MODELO PARA LA DETECCIÓN AUTOMÁTICA DE LA ACTIVIDAD HUMANA: "CORRER" USANDO REDES NEURONALES

MODEL TO THE AUTOMATIC DETECTION OF HUMAN ACTIVITY: "TO RUN" USING NEURAL NETWORKS

Eduardo López Méndez
Universidad Politécnica de Puebla, México
elopezm20@gmail.com

Jorge de la Calleja Mora
Universidad Politécnica de Puebla, México
jorge.delacalleja@uppuebla.edu.mx

Hugo Jair Escalante
Instituto Nacional de Astrofísica Óptica y Electrónica Puebla, México
hugojair@inaoep.mx

José David Alanís Urquieta
Universidad Tecnológica de Puebla, México
david.alanis@utpuebla.edu.mx

Paulo Daniel Vázquez Mora
Universidad Tecnológica de Puebla, México
daniel.vazquez@utpuebla.edu.mx

Mariel Pamela Morales Riveroll
Universidad Politécnica de Puebla, México
marymori28@gmail.com

Recepción: 3/noviembre/2021 **Aceptación:** 7/marzo/2022

Resumen

En este trabajo se presenta la implementación de un modelo entrenado base, capaz de detectar automáticamente acciones corporales; específicamente "correr", mediante el uso de redes neuronales. La implementación se construye en cinco etapas principales que son: investigación de metodologías previas, entrenamiento del modelo, diseño e implementación de una base de datos para los resultados, desarrollo de una interfaz para la administración de los resultados de la detección, finalizando con las pruebas de manera integral.

Este prototipo planteado partirá del uso de un par de redes neuronales reentrenadas; uno para la clasificación de frames del video, y el segundo es temporal, que toma en cuenta la cantidad de frames y el tiempo analizado dando como resultado la clasificación de los frames detectados como “Correr” o no.
Dados los resultados obtenidos, este modelo puede ser reentrenado para utilizarse en diversos ámbitos, pero principalmente en tareas de videovigilancia.
Palabras Clave: Detección automática, inteligencia artificial, redes neuronales convolucionales, video vigilancia, visión por computadora.

Abstract

In this work the implementation a base training model is presented, it is capable of automatically to detect corporal actions, specifically “to run”, by the use of neural networks. The implementation is built in five main stages that are: research of previous methodology, workout of the model, design and implementation of a data base for the results, development of an interface for the administration of the results of the detection, finalizing with the tests in an integral way.
This proposed prototype will start from the use of a couple of re trained neural network, one for the classification of frames of the video, and the second is temporal, that takes into account the quantity of frames and the analyzed time, giving as result the classification of the detected frames as “to run” or not.
Given the obtained results, this model can be re trained in order to be used in diverse scopes, but mainly in video surveillance tasks.
Keywords**:** *Artificial intelligence, automatic detection, convolutional neural networks, detection, video Surveillance, vision by computing.*

1. Introducción

Hace tiempo que la inteligencia artificial (IA) pasó de la ciencia ficción a formar parte de nuestra vida cotidiana. Desde controles de seguridad, hasta las compras por medio de una sonrisa [Prevent Security Systems, 2019].
El uso de la IA aplicado a tareas de video para simplificar procesos cotidianos ha crecido exponencialmente. Se prevén 1,000 millones de cámaras de video

conectadas a plataformas de inteligencia artificial hasta el 2020 [Secure Week, 2019].

La conexión de cámaras a sistemas de IA tiene innumerables funciones orientadas a tareas de seguridad y vigilancia, control de calidad, logística, entre otras, siendo una parte fundamental el monitoreo y administración de los videos, donde la aplicación de sistemas inteligentes permite mejorar la exactitud y costos.

Esto se debe a que a diferencia de los humanos, que perdemos entre el 50% y el 90% la percepción visual una vez transcurridos 20 minutos de supervisión continua, la capacidad de análisis de un sistema inteligente es constante [Prevent Security Systems, 2018]. Una rama de la IA son las redes neuronales artificiales las cuales han ido evolucionando durante casi ya 80 años, esta evolución la podemos dividir en 3 etapas:

- Primera etapa: de los años 40 a los años 70 en donde surgen estos nuevos modelos pasando de causar un gran asombro, al escepticismo total.
- Segunda etapa: En los años 80's, después de 10 años en el hielo, surgen mejoras como lo es la propagación hacia atrás (Back propagation) pero la falta de poder de cómputo formó una barrera infranqueable.
- Para el año 2006 que se logra superar esa barrera gracias a las GPU surgiendo de esta manera lo que hoy se conoce como aprendizaje profundo (Deep Learning) dando capacidad casi ilimitada a estas redes [Prevent Security Systems, 2018].

