



**ESCUELA SUPERIOR POLITÉCNICA AGROPECUARIA DE MANABÍ
MANUEL FÉLIX LÓPEZ**

CARRERA DE COMPUTACIÓN

**INFORME DE TRABAJO DE INTEGRACIÓN CURRICULAR
PREVIO A LA OBTENCIÓN DEL TÍTULO DE INGENIERO EN
CIENCIAS DE LA COMPUTACIÓN**

**MECANISMO: SISTEMATIZACIÓN DE EXPERIENCIAS PRÁCTICAS
DE INVESTIGACIÓN Y/O INTERVENCIÓN**

TEMA:

**SISTEMA DE ALERTA TEMPRANA DE LABOR DE PARTO EN
EL HATO PORCINO EMPLEANDO TÉCNICAS DE REDES
NEURONALES CONVOLUCIONALES Y LSTM**

AUTORES:

**KAREN BARBARITA ALAVA ZAMBRANO
WILLIANS EDUARDO BASURTO VIDAL**

TUTOR:

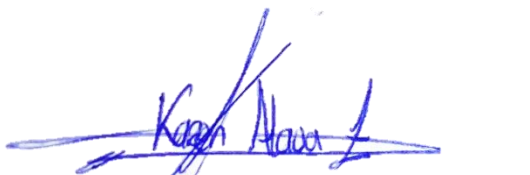
ING. VICTOR JOEL PINARGOTE BRAVO, MGTR.

CALCETA, JULIO DE 2024

DECLARACIÓN DE AUTORÍA

KAREN BARBARITA ÁLAVA ZAMBRANO , con cédula de ciudadanía 1315943371; y WILLIANS EDUARDO BASURTO VIDAL, con cédula de ciudadanía 1315121317 , declaramos bajo juramento que el Trabajo de Integración Curricular titulado: **SISTEMA DE ALERTA TEMPRANA DE LABOR DE PARTO EN EL HATO PORCINO EMPLEANDO TÉCNICAS DE REDES NEURONALES CONVOLUCIONALES Y LSTM** es de nuestra autoría, que no ha sido previamente presentado para ningún grado o calificación profesional, y que hemos consultado las referencias bibliográficas que se incluyen en este documento.

A través de la presente declaración, concedemos a favor de la Escuela Superior Politécnica Agropecuaria de Manabí Manuel Félix López una licencia gratuita, intransferible y no exclusiva para el uso no comercial de la obra, con fines estrictamente académicos, conservando a nuestro favor todos los derechos patrimoniales de autor sobre la obra, en conformidad con el Artículo 114 del Código Orgánico de la Economía Social de los Conocimientos, Creatividad e Innovación.



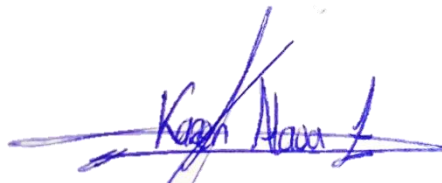
KAREN B. ÁLAVA ZAMBRANO
CC: 1315943371



WILLIANS E. BASURTO VIDAL
CC: 1315121317

AUTORIZACIÓN DE PUBLICACIÓN

KAREN BARBARITA ÁLAVA ZAMBRANO, con cédula de ciudadanía 1315943371; y WILLIANS EDUARDO BASURTO VIDAL, con cédula de ciudadanía 1315121317, autorizamos a la Escuela Superior Politécnica Agropecuaria de Manabí Manuel Félix López, la publicación en la biblioteca de la institución del Trabajo de Integración Curricular titulado: **SISTEMA DE ALERTA TEMPRANA DE LABOR DE PARTO EN EL HATO PORCINO EMPLEANDO TÉCNICAS DE REDES NEURONALES CONVOLUCIONALES Y LSTM**, cuyo contenido, ideas y criterios son de nuestra exclusiva responsabilidad y total autoría.



KAREN B. ÁLAVA ZAMBRANO
CC: 1315943371



WILLIANS E. BASURTO VIDAL
CC: 1315121317

CERTIFICACIÓN DEL TUTOR

Víctor Joel Pinargote Bravo, certifica haber tutelado el Trabajo de Integración Curricular titulado: **SISTEMA DE ALERTA TEMPRANA DE LABOR DE PARTO EN EL HATO PORCINO EMPLEANDO TÉCNICAS DE REDES NEURONALES CONVOLUCIONALES Y LSTM**, que ha sido desarrollado por Karen Barbarita Álava Zambrano y Willians Eduardo Basurto Vidal , previo a la obtención del título de **INGENIERO EN CIENCIAS DE LA COMPUTACIÓN**, de acuerdo al **REGLAMENTO DE LA UNIDAD DE INTEGRACIÓN CURRICULAR DE CARRERAS DE GRADO** de la Escuela Superior Politécnica Agropecuaria de Manabí Manuel Félix López.

VÍCTOR JOEL PINARGOTE BRAVO

CC: 1310867930

TUTOR

APROBACIÓN DEL TRIBUNAL

Los suscritos integrantes del Tribunal correspondiente, declaramos que hemos **APROBADO** el Trabajo de Integración Curricular titulado: **SISTEMA DE ALERTA TEMPRANA DE LABOR DE PARTO EN EL HATO PORCINO EMPLEANDO TÉCNICAS DE REDES NEURONALES CONVOLUCIONALES Y LSTM**, que ha sido desarrollado por Karen Barbarita Álava Zambrano y Willians Eduardo Basurto Vidal, previo a la obtención del título de **INGENIERO EN CIENCIAS DE LA COMPUTACIÓN**, de acuerdo al **REGLAMENTO DE LA UNIDAD DE INTEGRACIÓN CURRICULAR DE CARRERAS DE GRADO** de la Escuela Superior Politécnica Agropecuaria de Manabí Manuel Félix López.

MGTR. LUIS C. CEDEÑO VALAREZO

CC:1306246651

PRESIDENTE DEL TRIBUNAL

MGTR. ALFONSO T. LOOR VERA

CC:1311655938

MIEMBRO DEL TRIBUNAL

MGTR. ANGEL A. VELEZ MERO

CC:1308648565

MIEMBRO DEL TRIBUNAL

AGRADECIMIENTO

A la Escuela Superior Politécnica Agropecuaria de Manabí Manuel Félix López que nos dio la oportunidad de formarnos como profesionales con una educación de calidad, y en la cual hemos forjado nuestros conocimientos y valores profesionales cada día;

Así mismo estamos fraternamente agradecidos con la carrera de Computación por sus enseñanzas y experiencias durante todo el proceso académico, a sus docentes por su excelente labor de formación, por guiarnos y brindarnos sus conocimientos en transcurso de la carrera universitaria,

Al Ing. Joffre Moreira Pico director de la Carrera de Computación por estar siempre presto en ayudarnos y colaborarnos en esta etapa,

A nuestro tutor de tesis Ing. Víctor Pinargote por su entrega, compromiso y guía en el desarrollo del trabajo de titulación,

A la Ing. Jessica Morales Carrillo, excelente docente que nos apoyó y guio en esta última etapa de nuestra carrera,

A la Ing. Nadia Mendoza Gonzales encargada de la UDIV (Unidad de Docencia de Investigación y Vinculación) del hato porcino de la carrera de Medicina Veterinaria, por abrirnos las puertas para poder llevar a cabo el desarrollo de nuestro trabajo de titulación,

A nuestros compañeros de clases durante todos los niveles de universidad, por los buenos momentos compartidos y por todas las experiencias que quedaran plasmadas por siempre, y todas esas personas que de una u otra forma nos han apoyado.

Karen B. Álava Zambrano
Willians E. Basurto Vidal

DEDICATORIA

A Dios el pilar fundamental en mi vida; porque sin el nada somos, el que me ha dado fortaleza en los momentos más duros y difíciles durante este proceso académico.

A mis amados padres Rafael y Barbarita por su entrega, amor y paciencia, por haberme forjado como la persona que soy en la actualidad; muchos de mis logros se los debo a ellos entre los que se incluye este, por su motivación, consejos y por confiar siempre en mí.

A mis hermanos Jonathan y Luis Carlos, que día a día con su presencia, apoyo, respaldo y cariño, me impulsaron a seguir adelante, además saber que mis logros también son los suyos.

A mis compañeros y amigos: presentes y pasados, por ayudarme siempre, compartiéndome sus conocimientos, alegrías y tristezas, por su apoyo moral.

Karen B. Álava Zambrano

DEDICATORIA

A Dios por permitirme vivir gozando de salud y ganas de seguir adelante cada día.

A mi hermosa madre, a mi tía Carmen Elizabeth Basurto Álava y su esposo Eddy José Mora Vera que cumplieron el rol de padre y madre de corazón por haberme impulsado y apoyado para que continuara con mis estudios de tercer nivel.

A mi hermano, por apoyarme en los momentos más críticos de mi carrera para que también algún día sea profesional y sepa que cuando uno se propone algo, lo podemos cumplir.

A mi amigo de toda la vida primo y compadre José Daniel Vidal Párraga que con el dolor del alma partió de este mundo muy repentinamente, compadre sé que estarás feliz al saber que hoy concluyó con mis estudios superiores y agradeciéndote de todo corazón por motivarme siempre a seguir adelante sin importar que tan difícil se pongan las circunstancias a enseñarme el valor del trabajo duro, a hacerme entender que la prioridad de nuestra vida serán los hijos por eso en este día sé que con la culminación de mi preparación como profesional podre apoyar a tus dos hermosas nenas por ti.

Willians E. Basurto Vidal

CONTENIDO GENERAL

CARATULA.....	I
DECLARACIÓN DE AUTORÍA	II
AUTORIZACIÓN DE PUBLICACIÓN	III
CERTIFICACIÓN DEL TUTOR	IV
APROBACIÓN DEL TRIBUNAL.....	V
AGRADECIMIENTO	VI
DEDICATORIA	VII
DEDICATORIA	VIII
CONTENIDO GENERAL.....	IX
CONTENIDO DE FIGURAS.....	XI
RESUMEN	XII
PALABRAS CLAVE	XII
ABSTRACT.....	XIII
KEY WORDS	XIII
CAPÍTULO I. ANTECEDENTES	1
1.1 DESCRIPCIÓN DE LA INSTITUCIÓN.....	1
1.2 DESCRIPCIÓN DE LA INTERVENCIÓN	3
1.3 OBJETIVOS	5
1.3.1 OBJETIVO GENERAL	5
1.3.2 OBJETIVOS ESPECÍFICOS	5
CAPÍTULO II. DESARROLLO METODOLÓGICO DE LA INTERVENCIÓN	6
2.1. ETAPA 1: COMPRENSIÓN DEL PROBLEMA O NEGOCIO.....	7
2.2. ETAPA 2: COMPRENSIÓN DE DATOS.....	9
2.3. ETAPA 3: PREPARACIÓN DE DATOS.....	9
2.4. ETAPA 4: MODELADO	9
2.5. ETAPA 5: EVALUACIÓN DEL MODELO	10
2.6. ETAPA 6: IMPLEMENTACIÓN DEL MODELO (DESPLIEGUE).....	10
CAPITULO III. DESCRIPCIÓN DE LA EXPERIENCIA.....	11
3.1. COMPRENSIÓN DEL PROBLEMA O NEGOCIO	11
3.2. COMPRENSIÓN DE LOS DATOS	14
3.2.1. RECOPIACIÓN DE DATOS:.....	14
3.2.2. REVISIÓN Y ANÁLISIS VISUAL:.....	14

3.2.3. ANÁLISIS TEMPORAL:	14
3.2.4. IDENTIFICACIÓN DE CARACTERÍSTICAS RELEVANTES:	14
3.2.5. DETECCIÓN DE ANOMALÍAS:	15
3.3. PREPARACIÓN DE DATOS	15
3.3.1. LIMPIEZA DE DATOS (DATA CLEANING):	15
3.3.2. TRANSFORMACIÓN DE DATOS (DATA TRANSFORMATION):	16
3.3.3. OVERSAMPLING (SOBREMUESTREO):	17
3.3.4. PREPARACIÓN PARA EL ENTRENAMIENTO DEL MODELO:	18
3.4. MODELADO	19
3.4.1. EVALUACIÓN	21
3.4.2. CODIFICACIÓN	21
3.5. EVALUACIÓN DEL MODELO	23
3.5.1. ENTRENAMIENTO	23
3.5.2. PRUEBA	23
3.6. IMPLEMENTACIÓN DEL MODELO (DESPLIEGUE)	25
CAPÍTULO IV. CONCLUSIONES Y RECOMENDACIONES	28
4.1. CONCLUSIONES	28
4.2. RECOMENDACIONES	29
BIBLIOGRAFÍA	30
ANEXOS	34

CONTENIDO DE FIGURAS

Figura 3.1. Porcentajes de tasas de mortalidad de porcino	12
Figura 3.2. Captura de las cerdas en el proceso de parto	15
Figura 3.3. Ajuste de edición de video.....	16
Figura 3.4. Script de Python	16
Figura 3.5. Folder data y los datos que contienen Data_file.csv	17
Figura 3.6. Estructura de directorio de los conjuntos de entrenamientos.....	18
Figura 3.7. Vista por consola de las características del entorno LSTM_HATO	19
Figura 3.8. Especificaciones técnicas de Hardware Grafico	20
Figura 3.9. Estructura del directorio del folder principal del proyecto	20
Figura 3.10. Arquitectura del modelo.....	21
Figura 3.11. Código Python de la topología de la red.	22
Figura 3.12. Sentencia python del script train.py	23
Figura 3.13. Sentencia pythom del bloque de código clasify.py.....	23
Figura 3.14. Resultados de las pruebas preliminares	24
Figura 3.15. BotFather de Telegram.....	25
Figura 3.16. Interfaz de Telegram con los respectivos comandos ejecutados	26
Figura 3.17. Código Python para él envió de notificaciones mediante el chat	26
Figura 3.18. Prueba de la alarma enviada al chat de Telegram.....	27

RESUMEN

El presente trabajo de titulación tuvo como objetivo desarrollar un sistema de alerta temprana para detectar patrones de parto en las cerdas y emitir una alertar de forma anticipada a los cuidadores del hato porcino indicando que la cerda esta próxima a labor de parto empleando técnicas de inteligencia artificial tales como redes neuronales convolucionales y modelos LSTM (Long Short Term Memory). Para el desarrollo de este trabajo se emplearon dos metodologías CRISP-DM enfocada a la planificación y seguimiento mismo que cuenta con fases tale como: Comprensión del problema, comprensión de datos, preparación de datos, modelado, implementación del modelo; también se empleó la metodología SABOR IBM enfocada netamente al desarrollo de código del proyecto en cuestión. Finalmente, entrenado el modelo se creó un bot de Telegram para emitir la alerta de predicción, después de realizar la respectiva prueba en caliente, se confirmó que el sistema emite alertas efectivas a los cuidadores cuando la probabilidad de parto supera el 70%. Esto asegura una intervención oportuna por parte del equipo a cargo.

PALABRAS CLAVE

Redes neuronales, CRISP-DM, parto porcino, patrón de comportamiento, predicción de parto

ABSTRACT

The objective of this titration work was to develop an early warning system to detect farrowing patterns in sows and issue an early alert to the caretakers of the pig herd indicating that the sow is close to labor using artificial intelligence techniques. such as convolutional neural networks and LSTM (Long Short Term Memory) models. For the development of this work, two CRISP-DM methodologies were used, focused on planning and monitoring, which has phases such as: Understanding the problem, understanding data, data preparation, modeling, model implementation; The SABOR IBM methodology was also used, focused purely on the code development of the project in question. Finally, once the model was trained, a telegram bot was created to issue the prediction alert. After performing the respective hot test, it was confirmed that the system issues effective alerts to caregivers when the probability of birth exceeds 70%. This ensures timely intervention by the team in charge.

