**Compresión de imágenes para la ganadería de precisión**

|  |  |  |  |
| --- | --- | --- | --- |
| Samuel Palacio Jimenez  Universidad Eafit  Colombia  spalacioj@eafit.edu.co | Juan Esteban Jaramillo Ramos  Universidad Eafit  Colombia  jejaramilr@eafit.edu.co | Simón Marín Universidad Eafit Colombia smaring1@eafit.edu.co | Mauricio Toro  Universidad Eafit  Colombia  mtorobe@eafit.edu.co |

# **RESUMEN**

El problema que buscamos encontrarle una solución es con relación a la ganadería de precisión el cual trata la ganadería con el uso de tecnologías, en nuestro caso tratemos los animales sanos ya que muchas veces no se sabe si esta enfermo por lo cual nosotros consumimos esta carnes con enfermedades por lo cual esto propaga enfermedades uno de los tantos problemas relacionados que tenemos es por ejemplo la proteína que contiene la comida con la que se alimentan a el ganado.

## **Palabras clave**

|  |
| --- |
| Algoritmos de compresión, aprendizaje de máquina,  aprendizaje profundo, ganadería de precisión, salud animal. |

# **INTRODUCCIÓN**

Tenemos que en el mundo hay una población aproximada de 7.8 billones de personas, en Colombia hay 50,34 millones de personas y tenemos en cuenta que esto va en aumento por lo cual se necesita cumplir las necesidades básicas una de ellas el hambre y se están sobreexplotando los recursos naturales como agua y animales para satisfacer estas por lo cual necesitamos que estos animales aparte de dar comida suficiente den los nutrientes y proteína buena entonces nosotros buscamos saber si estos animales están en un estado de salud sano para que se pueda comer y darse a una población y de estar enfermo, encontrar cual es la enfermedad para que no se propague por esta misma. Aparte si lo hacemos con imágenes comprimida. estos procesos de clasificar el ganado se harán mucho más eficiente y rápido que con una imagen sin comprimir al consumir mucho tiempo para esta entonces no habría ningún beneficio.

# **1.1. Problema**

el principal problema que se plantea solucionar es la calidad del ganado ya que sabemos que un ganado enfermo lo que hará es no dar los suficientes nutrientes y proteína que se necesita que de, por ejemplo, un ganado enfermo no comerá lo mismo que uno sano y eso no es todo ya que puede contagiar de enfermedades. La importancia de resolver esto es que sabemos que cada vez hay mas escases de alimentos debido a la población que va en aumento.

**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 compresión de los datos.

Se usaran dos diferentes algoritmos para poder hacer más eficiente esta clasificación el primer algoritmo el cual tiene perdidas de pixeles es la interpolación bicubica, este algoritmo tiene una gran función al permitir con una imagen aumentarle el tamaño o reducirle el tamaño en este caso se reduce el tamaño y la principal característica que diferencia este algoritmo del resto es que la imagen se ves más borrosa en cambio con el bilineal se ve más pixelada, por otro lado se puede variar el porcentaje de escalado que se quiera si se reduce mucho o no tanto la imagen para la eficiencia de la GdP. Por otro lado, tenemos el algoritmo LZ77 el cual no presenta perdidas este algoritmo busca valores que se repitan para así hacerlos juntarlos en uno solo de esta manera al comprimir una imagen no se notaran cambios y el tamaño de esta se reducirá

**1.3 Estructura del artículo**

En lo que sigue, en la Sección 2, presentamos trabajos relacional con el problema. Más adelante, en la Ssección 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 En este artículo, se muestra la visión por computadora.

Para la realización de la detección automática de ganado individual en la agricultura relacionada, primero hicieron la prueba con imágenes fijas de arriba hacia abajo con cámaras fijas y después usaron nos drones para probar el algoritmo con videos a 43 fotogramas por segundo

El algoritmo que usaron se basa en obtener imágenes desde un punto de vista aéreo que detecta el conjunto limitante de la vaca que es la tierra a su alrededor y se conecta con varias capaz para llegar a una red neuronal que es donde hace el proceso de validación de la vaca individual

2.2 En este Proyecto se demuestra como, mediante los avances tecnológicos se es capaz de genera un reconocimiento de los animales ofreciendo información de problemas de salud, estado, y bienestar a nivel de la granja

[1] Andrew, Greatwood & Burghardt T.(2018) Visual Localisation and Individual Identification of Holstein Friesian Cattle via Deep LearninG. University of Bristol

