

Reconocimiento de Acciones Humanas en Video mediante Redes Neuronales Convolucionales

Joel Rojas Jiménez, Juan de Dios Cocoletzi Musito

1. Introducción

El reconocimiento automático de acciones humanas en secuencias de video es una de las áreas más relevantes y desafiantes dentro del campo de la visión por computadora y el aprendizaje profundo. Este problema consiste, de manera general, en identificar correctamente la acción que una persona está realizando a partir de una secuencia temporal de imágenes, lo cual implica analizar tanto la información espacial como la información temporal presente en los videos.

En este trabajo se presenta el desarrollo completo de un sistema de reconocimiento de acciones humanas, desde la concepción inicial de la idea, pasando por diferentes estrategias y enfoques descartados, hasta llegar a una solución final basada en redes neuronales convolucionales (CNN) utilizando el conjunto de datos UCF101. El proyecto fue desarrollado en equipo, lo que permitió discutir, evaluar y justificar cada una de las decisiones tomadas a lo largo del proceso.

2. Contexto y Motivación

El interés por este problema surge debido a sus múltiples aplicaciones prácticas, tales como la vigilancia inteligente, el análisis de comportamiento humano, los sistemas de recomendación de contenido multimedia, la robótica, así como aplicaciones en el ámbito deportivo y médico. Sin embargo, reconocer acciones en video no es una tarea trivial, ya que implica lidiar con variaciones de iluminación, cambios de fondo, diferentes puntos de vista, velocidades de movimiento y duraciones variables de las acciones.

Debido a estas dificultades, se decidió abordar el problema explorando diferentes técnicas de aprendizaje automático y aprendizaje profundo, evaluando sus ventajas y limitaciones antes de definir una solución final adecuada.

3. Antecedentes y Primera Propuesta

En una etapa inicial del proyecto, se planteó como primera idea el uso de redes neuronales recurrentes, específicamente modelos basados en LSTM (Long Short-Term Memory), debido a su capacidad para modelar

dependencias temporales en secuencias de datos. La motivación principal detrás de esta elección fue que los videos, al ser secuencias ordenadas de frames, pueden ser interpretados naturalmente como series temporales.

Para esta primera aproximación, se investigaron datasets clásicos utilizados en reconocimiento de acciones, tales como:

- **Weizmann Action Dataset**
- **KTH Human Action Dataset**

Estos conjuntos de datos son ampliamente utilizados en la literatura, principalmente por su simplicidad y por permitir validar ideas iniciales. Sin embargo, tras un análisis más profundo, se detectaron varias limitaciones importantes.

Por un lado, estos datasets contienen acciones relativamente simples, fondos muy controlados y poca variabilidad, lo que limita su capacidad para representar escenarios reales. Por otro lado, el uso directo de LSTM requería una etapa previa de extracción de características robustas, lo cual complicaba considerablemente la arquitectura del sistema y su entrenamiento dentro del tiempo y recursos disponibles.

4. Cambio de Estrategia: Enfoque con SVM

Como segunda estrategia, se evaluó el uso de clasificadores tradicionales, específicamente máquinas de soporte vectorial (SVM), combinadas con extracción manual de características. En este enfoque, la idea era obtener descriptores visuales a partir de los frames del video y utilizar dichos vectores como entrada para un clasificador SVM.

Si bien este método permitió comprender mejor el flujo completo de un sistema de reconocimiento (extracción de características + clasificación), se observaron varias desventajas. Principalmente, el desempeño dependía fuertemente de la calidad de las características diseñadas manualmente, y el modelo tenía dificultades para generalizar cuando las acciones eran visualmente similares.

Estas observaciones nos llevaron a reconsiderar el uso de modelos capaces de aprender características automáticamente a partir de los datos.

5. Propuesta Final: Redes Neuronales Convolucionales

Finalmente, se optó por una solución basada en redes neuronales convolucionales (CNN), las cuales han demostrado un desempeño sobresaliente en tareas de clasificación de imágenes y video. Las CNN tienen la capacidad de aprender automáticamente características jerárquicas, desde bordes y texturas hasta patrones más complejos, lo cual resulta especialmente útil en el análisis visual.

