Clasificación del estado de salud del ganado en base a imágenes comprimidas para el ahorro de batería.

Pablo Arango Castaño

Universidad EAFIT

Colombia

parangoc2@eafit.edu.co

Simón Marín

Universidad Eafit

Colombia

smaring1@eafit.edu.co

Mauricio Toro

Universidad Eafit

Colombia

mtorobe@eafit.edu.co

RESUMEN

El manejo óptimo de los datos ha sido una necesidad fundamental para el desarrollo y la mejora de nuevas tecnologías. Desde hace unos cuantos años, se han usado diversos algoritmos para optimizar el consumo de energía, el cual representa un componente vital para la creación de mejores y nuevos sistemas. Incluyendo el de la ganadería de precisión y otras actividades como los videojuegos, identificación visual, fotografía y muchos más.

Este informe tiene como objetivo analizar algoritmos de compresión para la optimización del consumo de batería en ganadería de precisión, además de presentar el diseño y resultados de un algoritmo basado en tablas de hash y árboles, para la compresión y descompresión de imágenes. Con el fin último de hacer uso de estas imágenes comprimidas poniéndolas a prueba al determinar con la ayuda de otro algoritmo de clasificación el estado de salud del ganado.

Palabras clave

Algoritmos de compresión, aprendizaje de máquina,

aprendizaje profundo, ganadería de precisión, salud animal.

1. INTRODUCCIÓN

Cada año, el ser humano consume más de 252 millones de toneladas de carne proveniente del ganado, que combinado con el crecimiento de la población y de este tipo de mercados se estima que anualmente la ganadería presente un crecimiento aproximado del 1.15% a nivel mundial, trayendo a su vez un incremento en la población animal explotada y diversas necesidades consigo [1].

Así comienza a surgir el concepto de Ganadería de Precisión (GdP) un sistema de producción sostenible que ayuda a reducir costos de inversión e impacto ambiental y contribuye a incrementar la producción pecuaria y el bienestar animal. [2]

Uno de los factores claves y el tratado en este informe, es la clasificación del estado de salud del ganado, que muchas veces representa un gran desafío para los ganaderos por la cantidad de cabezas de ganado de las que están a cargo o por simple desconocimiento al momento de clasificar. De esta manera, en el contexto de Ganadería de Precisión, surge la necesidad de comprimir imágenes tomadas al ganado para clasificar por medio de un algoritmo la salud del animal de una manera óptima y rápida. [3]

1.1. Problema

En pocas palabras, el problema con el cual nos enfrentamos en este proyecto es la creación y diseño de un algoritmo de compresión de las imágenes del ganado para alcanzar la mejor tasa de compresión y tiempo posible, sin que la exactitud de la clasificación se afecte en más del 5% al momento de usar el algoritmo que determina la salud del animal.

1.2 Solución

En este trabajo, utilizamos una red neuronal convolucional para clasificar la salud animal, en el ganado vacuno, en el contexto de la ganadería de precisión (GdP). Un problema común en la GdP es que la infraestructura de la red es muy limitada, por lo que se requiere la comprensión de los datos.

Para dar solución a esta problemática, inicialmente se usó una estructura de datos con matrices de la biblioteca NumPy de python para guardar la gran cantidad de datos de los píxeles de las imágenes. Para la compresión de dichos datos se planteó el uso de dos métodos de compresión diferentes. Seam carving o “tallado de costuras” el cual es un método de compresión con pérdidas y la codificación huffman, el cual es implementado para compresión de datos sin pérdidas. Se decidió utilizar estos algoritmos debido a que son ampliamente conocidos, cumplen con los requerimientos que se buscaron y no afectan al algoritmo de clasificación

1.3 Estructura del artículo

En lo que sigue, en la Sección 2, presentamos trabajos relacionados con el problema. Más adelante, en la Sección 3, presentamos los conjuntos de datos y los métodos utilizados en esta investigación. En la Sección 4, presentamos el diseño del algoritmo. Después, en la Sección 5, presentamos los resultados. Finalmente, en la Sección 6, discutimos los resultados y proponemos algunas direcciones de trabajo futuras.

2. TRABAJOS RELACIONADOS

En lo que sigue, explicamos cuatro trabajos relacionados. en el dominio de la clasificación de la salud animal y la compresión de datos. en el contexto del PLF.

