OPTIMIZATION OF ENERGY CONSUMPTION IN PRECISION LIVESTOCK FARMING USING IMAGE COMPRESSION ALGORITHMS

Martín Ospina Uribe
Universidad Eafit
Colombia
mospinau1@eafit.edu.co

María José Bernal Vélez
Universidad Eafit
Colombia
mjbernalv@eafit.edu.co

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

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

ABSTRACT

Recently, precision livestock farming (PLF) has been developed as a way of improving farm operations by closely monitoring each animal using technology. However, it presents the challenge of how to make the system more efficient in terms of energy consumption. When monitoring animals, pictures and data are recorded 24 hours a day, which can consume a lot of time and memory, making the system slower every time. Due to this, it is important to design an algorithm for the compression of images in PLF, so that animals can be correctly monitored while optimizing energy consumption. There are certain related problems such as using smartphones in PLF and using smartphones and sound sensors in PLF, which will be described later in the report.

Keywords
Compression algorithms, machine learning, deep learning, precision livestock farming, animal health.

1. INTRODUCTION

Animal products are a fundamental element in most humans' diets; for instance, it is predicted that the worldwide demand for meat will grow 40% in the next 15 years [5]. Due to this, animal health is necessary to ensure that these products for human consumption are produced in the best conditions possible. However, it is very hard, tiring and expensive for farmers to monitor their animals 24 hours a day, so precision livestock farming has been recently developed as a way of monitoring animal health, welfare, environmental impact, production, and reproduction by using technology [6].

Nevertheless, as animals are being monitored all day long, a lot of pictures and data need to be stored, and this can take a lot of time and memory. For this reason, it is important to compress the images that are being taken so that these resources can be efficiently used and distributed, without losing quality and effectiveness of PLF.

1.1. Problem

When using precision livestock farming, animal health is closely monitored to ensure that first rate products are produced. Nonetheless, computers don't have unlimited disc space, which becomes a problem when huge quantities of information are stored. Due to this, compressing images in PLF is fundamental to be able to ensure that animals are correctly supervised at all times, using a system that does not take a lot of time. It is of great importance to solve this problem as farming plays a huge role in the primary sector of the economy, which affects people worldwide.

1.3 Article structure

In what follows, in Section 2, we present related work to the problem. Later, in Section 3, we present the data sets and methods used in this research. In Section 4, we present the algorithm design. After, in Section 5, we present the results. Finally, in Section 6, we discuss the results, and we propose some future work directions.

2. RELATED WORK

In what follows, we explain four related works on the domain of animal-health classification and image compression in the context of PLF.

2.1 Cloud Services Integration for Farm Animals' Behavior Studies Based on Smartphones as Activity Sensors

This work was based on using smartphones, particularly iPhone 4s and 5s, as sensors to study animal behavior in the context of PLF, due to the fact that it provides valuable information on their health, performance and reproductive status. The factors that were studied include their location obtained by GPS, and the low and high frequency components of behavior as posture of the animal. In order to compress the images and data, the compression algorithms they used were LZ4 by default or LZF. Furthermore, they concluded that the compressibility of data massively acquired can be reduced by 43.5% on average, while individual parameters can be highly compressible. Finally, they mention that other data compression algorithms must be considered to optimize energy consumption of the battery. [9]

2.2 An Animal Welfare Platform for Extensive Livestock Production Systems

This study was focused on presenting a solution for monitoring and tracking animal activity and behavior in livestock farms by using wireless sensors to record animal activity, edge computing devices with computational capabilities (offline and real time data processing), cloud computing and usable and effective visualizations in mobile devices. Additionally, to develop an automated system with a single wireless sensor, it uses Deep Neural Network pattern recognition algorithms, which have a low implementation cost. Their results include the necessity to minimize the amount of transmitted data on the wearable device in order to optimize battery lifetime. [12]

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

In this article, they use computer vision pipelines that use deep neural architectures to automate Holstein Friesian cattle detection and identification in agriculture. To pull off their project, they used fixed and mobile camera platforms, and later brought in video processing pipelines. Finally, they demonstrated that Friesian cattle detection and localization can be performed with a 99.3% accuracy rate, and by using the video processing pipeline they showed and accuracy of 98.1% with 23 individuals. [1]

2.4 Animal Sound… Talks! Real-time Sound Analysis for Health Monitoring in Livestock

This study uses sound based PLF techniques to monitor pigs' health, which can have many advantages over other types of techniques, such as the fact that they are contactless, relatively cheap, and only one sensor can be used for large groups of animals. They used a respiratory distress monitor in order to monitor the respiratory health of pigs. Hence, the implementation of analyzing algorithms was necessary to identify if a pig's cough was pathological or non-pathological. They concluded that this tool could work to give early warning (about 2 weeks before) compared to human observations. [7]

3. MATERIALS AND METHODS

In this section, we explain how the data was collected and processed and, after different image-compression algorithm alternatives to solve improve animal-health classification.