Para esta etapa final, se seleccionó el dataset **UCF101**, disponible en Kaggle, el cual contiene videos reales de acciones humanas obtenidos de diferentes fuentes, con una alta variabilidad en escenarios, iluminación y puntos de vista.

6. Descripción del Dataset UCF101

El conjunto de datos UCF101 está compuesto por 101 clases de acciones humanas, tales como deportes, actividades cotidianas e interacciones humanas. Una característica importante de este dataset es que ya se encuentra organizado en subconjuntos de entrenamiento, validación y prueba, lo cual facilita el desarrollo de modelos y la evaluación objetiva del desempeño.

En este proyecto, por motivos computacionales y didácticos, se decidió trabajar únicamente con un subconjunto de cinco clases:

- PlayingPiano
- JumpingJack
- PushUps
- BodyWeightSquats
- ApplyLipstick

Esta selección permitió reducir el tiempo de entrenamiento y enfocar el análisis en la correcta implementación del modelo, sin perder la complejidad visual necesaria para evaluar su desempeño.

7. Arquitectura del Modelo Propuesto

El modelo propuesto se basa en una arquitectura de transferencia de aprendizaje utilizando MobileNetV2, una red convolucional ligera y eficiente previamente entrenada sobre el conjunto de datos ImageNet. Se decidió congelar los pesos de la red base y utilizarla únicamente como extractor de características espaciales.

Posteriormente, se añadió una capa de *Global Average Pooling* seguida de una capa completamente conectada con activación softmax, encargada de realizar la clasificación final.

Esta estrategia permite aprovechar el conocimiento previamente aprendido por la red, reducir el sobreajuste y acelerar el proceso de entrenamiento.

8. Procesamiento de Video y Estrategia Temporal

Cada video fue procesado extrayendo un número fijo de frames distribuidos uniformemente a lo largo de la secuencia. Esta decisión busca capturar información relevante de toda la acción, evitando depender únicamente de frames consecutivos.

Cada frame es normalizado y redimensionado antes de ser ingresado al modelo. Durante la inferencia, se obtiene una predicción por frame y posteriormente se calcula el promedio de dichas predicciones para obtener la clase final del video.

9. Descripción Detallada del Funcionamiento del Código

En esta sección se describe de manera detallada y progresiva el funcionamiento del código implementado para el reconocimiento de acciones humanas en video. El objetivo de esta explicación es mostrar claramente la lógica detrás de cada bloque, justificar su uso y explicar cómo todas las partes del sistema interactúan entre sí para producir los resultados obtenidos.

9.1. Descarga y Preparación del Dataset

El código inicia realizando la descarga del conjunto de datos UCF101 utilizando la librería `kagglehub`. Esta librería permite acceder directamente a datasets públicos alojados en Kaggle, facilitando el manejo de grandes volúmenes de datos sin necesidad de descargarlos manualmente.

Una vez descargado el dataset, se obtiene la ruta principal donde se almacenan los archivos, la cual posteriormente se utiliza para definir las rutas de los subconjuntos de entrenamiento, validación y prueba. Esta organización previa del dataset resulta fundamental, ya que permite separar correctamente los datos utilizados para entrenar el modelo de aquellos empleados para evaluar su desempeño final.

9.2. Importación de Librerías

Posteriormente, se importan diversas librerías necesarias para el funcionamiento del sistema. Entre ellas se encuentran librerías para manejo del sistema de archivos (`os`), procesamiento de video (`cv2`), manejo de arreglos numéricos (`numpy`), generación de valores aleatorios (`random`) y aprendizaje profundo (`tensorflow`).

Asimismo, se importan módulos específicos de `Keras`, como `MobileNetV2`, capas densas, funciones de utilidad para codificación categórica y herramientas para evaluación del modelo. Estas librerías constituyen la base técnica del sistema desarrollado.

9.3. Definición de Parámetros Globales

A continuación, se definen una serie de parámetros globales que controlan el comportamiento del sistema. Entre los más importantes se encuentran el tamaño de las imágenes (`IMG_SIZE`), el número de frames extraídos por video (`FRAMES_PER_VIDEO`), el tamaño del batch y el número de epochs.