KEY WORDS

Neural networks, CRISP-DM, pig parturition, behavior pattern, parturition prediction

CAPÍTULO I. ANTECEDENTES

1.1 DESCRIPCIÓN DE LA INSTITUCIÓN

La LOES (2018) señala que el Sistema de Educación Superior tiene como finalidad la formación académica y profesional con visión científica y humanista; brindando soluciones innovadoras en conjunto a la promoción, desarrollo, difusión de la academia en post de la sociedad ecuatoriana, en relación con los objetivos del régimen de desarrollo.

De acuerdo con el Estatuto de la Escuela Superior Politécnica de Agropecuaria de Manabí Manuel Félix López (2019) (ESPAM MFL), en su artículo número 3 establece que, es una institución pública sin fines de lucro creada mediante ley 99-25 publicado en el registro oficial No.181, del 30 de abril de 1999, estableciendo su sede en la parroquia Calceta, cantón Bolívar, provincia de Manabí. Mediante la enseñanza universitaria, la investigación científica y el emprendimiento, con un alto potencial en el contexto rural y socioeconómico.

La ESPAM MFL, por medio de la carrera de Ingeniería en Ciencias de la Computación impulsa la generación de conocimientos que contribuyan a transformar la matriz productiva a través de la industria tecnológica, misma que tiene como objetivo “formar profesionales que aporten innovaciones computacionales para la solución de problemas sociales, regionales y nacionales, vinculados al modelo constructivista y desarrollador productivo, dentro de equipos multidisciplinarios e interdisciplinarios, con énfasis en el sector agropecuario y agroindustrial, que actúen con responsabilidad económica, ambiental, ética y social, en sintonía con los planes y políticas públicas” (CES, 2019a).

CES (2019a) plantea en el rediseño que las Ciencias de la Computación tiene una relación prácticamente transversal con otras áreas del conocimiento, desde aquellas relacionadas con las ciencias de la vida, hasta las que tratan aspectos económicos, pasando por aquellas conexas con la producción e innovación tecnológica en múltiples campos y disciplinas.

La ESPAM MFL, a través de la Carrera de Medicina Veterinaria impulsa la generación conocimientos que contribuyan a transformar la matriz agro productiva a través de la industria zootécnica, misma que tiene como objetivo “Formar profesionales íntegros, capaces de actuar con criterios técnicos, principios éticos y humanos que le permitan resolver problemas de salud pública mediante el mejoramiento de la producción pecuaria, contribuyendo al desarrollo socio económico y ambiental del entorno” (CES, 2019b).

Rodríguez et al. (2020) afirman que las unidades de docencia, investigación y vinculación (UDIV) son consideradas pilares fundamentales de todas las carreras de la ESPAM MFL, forman parte de las funciones sustantivas de las Instituciones de Educación Superior, la cual permite la relación directa y cotidiana dentro de una sociedad que contribuye al desarrollo económico.

Vera (2019) indica que la ESPAM MFL cuenta con la Carrera de Medicina Veterinaria que entre sus medios de producción se destaca la UDIV hato porcino, que permitirá desarrollar el proceso investigativo, académico y de vinculación con los docentes, estudiantes y la comunidad en general.

El hato porcino cuenta con ambientes y personal dedicado al manejo del cuidado, alimentación y reproducción de los cerdos los cuales se ofrecen a la comunidad.

1.2 DESCRIPCIÓN DE LA INTERVENCIÓN

Ocaña et al. (2019) y Berryhill et al. (2020) establecen que, en la actualidad el empleo de técnicas de aprendizaje automático permite actualizar y optimizar un conjunto de procesos que tradicionalmente dependían de la intervención de mano de obra humana para su ejecución. Un caso particular se ve reflejado en el campo pecuario, específicamente en los procesos de porcicultura la cual, si bien no puede adoptar al cien por ciento la automatización de todos sus procesos con respecto a la reproducción pecuaria, se puede emplear mecanismos de aprendizaje automático para sobrellevar de forma ágil e innovadora ciertas etapas del proceso de crianza (Rodríguez 2020; Muñoz et al. 2020).

Las redes neuronales son un tipo de implementación de la inteligencia artificial que busca imitar el pensamiento humano a través de la replicación de la anatomía del cerebro como lo detallan Alemán (2017), Egüez (2020) y Valderrama et al. (2021) “Las RNA son un método de resolver problemas, de forma individual o combinadas con otros métodos, para aquellas tareas de clasificación, identificación, diagnóstico, optimización o predicción”.

Por otra parte, Bonilla (2020) y Sarmiento (2020) establecen que dentro de las técnicas de inteligencia artificial se encuentran las redes neuronales convolucionales, las cuales acorde a lo que los autores determinan que “son un tipo de modelo de aprendizaje profundo que se usa casi universalmente en aplicaciones de visión artificial. Las CNN se pueden aplicar a problemas de clasificación, particularmente aquellos que involucran conjuntos de imágenes de entrenamiento”.

Un caso particular del empleo de la inteligencia artificial en el campo pecuario se ve evidenciado en el estudio de Gavilanes (2020) desarrolló un sistema de monitoreo apícola el cual predecía la variación de población de abejas melíferas basado en la arquitectura artificial de red neuronal recurrente de memoria a corto y largo plazo (LSTM), misma que aprenden los comportamientos pasados más importantes y comprendía si esos comportamientos eran características importantes para hacer predicciones futuras, los resultados obtenidos en la predicción de la densidad poblacional de la colmena, confirman la aplicabilidad del modelo de la red con una asertividad mayor al 90 %, así se muestra la

viabilidad en la implementación del sistema. Otra aplicación de las redes neuronales el comportamiento animal la realizó Cañal (2017), quien implementó un método para llevar a cabo la clasificación de comportamientos de animales, como andar, trotar o estar parado, para resolver el problema de la clasificación, se utilizó redes neuronales y en concreto de las del tipo backpropagation. Así, entrenando dicha red, se consiguen clasificar los distintos patrones de movimiento del animal, aplicando la metodología concretamente a los caballos.

Marín et al. (2019) afirma que el porcentaje de mortalidad neonatal varía mucho entre granjas, oscilando entre el 5 y el 35%. Las muertes ocurren principalmente durante las primeras 48 horas después del parto siendo el aplastamiento por la cerda la primera causa de mortalidad. El final del parto es el momento más peligroso, y es donde hay que extremar la atención Arroyo (2019). Aquí podemos comprender lo necesario que es que la cerda tenga una persona que le esté ayudando o atendiendo a la hora del parto Rodríguez (2021).

La UDIV - hato porcino está enfocado en las áreas específicas de (crianza, reproducción y maternidad), el presente trabajo de integración curricular va a ser dirigido al área de maternidad donde se efectúan procesos como: seguimiento de gestación, partos y periodos de lactancia. El área de maternidad presenta inconvenientes con los procesos de parto, ya que estos se presentan por lo general en horas de la madrugada dificultando el trabajo del cuidador que muchas veces por no saber el momento exacto del alumbramiento, pueden ocurrir abortos o decesos involuntarios por parte de la cerda.

El presente trabajo de integración curricular se enfocó en el desarrollo un sistema de alerta temprana de labor de parto en el de la UDIV - Hato Porcino de la Carrera de Medicina Veterinaria de la ESPAM MFL, Para ello, se utilizó redes neuronales convolucionales y LSTM para el reconocimiento de patrones de la cerda al momento del proceso de alumbramiento, este mismo sistema también cuenta con un mecanismo que emite una alerta a los cuidadores señalando que la cerda está próxima a labor de parto, lo que facilita el trabajo de los cuidadores en UDIV y aumenta así la reproducción de los porcinos, reduciendo la probabilidad de mortalidad que se produce por lo general por abortos, aplastamientos o debilidad en la cerda para continuar con el parto.

1.3 OBJETIVOS

1.3.1 OBJETIVO GENERAL

Desarrollar un sistema de alerta temprana en el rebaño reproductor del hato porcino empleando técnicas de redes neuronales convolucionales y LSTM para la detección de patrones a través del análisis de video

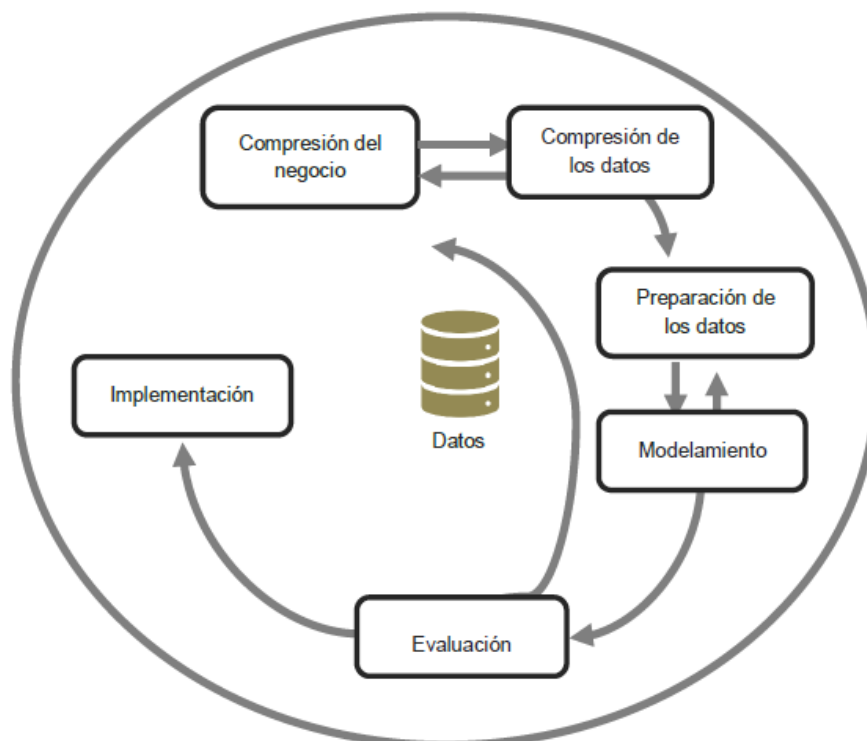
1.3.2 OBJETIVOS ESPECÍFICOS

- Clasificar vídeos de proceso de parto de porcinos.
- Estudiar el funcionamiento de las redes neuronales convolucionales y redes LSTM.
- Entrenar un modelo de red neuronal convolucional para la identificación y predicción de parto porcino.
- Implementar mecanismo de emisión de alertas en el hato porcino para la predicción de parto.

CAPÍTULO II. DESARROLLO METODOLÓGICO DE LA INTERVENCIÓN

Para llevar a cabo la ejecución del presente trabajo de integración curricular, se utilizó la metodología CRISP-DM (Cross Industry Standard Process for Data Mining), la cual proporciona una descripción normalizada del ciclo de vida de un proyecto estándar de análisis de datos, Espinosa (2020) establece que el modelo CRISP-DM consta de seis etapas, Cada objetivo específico del proyecto se alinea con una o más de estas etapas. Además, en el marco del tercer objetivo, que corresponde a las etapas cuatro y cinco de la metodología CRISP-DM, se implementó una metodología ágil conocida como SABOR IBM. Esta metodología fue seleccionada para optimizar el proceso de desarrollo y asegurar una ejecución eficaz y flexible, adecuada a las necesidades dinámicas del proyecto.

figura 2.1: Etapas de la metodología CRISP-DM

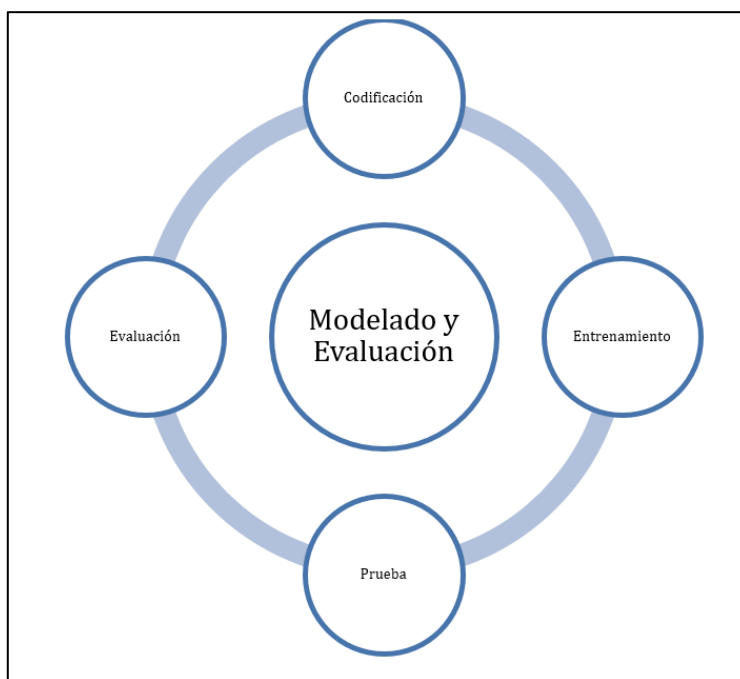


Fuente. Mancilla-Vela et al., (2020)

Para la construcción tanto de la etapa de modelado y la etapa de evaluación del modelo que comprende el objetivo 3 se empleó una metodología ágil denominada SABOR IBM, Sicilia et al., (2018) establece un conjunto de procesos que se ejecutan de forma iterativa a fin de obtener un producto cumpla con los

requerimientos del problema de los cuales se establecieron cuatro procesos fundamentales que son evaluación, codificación, entrenamiento y prueba, en la figura 2.2. se realizó un gráfico para una mejor apreciación, en la etapa de modelado comprende dos estos procesos que son la evaluación y codificación, y la etapa de evaluación del modelo abarca los otros dos procesos entrenamiento y prueba.

figura 2.2: Ciclo de modelado



Fuente. Los autores, 2022

2.1. ETAPA 1: COMPRENSIÓN DEL PROBLEMA O NEGOCIO

Esta etapa se centra en investigar y comprender a fondo el problema que se desea abordar. El objetivo es identificar posibles desafíos, definir con claridad los objetivos a alcanzar, incluyendo las métricas que se utilizarán para evaluar el éxito y establecer los recursos necesarios para llevar a cabo la tarea de manera efectiva.

Para la comprensión del problema se identificaron los procesos llevados a cabo por el hato porcino con la finalidad de definir la problemática que se pretende resolver con el sistema a realizar.

Inicialmente se tuvo contacto directo con la Ing. Nadia Mendoza encargada del hato porcino como parte de una visita técnica efectuada por los miembros del trabajo de titulación, con el fin de obtener información oportuna, en la cual se detalló a qué problema se están enfrentando, también se describieron aspectos, como la deficiente respuesta que tienen los cuidadores de las cerdas embarazadas y la situación actual del hato porcino en el área de maternidad.

En esta etapa inicial del proyecto, se logró comprender los desafíos específicos que enfrentan los cuidadores de cerdas embarazadas en la detección temprana del trabajo de parto. Estos desafíos incluyen una respuesta deficiente en la detección actual y la necesidad imperiosa de una alerta temprana para garantizar el bienestar de los animales y optimizar la producción en el hato porcino.

Además, se identificaron características relevantes que podrían indicar la presencia de trabajo de parto en cerdas embarazadas, como la frecuencia cardíaca, la actividad motora y comportamiento de nidificación, entre otras.

