# **COMPRESIÓN DE IMÁGENES PARA SISTEMAS DE CLASIFICACIÓN DE LA SALUD ANIMAL**

|  |  |  |  |
| --- | --- | --- | --- |
| Jian Carlo Gallego Mejía Universidad ...  País  Correo electrónico en Eafit | Carlos Alejandro Benitez H.  Universidad EAFIT  Colombia  cabenitezh@eafit.edu.co | Simón Marín Universidad EAFIT Colombia smaring1@eafit.edu.co | Mauricio Toro  Universidad EAFIT  Colombia  mtorobe@eafit.edu.co |

# **RESUMEN**

En el presente informe partimos de la problemática encontrada en la clasificación de animales según su estado de salud en el contexto de Ganadería de Precisión, ya que esta se hace a través de imágenes y se necesita que el sistema sea rápido, eficaz y capaz de trabajar en la zona rural, en la que conexión de red es muy limitada. A raíz de esto se es necesaria la compresión de las imágenes, pero sin perder exactitud, ya que la falta de eficacia en esta clasificación afecta directamente a toda la producción de carne, y esta a su vez a la salud de toda la población.

## **Palabras clave**

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

# **INTRODUCCIÓN**

Actualmente es casi indispensable el uso y avance de la tecnología a favor del ser humano y de todas las áreas que a este le compete, y la ganadería no es la excepción.

Es justo en esta área en la que encontramos un gran problema, la necesidad de que la clasificación de los animales según su salud en el contexto de Ganadería de Precisión, que se puede lograr por medio de imágenes, pueda ser usable en una infraestructura de red tan limitada como la que usa esta Ganadería de Precisión.

Es por ello que se necesita que tal sistema sea también capaz de comprimir las imágenes que se van a analizar, pero sin perder calidad, y así lograr que pueda ser usado en el área en el que se requiere, y no se pierda exactitud en su análisis.

# **Problema**

Para el día de doy es sumamente necesario un sistema capaz de reconocer y clasificar el estado de salud de los animales en el contexto de Ganadería de Precisión; pero tan necesario como este, es también que tal sistema sea eficaz y usable en el área para la que se propone. Es por eso por lo que se hace necesario que tal sistema pueda comprimir las imágenes para poderse usar en áreas de baja infraestructura de red, pero que a su vez las imágenes no pierdan calidad para lograr la exactitud requerida para su análisis.

* 1. **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.

Para la compresión de datos decidimos usar 2 algoritmos, uno con perdida y el otro sin perdida. El algoritmo de comprensión con perdida que usamos es Interpolación Bidimensional Lineal, ya que es el que, a nuestro parecer, evita eliminar información muy importante y única al comprimir a base del promedio de una función. **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.

1. **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.

## **An Animal Welfare Platform for Extensive Livestock Production Systems**

En las últimas décadas se ha visto la necesidad de que los alimentos se produzcan y procesen con mayor respeto por el bienestar de los animales, por la relación que ido estrechando entre estos y la salud del consumidor.

La arquitectura propuesta a través de este experimento utiliza sensores colocados en los animales, cuyos datos se preparan y procesan localmente en un dispositivo Edge que actúa como nodo central. Luego, los resultados se envían a la nube, desde donde se puede acceder a ellos a través de una aplicación móvil complementaria.

Tiene un sistema en el cual primero se hace un preprocesamiento de datos, en el que se utilizan técnicas de aprendizaje automático, como el filtrado (eliminación de ruido o corrección de mediciones falsas), procesos de integración o agregación de datos (como el que es basado en la marca de tiempo para muestreo redundante) y normalización (como escalando los valores de todos los atributos para tener el mismo peso en el tratamiento de datos).

## **Cloud services integration for farm animals’ behavior studies based on smartphones as activity sensors**

En esta Investigación el problema es la necesidad de usar la tecnología para la recolección de datos acerca del comportamiento animal, para la utilidad de los investigadores. Esto a través de Smartphones, principalmente de iPhone, y todos los sensores que en este se encuentran.