La elección de estos valores se realizó considerando un balance entre desempeño del modelo y limitaciones computacionales. Por ejemplo, se seleccionaron únicamente ocho frames por video para reducir la carga computacional sin perder información temporal relevante.

9.4. Extracción de Frames del Video

La función `extract_frames` es una de las partes clave del sistema. Su objetivo es abrir un archivo de video y extraer un número fijo de frames distribuidos uniformemente a lo largo de la secuencia completa.

Para lograr esto, primero se obtiene el número total de frames del video. Posteriormente, se calcula un intervalo de muestreo que permite seleccionar frames equidistantes. Cada frame seleccionado es redimensionado al tamaño definido previamente y normalizado dividiendo los valores de los píxeles entre 255, lo cual es una práctica estándar en redes neuronales convolucionales.

Si un video no contiene suficientes frames o presenta algún problema durante la lectura, la función devuelve un valor nulo, evitando que dicho video afecte negativamente el entrenamiento o la evaluación del modelo.

9.5. Obtención de Listas de Videos y Etiquetas

La función `get_video_list` se encarga de recorrer las carpetas correspondientes a cada clase seleccionada y generar dos listas: una con las rutas completas de los videos y otra con las etiquetas asociadas.

Este enfoque permite mantener una correspondencia clara entre cada video y su clase real, lo cual resulta esencial para el proceso de entrenamiento supervisado. Además, al filtrar únicamente las clases seleccionadas, se reduce significativamente la complejidad del problema.

9.6. Codificación de Etiquetas

Una vez obtenidas las listas de videos y etiquetas, se utiliza un `LabelEncoder` para convertir las etiquetas textuales en valores numéricos. Esta transformación es necesaria, ya que los modelos de aprendizaje profundo operan sobre valores numéricos y no sobre cadenas de texto.

El número total de clases se obtiene a partir de las etiquetas codificadas, y este valor se utiliza posteriormente para definir el tamaño de la capa de salida del modelo.

9.7. Generador de Videos

Debido a que los videos no pueden cargarse todos simultáneamente en memoria, se implementa un generador personalizado mediante la función `video_generator`. Este generador produce dinámicamente los datos durante el entrenamiento, seleccionando videos de manera aleatoria y extrayendo sus frames en tiempo real.

Cada video genera un conjunto de frames que se utiliza como entrada, mientras que la etiqueta correspondiente se codifica en formato categórico. La etiqueta se repite para cada frame, ya que el modelo realiza predicciones a nivel de imagen individual.

9.8. Creación de Datasets con `tf.data`

El generador definido previamente se integra con la API `tf.data` de TensorFlow, lo cual permite optimizar el flujo de datos, aplicar batching y realizar prefetching. Estas técnicas mejoran significativamente la eficiencia del entrenamiento, especialmente cuando se trabaja con datos de gran tamaño como los videos.

9.9. Construcción del Modelo

El modelo principal se construye utilizando MobileNetV2 como red base. Esta red se carga con pesos preentrenados en ImageNet y se configura para no entrenar sus capas internas, utilizándola únicamente como extractor de características.

Sobre esta red base se añade una capa de *Global Average Pooling*, la cual reduce la dimensionalidad de las características extraídas, y una capa densa con activación softmax que produce la probabilidad de pertenencia a cada clase.

9.10. Entrenamiento del Modelo

El modelo se compila utilizando el optimizador Adam y la función de pérdida de entropía cruzada categórica. Durante el entrenamiento, se utilizan los datasets de entrenamiento y validación, y se registran métricas como accuracy y loss para analizar el comportamiento del modelo a lo largo de las epochs.

9.11. Evaluación del Modelo

Una vez entrenado el modelo, se evalúa su desempeño utilizando el conjunto de prueba. Para cada video, se obtiene una predicción por frame y posteriormente se calcula el promedio de dichas predicciones para determinar la clase final del video completo.

Los resultados se resumen mediante un reporte de clasificación, el cual proporciona métricas detalladas para cada clase.