Para el análisis de acciones no basta con la obtención y comparación de características, sino que debemos considerar el tiempo como un factor indispensable en la citada detección, por lo que se requiere hacer el análisis de un conjunto de frames consecutivos que pueden o no presentar la acción deseada.

Para realizar estas tareas existen redes neuronales que consisten en unas celdas de memoria que permiten a la red recordar valores por períodos cortos o largos [Bagnato, 2018].

La aplicación del aprendizaje profundo para el análisis de videos es un tema relativamente reciente, desde el reconocimiento de patrones con redes en

transferencia de aprendizaje (Transfer Learning), hasta el uso de redes neuronales que factorizan el espacio y el tiempo [Martín, 2018].

Por ejemplo, en [Lopes, 2012] se usa la transferencia de aprendizaje para abordar la tarea del reconocimiento de la acción.

De la misma forma se aplica este enfoque en [Yaranga, 2018], realizando la detección de conductores distraídos mediante el análisis de imágenes extraídas de un video (frames) y la red de extracción de características VGG16, obteniendo una eficiencia de entrenamiento y validación del 99.3% y 99.46% respectivamente.

En seguida tenemos que en [Cano, 2015] se plantea la detección de la actividad realizada por las personas en el video, dividiendo las actividades en cinco grupos:

- Interacción humana – humano
- Interacción humana – objeto
- Deportes
- Movimientos corporales
- Personas tocando diversos objetos

En el año 2015, se llevó a cabo la "*28th The IEEE International Conference on Computer Vision and Pattern Recognition*", donde en [Du, 2015] se realizó la clasificación de las acciones humanas, a través de reconocer la trayectoria de las articulaciones esqueléticas como alternativa al reconocimiento de la dinámica del movimiento, en este caso se llevó a cabo el entrenamiento de un modelo de red neuronal recurrente (RNN).

El trabajo antes mencionado propone un modelo que divide el esqueleto en cinco partes con las cuales se alimenta a cinco subredes, cuyas salidas se fusionan de forma jerárquica para alimentar capas de convolución superiores. Las representaciones finales se introducen a un perceptrón el cual solo posee una capa que tiene por salida la decisión final del reconocimiento.

Posteriormente en ese mismo año, durante la "*The IEEE International Conference on Computer Vision*" se dio a conocer un trabajo en el cual se veían a las acciones humanas como señales espacio – temporales en 3D incluyendo no sólo las características visuales, sino también la dinámica propia del movimiento. Hasta ese

momento las redes neuronales convolucionales habían tenido gran éxito en la clasificación de imágenes, pero debido a la alta complejidad de realizar una convolución en 3D, se había encontrado con una barrera la cual solo permitía un avance limitado. En [Sun, 2015] se plantea una arquitectura capaz de manejar señales 3D de manera más efectiva, proponiendo el uso de una red neuronal espaciotemporal factorizada (FstCN) probando el modelo con un conjunto de videos de uso común como el Data Set UCF-101 sin videos auxiliares que mejoren el rendimiento. Finalmente para el año 2017 en la ciudad de Lima [Fernández, 2017] aplica una combinación de dos arquitecturas de redes neuronales para la identificación de acciones humanas, una red pre entrenada que realizará la extracción de características de cada uno de los frames del video y una red espacio temporal o Long Short Term Memory (LSTM) capas de clasificar no solo por las características de los frames si no también tomando en cuenta el tiempo, es decir, la sucesión de frames que forman la acción, de igual manera se utilizaron videos básicos obtenidos del DataSet UCF-101 y Virat 2.0.

El objetivo del presente proyecto es el desarrollo de un sistema integral de monitoreo de video para la detección automática de la acción humana “Correr” aplicando técnicas de aprendizaje profundo supervisado, particularmente usando una red neuronal profunda y otra recurrente.

En la segunda sección, se presentan los métodos realizados para el cumplimiento del objetivo planteado. En la tercera sección se describen los resultados obtenidos de los modelos que enriquecen la detección. En la cuarta sección se realiza una discusión o justificación de los métodos y resultados obtenidos. Finalmente, la última sección versa acerca las conclusiones, y aportaciones del trabajo, así como los trabajos futuros

2. Métodos

De forma general el proyecto se divide en tres etapas, etapa experimental para seleccionar el algoritmo adecuado para la detección de las acciones, implementación y reentrenamiento con los dataset etiquetados como “correr” o “no correr”, finalmente la etapa de integración donde se diseñó y desarrollo una interfaz

y una base de datos para la administración y almacenamiento de los resultados, ver figura 1.