Para alcanzar el objetivo 2 que abarca esta etapa se llevó a cabo un exhaustivo estudio del estado del arte, revisando investigaciones previas y proyectos similares que abordaron problemas relacionados. Especial atención se dio a aquellos que utilizaron técnicas de inteligencia computacional.

Basándose en la comprensión del problema y en la revisión de la literatura, se seleccionaron técnicas y herramientas adecuadas para abordar la problemática.

Finalmente, se definieron métricas de rendimiento para evaluar los modelos propuestos y se realizaron experimentos para ajustar los hiperparámetros y evaluar su rendimiento en datos de prueba.

Esta etapa proporcionó una comprensión profunda del problema de detección del trabajo de parto en el hato porcino, sentando las bases para el desarrollo del sistema de alerta temprana sin entrar aún en detalles sobre las técnicas específicas de modelado utilizadas.

2.2. ETAPA 2: COMPRENSIÓN DE DATOS

Esta y la siguiente etapa fueron cruciales para el desarrollo del objetivo 1 que fue la clasificación de video de parto porcino, aquí se enfocó en comprender en detalle los datos recopilados y en prepararlos de manera adecuada para su análisis posterior. El proceso de recopilación de los datos se llevó a cabo gracias a la coordinación entre las carreras de Computación y Medicina Veterinaria, en sinergia con este apartado al tratarse de un modelo de inteligencia artificial enfocado a la visión por computadora los datos necesarios para su entrenamiento está definido como imágenes o secuencia de imágenes (videos). En el caso particular de la investigación los datos requeridos son archivos de video, por lo cual se adquirieron equipo de video vigilancia para recopilar dichos datos. Una vez obtenido los datos se procedió a realizar un análisis profundo, mismo que se encuentra definido en la etapa “preparación de datos”. Según Huber et al. (2019) la finalidad de estos modelos de inteligencia artificial pretende buscar nuevos patrones de frecuencia en los flujos de datos.

2.3. ETAPA 3: PREPARACIÓN DE DATOS

Como parte de la preparación de los datos se procedió a aplicar técnicas de data cleansing, data transformation y oversampling, con el fin de discriminar datos innecesarios o duplicados dentro de la sede de archivos de vídeo, Jara et al. (2020) detalla que esta es una de las etapas más importantes para el correcto desarrollo del modelo ya que, esta es la base de conocimiento empleada en su entrenamiento misma que llega a condicionar índice de eficacia que puede arrojar a través de las métricas estadísticos definidos en la etapa de evaluación que determinan si el modelo está sobreentrenado, desfasado u optimizado.

El objetivo 3 se llevó a cabo atreves de las siguientes dos etapas: el modelado y la evaluación del modelo.

2.4. ETAPA 4: MODELADO

En esta etapa abarca dos procesos de la metodología SABOR IBM cuales son evaluación y codificación; para el desarrollo del modelo de inteligencia artificial se trabajó con el lenguaje Python ya que el mismo cuenta con librerías especializadas para el desarrollo estos modelos como lo son Numpy, SciPy,

Scikit-learn, Matplotlib y TensorFlow detallados por Delgado (2022), se definió el uso de los servicios de aceleración de GPU (unidad de procesamiento gráfico) que brinda Google Colab, en vista de que el core “núcleo” computacional para el proceso de entrenamiento es bastante alto si se lo realiza a través de los ordenadores convencionales, mismos que tienen una capacidad limitada de cómputo gráfico.

2.5. ETAPA 5: EVALUACIÓN DEL MODELO

En esta etapa continúa con los dos últimos procesos de la metodología SABOR IBM que son entrenamiento y prueba; Una vez entrenado el modelo con los datos suministrados se procedió a evaluar su rendimiento empleando ciertas métricas como son: la precisión, la recuperación (recall) y la puntuación f1 (f1 score), error absoluto medio (mae), el error cuadrático medio (mse). dando así una noción de que tan ajustado o desfasado se encuentra el modelo con respecto al conjunto de datos de prueba, y así realizar las correcciones respectivas para optimizar el modelo.

2.6. ETAPA 6: IMPLEMENTACIÓN DEL MODELO (DESPLIEGUE)

La implementación del modelo se llevó a cabo mediante la instalación de un circuito cerrado de cámaras de video vigilancia previamente enlazadas a un ordenador que contiene al modelo ya entrenado en ejecución indefinida a la espera de un patrón que haga saltar la alerta de parto, enfocado en el objetivo 4 que fue la implementación de una alarma para el control y asignación de alertas se creó un bot en la interfaz de mensajería instantánea Telegram con el fin de gestionar la emisión de alertas, mismas que son enviadas a través de mensajes a todos y cada uno de los responsables del cuidado de los porcinos.

CAPITULO III. DESCRIPCIÓN DE LA EXPERIENCIA

Para el desarrollo del proyecto, se realizaron grabaciones y se recolectaron videos de las cerdas en el hato porcino de la ESPAM MFL, proporcionando así el material inicial necesario.

Basado en la metodología CRIS-DM cada objetivo específico se abordó mediante una o dos etapas de la metodología, El primer objetivo, que fue clasificar videos porcinos ya previamente grabados, este mismo se estructuró en dos etapas de la metodología: la etapa 2 y la etapa 3. En la etapa 2, se llevaron a cabo los pasos necesarios para la clasificación, incluyendo la comprensión de los datos. Posteriormente, en la etapa 3, se realizó la preparación de los datos, lo que incluyó la compresión y limpieza de los mismos. Estos procesos se explican detalladamente más adelante.

Para el objetivo 2, que implicó el estudio del funcionamiento de las redes neuronales convolucionales y redes LSTM, se abordó en la etapa 1 de la metodología CRISP-DM. En esta fase, se realizó una introducción a la problemática inicial para adquirir el conocimiento necesario. Se llevó a cabo un exhaustivo estudio del estado del arte y se realizaron comparaciones con otras redes neuronales disponibles. Después de evaluar varias opciones, se identificó la red más adecuada y se procedió a aplicarla en el modelo del proyecto.

3.1. COMPRENSIÓN DEL PROBLEMA O NEGOCIO

En esta etapa, referente a la comprensión del problema, la falencia más evidente que se pudo identificar dentro del hato porcino se vio reflejada en el área de maternidad con los procesos de control partos, como ya se detalló en la descripción de la intervención. Esta es una de las áreas que más utilidades representan para la UDIV – hato porcino, en material económico ya que, los lechones que salen de esta área son ofertados a la población civil del cantón Bolívar.

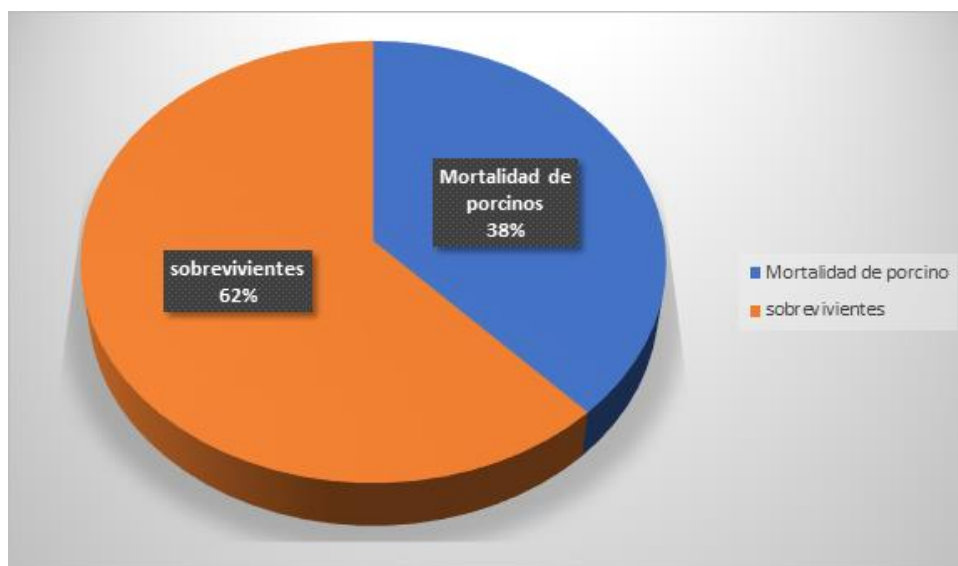
Al realizar la visita técnica (**Anexo 2**) en la UDIV – hato porcino, de acuerdo con la información proporcionada por la encargada del área Ing. Nadia Mendoza, “el proceso de alumbramiento se da generalmente en horas de la madrugada”,

presentándose complicaciones durante el proceso de alumbramiento, lo que influye directa o indirectamente en la cantidad de lechones vivos al final del parto.

También se brindó información de cómo los encargados del hato junto a los cuidadores de las cerdas en proceso de alumbramiento identifican los patrones que les ayudan a identificar que la cerda está en proceso de parto.

Evaluación de la situación actual: Según reportes del hato porcino el 38% de los lechones no sobreviven al proceso de parto como se muestra en la figura 3.1.

Figura 3.1. Porcentajes de tasas de mortalidad de porcino



Fuente. Hato porcino - ESPAM MFL

Para contextualizar el presente trabajo dentro del campo de la investigación, se realizó un exhaustivo estudio del estado del arte para comprender el funcionamiento las técnicas a utilizar como se indicó en el objetivo 2. Se revisaron investigaciones previas y proyectos similares que abordaron problemas relacionados con la detección de patrones y comportamiento en diferentes animales y otros ámbitos, pero no se encontró trabajos concretos en la detección temprana de parto en animales de producción. El proyecto se enfocó especialmente en aquellos trabajos que hicieron uso de técnicas avanzadas de inteligencia artificial, como las redes neuronales convolucionales (CNN) y las redes neuronales de memoria a largo plazo (LSTM) como se puede observar en la tabla 3.1.

Tabla 3.1. Estudio de estado del arte

Año	Autor	Tema	Rede utilizada	Base de datos utilizada	Resultados Clave
2020	Gavilanes Proaño David Andrés	Sistema de monitoreo apícola mediante el uso de redes neuronales Artificiales para identificar la variación de población	LSTM	Variables de temperatura, humedad, dióxido de carbono, peso y la densidad poblacional	Asertividad mayor al 90 %,
2014	Mariano Ferrero	Generación de un modelo mediante el uso de redes neuronales artificiales para la detección de mastitis en vacas lecheras del INTA Estación Experimental Agropecuaria Rafaela	Perceptrón multicapa (MLP).	Sistema automatizado de control de producción lechera.	Efectividad de valores arrojados son 98,8% y 88,4%
2019	Guillermo Eduardo Narváez	Red neuronal convolucional para la detección de aves exóticas en peligro de extinción	CNN	Imágenes de aves en peligro de extinción	efectividad del 89 %.
2022	Hernandez Neria Marco Antonio Rosas	Identificación automática de gatos mediante reconocimiento de imágenes usando redes neuronales convolucionales	CNN	Fotos de gatos	Los resultados que se obtuvieron fueron de un 98% de exactitud al momento de probar la red neuronal convolucional
2021	Iván Mindlin	Reconocimiento de Lengua de Señas con Redes Neuronales Recurrentes	RNN-LSTM	videos de Lengua de señas	alcanzando un 99.4% de tasa de acierto
2021	Renzo Alberto VILLANUEVA ALARCÓN	Sistema inteligente basado en redes neuronales para la identificación de cáncer de piel de tipo melanoma en imágenes de lesiones cutáneas	CNN	Imágenes de lesiones cutáneas clasificadas	Probabilidad de desempeño de 92.85% Precisión, 71.50% Sensibilidad, 94.89% Especificidad
2020	Orozco, C. I., Xamena, E., Buemi, M. E., & Berles, J. J.	Reconocimiento de Acciones Humanas en Videos usando una Red Neuronal CNN LSTM Robusta	CNN-LSTM	Videos de comportamiento humano	93% y 91% de precisión
2023	Castelo, R., Rodríguez, M., & Wins, D.	Inteligencia Artificial aplicada al reconocimiento de crímenes	LSTM	Dataset de videos de hechos violentos ocurridos en la vía publica	Se obtienen resultados positivos, con un f1-score de 0.87 para la solución basada en Long Short-term Memory.
2021	Sebastián Torres Mojica	Detección de mentiras a través de reconocimiento facial usando machine learning	LSTM	videos de testimonios y deposiciones para juicios.	Las predicciones de mentira son del 72%

Fuente. Los Autores.

Basándose en la comprensión profunda del problema y en la revisión crítica de la literatura existente, se seleccionaron cuidadosamente las técnicas y herramientas más adecuadas para abordar la investigación. Dada la naturaleza de los datos y la necesidad de modelar secuencias temporales complejas, se optó por utilizar tanto CNN como LSTM. Estas técnicas se destacaron por su capacidad para procesar datos complejos y extraer patrones significativos, lo que las hace especialmente apropiadas para el contexto.

3.2. COMPRESIÓN DE LOS DATOS

3.2.1. RECOPIACIÓN DE DATOS:

En la etapa dos, de la comprensión de los datos, la cual se encuentra dentro del primer objetivo: Clasificar vídeos de proceso de parto de porcinos. Se puede dar como resultado que la recopilación de datos se llevó a cabo mediante un sistema de video vigilancia instalado en el hato porcino. Este sistema consistió en un total de cuatro cámaras de vídeo modelo Hikvision Turbo Hd 1080p 2mp (**Anexo 3**), las cuales operaron de forma continua, grabando las actividades dentro de las instalaciones las 24 horas del día, durante un periodo de seis meses. Este enfoque permitió capturar una amplia gama de situaciones y eventos, incluyendo los procesos de parto de las cerdas.

La comprensión de datos en este contexto implica un análisis profundo de los vídeos grabados en el hato porcino para identificar los patrones de comportamiento asociados con los partos de las cerdas. Este proceso se llevó a cabo con el objetivo de extraer información valiosa y relevante que sirvieron como base para el desarrollo de sistemas de detección y alerta temprana.

3.2.2. REVISIÓN Y ANÁLISIS VISUAL:

Se procedió a revisar visualmente los vídeos grabados en el hato porcino para identificar los momentos en los que ocurren los partos de las cerdas. Se prestó especial atención a los comportamientos y señales que indican el inicio del trabajo de parto, como movimientos inquietos, y cambios en la postura de la cerda.

3.2.3. ANÁLISIS TEMPORAL:

Se analizó la secuencia temporal de los eventos registrados en los vídeos para entender la duración y la progresión del proceso de parto. Esto puede implicar la identificación de fases específicas del parto, como la dilatación, el expulsivo y la salida del saco amniótico.

3.2.4. IDENTIFICACIÓN DE CARACTERÍSTICAS RELEVANTES:

Se identificaron las características y patrones de comportamiento más relevantes asociados con los partos de las cerdas. Esto puede incluir la frecuencia y la

intensidad de las contracciones uterinas, la duración del parto, y la interacción entre la cerda y sus crías recién nacidas.

3.2.5. DETECCIÓN DE ANOMALÍAS:

Se busca identificar cualquier anomalía o problema durante el proceso de parto que pueda requerir intervención humana. Esto puede incluir la detección de partos difíciles, la identificación de cerdas en situación de estrés o malestar, y la presencia de cualquier complicación obstétrica.