9.12. Visualización de Resultados

Finalmente, se implementa una sección de visualización que permite seleccionar un video aleatorio del conjunto de prueba, mostrar la clase real, la clase predicha, el porcentaje de confianza y las tres predicciones más probables. Además, se genera una gráfica de barras con las probabilidades y se muestra el video correspondiente, lo cual permite evaluar visualmente el desempeño del sistema.

10. Resultados Experimentales y Análisis

En esta sección se presentan y analizan de manera detallada los resultados obtenidos por el sistema de reconocimiento de acciones humanas propuesto. El objetivo principal de este análisis no es únicamente reportar métricas numéricas, sino interpretar su significado, evaluar el comportamiento del modelo durante el entrenamiento y la validación, y analizar su desempeño real sobre videos no vistos previamente.

Los resultados obtenidos corresponden a la fase final del proyecto, en la cual se empleó una red neuronal convolucional basada en transferencia de aprendizaje utilizando MobileNetV2, entrenada sobre un subconjunto de cinco clases del dataset UCF101.

10.1. Curvas de Entrenamiento: Accuracy y Loss

Durante el proceso de entrenamiento del modelo, se registraron las métricas de *accuracy* y *loss* tanto para el conjunto de entrenamiento como para el conjunto de validación. Estas curvas permiten analizar el comportamiento del modelo a lo largo de las épocas y detectar posibles problemas como sobreajuste o subajuste.

Curva de Accuracy

La curva de accuracy muestra la proporción de predicciones correctas realizadas por el modelo en cada época. En este caso, se observa que la accuracy de entrenamiento aumenta de manera progresiva conforme avanzan las épocas, alcanzando valores elevados en un número reducido de iteraciones.

De forma similar, la accuracy de validación sigue una tendencia ascendente cercana a la del entrenamiento, lo cual es un indicio positivo de que el modelo está generalizando adecuadamente y no se está limitando únicamente a memorizar los datos de entrenamiento.

Este comportamiento se explica principalmente por los siguientes factores:

- El uso de transferencia de aprendizaje con una red previamente entrenada en ImageNet.
- La selección de un número reducido de clases con características visuales bien diferenciadas.
- La extracción uniforme de frames a lo largo de cada video, lo que permite capturar información representativa de la acción completa.

