada, obteniendo modelos capaces de resolver una tarea determinada. La idea es imitar el aprendizaje humano. Las redes neuronales son el mayor exponente de estas técnicas en los últimos años. En función de las características del problema y sus datos, existen distintos algoritmos que se agrupan en dos tipos de aprendizaje: supervisado y no supervisado.

#### 1.2.1.1 Aprendizaje supervisado

La mayor parte de desarrollos actuales son fruto del aprendizaje supervisado, basado en conjuntos de datos etiquetados, es decir, con información adicional que identifica o describe los datos de entrada.

#### 1.2.1.2 Aprendizaje no supervisado

El aprendizaje no supervisado es capaz de explotar los datos sin necesidad de etiquetas, es decir, prácticamente en bruto. En muchas ocasiones, su motivación consiste en aprender representaciones útiles de los datos que se puedan extraer y utilizar en otras tareas de diferente naturaleza. De esta forma, se obtienen modelos no finalistas, sino complementarios en la resolución de otros problemas.

En este proyecto se utiliza una técnica de *self-supervised learning*, una clase particular de aprendizaje no supervisado. Se trata de un método en el que, a partir de datos no etiquetados, se generan de forma automática durante el entrenamiento las etiquetas correspondientes para crear una tarea supervisada.

### 1.2.2 Datos multimedia. Vídeo

Los modelos obtenidos mediante técnicas de aprendizaje automático tienen una enorme dependencia de los datos de entrada. Por tanto, la naturaleza de dichos datos determina en muchas ocasiones las arquitecturas utilizadas y sus parámetros de entrenamiento.

Los datos multimedia tienen características específicas respecto a otros tipos. El vídeo, en particular, es una señal formada por *frames* que se presentan en el tiempo, de forma que resultan apropiados modelos que trabajen con secuencias y modelos aplicables a imagen.

## 1.3 Objetivos y Alcance

El objetivo fundamental de este trabajo es el desarrollo de técnicas basadas en redes neuronales profundas no supervisadas para la extracción de vectores de características representativos sin necesidad de utilizar datos etiquetados. Las etapas que definen el alcance del proyecto se especifican a continuación.

### 1.3.1 Revisión literatura científica y estado del arte

La fase inicial consiste en una revisión del estado del arte, lo que ha permitido obtener una visión general de las técnicas no supervisadas actuales que se pueden aplicar sobre datos multimedia y, en particular, vídeo, junto con las arquitecturas de red más avanzadas. Por otro lado, se ha recogido un conjunto amplio de bases de datos de vídeo accesibles de forma pública, caracterizándolas tanto en número de ejemplos como en duración de los mismos. De esta forma, se ha realizado una elección adecuada de los modelos a construir, el tipo de aprendizaje no supervisado a aplicar y las bases de datos utilizadas para su entrenamiento.

### 1.3.2 Experimentos y resultados

Los modelos diseñados se han programado en *Python*, utilizando la librería de aprendizaje automático *PyTorch* [5], de código abierto, aunque desarrollada principalmente por Facebook. Esta librería permite una gran versatilidad en el diseño de experimentos. Los modelos programados son de cierta complejidad y de tamaño considerable, por lo que su entrenamiento se ha realizado en tarjetas gráficas (GPUs). En particular, la mayor parte del trabajo se ha desarrollado dentro del *cluster* del grupo de investigación, con la disponibilidad de GPUs de alta capacidad.

#### 1.3.2.1 Desarrollo y validación de sistemas de aprendizaje no supervisado

La idea fundamental del proyecto consiste en el análisis e implementación de modelos no supervisados y la evaluación de sus prestaciones de forma que quede reflejada la potencia de estas técnicas para resolver distintos problemas de los sistemas automáticos actuales. Para ello, se han diseñado modelos no supervisados bajo una tarea principal y se han comprobado sus prestaciones en varias tareas secundarias. La tarea principal tiene por objetivo obtener un extractor de características potente que se pueda trasladar a otras tareas.

#### 1.3.2.2 Evaluación de prestaciones en tareas secundarias

Las tareas secundarias pueden ser muy diversas y se encargan de reflejar la potencia de las representaciones obtenidas. Por tanto, las tareas secundarias deben utilizar una base de datos diferente a la tarea principal.

El entrenamiento de las tareas principal y secundarias se desarrolla de forma conjunta, siendo estas últimas las que determinan la mejora en los sucesivos modelos. Los modelos diseñados se han modificado a lo largo del proyecto consiguiendo mejores resultados, aunque con un incremento de complejidad.

## 1.4 Cronología del proyecto

En el siguiente diagrama de Gantt (Figura 1.1) se muestra el desarrollo temporal del proyecto.

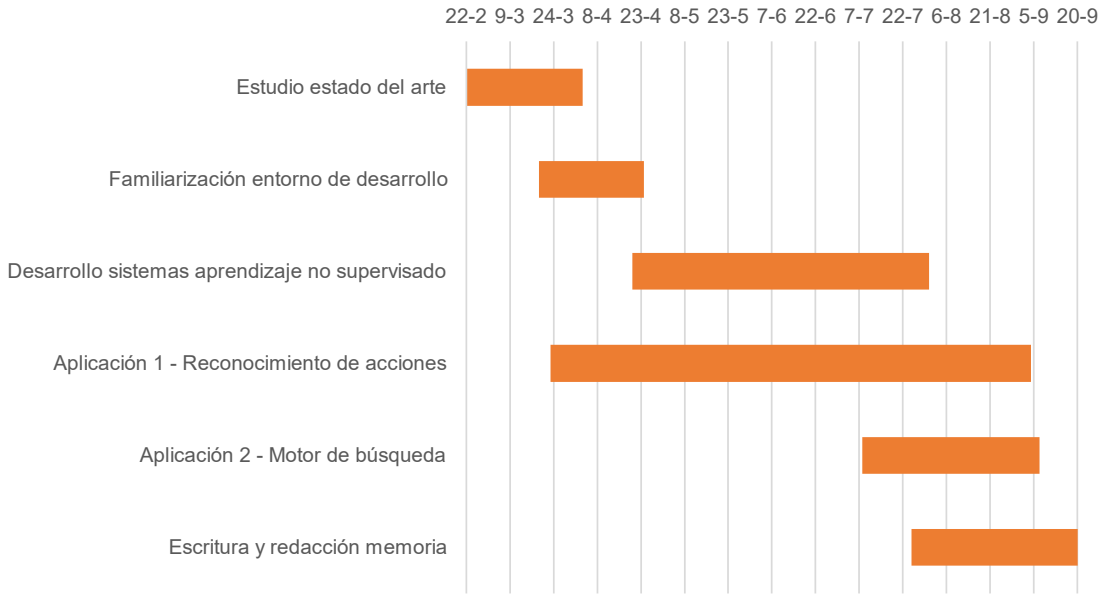


**Figura 1.1.** Diagrama de Gantt del proyecto

## 1.5 Estructura de la memoria

En el segundo capítulo del documento se introduce la problemática que justifica las características del proyecto. Para ello, se explican las limitaciones de los sistemas automáticos actuales desde el punto de vista de las bases de datos así como las características específicas de la información multimedia que determinan las técnicas utilizadas.

En el siguiente capítulo se presenta el enfoque elegido para la resolución del problema, el aprendizaje no supervisado. En concreto, se explica de forma detallada la técnica utilizada tanto de forma cualitativa como matemática. Se trata de una solución apropiada para el trabajo con secuencias y que tiene aplicación directa en datos multimedia.

El cuarto capítulo se centra en las arquitecturas utilizadas en el proyecto. Los modelos desarrollados se pueden agrupar según varios tipos de arquitecturas. En particular, el trabajo realizado ha evolucionado desde arquitecturas conocidas hasta propuestas muy recientes que permiten, a priori, conseguir mejores resultados. A lo largo del capítulo se realiza una descripción de las mismas, junto con una explicación de sus fundamentos teóricos.

En el quinto capítulo se desarrollan los experimentos realizados a lo largo del proyecto junto con los resultados obtenidos. Se han elegido diferentes tareas que permiten mostrar el potencial de la técnica no supervisada como solución a la problemática inicial.

Finalmente, el último capítulo recoge las conclusiones del trabajo junto con las posibles líneas futuras del proyecto.

# 2 Descripción del problema

En la última década, el aprendizaje supervisado se ha impuesto en la construcción de sistemas inteligentes, permitiendo grandes avances y mejoras en la resolución de tareas pertenecientes a distintos dominios. En el campo de la visión por computador, se han conseguido importantes éxitos en aplicaciones como clasificación de imágenes [6] , detección de objetos [7] o segmentación semántica [8].

En la actualidad, la complejidad de las nuevas tareas junto con el aumento de datos requeridos para su entrenamiento plantea la necesidad de utilizar técnicas alternativas de mayor viabilidad y disponibilidad. Además, las soluciones implementadas deben atender a las particularidades de cada dominio, en el caso de este proyecto, datos de vídeo.

## 2.1 Problemas actuales en los *datasets*

El proceso clásico para resolver un problema concreto desde el punto de vista supervisado consiste en etiquetar un conjunto de datos, definir una función de coste objetivo y entrenar una red neuronal sin ningún tratamiento previo. Los algoritmos de entrenamiento han sido ampliamente estudiados y las principales decisiones a tomar están relacionadas con la arquitectura de la red. Sin embargo, los resultados dependen fuertemente de la cantidad y calidad de los datos utilizados, lo que condiciona enormemente el tipo de problemas que se pueden resolver. Además, conforme aumenta la complejidad de la tarea, el tamaño requerido para el conjunto de datos etiquetados se incrementa enormemente [6].

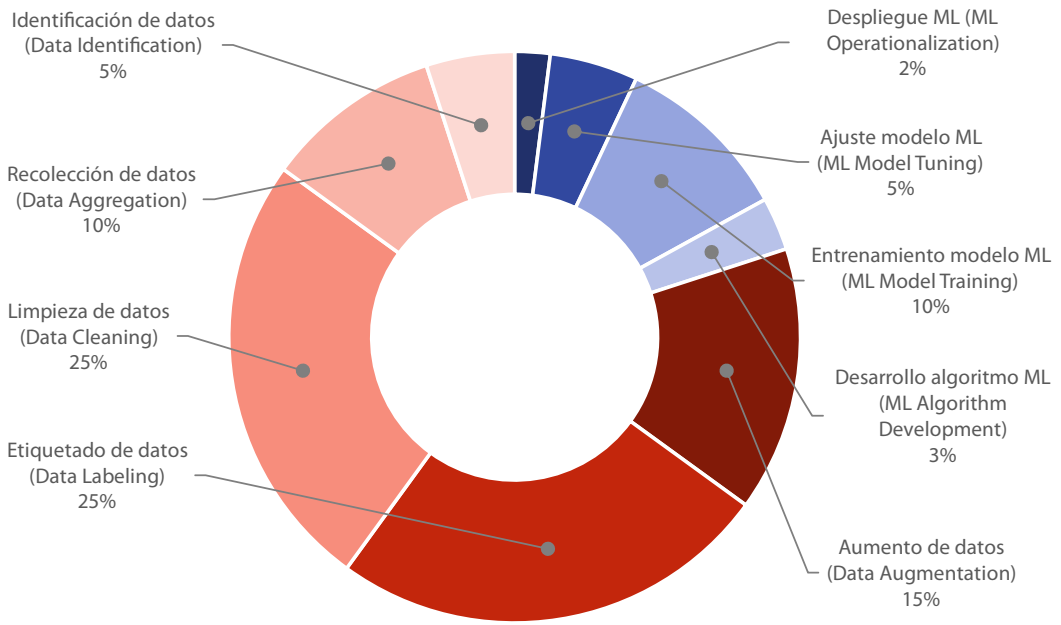


**Figura 2.1.** Distribución temporal de proyectos de *machine learning* [9]

Los estudios muestran que los proyectos de aprendizaje automático destinan hasta un 80% del tiempo a labores relacionadas con los datos utilizados, como se indica en la Figura 2.1. El etiquetado de datos, en particular, puede suponer hasta el 25% de la duración del proyecto, algo que muestra la importancia de la preparación de los datos.

Sin embargo, en muchos problemas reales no se dispone de una base de datos etiquetada del suficiente tamaño ni es posible construirla. El etiquetado de datos es muy caro, requiriendo, en muchas ocasiones, de personas humanas para realizar las correspondientes labores o la contratación de sistemas específicos desarrollados por terceros para obtener dichas etiquetas. Por ejemplo, un servicio de etiquetado de imágenes para una aplicación de segmentación semántica puede tener un coste de hasta 6,40 dólares por imagen, de forma que la construcción de un *dataset* completo de unos 80.000 ejemplos superaría los 500.000 dólares [10]. Las previsiones indican un crecimiento exponencial de este mercado de etiquetado de datos, triplicando su tamaño en los próximos años [11] [12]. Además, existen aplicaciones en las que se necesitan expertos para conseguir este etiquetado como por ejemplo la medicina, suponiendo un coste aún mayor.

Por otro lado, las bases de datos presentan otro tipo de problemas en las tareas actuales. Inicialmente, se construían para ser utilizadas como evaluación de sistemas o modelos construidos por diferentes equipos. El uso de los *datasets* en el entrenamiento de los sistemas introdujo el problema del sesgo de la base de datos [13], que da lugar a modelos y algoritmos con dificultades de generalización. Este sesgo está siempre presente, debido al tamaño finito de estos *datasets*. Por esta razón, se plantean otros puntos de vista o paradigmas basados en un aprendizaje continuo. La idea consiste en una preparación continua de los modelos y tiene una comparación directa con el aprendizaje del ser humano, diferente cada vez que se enfrenta a un problema. De esta forma, se utiliza un flujo continuo de datos en el que siempre se incorporarán nuevos ejemplos, en lugar de un *dataset* finito. Este enfoque es inviable en cualquier aspecto desde un punto de vista supervisado, al requerir de un etiquetado permanente de nuevos datos. Sin embargo, existen técnicas no supervisadas que no tienen esta limitación e incluso explotan esta forma de trabajo.

## 2.2 Aprendizaje automático en vídeo

El vídeo es una señal de gran riqueza, superior a la imagen, proporcionando información adicional de forma exclusiva debido a la componente temporal. Este tipo de señal proporciona información sobre la correspondencia de los objetos, al mantener la continuidad propia del mundo real. Además, el tiempo determina cuándo suceden los eventos de un vídeo, introduciendo una relación de orden. Las técnicas de inteligencia artificial deben tener en cuenta estas características particulares para construir sistemas en este dominio.

En los inicios del aprendizaje automático sobre vídeo, se realizaba un tratamiento del mismo como un volumen tridimensional (XYT), donde el tiempo se convertía simplemente en una dimensión adicional. Esta perspectiva permitía explotar los patrones que se generaban en el volumen o aplicar operadores matemáticos de carácter espacio-temporal para el reconocimiento de acciones. Se trata de una idea que se sigue aplicando en la actualidad, siendo la base de las convoluciones tridimensionales.

Sin embargo, los volúmenes tridimensionales presentan diferentes problemas. En primer lugar, el tiempo no puede entenderse como una simple dimensión adicional. El muestreo temporal es muy diferente al espacial, ya que se realiza de forma más gruesa, con un número de muestras inferior. En segundo lugar, estos volúmenes consiguen buenos resultados cuando se mantiene una correspondencia implícita entre los objetos presentes, por ejemplo, cuando se ha utilizado una cámara en posición fija. En el mundo real los objetos e individuos no mantienen una posición estacionaria, sino que se mueven y, en muchos casos, de forma rápida y siguiendo complicadas trayectorias.

Estos problemas conducen a trabajar identificando de forma directa y explícita los píxeles y objetos entre los que existe correspondencia. De esta forma, se encuentran desarrollos supervisados que utilizan el flujo óptico [14] y otros que se basan en el seguimiento de objetos [15]. El flujo óptico se basa en correspondencias locales, píxel a píxel, mientras que el seguimiento de objetos puede basarse en su detección en cada *frame*, consiguiendo resultados de mayor nivel de abstracción.

Por otro lado, el vídeo presenta un problema adicional en tareas específicas como el reconocimiento de acciones. Un algoritmo con buenas prestaciones puede utilizar únicamente 1 *frame* del vídeo para realizar la tarea, de forma que no se está resolviendo realmente el problema.

En la actualidad, se busca conseguir modelos que no requieran supervisión, que trabajen a nivel de píxel y que mantengan las relaciones contextuales y de orden propias de la componente temporal. La solución es utilizar el tiempo como supervisor, es decir, construir modelos no supervisados que aprovechen las características temporales de los vídeos como sus propias etiquetas. La componente temporal también obliga a utilizar todos los *frames* en la resolución de la tarea, asegurando, en cierta medida, que el modelo está teniendo en cuenta todo el vídeo. En este sentido, existen diferentes trabajos realizados, como la predicción de píxeles de los fotogramas siguientes [16], del color a lo largo del tiempo [17] o del sentido temporal del vídeo [18] [19].