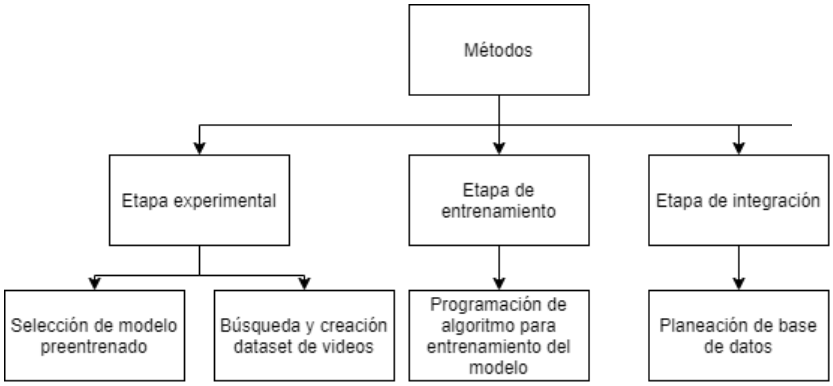


Figura 1 Etapas del proyecto.

Etapa experimental

La detección de la acción humana correr se desarrolla en dos en dos fases la primera encardada de caracterizar cada frame tratado como imagen y para posteriormente realizar su clasificación a partir de las características obtenidas de un conjunto de frames tomando en cuenta el tiempo como una tercera dimensión. Para sostener la primera fase de la detección se optó por utilizar un modelo de clasificación de frames de video, la selección del modelo y la comparación cualitativa enfocándonos directamente en el uso de redes neuronales. En la tabla 1 se enlista las principales redes neuronales y una breve descripción de ellas.

Tabla 1 Clasificación de las redes neuronales.

Modelo	Descripción
Redes neuronales convolucionales (CNN) [Calvo, 2017]	Red multicapa que consta de capas convolucionales y de reducción alternadas, y al final capas totalmente conectadas.
Redes neuronales recurrentes (RNN) [Bargnato, 2018]	A diferencia de las redes convolucionales clásicas las RNN permiten conexiones "hacia atrás" entre las capas. Esto las hace buenas para procesar datos de tipo "time series"
Redes neuronales pre entrenadas [Bagnato, 2018]	Redes utilizadas para una tarea y reutilizadas para su aplicación en otro dominio, mediante una técnica de aprendizaje profundo conocida como Transfer Learning [Martín, 2018]
Redes neuronales espacio temporales (LSTM) [Bargnato, 2018]	Redes recurrentes que constan de unas celdas de memoria que permiten a la red recordar valores por períodos cortos o largos.

Redes neuronales convolucionales (CNN)

Las redes neuronales convolucionales son muy similares a las redes neuronales ordinarias, se componen de neuronas que tienen pesos y sesgos que pueden aprender. Cada neurona recibe algunas entradas, realiza un producto escalar y luego aplica una función de activación; el cual contiene una función de pérdida o costo sobre la última capa, la cual estará totalmente conectada. Lo que diferencia a las redes neuronales convolucionales, es que suponen explícitamente que las entradas son imágenes, lo que nos permite codificar ciertas propiedades en la arquitectura; permitiendo ganar en eficiencia y reducir la cantidad de parámetros en la red, figura 2. De esta forma las redes neuronales convolucionales son capaces de modelar complejas variaciones y comportamientos dando predicciones bastantes precisas [Bagnato,2018].

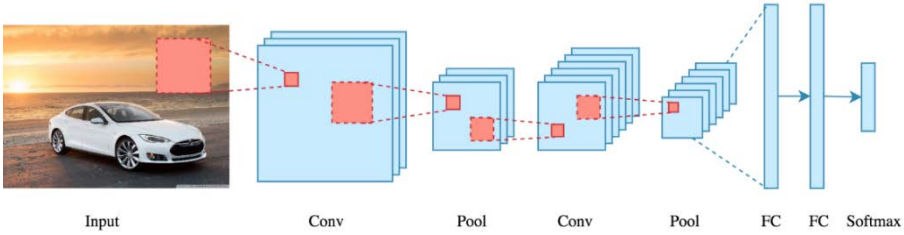


Figura 2 Estructura redes neuronales convolucionales (CNN).

Inception V3

Es un modelo de reconocimiento de imágenes desarrollado por Google, figura 3. El cual se caracteriza por su alto rendimiento de clasificación y fácil accesibilidad dado que se encuentra incluido en la biblioteca de código abierto Tensorflow, el cual se ha demostrado que alcanza una exactitud superior al 78.1% en el conjunto de datos de ImageNet, tabla 2 [Google Cloud, 2021].

Este prototipo está formado por bloques de construcción simétricos y asimétricos que incluyen convoluciones, reducción promedio, reducción máxima, concatenaciones, retirados y capas completamente conectadas. La normalización por lotes se usa con frecuencia en todo el modelo y se aplica a las entradas de activación. Las pérdidas se calculan a través de Softmax.