Figura 3.2. Captura de las cerdas en el proceso de parto



Fuente. Hato porcino - ESPAM MFL

3.3. PREPARACIÓN DE DATOS

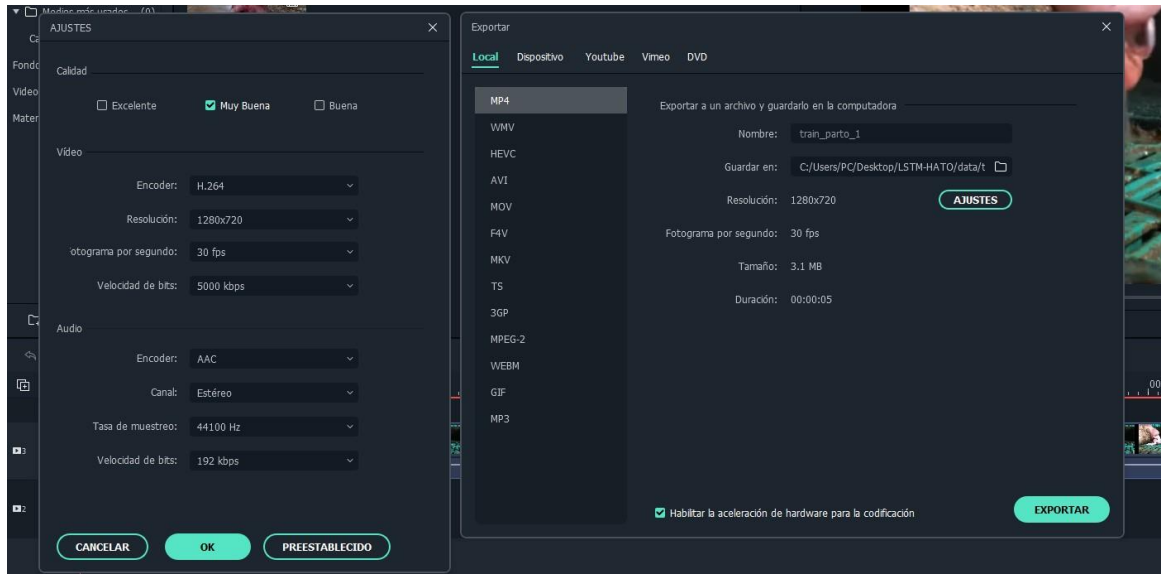
Como resultado de esta selección, se obtuvo un conjunto de 40 videos (**Anexo 4-a**) que sirvieron como base para el análisis y desarrollo del sistema de clasificación.

3.3.1. LIMPIEZA DE DATOS (DATA CLEANING):

El proceso de data cleaning implicó la identificación y eliminación de datos innecesarios, duplicados o inconsistentes dentro del conjunto de archivos de video. Se realizaron exhaustivas revisiones y depuraciones de los datos para asegurar que solo se utilizara información relevante y de alta calidad en el entrenamiento del modelo. Este proceso de selección de los fragmentos de video fue llevado a cabo con el apoyo de la herramienta “filmora” que es un programa de edición de videos creado por Wondersahre Technology, el cual cuenta con un diseño intuitivo y de fácil manejo, incluyendo funciones como: fragmentación de videos, dimensionamiento de resolución, selección de puntos fijos de escenas entre otras (**Anexo 4-b**) cumpliendo así con el primer objetivo que fue clasificar de los videos con patrones específicos de parto porcino.

las tres funciones antes mencionadas fueron de vital importancia para la estandarización de los videos a un formato compatible para el consumo del modelo, mismas que deben cumplir con las siguientes características: como se muestra en la figura 3.3.

Figura 3.3. Ajuste de edición de video



Fuente. Los autores.

3.3.2. TRANSFORMACIÓN DE DATOS (DATA TRANSFORMATION):

Cabe destacar que los videos por definición son imágenes continuas, acompañadas o no de sonidos, almacenados mediante cinta magnética u otros medios electrónicos. Por tal situación se empleó la técnica de data transformación; para convertir los datos de formato .mp4 a .jpg a través de la ejecución del script que se muestra en figura 3.4:

Figura 3.4. Script de Python

```
script de shell
1 python extract_files.py mp4
```

Fuente. Los autores.

Dicha función no recepta parámetros de entrada o argumentos, dado a que no son requeridos por el usuario, en consecuencia, esto origina un proceso iterativo el cual extrae una cantidad determinada de imágenes por segundo de los video alojados en los conjuntos de train y test respectivamente (**Anexo 5**).

3.3.3. OVERSAMPLING (SOBREMUESTREO):

Para reducir el riesgo de desequilibrios o desbalanceo en el conjunto de datos, se aplicó la técnica de oversampling. Esto implicó aumentar la representación de algunas características minoritarias en el conjunto de datos, asegurando así que el modelo se encuentre balanceado de manera equitativa y que todas las clases fueran consideradas de manera justa durante el proceso de entrenamiento.

La cantidad de imágenes generadas estará determinada por la siguiente fórmula

$$\text{Imágenes} = (\text{segundos de video}) * (\text{fotogramas por segundo})$$

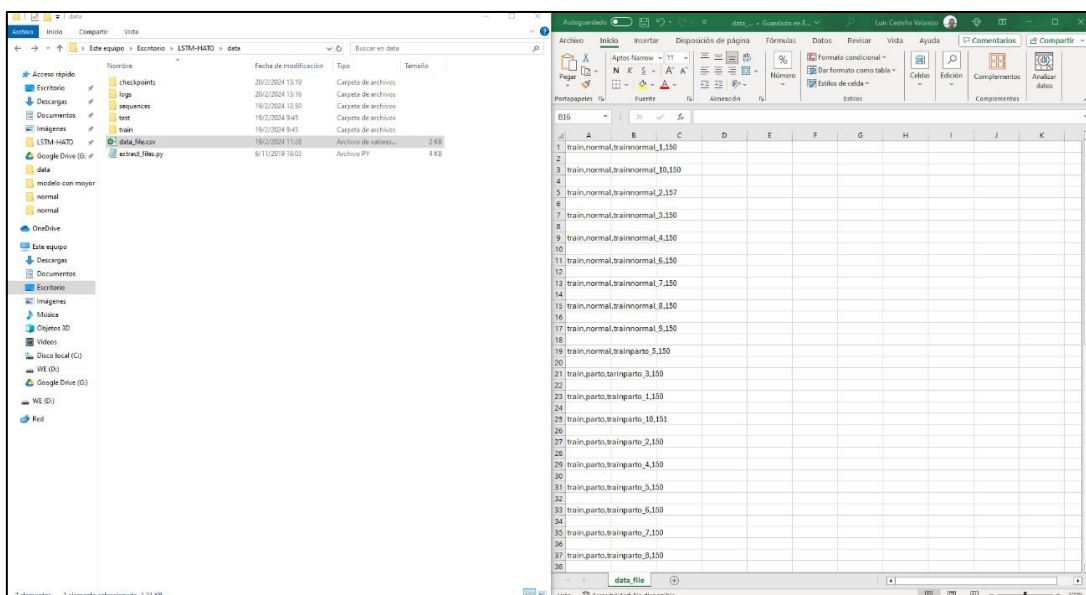
Reemplazando los valores con el experimento que se realizó utilizando fragmentos de videos de 5 segundos de duración en un formato de 30 FPS quedaría de la siguiente manera

$$\text{Imágenes} = (5) * (30)$$

$$\text{Imágenes} = 150$$

Una vez concluida la ejecución del script queda un archivo.csv llamado “data_file.csv” que a su vez registrara los siguientes datos, conjunto al que pertenece, la característica que representa, el nombre del video y la cantidad de imágenes generadas que en este caso es igual o similar a la de la fórmula previamente mencionada.

Figura 3.5. Folder data y los datos que contienen Data_file.csv



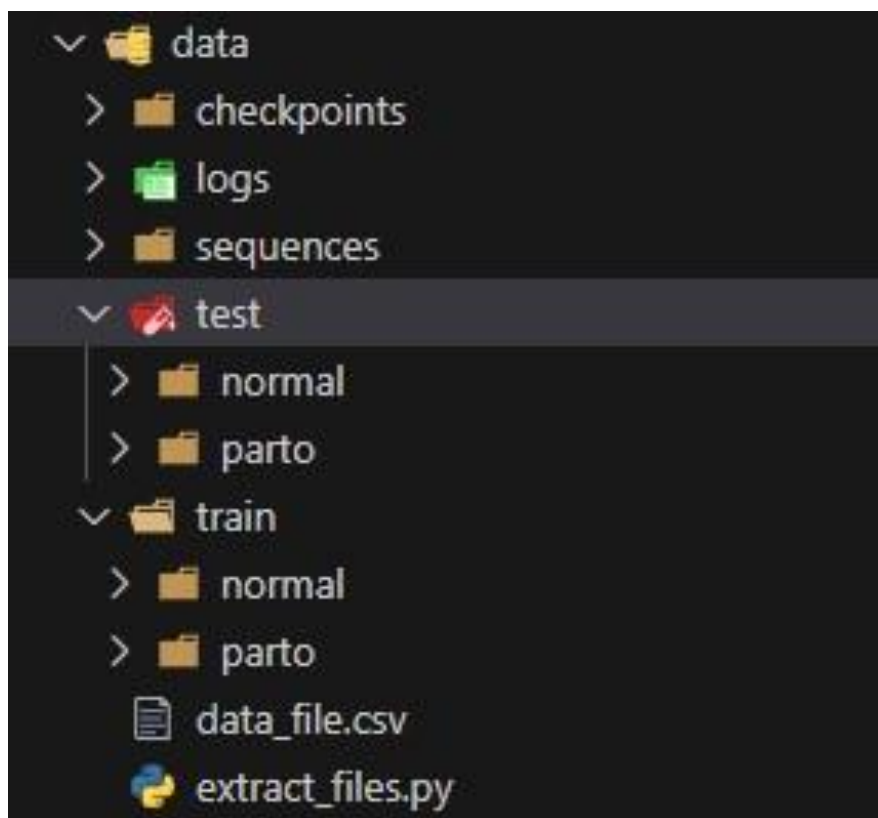
Fuente. Los autores.

Cabe recalcar que es muy importante que estas cantidades sean equivalentes o lo más parecidas posibles ya que existe una validación interna en el proceso de entrenamiento el cual determina que si alguno de estos conjuntos de datos supera en 2 a otro evidenciando un claro desbalanceo de datos, lo que puede ocasionar sobre ajuste (overfitting): es decir; que el modelo memorice los pesos y no generalice, lo que ocasionaría bajo rendimiento en la fase de test.

3.3.4. PREPARACIÓN PARA EL ENTRENAMIENTO DEL MODELO:

Una vez que se realizó la preparación de los datos y en simultaneo de la ejecución del script “extract_file.py”, esta función genero tres carpetas denominadas “logs”, “sequences” y “checkpoints” como podemos apreciar en la figuras 3.6.

Figura 3.6. Estructura de directorio de los conjuntos de entrenamientos



Fuente. Los autores.

Estas carpetas alojaran archivos con tres extensiones distintas (archivo.npy, archivo.log, archivo.hdf5) donde se detallaran características específicas de los modelos en cada una de sus etapas y épocas de entrenamiento.

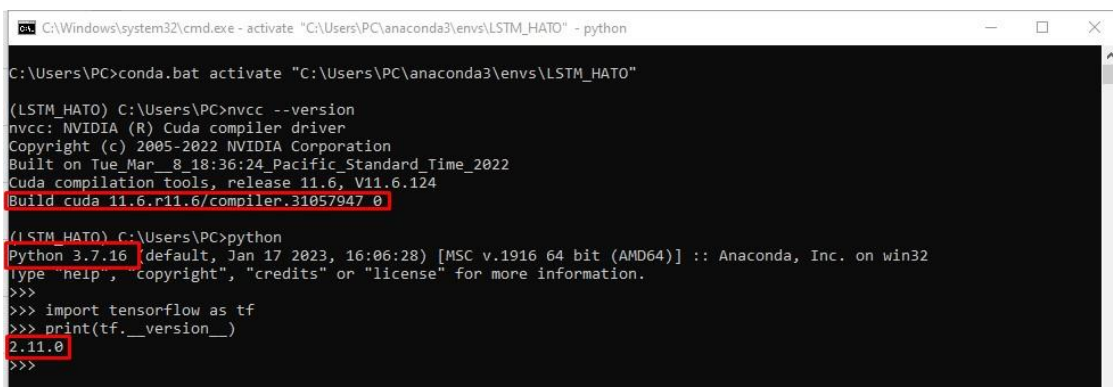
Las siguientes dos etapas modelado y evolución del modelo comprendieron lo que fue el objetivo 3, Entrenar un modelo de red neuronal convolucional para la identificación y predicción de parto porcino.

Donde se explica paso a paso lo que se relizo para completar el objetivo. Aquí también se muestra la metodología SABOR-IBM covinada en las dos etapa

3.4. MODELADO

Como punto de partida del modelado se instaló el intérprete de "Python", luego se procedió a crear un entorno de desarrollo denominado "LSTM-HATO" en el ID "Anaconda Navigator" que consta con las siguientes características, Python versión "3.7.16", tensor flow versión "2.11.0", Cuda versión "11.6.r11.6" compatible con la tarjeta gráfica "NVIDIA GeForce RTX 3070 TI" como se evidencia en las figuras 3.7 y 3.8.

Figura 3.7. Vista por consola de las características del entorno LSTM_HATO



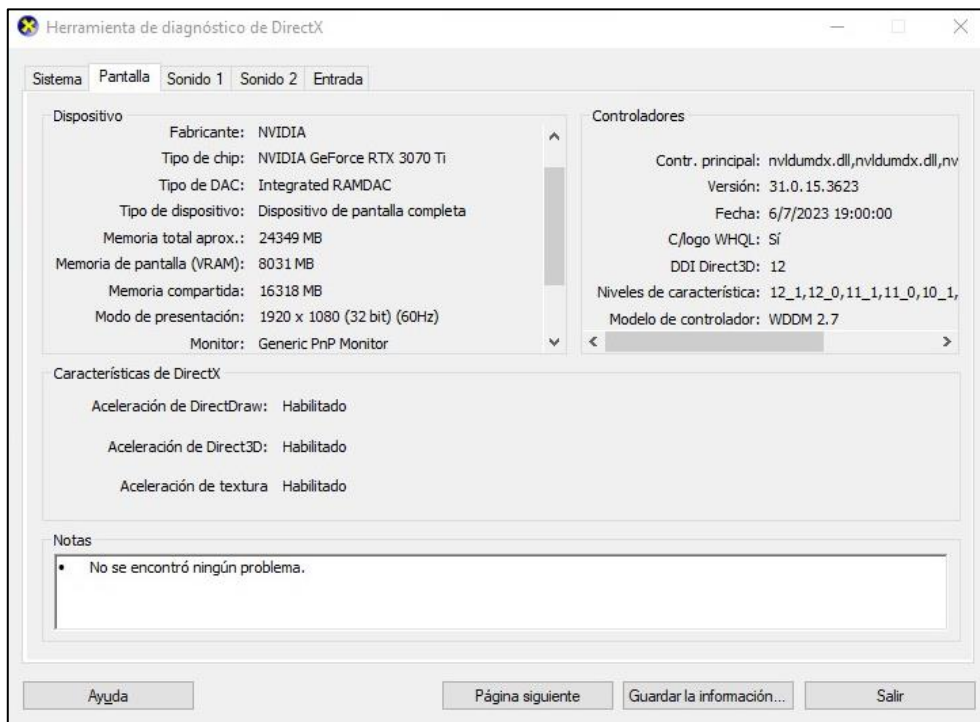
```

C:\Windows\system32\cmd.exe - activate "C:\Users\PC\anaconda3\envs\LSTM_HATO" - python
C:\Users\PC>conda.bat activate "C:\Users\PC\anaconda3\envs\LSTM_HATO"
(LSTM_HATO) C:\Users\PC>nvcc --version
nvcc: NVIDIA (R) Cuda compiler driver
Copyright (c) 2005-2022 NVIDIA Corporation
Built on Tue_Mar_ 8_18:36:24_Pacific_Standard_Time_2022
Cuda compilation tools, release 11.6, V11.6.124
Build cuda 11.6.r11.6/compiler.31057947.0

(LSTM_HATO) C:\Users\PC>python
Python 3.7.16 (default, Jan 17 2023, 16:06:28) [MSC v.1916 64 bit (AMD64)] :: Anaconda, Inc. on win32
Type "help", "copyright", "credits" or "license" for more information.
>>>
>>> import tensorflow as tf
>>> print(tf.__version__)
2.11.0
>>>
  