En este trabajo, se construyen modelos capaces de aprender características de vídeos de forma no supervisada. En concreto, se utilizan modelos que explotan el carácter secuencial de un vídeo y sus restricciones temporales, como las relaciones de orden o la consistencia de ciclo, para conseguir un seguimiento explícito de píxeles y objetos mediante medidas de similitud. La mayor parte de trabajos anteriores relativos al tratamiento de vídeo de forma no supervisada se centran en la búsqueda de correspondencias a nivel de imagen sin tener en cuenta la componente temporal. De esta forma, tanto el enfoque como la estrategia utilizadas en este proyecto son innovadores dentro del aprendizaje no supervisado de vídeo.

# 3 Aprendizaje no supervisado

El aprendizaje supervisado ha permitido conseguir importantes avances en diferentes dominios, pero con una gran dependencia de datos etiquetados. En escenarios con falta de etiquetas, las estrategias supervisadas proporcionan modelos con una generalización muy pobre. Una posible solución a este problema es aplicar técnicas de *transfer learning*, es decir, utilizar una red neuronal de gran tamaño preentrenada con una extensa base de datos etiquetados y realizar un entrenamiento pequeño o  que particularice el modelo para otra tarea más específica [20]. Sin embargo, estas técnicas están limitadas a tareas para las que existan desarrollos previos y resultan poco útiles en aplicaciones muy concretas. No obstante, la cantidad de datos generados en todo el mundo crece anualmente de forma exponencial, por lo que el verdadero problema radica en cómo tratarlos. Por tanto, la alternativa real a la inversión en etiquetado consiste en aprovechar estos datos en bruto, incorporándolos al entrenamiento de forma que permitan alcanzar resultados similares o superiores con menos etiquetas. Los métodos de aprendizaje no supervisado se basan en esta idea.

El aprendizaje no supervisado engloba un conjunto de técnicas que persiguen explotar la disponibilidad de datos masivos no etiquetados [21]. Inicialmente, se trataba de procedimientos de *clustering*, encargados del agrupamiento o la creación automática de grupos a partir de los datos sin ningún tratamiento previo. La investigación ha evolucionado desarrollando diferentes técnicas o enfoques particulares, como *semi-supervised learning* o *self-supervised learning* (generación automática de etiquetas). Estas estrategias tienen características específicas, manteniendo los datos no etiquetados como principio básico.

El paradigma del aprendizaje no supervisado es distinto al caso supervisado, de forma que no se trata de construir modelos limitados a resolver una tarea aislada y específica, sino de obtener resultados complementarios que sean válidos para diferentes problemas más pequeños. La idea reside en la capacidad de los seres humanos para aprender ciertos mecanismos o información previa directamente del entorno y que permiten adaptar la forma de conocimiento o aprendizaje para nuevas tareas [22].

En la actualidad, una de las líneas de trabajo más importantes del aprendizaje no supervisado se realiza en el ámbito del *representation learning* [23]. En este campo se engloban diferentes técnicas que tienen por objetivo obtener representaciones (*embeddings*) potentes y generalizables a partir de un amplio conjunto de datos no etiquetados. De esta forma, el modelo conseguido constituye un extractor de características robusto aplicable a otras tareas en las que el número de datos etiquetados es muy reducido. En el ámbito de datos multimedia existen trabajos previos que desarrollan esta idea, obteniendo buenos extractores de características, por ejemplo, para audio [4] o vídeo [24].

## 3.1 *Contrastive Predictive Coding* (CPC)

El aprendizaje no supervisado utiliza diferentes estrategias para el entrenamiento de los modelos. Una de las más comunes es el caso del *self-supervised learning*, que consiste en predecir información futura a partir de cierta información pasada y contextual. Se trata de una idea muy similar a la codificación predictiva utilizada en compresión de datos y permite obtener importantes resultados desde el punto de vista del *representation learning*.

El objetivo de esta estrategia no está en conseguir un buen predictor sino en construir un extractor de características potente que permita realizar buenas predicciones posteriormente, con mayor facilidad. Este enfoque se ha aplicado, por ejemplo, en tareas de procesado de lenguaje natural (predicción de palabras a partir de vecinas) o en el campo de imagen (predicción de un bloque de una imagen a partir del resto).

El éxito en la resolución de estas tareas de predicción supone que el contexto proporciona información útil para conseguir una representación de los datos de alto nivel, con mayor abstracción. De esta forma, es posible obtener una codificación latente que recoja la información compartida entre las distintas secciones de una señal, descartando información de bajo nivel y ruido. Este tipo de técnicas tienen una aplicación clara en problemas que trabajan datos en secuencia, como por ejemplo, señales de audio y vídeo.

En este proyecto, los modelos propuestos están basados en el trabajado realizado en [1], centrado en ámbitos de audio e imagen, por lo que se ha llevado a cabo su adaptación al tratamiento de vídeo. La técnica propuesta se denomina *Contrastive Predictive Coding* (CPC) y consiste en el entrenamiento de los modelos mediante la comparación entre las predicciones de muestras futuras y un conjunto de posibles muestras reales entre las que se encuentra la realmente correcta o *ground truth*.

El CPC se basa en potenciar la información mutua entre la información contextual y las predicciones realizadas. En concreto, se busca maximizar la información mutua media entre ambas, que se expresa como:

$$I(x;c) = \sum_{x,c} p(x,c) \cdot \log \frac{p(x|c)}{p(x)} \qquad \text{(Ec. 3.1)}$$

La información mutua es una medida de dependencia entre dos variables, en este caso, la señal $x$ y el contexto $c$, indicando el grado de información que comparten dos variables. La particularidad del CPC está en que busca maximizar la información mutua de la señal a predecir y el contexto pero en un espacio de representación de mayor nivel, consiguiendo una codificación latente en la que se explote lo que haya en común entre dichas variables.

La Figura 3.1 muestra la arquitectura del modelo CPC y el funcionamiento del aprendizaje constrativo en una aplicación de vídeo, basada en la predicción del *frame* inmediatamente posterior al actual (instante t) a partir de los anteriores. En primer lugar, se define un conjunto de muestras $X = \{x_1, \dots, x_N\}$, formado por 1 muestra positiva *(ground truth)* a predecir y N-1 muestras negativas, del que se obtienen sus respectivos *embeddings*. En el caso de aplicación de vídeo, el conjunto está formado por el *embedding* real del *frame* en el instante de predicción del vídeo de entrada y N-1 *embeddings* de *frames* de otros ejemplos en dicho instante.

La arquitectura se compone de diferentes bloques. El primer bloque, denominado *encoder* (codificador), se utiliza para obtener la representación, *embedding* o codificación latente ($z_i$) de cada uno de los *frames* del vídeo de entrada. Este *encoder* puede construirse según diferentes modelos o arquitecturas, flexibilizando enormemente el modelo completo.

Posteriormente, se utiliza otra red para resumir las representaciones resultantes de todos los fotogramas previos al instante actual en una representación o *embedding* de contexto, $c_t$. Esta red debe ser apropiada para trabajar con datos en secuencia, por lo que se suele utilizar una red recurrente (RNN). En concreto, se ha escogido una *Gated Recurrent Unit* (GRU).

Finalmente, las salidas de la RNN se utilizan para predecir la representación del *frame* futuro, $p_{t+1}$. Como se puede comprobar, no es una predicción a nivel de señal, sino a nivel de codificación latente.

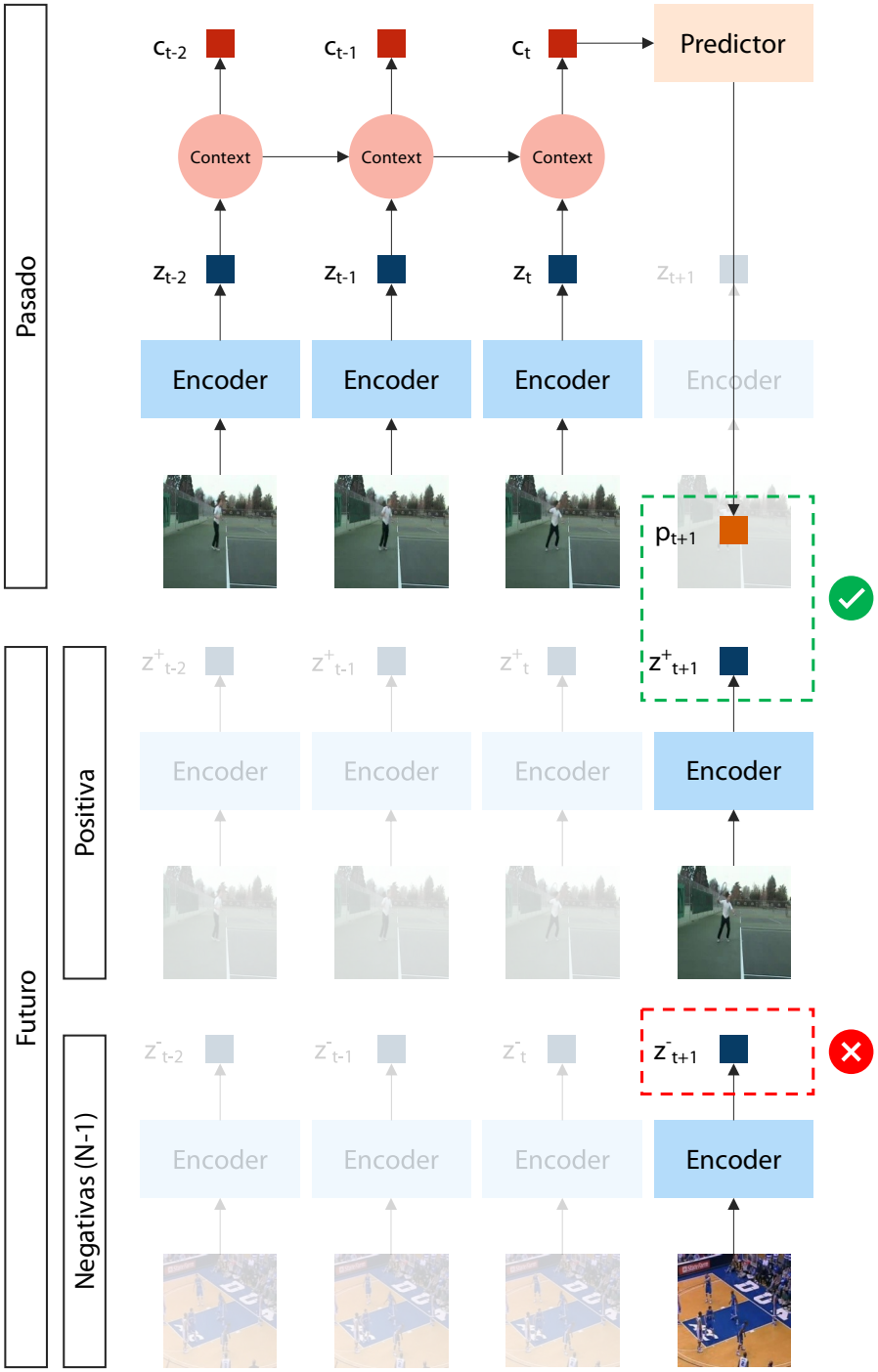


**Figura 3.1.** Arquitectura y funcionamiento de modelo CPC para vídeo

La predicción obtenida, $p_{t+1}$, se compara mediante producto escalar con la representación del fotograma real en el instante t+1, $z_{t+1}^{+}$ (muestra positiva), y las representaciones correspondientes a un conjunto de fotogramas de otros vídeos en ese mismo instante, $z_{t+1}^{-}$, (muestras negativas). El aprendizaje contrastivo radica en que los modelos entrenados (*encoder* y red recurrente) deben generar representaciones o *embeddings* suficientemente potentes para maximizar el parecido de la predicción con la muestra positiva y minimizarlo en el caso de cada una de las muestras negativas. Por otro lado, aunque en la Figura 3.1 sólo se indica la predicción de 1 *frame*, se pueden predecir varios fotogramas e incluso no consecutivos (t+k).

La explicación anterior cuenta con una base matemática clara que sirve de complemento para su comprensión. Si se define t como el instante o fotograma actual, su correspondiente *embedding* y la información de contexto se obtienen según las siguientes expresiones:

$$z_t = g_{Encoder}(x_t)$$
$$c_t = g_{RNN}(z_i), \qquad i \leq t \tag{Ec. 3.2}$$

Una vez obtenidas dichas variables, se debe realizar la predicción de uno o varios *frames* siguientes, en el caso de la Figura 3.1, el fotograma inmediatamente posterior. Esta predicción se puede realizar de diversas formas y se modela mediante una función concreta. El CPC se caracteriza por modelar una función de densidad proporcional a la información mutua, capaz de preservarla. En [1] se propone el uso de un modelo log-bilineal, donde el predictor consiste simplemente en una capa lineal aplicada sobre la información de contexto $c_t$.

$$f_k(x_{t+k}, c_t) = e^{g_{Encoder}(x_{t+k})\cdot p_{t+k}} = e^{z_{t+k}\cdot p_{t+k}} = e^{z_{t+k}\cdot W_k c_t} \tag{Ec. 3.3}$$

La principal característica de la función escogida radica en que no se está realizando la predicción del *frame* en la posición k, $x_{t+k}$, sino que dicha predicción, $p_{t+k}$, debe corresponderse con la representación o codificación latente de dicho *frame*, $z_{t+k}$. De esta forma, la comparación entre predicciones y *embeddings* positivos y negativos se realizará a nivel de *encoder*, como se ha indicado con anterioridad.

El aprendizaje no supervisado se basa también en la optimización de una función de coste o de pérdidas, como en el caso supervisado. La función de coste utilizada en esta técnica es de tipo contrastivo y se basa en *Noise Cross-Entropy* (NCE):

$$\mathcal{L}_N = -\mathbb{E}_X\left[\log\frac{f_k(x_{t+k}, c_t)}{\sum_{x_j\in X} f_k(x_j, c_t)}\right] = -\mathbb{E}_X\left[\log\frac{e^{z_{t+k}\cdot W_k c_t}}{\sum_{x_j\in X} e^{z_j\cdot W_k c_t}}\right] \tag{Ec. 3.4}$$

Esta función (Ec. 3.4) requiere de etiquetas para poder computar las pérdidas y determinar si el modelo se está aproximando o no a la resolución del problema. Sin embargo, el CPC es un modelo de aprendizaje no supervisado que se puede clasificar dentro del grupo de técnicas *self-supervised* o de autoaprendizaje. Por lo tanto, en este tipo de modelos, las etiquetas son los *embeddings* de las muestras positivas a los que deben parecerse las predicciones, por lo que se generan de forma automática a partir de los datos de entrada sin etiquetar.

La función de coste NCE (Ec. 3.4) es similar a las funciones utilizadas en aprendizaje supervisado para tareas de clasificación (*cross-entropy*), donde se maximiza la salida de la red para la clase correcta al mismo tiempo que se minimiza el resto (salidas incorrectas). La diferencia, en este caso, es que las etiquetas son las representaciones de los *frames* siguientes conocidos, $z_{t+k}$. Por tanto, la tarea no se centra únicamente en maximizar el parecido de la predicción con el *embedding* real, sino que debe parecerse a dicho *embedding* (muestra positiva) al mismo tiempo que debe diferenciarse con la mayor claridad posible de un conjunto *embeddings* incorrectos (muestras negativas) correspondientes a representaciones de *frames* de otros vídeos.

La técnica CPC se puede entender como una tarea de clasificación a nivel de *embeddings*, obligando al *encoder* a generar representaciones potentes que permitan la discriminación entre muestra positiva y muestras negativas. Por otro lado, es una técnica adecuada para datos en secuencia, como es el caso de los *frames* de vídeo utilizados en este proyecto.

## 3.2 Evaluación de técnicas no supervisadas

La mayoría de modelos supervisados se diseñan para una aplicación concreta, de forma que resulta evidente cómo evaluarlos. Se trata de determinar la capacidad real del modelo en términos de resolución de la tarea y de generalización, es decir, de independencia con respecto a nuevos datos no conocidos por la red neuronal.

El aprendizaje no supervisado no tiene una evaluación tan clara debido a los objetivos con los que se diseñan los modelos. Los modelos no supervisados no se crean con un carácter finalista sino que se desarrollan, en muchos casos, con un carácter complementario. El objetivo no es conseguir resolver una tarea específica sino conseguir resultados o modelos que puedan ser aplicados a otras tareas que sí serán finalistas. En este caso, la técnica utilizada (CPC) tiene por objetivo conseguir un potente extractor de características a partir de muchos datos en bruto, de forma que sea útil como modelo preentrenado y complementario en la resolución de otras tareas más específicas, denominadas secundarias, como la clasificación supervisada de un *dataset* reducido o la búsqueda de ejemplos similares a uno dado a partir del parecido de sus vectores de características. Por tanto, la evaluación real de la capacidad de esta técnica radica en la versatilidad y potencia que aporta en la resolución de dichas tareas secundarias.