El algoritmo que usan se aplica mediante la utilización de collares puestos en los animales. Para el procesamiento de los datos primero se pasa la información por un filtro de corrección de errores o de ruidos durante la captación de esta información, después esta información llega a una red neuronal donde puede dar un resultado dependiendo de esta informacion

[1] Vasileios, Dimitrios, Enkeleda, Christos, Michalis, George. An Animal Welfare Platform for Extensive Livestock Production Systems. University of West Attica

Agricultural University of Athens

2.3 En este proyecto se descubrieron que mediante los celulares se pudo almacenar muchas variables sobre los individuos en cuestión y mediante una plataforma para archivar y procesar los datos de alta frecuencia

El algoritmo usado en este proyecto fue utilizar celulares de la marca iPhone, lo colocaron en el animal en cuestión y mediante sus sensores pudieron ir almacenando datos para recolectarlo. También tuvieron que hacer pruebas con sus baterías de los celulares y como comprimían los datos para poder usarlos

[1]Debabuche, Mahmoudi, Manneback, Bindelle, Lebeau (2018) Cloud services integration for farm animals’ behavior studies based on smartphones as activity sensors

## **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.

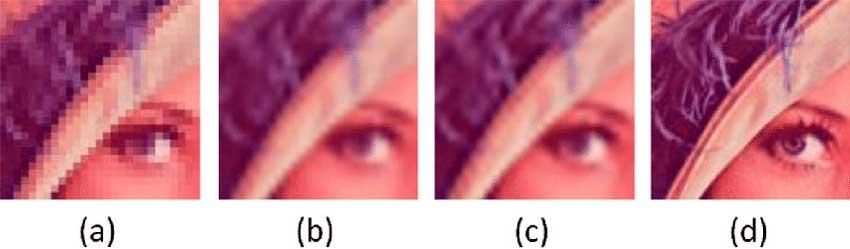
**3.2.1 talla de costuras**

En este algoritmo a cada valor de pixel de una amigan se le asigna un valor de energía y con programación dinámica y con una función se sacar el valor mínimo del pixel y así con cada pixel y los pixeles con un valor mas pequeño se eliminan de la original así cambia la imagen, pero la parte importante de la imagen se mantiene

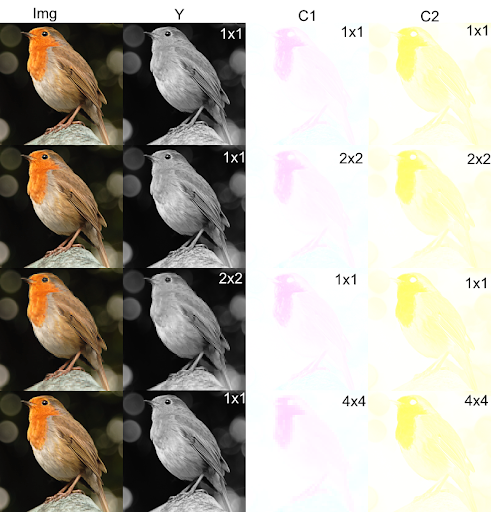


**3.2.2 Escalado de imágenes**

En este algoritmo cuando se trata de aumentar la resolución de una imagen se crean mas pixeles, pero estos no tienen color entonces copian el pixel de alrededor y algunas ocasiones se pueden



**3.2.3 algoritmo de compresión de JEPG**

Este algoritmo divide la imagen en dos canales de color y un de brillo para después quitar resolución a estos canales de resolución es decir pixelea la imagen ya que el ojo tiene más resolución de brillo. y quitan unos brillos que al final no se nota en la imagen final 

**3.2.4 Compresión fractal**

El método es el más apropiado para texturas e imágenes naturales, basándose en el hecho de que partes de una imagen, a menudo, se parecen a otras partes de la misma imagen. Los algoritmos fractales convierten estas partes en datos matemáticos llamados «códigos fractales» los cuales se usan para recrear la imagen codificada.



## **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 Compresión de Burrows-Wheeler**

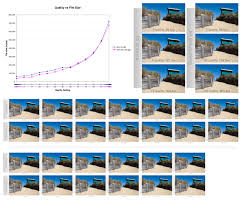
La transformación de Burrows-Wheeler ha demostrado ser fundamental para las aplicaciones de compresión de imágenes. Por ejemplo, mostró una tubería de compresión basada en la aplicación de la transformación Burrows-Wheeler seguida de codificadores de inversión, longitud de ejecución y aritmética.