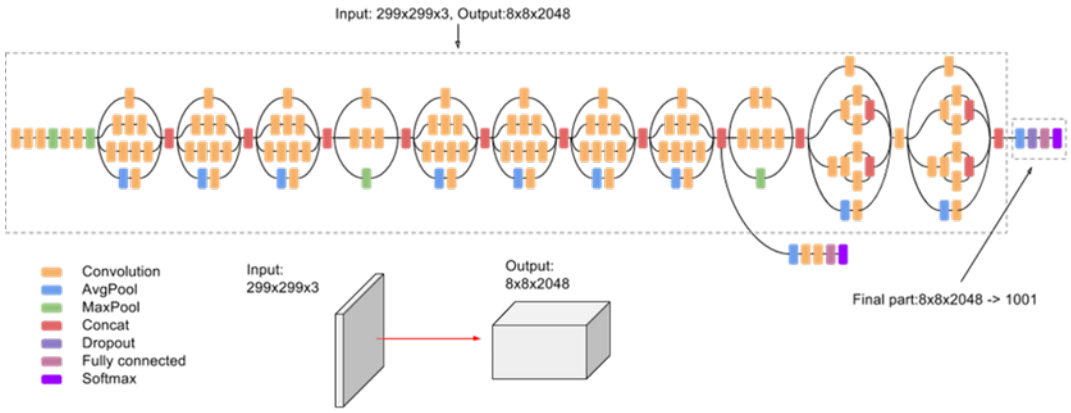


Figura 3 Estructura Inception V3 [Google Cloud, 2021].

Tabla 2 Comparación de precisión de acuerdo con el tamaño de las imágenes de entrada.

Tamaño imágenes	Precisión
79 x 79	75.2%
151 x 151	76.4%
299 x 299	76.6%

Long Short-Term Memory (LSTM)

Además, para la clasificación de acciones humanas requerimos el manejo de las convoluciones en 3D para lo cual se optó por utilizar redes neuronales Long Short-Term Memory (LSTM), figura 4.

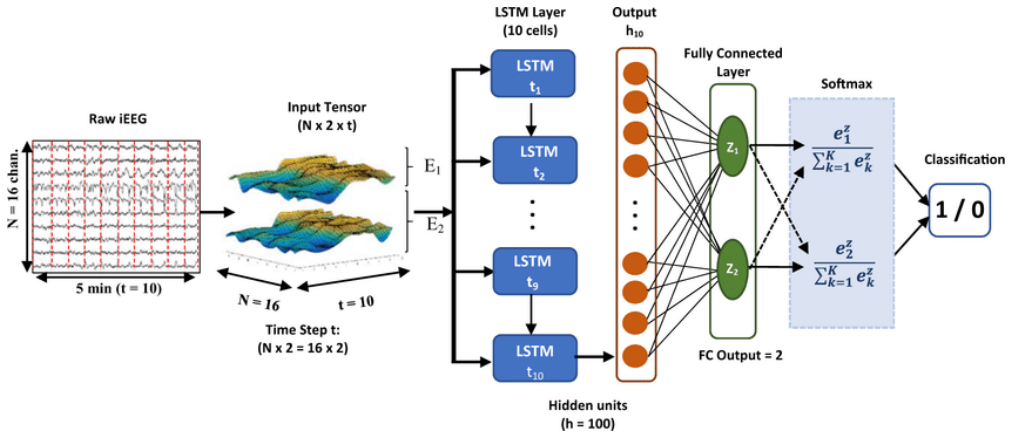


Figura 4 Estructura LSTM.

Estas redes son un tipo de redes neuronales recurrentes (RNN). Toma secuencias de información y utiliza mecanismos recurrentes y técnicas de puerta. LSTM tiene

muchas ventajas sobre otras NN recurrentes y de retroalimentación en el modelado de series de tiempo, como audio y video [González, 2018].

Creación de DataSet de videos para el entrenamiento

Para el entrenamiento de los modelos se creó un DataSet etiquetado, a partir de videos obtenidos de repositorios abiertos de internet, y videos grabados de personas corriendo y se hizo una comparación contra personas caminando y así tener un punto de referencia para las clasificaciones, figura 5.

A los videos que forman parte del DataSet se le aplica técnicas de computer vision como reducción de ruido, cada video se dividió en videos cortos de aproximadamente 15 frames, para obtener los frames característicos de la acción significativa de correr, eliminando así los frames en blanco

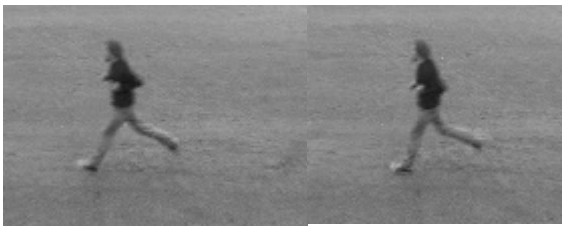


Figura 5 Ejemplo de frames del DataSet.