# 4 Arquitecturas

En este capítulo se describen las diferentes arquitecturas de red neuronal utilizadas en los modelos desarrollados a lo largo del proyecto. La elección de arquitecturas en cualquier problema de aprendizaje automático depende del dominio al que pertenecen los datos utilizados, en este caso, datos de vídeo, y de la tarea a resolver. El vídeo se define como una secuencia temporal de imágenes o *frames*, de forma que las arquitecturas escogidas deben combinar tratamientos a nivel de imagen y a nivel temporal.

La técnica no supervisada presentada anteriormente, CPC, se corresponde, en cierta medida, con una tarea de clasificación. Las tareas secundarias comprueban su potencia como extractor de características, por lo que interesan arquitecturas que transformen, de forma progresiva, un conjunto de *frames* en un vector o conjunto de vectores de características *(embeddings)*. En [25] se realiza un estudio de los desarrollos en tareas de reconocimiento de acciones, mostrando posibles arquitecturas como redes neuronales convolucionales (CNN) bidimensionales y tridimensionales, redes que combinan imagen y flujo óptico o redes recurrentes que aprovechan el carácter temporal del vídeo.

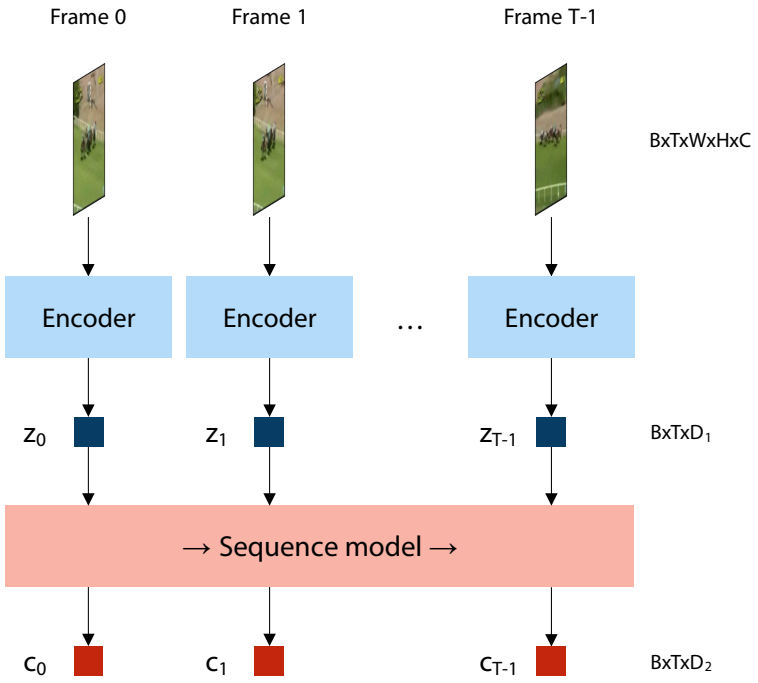


**Figura 4.1.** Esquema genérico de los modelos desarrollados para vídeos de T *frames*

En este proyecto, los modelos desarrollados comparten una arquitectura genérica (Figura 4.1), caracterizada por realizar un tratamiento inicial a nivel de fotograma *(encoder)* y, posteriormente, a nivel de secuencia temporal *(sequence model)*. La entrada es un tensor de dimensión BxTxWxHxC, siendo B el tamaño del *batch*, T la longitud de la secuencia, W la anchura de cada *frame*, H su altura y C el número de canales. Este tensor se procesa generando vectores de características de dimensiones $D_1$ y $D_2$ a la salida del *encoder* y del *sequence model*, respectivamente. Se trata de una arquitectura compatible con la presentada para el CPC en el capítulo anterior.

## 4.1 *Encoder*

El *encoder* o codificador es el primer bloque de la arquitectura genérica presentada y se encarga del procesado individual de cada uno de los *frames* del vídeo. En este sentido, la primera etapa se puede considerar como un tratamiento de imagen, es decir, sin considerar la componente temporal del vídeo. Los desarrollos realizados en el campo de imagen son muy amplios y diversos, por lo que se han probado diferentes implementaciones.

### 4.1.1 2D *Convolutional Neural Networks* (CNNs)

Las redes neuronales convolucionales (CNNs) bidimensionales, a pesar de su antigüedad [26], han sido los modelos más relevantes en la última década dentro del procesado de imagen mediante inteligencia artificial. Estos modelos tienen una inspiración biológica basada en el funcionamiento de la corteza visual humana. Las neuronas presentes en la corteza tienen campos receptivos locales, es decir, reaccionan a una región limitada del campo visual. Posteriormente, los campos receptivos de muchas neuronas a nivel local se solapan y combinan para conseguir la imagen completa.

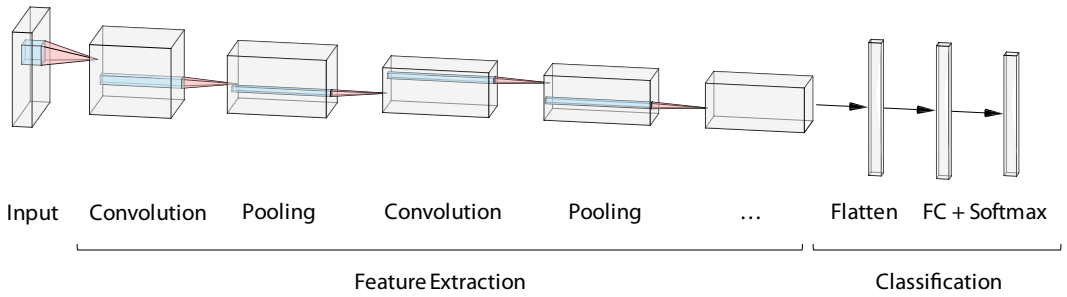


**Figura 4.2.** Esquema ejemplo red neuronal convolucional [27]

Las CNNs bidimensionales tratan de imitar estas características mediante un procesado progresivo de la imagen siguiendo una jerarquía de abstracción en la que se aplican operaciones a nivel local al mismo tiempo que se combina y compacta la información. El resultado de estas redes es la transformación de una imagen como matriz en un vector de características o *embeddings* que resume su contenido.

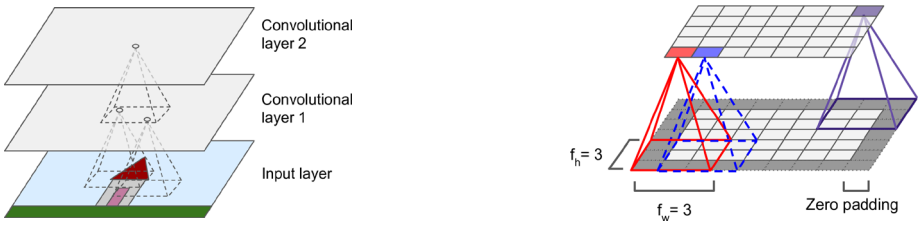


**Figura 4.3.** Funcionamiento capa convolucional bidimensional [28]

El elemento básico de las CNNs son las capas convolucionales bidimensionales. Estas capas consisten en filtros aplicados como ventana deslizante sobre una matriz, obteniendo una nueva matriz generalmente de menor tamaño. Las neuronas de la capa convolucional no están conectadas con todos los píxeles de la imagen, sino únicamente con una sección de la misma, imitando el funcionamiento de los campos receptivos locales humanos.

Las CNNs se componen de sucesivas capas convolucionales, de forma que los resultados de los campos receptivos se combinan progresivamente hasta resumir el contenido de la imagen en un vector de características. Esta estructura jerárquica permite que la red procese en las capas superiores características de bajo nivel de abstracción y, posteriormente, las combine para conseguir características de alto nivel. Los vectores de características obtenidos se pueden procesar mediante capas lineales estándar para resolver un problema concreto, como la clasificación de imágenes. En este proyecto las CNNs se utilizan como extractores de características de cada uno de los *frames* de un vídeo, obteniendo sus *embeddings* correspondientes.

### 4.1.2 *Residual Networks* (ResNet)

Las CNNs típicas están formadas por grupos de capas convolucionales junto con una capa de *pooling* que se repiten a lo largo de la arquitectura (Figura 4.2). La imagen se reduce en tamaño a medida que progresa en la red, al mismo tiempo que aumenta en profundidad, es decir, se incrementa el número de canales o *features* que sirven de características.

Esta arquitectura genérica ha tenido distintas modificaciones, desarrollando variantes que han permitido conseguir importantes progresos en el campo de imagen. En este proyecto, se han utilizado las redes residuales (ResNet) [29], ganadoras de la competición en 2015. La principal novedad de su arquitectura consiste en realizar saltos de conexiones en la red, es decir, la señal de entrada a una capa se suma a su salida o a la salida de capas de mayor profundidad.

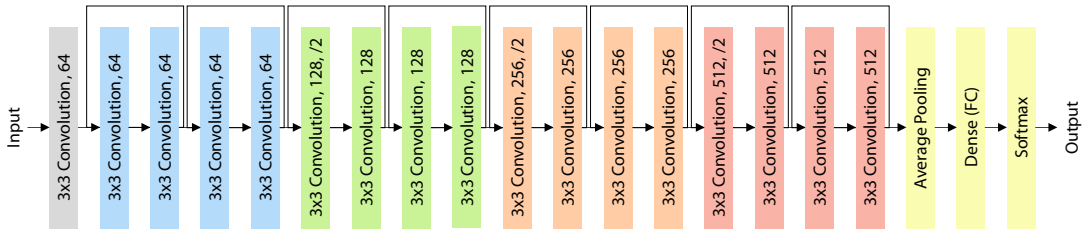


**Figura 4.4.** Arquitectura ResNet-18

La red residual utilizada en el trabajo es la ResNet-18, con una arquitectura de 18 capas (Figura 4.4). Las capas convolucionales utilizadas incrementan el número de canales desde 64 hasta 512, con progresivos diezmados en un factor 2. El salto de conexiones se realiza cada 2 capas, generando caminos residuales que se vuelven a incorporar a la red.

La arquitectura profunda de las redes residuales se puede interpretar como una pila de unidades residuales, donde cada una de ellas se convierte en una pequeña subred neuronal separada del resto por su correspondiente camino residual. Esta característica permite un entrenamiento más rápido de la red completa, al propagarse fácilmente la señal por la red incluso con capas intermedias poco entrenadas y cercanas a su estado inicial.

### 4.1.3 Modelos de atención. *Transformer. Vision Transformer* (ViT)

Las arquitecturas más novedosas y de mayores prestaciones utilizadas en este proyecto están basadas en modelos de atención. En particular, se han implementado modelos conocidos como *transformers* [30]. Se trata de modelos originalmente dirigidos al tratamiento de secuencias de texto que se han rediseñado para su adaptación a diferentes disciplinas. En este proyecto, se hace uso de su aplicación al campo de imagen, el *Vision Transformer* (ViT) [31]. La comprensión de la arquitectura del ViT requiere el conocimiento de conceptos básicos relativos a la atención y a los *transformers* en general.

#### 4.1.3.1 *Transformers. Self-Attention*

Los *transformers* se han convertido en los modelos dominantes del *deep learning* en la actualidad y en la mayor parte de campos. Su potencia reside en una operación básica denominada autoatención o *self-attention* que trata, fundamentalmente, del cálculo del producto escalar entre dos secuencias de vectores de características.

La entrada del *transformer* es una secuencia de vectores de características, formando una matriz. Estos vectores se procesan para generar dos representaciones ligeramente distintas de la misma matriz y se realiza el producto entre ellas. El resultado obtenido se corresponde con el producto escalar o la similitud coseno entre los diferentes vectores de características de la secuencia, es decir, una matriz que refleja el grado de parecido entre ellos. La autoatención incluye un paso adicional consistente en aplicar la operación *softmax* a cada fila de la matriz, de forma que sus elementos suman 1, algo necesario para acotar el parecido.

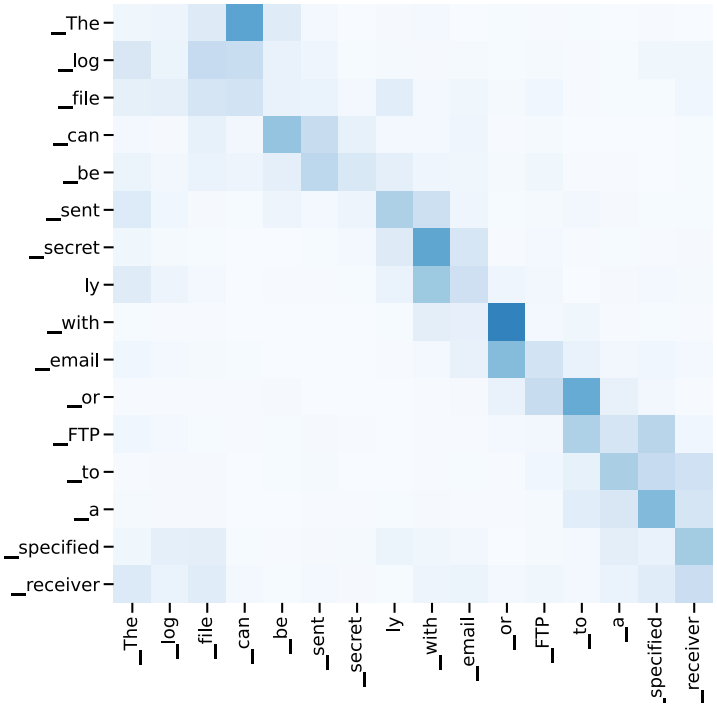


**Figura 4.5.** Ejemplo matriz de autoatención [32]

El campo de aplicación original de los *transformers* es el procesado del lenguaje natural (NLP), por lo que resulta interesante explicar sus fundamentos utilizando ejemplos de este ámbito [32]. La matriz de atención generada se puede representar gráficamente (Figura 4.5), donde los colores más azules indican un mayor parecido, es decir, que los vectores de características están alineados. En el caso concreto de NLP, se interpreta como una matriz que indica qué palabras tienen relación entre sí dentro de una oración o un párrafo. Esta relación puede ser sintáctica, semántica o de otro tipo, proporcionando una potencia enorme para resolver una tarea concreta.

Esta matriz de atención se puede multiplicar por un vector de características, generando una salida que tiene en cuenta las relaciones existentes entre las palabras del texto. La operación se denomina autoatención porque la matriz se ha obtenido utilizando únicamente la entrada al modelo.

La ventaja de los *transformers* radica en que utiliza múltiples matrices de autoatención en paralelo, de forma que cada una de ellas se ocupa de analizar un tipo de relaciones entre las palabras y, posteriormente, se combina su procesado. Por otra parte, el modelo obtenido no es de tipo caja negra, sino que es posible entender a posteriori, visualizando las matrices de atención, qué ha aprendido el modelo para resolver un problema, es decir, las relaciones que ha establecido entre palabras. Esta característica también se puede explotar en su aplicación al campo de imagen, el ViT, generando máscaras que resaltan los píxeles en los que el modelo centra su atención, algo que se mostrará mediante ejemplos en los resultados del capítulo siguiente.

El *transformer* es uno de los modelos más exitosos en la actualidad para el procesado de secuencias. Su funcionamiento está basado en la operación de autoatención explicada, aunque su arquitectura cuenta con otros elementos que también requieren de explicación.

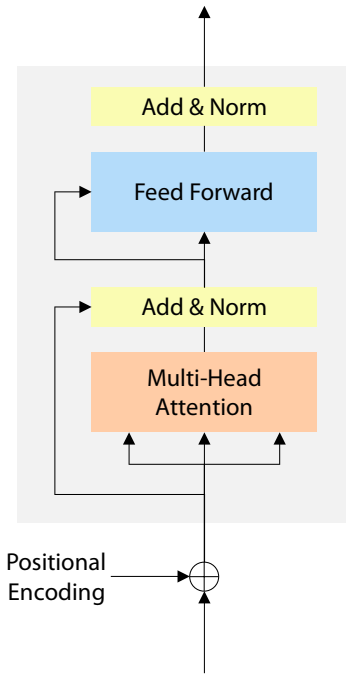


**Figura 4.6.** Arquitectura de *transformer (encoder)*

La arquitectura del *transformer* está formada por 2 partes diferenciadas, un *encoder* que se ocupa de extraer una representación de los datos y un *decoder* que permite obtener una predicción o la generación de una salida, como por ejemplo, un texto. En este trabajo se han utilizado, a través del ViT, *transformers* como extractores útiles de características, por lo que se ha representado únicamente la arquitectura del *encoder* (Figura 4.6).