Para esta solución ellos aplicaron una compresión de datos basada en la reducción de la precisión. En datos del sensor, todos los datos sin procesas se registran con una precisión de 6 dígitos. Esta compresión se realiza en 2 wats: primero elimina redundancias sustituyendo los datos redundantes por un intervalo de tiempo en el que el valor permanece constante, y así preservar la integridad de los datos; y el segundo es truncando los datos a 3, 4 y 5 dígitos decimales.

## **Visual Localisation and Individual Identification of Holstein Friesian Cattle via Deep Learning**

En este artículo se busca demostrar que las tuberías de visión por computadora, que utilizan arquitecturas neuronales profundas, son adecuadas para realizar la detección de ganado de Holstein Friesian, así como la identificación individual en configuraciones relevantes para la agricultura.



Demuestran que las redes disponibles en el mercado pueden realizar la identificación de individuos en imágenes adquiridas por cámaras fijas. Luego presenta una tubería de procesamiento de video que procesa de manera eficiente la filmación dinámica de rebaño que toman Vehículos Aéreos no Tripulados (UAV).

## **A systematic literature review on the use of machine learning in precision livestock farming**

En este artículo se hace una revisión bibliográfica de trabajos recientes sobre el uso del aprendizaje automático (Machine Learning) en la Ganadería de Precisión, en el que se destacan las oportunidades de este para la ganadería, se muestran los sensores, software y técnicas actuales para el análisis de datos; y se detalla la creciente apertura de las fuentes de datos.

## Más que resolver un problema, en este trabajo se pretende analizar y dar cuenta de cómo el Machine Learning ha actuado sobre el sector de la ganadería, por medio de la Ganadería de Precisión; resolviendo problemáticas como la salud animal, la adecuada alimentación, la gestión de los recursos, etc.

## **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.

## **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.

## **Alternativas de compresión de imágenes con pérdida**

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

* + 1. **Tallado de costuras**

Es un algoritmo de compresión por medio de recorte de imagen, el cual está basado en la energía para adaptar tal recorte.

El algoritmo trabaja de la siguiente manera:

* Primero le asigna un valor de energía a cada píxel.
* Luego encuentra ocho rutas que están conectadas al píxel con el menor valor de energía.
* Y por último elimina todos los pixeles en la ruta

El algoritmo repite estos 3 pasos hasta que el número de filas o el número de columnas alcance el estado ideal.

De esta manera, el algoritmo logra recortar pixeles de la imagen sin recortar necesariamente objetos principales de esta, sino borrando lo que parece no tener relevancia.

Una montaña con vista al mar

Descripción generada automáticamente con confianza media

Una montaña con vista al mar

Descripción generada automáticamente con confianza media

*Información e imágenes sacadas de: https://programmerclick.com/article/4510360417/*

* + 1. **Escalado de imágenes**

Este método consiste en cambiar el tamaño de una imagen digital, de manera de que los pixeles disminuyen y se pierde visiblemente la calidad de la imagen.

