Informe sobre ganadería de precisión

Juan Pablo Restrepo
Universidad Eafit
Colombia
jprestrep@eafit.edu.co

Tomás Sepúlveda
Universidad Eafit
Colombia
tsepulvedf@eafit.edu.co

Simón Marín
Universidad Eafit
Colombia
smaring1@eafit.edu.co

Mauricio Toro
Universidad Eafit
Colombia
mtorobe@eafit.edu.co

RESUMEN
El problema se encuentra en la ineficiencia y alto consumo de energía del monitoreo de la alimentación, producción, y bienestar del ganado en la ganadería de precisión.

Este problema es importante ya que, al tener un modelo o sistema ineficiente, este afecta al bienestar del ganado, lo cual influye directamente en la producción de productos derivados de la ganadería. Por otra parte, también es importante tener en cuenta la optimización de recursos al implementar el sistema, pues los recursos en el campo suelen ser limitados. Se implemento un algoritmo de escalado de imágenes mediante interpolación, el cual demostró un alto nivel de eficiencia en cuanto a los siguientes promedios de tiempos y memoria, siendo estas en la compresión: tiempo de ejecución de 0.81 s para un tamaño de archivo de 0.87 MB y un consumo de memoria 147.45 MB para un tamaño del archivo de 0.87 MB, respectivamente. De igual forma, en los siguientes promedios de tiempos y memoria para la descompresión: tiempo de ejecución de 0.18 s para un tamaño de archivo de 1.28 MB y un consumo de memoria de 112.55 MB para un tamaño del archivo de 1.28 MB, respectivamente. Por otro lado, la tasa de compresión de sanos es 2,4: 1, es decir, cada 2,4 unidades del original se comprimen en 1, y, mientras que la tasa de compresión de enfermos es 2,2: 1, esto es, cada 2,2 unidades del original se vuelven 1 en el comprimido.

Palabras clave
Algoritmos de compresión, aprendizaje de máquina, aprendizaje profundo, ganadería de precisión, salud animal.

1. INTRODUCCIÓN

Explica la motivación, en el mundo real, que lleva al problema. Incluyan la historia de este problema. *(En este semestre, la motivación es la razón por la que necesitamos comprimir imágenes para clasificar la salud animal en el contexto de la ganadería de precisión).* La compresión de imágenes es algo muy importante en el campo de la ganadería de precisión, ya que los recursos en las zonas rurales pueden ser limitados, lo que nos lleva a tener un consumo de energía y memoria bastante alto si no se tiene ninguna manera de procesar estos datos eficientemente.

1.1. Problema

En pocas palabras, expliquen el problema, el impacto que tiene en la sociedad y por qué es importante resolver el problema. *(En este semestre, el problema es comprimir las imágenes para clasificar la salud animal en el contexto de la ganadería de precisión).*

El problema radica en la baja eficacia de los métodos tradicionales para el monitoreo de ganado. Esto puede llegar a perjudicar directamente la producción del ganado, lo que terminaría afectando tanto al ganadero como a los consumidores, pues el ganadero vería pérdidas en sus ganancias y a los consumidores les llegaría un producto de menor calidad.

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.

1.2.1 Solución

Para esta segunda entrega en la que debemos utilizar un algoritmo de compresión de imágenes con pérdidas hemos decidido comenzar con el escalado de imágenes mediante interpolación, pues pensamos que es un algoritmo óptimo para comenzar nuestro primer acercamiento a los diferentes métodos de compresión de imágenes, esto porque es uno de los algoritmos más utilizados y documentados que hay, además de la sencillez en la implementación del mismo.

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.

Explique cuatro (4) artículos relacionados con el problema descrito en la sección 1.1. Puede encontrar los problemas relacionados en las revistas científicas, en lo posible, en inglés. Considere Google Scholar para su búsqueda. *(En este semestre, el trabajo relacionado es la investigación sobre la clasificación de la salud animal y la compresión de datos, en el contexto de la GdP).*

1. Uno de los problemas relacionados lo podemos observar en el artículo “Cloud services integration for farm animals’ behavior studies based on smartphones as activity sensors” en el cual se habla sobre como los iPhones pueden ser instrumentos utilizados para la monitorización del comportamiento del ganado, y como la transmisión de los datos recopilados puede ser una tarea muy complicada por la gran cantidad de información como imágenes y variables que tienen que ser transportadas sin perder eficiencia.

2. Usualmente, el foco de los estudios dirigidos al aumento de la eficiencia en la producción de alimento, en las granjas de animales, está divido en dos entornos. Primero, el seguimiento de un animal se realiza en varios ámbitos, siendo dos de los más importantes el comportamiento y el monitoreo del mismo (temperatura, producción, alimentación), no obstante no se suelen tomar ciertos factores en cuenta como lo son el entorno (temperatura del lugar, viendo, lluvia, calidad del suelo), de la misma forma, los del entorno no indagan ampliamente en el monitoreo, junto a los dos anteriores, lo cual permitiría una mayor comprensión y facilitaría la manera de diagnosticar los problemas que se puedan presentar.