```

Fuente. Los autores

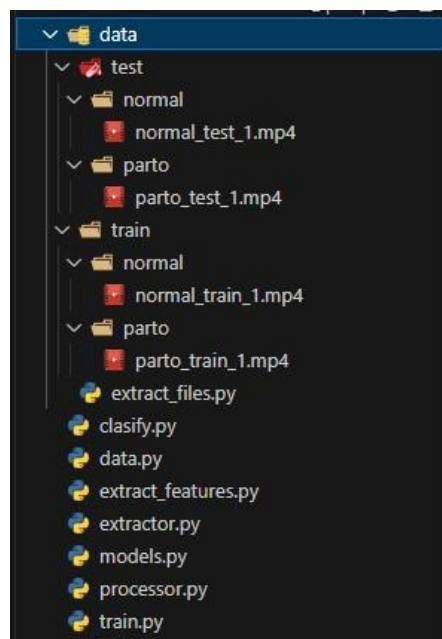
Figura 3.8. Especificaciones técnicas de Hardware Grafico



Fuente. Los autores

Una vez hecha las respectivas configuraciones previas en el entorno de desarrollo se definió la siguiente estructura de directorios:

Figura 3.9. Estructura del directorio del folder principal del proyecto



Fuente. Los autores

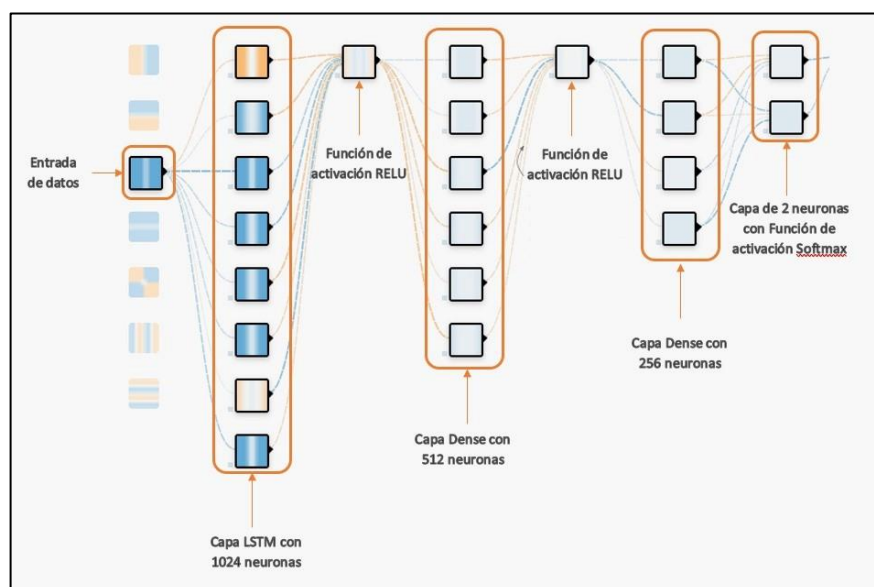
3.4.1. EVALUACIÓN

- **Data:** es un directorio que contiene dos subdirectorios que están denotados como conjunto de entrenamiento “train” y conjunto de prueba “test”.
- **Train:** está compuesto por dos directorios que separan claramente las características de “parto” y “normal” a través de los archivos.mp4 alojados en ellos. Estos archivos fueron la base de aprendizaje de los modelos en la fase de entrenamiento.
- **Test:** directorio compuesto por la misma estructura del fichero “train” con la diferencia que los archivos contenidos en estos directorios sirvieron como base para la evaluación del comportamiento de los modelos conforme a cada una de las épocas de entrenamiento.

3.4.2. CODIFICACIÓN

- **Models.py:** en este archivo se describe la arquitectura de la red neuronal, misma que está compuesta por una capa LSTM con 1024 neuronas y dos capas densas cada una con 512 y 128 neuronas respectivamente con función de activación relu y por último una capa densa con dos neuronas (capa de salida) con la función de activación softmax como se puede apreciar en la figura 3.10 y 3.11.

Figura 3.10. Arquitectura del modelo



Fuente. Los autores, desde la herramienta playground de TensorFlow

Figura 3.11. Código Python de la topología de la red.

```

54
55     def lstm(self):
56
57         # Model.
58         model = Sequential()
59         model.add(LSTM(1024, return_sequences=False,
60                       input_shape=self.input_shape,
61                       dropout=0.4))
62         model.add(Dense(512, activation='relu'))
63         model.add(Dropout(0.5))
64         model.add(Dense(256, activation='relu'))
65         model.add(Dropout(0.2))
66         model.add(Dense(self.nb_classes, activation='softmax'))
67
68         return model
69

```

Fuente. Los autores

- **Extract_file.py:** script de Python que generó un proceso recursivo de generación de imágenes de dimensión 1280x720 pixeles almacenándolos en los mismos directorios que se ubican los archivos.mp4, este script dio como origen tres ficheros más y un archivo.csv como se lo pudo evidenciar en el proceso de preparación para el entrenamiento del modelo en el apartado de la etapa de preparación de los datos, me muestra código completo (**Anexo 5-a**).
- **Data.py:** en este block de código cargan en memoria la ruta a los archivos extencion.jpg que se encuentran en las carpetas test y train junto con el data_file.csv generado en la ejecución del extract_file.py
- **Extract_features.py:** este conjunto de sentencias se encarga de volcar en memoria la información que se obtiene del apartado anterior (data.py) en formato numpy array (archivo.npy)
- **Prosesor.py:** es una función que se encarga de cargar las imágenes y colocarlas en arreglos aplicando un proceso de normalización; se divide cada pixel de las imágenes por el valor mayor que pueden tomar (255), esta es referenciada desde el archivo data.py

3.5. EVALUACIÓN DEL MODELO

3.5.1. ENTRENAMIENTO

- **Train.py:** script de Python que consta de la siguiente sintaxis, Rango de imágenes (75) clases (2, conjunto train y conjunto test) y la resolución de las imágenes en pixeles, bloque de código completo (**Anexo 5-b**).

Figura 3.12. Sentencia python del script train.py

```
script de shell
1 python train.py 75 2 720 1280
```

Fuente. Los autores

una vez se ejecuta este script se iniciará un proceso iterativo en el cual se mostrara las métricas de epoch, accuracy, loss, val_accuracy, val_loss como se muestra en el anexo (**anexo 5**) dando así una vista preliminar de la evolución del modelo no obstante en paralelo se ejecuta una estructura condicional la cual establece que si el modelo de la época actual tiene mejor (accuracy) y menor (val_loss) lo guardara como un archivo de extensión .hdf5 en el folder (checkpoints) y sabiendo que el ultimo que se generó es el mejor ajustado con las datos de entrada suministrados

3.5.2. PRUEBA

- **Clasify.py:** Como dato importante para llevar a cabo la ejecución de esta sentencia hay que mover el modelo de extensión.hdf5 de la carpeta (checkpoints) a el folder principal (LSTM-HATO), teniendo en cuenta que el modelo con mejores métricas será el elegido. Una vez realizado este proceso, se colocan como argumentos la dimensión de FPS (75) el número de clases (conjunto de entrenamiento y de prueba) el modelo entrenado más eficiente que guardo el entrenamiento previamente realizado y como último argumento se le pasa un archivo de extensión.mp4 mismo que no es conocido por el modelo, para que evalué cada 3 segundos lo que acontece en el video

Figura 3.13. Sentencia python del bloque de código clasify.py

```
script de shell
1 python clasify.py 75 2 lstm-features.hdf5 video_file.mp4
```

Fuente. Los autores

Bloque de código clasifly.py en anexo **(Anexo 5-c)**.

Como resultado de la compilación de esta sentencia de código nos entregó un archivo de video de extensión result.avi el cual en su parte superior izquierda consta de dos etiquetas determinando el estado del contexto que se está desarrollando con el nombre de parto el cual se superpone o encabeza la columna cuando dicho porcentaje es mayor al 0.50 y de esta manera el modelo determina que la cerda se encuentra en labor de parto.

Figura 3.14. Resultados de las pruebas preliminares



Fuente. Los autores

3.6. IMPLEMENTACIÓN DEL MODELO (DESPLIEGUE)

En esta etapa final, se logró el desarrollo del objetivo 4 que consistió en implementar mecanismo de emisión de alertas en el hato porcino para la predicción de parto, como parte del despliegue se optó por implementar un bot plataforma de mensajería Telegram como mecanismo de alerta, dado que este es una solución más versátil a comparación del diseño de un sitio web.

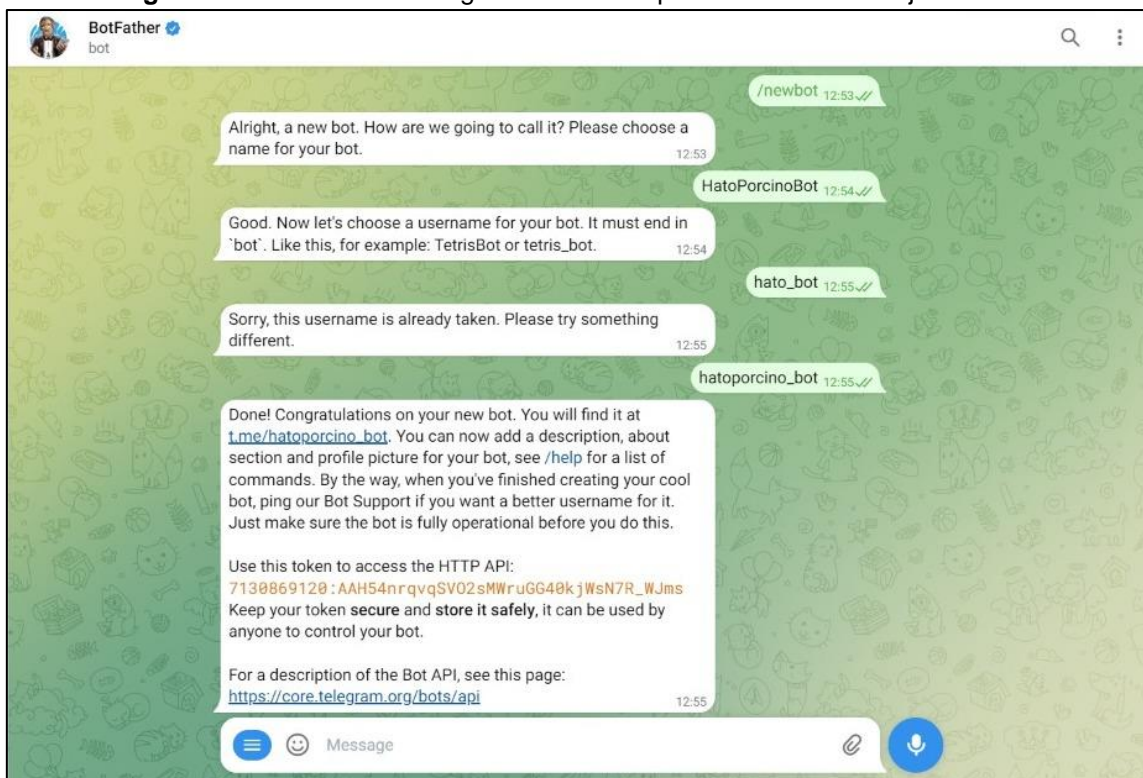
El primer paso consistió en acceder a la aplicación de Telegram y localizar al "BotFather". Dicho bot está especialmente diseñado para crear y configurar bots dentro de esta plataforma de mensajería instantánea.

Figura 3.15. BotFather de Telegram



Fuente. Los autores

Después de identificar el chat, se ingresó el comando /newbot para iniciar el proceso de creación de un nuevo bot. El BotFather, que es el bot oficial de Telegram para crear y gestionar bots, solicitó un nombre para el nuevo bot. Se decidió llamar al bot "HatoPorcinoBot". Luego de ingresar el nombre, BotFather pidió crear un nombre de usuario único para el bot. Después de ingresar un nombre de usuario, en este caso, "hatoporcino_bot", BotFather validó que el nombre no estuviera en uso por otro bot y procedió a crear el nuevo bot con ese nombre de usuario.

Figura 3.16. Interfaz de Telegram con los respectivos comandos ejecutados

Fuente. Los autores

Luego de completar el proceso de creación del bot, BotFather mostró un mensaje que incluía un token o identificador único para el bot que se creó. No obstante, el mensaje también proporcionaba un enlace a la documentación de la API telegram, donde se detallan las funciones disponibles para la implementación con varios intérpretes, incluido Python. La cual fue de vital importancia debido a la popularidad y la robustez del lenguaje de programación. Una vez realizada la identificación de las herramientas otorgadas por la API de Telegram se implementó de la siguiente manera la emisión de alertas en el script denominado clacify.py como lo muestra la figura 3.17.

Figura 3.17. Código Python para el envío de notificaciones mediante el chat

```
token = "7130869120:AAH54nrqvqSV02sMWruGG40kjWsN7R_WJms"
url = "https://api.telegram.org/bot"+token+"/sendMessage"
corpus = {"chat_id": "1743337476", "text": "Alerta de parto en proceso "+str(datetime.datetime.now())}
if float(prediction[0,1])>=0.70:
    response = requests.post(url, data=corpus)
```

Fuente. Los autores

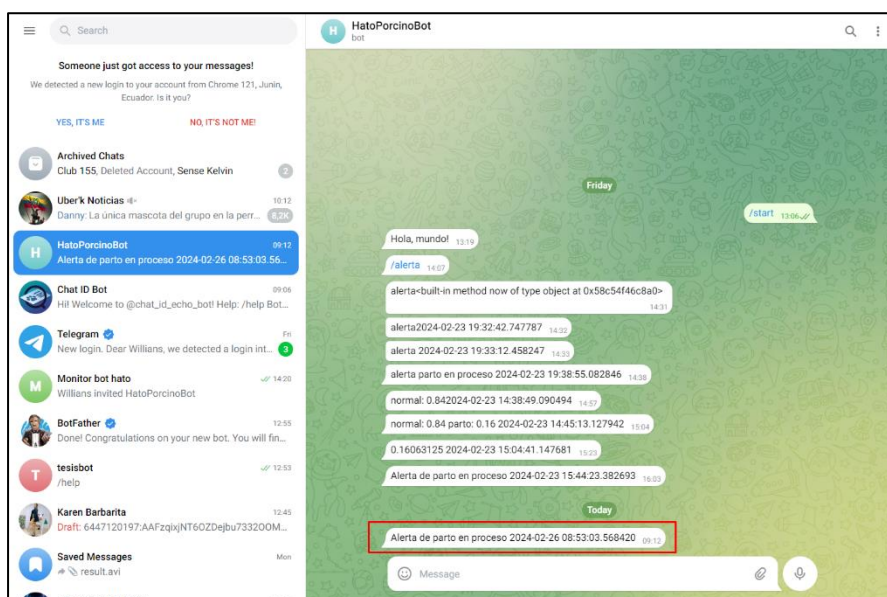
Se definió una variable tipo “string” que almacena el token de acceso del bot de Telegram que se utiliza para enviar mensajes. Este token identifica el bot en la plataforma de Telegram. Posteriormente se contruyó la URL de la API de Telegram para enviar un mensaje utilizando el token proporcionado. La URL se construye concatenando el token al final de la URL base de la API de Telegram.