3.1 Data Collection and Processing

We collected data from Google Images and Bing Images divided into two groups: healthy cattle and sick cattle. For healthy cattle, the search string was “cow”. For sick cattle, the search string was “cow + sick”.

In the next step, both groups of images were transformed into grayscale using Python OpenCV and they were transformed into Comma Separated Values (CSV) files. It was found out that the datasets were balanced.

The dataset was divided into 70% for training and 30% for testing. Datasets are available at https://github.com/mauriciotoro/ST0245-Eafit/tree/master/proyecto/datasets.

Finally, using the training data set, we trained a convolutional neural network for binary image-classification using Google Teachable Machine available at https://teachablemachine.withgoogle.com/train/image.

3.2 Lossy Image-compression alternatives

In what follows, we present different algorithms used to compress images.

3.2.1 Seam carving

Seam carving, developed by Shai Avidan and Ariel Shamir, consists of establishing seams (paths of pixels) either horizontally or vertically to remove rows or columns of an image. This algorithm has only a local effect and shifts the remaining pixels left or up to compensate for the missing ones. To achieve this, each pixel on the image has a value for its energy, and dynamic programming is used to find seams through the following steps. Fist, each pixel in the first row/column keeps its energy as there are no rows/columns before. Then, from the second row/column to the last one its energy is computed with the cumulative minimum energy for all the possible connected seams of the row/column before. Finally, the minimum energy value found will indicate the seam that will be removed. [2]

In general, this is a rather simple algorithm that does not require a lot of work; however, when it is used with images with a lot of detail, this strategy will not work appropriately. [3]

Figure 1: Seam Carving process diagram

3.2.2 Image scaling

Image scaling, or image interpolation, resizes an image from one resolution to another one by using geometric transformations of each pixel without losing the content and quality of the image. There are different algorithms used to scale an image, such as Bilinear, Bicubic, Nearest-Neighbor, Bicubic Cubic B-spline and Lanczos. We will explain how Nearest-Neighbor and Bilinear work. [20]

Nearest-Neighbor: it is the simplest algorithm of all as it only considers the nearest pixel to fill the empty spaces and replicates it as the image grows. It is very efficient, as it requires the least computation and processing time, but the quality of the image created is very poor. [17]

Bilinear: it identifies the distance from the four nearby pixels in an image and takes the distance-weighted average of these four pixels to determine a new value, and a new pixel is estimated with the relative distance of the nearby pixels. This algorithm results in smoother images than Nearest-Neighbor but will also have a blurring effect. Finally, as it takes more pixels, it requires more processing time and produces better outputs. [17]

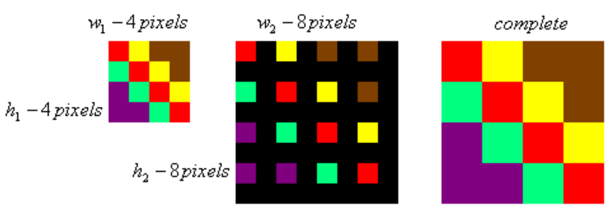

Figure 2: Image scaling using nearest-neighbor

3.2.3 Fractal compression

Fractal compression consists of representing an image using fractals, which is then represented by IFS (iterated function systems), a group of affine transformations. In order to achieve this, the image is split up into different segments that help to find similarities easier. Its main goal is to find the affine transformation for each fractal, and later look for the fractal that can be the best fit to the original image.