– Artículo "A systematic literature review on the use of machine learning in precision livestock farming"

3. En este estudio se buscaba poder analizar y reconocer el ganado, a cada vaca que se encontrara en el campo, y sus características propias, para así identificarla de las demás, la raza bovina Holstein, que destaca por su alta producción de leche, carne y su buena adaptabilidad, fue la utilizada en este estudio. Un objetivo del sistema, es que se podrían eliminar algunos métodos utilizados para marcar vacas, como son los tatuajes en las orejas, que hieren físicamente al animal.

el estudio fue realizado utilizando drones, videos de grupos de bovinos eran tomados en su entorno de campo natural, y utilizando Deep learning se crearon algoritmos para reconocer a las vacas y sus características individuales, utilizando los patrones de su piel, tamaño, color y ecuaciones para determinar que tanto se movería la vaca de donde primero fue vista, se lograba diferenciar de una vaca a otra, el sistema tomaba segmentos del video y analizaba a cada vaca que podía encontrar y le asignaba un número y guardaba su información. Este proceso fue repetido múltiples veces en donde el algoritmo mejoraba cada vez más en identificar a una vaca de otra, mostrando resultados increíbles y con cada vez menos fallos.

Fue concluido que es posible utilizar sistemas para reconocer las caracterices bovinas, identificar una vaca de otra y en una manera que no resulte obstructiva para el animal, a diferencia de otros métodos utilizados que hieren a la vaca.

-Articulo: "Visual localisation and Individual identification of Holstein Friesian Cattle via Deep Learning. "

4. El artículo “An Animal Welfare Platform for Extensive LivestockProduction Systems” habla sobre cómo recientemente los consumidores exigen mejorar cada vez más el bienestar del ganado durante la producción de alimentos, lo que nos lleva a buscar maneras de mejorar el monitoreo del ganado para que de esta manera los granjeros o usuarios finales puedan tener mayor información o un mayor control sobre sus animales, para así darles una mejor calidad de vida. La solución que dan a este problema se fundamente en el uso de un collar que deben llevar los animales en todo momento, este collar estará enviando información diariamente sobre diferentes datos de los animales como, sus signos vitales, la comida consumida en las últimas 24 horas, sus movimientos, entre otros, estos datos se envían a través de Bluetooth a un “Edge device” o punto de entrada para que esta información pase así a las bases de datos. Además de esto, este dispositivo enviará los últimos datos obtenidos a dicha base de datos (nube) a la cual el usuario podrá acceder mediante una aplicación móvil para así poder ver el estado del ganado y los movimientos que realizan en tiempo real. De esta manera se logra que el usuario tenga mayor control sobre los animales para así poder intervenir en caso de que la salud del animal se encuentre en peligro.

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.

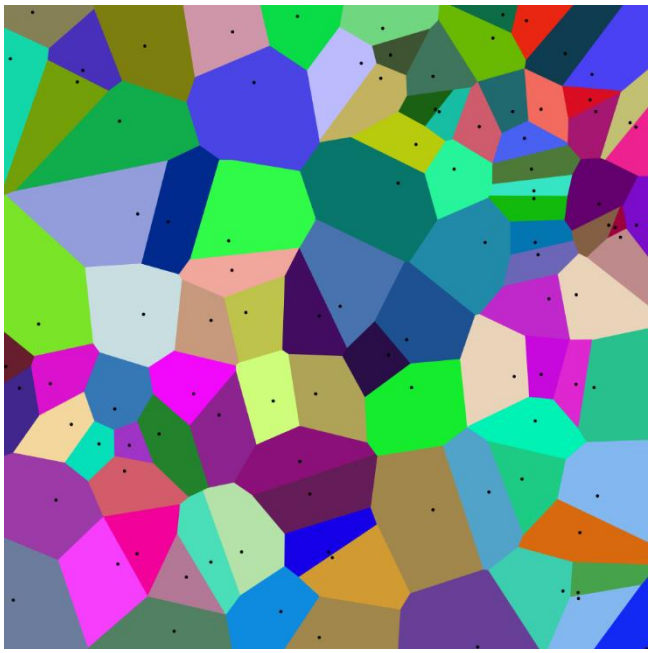
3.1 Recopilación y procesamiento de datos

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.

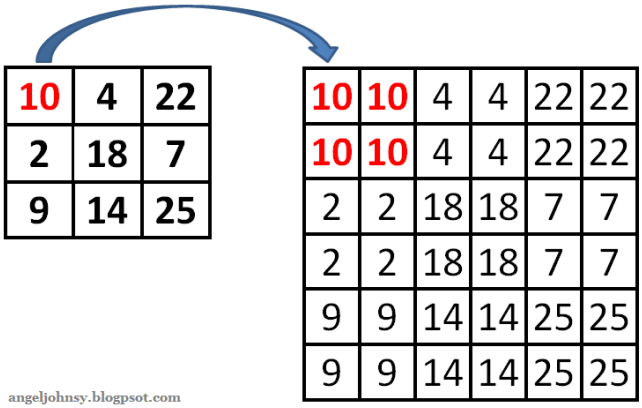
El conjunto de datos se dividió en un 70% para entrenamiento y un 30% para pruebas. Los conjuntos de datos están disponibles en https://github.com/mauriciotoro/ST0245-Eafit/tree/master/proyecto/datasets .