Definición del contenido del mensaje, se crea un diccionario llamado corpus que contiene los detalles del mensaje a enviar. El diccionario incluye el chat_id (identificador del chat al que se envía el mensaje) y el texto del mensaje. En este caso, el texto del mensaje incluye una alerta de parto en proceso y la hora actual obtenida mediante `datetime.datetime.now()`.

Como paso final se realizó una condición para verificar si la predicción realizada (variable prediction) supera un umbral de confianza del 70%.

Si la probabilidad de la predicción de que ocurra el evento supera el umbral del 70%, se envía una solicitud POST a la URL de la API de Telegram, enviando el contenido del mensaje definido en el diccionario corpus tan como se observa en la figura 3.18.

Figura 3.18. Prueba de la alarma enviada al chat de Telegram



. Fuente. Los autores

CAPÍTULO IV. CONCLUSIONES Y RECOMENDACIONES

4.1. CONCLUSIONES

- El proceso de clasificación de vídeos de partos en cerdas porcinas y el análisis de los datos han revelado patrones de comportamiento clave, proporcionando una base sólida para mejorar la predicción y eficiencia en la industria porcina.
- Las técnicas de Redes Convolucionales (CNN) y Long Short-Term Memory (LSTM) se identificaron como las más adecuadas para el modelo de predicción parto porcino a través de videos, aprovechando la capacidad de las CNN para detectar características visuales y de las LSTM para analizar secuencias temporales. Este enfoque integrado mejora significativamente la precisión y reducción de falsos positivos y negativos.
- Se desarrolló y entrenó un modelo diseñado específicamente para la identificación y predicción del parto porcino. Este modelo cuenta con la capacidad de reconocer patrones visuales en frame de vídeos del proceso de parto, lo que permitió realizar predicciones precisas y oportunas.
- Se creó un sistema eficaz de alertas de parto porcino a través de un bot de Telegram, que notifica a los cuidadores cuando una cerda alcanza un umbral de confianza superior al 70%. Este mecanismo proporciona detalles precisos como la hora y fecha del evento, asegurando una comunicación rápida y efectiva que facilita una intervención oportuna y mejora significativamente el manejo durante el parto.

4.2. RECOMENDACIONES

- Es fundamental asegurarse de que el hardware utilizado para capturar los vídeos ofrezca una calidad de imagen óptima, ya que una alta calidad visual es crucial para captar con precisión los detalles esenciales durante el entrenamiento del modelo. Ya que esto impacta directamente en la precisión de la predicción.
- Se sugiere seguir investigando y aplicando las redes CNN y LSTM, dada su efectividad demostrada en el proyecto. Además, es aconsejable explorar su uso en otros ámbitos, aprovechando su capacidad para aprender y adaptarse a patrones complejos en diversos tipos de datos.
- Invertir en un hardware de primera línea, con una CPU y GPU de alto rendimiento, así como una mayor cantidad de memoria RAM y almacenamiento rápido, ayudará a aprovechar al máximo el potencial del modelo y mejorar su eficacia en futuras implementaciones.
- Es importante que los encargados del hato porcino cuenten con acceso a Internet y presten atención a las notificaciones del bot de Telegram para asegurar una intervención oportuna durante el proceso de parto, lo que contribuirá significativamente a la salud y bienestar de las cerdas y sus crías.

BIBLIOGRAFÍA

- Alemán, D. (2017). Técnicas de inteligencia artificial aplicadas a problemas de ingeniería civil. *Revista de Arquitectura e Ingeniería*, 11(3), 1–7.
<https://www.redalyc.org/articulo.oa?id=193955164005>
- Arroyo, P. 2019. “Evaluación Del Comportamiento Materno En Cerdas . Componentes Genéticos y Ambientales.” UNIVERSIDAD NACIONAL DE LA PLATA.
- Berryhill, Jamie, and Rob Clogher. 2020. “LA INTELIGENCIA ARTIFICIAL.” *Intel* (<https://dx.doi.org/10.1787/726fd39d-en>).
- Bonilla, C. (2020). *Redes Convolucionales* [Universidad de Sevilla].
[https://idus.us.es/bitstream/handle/11441/115221/TFG_DGMyE_Bonilla Carrion%2C Carmelo.pdf?sequence=1&isAllowed=y](https://idus.us.es/bitstream/handle/11441/115221/TFG_DGMyE_Bonilla_Carrion%2C_Carmelo.pdf?sequence=1&isAllowed=y)
- Cañal, C. M. (2017). Clasificación de movimiento de animales mediante Redes Neuronales Clasificación de movimiento de Neuronales. Universidad de Sevilla. *Escuela Politécnica Superior de Sevilla*. Universidad de Sevilla. Escuela Politécnica Superior de Sevilla.
- CES. (2019a). *Datos institucionales Datos generales de la carrera Computación* (pp. 1–135). CES (Consejo de Educación Superior).
<http://www.espam.edu.ec/recursos/sitio/carreras/computacion/ComputacionRediseño.pdf>
- CES. (2019b). *Datos institucionales Datos generales de la carrera Veterinaria*.
<http://www.espam.edu.ec/recursos/sitio/carreras/veterinaria/VeterinariaRediseño.pdf>
- Delgado, S. (2022). Aprende Python. Aprende Python, python, 1–488.
https://aprendepython.es/_downloads/907b5202c1466977a8d6bd3a2641453f/aprendepython.pdf

- Egüez, G. 2020. "Análisis Del Desempeño de Redes Neuronales Artificiales En La Reconstrucción de Datos Pluviométricos de La Ciudad de Quito." *Creative Common*.
- Espinosa, J. J. (2020). Aplicación de metodología CRISP-DM para segmentación geográfica de una base de datos pública. *Ingeniería Investigación y Tecnología*, 21(1), 1–13. <https://doi.org/10.22201/fi.25940732e.2020.21n1.008>
- Estatuto ESPAM MFL, Escuela Superior Politécnica Agropecuaria De Manabí Manuel Félix López 4 (2019). <http://www.espam.edu.ec/recursos/sitio/espam/ESTATUTOESPAMMFL20190910-CES.pdf>
- Gavilanes, D. (2020). Sistema de monitoreo apícola mediante el uso de redes neuronales artificiales para identificar la variación de población. *Universidad Técnica de Ambato*. <https://repositorio.uta.edu.ec/bitstream/123456789/31507/1/t1729ec.pdf>
- Huber, S., Wiemer, H., Schneider, D., & Ihlenfeldt, S. (2019). DMME: Data mining methodology for engineering applications - A holistic extension to the CRISP-DM model. *Procedia CIRP*, 79, 403–408. <https://doi.org/10.1016/j.procir.2019.02.106>
- Jara, I., & Ochoa, J. M. (2020). Usos y efectos de la Inteligencia Artificial en educación. Banco Interamericano de Desarrollo (Grupo BID), 3–27. https://siip.produccion.gob.bo/noticias/files/BI_19062020cb063_4artifBID.pdf
- LOES. (2018). Ley Orgánica de Educación. *Boletín Oficial Del Estado*, 106, 17158–17207. <http://www.conocimiento.gob.ec/wp-content/uploads/2015/07/Ley-Organica-de-Educacion-Superior-LOES.pdf>
- Mancilla-Vela, G., Leal-Gatica, P., Sánchez-Ortiz, A., & Vidal-Silva, C. (2020). Factors associated to student success in online learning: a data

mining analysis. *Formacion Universitaria*, 13(6), 23–36.
<https://doi.org/10.4067/S0718-50062020000600023>

Muñoz, D., D. Franco, and C. Mendoza. 2020. “Etapas de Adquisición de Tecnología y Conocimiento En El Sector Ganadero de Tipo Exportación.” *Revista Espacios* 41:68–75. doi: 10.48082/espacios-a20v41n49p05.

Marín, K. R., Marina, L., & Aristizábal, R. (2019). *Factores de manejo asociados a la mortalidad en lechones lactantes en granja porcícola la vitrina* [Corporación Universitaria Lasallista].
http://repository.unilasallista.edu.co/dspace/bitstream/10567/2435/1/Mortalidad_lechones_lactantes.pdf

Ocaña-Fernández, Y., Valenzuela-Fernández, L. A., & Garro-Aburto, L. L. (2019). Inteligencia artificial y sus implicaciones en la educación superior. *Propósitos y Representaciones*, 7(2), 536–552.
<https://doi.org/10.20511/pyr2019.v7n2.274>

Rodríguez, m. F. B., Molina, w. B. L. (2020). Análisis y mejora de los procesos y procedimientos para la docencia de las unidades de docencia, investigación y vinculación de la Espam “MFL” [escuela superior politécnica agropecuaria de Manabí Manuel Félix López]. *La escuela superior politécnica agropecuaria de Manabí Manuel Félix López*.
<https://repositorio.espam.edu.ec/bitstream/42000/1253/1/ttap04d.pdf>

Rodríguez, J. 2021. ““ Evaluación Morfológica Del Cordón Umbilical y Su Relación Con Indicadores de Vitalidad En Lechones Neonatos Del Cerdo Entrepelado (Ts ´ Üdi Xirgo) Del Valle Del Mezquital .”” 1–21.

Rodríguez, H. 2020. “Sistemas de Tutoría Inteligente y Su Aplicación En La Educación Superior.” *REVISTA Iberoamericana Para La Investigacion y El Desarrollo Educativo* 12.

- Sarmiento, José Luis. 2020. "Aplicaciones de Las Redes Neuronales y El Deep Learning a La Ingeniería Biomédica Applications of Neural Networks and Deep Learning to Biomedical Engineering." 19(4):1–18.
- Sicilia, M. Á., Valvanera, P., Asesor, C., Profesor, Lucania, M., Andrés, Á., & Eraso, N. (2018). Enfoque metodológico ágil para adopción de soluciones de Inteligencia Artificial y Big Data Trabajo de Grado Enfoque metodológico ágil para adopción de soluciones de Inteligencia Artificial y Big Data 2.
- Valderrama-Purizaca, Frank Jesús, Daniel Armando Chávez-Barturen, Sócrates Pedro Muñoz-Pérez, Victor Tuesta-Monteza, Heber Ivan Mejía-Cabrera, Frank Jesús Valderrama-Purizaca, Daniel Armando Chávez-Barturen, Sócrates Pedro Muñoz-Pérez, Victor Tuesta-Monteza, and Heber Ivan Mejía-Cabrera. 2021. "Importancia de Las Redes Neuronales Artificiales En La Ingeniería Civil: Una Revisión Sistemática de La Literatura." *Iteckne* 18(1):71–83.
- Vera, r. A. P. (2019). Sistema web de registro y control de los procesos reproductivos y venta de lechones en la unidad de docencia investigación y vinculación - hato porcino en la Espam-MFL [escuela superior politécnica agropecuaria de Manabí Manuel Félix López]. <https://repositorio.esпам.edu.ec/bitstream/42000/974/1/ttc17.pdf>

ANEXOS

ANEXO 1.

**OFICIO DE SOLICITUD DE COLABORACIÓN EN EL HATO PORCIÓN PARA
EL DESARROLLO DE DEL SISTEMA DE ALERTA TEMPRANA.**

.



**ESCUELA SUPERIOR POLITÉCNICA AGROPECUARIA
DE MANABÍ MANUEL FÉLIX LÓPEZ**

Carrera de Medicina Veterinaria



Memorando Nº: ESPAM MFL-DCMV-2022-573-M
Calcuta, 05 de julio de 2022

PARA: Mg. Ramón Joffre Moreira Pico
DIRECTOR CARRERA DE COMPUTACIÓN

ASUNTO: Solicitud de colaboración de estudiantes con trabajo de integración curricular para el Hato Porcino

Reciba atento saludos y éxitos en su gestión como Director de la Carrera de Computación de nuestra institución.

En virtud de conocer la excelencia académica de los estudiantes de la carrera que acertadamente Usted dirige, solicito de la manera más cordial que bajo su digno intermedio se gestione la colaboración de los estudiantes, Karen Barbarita Álava Zambrano y Williams Eduardo Basurto Vidal, estudiantes de noveno semestre de la Carrera de Computación de la ESPAM MFL, para el desarrollo del trabajo de integración curricular dentro de Carrera de Medicina Veterinaria, en la unidad de docencia investigación y vinculación Hato Porcino, en vista de que se han venido acarreado problemas con los procesos de parto porcino, dando así paso al desarrollo de la investigación denominada **SISTEMA DE ALERTA TEMPRANA DE LABOR DE PARTO EN EL HATO PORCINO EMPLEANDO TÉCNICAS DE REDES NEURONALES CONVOLUCIONALES Y LSTM**, designando así al Mg. Mauro Guillen Mendoza como docente de apoyo logístico en el transcurso de la investigación.

Agradeciendo su apoyo y contribución, me suscribo.

Atentamente,



ERNESTO
ANTONIO
HURTADO

Dr. C. Ernesto Antonio Hurtado
DIRECTOR CARRERA DE MEDICINA VETERINARIA, Encargado.

EAH/czc

ANEXO 2.

**REUNIÓN CON LA ENCARGADA DEL ÁREA Y ENTREGA DE DATOS
SOBRE LOS PARTOS DEL HATO PORCINO ESPAM MFL.**



Ilustración 1. Reunión con la encargada del hato porcino sobre la problemática en cuestión

REGISTRO DE NEONATALES

UNIDAD DE DOCENCIA INVESTIGACIÓN Y VINCULACIÓN LAB. DE BIOTECNOLOGÍA DE LA REPRODUCCIÓN ANIMAL Y HATO PORCINO

TÉCNICO RESPONSABLE		CÓDIGO DE LA CERDA		2062		CÓDIGO PADRE				
FECHA DE NACIMIENTO		PP. NACIMIENTO		PP. DESTETE		PARTO. NUMERO				
ESTUDIANTE NEONATAL		1,48				VACUNA MYCOPLASMA				
COLOR	SEXO	PESO AL NACIMIENTO	PESO 8 DIAS	PESO 15 DIAS	PESO 22 DIAS	PESO AL DESTETE 30 DIAS	LECHONES MUERTOS ANTES DEL DESTETE	NUMERO DE LECHONES ACUMULADOS	VACUNA MYCOPLASMA	CODIGO LECHON
*1	Blanco	Hembra	1,29	2,79		3,09	4,21			
*2	Roja	Hembra	1,92	3,07		6,28	7,41			
*3	Blanca	Hembra	1,44	3,02		5,30	6,44			
*4	Blanca	Hembra	1,12	2,55		3,92	3,70			
*5	B/N	Hembra	1,66	2,91		4,74	5,42			
*6	Roja/Blanca	Hembra	1,50	2,58		3,58	4,46			
*7	Blanco	Macho	1,59	2,75		4,28	5,34			
*8	Blanco	Macho	1,59	3,12		4,53	5,19			
*9	Blanco	Macho	1,70	2,76		3,75	2,67			
*10	Blanco	Macho	1,75	3,48		5,30	5,87			
*11	Roja	Macho	1,53	2,85		4,60	5,46			
*12	B/N	Macho	1,43	3,05		4,79	5,16			
*13	Blanco	Macho	1,42	2,60		4,49	4,97			
*14	Blanco	Macho	1,10							
15										
16			1,48	2,86		4,65	5,10			
17										

1 mortinato: - 2 muertos
 + 1
 Aplicación de PABO 26 febrero 2024.