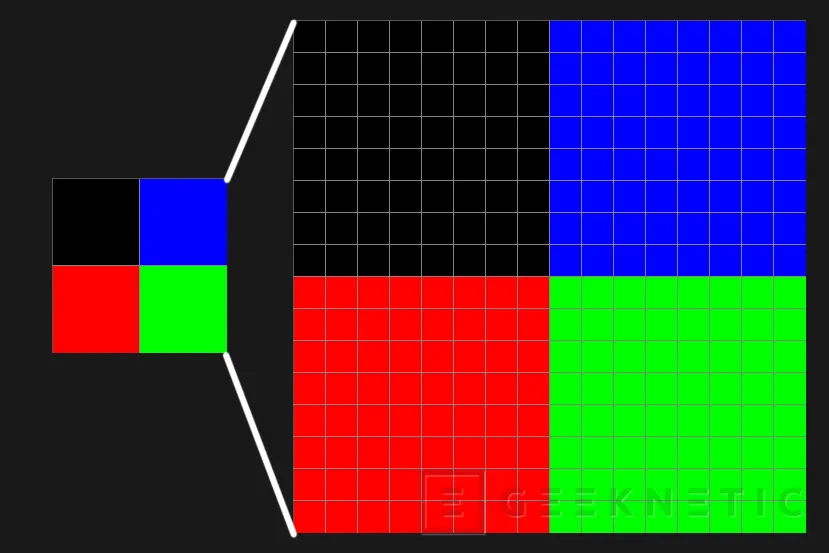
****

Imagen sacada de: <https://acf.geeknetic.es/imgw/imagenes/tutoriales/2019/1662-integer-scaling-que-es-como-funciona/1662-integer-scaling-que-es-como-funciona-2.jpg?f=webp>

* + 1. **Compresión Fractal:**

Es un método de compresión que se basa en fractales. Es principalmente apropiado para texturas e imágenes naturales, ya que se basa en el hecho de que partes de una imagen se parecen a otras partes de la misma imagen, en muchas ocasiones. Estos algoritmos convierten estas partes en datos matemáticos llamados “códigos fractales”, que se usan para recrear la imagen codificada.

Cabe resaltar que para este método la codificación es cara, a nivel computacional, gracias a la búsqueda de similitudes propias. Aunque la decodificación es bastante rápida.

[*https://es.wikipedia.org/wiki/Compresi%C3%B3n\_fractal*](https://es.wikipedia.org/wiki/Compresi%C3%B3n_fractal)



*https://www3.gobiernodecanarias.org/medusa/ecoblog/mrodperv/files/2015/12/lena.png*

* + 1. **Interpolación Bidimensional**

La Interpolación Bidimensional es un método de compresión que consiste en buscar un valor intermedio en una función de 2 variables. En este método se usa la Interpolación Unidimensional, la cual es lineal, dos veces: primero en una dirección y luego en otra. Esto quiere decir que la Interpolación Bidimensional es cuadrática, pues consta de 2 funciones lineales.

Gráfico, Gráfico de superficie

Descripción generada automáticamente

*Información e imágenes sacadas de: https://slideplayer.es/slide/13084804/*

## **Alternativas de compresión de imágenes sin pérdida**

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

* + 1. **Run-length encoding**

Es un método de comprensión de datos en el que se almacenan como un único valor las secuencias de datos que tienen el mismo valor, y el recuento de estas. Es un método diseñado especialmente para datos que contengan muchas secuencias de un mismo valor, tales son gráficos sencillos como iconos y logos, que están principalmente construidas por áreas de colores planos.

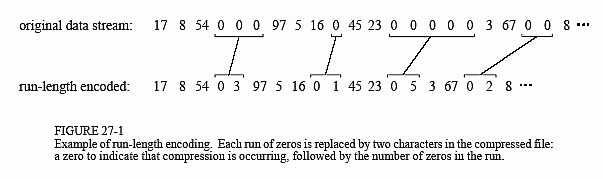
Usando esta compresión pasamos de:

BBBBBBBBBBBBNBBBBBBBBBBBBNNNBBBBBBBBBBBBBBBBBBBBBBBBNBBBBBBBBBBBBBB

A:

12B1N12B3N24B1N14B

[*https://es.wikipedia.org/wiki/Run-length\_encoding*](https://es.wikipedia.org/wiki/Run-length_encoding)



*Imagen sacada de: https://www.dspguide.com/graphics/F\_27\_1.gif*

* + 1. **Deflación**

Es un algoritmo de Compresión que combina la codificación LZSS y la de Huffman, en el que se comprimen los datos por bloques, y a cada uno de estos bloques se les precede una cabecera de 3 bits, que se define así:

**Primer bit:** Marca si el bloque es el último del archivo

**1:** Es el último bloque del archivo

**0:** Hay más bloques por procesar después.

**Segundo y Tercer bit:** Determinan la codificación del bloque

**00:** Sección almacenada en bruto y literal.

**01:** Bloque de Huffman estático comprimido, usando un árbol de Huffman ya definido.

**10:** El Bloque comprimido completado con la tabla de Huffman dada.

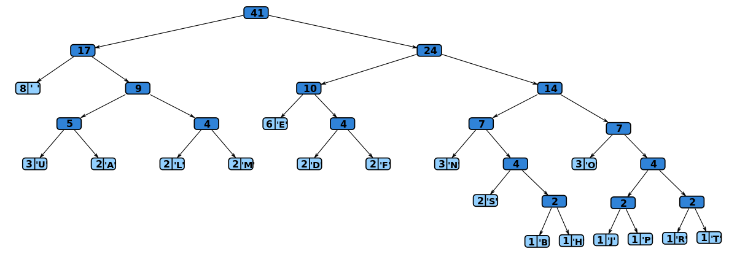
**11:** Reservado, no está en uso.

En esta compresión se realizan 2 pasos:

* Se buscan y reemplazan por punteros las cadenas de bits duplicados.
* Se reemplazan símbolo con otros nuevos basados en la frecuencia de uso.

Se codifican en su mayor parte a través del método de codificación dinámica de Huffman, para los datos que sean compresibles; en esta codificación se produce un árbol de Huffman optimizado, el cual es personalizado para cada bloque de datos individualmente. Cabe resaltar que las instrucciones para generar este árbol aparecen inmediatamente después del bloque de la cabecera.

*Sacado de: https://es.wikipedia.org/wiki/Deflaci%C3%B3n\_(algoritmo)*



*Imágenes sacadas de:* *https://es.wikipedia.org/wiki/Portable\_Network\_Graphics*

* + 1. **DPCM**

*Differential Pulse Code Modulation* es un método en el que se codifica la diferencia entre un valor y el anterior. Estas codificaciones representan el nivel de gris medio del bloque, y es muy común que el nivel medio de un bloque sea similar al del bloque anterior, por lo que la diferencia entre 2 bloques consecutivos es mínima.

Un gran ejemplo es al codificar una imagen donde se encuentra gran parte de cielo, en este caso se enviarán números más pequeños ya que la diferencia entre los bloques será casi 0.

*https://es.wikipedia.org/wiki/Codificaci%C3%B3n\_entr%C3%B3pica*

Diagrama

Descripción generada automáticamente con confianza baja

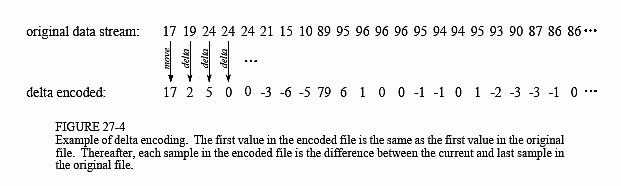
*Imagen sacada de: https://es.wikipedia.org/wiki/Codificaci%C3%B3n\_entr%C3%B3pica*

* + 1. **Delta encoding**

Es un método basado en la compresión compacta de un archivo. En el caso de los archivos, la función Delta se refiere a la diferencia que hay entre dos versiones distintas dentro de un mismo archivo.

De esta manera la codificación solo se enfoca en los bytes que han sido modificados desde la última versión del archivo. El archivo original se puede obtener con la versión de referencia del archivo y el generado por el algoritmo Delta encoding, teniendo el archivo comprimido.

*https://es.wikipedia.org/wiki/Compresi%C3%B3n\_de\_imagen#La\_compresi%C3%B3n\_con\_y\_sin\_p%C3%A9rdida\_(LOSSLESS)*



*Imagen sacada de:* [*https://www.dspguide.com/graphics/F\_27\_4.gif*](https://www.dspguide.com/graphics/F_27_4.gif)

## **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-2).

## **Estructuras de datos**

## La estructura de datos que usamos fue una matriz de numpy. Una matriz es un conjunto de elementos del mismo tipo que se guardan de manera consecutive en memoria.

## **Figura 1:** Estructura de una matriz.

* 1. **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.