This method is very computationally expensive due to the fact that it looks for similarities in the picture. Because of this, it is not very practical to use in real time applications, as it takes more time and energy. [14]

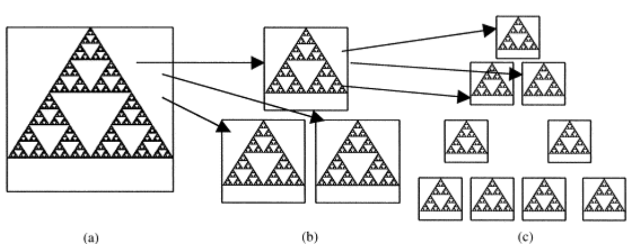

Figure 3: Fractal compression diagram

3.2.4 Discrete cosine transform

The discrete cosine transform (DCT) represents an image as the sum of cosine functions of different frequencies and magnitudes. This method has 3 main steps: quantification, coding and transmission, which work by decomposing an image to its spatial frequency spectrum as a finite sequence. First, the image is decoded into blocks of pixels; then, some of those blocks are discarded using the inverse discrete cosine transform of each one; and finally, the image is reconstructed by putting the blocks back together. [10]

This algorithm is known to have a good capacity of energy compaction, which means that most of the image is divided into a few blocks. Hence, it produces few mistakes in the image compression, creating a clear output. [11]

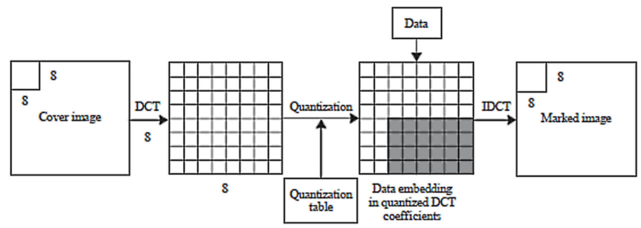

Figure 4: Discrete cosine transform vector figure

3.3 Lossless Image-compression alternatives

In what follows, we present different algorithms used to compress images.

3.3.1 Huffman coding

Huffman coding is based on finding the probabilities of data occurring in the sequence, so symbols that occur less frequently will need more bits than those with less frequencies. To do this, it starts by adding together the two pixels with the lowest probabilities, repeating this process until there is only one pixel left with a total probability of 1. Then, it goes back through the same path and codes each probability with binary code (0 and 1) until it is back at the beginning. [19]

Its time complexity is determined by O(nlogn), where n is the number of unique characters. [16]

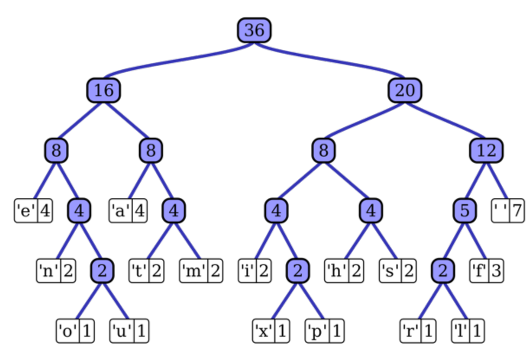

Figure 5: Huffman coding vector figure\

3.3.2 Burrows & Wheeler transform

The Burrows-Wheeler transform is based on block sorting lossless data compression algorithms. It consists of transforming a block of data (or pixels) into a form that allows easier compression. It follows the following steps. First, the original sequence is copied to the first row; then, it is sorted with all possible left-cycling permutations in the following rows; after that, rows are sorted lexicographically; and finally, the output is the last column. [18]

This algorithm is known to be very effective, due to the fact that it is reversible, and it doesn't need to store any additional data to achieve this. Its time complexity is determined by the expression O(n). [8]

	Transformation			
Input	All Rotations	Sorting All Rows into Lex Order	Taking Last Column	Output Last Column
^BANANA|	^BANANA| |^BANANA A|^BANAN NA|^BANA ANA|^BAN NANA|^BA ANANA|^B BANANA|^	ANANA|^B ANA|^BAN A|^BANAN BANANA|^ NANA|^BA NA|^BANA ^BANANA| |^BANANA	ANANA|^B ANA|^BAN A|^BANAN BANANA|^ NANA|^BA NA|^BANA ^BANANA| |^BANANA	BNN^AA|A

Figure 6: Burrows and Wheeler transform implementation

3.3.3 LZ77

The LZ77 Compression Algorithm is based on replacing redundant information with metadata (data that describes other data). To do this, sections that are identical to others that have already been encoded are replaced by metadata that includes information on how to expand those sections again. Firstly, the algorithm looks for the window with the longest match to the beginning of the lookahead buffer (the byte sequence from the actual coding position to the end of the input stream) and outputs a pointer to that match. If a match is not found, the algorithm outputs a null-pointer and the byte at the coding position. [15]