Etapa de entrenamiento

Una vez seleccionado los algoritmos y creado el DataSet de videos se realiza el entrenamiento de los modelos. En primer lugar, para el reentrenamiento del modelo Inspección V3, se consideran los pesos y capas del modelo original, omitiendo la última capa encargada de la clasificación de las imágenes, ya que el objetivo de este modelo únicamente es caracterizar cada uno de los frames que componen el video a clasificar, este reentrenamiento se realizó con ayuda del DataSet de imágenes ImageNet, figura 6 [Stanford University, 2020].

```
base_model = InceptionV3(weights='imagenet')
model = Model(input=base_model.input, output=base_model.get_layer('avg_pool').output)
```

Figura 6 Ejemplo del reentrenamiento modelo Ispection V3 sin la capa de clasificación.

En segundo lugar, para realizar el reentrenamiento de la red neuronal recurrente dedicada a la clasificación de videos en dos posibilidades: Personas corriendo y personas no corriendo. La figura 7, muestra las capas que componen la red:

```
chuck_size = 2048
n_chucks = 15
rnn_size = 512
model1 = Sequential()
model1.add(LSTM(rnn_size,input_shape=(15,chuck_size)))
model1.add(Dense(1024))
model1.add(Activation('relu'))
model1.add(Dense(50))
model1.add(Activation('sigmoid'))
model1.add(Dense(2))
model1.add(Activation('softmax'))
model1.compile(loss='mean_squared_error', optimizer='adam', metrics=['accuracy'])
```

Figura 7 Ejemplo de la estructura utilizada para la red LSTM.

Las características requeridas para el entrenamiento de los modelos fueron:

- 128 Gb de RAM.
- 100 Tb de almacenamiento.
- Procesador Intel con un total de 64 núcleos.

Las capas utilizadas en los diferentes entrenamientos se muestran en tablas 3 y 4.

Tabla 3 Modelo 1 LSTM.

Modelo 1	
Red LSTM	Neuronas
Capa 1	1024 neuronas y Activación "relu"
Capa 2	50 neuronas, Activación "sigmoide"
Capa 3	2 neuronas, Activación "softmax" y Salida
Accuracy	98.36%

Tabla 4 Modelo 2 LSTM.

Modelo 2	
Red LSTM	Neuronas
Capa 1	1024 neuronas y Activación "relu"
Capa 2	512 neuronas y Activación "relu"
Capa 3	256 neuronas y Activación "relu"
Capa 4	128 neuronas y Activación "relu"
Capa 5	64 neuronas y Activación "relu"
Capa 6	32 neuronas y Activación "relu"
Capa 7	2 neuronas y Activación "softmax" Salida
Accuracy	89.7%

Etapa de integración

Diseño y desarrollo de una interfaz gráfica

Una vez entrenados los modelos procedemos a integrar el sistema completo. Por lo tanto, se procede a el diseño y desarrollo de una interfaz gráfica, para llevar a cabo la administración de los resultados al momento de detectar la acción buscada. Al ser un sistema desarrollado en Python, se busca el uso de un framework para el desarrollo de la interfaz resultando en una comparación cualitativa. En la tabla 5 se observa una comparativa de frameworks más populares de Python [López, 2019].

Tabla 5 Comparativa entre los 3 frameworks más populares de Python [López, 2019].

	Tkinter	**PyQt**	**wxPython**
Orientado a objetos	X	X	X
Licencia	Open Source	GPL, Comercial	Libre
Plataforma	Python nativo	C++	C++
Diseño	Simple	Moderno	Basado en widgets
Instalación	Incluido en Python	Externo	Externo
Documentación	Débil	Completa	Completa

Diseño e implementación de una base de datos

El punto fundamental del sistema es la detección de las actividades, pero para complementar su funcionalidad se registra un historial de acciones detectadas, debido a la poca la información a almacenar y teniendo pocos recursos de hardware el lenguaje seleccionado para su desarrollo es SQLite dado que no requiere de un servidor y es muy ligera al tratarse de una base de datos basada 100% en archivos, figura 8.

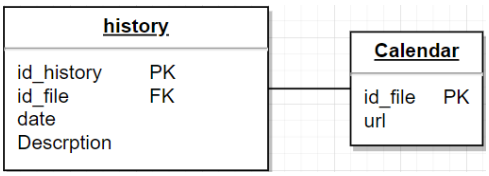


Figura 8 Base de datos para el historial de acciones.

3. Resultados

El modelo de red Inspección V3 se reentreno directamente respetando la estructura, con la excepción de que la capa para la clasificación de las imágenes

omitiendo la última capa encargada de la clasificación dejando paso al modelo de la figura 9.