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. *(En este semestre, ejemplos de tales algoritmos son el tallado de costuras, el escalado de imágenes, la transformación de coseno discreto, la compresión con ondeletas y la compresión fractal).*



3.2.1 Escalado de imágenes:

El escalado de imágenes se refiere al proceso realizado para cambiar el tamaño de imágenes digitales. Existen dos tipos de imágenes digitales, las vectoriales y las imágenes por tramos (píxeles).

Uno de los métodos de escalado de imágenes es el de interpolación proximal o interpolación por el vecino más cercano, este algoritmo aproxima el valor de una función a un punto desconocido o no dado, de esta manera el algoritmo encontrará el punto siguiente más cercano, produciendo así un interpolante o sección de la imagen.

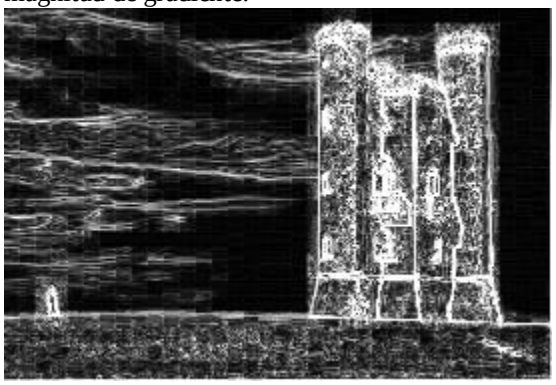
Ejemplo:

3.2.2 Compresión con costuras (tallado de costuras):

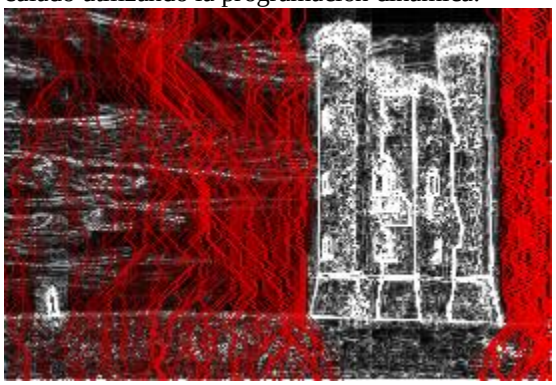
Este algoritmo es utilizado para el cambio de tamaño de una imagen, este funciona al establecer "costuras" (partes de menos importancia) en la imagen, y procede a remover o añadir costuras para incrementar o reducir el tamaño de una imagen. Esto ocurre en los siguientes pasos:

1) Comienza con una imagen

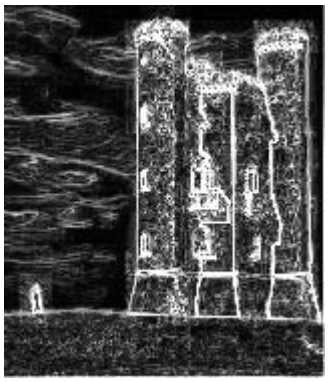
2) Determina la densidad, peso y energía de cada pixel, utilizando varios algoritmos como son la entropía, la prominencia visual, la magnitud de gradiente, entre otras. En este ejemplo se utiliza la magnitud de gradiente.



3) Las costuras son enlistadas en base a su nivel de energía, si es bajo, pasan a ser de las partes de menor importancia de la imagen, esto puede ser calculado utilizando la programación dinámica.



4) Se remueven las costuras que tienen niveles bajos de energía necesarias.



5) Así resulta la imagen final.

3.2.3 Transformación de coseno directo:

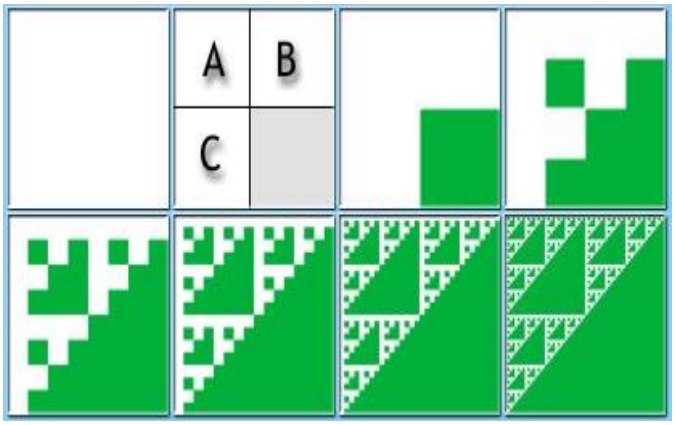
La transformación de coseno discreta, permite compilar y comprimir cantidades de datos reales en un número limitado de coeficientes, lo cual facilita su portabilidad.