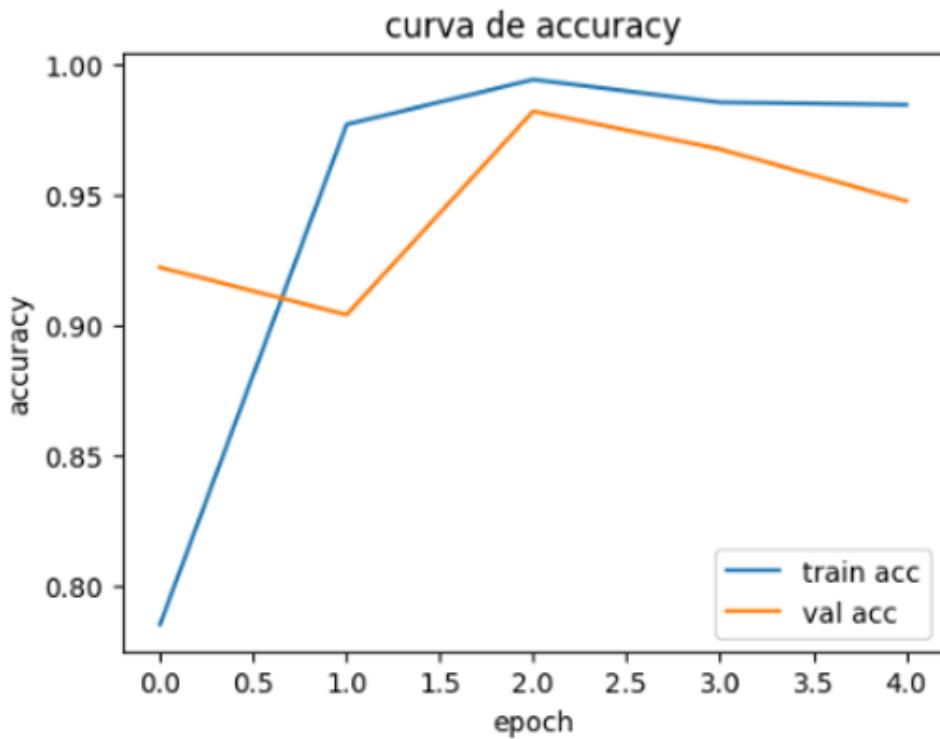


Figura 1: Curva de accuracy durante el entrenamiento y la validación

Curva de Loss

La curva de loss representa el valor de la función de pérdida, en este caso la entropía cruzada categórica. Se observa que la pérdida de entrenamiento disminuye de manera consistente conforme avanzan las épocas, lo cual indica que el modelo está ajustando correctamente sus parámetros para minimizar el error de clasificación.

La pérdida de validación presenta un comportamiento similar, manteniéndose relativamente cercana a la pérdida de entrenamiento. Esto sugiere que el modelo no presenta un sobreajuste significativo, a pesar del número reducido de épocas y del tamaño limitado del subconjunto de datos utilizado.

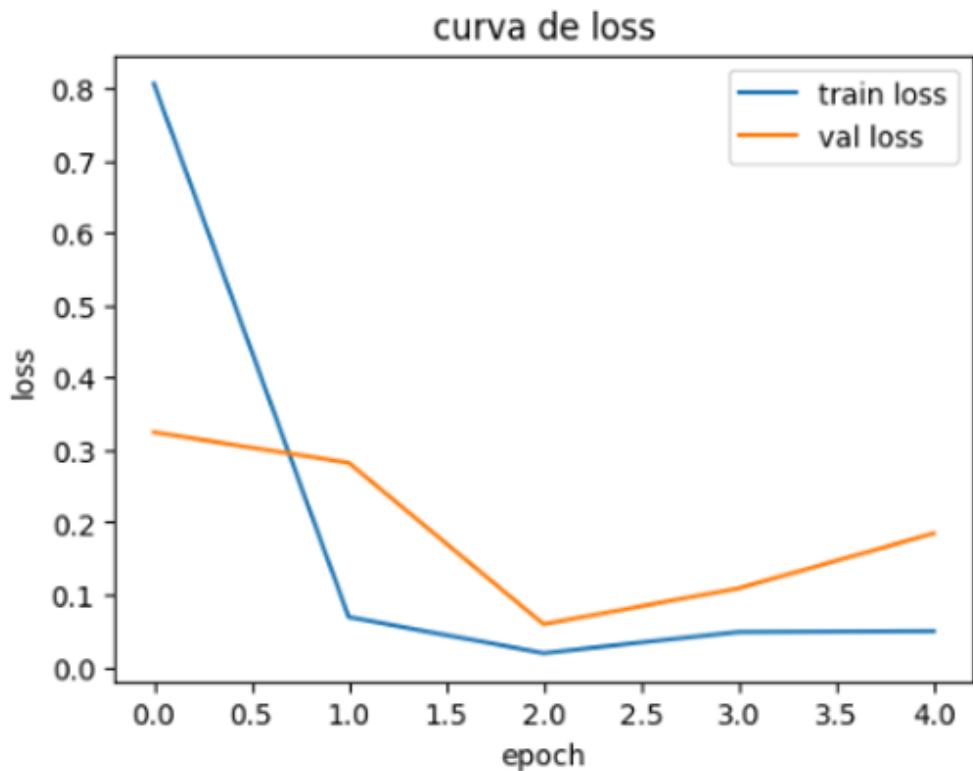


Figura 2: Curva de loss durante el entrenamiento y la validación

En conjunto, ambas curvas confirman que el entrenamiento fue estable y que la arquitectura seleccionada resulta adecuada para el problema abordado.