2.1 Una plataforma de bienestar animal para sistemas extensivos de producción ganadera

En el año 2018 dos universidades de Grecia, de manera conjunta realizaron un prototipo, el cual consistía en un collar idealizado para el monitoreo del ganado, tanto en características como localización como de aceleración al moverse. Cabe destacar que estos datos que se obtenían se almacenaban en tiempo real en la nube y podían ser observados desde una aplicación móvil. [4]

2.2 Técnicas automatizadas para el seguimiento del comportamiento y bienestar de pollos de engorde y gallinas ponedoras

La calidad del bienestar animal y la salud de los consumidores está estrechamente relacionada con el bienestar animal, por este motivo, el informe publicado por la universidad de Cambridge, se da a la tarea de analizar el monitoreo del comportamiento de pollos de engorde y gallinas ponedoras, además de proporcionar herramientas potenciales para llevar a cabo un monitoreo correcto por medio de sensores inalámbricos portátiles con dispositivos de identificación por

radiofrecuencia, que pueden identificar automáticamente pollos individuales, rastrear la ubicación y el movimiento de los individuos en tiempo real y cuantificar algunos rasgos de comportamiento en consecuencia y tecnología de procesamiento de imágenes, que puede considerarse una herramienta directa para medir comportamientos, especialmente los comportamientos de actividad y la alerta temprana de enfermedades. [5]

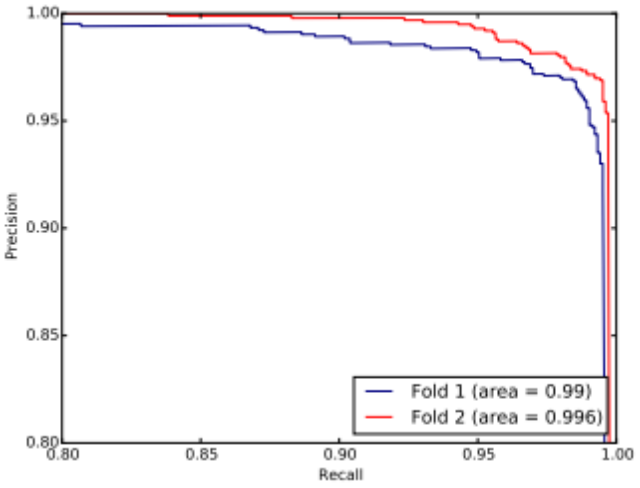
En lo que procesamiento de imágenes respecta se presenta una tabla con materiales y métodos típicos de procesamiento de imágenes automatizado utilizados en el monitoreo del comportamiento de los pollos:

Reference	Camera type	Image pixel and frame rate	Main processing method	Detecting number	Main software
Zaninelli *et al.*, 2018	Thermografic camera	320 × 240; 1 fps	Threshold method and image pattern recognitions	10	LabVIEW, NI Vision Acquisition Software
Li *et al.*, 2018	IR camera	–	Threshold method	8	MATLAB
Pu *et al.*, 2018	Kinect camera	512 × 424; 30 fps	CNN	Commercial flocks	Open CV
Zhuang *et al.*, 2018	CCD camera	640 × 480	K-means clustering and the ellipse model	20	Open CV
Aydin *et al.*, 2017b	Camera	1024 × 768; 5 fps	Background subtraction and feature variables	1	MATLAB
Dawkins *et al.*, 2017	Camera	320 × 240	Optical flow	Commercial flocks	–
Fraess *et al.*, 2016	IR camera	320 × 240; 30 fps	The proportion of pixel changes	34	Etho Vision XT 10
Colles *et al.*, 2016	Web camera	320 × 240; 1 fps	Optical flow	Commercial flocks	MATLAB Python
Youssef *et al.*, 2015	CCD camera	640 × 480; 1 fps	The proportion of pixel changes	45	–
Kashiha *et al.*, 2014	Camera	1280 × 960; 0.5 fps	Binary image and count the pixel proportion	Commercial flocks	eYeNamic

2.3 Localización visual e identificación individual del ganado a través del Deep learning

En 2018 la universidad británica de Bristol se dio a la tarea de solucionar el problema al etiquetar el ganado bovino. En el que la identificación y la trazabilidad del ganado no solo es necesario para las demandas de los consumidores, sino, de hecho, muchos países han introducido marcos legales obligatorios para la protección del bienestar animal y evitar pérdidas y daños físicos que frecuentemente ocurren al marcar al ganado, ya sea con marcas auriculares, tatuajes u otras soluciones electrónicas. Así pues, los investigadores plantean proporcionar una prueba de concepto de que la identificación robusta de un tipo de ganado individual específico (Holstein Friesian) puede ocurrir de forma automática y no intrusiva, utilizando computer vision pipelines alimentadas por arquitecturas estándar que utilizan deep neural networks.