La transformada de coseno discreta F(k) de una función discreta f(i): $R^N \rightarrow R^N$, (donde R denota el conjunto de los números reales) en la cual i = 0, 1, 2, … N – 1 se define como: $F(k) = \frac{\sqrt{2}c(k)}{\sqrt{N}} \sum_{i=0}^{N-1} f(i) cos\left[\frac{(2i+1)k\pi}{2N}\right]$ donde $c(k) = \frac{1}{\sqrt{2}}$ para k = 0 y c(k) = 1 para otros números enteros hasta N - 1

3.2.4 Compresión Fractal:

La compresión fractal es un método de compresión con perdida, utilizado en imágenes digitales, el cual está basado en fractales, que son objetos geométricos cuya estructura básica se repite a diferentes escalas.

El método parte de una imagen que suele parecerse a partes de la misma imagen, y estas son convertidas en códigos fractales los cuales son utilizados para recrear la imagen codificada.



La representación de una imagen fractal, es descrita matemáticamente como un sistema de funciones iteradas. Que son una construcción matemática usada para representar conjuntos fractales que presenten auto similitud.

Algunas de las características de la compresión fractal son: La codificación es extremadamente cara a nivel computacional, aunque la decodificación es bastante rápida. Otra característica es que las imágenes se vuelven independientes de la resolución después de ser convertidos al código fractal, debido a que el sistema de función iterada en el archivo comprimido se escala indefinidamente.

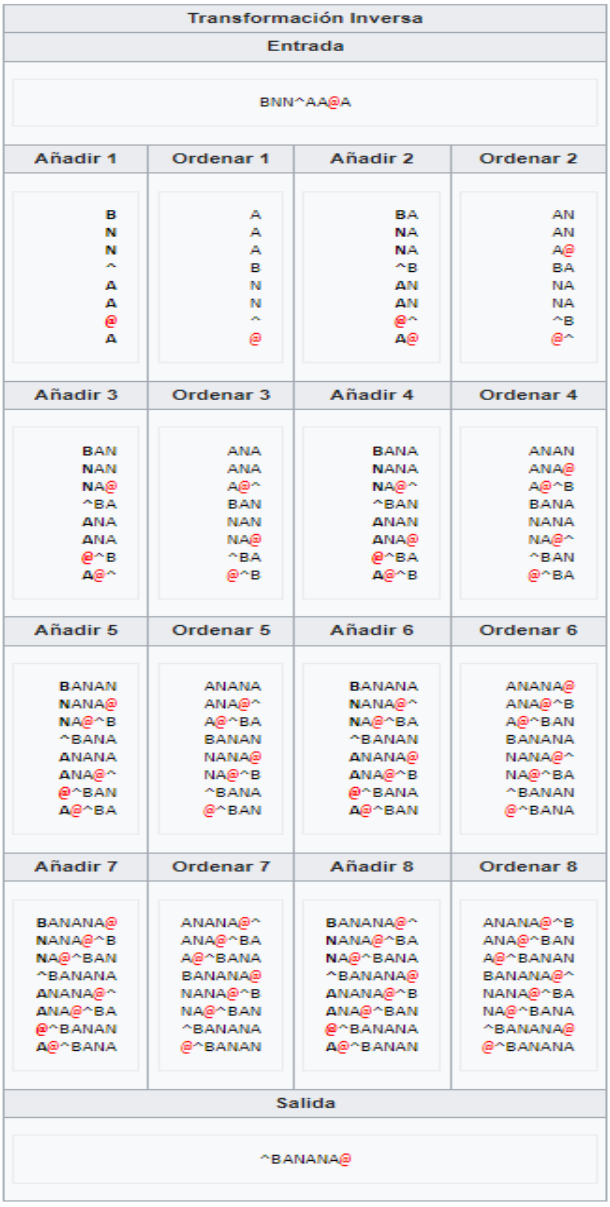
3.3 Alternativas de compresión de imágenes sin pérdida

En lo que sigue, presentamos diferentes algoritmos usados para comprimir imágenes sin pérdida. *(En este semestre, ejemplos de tales algoritmos son la transformada de Borrows y Wheeler, LZ77, LZ78, la codificación Huffman y LZS).*

3.3.1 Compresión de Borrows y Wheeler:

La compresión de Borrows-Wheeler se basa en la permutación del orden de caracteres para garantizar una mejor compresión de la cadena. Asignándoles con referencia a un valor e identificándolos para su reorganización.

Ejemplos:



Esta implementación en Python sacrifica velocidad frente a simplicidad: el programa es corto, pero tiene un coste temporal mayor que lineal, que el deseado para una implementación práctica.

Usando un carácter nulo como marca de final de fichero y usando `s[i:] + s[:i]` para construir la i-ésima rotación de `s`, la transformación final recoge el último carácter de cada una de las filas ordenadas.

```
def bwt(s):
    """Aplica la transformación de Burrows-Wheeler a la cadena de entrada."""
    assert "\0" not in s, "La cadena de entrada no puede contener el carácter nulo ('\0')"
    s += "\0"  # Se añade la marca de final de fichero
    table = sorted(s[i:] + s[:i] for i in range(len(s)))  # Tabla de rotaciones de la cadena
    last_column = [row[-1] for row in table]  # Últimos caracteres de cada fila
    return "".join(last_column)  # Convierte la lista de caracteres en una cadena
```