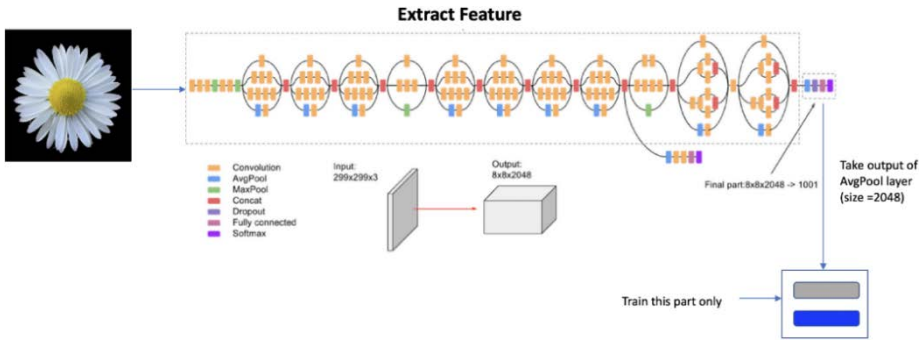


Figura 9 Modelo Inception V3.

Al combinar ambos modelos Incepcion V3 y LTSM, se completa el proceso de clasificación de los frames de los videos en la figura 10 se muestra el proceso final de caracterización y clasificación de las acciones.

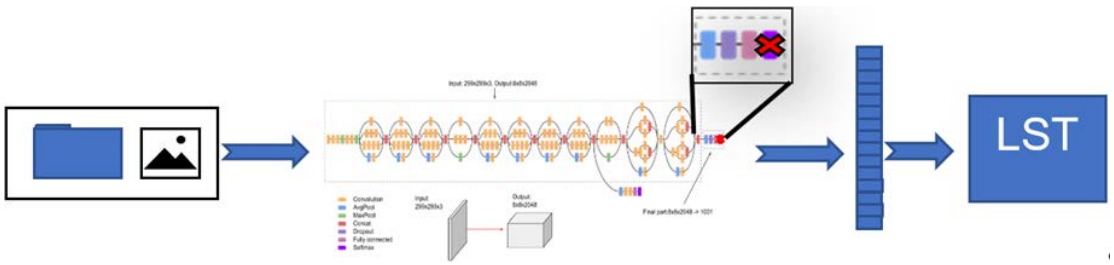


Figura 10 Modelo de todo el proceso de clasificación.

Para el DataSet de videos para el entrenamiento de los modelos, se utilizaron principalmente dos fuentes los DataSet UCF-101 y Virat 2.0 de donde se obtuvieron un total de 25 minutos de video continuo de personas corriendo y 25 minutos de personas caminando, adicional a esto 20 minutos de grabación de personas corriendo y 10 minutos de personas caminando ambos a una velocidad de 25 fps. como se muestra en la tabla 6.

El modelo LSTM cuenta con 3 capas teniendo en la primera capa 1024 neuronas en una capa de activación RELU, la segunda capa cuenta con 50 neuronas en una capa de activación sigmoide y por último una capa softmax de 2 neuronas en donde

la neurona que se activa da la salida indicando si es o no la actividad de correr, figura 11. A su vez en las tablas 7 y 8 se muestran los parámetros y resultados del entrenamiento de la red LSTM.

Tabla 6 DataSets de videos para el entrenamiento de los modelos.

	No. videos	**Duración**	**Velocidad**
Dataset	100 videos – Corriendo 100 videos – Caminando	25 minutos 25 minutos	25 fps.
Grabados	80 videos – Corriendo 80 videos – Caminando	20 minutos 15 minutos	25 fps.

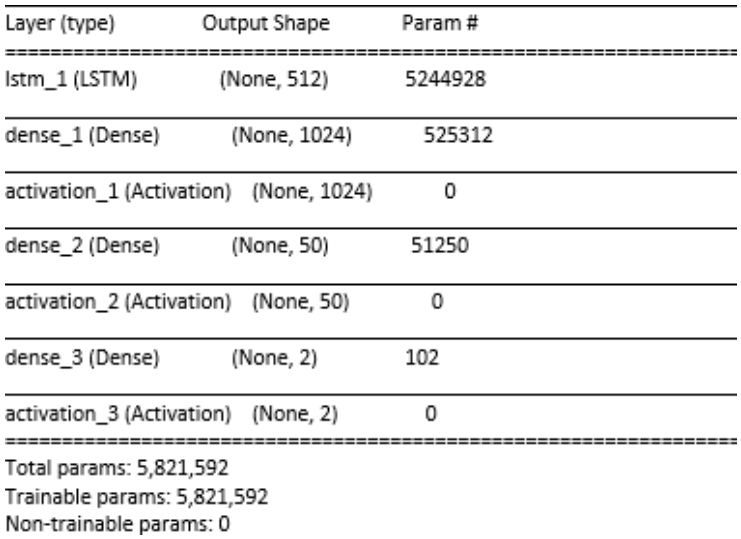


Figura 11 Estructura del modelo LSTM.