Para la detección y localización de objetos que se producen por las imágenes capturadas, se crean regiones de interés (RoI) de imágenes en forma de cuadros delimitadores, para posteriormente ser pasadas a un proceso posterior, como un algoritmo de seguimiento o un clasificador. De una forma similar se trabajó con videos del ganado, pero las imágenes es lo que nos compete con este informe.[6]



2.4 Integración de servicios en la nube para estudios en el comportamiento de animales de granja usando teléfonos inteligentes como sensores de actividad

En este proyecto se buscó dar solución a la falta de información esencial sobre el comportamiento del ganado que es fundamental para conocer el estado reproductivo o de salud del animal. Para dar solución a este problema se usaron sensores de teléfonos inteligentes y los siguientes tres componentes claves fueron analizados: i) La ubicación obtenida por radiofrecuencia triangulación o por sistema de posicionamiento global. ii) el componente de baja frecuencia del comportamiento como postura del animal (por ejemplo: posición de la cabeza, inclinación del cuello, etc.) y iii) El componente de comportamiento de alta frecuencia (por ejemplo, movimiento de las mandíbulas).

Por la gran cantidad de datos recolectados que debían ser enviados a la nube, una aplicación en Xaramin2 fue desarrollada para medir el

compresibilidad de los datos al reducir la precisión con un algoritmo que actúa de la siguiente manera: (1) eliminando redundancias, es decir, reemplazo de datos redundantes por un intervalo de tiempo durante el cual los valores permanecen constantes para preservar la integridad de los datos, y (2) truncando los datos a 3, 4 y 5 dígitos decimales, de los 6 dígitos decimales originales. Las métricas obtenidas tras la comprensión fueron las siguientes (Figura). [7]

Type of data	Variable number	Data treated	Data size (Mb)	Comp rate [%]
Acceleration in the X, Y, Z axis	3	25,920,000	98.87	4.26
Euler angles of the device	3	25,920,000	98.87	24.13
Attitude quaternion	4	34,560,000	131.84	24.13
Rotation matrix (3×3)	9	77,760,000	296.63	24.13
Gravitational component of 3D acceleration	3	25,920,000	98.87	24.13
User acceleration component of 3D acceleration	3	25,920,000	98.87	24.13
Rotation rate	3	25,920,000	98.87	24.13
Magnetic and true heading	3	25,920,000	98.87	0.01
Magnetometer data	3	25,920,000	98.87	60.79
Position (latitude, longitude)	3	25,920,000	98.87	0.01
Course and speed	2	17,280,000	95.92	99.75
Altitude	1	8,640,000	32.96	99.99

(Figura : Tasa de compresión obtenido para cada categoría de datos recogidos en 24h)

3. MATERIALES Y MÉTODOS

En esta sección, explicamos cómo se recogieron y procesaron los datos y, después, diferentes alternativas de algoritmos de compresión de imágenes para mejorar la clasificación de la salud animal.

3.1 Recopilación y procesamiento de datos

Recogimos datos de *Google Images* y *Bing Images* divididos en dos grupos: ganado sano y ganado enfermo. Para el ganado sano, la cadena de búsqueda era "cow". Para el ganado enfermo, la cadena de búsqueda era "cow + sick".

En el siguiente paso, ambos grupos de imágenes fueron transformadas a escala de grises usando Python OpenCV y fueron transformadas en archivos de valores separados por comas (en inglés, CSV). Los conjuntos de datos estaban equilibrados.

El conjunto de datos se dividió en un 70% para entrenamiento y un 30% para pruebas. Los conjuntos de datos están disponibles en https://github.com/mauriciotoro/ST0245-Eafit/tree/master/proyecto/datasets .

Por último, utilizando el conjunto de datos de entrenamiento, entrenamos una red neuronal convolucional para la clasificación binaria de imágenes utilizando *Teachable Machine* de Google disponible en https://teachablemachine.withgoogle.com/train/image.