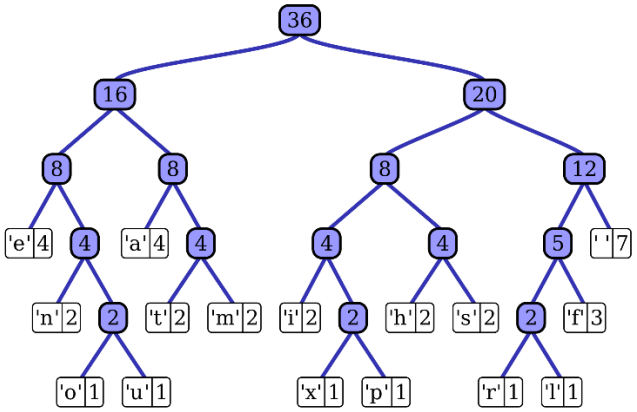
La transformada inversa inserta `r` repetidas veces como columna izquierda de la tabla y ordena la tabla. Tras completar la tabla entera, la función devuelve la fila que acaba en carácter nulo, salvo el carácter nulo.

```
def ibwt(r):
    """Aplica la transformación inversa de Burrows-Wheeler."""
    table = [""] * len(r)  # Crea una tabla vacía
    for i in range(len(r)):
        table = sorted(r[i] + table[i] for i in range(len(r)))  # Añade una columna de r
    s = [row for row in table if row.endswith("\0")][0]  # Encuentra la fila correcta (acabada en "\0")
    return s.rstrip("\0")  # Deshacerse del carácter nulo
```

3.3.2 Codificación de Huffman:

La codificación Huffman toma una cadena de texto y crea un nodo para cada carácter presente en dicha cadena junto con su frecuencia de aparición, es decir, el número de veces que aparece un mismo carácter, después de esto, los valores se ordenan mediante una cola de prioridad, esta tras determinar el orden de la cola de prioridad tomará los dos valores con mayor prioridad, estos valores se suman generando así un nuevo nodo "padre" cuyo valor será la suma generada anteriormente, este nodo volverá a la cola de prioridad y se repetirá el mismo procedimiento hasta acabar con la cola creando de esta forma un árbol de Huffman, cada vez que se baje por la izquierda de este árbol el valor será interpretado como un 0, mientras que cada vez que se baje por la derecha del árbol se interpretará como un 1, creando cadenas con una menor cantidad de bits, es decir, cadenas comprimidas.

Ejemplo:



3.3.3 Algoritmo LZ77:

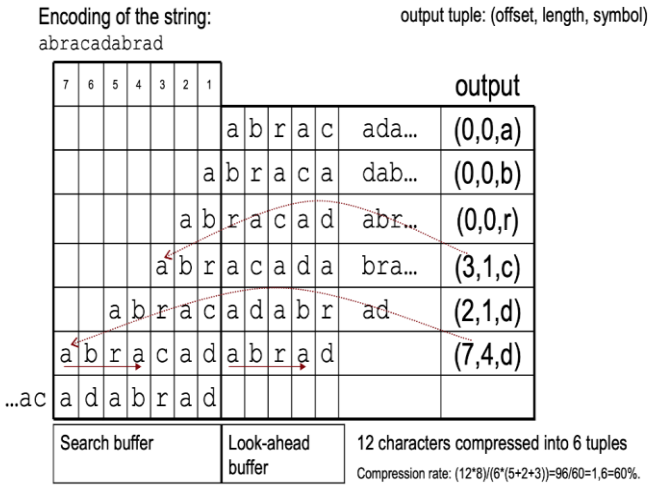
LZ77 o también llamado lz1, es un algoritmo de compresión sin perdida, el cual es utilizado cuando la información a comprimir es crítica y no puede perderse, como por ejemplo los archivos ejecutables.

En el algoritmo el codificador analiza el texto como una secuencia de caracteres, y utilizando una ventana deslizable conformada por dos partes, un buffer de anticipación que es la opción que está a punto de ser codificada y un buffer de búsqueda en donde se buscan secuencias iguales a las existentes en el buffer de anticipación. Para codificar una parte del contenido, del buffer de anticipación, se busca la secuencia igual en el buffer de búsqueda y la codificación resulta en indicar esta repetición como una tripleta [d, l, c].

Donde:

- d es la distancia desde el principio del buffer de anticipación hasta el comienzo de la secuencia repetida.
- l es la cantidad de caracteres repetidos.
- c es el símbolo siguiente a la secuencia en el buffer de anticipación.

Ejemplo:



Un ejemplo de pseudocódigo seria este:

```
while input is not empty do
    prefix := longest prefix of input that begins in window

    if prefix exists then
        d := distance to start of prefix
        l := length of prefix
        c := char following prefix in input
    else
        d := 0
        l := 0
        c := first char of input
    end if

    output (d, l, c)

    discard l + 1 chars from front of window
    s := pop l + 1 chars from front of input
    append s to back of window
repeat
```

3.3.4 Algoritmo LZ78:

El algoritmo LZ78 permite la creación de diccionarios o nuevas entradas dentro del mismo diccionario si el respectivo carácter de la cadena anterior no ha sido reconocida como existente. Las referencias que implementa el diccionario y la forma en que se aplican para generar un producto de un

índice, tomando como primer factor el primer índice dentro del diccionario, añadiéndolo a la cadena del output, y el cómo segundo aquel que se le esté agregando. Al final, se organizará de forma ordenada en cómo se ejecutaron dichas instrucciones para generar su respectivo output.