The time complexity of this algorithm is determined by the expression O(n), where n is the size of the input stream. Additionally, a huge advantage of this algorithm is that the backward process (decoding) is very simple and fast. Due to this, LZ77 is best for a file that is going to be encoded 1 time and decoded many times. [4]

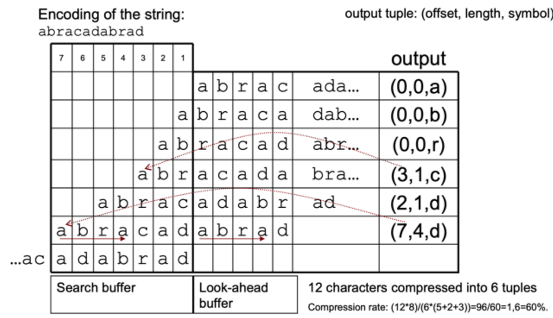

Figure 7: LZ77 example diagram

3.3.4 LZS

The LZS algorithm combines the LZ77 compression algorithm and Huffman coding. It looks for repeated data in the input data and replaces them with encoded tokens that are shorter. When a match between previous encoded data, it creates another encoded token that points to the match. Then, the encoded tokens replace redundant data into compressed streams. To do this, it maintains a compression history of 2 kilobytes of raw input data and other data structures that accelerate the process.

In addition, when there is more repetition of data, the compression ratio is higher, and vice versa. Thus, compression ratio is the quotient between the number of input bytes by the number of output bytes, so the less output bytes, the higher the compression ratio. [13]

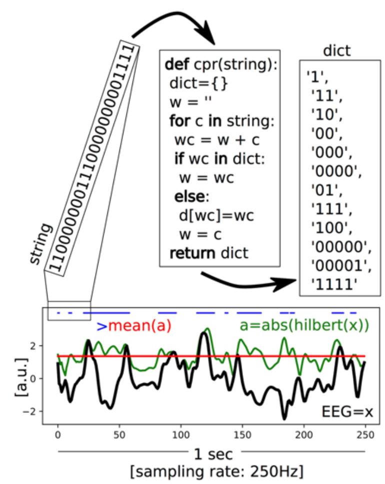

Figure 8: LZS vector figure

REFERENCES

1. Andrew, W., Greatwood, C., Burghardt, T., 2017. Visual localisation and individual identification of Holstein Friesian cattle via deep learning. In: 2017 IEEE International Conference on Computer Vision Workshops (ICCVW), pp. 2850–2859.

2. Anh, N., Cai, J., Yan, W., Seam carving extension: a compression perspective. in 17th International Conference on Multimedia (Vancouver, British Columbia, Canada, 2009).