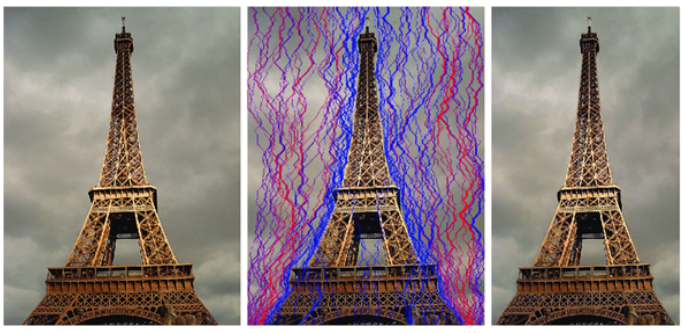
3.2 Alternativas de compresión de imágenes con pérdida

En lo que sigue, presentamos diferentes algoritmos usados para comprimir imágenes con pérdida.

3.2.1 Seam Carving

El algoritmo de Seam Carving o en español “Tallado de costuras” es un método de compresión con pérdidas que funciona principalmente para la reorientación de imágenes para evitar la distorsión del contenido en diferentes dispositivos de varios tamaños como televisores, celulares y computadores.

El algoritmo funciona estableciendo uniones entre píxeles que definen una ruta (Por eso el nombre de costura) en función de la energía de los pixeles comparados con los píxeles vecinos para determinar la ruta de píxeles que se debe borrar (La de menor energía), todo según el contenido de la imagen. La complejidad está dada por el orden o(m n), ancho y largo de la imagen en pixeles. [8]



3.2.2 Transformada de coseno discreta

La transformada de coseno discreto, al igual que el método de compresión de Burrows Wheeler no es un método de compresión como tal, sino uno que facilita el proceso de compresión que está basado en la transformada de Fourier discreta. La transformada de coseno discreta expresa una secuencia finita de varios puntos como resultado de la suma de distintas señales sinusoidales que trabaja con una serie de números finitos reales concentrando la información en pocos coeficientes que son realmente relevantes para la compresión.

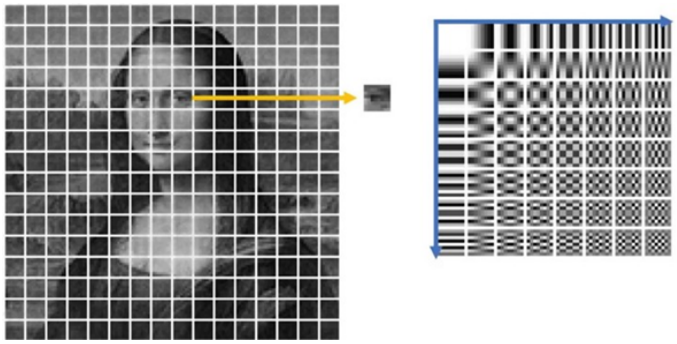
La complejidad del algoritmo dependerá del tipo de DCT que se esté empleando. Siendo las más comunes: i) DCT-I que se usa empleando la fórmula:

$$X_k = \frac{1}{2}(x_0 + (-1)^k x_{N-1}) + \sum_{n=1}^{N-2} x_n \cos\left[\frac{\pi}{N-1} nk\right] \qquad k = 0, \ldots, N-1.$$

Con una complejidad de O(N log2 N) y ii) DCT-II que se usa empleando la fórmula:

$$X_k = \sum_{n=0}^{N-1} x_n \cos\left[\frac{\pi}{N}\left(n + \frac{1}{2}\right)k\right] \qquad k = 0, \ldots, N-1.$$

Con una complejidad de O(N2 log2 N). [9]

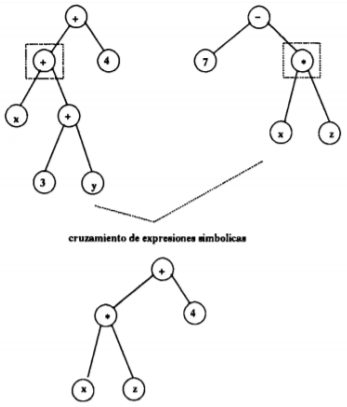


3.2.3 Programación Genética

La programación genética consiste en un algoritmo de compresión con pérdida que ha resultado eficiente para dar soluciones a problemas haciendo uso de generación automática de programación, es decir, sigue el camino de los algoritmos genéticos de modo que busca diferentes caminos para resolver una tarea.