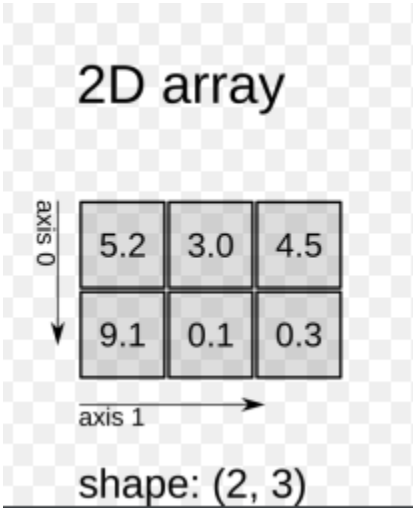
Example 1: LZ78 Compression Step 7

ABBCBCABABCAABCAAB **POS = 14**
C = empty
P = empty

STEP	POS	OUTPUT	DICTIONARY
1	1	(0,A)	A
2	2	(0,B)	B
3	3	(2,C)	BC
4	5	(3,A)	BCA
5	8	(2,A)	BA
6	10	(4,A)	BCAA
137	14	(6,B)	BCAAB

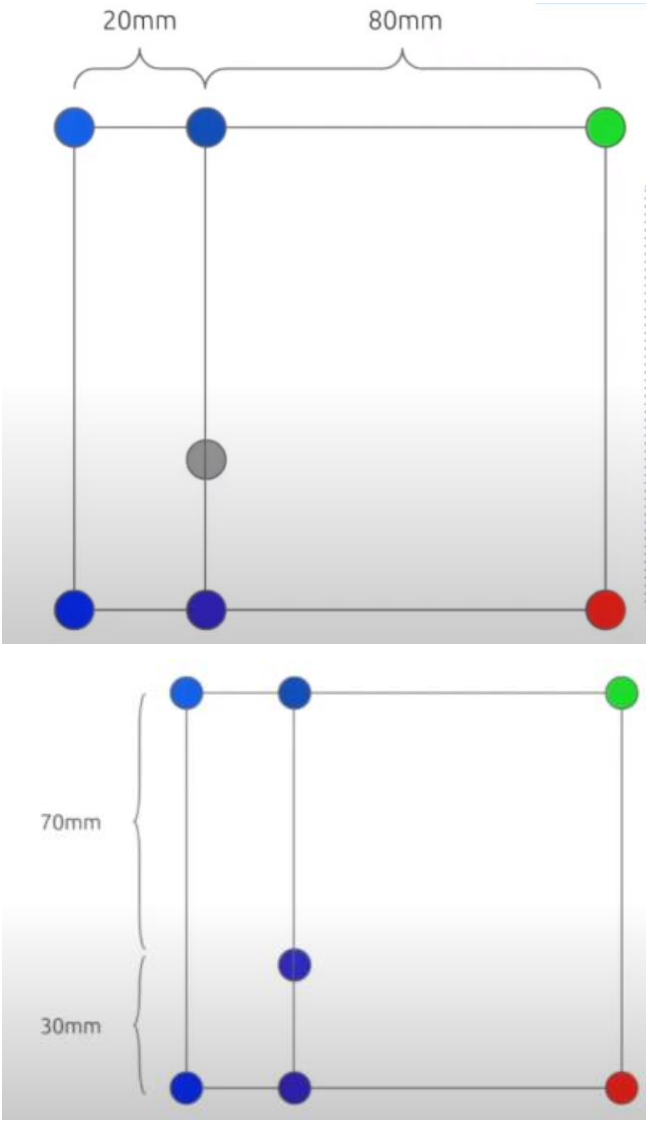
4.1 Estructuras de datos

Para realizar la compresión de imágenes con el algoritmo de escalado de imágenes mediante interpolación, optamos por utilizar matrices, ya que estas son una estructura de 2 dimensiones que posee elementos de un mismo tipo, lo cual nos permite implementar una interpolación bilineal fácilmente, pues este tipo de interpolación ocurre con 2 ejes, uno horizontal (eje x) y uno vertical (eje y), de esta manera el algoritmo puede trabajar con los números almacenados dentro de esta matriz para así realizar las aproximaciones necesarias para comprimir la imagen.