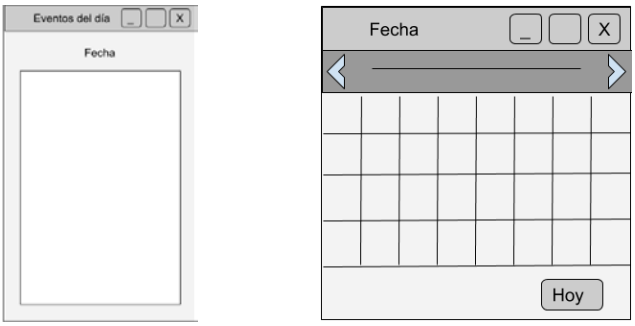
Tabla 7 Parámetros para el entrenamiento red LSTM.

Parámetros	**Valor**
Épocas	100
Frames por acción	15
Videos de entrenamiento	140
Videos de validación	60

Tabla 8 Resultados del entrenamiento de la red LSTM.

Resultados	**%**
Exactitud	98.36
Exactitud en la validación	94.32

Las figuras 12, 13, 14 y 15 muestran la interfaz gráfica diseñada para la administración del sistema.



a) Lista de acciones. *b) Calendario.*

Figura 12 Wireframes de la interfaz.

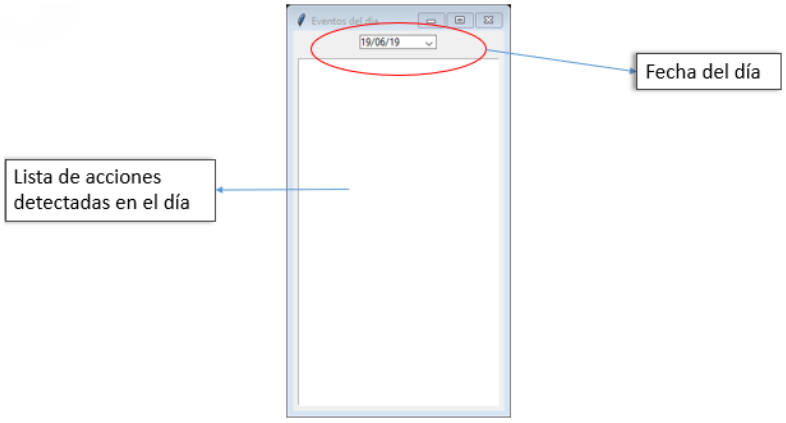


Figura 13 Programación de la interfaz.

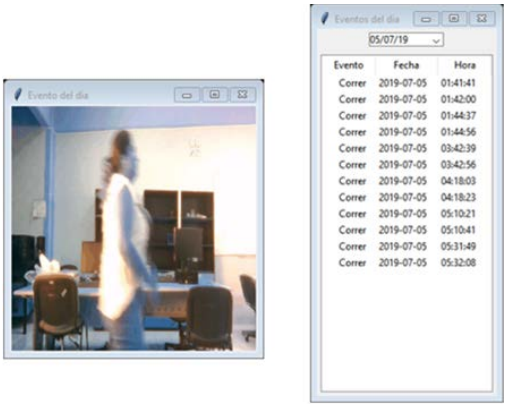


Figura 14 Lista de acciones y acción seleccionada.

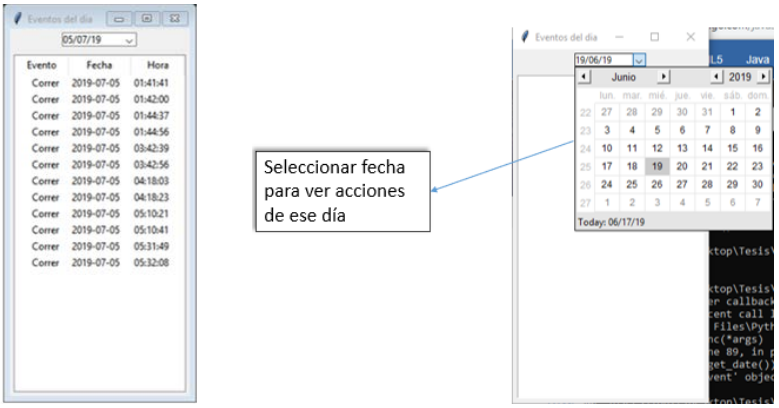


Figura 15 Programación en calendario.

4. Discusión

Al estudiar los principales frameworks de Python, se eligió el más ligero, ya que, de mejor la implementación de las interfaces gráficas, que resultan robustas sin

dejar de lado la facilidad para adaptarse al lenguaje y a cualquier entorno, incluso con bajas prestaciones.

Al completar las fases planteadas usando un Data Set de entrenamiento, enfocado en videos tratados previamente, limpiando la información y quitando tiempos muertos que provoquen falsos positivos a futuro.