Este algoritmo se representa a través de un árbol, en el cual cada nodo representa un componente o acción específica, siendo una función, una variable, una constante o un procedimiento algunos de los posibles.

Este algoritmo funciona de la siguiente manera: inicialmente, se toma un nodo al azar de cada uno de los dos árboles, posteriormente, se reemplaza el subárbol que tiene como raíz al nodo seleccionado del primer subárbol y, en su lugar, se coloca el subárbol del segundo árbol que tiene como raíz el nodo seleccionado. En la siguiente imagen se ilustra un ejemplo.



La complejidad del algoritmo está dada por O(m,n) ancho y largo de la imagen en píxeles. [10]

3.2.4 JPEG

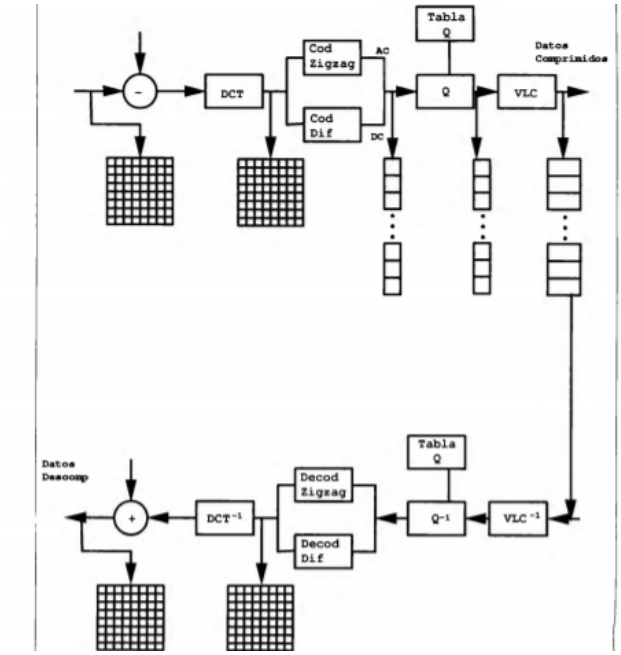
El método JPEG pertenece a los algoritmos de compresión con pérdida, este emplea la transformada discreta del coseno, lo cual lo hace un algoritmo veloz.

El algoritmo comienza dividiendo la imagen en bloques NxN, donde la cantidad de bloques puede beneficiar o no la compresión, sin embargo, la mejor división que se utiliza es 8x8 píxeles.

Posteriormente, se aplica la transformada discreta del coseno (algoritmo explicado anteriormente) donde se busca obtener la mayor cantidad de ceros

posibles mediante un redondeo a números enteros, en este proceso se pierde información.

El hecho de que el JPEG sea un algoritmo de alta compresión implica que se pierda considerablemente la calidad de la imagen, por lo que no se recomienda utilizar este algoritmo para casos en lo que se busque la menor pérdida de calidad posible. [11]



La complejidad del algoritmo es igual a la del algoritmo DTC, es decir O(n^2 * log2 n)

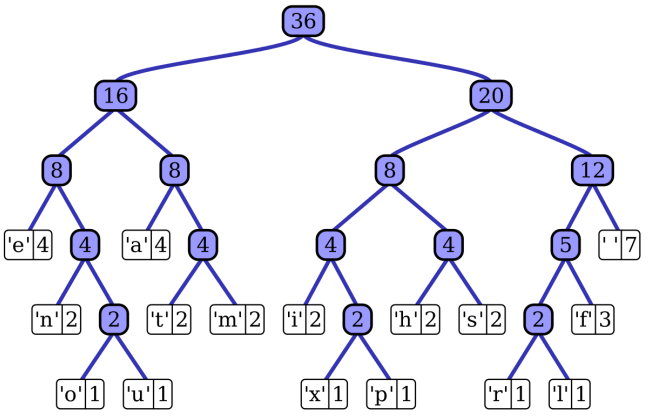
3.3 Alternativas de compresión de imágenes sin pérdida

En lo que sigue, presentamos diferentes algoritmos usados para comprimir imágenes sin pérdida.