4.2.1 Algoritmo de compresión de imágenes con perdida

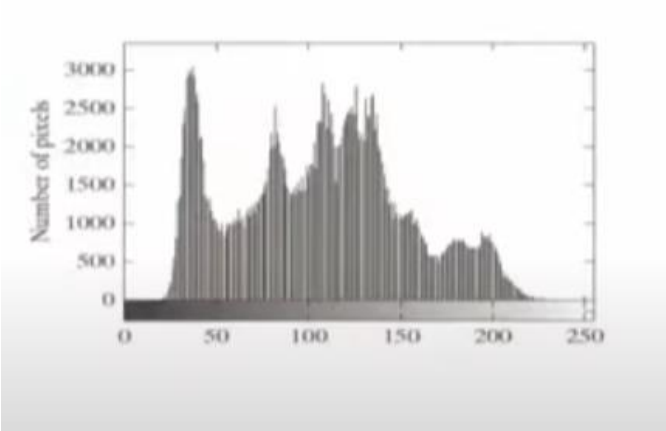
El algoritmo se implementó para que la imagen se adecuara al tamaño necesario, reuniendo los datos de ciertos puntos específicos dentro de la misma imagen para comprimirla, perdiendo así la posibilidad de restaurarla a su estado original. Esto se logra, ponderando primero el promedio horizontal y luego el vertical sobre el punto en concreto, en el cual se define un dato. Se cataloga bilineal por esta misma razón, debido a que se ejecuta dos veces el mismo proceso de ponderación en ambos ejes.



4.2.2 Algoritmo de compresión de imágenes sin pérdida

Un ejemplo de algoritmo de compresión de imágenes sin pérdidas en este caso puede ser la codificación Huffman, la codificación Huffman para el procesamiento (compresión) de imágenes generalmente funciona realizando un histograma que nos permite saber ciertos datos sobre la imagen, como por ejemplo la probabilidad o porcentaje de

aparición de un mismo pixel en dicha imagen. Una vez se tengan los pixeles de la imagen, la codificación Huffman se puede aplicar de manera “tradicional”, pues se puede comprimir cada cadena de bits de los pixeles ya vistos anteriormente, reduciendo de esta manera el número de bits que utiliza en memoria cada pixel.



Original source		Source reduction			
Symbol	Probability	1	2	3	4
a_2	0.4	0.4	0.4	0.4	0.6
a_6	0.3	0.3	0.3	0.3	0.4
a_1	0.1	0.1	0.2	0.3	
a_4	0.1	0.1	0.1		
a_3	0.06	0.1			
a_5	0.04				

4.3 Análisis de la complejidad de los algoritmos

El algoritmo de interpolación bilineal utiliza una matriz que se representa en filas y columnas, lo que para este caso significa una complejidad de O(N*M) en el peor caso, pues se tendría que recorrer toda la matriz que representa finalmente la imagen.

Algoritmo	La complejidad del tiempo
Interpolación bilineal	O(N*M)
Codificación Huffman	O(N)

Complejidad temporal de los algoritmos de compresión y descompresión de imágenes.

Por otra parte, la codificación de Huffman tiene una complejidad de O(N), pues para este caso se codifica toda la cadena de números que representan la imagen en un nodo único para cada cadena de números convirtiéndolos en un solo valor, creando de esta forma un "árbol" con la imagen ya codificada, lo que resulta en tener que recorrer todo el árbol para obtener la imagen

Algoritmo	Complejidad de la memoria
Interpolación bilineal	O(N*M)
Codificación Huffman	O(N)

Complejidad de memoria de los algoritmos de compresión y descompresión de imágenes.

4.4 Criterios de diseño del algoritmo

Los algoritmos fueron elegidos por las siguientes razones: La interpolación bilineal la elegimos sobre los demás tipos de interpolación como el vecino más cercano porque produce unas imágenes de "mayor calidad" tras ser comprimidas, pues las imágenes que produce suelen tener un "acabado" liso no tan pixelado como el de otros tipos de interpolación, todo esto para que el algoritmo de clasificación pueda obtener mejores resultados a largo plazo. Mientras que la codificación de Huffman la elegimos por su baja complejidad frente a otros algoritmos como pueden ser el "Run lenght" o recorrido de longitudes o el algoritmo LZ77.

5. RESULTADOS

5.1 Evaluación del modelo

En esta sección, presentamos algunas métricas para evaluar el modelo. La exactitud es la relación entre el número de predicciones correctas y el número total de muestras de entrada. La precisión es la proporción de estudiantes exitosos identificados correctamente por el modelo a estudiantes exitosos identificados por el modelo. Por último, sensibilidad es la proporción de estudiantes exitosos identificados correctamente por el modelo a estudiantes exitosos en el conjunto de datos.

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.

	Tiempo promedio de ejecución (s)	*Tamaño promedio del archivo (MB)*
Compresión	0.81 s	0.87 MB
Descompresión	0.18 s	1.28 MB

Tiempo de ejecución de los algoritmos interpolación bilineal y Huffman.

5.3 Consumo de memoria

Presentamos el consumo de memoria de los algoritmos de compresión y descompresión.

	Consumo promedio de memoria (MB)	*Tamaño promedio del archivo (MB)*
Compresión	14.45 MB	0.87 MB
Descompresión	11.55 MB	1.28 MB

Consumo promedio 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.

	Ganado sano	*Ganado enfermo*
Tasa de compresión promedio	2.4: 1	2,2: 1

Promedio redondeado de la tasa de compresión de todas las imágenes de ganado sano y ganado enfermo.

6. DISCUSIÓN DE LOS RESULTADOS

El algoritmo de compresión logra un porcentaje de compresión aproximado de un 60% en promedio, esto con un consumo de memoria bajo de unos 14,45 MB en promedio

Sin embargo, la tasa de compresión se llega a ver afectada si el tamaño de la imagen a comprimir es bastante grande, esto fue comprobado con algunas de las imágenes proporcionadas en los datasets.