El modelo es muy sencillo, compuesto por 4 bloques o capas diferentes:

- *Multi-Head Attention:* Es el núcleo del *transformer*, formado por varios bloques de autoatención en paralelo, denominados cabezales o *heads*. Este bloque permite una flexibilidad enorme y tiene la capacidad de aprender subtareas diferentes en paralelo, encargándose de encontrar relaciones distintas dentro de los mismos datos de entrada. Los cabezales son capaces de relacionar conceptos distintos en paralelo, pero estas relaciones sólo tienen sentido y utilidad si se combinan posteriormente. En la Figura 4.7 se muestran las matrices de autoatención de distintos *heads* en paralelo para una aplicación concreta de NLP [32].
- *Positional Encoding:* La codificación o *embedding* posicional es la segunda novedad introducida en estos modelos. Se trata de una capa que suma a la entrada información relacionada con la posición absoluta y relativa de los ítems que hay en la secuencia. El objetivo es conseguir que el modelo tenga en cuenta, además del grado de parecido entre los elementos de la secuencia, su posición en la misma.
- *Add & Norm:* En esta capa se añade un camino residual y se aplica un tipo de normalización específica de mayor estabilidad frente a otras más extendidas, como *batch normalization*.
- *Feed Forward:* Se trata de una red de tipo perceptrón multicapa (MLP) simple encargada de unificar las diferentes versiones en paralelo de los datos procedentes de cada cabezal y volver a generar una salida conjunta.

Los bloques se repiten progresivamente según la tarea, generando modelos de distinto tamaño. La arquitectura final proporciona resultados muy competitivos a pesar de su sencillez.

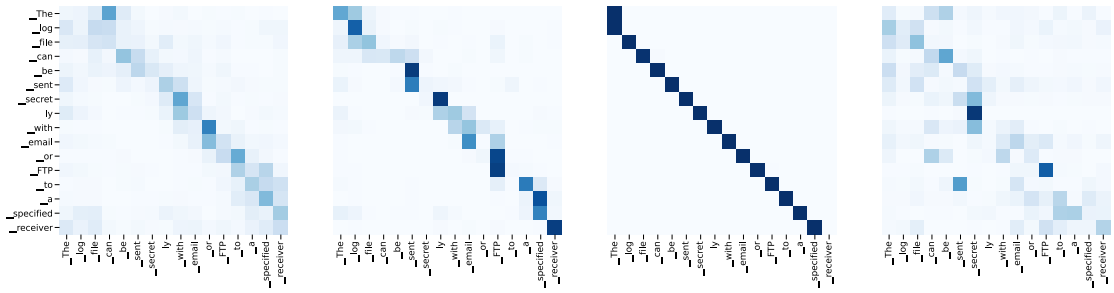


**Figura 4.7.** Ejemplo *multi-head attention* [32]

En este proyecto, los *transformers* se han utilizado para el procesado a nivel de *frame*. No obstante, también se podrían utilizar como *sequence models* dentro de la arquitectura presentada en la Figura 4.1, encargándose del tratamiento de la secuencia a nivel temporal.

#### 4.1.3.2 *Vision Transformer* (ViT)

La arquitectura *transformer* se ha convertido en el estándar actual de las tareas de procesado de lenguaje natural. Sin embargo, su aplicación en otros ámbitos, como la visión por computador, es una de las líneas de investigación más recientes. Los modelos de mayores prestaciones en la última década dentro del campo de visión por computador se han basado en CNNs y determinados aspectos de atención. El *Vision Transformer* (ViT) ha supuesto una revolución en este campo, superando los mejores resultados utilizando arquitecturas completamente distintas.

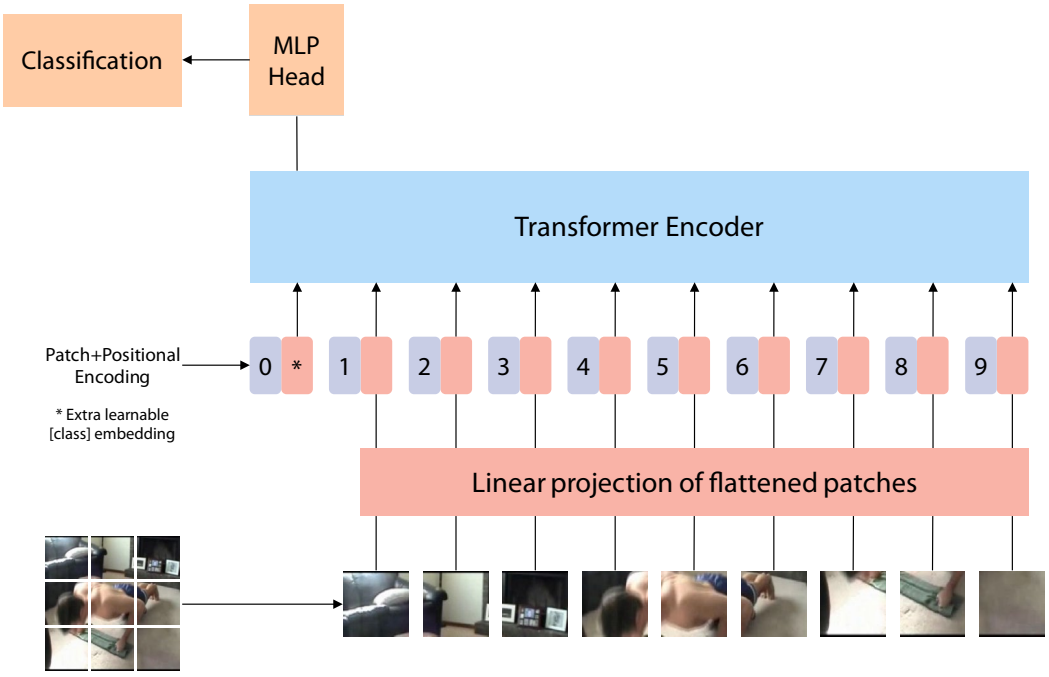


**Figura 4.8.** Arquitectura *Vision Transformer* (ViT)

La idea fundamental del ViT consiste en aplicar un *transformer* de forma directa sobre una secuencia de bloques (*patches*) de una imagen. La Figura 4.8 muestra la arquitectura del ViT para una imagen dividida en 9 *patches*, como por ejemplo, imágenes de 180x180 píxeles en bloques de 60x60 píxeles.

El primer paso es dividir la imagen en bloques y realizar una proyección lineal de los mismos para obtener una secuencia de vectores de características. De esta forma, se consigue una entrada compatible con la arquitectura original del *transformer* explicada anteriormente. Posteriormente, se añade un *embedding* adicional a la secuencia obtenida, que servirá de vector de características resumen de la imagen (*token* de clase). También se incorpora su *encoding* posicional y, finalmente, se aplica el *encoder* del *transformer* a la nueva secuencia.

Una vez aplicado el *encoder*, el primer vector de características de la secuencia de salida se corresponde con el *token* de clase. Se trata de un *embedding* global que representa a la imagen y que puede ser utilizado para una tarea de clasificación de imagen aplicando una capa lineal previa (MLP) o como vector de características útil para otras tareas. En este proyecto, se utiliza el ViT como extractor de características de cada uno de los *frames* de un vídeo que, posteriormente, se procesan teniendo en cuenta la componente temporal en un modelo de secuencia puro.

## 4.2 *Sequence model*

El segundo bloque de la arquitectura genérica presentada en la Figura 4.1 es el modelado de secuencia *(sequence model)*. La secuencia temporal de fotogramas se ha transformado en una secuencia de *embeddings* mediante las arquitecturas del apartado anterior (Figura 4.9). El resultado se obtiene realizando un tratamiento individual a nivel de *frame*. El siguiente paso consiste en procesar los *embeddings* utilizando las relaciones temporales y de orden presentes, es decir, incorporar el tratamiento del vídeo como secuencia temporal.

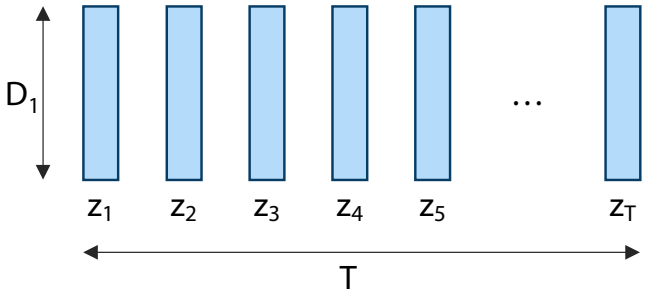


**Figura 4.9.** Secuencia de *embeddings* correspondientes a *frames* de vídeo

Los *sequence models* procesan este tipo de señales, representadas como una secuencia de vectores de características entre los que existe una relación de orden temporal. Estos modelos son capaces de explotar las características de la secuencia, generando salidas sin tener en cuenta únicamente los vectores de forma aislada.

Los datos de entrada se disponen en un tensor de dimensiones $BxTxD_1$ (Figura 4.9), siendo B el tamaño del *batch*, T la longitud de la secuencia y $D_1$ la dimensión de los *embeddings*. Los modelos pueden seguir diferentes tipologías, obteniendo una salida global por cada secuencia o una salida múltiple, es decir, una nueva secuencia de vectores de características en un nuevo espacio que utiliza, a priori, el carácter temporal y de orden de la entrada.

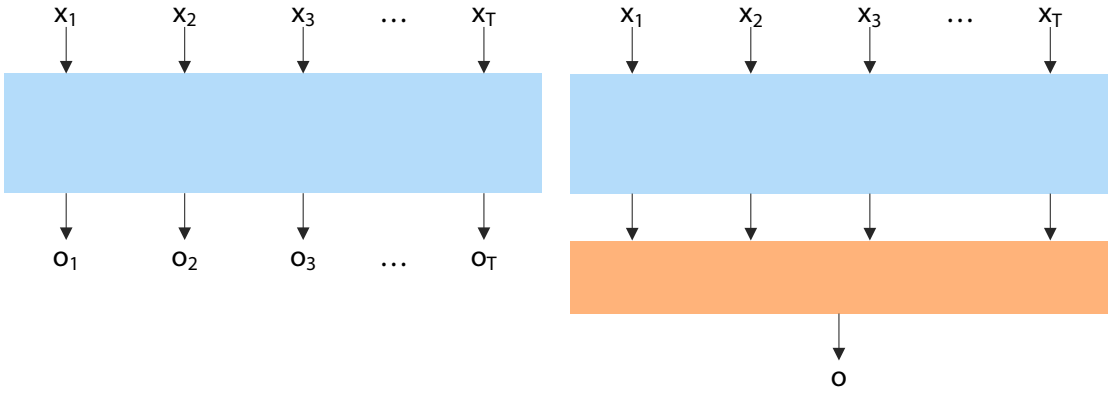


**Figura 4.10.** *Sequence models* con salida múltiple (izquierda) y global (derecha)

En los modelos de salida múltiple, la secuencia se puede procesar posteriormente para obtener una salida global según la tarea, como por ejemplo, una clasificación de vídeo. En este proyecto se han utilizado modelos secuenciales de salida múltiple con la misma longitud que la entrada, en concreto, redes neuronales recurrentes, introducidas a continuación.

### 4.2.1 *Recurrent Neural Networks* (RNN)

Los *sequence models* utilizados en el proyecto han sido las redes neuronales recurrentes (RNNs). Las RNNs son modelos que realizan un procesamiento sucesivo de los instantes temporales o elementos presentes en una secuencia. De esta forma, cuando se procesa una entrada, se obtiene su salida junto con una información adicional de secuencia que se utiliza en el cálculo de la salida de la siguiente entrada.

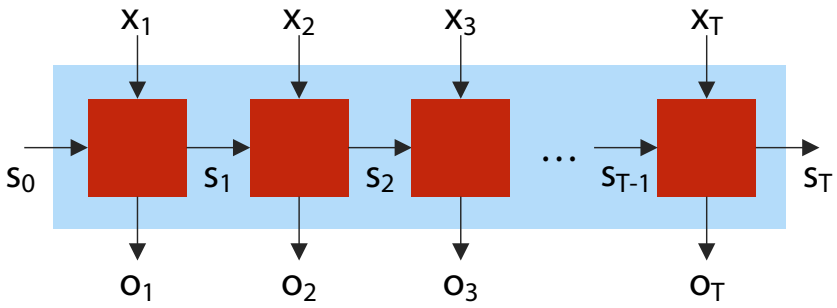


**Figura 4.11.** Esquema general redes neuronales recurrentes (RNNs)

La Figura 4.11 muestra la estructura funcional de una red neuronal recurrente. El modelo se compone realmente de un único bloque de procesado que se aplica de forma sucesiva. Este bloque dispone de una entrada, $x_i$, y una información de estado previa, $s_{i-1}$, para generar su salida, $o_i$, y la siguiente información de estado, $s_i$. El bloque de procesado puede contar con diferentes arquitecturas, de forma que existen distintas implementaciones de este tipo de modelos.

La representación escogida en la figura pretende resaltar que el procesamiento de cada entrada se realiza con una información de secuencia o estado distinta. Por tanto, las operaciones deben seguir un orden, sin posibilidad de paralelizar. El camino de información de secuencia, $s_0, s_1, \dots, s_{T-1}$, es el que impone la recurrencia.

La información que se transmite de una etapa a la siguiente se denomina estado. Se trata de una información de secuencia que define la situación de memoria que interesa conservar para el procesamiento de las siguientes etapas. Esta información queda codificada y almacenada en el vector de estado, $s_i$. La cantidad de información que se mantiene entre etapas depende de la tarea, de forma que el tamaño de dicho vector se puede configurar.

#### 4.2.1.1 *Long Short-Term Memory* (LSTM)

La primera implementación utilizada en el proyecto son las redes *Long Short-Term Memory* (LSTM), propuestas en 1997 [33]. La idea de este tipo de arquitectura, como se puede extraer del nombre, es que tenga en cuenta tanto eventos cercanos como lejano. Se trata de uno de los modelos más utilizados en *deep learning* en los últimos años, resultando ganadora en cualquier tarea que requiriese modelos de secuencia a pesar de su antigüedad. En la actualidad, existen modelos ligeramente superiores en prestaciones. Sin embargo, se trata de una arquitectura muy competitiva y exitosa en escenarios donde la cantidad de datos de entrenamiento disponible es reducida, como es el caso de este trabajo.

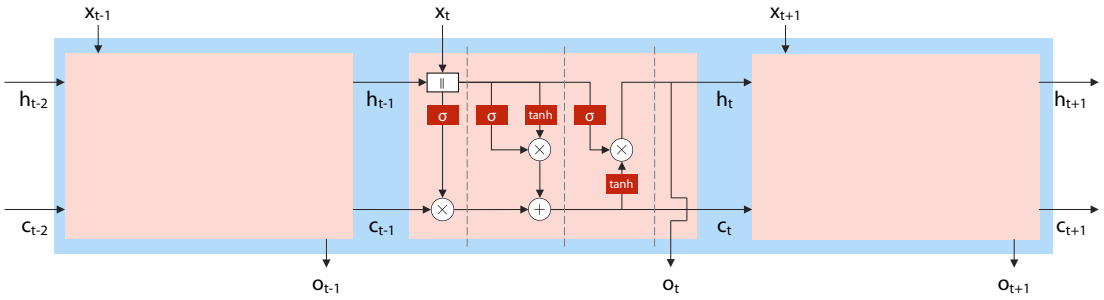


**Figura 4.12.** Esquema funcional red LSTM con detalle bloque procesado (instante t)

El esquema de funcionamiento de las redes LSTM (Figura 4.12) es muy similar a la estructura genérica presentada anteriormente para redes recurrentes en general. La primera diferencia que se puede observar es que la red LSTM cuenta con una comunicación entre celdas de mayor complejidad. La información de estado queda dividida en dos señales en paralelo, denominadas información oculta, $h_i$, e información de celda o contexto, $c_i$.


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

**Figura 4.13.** Mecanismo de atención sigmoide

El funcionamiento interno a nivel de celda o bloque se puede estudiar en detalle. El elemento principal del bloque es el mecanismo de puertas o *gates* de tipo sigmoide (Figura 4.13). Se trata de un mecanismo de atención que determina la cantidad de información que se propaga a lo largo de la celda. Es importante interpretar las distintas señales como vectores de dimensión D o buses de datos, de forma que existen múltiples sigmoides en paralelo que aplican un valor entre 0 y 1 diferente a su dimensión correspondiente. La función sigmoide limita, por tanto, la información de estado transmitida entre etapas, manteniendo únicamente la información necesaria para resolver la tarea.

El bloque interno de la LSTM se puede dividir en 3 secciones funcionales delimitadas con línea discontinua en la Figura 4.12: *forget gate* o puerta del olvido, *input gate* o puerta de entrada y *output gate* o puerta de salida. Las dos primeras puertas se encargan de determinar qué parte de la información de estado pasada se mantiene en la celda actual y qué parte de la información de entrada se añade a la información de estado nueva propagada hacia la siguiente celda, respectivamente. La tercera puerta se encarga de generar la salida de la celda actual. El funcionamiento interno a nivel de celda y de cada puerta se detalla en el Anexo B.