3.3.1 *codificación Huffman*

El código Huffman es un algoritmo desarrollado por David A. Huffman que es mundialmente usado para la compresión de datos sin pérdidas. El algoritmo consiste en la creación de un árbol binario en el que los nodos hoja son etiquetados por los caracteres junto a su frecuencia y de forma consecutiva se van uniendo cada pareja de nodos que menos frecuencia sumen, pasando a crear un nuevo nodo intermedio etiquetado con dicha suma. Repitiendo esta acción hasta que no quedan nodos hoja por unir a ningún otro nodo superior.

En términos simples la creación del árbol se realiza de la siguiente manera. Inicialmente, se crea un nodo hoja para cada símbolo, asociado un peso según su frecuencia de aparición que es insertado en la lista ordenada ascendentemente. Posteriormente, mientras haya nodos en la lista realizar los siguientes pasos: I) Eliminar los dos nodos con menor probabilidad de la lista. II) Crear un nuevo nodo interno que enlace a los nodos anteriores, asignándole como peso la suma de los pesos de los nodos hijos. III) Insertar el nuevo nodo en la lista, (en el lugar que le corresponda según el peso). Y por último el nodo que queda al final sera el encargado de crear la raíz del árbol binario. La complejidad de dicho algoritmo está dada por el número de símbolos que se codifican “n” y esta dada por el orden O(nlogn) que es el peor de los casos al ordenar las frecuencias de los símbolos. [12]



3.3.2 Transformación de Burrows-Wheeler

La compresión de Burrows-Wheeler es un algoritmo inventado por Michael Burrows y David Wheeler en 1994 que es usado en técnicas de compresión de datos, pero que en realidad no es una técnica de compresión como tal sino una transformación permutada del orden de caracteres que no altera el valor de los caracteres y que puede ser revertido con facilidad. Al usar dicho algoritmo

se creará una nueva cadena transformada que contendrá múltiples posiciones en las que un mismo carácter esté repetido varias veces en una fila, lo que permite una compresión fácil ya que contiene secuencias de caracteres repetidos. La transformación se realiza ordenando todas las rotaciones del texto en orden lexicográfico, y seleccionando la última columna de letras de todas las cadenas rotadas. Generando una complejidad de orden O(n) en la que la “n” representa la cantidad de caracteres de la cadena. [13]

		F	L
mississippi#		# mississipp	i
ississippi#m		i #mississip	p
ssissippi#mi		i ppi#missis	s
sissippi#mis		i ssippi#mis	s
issippi#miss		i ssissippi#	m
ssippi#missi	$\Longrightarrow$	m ississippi	#
sippi#missis		p i#mississi	p
ippi#mississ		p pi#mississ	i
ppi#mississi		s ippi#missi	s
pi#mississip		s issippi#mi	s
i#mississipp		s sippi#miss	i
#mississippi		s sissippi#m	i

3.3.3 RLE

El algoritmo RLE tiene como fundamento la repetición de elementos consecutivos, el cual consiste en codificar la cantidad de veces que se repetirá el primer elemento seguido del elemento y así sucesivamente hasta que se recorra toda la información.

A continuación un ejemplo: el texto “Naaaaaahhhhhhhhh” se comprimiría como “1N6a9h”, puesto a que hay una “N”, seis “a” y nueve “h”. En el caso anterior, se pasó de tener un texto de 16 caracteres a un nuevo texto de 6 caracteres, lo cual es un muy buen resultado, sin embargo, esta estrategia solo funciona cuando se trata de elementos consecutivos y repetidos, puesto que a que si comprimimos el texto “Mauricio”, obtenemos el texto “1M1a1u1r1i1c1i1o” el cual contiene más caracteres a comparación del inicial.

Debido a esto, el algoritmo RLE solo se utiliza cuando se repiten tres o más elementos consecutivos.

Este algoritmo posee una complejidad O(n), donde n son el número de elementos que se comprimirán y serán recorridos uno por uno para contar el número de repeticiones consecutivas.[14]

3.3.4 LZW

El algoritmo de compresión con pérdida LZW es muy rápido en términos de compresión y descompresión, además, este algoritmo es utilizado en el formato GIF.

El método consiste en que el algoritmo crea un diccionario con los 256 valores de la tabla ASCII, luego, el archivo a comprimir se divide en cadenas de 1 byte y se compara una a una con el diccionario y, si la cadena no está ahí, se agrega. Por lo que se puede deducir que si una cadena es más larga que la palabra más larga del diccionario, esta se debe codificar.