Torreón

Ilustración 2. ficha técnica de partos del área de maternidad

ANEXO 3.

**ADQUISICIÓN E INSTALACIÓN DE CÁMARAS DE VIDEO VIGILANCIA
PARA MONITOREAR A LAS CERDAS EN PROCESO DE PARTO EN LAS
INSTALACIONES DE LA UDIV – HATO PORCINO DE LA ESPAM MFL.**



Ilustración 1. Adquisición de cámaras HIKVISION turbo HD



Ilustración 2. Instalación de la base para las cámaras



Ilustración 3. procesos de conexión y alimentación con el DVR y la fuente centralizada



Ilustración 4. cámaras instaladas y funcionales

ANEXO 4.

**CONJUNTO Y SELECCIÓN DE DATOS DE LOS VIDEOS OBTENIDOS EN
LAS INSTALACIONES DE LA UDIV – HATO PORCINO DE LA ESPAM MFL.**

ANEXO 4-A.
CAPTURA DE LAS CARPETAS QUE CONTIENEN LOS VIDEOS

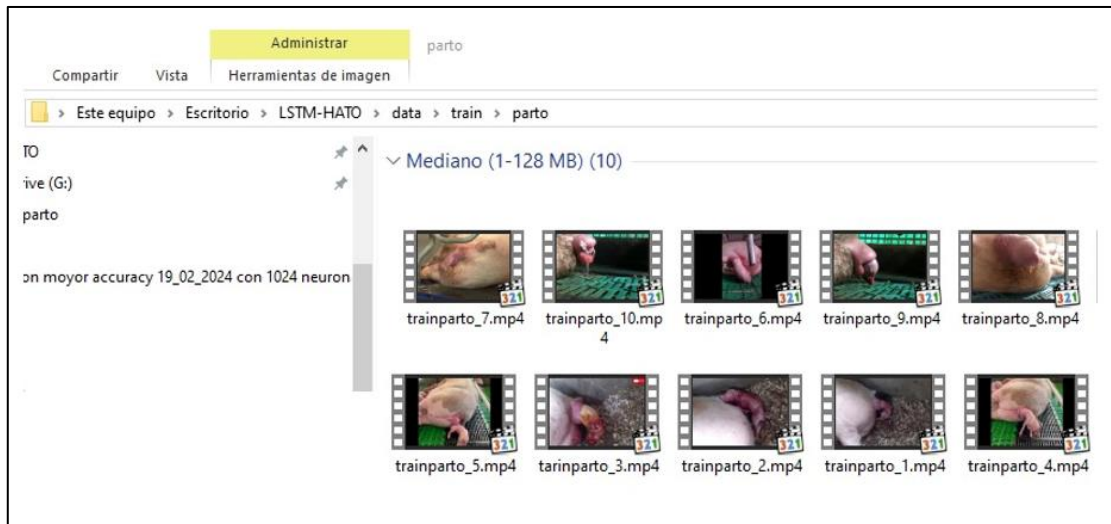


Ilustración 2. captura de la carpeta parto en el conjunto de entrenamiento train

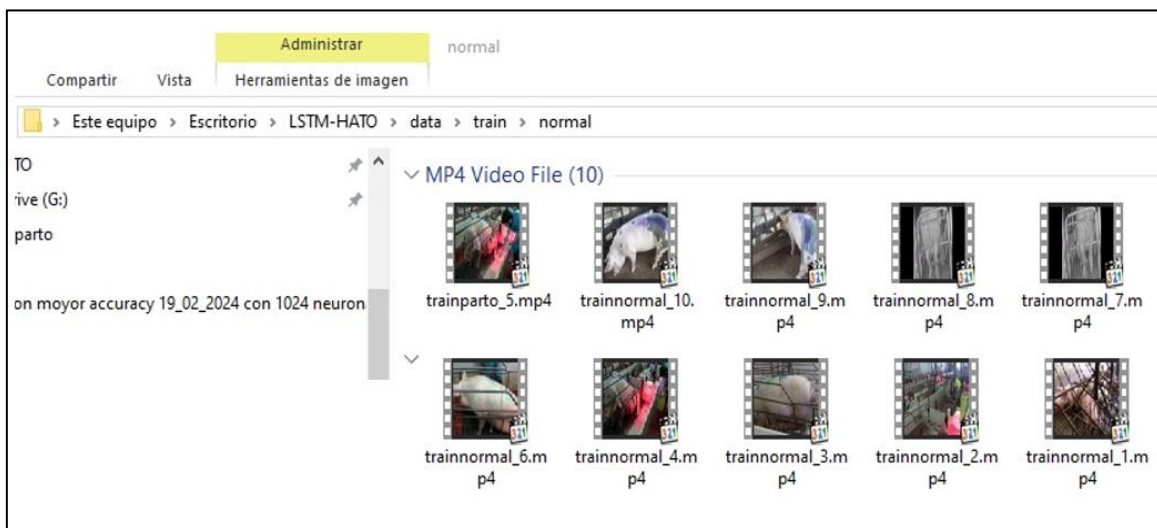


Ilustración 2. captura de la carpeta normal en el conjunto de entrenamiento train

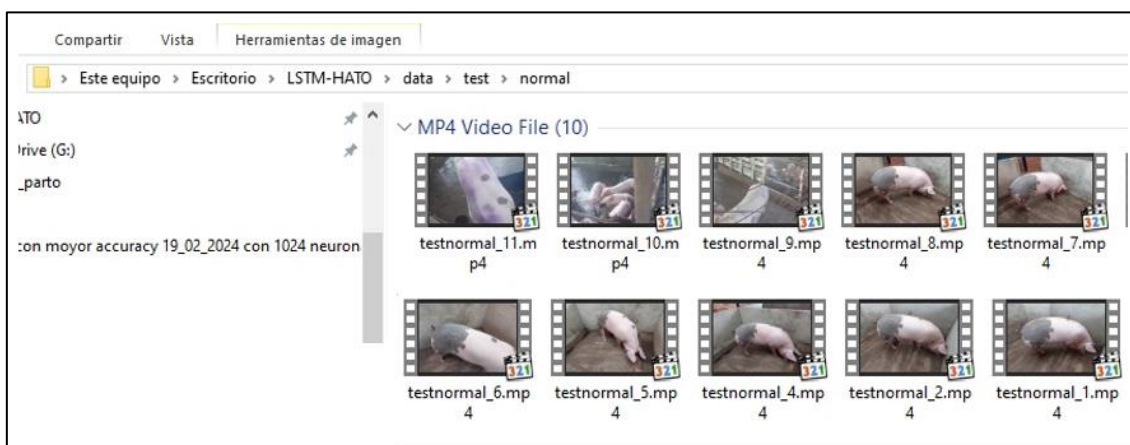


Ilustración 3. captura de la carpeta normal en el conjunto de entrenamiento test

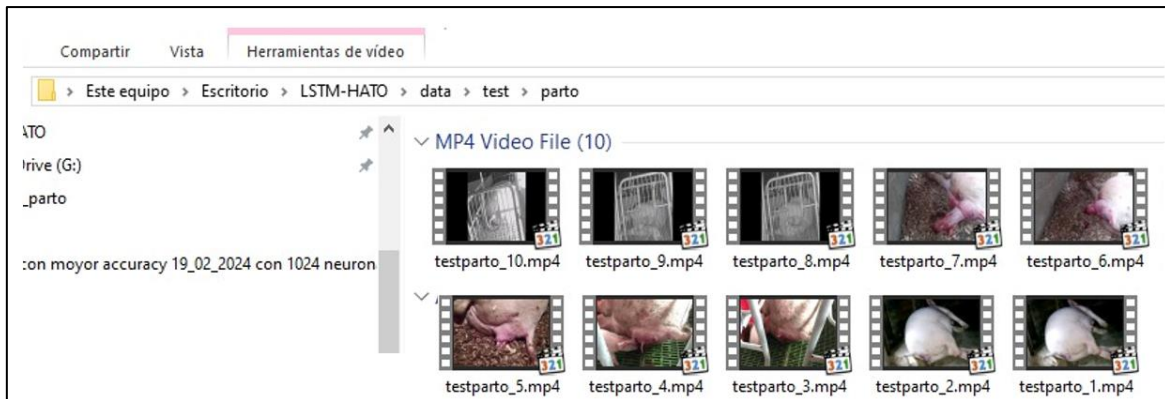


Ilustración 4. captura de la carpeta parto en el conjunto de entrenamiento test



Ilustración 5. Captura de imágenes de los videos almacenados para el previo entrenamiento

ANEXO 4-B.

**CAPTURA DE LA CLASIFICACIÓN DE VIDEOS MEDIANTE LA
HERRAMIENTA DE EDICIÓN DE VIDEOS FILMORA**

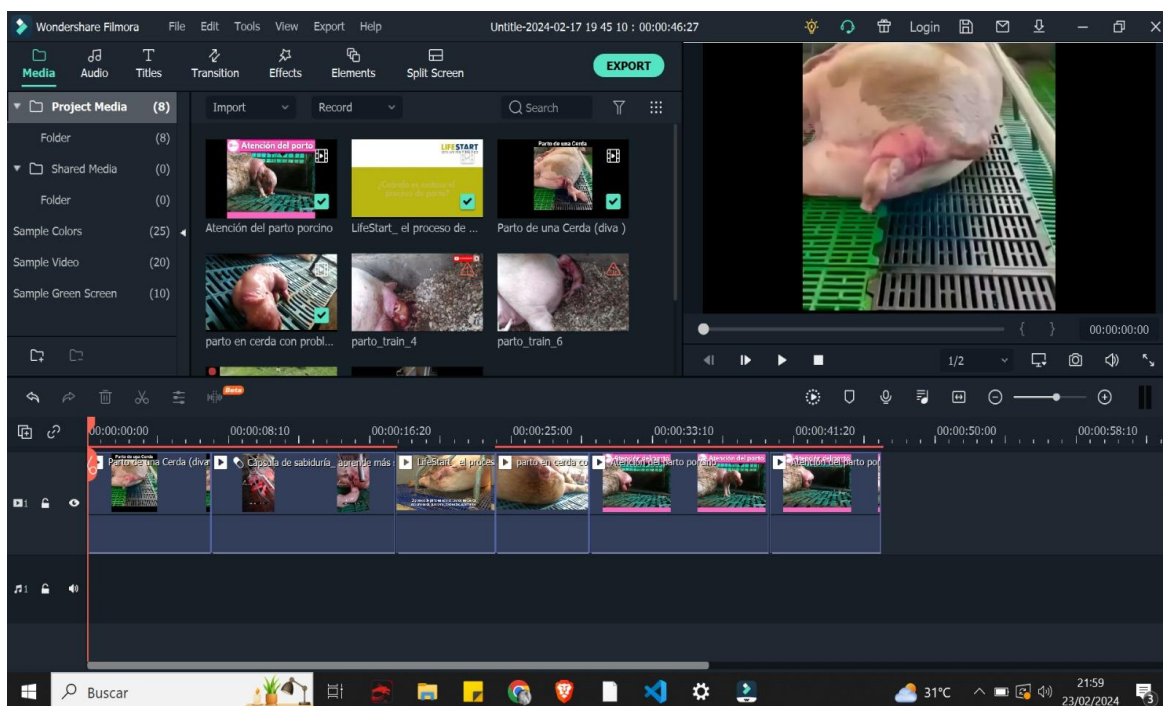


Ilustración 6. Selección de videos para edición de clasificación

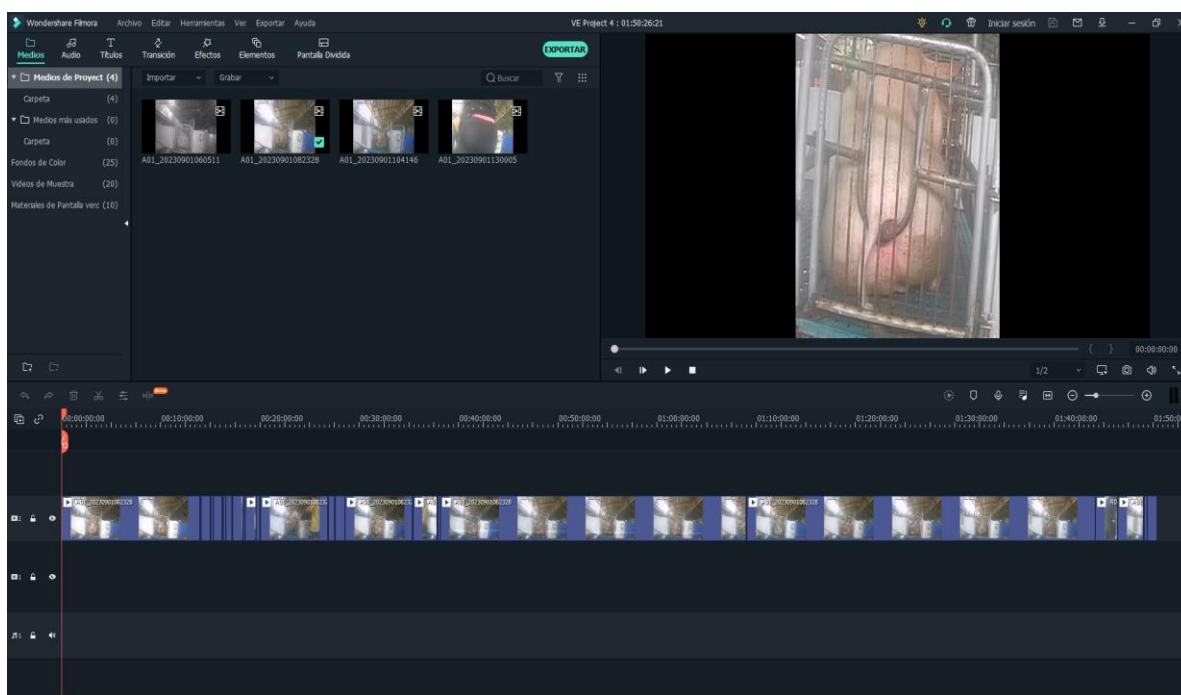


Ilustración 7. Captura de edición de video

ANEXO 5.
CERTIFICACIÓN

República del Ecuador

**ESCUELA SUPERIOR POLITÉCNICA AGROPECUARIA
DE MANABÍ MANUEL FÉLIX LÓPEZ**

Carrera de Medicina Veterinaria



Cajeta, 20 de mayo de 2024

CERTIFICADO

Por medio del presente, certifico que:

Los estudiantes egresados de la Carrera de Computación de la Escuela Superior Politécnica Agropecuaria de Manabí Manuel Félix López, **KAREN BARBARITA ÁLAVA ZAMBRANO** con C.I. 1315943371 y **WILLIAMS EDUARDO BASURTO VIDAL**, con C.I. 1315121317, han colaborado de manera exitosa en el desarrollo del trabajo de la Unidad de Integración Curricular dentro de la Carrera de Medicina Veterinaria, en la Unidad de Docencia, Investigación y Vinculación Hato Porcino. El proyecto llevado a cabo, titulado **"SISTEMA DE ALERTA TEMPRANA DE LABOR DE PARTO EN EL HATO PORCINO EMPLEANDO TÉCNICAS DE REDES NEURONALES CONVOLUCIONALES Y LSTM, "**, se certifica que la colaboración de los estudiantes mencionados ha sido completa y satisfactoria, culminando con éxito el desarrollo del proyecto mencionado.