#### 4.2.1.2 *Gated Recurrent Unit* (GRU)

Las redes LSTM son modelos complejos con numerosas operaciones, de forma que su tamaño es considerable. Las redes *Gated Recurrent Unit* (GRU), propuestas en 2014 [34], tienen por objetivo conseguir una arquitectura más simple y compacta tanto en implementación como en su entrenamiento.

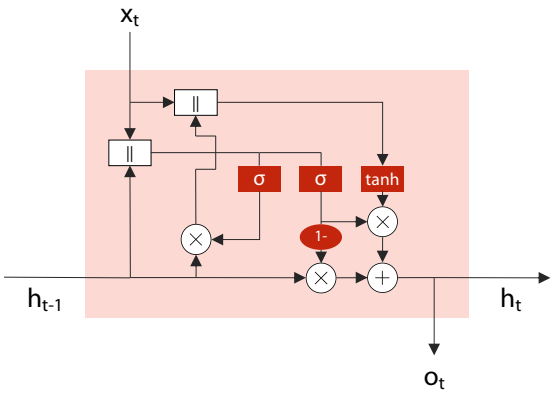


**Figura 4.14.** Bloque de procesado red GRU

La arquitectura general de la GRU es idéntica a la LSTM. La diferencia entre ambas radica en el bloque de procesado, con un diseño interno distinto. En primer lugar, se unifican las dos variables de estado en un única que recoge toda la información de contexto. En segundo lugar, las puertas *input* y *forget* no tienen mecanismos de atención independientes, sino que son complementarios (σ y 1-σ), de forma que se utiliza la misma subred. Estas dos modificaciones permiten una implementación simplificada y de menor tamaño.

# 5 Experimentos y resultados

En este capítulo se describen las bases de datos empleadas en este trabajo así como los diferentes modelos implementados junto con los resultados obtenidos. El objetivo del trabajo es mostrar la capacidad del aprendizaje no supervisado, a partir de la técnica CPC propuesta, para construir modelos preentrenados mediante grandes cantidades de datos en bruto que sean útiles en aplicaciones donde se dispone de pocos datos etiquetados.

La metodología seguida consiste en entrenar distintos modelos de forma no supervisada mediante una base de datos determinada y, por otro lado, evaluar su grado de utilidad en tareas consideradas secundarias en las que se hace uso de otro *dataset* de menor tamaño. El entrenamiento no supervisado mediante CPC se ha denominado tarea principal, mientras que las diferentes tareas sobre las que se ha aplicado se denominan tareas secundarias.

## 5.1 Bases de datos

### 5.1.1 UCF101 – *Action Recognition Data Set*

La base de datos UCF101 [35] es un *dataset* de reconocimiento de acciones formado por 13.320 vídeos recogidos de *YouTube* y clasificados en 101 categorías diferentes. El *dataset* proporciona una gran diversidad de vídeos tanto desde el punto de vista de acciones como de características de cada escena (Figura 5.1). Este *dataset* se utiliza como base de datos principal, es decir, en el entrenamiento no supervisado de modelos, por lo que no interesan las etiquetas o clases en las que se divide sino únicamente el grado de variedad de los ejemplos de vídeo.

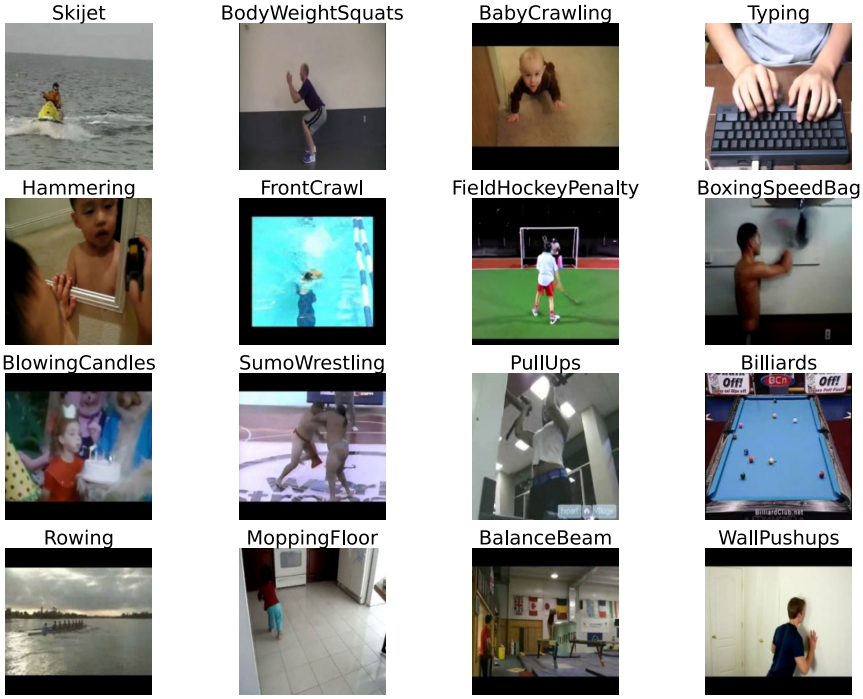


**Figura 5.1.** Ejemplos *dataset* UCF101 (fotograma intermedio)

### 5.1.2 HMDB51 – *Dataset for human motion recognition*

El *dataset* HMDB51 [36] es una recopilación de vídeos procedentes de diferentes fuentes, incluyendo películas y vídeos disponibles en la red. En concreto, la base de datos está compuesta por 6.849 vídeos agrupados en 51 categorías de acciones humanos. La Figura 5.2 muestra el fotograma inicial de 16 ejemplos del *dataset* junto con sus etiquetas. Esta base de datos, denominada secundaria, se utiliza en las tareas propuestas, que sí hacen uso de etiquetas, como evaluación de la potencia del aprendizaje no supervisado realizado con el *dataset* principal.


**Figura 5.2.** Ejemplos *dataset* HMDB51 (fotograma intermedio)

## 5.2 Tareas desarrolladas

Los experimentos se han implementado en *Python* mediante la librería de aprendizaje automático *PyTorch* [5]. Los modelos se han entrenado utilizando potentes GPUs disponibles en el *cluster* del grupo de investigación. Las capacidades de estos recursos limitan la extensión de los experimentos. De esta forma, se han fijado determinados parámetros entre los que destacan la longitud de los vídeos y el tamaño de los *frames* almacenados.

Los desarrollos muestran la capacidad del aprendizaje no supervisado (tarea principal) para construir extractores de características potentes y versátiles aplicados a dos problemas diferentes (tareas secundarias).

### 5.2.1 Tarea secundaria 1 – Reconocimiento de acciones

La primera tarea secundaria consiste en evaluar las prestaciones del CPC sobre el problema de clasificación de vídeo en las 51 categorías de la base de datos secundaria, HMDB51. Se trata de un *benchmark* de reconocimiento de acciones en el campo de visión por computador. No obstante, el objetivo de este trabajo no es conseguir el mejor resultado en la tarea, sino comparar los resultados obtenidos en modelos preentrenados de forma no supervisada con modelos entrenados de forma completamente supervisada.

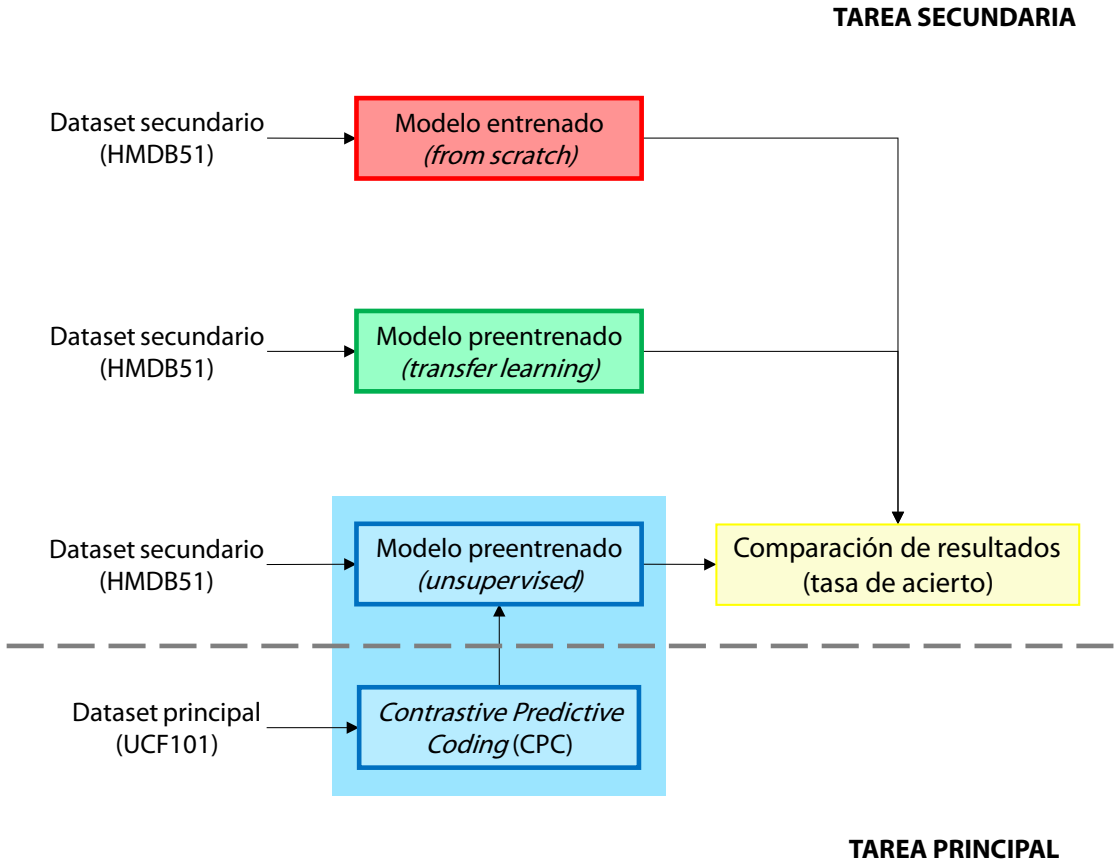


**Figura 5.3.** Esquema tarea secundaria de reconocimiento de acciones

La Figura 5.3 muestra la metodología de trabajo seguida en esta tarea secundaria. En primer lugar, se han desarrollado modelos para conseguir un conjunto de resultados de referencia de forma supervisada. Para ello, se han entrenado modelos desde cero *(from scratch)* y también se han implementado modelos mediante *transfer learning*, es decir, incorporando modelos preentrenados con otros *datasets* y disponibles de forma pública que permiten conseguir mejores resultados.

En segundo lugar, se han construido modelos comparables a dichas referencias tanto en arquitectura como número de parámetros y se han preentrenado mediante aprendizaje no supervisado, a través de la técnica CPC, a partir de la base de datos UCF101. Esta fase de entrenamiento no supervisado es la tarea principal del proyecto. Posteriormente, se ha incorporado el modelo preentrenado *(unsupervised)* a una nueva arquitectura con unas pocas capas finales entrenables para la clasificación del *dataset* secundario HMBD51.

Las resultados obtenidos a partir de modelos no supervisados y las referencias completamente supervisadas se comparan a nivel de tasa de acierto. El objetivo es conseguir porcentajes de *accuracy* comparables a las referencias iniciales, demostrando la ganancia que ha aportado el aprendizaje no supervisado y, en concreto, la técnica CPC.

Los modelos implementados siguen la arquitectura presentada en el capítulo anterior, es decir, un *encoder* junto con un modelo de secuencia. En este caso, el entrenamiento no supervisado pretende conseguir que el *encoder* sea un potente extractor de características para datos de vídeo en general. De esta forma, la tarea secundaria se puede resolver con este extractor para los ejemplos del *dataset* HMDB51 a pesar de que haya sido entrenado con la base de datos UCF101.

Como se ha indicado anteriomente, los experimentos desarrollados comparten una serie de características acordes a los recursos y el tiempo disponibles. La máxima capacidad de memoria, en algunas de las GPUs, es de 11 GB. Todas las tarjetas cuentan con, al menos, 6 GB. Esto ha limitado los modelos a un tamaño de entre 7 y 8 millones de parámetros entrenables. De esta forma, se han fijado los siguientes parámetros:

- Longitud de cada vídeo: 16 *frames*.
- Tamaño de *frames* (resolución): 224x224, 64x64, 32x32 píxeles. La resolución más alta se corresponde con el tamaño utilizado en el *dataset* más conocido en clasificación de imagen, *ImageNet* [37]. Los tamaños reducidos de *frame* permiten construir modelos más pequeños, con un mayor margen respecto a las restricciones de memoria y entrenamientos más rápidos.

En el Anexo A se incluye un listado más amplio de experimentos desarrollados, sus resultados y un resumen de tiempos de entrenamiento e inferencia. En este capítulo se muestran los resultados más relevantes y que permiten validar la potencia del aprendizaje no supervisado en escenario con escasez de datos etiquetados. El tipo de aprendizaje utilizado queda indicado mediante un código de colores que diferencia entre modelos supervisados entrenados desde cero (rojo), modelos supervisados preentrenados mediante técnicas de *transfer learning* (verde) o modelos preentrenados de forma no supervisada (azul).

#### 5.2.1.1 Referencias supervisadas

Los sistemas implementados para obtener las referencias supervisadas (Figura 5.4) cubren las 3 clases de *encoder* explicadas en el capítulo anterior y cuentan con las siguientes características:

- Modelos CNN: El *encoder* sigue una arquitectura con 4 capas convoluciones incrementando el número de canales de 3 a 48, al mismo tiempo que se aplican diezmados en un factor 2. Es una arquitectura sencilla, correspondiente con los modelos iniciales del proyecto, entrenados desde cero.
- Modelos ResNet18: El *encoder* es un modelo ResNet-18 preentrenado *(ImageNet)* y disponible en la propia librería *PyTorch*.
- Modelos ViT: El *encoder* es un modelo ViT preentrenado *(ImageNet)* y disponible de forma pública en [38].

Además, los 3 tipos de modelos mantienen la misma LSTM como modelo de secuencia.

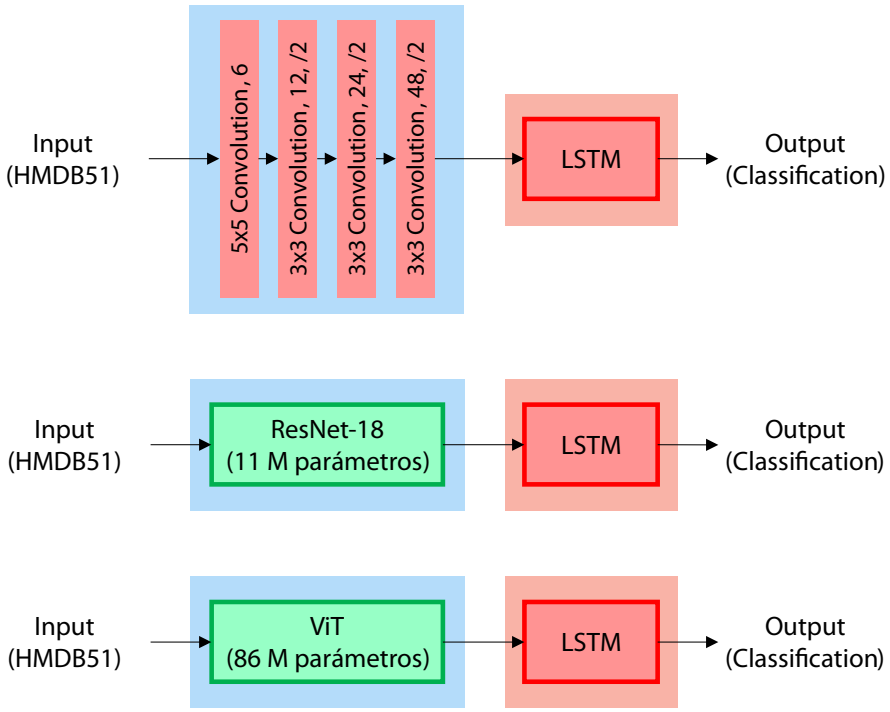


**Figura 5.4.** Esquema modelos supervisados de referencia

Los modelos iniciales, entrenados desde cero, no proporcionaban grandes resultados. Al analizar los mejores resultados públicos para la clasificación del *dataset* HMDB51, se ha comprobado que la mayor parte de ellos se consiguieron preentrenando modelos con bases de datos mayores [25], aplicando *transfer learning*, por lo que se han desarrollado varios modelos (ResNet y ViT) siguiendo esta técnica.

| | | Tarea secundaria | | |
|---|---|---|---|---|
| Frames | Modelo | Validation Accuracy | Test Accuracy | Parámetros |
| 32x32 | CNN 1 | 19,33% | 18,17% | 0,023M |
| - | ResNet18 1 | 19,61% | **19,48%** | 11,255M |
| 64x64 | CNN 2 | 18,67% | 18,24% | 0,023M |
| - | ResNet18 2 | 29,51% | **29,41%** | 11,255M |
| 224x224 | CNN 3 | 21,76% | 20,07% | 0,023M |
| - | ResNet18 3 | 43,14% | 43,66% | 11,255M |
| - | ViT 1 | 58,82% | **62,48%** | 86,271M |

**Tabla 5.1.** Resultados referencias supervisadas (entrenado y evaluado con HMDB51)

Los mejores resultados de modelos supervisados de referencia se muestran en la Tabla 5.1. Los modelos CNN cuentan con muy pocos parámetros debido a problemas de sobreajuste *(overfitting)* relativos al reducido tamaño del *dataset*. Los resultados apenas superan el 20% de acierto, siendo ligeramente inferiores para las dimensiones de 32x32 y 64x64 píxeles, donde la tarea se dificulta debido, lógicamente, al menor tamaño de los *frames*.