**3.3.2 LZ77**

Este algoritmo el código analiza el texto en nuestro caso los archivos csv en busca de secuencias iguales en un buffer de anticipación que guarda la información y así poder comprimir algunas partes de las imágenes sin perder pixeles

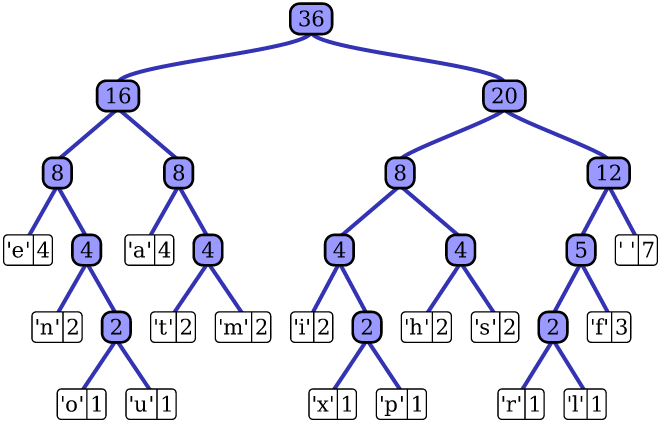
**3.3.3 LZW**

La diferencia de este algoritmo y los otros es que se puede crear mediante el uso entonces como una versión muy mejorada del LZ77 y LZ78 y si tenemos un diccionario que se le van añadiendo elementos aun así puede seguir trabajando comprimiendo estos



**3.3.4 Codificación de Huffman**

Para este algoritmo se usa una tabla de códigos de longitud de variables con la cual se pueden cambiar por una letra por ejemplo y en ese caso estamos comprimiendo una imagen sin que haya ningún tipo de perdida



## **4. DISEÑO E IMPLEMENTACIÓN DE LOS ALGORITMOS**

## En lo que sigue, explicamos 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[[1]](#footnote-1).

## **4.1 Estructuras de datos**

Usamos el árbol como estructura de datos ya que era visualmente la mas entendible para la explicación de este y también es una estructura de datos mas organizada

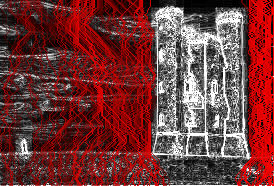


**4.2 Algoritmos**

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 Tallado de costuras**

Este algoritmo lo que hace es calcular la energía, el peso y la densidad de cada pixel, y a partir de esta energía se realiza una lista de costuras, estas costuras se clasifican por energía, siendo las costuras con baja energía las menos importantes para el contenido de la imagen, Después de esto se retiran las costuras de mas baja energía según sea necesario y así llega a un resultado de compresión de la imagen. Las costuras que se van a eliminar depende solo del ancho que se quiera encoger



**4.2.2 Algoritmo de compresión de Huffman**

El código de huffman de cada carácter se puede obtener mediante la creación de un árbol binario, de una lista de nodos en l que se etiquetan nodos con los caracteres junto a sus frecuencias, y asi de forma consecutiva se va extiendo. Cabe decir que para descomprimir los datos hay que conocer las correspondecias ya que es relativo para archivos de poco tamaño la compresión no varia mucho, pero al ser archivos de gran tamaño el resultado cambia bastante.

**4.3 Análisis de la complejidad de los algoritmos**

Primero se Tiene que recorrer el array entonces de ahí sale el O(N) y el log n se usa cuando se crea el espacio de memoria para el arbol

|  |  |
| --- | --- |
| **Algoritmo** | **La complejidad del tiempo** |
| Compresión | O(N log n) |
| Descompresión | O(N log n) |

***Tabla* 2:** Complejidad temporal de los algoritmos de compresión y descompresión de imágenes.

|  |  |
| --- | --- |
| **Algoritmo** | **Complejidad de la memoria** |
| Compresión | O(N log n ) |
| Descompresión | O(N log n) |

***Tabla* 3:** Complejidad de memoria de los algoritmos de compresión y descompresión de imágenes.

**4.4 Criterios de diseño del algoritmo**

Este algoritmo fue diseñado de tal que se crea un árbol binario para poder reducir el tamaño del archivo o ya sea un frase, al crear el árbol binario los caracteres repetidos se eliminarían y así tendría una reducción grande en el tamaño, en el tiempo a la hora de ejecutarse este algoritmo empieza a recorrer el árbol binario que creo y así empieza a volver a armar el archivo original, El tiempo de este algoritmo es optimo por que usa un complejidad adecuada.