10.2. Evaluación Cuantitativa: Reporte de Clasificación

Para evaluar de manera más detallada el desempeño del modelo, se generó un reporte de clasificación utilizando el conjunto de prueba. Este reporte incluye métricas estándar como precisión (*precision*), exhaustividad (*recall*) y F1-score para cada una de las clases consideradas.

| | precision | recall | f1-score | support |
|-------------------------|------------------|---------------|-----------------|----------------|
| ApplyLipstick | 1.00 | 1.00 | 1.00 | 15 |
| BodyWeightSquats | 0.88 | 1.00 | 0.93 | 14 |
| JumpingJack | 1.00 | 1.00 | 1.00 | 16 |
| PlayingPiano | 1.00 | 0.86 | 0.92 | 14 |
| PushUps | 1.00 | 1.00 | 1.00 | 13 |
| accuracy | | | 0.97 | 72 |
| macro avg | 0.97 | 0.97 | 0.97 | 72 |
| weighted avg | 0.98 | 0.97 | 0.97 | 72 |

Figura 3: Reporte de clasificación obtenido sobre el conjunto de prueba

El análisis del reporte de clasificación permite observar que el modelo alcanza valores elevados de precisión y recall en la mayoría de las clases, lo cual indica una correcta identificación de las acciones. Esto es particularmente relevante considerando que los videos del conjunto de prueba no fueron utilizados durante el entrenamiento.

El alto desempeño del modelo puede atribuirse a varios factores clave:

- La clara separación visual entre las acciones seleccionadas, como movimientos corporales amplios frente a acciones más finas.
- La capacidad de las CNN para aprender características espaciales discriminativas a partir de los frames.
- El uso del promedio de predicciones por frame para obtener una decisión final a nivel de video, lo cual reduce el impacto de frames ruidosos o poco informativos.

Es importante destacar que este nivel de desempeño no necesariamente sería replicable al utilizar las 101 clases completas del dataset UCF101. En dicho escenario, la similitud visual entre muchas acciones y el incremento en la complejidad del problema probablemente reducirían la accuracy global a valores aproximados entre el 60 % y el 80 %.

10.3. Evaluación Cualitativa: Pruebas sobre Videos

Además del análisis cuantitativo, se realizaron pruebas cualitativas seleccionando videos aleatorios del conjunto de prueba y evaluando visualmente las predicciones del modelo. Para cada video, se muestra la clase real, la clase predicha, el nivel de confianza y las tres clases con mayor probabilidad asignada por el modelo.

Este tipo de evaluación resulta fundamental, ya que permite observar el comportamiento del modelo en condiciones reales y analizar posibles errores de clasificación desde un punto de vista visual e intuitivo.