Los resultados conseguidos mediante *transfer learning* a partir de un modelo ResNet-18 preentrenado son superiores, como cabría esperar. El mayor número de parámetros (millones frente a miles) permite un mayor grado de acierto en la clasificación de vídeo al mismo tiempo que el preentreno garantiza una mejor generalización, es decir, menor *overfitting*.

Los modelos CNN y ResNet18, basados en redes convolucionales, permiten construir arquitecturas invariantes respecto al tamaño de los *frames* de entrada. El ViT no permite, por construcción, resoluciones de entrada diferentes a la utilizada en el entrenamiento (224x224). Como consecuencia, el modelo preentrenado de ViT sólo se ha podido incorporar a los experimentos de mayor resolución. No obstante, se consigue la máxima tasa de acierto, cercana al 60%, demostrando la potencial superioridad de los *transformers* a las redes basadas en CNNs.

La potencia mostrada por el ViT invita a construir modelos para las resoluciones más bajas. Sin embargo, se trata de modelos con un número de parámetros muy elevado, de forma que no se podrían entrenar simplemente con el *dataset* secundario. En este contexto, el aprendizaje no supervisado cobra la totalidad de su sentido, permitiendo preentrenar estos modelos de gran tamaño con una base de datos mucho mayor sin etiquetar y utilizar el extractor de características conseguido para completar la tarea secundaria de forma supervisada. La idea es similar al *transfer learning* pero sin hacer uso de etiquetas, un factor claramente diferencial tanto desde un punto de vista de coste como de escalabilidad.

#### 5.2.1.2 Modelos no supervisados (CPC) - ResNet-18

Los primeros experimentos de aprendizaje no supervisado se han construido utilizando la ResNet-18 como *encoder*. En concreto, se han sustituido las últimas 11 capas del modelo (Figura 4.4) por dos capas convolucionales que incrementan de 128 a 512 el número de canales (Figura 5.5).

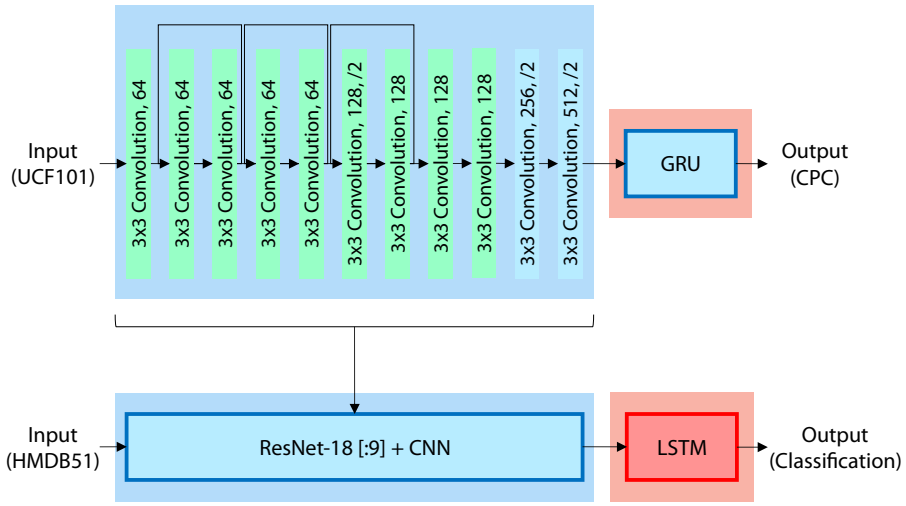


**Figura 5.5.** Esquema entrenamiento no supervisado ResNet-18

La Figura 5.5 muestra el entrenamiento seguido para estos modelos iniciales no supervisados. La arquitectura del modelo no supervisado está formada por las 9 primeras capas del modelo ResNet-18 preentrenado ya comentado, 2 capas convolucionales y una GRU como modelo de secuencia. En los modelos supervisados se han utilizado LSTMs como *sequence models*, mientras que en los no supervisados se utilizan GRUs, de acuerdo con los resultados presentados en [4]. Los elementos no supervisados de la arquitectura se entrenan mediante CPC a partir del *dataset* UCF101 (tarea principal), mientras las capas iniciales permanecen fijas.

El *encoder* del modelo no supervisado entrenado constituye un extractor de características de vídeo a nivel de *frame* gracias a la técnica CPC, es decir, considera no sólo cada fotograma individualmente sino también las relaciones temporales del vídeo completo. Este extractor se utiliza en un nuevo modelo, en este caso, supervisado, en el que únicamente se entrena una LSTM mediante la base de datos HMDB51 para construir el clasificador final (tarea secundaria).

| | | Tarea principal | | | Tarea secundaria | |
|---|---|---|---|---|---|---|
| Frame | Modelo | Batch | Muestras negativas | Test Accuracy | Test Accuracy | Parámetros |
| 32x32 | ResNet18 4 | 128 | 16 | 94,96% | 26,86% | 2,239M |
| - | ResNet18 5 | - | 127 | 76,71% | 29,31% | - |
| - | ResNet18 6 | 256 | 255 | **75,28%** | **30,72%** | - |
| 64x64 | ResNet18 7 | 128 | 64 | 85,50% | 31,70% | - |
| - | ResNet18 8 | - | 96 | 77,07% | 31,81% | - |
| - | ResNet18 9 | - | 127 | **76,87%** | **31,90%** | - |
| 224x224 | ResNet18 10 | 64 | 16 | 87,56% | 31,12% | - |
| - | ResNet18 11 | - | 32 | 81,26% | 31,77% | - |
| - | ResNet18 12 | - | 64 | **73,17%** | **32,42%** | - |

**Tabla 5.2.** Resultados modelos no supervisados (CPC) ResNet-18 (entrenado con UCF101 y evaluado en HMDB51)

Los resultados más significativos de este tipo de modelos se recogen en la Tabla 5.2, que deben compararse con las 3 referencias supervisadas para ResNet-18 mostrada en la Tabla 5.1. Los modelos tienen un menor número de parámetros debido a las capas finales sustituidas respecto al modelo preentrenado. Esta decisión tiene por objetivo que el modelo pueda entrenarse en todas las GPUs disponibles, no sólo en las de mayor capacidad.

En el caso de *frames* de 224x224 píxeles, la tasa de acierto es un 10% inferior al valor de referencia recogido en la Tabla 5.1 (43,66%). Sin embargo, es necesario recordar que el modelo preentrenado cuenta con 11 millones de parámetros y ha sido entrenado con la base de datos *ImageNet*, con 14 millones de ejemplos frente a los 13.320 ejemplos del *dataset* UCF101.

Los modelos de menor tamaño de *frame*, 32x32 y 64x64 píxeles, presentan una tasa de acierto superior a las referencias supervisadas (19,48 y 29,41%). Se trata de una primera muestra de la capacidad del aprendizaje no supervisado. Por otro lado, es necesario recordar que el modelo preentrenado está concebido para fotogramas de 224x224 píxeles, por lo que su funcionamiento puede no ser del todo correcto para tamaños menores. No obstante, estos resultados sirven de motivación para construir modelos ViT para *frames* reducidos y entrenarlos de forma no supervisada.

Finalmente, se han analizado los efectos de distintos parámetros (tamaño de *batch* y número de muestras negativas) del CPC, encontrando tendencias que se mantienen en todos los modelos. El tamaño de *batch* es diferente según la resolución de los *frames* debido a las limitaciones de memoria en los recursos disponibles. Los resultados en la tarea secundaria son mejores cuanto mayor es el *batch* utilizado, demostrando lo indicado en [1]. El aumento del número de muestras negativas reduce la tasa de acierto en la tarea principal pero mejora el resultado de la tarea secundaria. La dificultad de la tarea principal se incrementa al contar con un mayor número de opciones para determinar el siguiente *frame*, obligando a construir un extractor de características más potente y generalizable. Por esta razón, en los experimentos sucesivos se han mantenido N-1 muestras negativas para el CPC, siendo N el tamaño del *batch*.

#### 5.2.1.3 Modelos no supervisados (CPC) - ViT

Los modelos no supervisados basados en ViT se han centrado en *frames* de 64x64 píxeles de resolución, siguiendo la misma estructura utilizada hasta ahora (Figura 5.6).

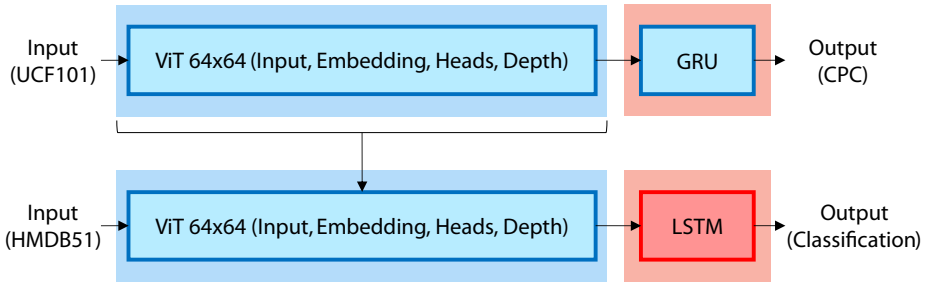


**Figura 5.6.** Esquema entrenamiento no supervisado ViT (64x64)

La arquitectura se diseña desde cero, por lo que ha sido necesario implementar configuraciones distintas para encontrar las más adecuadas (Tabla 5.3).

| | Arquitectura | | | | | Tarea Principal | Tarea Secundaria | |
|---|---|---|---|---|---|---|---|---|
| Modelo | Input | Embedding | Heads | Head Dimension | Depth | Test Accuracy | Test Accuracy | Parámetros |
| ViT 1 | 192 | 256 | 6 | 16 | 6 | 77,00% | 22,09% | 1,479M |
| ViT 2 | - | - | - | - | 8 | 77,10% | 22,61% | 1,924M |
| ViT 3 | - | - | - | - | 12 | 77,02% | 23,66% | 2,815M |
| ViT 4 | - | - | - | - | 16 | 76,81% | **23,92%** | 3,706M |
| ViT 5 | - | - | - | - | 20 | 77,00% | 22,61% | 4,597M |
| ViT 6 | - | - | - | - | 24 | 76,92% | 21,96% | 5,488M |
| ViT 7 | - | - | 12 | - | 2 | 76,30% | 21,44% | 0,735M |
| ViT 8 | - | - | - | - | 4 | 77,01% | 23,46% | 1,328M |
| ViT 9 | - | - | - | - | 8 | 77,67% | 22,81% | 2,514M |
| ViT 10 | - | - | - | - | 12 | 77,57% | 23,33% | 3,700M |
| ViT 11 | - | - | - | - | 16 | 77,28% | 24,18% | 4,886M |
| ViT 12 | - | - | - | - | 20 | 77,88% | **25,23%** | 6,071M |
| ViT 13 | - | - | - | 32 | 2 | 76,94% | 20,07% | 1,030M |
| ViT 14 | - | - | - | - | 4 | 77,84% | 23,33% | 1,918M |
| ViT 15 | - | - | - | - | 8 | 77,55% | **23,92%** | 3,694M |
| ViT 16 | - | - | - | - | 12 | 78,13% | 21,90% | 5,469M |
| ViT 17 | - | - | - | - | 16 | 78,00% | 22,88% | 7,245M |
| ViT 18 | 384 | 512 | 6 | 16 | 2 | 77,10% | 25,10% | 1,857M |
| ViT 19 | - | - | - | - | 4 | 78,13% | 25,62% | 3,337M |
| ViT 20 | - | - | - | - | 6 | 78,37% | **26,67%** | 4,818M |
| ViT 21 | - | - | - | - | 8 | 78,56% | 25,52% | 6,299M |
| ViT 22 | - | - | 12 | - | 2 | 77,31% | 25,42% | 2,152M |
| ViT 23 | - | - | - | - | 4 | 78,17% | 25,23% | 3,927M |
| ViT 24 | - | - | - | - | 6 | 78,22% | **28,04%** | 5,703M |
| ViT 25 | - | - | - | - | 8 | 78,40% | 26,73% | 7,479M |
| ViT 26 | - | - | - | 32 | 2 | 77,62% | 25,29% | 2,741M |
| ViT 27 | - | - | - | - | 4 | 78,60% | 26,54% | 5,107M |
| ViT 28 | - | - | - | - | 6 | 78,83% | **27,39%** | 7,472M |

**Tabla 5.3.** Análisis configuraciones ViT (64x64) no supervisado (entrenado con UCF101 y evaluado en HMDB51)

Como se ha explicado en el capítulo anterior, la estructura interna del ViT consiste en un conjunto de bloques sucesivos cuyo funcionamiento está basado en esquemas de atención en paralelo. Las distintas configuraciones evaluadas analizan 5 aspectos o hiperparámetros diferentes: el número de bloques *(depth)*, las dimensiones de entrada *(input)* y salida *(embedding)*, el número de cabezales *(heads)* y su dimensión *(head dimension)*.

El primer hiperparámetro analizado es el número de bloques o la profundidad del ViT. La configuración inicial en cuanto a hiperparámetros se ha escogido a partir del modelo preentrenado utilizado anteriormente [38]. En los 6 primeros modelos presentados en la Tabla 5.3 se incrementa el número de bloques desde 6 hasta 24. Se puede comprobar que la mayor profundidad del modelo permite conseguir un mejor resultado en la tarea secundaria. Sin embargo, a partir de un número de bloques muy alto las prestaciones comienzan a degradarse.

Los siguientes aspectos analizados son el número de cabezales y su dimensión. El aumento de estos hiperparámetros requiere de una reducción de la profundidad, ya que los recursos disponibles son limitados. El número de cabezales se incrementa entre los modelos 7 y 12. El resultado es una mejora en la tasa de acierto de esta tarea secundaria o la posibilidad de conseguir la misma tasa con menos bloques. En el caso de su dimensión, la mejora es más reducida. De esta forma, el número de subtareas en paralelo es un factor más importante que la complejidad de estos cabezales en este caso.

Los últimos hiperparámetros evaluados son las dimensiones de entrada y salida. Se trata de cambios similares a otro tipos de modelos, como el número de canales en una CNN. La evolución de los modelos 18 a 26 muestra claramente que el aumento de dimensiones resulta en un incremento más significativo de la tasa de acierto en la clasificación de vídeo.

El aumento de las dimensiones de entrada y salida permite reducir el número de bloques junto con un incremento de cabezales y dimensión, como se puede comprobar en los últimos 3 modelos. En conclusión, la mejor configuración consiste en una combinación equilibrada de los hiperparámetros, siendo importante tanto la cantidad de cabezales como la profundidad completa del modelo. Los siguientes modelos desarrollados, también entrenados desde cero, mantienen la última configuración presentada, de máxima tasa de acierto, como arquitectura del ViT.

En los modelos basados en ResNet-18 se ha estudiado la influencia del tamaño del *batch* y del número de muestras negativas en la técnica no supervisada utilizada. En este caso, se incorpora el análisis del número de predicciones realizadas, es decir, cuántos *frames* y en qué posición se predicen y computan en las pérdidas del CPC.