**5. RESULTADOS**

**5.2 Tiempos de ejecución**

En lo que sigue explicamos la relación entre el tiempo promedio de ejecución y el tamaño promedio de las imágenes del conjunto de datos completo, en la Tabla 6.

|  |  |  |
| --- | --- | --- |
|  | ***Tiempo promedio de ejecución (s)*** | ***Tamaño promedio del archivo (MB)*** |
| *Compresión* | 6.3 s | 500kb |
| *Descompresión* | 7.1 | 500kb |

## **Tabla 6:** Tiempo deejecución de los algoritmos

## **5.3 Consumo de memoria**

Presentamos el consumo de memoria de los algoritmos de compresión y descompresión en la Tabla 7.

|  |  |  |
| --- | --- | --- |
|  | ***Consumo promedio de memoria (MB)*** | ***Tamaño promedio del archivo (MB)*** |
| Compresión | 67 MB | 500kb |
| Descompresión | 64 MB | 500kb |

## **Tabla 7:** Consumopromedio de memoria de todas las imágenes del conjunto de datos, tanto para la compresión como para la descompresión.

## **5.3 Tasa de compresión**

Presentamos los resultados de la tasa de compresión del algoritmo en la Tabla 8.

|  |  |  |
| --- | --- | --- |
|  | ***Ganado sano*** | ***Ganado enfermo*** |
| Tasa de compresión promedio | 11:6 | 5.2 |

## **Tabla 8:** Promedio redondeado de la tasa de compresión de todas las imágenes de ganado sano y ganado enfermo.

## **6. DISCUSIÓN DE LOS RESULTADOS**

Los resultados de este proyecto para nosotros fueron positivos ya que pudimos entender mas allá sobre los tipos de compresión de imagen para nosotros el consumo de tiempo obre estos algoritmos de compresión es adecuado ya que para ser imágenes tan grandes no necesita de tanto tiempo, el consumo de la memoria si nos pareció un poco alto que lo que esperábamos pero entendemos que es por la el tamaño de ciertos archivos la exactitud y la precisión son apropiadas para este problema ya que para determinar si la vaca esta bien o mal necesitamos de unos datos claros. Y la tasa de compresión nos parece adecuada ya que el objetivo principal es como reducir los pixeles y es algo que lo pudimos llegar

**6.1 Trabajos futuros**

Para un futuro nos gustaría trabajar en buscar una forma de mejorar el algoritmo de huffman de manera que encontremos una forma que los archivos no queden guardados como un binario y tener esa forma de visualizarlo para que sea mucho mas entendible este algoritmo y sea más fácil de implementar

# **REFERENCIAS**

Programmerclick: algoritmo de talla de costura Obtenido de <https://programmerclick.com/article/4510360417/>

Carlos López , 2021 ¿Qué es la compresión LZW y cómo funciona? Obtenido de https://es.ccm.net/contents/731-compresion-lzw

Ecured Lz77 obetenido de <https://www.ecured.cu/Lz77>

Kripkit Burrows-Wheeler transformación Obtenido de

<https://kripkit.com/burrows-wheeler-transformacin/>

Wikipedia 2021 Compresión fractal Obtenido de <https://es.wikipedia.org/wiki/Compresi%C3%B3n_fractal>

Netinbag.com ¿Qué es el escalado de imagen? Obtenido de https://www.netinbag.com/es/internet/what-is-image-scaling.html

PensamientosComputables 2013, Como el algoritmo de compresión JPEG engaña al ojo humano obtenido de <http://www.pensamientoscomputables.com/entrada/algoritmo-compresion-jpeg-jpg.html>

Towards data science, Image Processing: Image Scaling Algorithms Obtenido de <https://towardsdatascience.com/image-processing-image-scaling-algorithms-ae29aaa6b36c>

Spantip, 2012, LZ77 Y LZ88 obtenido de <https://www.spantip.com/wiki/LZ77>

neo.lcc.uma.es, Huffman Encoding obtenido de

https://neo.lcc.uma.es/evirtual/cdd/tutorial/presentacion/huffman.html

1. http://www.github.com/ ????????? /proyecto/ [↑](#footnote-ref-1)