En total el DataSet tiene una duración de 25 minutos, lo cual equivale en promedio a 100 cortometrajes con 120 frames cada uno. Por otro lado, las pruebas realizadas tienen los parámetros, donde se descubrió que, a mayor número de épocas, manejando una cantidad estática de frames, la exactitud se ve disminuida. Por el contrario, a mayor cantidad de frames analizados para la red LSTM mejora la exactitud, teniendo como límite superior 15 frames porque a partir de ese punto la exactitud empieza a disminuir.

Por último, el tiempo requerido para realizar el entrenamiento del modelo fue de aproximadamente 1 hora 25 minutos, dicho tiempo incrementaba exponencialmente al incrementar el número de videos en el DataSet de entrenamiento, sin incrementar considerablemente el porcentaje de exactitud de ahí el hecho de las dimensiones finales de la base de datos de videos.

5. Conclusiones

En este trabajo se logró un modelo de aprendizaje profundo supervisado capaz de analizar automáticamente un video e identifica sí una persona corre, integrado a un sistema que administra y visualiza los eventos detectados.

El Dataset utilizado para el entrenamiento y testeo del modelo fue diseñado a partir de videos obtenidos de repositorios en línea, y grabaciones propias, dichos videos fueron tratados y recortados para maximizar el tiempo de entrenamiento efectivo, dando pie a un nuevo DataSet con resultados de más del 98 por ciento de exactitud para la acción de correr comparado con la acción de caminar.

Como trabajos futuros se tiene en primer lugar el mejoramiento del DataSet de entrenamiento adaptando parámetros como ángulo y la distancia, mejorando la capacidad de detección en imágenes captadas con cámaras como las usadas en la video vigilancia. En segundo lugar, el entrenamiento del modelo de detección con

acciones como fumar y/o caminar, así como evaluar la capacidad de detención en acciones más complejas: como el cruzar zonas prohibidas o transitar por lugares indebidos.

6. Bibliografía y Referencias

[1] Bargnato J. I., Breve historia de las redes neuronales artificiales, La coruña, España, 2018.

[2] Bagnato J. I., (2018). Aprende Machine Learning: https://www.aprendemachinelearning.com/como-funcionan-las-convolutional-neural-networks-vision-por-ordenador/.

[3] Calvo D., (2017). Red Neuronal Convolucional: http://www.diegocalvo.es/red-neuronal-convolucional/.

[4] Cano Jorge J., Clasificación de vídeos mediante Redes Neuronales Artificiales, Tesis de maestría, Universitat Politècnica de València, Valencia, España, 2015.

[5] Du Y., Wang W. y Wang L., Hierarchical Recurrent Neural Network for Skeleton Based Action Recognition, The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, pp. 1110 – 1118, 2015.

[6] Fernández L. C., Identificación automática de acciones humanas en secuencias de video para soporte de videovigilancia, Pontificia Universidad Católica del Perú, Lima, Perú, 2017.

[7] Gonzalez J., & Yu W., (2018). Non-linear system modeling using LSTM neural networks. IFAC-Papers, 51(13), 485–489. https://doi.org/10.1016/j.ifacol.2018.07.326.

[8] Google Cloud, (2021). Guía avanzada de Inception v3 para Cloud TPU | Cloud TPU: https://cloud.google.com/tpu/docs/inception-v3-advanced?hl=es-419.

[9] López M., (2019). UNOCERO: https://www.unocero.com/software/python-lenguaje-mas-popular/. [Último acceso: 16 junio 2019].

[10] Martin S., (2019) NVIDIA Blogs: https://blogs.nvidia.com/blog/2019/02/07/what-is-transfer-learning/.

[11] Lopes A. P., Santos E., Valle E., Almeida J. y Araujo A., Transfer Learning for Human Action Recognition, 24th SIBGRAPI Conference on Graphics, Patterns, and Images, Maceió, Los Alamitos, pp. 28-31, 2012.

[12] Prevent Security Systems, (2019). Cámaras de Seguridad y Videovigilancia con Inteligencia Artificial: https://www.prevent.es/camaras-de-seguridad-y-videovigilancia-con-inteligencia-artificial

[13] Secure Week, (2019). Secure Week: https://www.secureweek.com /2019/02/14/el-potencial-de-la-inteligencia-artificial-en-la-video-vigilancia/.

[14] Stanford University (2020). ImageNet, Stanford Vision Lab: https://imagenet.org.

[15] Sun L., Jia K., Yeung D.-Y. y Shi B. E., Human Action Recognition Using Factorized Spatio-Temporal Convolutional Networks, de The IEEE International Conference on Computer Vision (ICCV), Santiago, pp. 4597 – 4605, 2015.

[16] Yaranga C. B., Sánchez S. y Rodríguez R., Transferencia de Aprendizaje Mediante Redes Neuronales Convolucionales para el Reconocimiento de Conductores Distraídos, Tecnia, vol. 28, No. 2, 2018.