Diagrama

Descripción generada automáticamente

**Figura 5:** https://es.wikipedia.org/wiki/Interpolaci%C3%B3n\_bilineal

* + 1. **Algoritmo de compresión de imágenes con pérdida**

El algoritmo de compression en esta etapa fue el de Interpolación Bidimensional Lineal, el cual consiste en interpolar funciones de 2 variables en una malla de 2 dimensiones, hacienda una interpolación lineal en una dirección y luego en la otra.

**Figura 2:** Compresiónde la imagen mediante interpolación bilineal.

# **REFERENCIAS**

1. Doulgerakis, V., Kalyvas, D., Bocaj, E., Giannousis, C., Feidakis, M., Laliotis, G., Patrikakis, C. and Bizelis, I. *An Animal Welfare Platform for Extensive LivestockProduction Systems.* Recuperado de: <http://ceur-ws.org/Vol-2492/paper1.pdf>
2. Debauche, O., Mahmoudi, S., Herinaina, A., Manneback, P., Bindelle, J and Lebeau, F. Cloud services integration for farm animals’ behavior studies based on smartphones as activity sensors. *Journal Ambient Intell Human Comput, 10,* 4651–4662 (2019). <https://doi.org/10.1007/s12652-018-0845-9>
3. Andrew, W., Greatwood, C. and Burghardt, T. Visual Localisation and Individual Identification of Holstein Friesian Cattle via Deep Learning. *IEEE, International Conference on Computer Vision (ICCV),* 2017, pp. 2850-2859. doi: 10.1109/ICCVW.2017.336.
4. Correa, L. Ajuste de curvas e interpolación: https://slideplayer.es/slide/13084804/
5. García, R., Aguilar, J., Toro, M., Pinto, A. and Rodríguez, P. A systematic literature review on the use of machine learning in precision livestock farming. *Computers and Electronics in Agriculture, 179,* 2020*.* doi: https://doi.org/10.1016/j.compag.2020.105826.
6. Programmer click. *Algoritmo de talla de costura.* [Recuperado](https://programmerclick.com) de: https://programmerclick.com/article/4510360417/
7. Bercial, J. Geeknetic. *Integer Scaling: Qué es y cómo funciona.* 2019. Recuperado de: https://www.geeknetic.es/Guia/1662/Integer-Scaling-Que-es-y-como-funciona.html
8. Wikipedia. *Compresión fractal.* Recuperado de: <https://es.wikipedia.org/wiki/Compresi%C3%B3n_fractal>
9. *Fractales Generados por IFS.* Recuperado de: <https://www3.gobiernodecanarias.org/medusa/ecoblog/mrodperv/fractales/fractales-generados-por-ifs/>
10. Mathworks. *Transformación de Coseno Discreta.* Recuperado de: <https://la.mathworks.com/help/images/discrete-cosine-transform.html>
11. Wikipedia. *Run-length encoding.* Recuperado de: <https://es.wikipedia.org/wiki/Run-length_encoding>
12. Smith S. The Scientist and Engineer’s Guide to Digital Signal Processing. *Chapter 27: Data Compression.* Recuperado de: http://www.dspguide.com/ch27.htm
13. Wikipedia. *Deflación (algoritmo).* Recuperado de: <https://es.wikipedia.org/wiki/Deflaci%C3%B3n_(algoritmo)>
14. Wikipedia. *Portable Network Graphics.* Recuperado de: <https://es.wikipedia.org/wiki/Portable_Network_Graphics>
15. Wikipedia. *Codificación entrópica.* Recuperado de: <https://es.wikipedia.org/wiki/Codificaci%C3%B3n_entr%C3%B3pica>
16. Wikipedia. *Compresión de imagen.* Recuperado de: <https://es.wikipedia.org/wiki/Compresi%C3%B3n_de_imagen>
17. Wikipedia. *Interpolación Bilineal.* Recuperado de: <https://es.wikipedia.org/wiki/Interpolaci%C3%B3n_bilineal>

1. http://www.github.com/ ????????? /proyecto/ [↑](#footnote-ref-2)