En el caso de las imágenes, el proceso es similar, sin embargo, existen ciertos tipos de imágenes codificadas en bits y no en bytes, por lo que el LZW no funciona para estas imágenes.

Su complejidad es de O(m,n) donde se representan los píxeles como un arreglo bidimensional. [15]

4. DISEÑO E IMPLEMENTACIÓN DE LOS ALGORITMOS

En lo que sigue, explicaremos las estructuras de datos y los algoritmos utilizados en este trabajo. Las implementaciones de las estructuras de datos y los algoritmos están disponibles en Github .

4.1 Estructuras de datos

La estructura de datos utilizada para hacer la compresión de las imágenes son matrices de la biblioteca NumPy de Python. Por ejemplo, el siguiente conjunto de datos entregado en formato csv separado por comas:

```
255,0,120,255,255,255,255,255,255,255
255,120,0,120,120,255,255,80,255,255
255,255,120,0,120,120,255,255,255,255
255,120,0,0,0,120,120,255,255,255
255,255,120,0,120,120,255,255,0,255
255,255,255,120,0,120,255,255,255,0
```

sería organizado en una matriz numpy de la siguiente manera:

```
[[255,0,120,255,255,255,255,255,255,255],
[255,120,0,120,120,255,255,80,255,255],
[255,255,120,0,120,120,255,255,255,255],
[255,120,0,0,0,120,120,255,255,255],
[255,255,120,0,120,120,255,255,0,255],
[255,255,255,120,0,120,255,255,255,0]]
```

4.2 Algoritmos

En este trabajo, proponemos un algoritmo de compresión que es una combinación de un algoritmo de compresión de imágenes con pérdidas y un algoritmo de compresión de imágenes sin pérdidas. También explicamos cómo funciona la descompresión para el algoritmo propuesto.

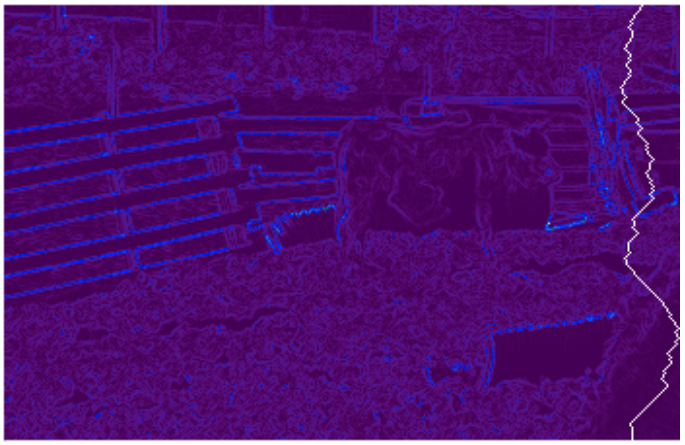
4.2.1 Algoritmo de compresión de imágenes con pérdida

En el algoritmo de compresión con pérdidas utilizado, hay una reducción en la cantidad de información que no podrá ser reconstruido exactamente igual a la original, pero que no representa un problema gracias a que dicho cambio no es significativo al momento de ser clasificado por el algoritmo que decide el estado de salud del animal.

En este caso, se implementó el método de Seam Carving o “tallado de costuras” el cual funciona de la siguiente manera:

Imagen original (Sacada del repertorio de las imágenes por clasificar)

Se asigna un valor de energía a cada píxel y se crea una ruta de píxeles con la menor energía posible, los cuales serán borrados.



La línea blanca representa la ruta de pixeles con menor energía próxima a ser borrada

Y se eliminan la cantidad de píxeles que sean necesarios.

A la izquierda se muestra la imagen comprimida a la que se le eliminaron varios pixeles, y a la derecha la imagen original.