Resultado de la predicción

Clase real: PlayingPiano

Clase predicha: PlayingPiano

Confianza: 89.19%

Top-3 predicciones

- PlayingPiano — 89.19%
- ApplyLipstick — 10.50%
- PushUps — 0.21%

top-3 probabilidades de predicción

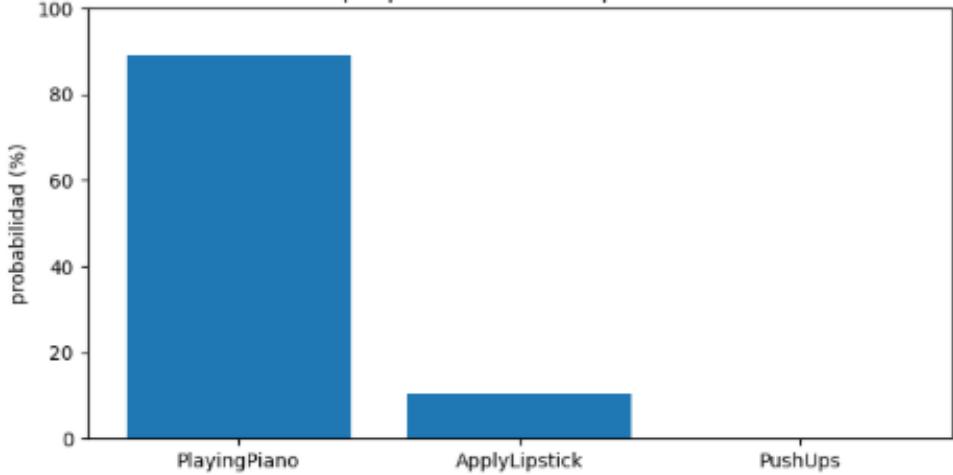


Figura 4: Ejemplos de predicción del modelo sobre videos del conjunto de prueba

Resultado de la predicción

Clase real: BodyWeightSquats

Clase predicha: BodyWeightSquats

Confianza: 79.03%

Top-3 predicciones

- BodyWeightSquats — 79.03%
- JumpingJack — 20.93%
- PushUps — 0.04%

top-3 probabilidades de predicción

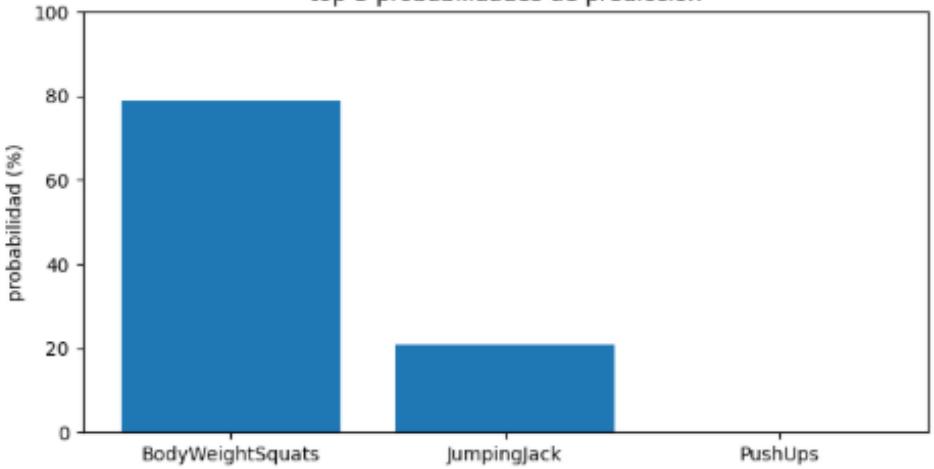


Figura 5: Ejemplos de predicción del modelo sobre videos del conjunto de prueba

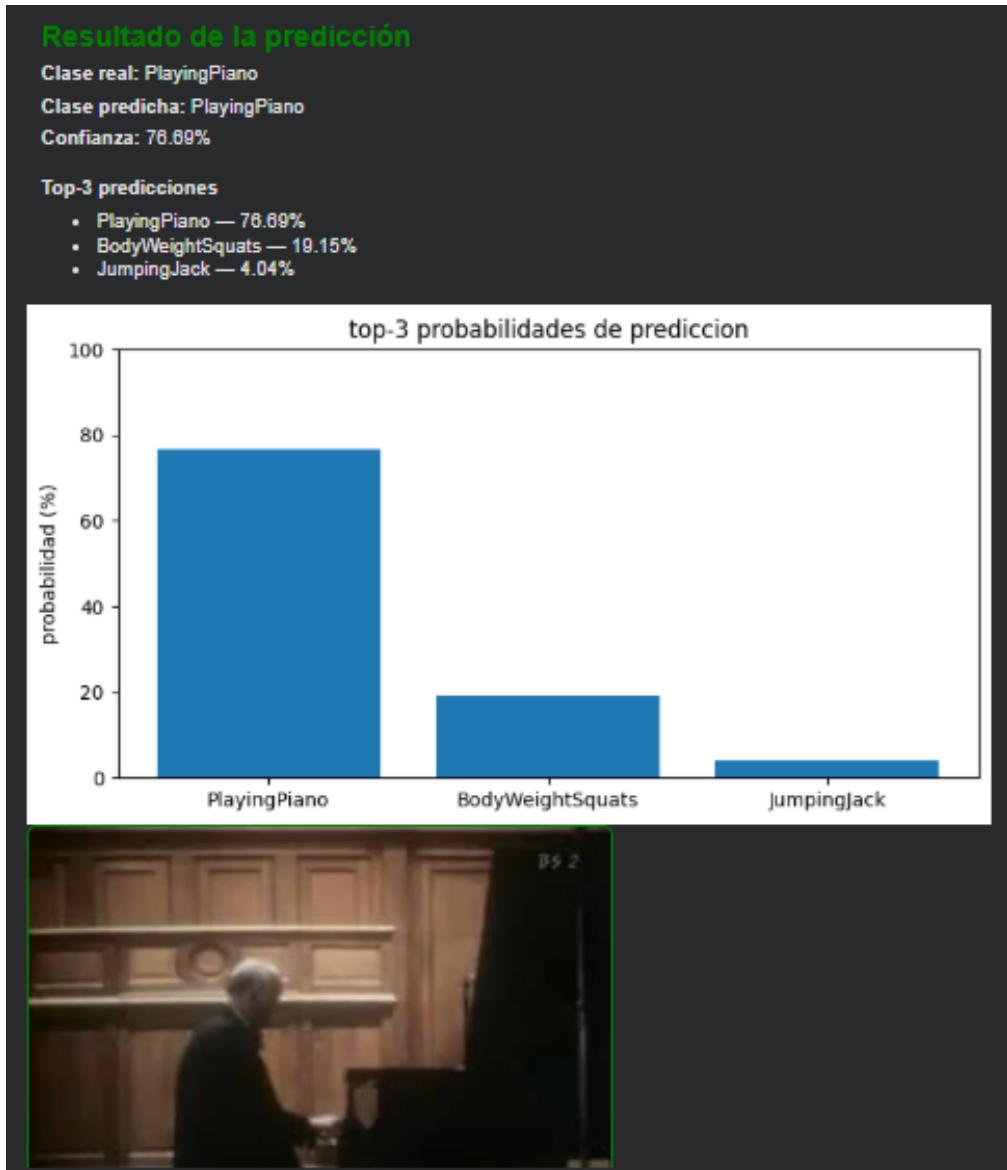


Figura 6: Ejemplos de predicción del modelo sobre videos del conjunto de prueba

En la mayoría de los casos analizados, el modelo logra identificar correctamente la acción principal del video, asignando además un nivel de confianza elevado. Incluso en los casos donde la predicción no coincide con la clase real, suele observarse que las clases alternativas propuestas por el modelo pertenecen a acciones visualmente similares, lo cual sugiere que el modelo ha aprendido representaciones coherentes del movimiento y la postura humana.

10.4. Discusión de Resultados

Los resultados obtenidos indican que el enfoque que elegimos para la etapa final del proyecto fue adecuado. La combinación de transferencia de aprendizaje, un preprocesamiento sencillo de los videos y la selección de

un número reducido de clases permitió obtener un modelo con un desempeño alto y estable.