3. Avidan, S., Shamir, A., 2007. Seam Carving for Content-Aware Image Resizing. SIGGRAPH '07: ACM SIGGRAPH 2007 papers.
4. Bell, T.C., Better OPM/L Text Compression. in IEEE Transactions on Communications (1986), IEEE, 1176-1177.
5. Berckmans, D., 2014. Precision livestock farming technologies for welfare management in intensive livestock systems. Scientific and Technical Review of the Office International des Epizooties, 33 (1), 189-196.
6. Berckmans, D., 2017. General introduction to precision livestock farming. Animal Frontiers, 7 (1), 6-11.
7. Berckmans, D., Hemeryck, M., Berckmans, D., Vranken, E., Waterschoot, T. Animal Sound… Talks! Real-time Sound Analysis for Health Monitoring in Livestock. Retrieved February 12, 2021, from Sound Talks: https://www.soundtalks.com/paper/animal-sound-talks-real-time-sound-analysis-for-health-monitoring-in-livestock/.
8. Burrows, M., Wheeler, D.J., 1994. A Block-sorting Lossless Data Compression Algorithm. Systems Research Center (SRC Research Report), 1-18.
9. Debauche, O., Mahmoudi, S., Andriamandroso, A.L.H., Manneback, P., Bindelle, J., Lebeau, F., 2019. Cloud services integration for farm animals' behavior studies based on smartphones as activity sensors. J. Ambient Intell. Humanized Comput. 10 (12), 4651–4662.
10. Discrete Cosine Transform, 1994-2021. Retrieved February 12, 2021, from Math Works Inc: https://www.mathworks.com/help/images/discrete-cosine-transform.html
11. Discrete Cosine Transform, 2020. Retrieved February 12, 2021, from Wikipedia: https://en.wikipedia.org/wiki/Discrete_cosine_transform
12. Doulgerakis, V., Kalyvas, D., Bocaj, E., Giannousis, C., Feidakis, M., Laliotis, G.P., Patrikakis, C., Bizelis, I., 2019. An animal welfare platform for extensive livestock production systems. In: CEUR Workshop Proceedings, vol. 2492.
13. Friend, R. Understanding LZS Compression in TLS security: A Tutorial, 2005. Retrieved February 12, 2021, from EE Times: https://www.eetimes.com/understanding-lzs-compression-in-tls-security-a-tutorial/#.
14. Jiang, J., Image Compression with Fractals. in IEE Colloquium on Fractals in Signal and Image Processing (London, UK, 1995), IET, 71-73.
15. LZ77 Compression Algorithm, 2020. Retrieved February 12, 2021, from Microsoft Docs: https://docs.microsoft.com/en-us/openspecs/windows_protocols/ms-wusp/fb98aa28-5cd7-407f-8869-a6cef1ff1ccb.
16. Morris, J. Huffman Encoding, 1998. Retrieved February 12, 2021, from: https://www.cs.auckland.ac.nz/software/AlgAnim/huffman.html.
17. Parsania, P.S, Virparia, P.V., 2018. Computational Time Complexity of Image Interpolation Algorithms. International Journal of Computer Sciences and Engineering, 6 (7), 491-496.
18. Van, V.S., 2009. Image Compression Using Burrows-Wheeler Transform. Helsinki University of Technology.
19. Vences Salcedo, L. 1994. Compresión fractal de imágenes fijas y secuencias de imágenes utilizando algoritmos genéticos. Facultad del Instituto Tecnológico y de Estudios Superiores de Monterrey, 5-6.
20. Wu, R., Yan, S., Shan, Y., Dang, Q., Sun, G., 2015. Deep Image: Scaling up Image Recognition. ArXiv.

FIGURES CITATIONS

1. Jane, 2020. Seam Carving. Retrieved February 12, 2021, from Colorado College: https://sites.coloradocollege.edu/blockfeatures/2020/11/08/seam-carving/.
2. Tech-Algorithm, 2007. Nearest Neighbor Image Scaling. Retrieved February 12, 2021, from Algorithm and Programming: http://tech-algorithm.com/articles/nearest-neighbor-image-scaling/.
3. Sun, K.T., Lee, S.J., Wu, P.Y., 2001. Neural network approaches to fractal image compression and decompression. Retrieved February 12, 2021, from Science Direct: https://www.sciencedirect.com/science/article/abs/pii/S0925231200003490.
4. McAteer, I., Ibrahim, A., Guanglou, Z., Valli, C., 2019. Integration of Biometrics and Steganography: A Comprehensive Review. Retrieved February 12, 2021, from Research Gate: https://www.researchgate.net/figure/Discrete-cosine-transform-DCT-based-data-hiding-using-the-JPEG-compression-model-A_fig2_333559334.
5. Wikipedia, the Free Encyclopedia. Retrieved February 12, 2021, from Huffman Coding: https://en.wikipedia.org/wiki/Huffman_coding.
6. Bhusal, S., 2019. Burrows Wheeler Transform. Retrieved February 12, 2021, from AlgoPods: https://medium.com/algopods/burrows-wheeler-transform-c743a2c23e0a.
7. Code Review, 2019. Retrieved February 12, 2021, from LZ77 compression (also longest string match): https://codereview.stackexchange.com/questions/233262/lz77-compression-also-longest-string-match.
8. Timmermann, C., Roseman, L., Schartner, M., Carhart-Harris, R.L., 2019. Neural correlates of the DMT

experience assessed with multivariate EEG. Retrieved February 12, 2021, from Research Gate: https://www.researchgate.net/figure/Schematic-of-the-LZs-computation-An-example-EEG-signal-with-a-sampling-rate-of-250-Hz_fig6_337342080.