6.1 **Trabajos futuros**

Se optaría por una reestructuración del código, de tal forma que mejore la eficiencia del mismo a través de una mayor compresión e implementación de recursos dentro del código. De esta forma, se podría incrementar su funcionalidad y capacidades actuales.

RECONOCIMIENTOS

Por último, nos gustaría agradecer a Andrés Ochoa Tirado por su colaboración en el informe sobre el funcionamiento del algoritmo de Seam Carving y demás ayudas a la hora de búsqueda de la información.

QREFERENCIAS

1. 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

2. Debauche, O., Mahmoudi, S., Andriamandroso, A.L.H. *et al.* Cloud services integration for farm animals' behavior studies based on smartphones as activity sensors. *J Ambient Intell Human Comput* 10, 4651–4662 (2019).

https://link.springer.com/article/10.1007/s12652-018-0845-9

3. Keelia Brannagh, Lempel-Ziv Compression Techniques presentation, 2014.

https://www.slideserve.com/keelia/lempel-ziv-compression-techniques

4. Rodrigo García, Jose Aguilar, Mauricio Toro, Angel Pinto, Paul Rodríguez. A systematic literature review on the use of machine learning in precision livestock farming, 2020, 1-10.

https://doi.org/10.1016/j.compag.2020.105826

5. Vasileios Doulgerakis, Dimitrios Kalyvas, Enkeleda Bocaj, Christos Giannousis, Michalis Feidakis, George P. Laliotis, Charalampos Patrikakis, Iosif Bizelis. An Animal Welfare Platform for Extensive Livestock Production Systems, Agricultural University of Athens, Department of Animal Breeding & Husbandry 75 Iera Odos, 11855, Athens, Greece, University of West Attica, Department of Electrical & Electronics Engineering 250 Thivon & P. Ralli, 12241, Athens, Greece, 2019, 1-7.

http://ceur-ws.org/Vol-2492/paper1.pdf

6. Wikipedia Compresión Fractal

https://es.wikipedia.org/wiki/Compresi%C3%B3n_fractal#V%C3%A9ase_tambi%C3%A9n

7. Wikipedia Huffman coding

https://en.wikipedia.org/wiki/Huffman_coding

8. Wikipedia Image scaling

https://en.wikipedia.org/wiki/Image_scaling

9. Wikipedia LZ77 and LZ78

https://en.wikipedia.org/wiki/LZ77_and_LZ78

10. Wikipedia Nearest-neighbor interpolation

https://en.wikipedia.org/wiki/Nearest-neighbor_interpolation

11. Wikipedia Seam carving

https://en.wikipedia.org/wiki/Seam_carving

12. Wikipedia Sistema interactivo de funciones

https://es.wikipedia.org/wiki/Sistema_iterativo_de_funciones

13. Wikipedia Transformada de coseno discreta

https://es.wikipedia.org/wiki/Transformada_de_coseno_discreta

14. Librerías y guías

https://www.youtube.com/watch?v=hwrfpQgc53c

15. Librerías

https://www.google.com/url?sa=i&url=https%3A%2F%2Fslideplayer.com%2Fslide%2F7587550%2F&psig=AOvVaw0CulEC9ZMoaCrV5Ddy0ZYz&ust=1633866715480000&source=images&cd=vfe&ved=0CAsQjRxqFwoTCJjFnaSivfMCFQAAAAAdAAAAABAD

16. Librerías

https://www.google.com/url?sa=i&url=https%3A%2F%2Faprendeconalf.es%2Fdocencia%2Fpython%2Fmanual%2Fnumpy%2F&psig=AOvVaw0nV6Csf3yd8d_3DlL8lE4F&ust=1633866949311000&source=images&cd=vfe&ved=0CAsQjRxqFwoTCICgxe-ivfMCFQAAAAAdAAAAABAS

17. Acerca del Power Point

https://fegasacruz.org/fao-el-ganado-consume-principalmente-alimentos-no-aptos-para-el-consumo-humano/ganado-campo-1/

https://www.contextoganadero.com/ganaderia-sostenible/que-hacer-cuando-se-presenta-asfixia-en-los-bovinos

https://ichi.pro/es/cambiar-el-tamano-de-las-imagenes-mediante-diversas-tecnicas-de-interpolacion-83122650192370

https://www.contextoganadero.com/ganaderia-sostenible/4-enfermedades-que-afectan-la-piel-de-los-bovinos

https://www.researchgate.net/figure/Example-of-bilinear-interpolation_fig5_341645510

https://es.wikipedia.org/wiki/Interpolaci%C3%B3n_(fotograf%C3%ADa)#:~:text=En%20el%20campo%20de%20la,partir%20de%20un%20algoritmo%20espec%C3%ADfico

https://youtu.be/R9mnjPgDCQk?t=163

https://www.zarebasystems.com/articles/the-most-profitable-livestock-to-raise

https://www.canr.msu.edu/news/three_common_summer_cattle_diseases

https://www.wur.nl/en/article/Healthy-livestock-farming.htm