En concreto, se han propuesto modelos en los que se añade progresivamente un *frame* adicional al siguiente (k+1) utilizado hasta ahora. De esta forma, se pueden llegar a predecir conjuntamente, por ejemplo, fotogramas a distancia 1, 2, 4, 6 del actual. Este análisis se realiza para las tres últimas configuraciones presentadas en la Tabla 5.3.

| | Tarea Principal | | | Tarea Secundaria | |
|---|---|---|---|---|---|
| Modelo | Depth | Predicted frames | Test Accuracy | Test Accuracy | Parámetros |
| ViT 29 | 6 | k+1 | 78,83% | 26,54% | 7,472M |
| ViT 30 | - | k+1, k+2 | 72,80% | 28,82% | - |
| ViT 31 | - | k+1, k+2, k+4 | 67,18% | 28,37% | - |
| ViT 32 | - | k+1, k+2, k+4, k+6 | 62,85% | 29,22% | - |
| ViT 33 | - | k+1, k+2, k+4, k+6, k+8 | 60,60% | 29,07% | - |
| ViT 34 | - | k+1, k+2, k+4, k+6, k+8, k+10 | 59,23% | **30,92%** | - |
| ViT 35 | - | k+1, k+2, k+4, k+6, k+8, k+10, k+12 | 58,80% | 29,93% | - |
| ViT 36 | - | k+1, k+2, k+4, k+6, k+8, k+10, k+12, k+14 | 59,50% | 30,33% | - |
| ViT 37 | 4 | k+1 | 78,60% | 27,39% | 5,107M |
| ViT 38 | - | k+1, k+2 | 72,41% | 27,25% | - |
| ViT 39 | - | k+1, k+2, k+4 | 66,82% | 28,54% | - |
| ViT 40 | - | k+1, k+2, k+4, k+6 | 62,85% | 29,80% | - |
| ViT 41 | - | k+1, k+2, k+4, k+6, k+8 | 60,31% | 29,28% | - |
| ViT 42 | - | k+1, k+2, k+4, k+6, k+8, k+10 | 59,65% | **30,87%** | - |
| ViT 43 | - | k+1, k+2, k+4, k+6, k+8, k+10, k+12 | 59,14% | 30,46% | - |
| ViT 44 | - | k+1, k+2, k+4, k+6, k+8, k+10, k+12, k+14 | 59,42% | 28,76% | - |
| ViT 45 | 2 | k+1 | 77,62% | 25,29% | 2,741M |
| ViT 46 | - | k+1, k+2 | 71,68% | 26,99% | - |
| ViT 47 | - | k+1, k+2, k+4 | 66,21% | 26,41% | - |
| ViT 48 | - | k+1, k+2, k+4, k+6 | 62,55% | 27,78% | - |
| ViT 49 | - | k+1, k+2, k+4, k+6, k+8 | 60,59% | 29,35% | - |
| ViT 50 | - | k+1, k+2, k+4, k+6, k+8, k+10 | 59,16% | **29,87%** | - |
| ViT 51 | - | k+1, k+2, k+4, k+6, k+8, k+10, k+12 | 58,98% | 28,04% | - |
| ViT 52 | - | k+1, k+2, k+4, k+6, k+8, k+10, k+12, k+14 | 59,22% | 27,58% | - |

**Tabla 5.4.** Análisis número y posición de *frames* predichos ViT 64x64 (entrenado con UCF101 y evaluado en HMDB51)

La Tabla 5.4 permite comprobar, para las tres configuraciones del ViT, que el aumento del número de *frames* predichos proporciona un mejor resultado en la tarea secundaria. La complejidad de la tarea principal se incrementa, con tasas de acierto inferiores, ya que se debe escoger correctamente más de 1 fotograma y en posiciones cada vez más alejadas del actual. De esta forma, se está obligando a construir un extractor de características más potente, que considere tanto relaciones temporales de baja y alta resolución temporal, es decir, con un nivel de abstracción elevado que englobe el contenido del vídeo.

La tendencia alcista del acierto en la tarea secundaria alcanza su máximo en el décimo *frame* (k+10) y comienza a descender. La distancia entre fotogramas resulta demasiado elevada, forzando la búsqueda de un contexto temporal excesivamente grande e improbable.

En el capítulo anterior se indica que el ViT, a diferencia de muchas arquitecturas de redes neuronales, no es un modelo de caja negra. De esta forma, se puede encontrar cierta explicabilidad en el comportamiento de la red.

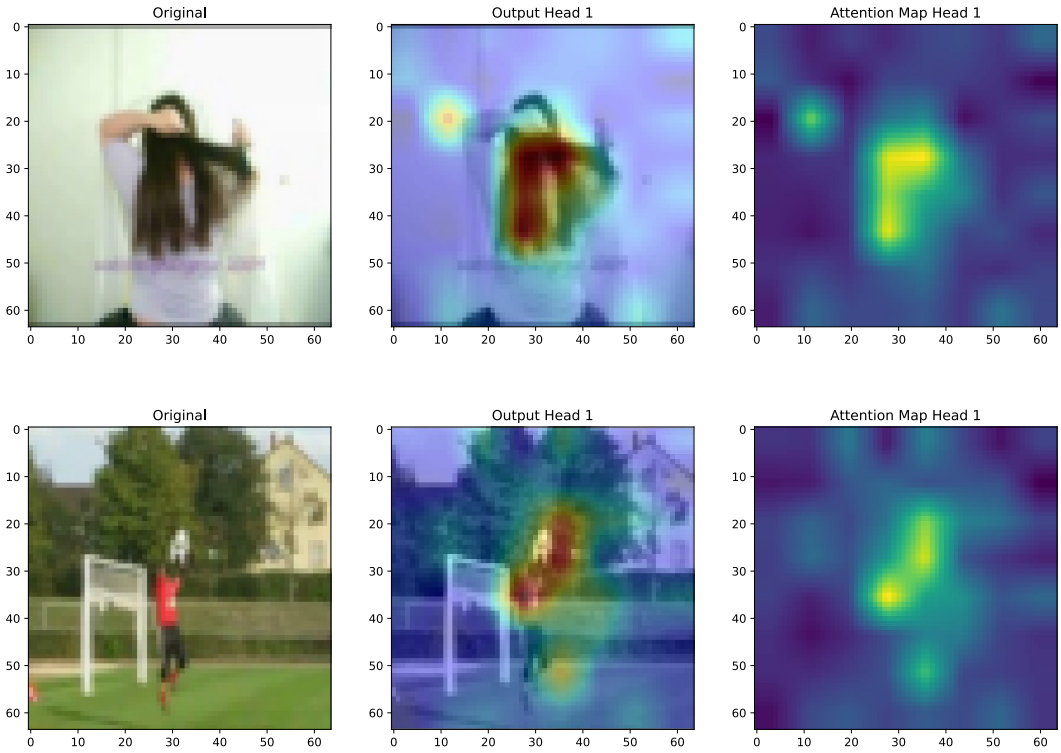


**Figura 5.7.** Original, salida y mapa de atención para 1 cabezal ejemplos clases *brush hair* y *catch*

La Figura 5.7 muestra el *frame* original, su salida y el mapa de atención global que la origina para 1 cabezal concreto del modelo ViT 34 en dos ejemplos de vídeo para las clases *brush hair* y *catch*, respectivamente. El mapa de atención se obtiene, para cada cabezal por separado, como producto de las matrices de atención en cada bloque del ViT. De esta forma, se obtiene una imagen global que representa la atención del cabezal a un *frame* concreto. La salida se calcula multiplicando la máscara de atención por el fotograma original.

El resultado obtenido para los dos ejemplos presentados muestra la capacidad del modelo preentrenado para fijar su atención en las zonas donde se produce el movimiento. En concreto, se puede comprobar un valor alto en el mapa de atención en los píxeles correspondientes al peine y en los correspondientes al balón en las manos del portero.

Finalmente, se analiza la dependencia del modelo no supervisado respecto al tamaño del *dataset* utilizado. El objetivo del trabajo, como ya se ha explicado con anterioridad, es mostrar la potencia de la técnica utilizada para construir un extractor de características útil en aplicaciones supervisadas con pocos datos etiquetados. Este tipo de solución permitiría utilizar enormes conjuntos de datos sin etiquetas, muy fáciles de generar y disponibles, en muchas ocasiones, de forma pública.

El análisis consiste en repetir el entrenamiento para el modelo más importante del proyecto, ViT 34, reduciendo la cantidad de ejemplos utilizados en la tarea principal, es decir, en el entrenamiento no supervisado mediante CPC.

| | Tarea Principal | | | Tarea Secundaria | |
|---|---|---|---|---|---|
| Modelo | Tamaño Dataset (%) | Ejemplos entrenamiento | Test Accuracy | Test Accuracy | Parámetros |
| ViT 53 | 25 | 1998 | 62,74% | 26,67% | 7,472M |
| ViT 54 | 50 | 3996 | 63,04% | 28,89% | - |
| ViT 55 | 75 | 5994 | 61,12% | 29,22% | - |
| ViT 34 | 100 | 7992 | 59,23% | **30,92%** | - |

**Tabla 5.5.** Análisis tamaño *dataset* no supervisado (entrenado con UCF101 y evaluado en HMDB51) en ViT (64x64)

Los resultados obtenidos (Tabla 5.5) permiten comprobar que el acierto en la tarea secundaria decae progresivamente al reducir la cantidad de datos utilizados en el entrenamiento. La Figura 5.8 muestra una tendencia clara de mejora en la clasificación de vídeo al aumentar la cantidad de datos en bruto utilizados en la tarea principal.

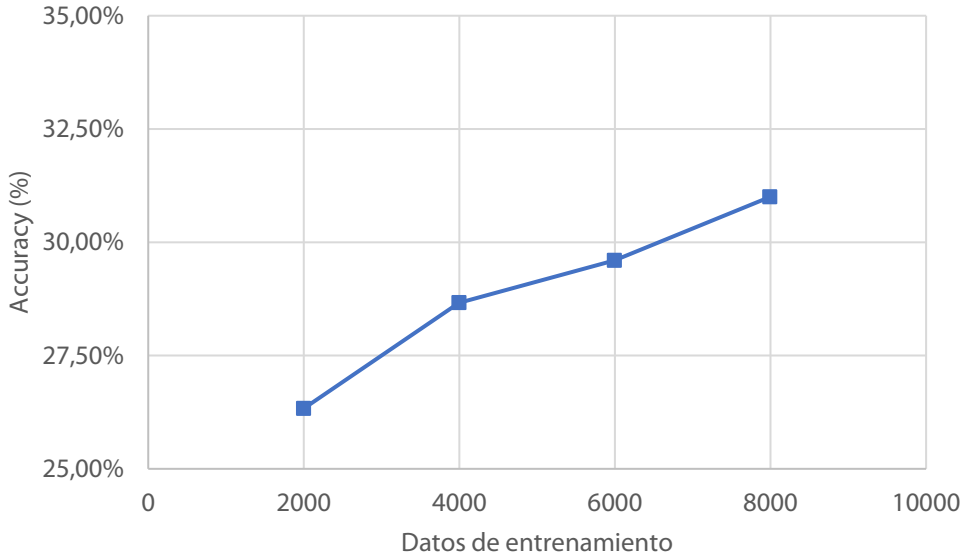


**Figura 5.8.** Efecto tamaño *dataset* no supervisado

Esta tendencia se puede encontrar también en tareas supervisadas. La diferencia fundamental de este proyecto radica en que los datos utilizados no tienen etiquetas, de forma que cualquier nuevo dato, independientemente de su contenido contribuiría a construir un extractor más potente. Además, es esencial recordar que los *datasets* utilizado son de diferente temática.

#### 5.2.1.4 Resumen y comparativa de resultados

Los resultados obtenidos a partir de modelos entrenados de forma no supervisada se comparan con las referencias supervisadas en la siguiente tabla resumen (Tabla 5.6).

| | | Tarea secundaria | | |
|---|---|---|---|---|
| Frames | Modelo | Validation Accuracy | Test Accuracy | Parámetros |
| 32x32 | CNN 1 | 19,33% | 18,17% | 0,023M |
| - | ResNet18 1 | 19,61% | 19,48% | 11,255M |
| - | ResNet18 6 | 30,35% | 30,72% | 2,239M |
| 64x64 | CNN 2 | 18,67% | 18,24% | 0,023M |
| - | ResNet18 2 | 29,51% | 29,41% | 11,255M |
| - | ResNet18 9 | 32,40% | 31,90% | 2,239M |
| - | ViT 34 | 31,00% | 30,92% | 7,472M |
| 224x224 | CNN 3 | 21,76% | 20,07% | 0,023M |
| - | ResNet18 3 | 43,14% | 43,66% | 11,255M |
| - | ViT 1 | 58,82% | 62,48% | 86,271M |
| - | ResNet18 12 | 32,59% | 32,42% | 2,239M |

**Tabla 5.6.** Tabla resumen de comparativa mejores resultados diferentes arquitecturas (referencias supervisadas entrenadas y evaluadas con HMD51, modelos no supervisados entrenados con UCF101 y evaluados en HMDB51)

Los resultados obtenidos permiten comprobar que los modelos no supervisados son competitivos frente a las referencias supervisadas. En el caso de los *frames* de mayor tamaño, 224x224 píxeles, el modelo ResNet18 12 se encuentra 11 puntos por debajo del modelo completamente preentrenado, pero cuenta con un total de parámetros 5 veces menor.

En el caso de los *frames* de tamaño inferior, 32x32 y 64x64 píxeles, los modelos preentrenados de forma no supervisada superan a los modelos de *transfer learning*. Cabe recordar que el modelo ResNet18 está preentrenado a partir de *ImageNet*, que utiliza imágenes de 224x224 píxeles, por lo que probablemente exista una degradación de prestaciones importante al utilizarlo con otros tamaños. No obstante, los modelos no supervisados han permitido conseguir tasas de acierto superiores al 30% para resoluciones realmente pequeñas y siempre a partir de datos sin etiquetar, uno de los aspectos fundamentales de este trabajo. Se trata de una evidencia clara sobre la capacidad del aprendizaje no supervisado para conseguir resolver tareas en las que se dispone de muy pocos datos etiquetados.

Por otro lado, se demuestra la capacidad del ViT para conseguir modelos potentes que compitan con los mejores modelos convolucionales. En la resolución de 64x64 píxeles el modelo no supervisado ViT 34 consigue una tasa de acierto ligeramente inferior al modelo ResNet18 9. Sin embargo, es muy importante recordar que el modelo ViT se ha construido y entrenado de forma no supervisada desde cero, mientras que el modelo ResNet cuenta con hasta 9 capas convolucionales preentrenadas de forma supervisada con el *dataset ImageNet*, de forma que el entrenamiento no supervisado termina ajustando únicamente las capas finales del mismo. Por tanto, el modelo ViT 34 se considera como el resultado más importante de este proyecto, mostrando claramente las capacidades del aprendizaje no supervisado y la técnica CPC como solución en un escenario de pocos datos etiquetados.

### 5.2.2 Tarea secundaria 2 – Motor de búsqueda

La segunda tarea secundaria se basa en la implementación de un motor de búsqueda a partir de la representación aprendida de forma no supervisada. Los resultados se presentan de forma más cualitativa que en el caso anterior. El objetivo es mostrar que la utilidad del extractor aprendido no se limita a un problema de clasificación de vídeo, sino que cuenta con una gran versatilidad, de forma que puede aplicarse a otro tipo de tareas.

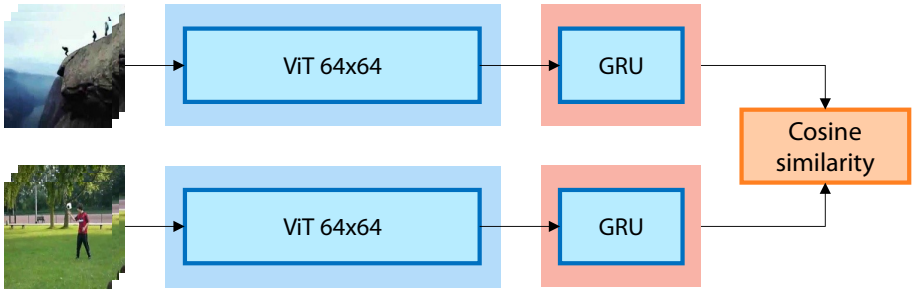


**Figura 5.9.** Esquema tarea secundaria de motor de búsqueda

La Figura 5.9 muestra el esquema seguido en esta tarea. En este caso, se utiliza un modelo no supervisado completo, es decir, *encoder* y modelo de secuencia. En concreto, se utiliza el mejor modelo conseguido en la tarea anterior, ViT 34. Las salidas de la GRU se promedian y se obtiene un vector de características que engloba el vídeo completo. De esta forma, se extraen representaciones de los vídeos del *dataset* secundario, HMDB51, a partir de un modelo preentrenado con la base de datos principal, UCF101, sin realizar ningún entrenamiento adicional.

Las representaciones obtenidas para dos vídeos distintos se comparan mediante su similitud coseno.

$$\cos(\theta) = \frac{A \cdot B}{\|A\| \cdot \|B\|} \qquad \text{(Ec. 5.1)}$$

La similitud coseno indica el alineamiento entre dos vectores de características en el nuevo espacio de representación. La medida toma valores entre 0 y 1, siendo mayor cuanto más próximos son los vectores.

La tarea de motor de búsqueda consiste en encontrar un conjunto de vídeos semejantes, desde el punto de vista de similitud coseno, a uno dado. Se trata, por tanto, de introducir un vídeo *(query)* y obtener una respuesta *(response)* en forma de lista de vídeos similares.

La Figura 5.10 muestra un ejemplo de funcionamiento del motor de búsqueda. Se representa el vídeo *query*, correspondiente a la clase *catch* y dos de los elementos de la lista obtenida como *response*. Se ha escogido la novena *response* por su utilidad en la explicación. La primera respuesta indica que la representación obtenida permite distinguir el movimiento correspondiente a atrapar un objeto. Sin embargo, la novena respuesta parece indicar que la representación tiene una fuerte dependencia con los colores de fondo, aunque también podría deberse a la postura inicial del jugador.