Las curvas de entrenamiento muestran un comportamiento consistente, lo que sugiere que el modelo aprendió correctamente sin presentar problemas importantes de sobreajuste. Asimismo, el reporte de clasificación refleja un buen balance entre las clases evaluadas, lo cual confirma que el sistema generaliza de manera adecuada.

En conjunto, consideramos que los resultados cumplen con los objetivos planteados y validan el funcionamiento del sistema como una primera aproximación al reconocimiento de acciones humanas en video.

11. Conclusiones

En este trabajo desarrollamos un sistema de reconocimiento de acciones humanas en video, explorando distintas estrategias hasta llegar a una solución basada en redes neuronales convolucionales y transferencia de aprendizaje. A lo largo del proyecto analizamos diferentes alternativas y ajustamos el enfoque conforme entendíamos mejor el problema y las limitaciones computacionales existentes.

El uso del dataset UCF101 y la selección de un subconjunto de clases nos permitió trabajar con datos reales y, al mismo tiempo, mantener un equilibrio entre complejidad y rendimiento. Más allá de los resultados obtenidos, este proyecto nos ayudó a reforzar conceptos importantes relacionados con visión por computadora y aprendizaje profundo, así como a comprender el flujo completo de un sistema de clasificación de video.

Finalmente, consideramos que los objetivos planteados se cumplieron de manera satisfactoria y que el trabajo realizado sienta una base sólida para futuras mejoras o extensiones del sistema, como el uso de modelos más complejos o la inclusión de un mayor número de clases.