REFERENCIAS

1. Valeria Errecat, Mariana Lucero, and Maria Alejandra Sosa. 2015.(May 2015). ANÁLISIS DEL MERCADO MUNDIAL DE CARNES. Retrieved August 2021 from http://www.unsam.edu.ar/escuelas/economia/economia_regional/CERE%20-%20Mayo%20-%202015.pdf
2. Orlando Santillan. Ganadería de Precisión. Retrieved August 2021 from https://foroconsultivo.org.mx/INCyTU/index.php/notas/102-23-ganaderia-de-precision
3. Anon. 2013.(November 2013). IMPLEMENTACIÓN DE BUENAS PRÁCTICAS PARA EL MANEJO ADAPTATIVO DEL SISTEMA PECUARIO Y LA CONSERVACIÓN DEL ECOSISTEMA PÁRAMO EN LA MICROCUENCA DE PAPALLACTA Retrieved August 2021 from https://www.ambiente.gob.ec/wp-content/uploads/downloads/2014/07/Gu%C3%ADa-Sanitaria-Ganado.pdf
4. Doulgerakis, V., Kalyvas, D., Bocaj, E., Giannousis, C., Feidakis, M., Laliotis, G. P., Patrikakis, C. and Bizelis, I. (2020). An Animal Welfare Platform for Extensive Livestock Production Systems.
5. N. Li, Z. Ren, D. Li, and L. Zeng. 2019. Review: Automated techniques for monitoring the behaviour and welfare of broilers and laying hens: Towards the goal of precision livestock farming: Animal. (September 2019). Retrieved August 15, 2021 from https://www.cambridge.org/core/journals/animal/article/review-automated-techniques-for-monitoring-the-behaviour-and-welfare-of-broilers-and-laying-hens-towards-the-goal-of-precision-livestock-farming/7D334A718C877E8E8F8DDB660EC98A4F
6. Andrew, W., Greatwood, C., & Burghardt, T. (2018). Visual Localisation and Individual Identification of Holstein Friesian Cattle via Deep Learning. In 2017 IEEE International Conference of Computer Vision Workshop (ICCVW 2017) (pp. 2850-2859). Institute of Electrical and Electronics Engineers (IEEE). https://doi.org/10.1109/ICCVW.2017.336
7. Olivier Debauche, Saïd Mahmoudi, Andriamasinoro Lalaina Andriamandroso, Pierre Manneback, Jérôme Bindelle, and Frédéric Lebeau. 2018. Cloud services integration for farm animals' behavior studies based on smartphones as activity sensors. *Journal of Ambient Intelligence and Humanized Computing* 10, 12 (2018), 4651–4662. DOI:http://dx.doi.org/10.1007/s12652-018-0845-9
8. Shai Avidan Mitsubishi Electric Research Labs et al. 2007. Seam carving for content-aware image resizing. (August 2007). Retrieved September 29, 2021 from https://dl.acm.org/doi/10.1145/1275808.1276390
9. Chen, Weng-Hsiung; Smith, C. Harrison; Fralick, Stanley C. (Septiembre de 1977). «A Fast Computational Algorithm for the Discrete Cosine Transform». *IEEE TRANSACTIONS ON COMMUNICATIONS* **25**.
10. Vences Salcedo, L. (1994,Mayo).COMPRESIÓN FRACTAL DE IMÁGENES FIJAS Y / SECUENCIAS DE IMÁGENES UTILIZANDO / ALGORITMOS GENÉTICOS. https://repositorio.tec.mx/bitstream/handle/11285/628654/CEM337194.pdf?sequence=1
11. Sandoval, M. (2008). ALGORITMO DE COMPRESIÓN DE IMÁGENES DE ALTA RESOLUCIÓN SIN PÉRDIDAS. [Tesis Pregrado, Instituto Politécnico Nacional].

https://tesis.ipn.mx/jspui/bitstream/123456789/5745/1/ALGORITMODECOMPR.pdf
12. D.A. Huffman, "A method for the construction of minimum-redundancy codes", Proceedings of the I.R.E., sept 1952, pp 1098-1102
13. Burrows M and Wheeler D (1994), *A block sorting lossless data compression algorithm*, Technical Report 124, Digital Equipment Corporation
14. Sandoval, M. (2008). ALGORITMO DE COMPRESIÓN DE IMÁGENES DE ALTA RESOLUCIÓN SIN PÉRDIDAS. [Tesis Pregrado, Instituto Politécnico Nacional]. https://tesis.ipn.mx/jspui/bitstream/123456789/5745/1/ALGORITMODECOMPR.pdf
15. Sandoval, M. (2008). ALGORITMO DE COMPRESIÓN DE IMÁGENES DE ALTA RESOLUCIÓN SIN PÉRDIDAS. [Tesis Pregrado, Instituto Politécnico Nacional]. https://tesis.ipn.mx/jspui/bitstream/123456789/5745/1/ALGORITMODECOMPR.pdf

---

[1] https://github.com/acpablox/ST0245-002