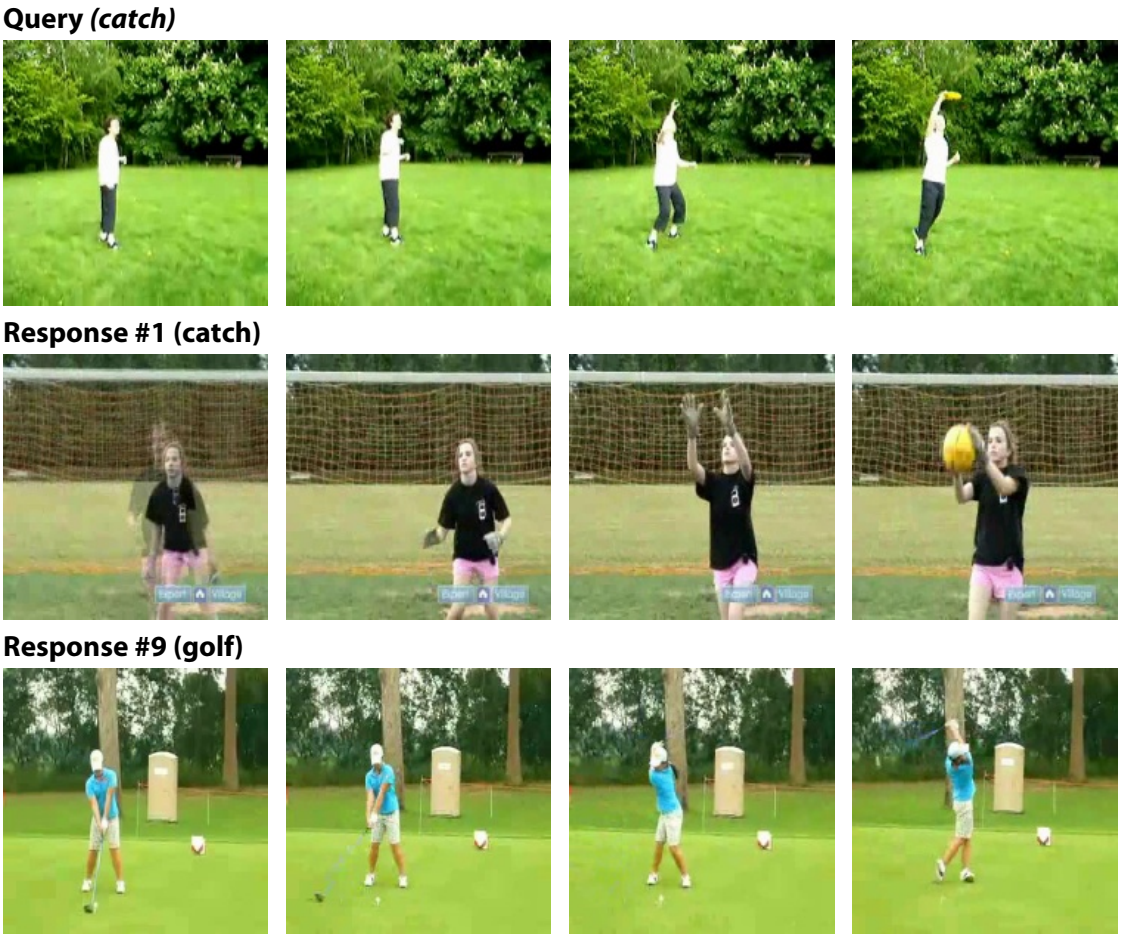


**Figura 5.10.** Ejemplo motor de búsqueda *query-response* para ejemplo clase *catch* (*embeddings* de HMDB51 generados a partir de modelo entrenado con UCF101)

El motor de búsqueda se puede utilizar para identificar la clase de un vídeo, desconocida a priori, a partir de su similitud coseno respecto a distintos vídeos de clase conocida. El sistema implementado proporciona las 5 clases de mayor similitud coseno con el vídeo de entrada.

1. catch – 0,9750
2. somersault – 0,8887
3. flic flac – 0,8810
4. golf – 0,8706
5. cartwheel – 0,8677

**Figura 5.11.** Ejemplo motor de búsqueda identificación vídeo clase *catch*

En la Figura 5.11 se muestra un ejemplo de identificación para el mismo vídeo. La clase de mayor similitud es *catch*, permitiendo un etiquetado correcto. Al analizar el resto de clases, se ha comprobado que los ejemplos de mayor similitud coseno comparten el color de fondo verde junto con la postura inicial de pie, por lo que la representación no captura únicamente el movimiento. No obstante, es importante recordar que se ha utilizado directamente, sin entrenamientos adicionales, una representación obtenida a partir de una base de datos muy pequeña (13.320 vídeos), por lo que parece lógico encontrar limitaciones.

# 6 Conclusiones

El proyecto ha consistido en el desarrollo de modelos basados en redes neuronales profundas mediante técnicas no supervisadas para la construcción de extractores de características generalizables aplicados a datos de vídeo. El trabajo muestra la necesidad de enormes bases de datos etiquetadas para la resolución de problemas actuales y propone el aprovechamiento masivo de datos en bruto sin etiquetar, fácilmente generables y disponibles en grandes cantidades de forma pública, a través de distintos modelos que evalúan su capacidad en el campo multimedia de vídeo.

La técnica no supervisada (CPC) utilizada se ha evaluado mediante dos tareas, que se consideran secundarias, en las que el extractor de características resultante del entrenamiento no supervisado, la tarea principal, se aplica directamente o se incorpora a un modelo secundario, utilizando un conjunto de datos diferente y no conocido a priori por el extractor denominado *dataset* secundario. La primera tarea secundaria consiste en una clasificación supervisada de vídeo, mientras que, en la segunda, el extractor se utiliza para construir un sistema de motor de búsqueda en el que se determina la etiqueta de un vídeo a partir de su parecido con vídeos de diferentes clases.

En la primera tarea secundaria se construye un sistema de reconocimiento de acciones a partir del extractor genérico para vídeo entrenado de forma no supervisada. Como paso previo, se han propuesto una serie de modelos supervisados, entrenados desde cero o incorporando modelos preentrenados *(transfer learning)* disponibles públicamente que sirven de referencia comparativa. En esta fase se ha comprobado la dificultad para conseguir buenos resultados en escenarios con pocos datos etiquetados.

Los modelos desarrollados en esta tarea mediante aprendizaje no supervisado demuestran la capacidad de la técnica utilizada para conseguir resultados competitivos a partir de un conjunto reducido de datos no etiquetados. También se ha demostrado que existe una relación directa entre la tasa de acierto obtenida y la cantidad de datos sin etiquetar utilizados en el entrenamiento. La conclusión principal es que el aprendizaje no supervisado se convierte en una estrategia clara para conseguir resolver tareas para las que se dispone de pocos datos etiquetados, utilizando grandes cantidades de datos en bruto, generables con mucha mayor facilidad.

Por otra parte, los modelos construidos se basan en diferentes arquitecturas. La principal novedad utilizada en el proyecto es la aplicación de los modelos *transformers* , mediante el ViT, al campo de vídeo, dentro de la visión por computador. Estos modelos, basados en esquemas de atención, han permitido conseguir resultados que pueden superar a las arquitecturas convolucionales más extendidas, como las ResNet. En este sentido, los *transformers* se postulan como una solución de máximas prestaciones no sólo en su ámbito original, los modelos de lenguaje, sino en cualquier otro, como es el caso del vídeo.

La segunda tarea secundaria consiste en un motor de búsqueda mediante comparación de vectores de características obtenidos directamente del extractor entrenado de forma no supervisada. El *embedding* de un vídeo de clase desconocida se identifica por medio de una medida similitud con las representaciones de un conjunto de vídeos de etiqueta conocida.

Esta segunda tarea se corresponde con una evaluación simple del extractor de vídeo sin entrenamientos adicionales que sirve de muestra tanto de su generalización como de su versatilidad. Se han presentado diferentes casos de identificación o búsqueda exitosa, en los que es importante recordar que los vídeos del *dataset* utilizado nunca han sido tratados por la red, demostrando la utilidad del modelo no supervisado.

El tratamiento de vídeo como secuencia aplicado en este proyecto es un enfoque innovador dentro del aprendizaje no supervisado, distinto al habitual en desarrollos anteriores en los que se suele realizar únicamente un tratamiento de imagen con imagen, sin tener en cuenta la componente temporal. La técnica CPC se convierte en una estrategia novedosa, realizando un tratamiento diferente, con una función de coste específica y una tarea pretexto, la predicción de *frames*, no utilizada hasta ahora en trabajos anteriores.

El trabajo desarrollado sirve como base para diferentes líneas futuras:

- Entrenamiento no supervisado de modelos mediante bases de datos de mayor tamaño, como por ejemplo, los *datasets* de la familia *Kinetics* [39] (300K – 600K vídeos) o el extenso conjunto *Youtube8M* [40] (8M de vídeos). El objetivo es mejorar los resultados conseguidos, demostrando que el sistema obtiene un extractor de mayores prestaciones simplemente analizando más vídeos, de cualquier naturaleza y temática.
- Modificación del entrenamiento para que el sistema sea capaz de procesar vídeos con diferente número de *frames*. Los resultados obtenidos quedan enmarcados en un escenario con 16 fotogramas por vídeo, independientemente de su longitud. El extractor generado conseguiría prestaciones probablemente superiores utilizando una tasa de *frames* constante, es decir, la misma resolución temporal en cada vídeo.
- Utilización del *transformer* como modelo de secuencia en las diferentes tareas desarrolladas, sustituyendo tanto a la LSTM como a la GRU. Esta línea se apoya en la potencia, no sólo teórica sino también comprobada con muy pocos datos, de esta novedosa arquitectura de atención. En este sentido, el paso adicional sería construir un único *transformer* que realice un tratamiento completo, a nivel espacial y temporal.

La valoración global del proyecto es positiva y satisfactoria. El conocimiento adquirido a lo largo del trabajo es muy amplio y cobra mayor importancia teniendo en cuenta el punto de partida y el tiempo dedicado. Además, supone una introducción bastante real y completa a una temática de actualidad de gran utilidad para comprender su complejidad y profundidad más allá de las aplicaciones finales visibles y conocidas por muchas personas.

# 7 Referencias


[1] A. Oord, Y. Li, and O. Vinyals, "Representation Learning with Contrastive Predictive Coding," *arXiv preprint arXiv:1807.03748,* 2018.

[2] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A Simple Framework for Contrastive Learning of Visual Representations," in *International conference on machine learning*, 2020: PMLR, pp. 1597-1607.

[3] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick, "Momentum Contrast for Unsupervised Visual Representation Learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 9729-9738.

[4] R. Molinero Millán, "Estudio de técnicas de extracción automática de características de señales de voz mediante aprendizaje no supervisado," Trabajo Fin de Máster, Escuela de Ingeniería y Arquitectura (EINA). Universidad de Zaragoza, 2020.

[5] PyTorch - Official Website. https://pytorch.org/.

[6] A. Krizhevsky, I. Sutskever, and G. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," *Neural Information Processing Systems,* vol. 25, 2012.

[7] J. Redmon and A. Farhadi, "YOLOv3: An Incremental Improvement," *arXiv preprint arXiv:1804.02767,* 2018.

[8] E. Shelhamer, J. Long, and T. Darrell, "Fully Convolutional Networks for Semantic Segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 39, pp. 1-1, 2016.

[9] C. Research, "Data Preparation & Labeling for AI 2020," Tech. Rep. 2020.

[10] Scale-Labs.Scale AI: The Data Platform for AI. https://scale.com.

[11] T. Fredriksson, D. Issa Mattos, J. Bosch, and H. Olsson, "Data Labeling: An Empirical Investigation into Industrial Challenges and Mitigation Strategies," in *International Conference on Product-Focused Software Process Improvement*, 2020: Springer, pp. 202-216.

[12] Y. Roh, G. Heo, and S. E. Whang, "A Survey on Data Collection for Machine Learning: A Big Data - AI Integration Perspective," *IEEE Transactions on Knowledge and Data Engineering,* vol. 33, no. 4, pp. 1328-1347, 2021.

[13] A. Torralba and A. Efros, "Unbiased look at dataset bias," in *CVPR 2011*, 2011: IEEE, pp. 1521-1528.

[14] B. Lucas and T. Kanade, "An Iterative Image Registration Technique with an Application to Stereo Vision (IJCAI)," 1981: Vancouver, British Columbia.

[15] D. Ramanan, D. A. Forsyth, and A. Zisserman, "Strike a pose: Tracking people by finding stylized poses," in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, 2005, vol. 1: IEEE, pp. 271-278.

[16] M. Mathieu, C. Couprie, and Y. Lecun, "Deep multi-scale video prediction beyond mean square error," *arXiv preprint arXiv:1511.05440,* 2015.

[17] C. Vondrick, A. Shrivastava, A. Fathi, S. Guadarrama, and K. Murphy, "Tracking Emerges by Colorizing Videos," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 391-408.

[18] I. Misra, C. Zitnick, and M. Hebert, "Shuffle and Learn: Unsupervised Learning Using Temporal Order Verification," in *European Conference on Computer Vision*, 2016: Springer, pp. 527-544.

[19] D. Wei, J. Lim, A. Zisserman, and W. Freeman, "Learning and Using the Arrow of Time," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 8052-8060.

[20] S. Pan and Q. Yang, "A Survey on Transfer Learning," *IEEE Transactions on knowledge and data engineering,* vol. 22, no. 10, pp. 1345-1359, 2010.

[21] T. Borgi, N. Zoghlami, M. Abed, and N. Mohamed Saber, "Big Data for Operational Efficiency of Transport and Logistics: A Review," in *2017 6th IEEE International Conference on Advanced Logistics and Transport (ICALT)*, 2017: IEEE, pp. 113-120.

[22] R. Dubey, P. Agrawal, D. Pathak, T. Griffiths, and A. Efros, "Investigating Human Priors for Playing Video Games," *arXiv preprint arXiv:1802.10217,* 2018.

[23] G. Zhong, L.-N. Wang, and J. Dong, "An Overview on Data Representation Learning: From Traditional Feature Learning to Recent Deep Learning," *The Journal of Finance and Data Science,* vol. 2, no. 4, pp. 265-278, 2016.

[24] T. Han, W. Xie, and A. Zisserman, "Video Representation Learning by Dense Predictive Coding," in *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops*, 2019.

[25] Y. Zhu *et al.*, "A Comprehensive Study of Deep Video Action Recognition," *arXiv preprint arXiv:2012.06567,* 2020.

[26] 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.

[27] NN SVG - Alex Lenail. https://alexlenail.me/NN-SVG/.

[28] A. Gron, *Hands-On Machine Learning with Scikit-Learn and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems*. O'Reilly Media, Inc., 2017.

[29] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770-778.

[30] A. Vaswani *et al.*, "Attention Is All You Need," in *Advances in neural information processing systems*, 2017, pp. 5998-6008.

[31] A. Dosovitskiy *et al.*, "An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale," *arXiv preprint arXiv:2010.11929,* 2020.

[32] A. Rush.The Annotated Transformer - Harvard Natural Language Processing (Harvard NLP). https://nlp.seas.harvard.edu/2018/04/03/attention.html.

[33] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," *Neural Computation,* vol. 9, no. 8, pp. 1735-1780, 1997.

[34] K. Cho, B. van Merriënboer, C. Gulcehre, F. Bougares, H. Schwenk, and Y. Bengio, "Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation," *arXiv preprint arXiv:1406.1078,* 2014.

[35] K. Soomro, A. Zamir, and M. Shah, "UCF101: A Dataset of 101 Human Actions Classes From Videos in The Wild," *arXiv preprint arXiv:1212.0402,* 2012.

[36] H. Kuehne, H. Jhuang, E. Garrote, T. Poggio, and T. Serre, "HMDB51: A Large Video Database for Human Motion Recognition," in *Proc. of IEEE International Conference on Computer Vision*, 2011, vol. 4, no. 5, p. 6.

[37] O. Russakovsky *et al.*, "ImageNet Large Scale Visual Recognition Challenge," *International Journal of Computer Vision,* vol. 115, no. 3, pp. 211-252, 2014.

[38] Vision Transformer (ViT) rwightman. https://rwightman.github.io/pytorch-image-models/models/vision-transformer/#how-do-i-finetune-this-model.

[39] W. Kay *et al.*, "The Kinetics Human Action Video Dataset," *arXiv preprint arXiv:1705.06950,* 2017.

[40] S. Abu-El-Haija *et al.*, "YouTube-8M: A Large-Scale Video Classification Benchmark," *arXiv preprint arXiv:1609.08675,* 2016.

Índice de figuras

Índice de tablas