Se extiende el presente certificado a petición de la parte interesada para los fines que mejor convengan.

Atentamente,



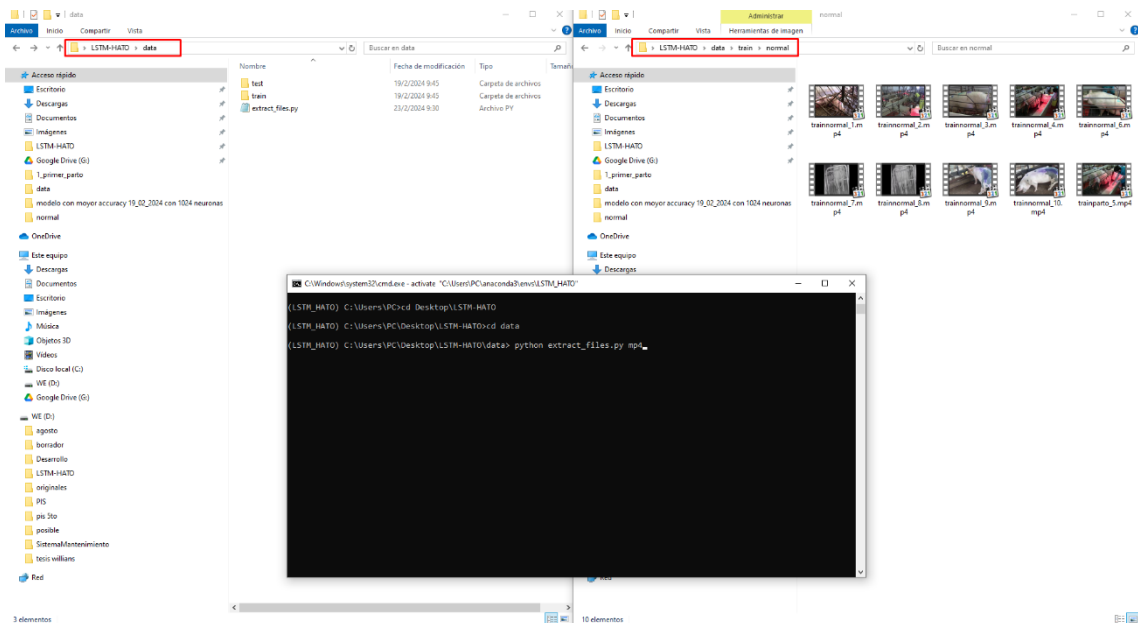
Ernesto Antonio Hurtado
Director Carrera de Medicina Veterinaria

Dr. C. Ernesto Antonio Hurtado
DIRECTOR CARRERA DE MEDICINA VETERINARIA, Encargado.

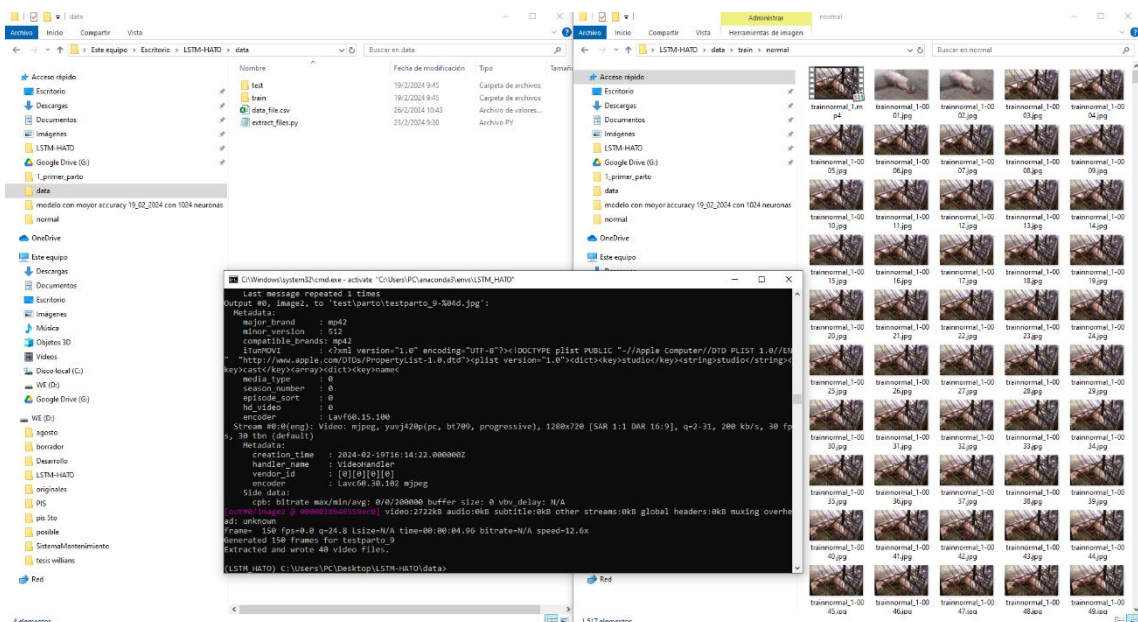
EAH/jpg

ANEXO 6.
CODIFICACIÓN DEL MODELO.

Antes de ejecutar `extract_files.py`, los archivos y su contenido estaban distribuidos como se visualizan a continuación:



posterior a la ejecución del script, la distribución y los datos de los archivos quedaron como se detalla a continuación:



ANEXO 6-A.
CÓDIGO FUENTE DEL ARCHIVO “EXTRACT_FILES.PY”

```
import csv
import glob
import os
import os.path
import sys
from subprocess import call

def extract_files(extenssion='mp4'):

    data_file = []
    folders = ['train', 'test']

    for folder in folders:
        class_folders = glob.glob(os.path.join(folder, '*'))

        for vid_class in class_folders:
            class_files = glob.glob(os.path.join(vid_class, '*' + extenssion))

            for video_path in class_files:
                # Get the parts of the file.
                video_parts = get_video_parts(video_path)

                train_or_test, classname, filename_no_ext, filename = video_parts

                # Only extract if we haven't done it yet. Otherwise, just get
                # the info.
                if not check_already_extracted(video_parts):
                    # Now extract it.
                    src = os.path.join(train_or_test, classname, filename)
                    dest = os.path.join(train_or_test, classname,
```

```

        filename_no_ext + '-%04d.jpg')
    call(["ffmpeg", "-i", src, dest])

    # Now get how many frames it is.
    nb_frames = get_nb_frames_for_video(video_parts)

    data_file.append([train_or_test, classname, filename_no_ext,
nb_frames])

    print("Generated %d frames for %s" % (nb_frames, filename_no_ext))

with open('data_file.csv', 'w') as fout:
    writer = csv.writer(fout)
    writer.writerows(data_file)

print("Extracted and wrote %d video files." % (len(data_file)))

def get_nb_frames_for_video(video_parts):

    train_or_test, classname, filename_no_ext, _ = video_parts
    generated_files = glob.glob(os.path.join(train_or_test, classname,
        filename_no_ext + '*.jpg'))
    return len(generated_files)

def get_video_parts(video_path):

    parts = video_path.split(os.path.sep)
    filename = parts[2]
    filename_no_ext = filename.split('.')[0]
    classname = parts[1]
    train_or_test = parts[0]

```

```
    return train_or_test, classname, filename_no_ext, filename

def check_already_extracted(video_parts):
    """Check to see if we created the -0001 frame of this file."""
    train_or_test, classname, filename_no_ext, _ = video_parts
    return bool(os.path.exists(os.path.join(train_or_test, classname,
                                             filename_no_ext + '-0001.jpg'))))

def main():

    if (len(sys.argv) == 2):
        extract_files(sys.argv[1])
    else:
        print ("Usage: python extract_filese.py [videos extession]")
        print ("Example: python extract_files.py mp4")

if __name__ == '__main__':
    main()
```

ANEXO 6-B.
CÓDIGO FUENTE DEL ARCHIVO “TRAIN.PY”

```
"""
```

```
Train our LSTM on extracted features.
```

```
"""
```

```
from tensorflow.keras.callbacks import TensorBoard, ModelCheckpoint, EarlyStopping,
CSVLogger
```

```
from models import ResearchModels
```

```
from data import DataSet
```

```
from extract_features import extract_features
```

```
import time
```

```
import os.path
```

```
import sys
```

```
def train(data_type, seq_length, model, saved_model=None,
```

```
        class_limit=None, image_shape=None,
```

```
        load_to_memory=False, batch_size=32, nb_epoch=100):
```

```
# Helper: Save the model.
```

```
checkpointer = ModelCheckpoint(
```

```
    filepath=os.path.join('data', 'checkpoints', model + '-' + data_type + \
```

```
        '{epoch:03d}-{val_loss:.3f}.hdf5'),
```

```
    verbose=1,
```

```
    save_best_only=True)
```

```
# Helper: TensorBoard
```

```
tb = TensorBoard(log_dir=os.path.join('data', 'logs', model))
```

```
# Helper: Stop when we stop learning.
```

```
#early_stopper = EarlyStopping(patience=25)
```

```
# Helper: Save results.
```

```
timestamp = time.time()
```

```
csv_logger = CSVLogger(os.path.join('data', 'logs', model + '-' + 'training-' + \
```

```

str(timestamp) + '.log'))

# Get the data and process it.
if image_shape is None:
    data = DataSet(
        seq_length=seq_length,
        class_limit=class_limit
    )
else:
    data = DataSet(
        seq_length=seq_length,
        class_limit=class_limit,
        image_shape=image_shape
    )

# Get samples per epoch.
# Multiply by 0.7 to attempt to guess how much of data.data is the train set.
steps_per_epoch = (len(data.data) * 0.7) // batch_size

if load_to_memory:
    # Get data.
    X, y = data.get_all_sequences_in_memory('train', data_type)
    X_test, y_test = data.get_all_sequences_in_memory('test', data_type)
else:
    # Get generators.
    generator = data.frame_generator(batch_size, 'train', data_type)
    val_generator = data.frame_generator(batch_size, 'test', data_type)
# Get the model.
rm = ResearchModels(len(data.classes), model, seq_length, saved_model)

# Fit!

```

```
if load_to_memory:
    # Use standard fit.
    rm.model.fit(
        X,
        y,
        batch_size=batch_size,
        validation_data=(X_test, y_test),
        verbose=1,
        #callbacks=[tb, early_stopper, csv_logger, checkpointer],
        callbacks=[tb, csv_logger, checkpointer],
        epochs=nb_epoch)
else:
    # Use fit generator.
    rm.model.fit(
        generator,
        steps_per_epoch=steps_per_epoch,
        epochs=nb_epoch,
        verbose=1,
        #callbacks=[tb, early_stopper, csv_logger, checkpointer],
        callbacks=[tb, csv_logger, checkpointer],
        validation_data=val_generator,
        validation_steps=40,
        workers=4)

def main():
    """These are the main training settings. Set each before running
    this file."""

    if (len(sys.argv) == 5):
        seq_length = int(sys.argv[1])
        class_limit = int(sys.argv[2])
```



```

image_height = int(sys.argv[3])
image_width = int(sys.argv[4])
else:
    print ("Usage: python train.py sequence_length class_limit image_height image_width")
    print ("Example: python train.py 75 2 720 1280")
    exit (1)
sequences_dir = os.path.join('data', 'sequences')
if not os.path.exists(sequences_dir):
    os.mkdir(sequences_dir)

checkpoints_dir = os.path.join('data', 'checkpoints')
if not os.path.exists(checkpoints_dir):
    os.mkdir(checkpoints_dir)

# model can be only 'lstm'
model = 'lstm'
saved_model = None # None or weights file
load_to_memory = True # pre-load the sequences into memory
batch_size = 32
nb_epoch = 100
data_type = 'features'
image_shape = (image_height, image_width, 3)

extract_features(seq_length=seq_length, class_limit=class_limit,
image_shape=image_shape)

train(data_type, seq_length, model, saved_model=saved_model,
class_limit=class_limit, image_shape=image_shape,
load_to_memory=load_to_memory, batch_size=batch_size, nb_epoch=nb_epoch)

if __name__ == '__main__':
    main()

```

ANEXO 6-C.
CÓDIGO FUENTE DEL ARCHIVO “CLASIFY.PY”

```
import os
import sys
import cv2
import numpy as np
from data import DataSet
from extractor import Extractor
from tensorflow.keras.models import load_model
import datetime
import requests

if (len(sys.argv) == 5):
    seq_length = int(sys.argv[1])
    class_limit = int(sys.argv[2])
    saved_model = sys.argv[3]
    video_file = sys.argv[4]
else:
    print ("Usage: python classify.py sequence_length class_limit
saved_model_name video_file_name")

    print ("Example: python classify.py 75 2 lstm-features.095-0.090.hdf5
some_video.mp4")

    exit (1)

capture = cv2.VideoCapture(os.path.join(video_file))
width = capture.get(cv2.CAP_PROP_FRAME_WIDTH) # float
height = capture.get(cv2.CAP_PROP_FRAME_HEIGHT) # float

fourcc = cv2.VideoWriter_fourcc(*'XVID')
video_writer = cv2.VideoWriter("result.avi", fourcc, 15, (int(width), int(height)))

# Get the dataset.
data = DataSet(seq_length=seq_length, class_limit=class_limit,
image_shape=(height, width, 3))
```

```
# get the model.
extract_model = Extractor(image_shape=(int(height), int(width), 3))
saved_LSTM_model = load_model(saved_model)

frames = []
frame_count = 0
while True:
    ret, frame = capture.read()

    # Bail out when the video file ends
    if not ret:
        break

    # Save each frame of the video to a list
    frame_count += 1
    frames.append(frame)

    if frame_count < seq_length:
        continue # capture frames until you get the required number for sequence
    else:
        frame_count = 0

    # For each frame extract feature and prepare it for classification
    sequence = []
    for image in frames:
        features = extract_model.extract_image(image)
        sequence.append(features)

    # Classify sequence
```

```

prediction = saved_LSTM_model.predict(np.expand_dims(sequence,
axis=0))

#print(prediction)

values = data.print_class_from_prediction(np.squeeze(prediction, axis=0))

# Obtén el token de API del bot
# token = "6447120197:AAFzqixjNT6OZDejbu7332OOMzVIVjoUPM0"
token = "7130869120:AAH54nrqvqSVO2sMWruGG40kjWsN7R_WJms"

# Envía un mensaje de texto
url = "https://api.telegram.org/bot"+token+"/sendMessage"

#print(prediction[0,1])

corpus = {"chat_id": "1743337476", "text": "Alerta de parto en proceso
"+str(datetime.datetime.now())}

if float(prediction[0,1])>=0.70:
    response = requests.post(url, data=corpus)

# Add prediction to frames and write them to new video
for image in frames:
    for i in range(len(values)):
        cv2.putText(image, values[i], (40, 40 * i + 40),
cv2.FONT_HERSHEY_SIMPLEX, 1.0, (255, 255, 255), lineType=cv2.LINE_AA)
        video_writer.write(image)

frames = []

#capture.release()
video_writer.release()
#cv2.destroyAllWindows()

```